Manual do usuário

AWS Secrets Manager

AWS Secrets Manager: Manual do usuário

Table of Contents

O que é AWS Secrets Manager?

AWS Secrets Manager ajuda você a gerenciar, recuperar e alternar credenciais de banco de dados, credenciais de aplicativos, tokens OAuth, chaves de API e outros segredos em todo o ciclo de vida. Muitos AWS serviços armazenam e usam segredos no Secrets Manager.

O Secrets Manager ajuda você a melhorar seu procedimento de segurança, porque você não precisa mais de credenciais de codificação rígida no código-fonte da aplicação. Armazenar as credenciais no Secrets Manager evita um possível comprometimento por qualquer pessoa que possa inspecionar sua aplicação ou os componentes. Você substitui credenciais de codificação rígida com uma chamada de runtime ao serviço Secrets para recuperar as credenciais dinamicamente quando necessário.

Com o Secrets Manager, você pode configurar uma programação de alternância automática para seus segredos. Isso permite que você substitua os segredos de longo prazo por outros de curto prazo, reduzindo significativamente o risco de comprometimento. Como as credenciais não são mais armazenadas na aplicação, a alternância das credenciais não exige mais a atualização das aplicações e a implantação de alterações nos clientes da aplicação.

Para outros tipos de segredos que você pode usar em sua organização:

- AWS credenciais — AWS Identity and Access ManagementRecomendamos.
- Chaves de criptografia: recomendamos o AWS Key Management Service.
- Chaves SSH: recomendamos o Amazon EC2 Instance Connect.
- Chaves privadas e certificados: recomendamos o AWS Certificate Manager.

Conceitos básicos do Secrets Manager

Se você é novo no Secrets Manager, comece com um dos seguintes tutoriais:

- the section called “Substituir segredos codificados ”
- the section called “Substituir credenciais de banco de dados codificadas ”
- the section called “Alternância de usuários alternados”
- the section called “Alternâncias de usuário único”

Outras tarefas que você pode realizar com segredos:

- Criar e gerenciar segredos
- Controlar o acesso a seus segredos
- Obtenha segredos
- Alternar segredos
- Monitorar segredos
- Monitore segredos para verificar a conformidade
- Crie segredos em AWS CloudFormation

Compatibilidade com padrões

AWS Secrets Manager foi submetido à auditoria de vários padrões e pode fazer parte de sua solução quando você precisar obter a certificação de conformidade. Para ter mais informações, consulte Validação de conformidade.

Definição de preço

Ao usar o Secrets Manager, você paga somente pelo que usa, e não há taxas mínimas ou de configuração. Não há cobrança para segredos que são marcados para exclusão. Para obter a lista de preços atual completa, consulte Definição de preço do AWS Secrets Manager. Para monitorar seus custos, consultethe section called “Monitore os custos do Secrets Manager”.

Você pode usar o Chave gerenciada pela AWS `aws/secretsmanager` que o Secrets Manager cria para criptografar seus segredos gratuitamente. Se você criar suas próprias chaves KMS para criptografar seus segredos, AWS cobrará a taxa atual AWS KMS . Para obter mais informações, consulte Preços do AWS Key Management Service.

Quando você ativa a rotação automática (exceto a rotação gerenciada), o Secrets Manager usa uma AWS Lambda função para girar o segredo, e você é cobrado pela função de rotação na taxa Lambda atual. Para obter mais informações, consulte Preços do AWS Lambda.

Se você ativar AWS CloudTrail em sua conta, poderá obter registros das chamadas de API que o Secrets Manager envia. O Secrets Manager registra todos os eventos como eventos de gerenciamento. AWS CloudTrail armazena gratuitamente a primeira cópia de todos os eventos de gerenciamento. No entanto, se você habilitar as notificações, poderão ser cobradas taxas pelo armazenamento de logs do Amazon S3 e pelo uso do Amazon SNS. Além disso, se você configurar

trilhas adicionais, as cópias adicionais de eventos de gerenciamento poderão ter custos. Para obter mais informações, consulte Preços do AWS CloudTrail.

Acesso AWS Secrets Manager

Você pode trabalhar com o Secrets Manager de qualquer uma das seguintes formas:

- Console do Secrets Manager
- Ferramentas da linha de comando
- AWS SDKs
- API de consulta HTTPS
- AWS Secrets Manager endpoints

Console do Secrets Manager

É possível gerenciar os seus segredos usando o console do Secrets Manager baseado no navegador e executar praticamente qualquer tarefa relacionada aos seus segredos pelo console.

Ferramentas da linha de comando

As ferramentas de linha de AWS comando permitem que você emita comandos na linha de comando do sistema para executar o Secrets Manager e outras AWS tarefas. Isso pode ser mais rápido e mais conveniente do que usar o console. As ferramentas de linha de comando podem ser úteis se você quiser criar scripts para realizar AWS tarefas.

Quando você insere comandos em um shell de comando, existe o risco de o histórico de comandos ser acessado ou de utilitários terem acesso aos seus parâmetros de comando. Consulte the section called “Mitigação de riscos do uso da AWS CLI para armazenar segredos do AWS Secrets Manager”.

As ferramentas da linha de comando usam automaticamente o endpoint padrão para o serviço em uma AWS região. Você pode especificar um endpoint diferente para suas solicitações de API. Consulte the section called “Endpoint do Secrets Manager”.

AWS fornece dois conjuntos de ferramentas de linha de comando:

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

AWS SDKs

Os AWS SDKs consistem em bibliotecas e exemplos de código para várias linguagens e plataformas de programação. Os SDKs incluem tarefas como assinatura criptográfica de solicitações, gerenciamento de erros e novas tentativas automáticas de solicitações. Para baixar e instalar qualquer um dos SDKs, consulte Ferramentas da Amazon Web Services.

Os AWS SDKs usam automaticamente o endpoint padrão para o serviço em uma AWS região. Você pode especificar um endpoint diferente para suas solicitações de API. Consulte the section called "Endpoint do Secrets Manager".

Para obter a documentação do SDK, consulte:

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

API de consulta HTTPS

A API de consulta HTTPS fornece acesso programático ao Secrets Manager e. AWS A API de consulta HTTPS permite executar solicitações HTTPS diretamente ao serviço.

Embora você possa fazer chamadas diretas para a API de consulta HTTPS do Secrets Manager, recomendamos usar um dos SDKs. O SDK executa muitas tarefas úteis que você precisaria executar manualmente. Por exemplo, os SDKs assinam automaticamente as suas solicitações e convertem respostas em uma estrutura sintaticamente adequada para a sua linguagem.

Para fazer chamadas HTTPS para o Secrets Manager, você se conecta a ???.

AWS Secrets Manager endpoints

Para estabelecer conexão programaticamente com o Secrets Manager, você usa um endpoint, o URL do ponto de entrada do serviço. Os endpoints do Secrets Manager são endpoints de pilha dupla, o que significa que são compatíveis com IPv4 e IPv6.

Em algumas regiões, o Secrets Information oferece endpoints compatíveis com Federal Information Processing Standard (FIPS) 140-2.

O Secrets Manager é compatível com o TLS 1.2 e 1.3. O Secrets Manager é compatível com o PQTLS em todas as regiões, exceto nas da China.

> **Note**
>
> O AWS SDK do Python e a AWS CLI tentativa de chamar o IPv6 e depois o IPv4 em sequência. Portanto, se você não tiver o IPv6 ativado, pode levar algum tempo até que a chamada expire e tente novamente com o IPv4. Para contornar esse problema, você pode desabilitar completamente o IPv6 ou migrar para o IPv6.

Veja a seguir os endpoints de serviço do Secrets Manager. Observe que a nomenclatura difere da convenção de nomenclatura típica de pilha dupla.

Nome da região	Região	Endpoint	Protocolo	
Leste dos EUA (Ohio)	us-east-2	secretsmanager.us-east-2.amazonaws.com secretsmanager-fips.us-east-2.amazonaws.com	HTTPS HTTPS	
Leste dos EUA (Norte da Virgínia)	us-east-1	secretsmanager.us-east-1.amazonaws.com secretsmanager-fips.us-east-1.amazonaws.com	HTTPS HTTPS	

Nome da região	Região	Endpoint	Protocolo	
Oeste dos EUA (N. da Califórnia)	us-west-1	secretsmanager.us-west-1.amazonaws.com secretsmanager-fips.us-west-1.amazonaws.com	HTTPS HTTPS	
Oeste dos EUA (Oregon)	us-west-2	secretsmanager.us-west-2.amazonaws.com secretsmanager-fips.us-west-2.amazonaws.com	HTTPS HTTPS	
África (Cidade do Cabo)	af-south-1	secretsmanager.af-south-1.amazonaws.com	HTTPS	
Ásia-Pacífico (Hong Kong)	ap-east-1	secretsmanager.ap-east-1.amazonaws.com	HTTPS	
Ásia-Pacífico (Hyderabad)	ap-south-2	secretsmanager.ap-south-2.amazonaws.com	HTTPS	
Ásia-Pacífico (Jacarta)	ap-southeast-3	secretsmanager.ap-southeast-3.amazonaws.com	HTTPS	
Ásia-Pacífico (Melbourne)	ap-southeast-4	secretsmanager.ap-southeast-4.amazonaws.com	HTTPS	

Nome da região	Região	Endpoint	Protocolo
Ásia-Pacífico (Mumbai)	ap-south-1	secretsmanager.ap-south-1.amazonaws.com	HTTPS
Ásia-Pacífico (Osaka)	ap-northeast-3	secretsmanager.ap-northeast-3.amazonaws.com	HTTPS
Ásia-Pacífico (Seul)	ap-northeast-2	secretsmanager.ap-northeast-2.amazonaws.com	HTTPS
Ásia-Pacífico (Singapura)	ap-southeast-1	secretsmanager.ap-southeast-1.amazonaws.com	HTTPS
Ásia-Pacífico (Sydney)	ap-southeast-2	secretsmanager.ap-southeast-2.amazonaws.com	HTTPS
Ásia-Pacífico (Tóquio)	ap-northeast-1	secretsmanager.ap-northeast-1.amazonaws.com	HTTPS
Canadá (Central)	ca-central-1	secretsmanager.ca-central-1.amazonaws.com secretsmanager-fips.ca-central-1.amazonaws.com	HTTPS HTTPS
Oeste do Canadá (Calgary)	ca-west-1	secretsmanager.ca-west-1.amazonaws.com secretsmanager-fips.ca-west-1.amazonaws.com	HTTPS HTTPS

Nome da região	Região	Endpoint	Protocolo	
Europa (Frankfurt)	eu-central-1	secretsmanager.eu-central-1.amazonaws.com	HTTPS	
Europa (Irlanda)	eu-west-1	secretsmanager.eu-west-1.amazonaws.com	HTTPS	
Europa (Londres)	eu-west-2	secretsmanager.eu-west-2.amazonaws.com	HTTPS	
Europa (Milão)	eu-south-1	secretsmanager.eu-south-1.amazonaws.com	HTTPS	
Europa (Paris)	eu-west-3	secretsmanager.eu-west-3.amazonaws.com	HTTPS	
Europa (Espanha)	eu-south-2	secretsmanager.eu-south-2.amazonaws.com	HTTPS	
Europa (Estocolmo)	eu-north-1	secretsmanager.eu-north-1.amazonaws.com	HTTPS	
Europa (Zurique)	eu-central-2	secretsmanager.eu-central-2.amazonaws.com	HTTPS	
Israel (Tel Aviv)	il-central-1	secretsmanager.il-central-1.amazonaws.com	HTTPS	
Oriente Médio (Barém)	me-south-1	secretsmanager.me-south-1.amazonaws.com	HTTPS	

Nome da região	Região	Endpoint	Protocolo	
Oriente Médio (Emirados Árabes Unidos)	me-central-1	secretsmanager.me-central-1.amazonaws.com	HTTPS	
América do Sul (São Paulo)	sa-east-1	secretsmanager.sa-east-1.amazonaws.com	HTTPS	
AWS GovCloud (Leste dos EUA)	us-gov-east-1	secretsmanager.us-gov-east-1.amazonaws.com secretsmanager-fips.us-gov-east-1.amazonaws.com	HTTPS HTTPS	
AWS GovCloud (Oeste dos EUA)	us-gov-west-1	secretsmanager.us-gov-west-1.amazonaws.com secretsmanager-fips.us-gov-west-1.amazonaws.com	HTTPS HTTPS	

O que há em um segredo do Secrets Manager?

No Secrets Manager, um segredo são informações de segredos, isto é, o valor do segredo mais alguns metadados sobre o segredo. Um valor do segredo pode ser uma string ou um binário.

Para armazenar vários valores de string em um segredo, recomendamos que você use uma string de texto JSON com pares de valores-chave, por exemplo:

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

Para segredos do banco de dados, se você quiser ativar a rotação automática, o segredo deverá conter informações de conexão do banco de dados na estrutura JSON correta. Para ter mais informações, consulte the section called “Estrutura JSON de um segredo”.

Metadados

Os metadados de um segredo incluem:

- Um nome do recurso da Amazon (ARN) com o seguinte formato:

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

  O Secrets Manager inclui seis caracteres aleatórios ao final do nome do segredo para ajudar a garantir que o ARN do segredo seja único. Se o segredo original for excluído e, então, for criado um novo com o mesmo nome, os dois segredos terão ARNs diferentes, em função desses caracteres. Os usuários com acesso ao segredo antigo não terão acesso automático ao novo segredo, uma vez que os ARNs são diferentes.
- O nome do segredo, uma descrição, uma política de recursos e etiquetas.
- O ARN de uma chave de criptografia, e AWS KMS key que o Secrets Manager usa para criptografar e descriptografar o valor secreto. O Secrets Manager sempre armazena o texto do

segredo de forma criptografada e criptografa o segredo em trânsito. Consulte the section called “Criptografia e descriptografia de segredos”.

- Informações sobre como alternar o segredo, se você configurar a alternância. Consulte Alternar segredos .

O Secrets Manager usa políticas de permissões do IAM para garantir que somente usuários autorizados possam acessar ou modificar um segredo. Consulte Autenticação e controle de acesso para AWS Secrets Manager.

Um segredo tem versões que contêm cópias do valor secreto criptografado. Quando você altera o valor do segredo, ou o segredo é alternado, o Secrets Manager cria uma nova versão. Consulte the section called “Versões secretas”.

Você pode usar um segredo em vários Regiões da AWS replicando-o. Quando você replica um segredo, você cria uma cópia do segredo primário ou original, chamado de segredo de réplica. O segredo de réplica permanece vinculado ao segredo primário. Consulte Replique segredos em todas as regiões.

Consulte Criar e gerenciar segredos.

Versões secretas

Um segredo tem versões que contêm cópias do valor secreto criptografado. Quando você altera o valor do segredo, ou o segredo é alternado, o Secrets Manager cria uma nova versão.

O Secrets Manager não armazena um histórico linear de segredos com versões. Em vez disso, ele acompanha três versões específicas rotulando-as:

- A versão atual - `AWSCURRENT`
- A versão anterior - `AWSPREVIOUS`
- A versão pendente (durante a alternância): `AWSPENDING`

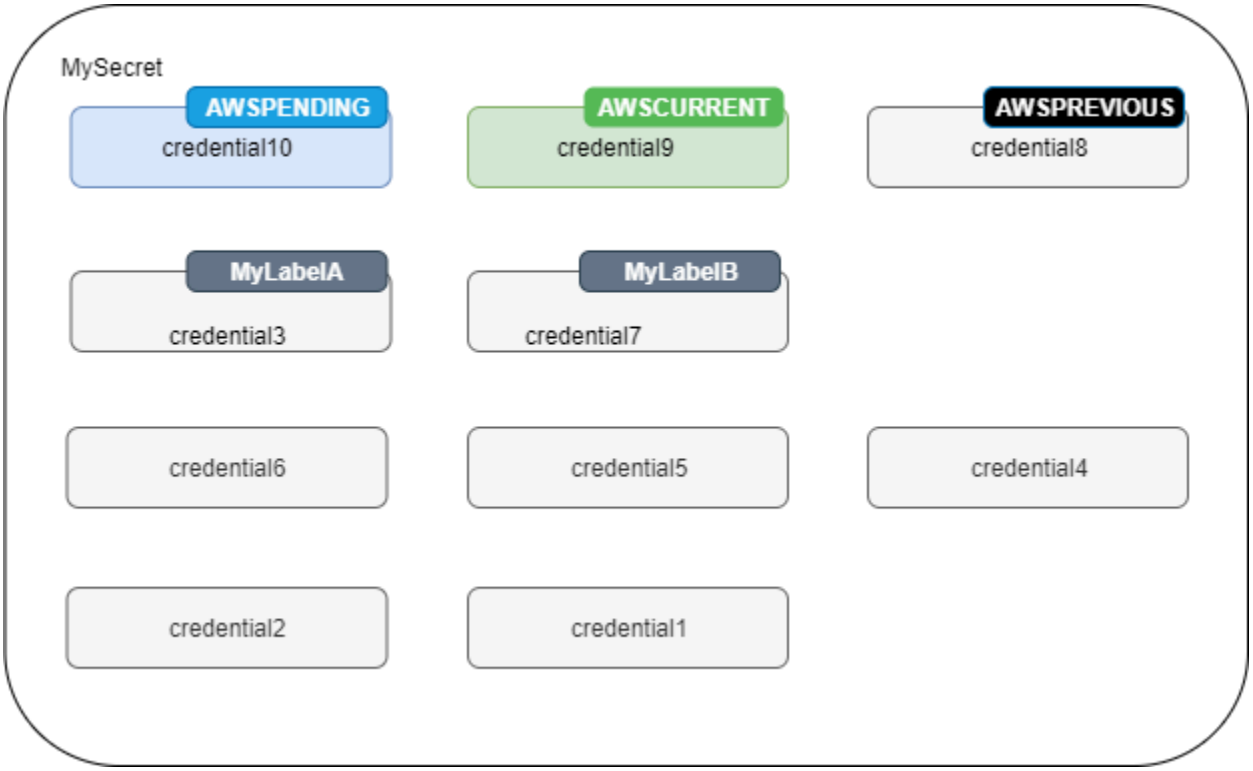
Um segredo sempre tem uma versão rotulada `AWSCURRENT`, e o Secrets Manager retorna essa versão por padrão quando você recupera o valor secreto.

Você também pode rotular versões com suas próprias etiquetas `update-secret-version-stage`ligando para AWS CLI o. É possível anexar até 20 rótulos a versões em um segredo. Duas

versões do segredo não podem ter o mesmo rótulo de preparação. As versões podem ter vários rótulos.

O Secrets Manager nunca remove as versões rotuladas, mas as versões sem rótulos são consideradas obsoletas. O Secrets Manager remove versões obsoletas quando há mais de 100. O Secrets Manager não remove versões criadas há menos de 24 horas.

A figura a seguir mostra um segredo que tem versões AWS rotuladas e versões rotuladas pelo cliente. As versões sem rótulos são consideradas obsoletas e serão removidas pelo Secrets Manager em algum momento no future.



Tutoriais do AWS Secrets Manager

Tópicos

Encontre segredos desprotegidos no código com o Amazon CodeGuru Reviewer

O Amazon CodeGuru Reviewer é um serviço que usa análise de programas e machine learning para detectar possíveis defeitos que os desenvolvedores têm dificuldade em encontrar e oferece sugestões para melhorar seu código Java e Python. O CodeGuru Reviewer se integra ao Secrets Manager para encontrar segredos desprotegidos no código. Para saber os tipos de segredos que o serviço pode encontrar, consulte Types of secrets detected by CodeGuru Reviewer (Tipos de segredos detectados pelo CodeGuru Reviewer) no Guia do usuário do Amazon CodeGuru Reviewer.

Depois de encontrar segredos codificados, realize ações para substituí-los:

- the section called “Substituir credenciais de banco de dados codificadas ”
- the section called “Substituir segredos codificados ”

Mova segredos codificados para AWS Secrets Manager

Caso você tenha segredos de texto simples no código, recomendamos alterná-los e armazená-los no Secrets Manager. Transferir o segredo para o Secrets Manager resolverá o problema de deixar o segredo visível para qualquer pessoa que visualizar o código, porque, a partir de então, seu código recuperará o segredo diretamente do Secrets Manager. Alternar o segredo revogará o segredo codificado atual para que não seja mais válido.

Para obter segredos de credenciais de banco de dados, consulte Mova as credenciais codificadas do banco de dados para AWS Secrets Manager.

Antes de começar, é necessário determinar quem precisa acessar o segredo. Recomendamos usar dois perfis do IAM para gerenciar a permissão para seu segredo:

- Um perfil que gerencia os segredos da organização. Para ter mais informações, consulte the section called “Permissões de administrador do Secrets Manager”. Você criará e alternará o segredo usando esse perfil.
- Uma função que pode usar o segredo em tempo de execução, por exemplo, neste tutorial que você usa *RoleToRetrieveSecretAtRuntime*. Seu código assume esse perfil para recuperar o segredo. Neste tutorial, você concede ao perfil apenas a permissão para recuperar um valor de segredo e concede permissão usando a política de recursos do segredo. Para conhecer as alternativas, consulte the section called “Próximas etapas”.

Etapas:

Etapa 1: criar o segredo

A primeira etapa é copiar o segredo codificado existente para o Secrets Manager. Se o segredo estiver relacionado a um AWS recurso, armazene-o na mesma região do recurso. Caso contrário, armazene-o na região que tem a menor latência para seu caso de uso.

Para criar um segredo (console)

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Selecione Armazenar um novo segredo.
3. Na página Choose secret type (Selecionar tipo de segredo), faça o seguinte:
    a. Em Secret type (Tipo de segredo), escolha Other type of secret (Outro tipo de segredo).
    b. Digite seu segredo como pares de chave-valor ou em texto não criptografado. Alguns exemplos:

Os pares de chave-valor da API:

ClientID: ***my_client_id***

ClientSecret : ***WJALRXUTNFEMI/K7MDENG/CYEXAMPLEKEY bPxRfi***

Credenciais de pares de chave-valor:

Username: *saanvis*

Password: *EXAMPLE-PASSWORD*

Texto não criptografado do token OAuth:

AKIAI44QH8DHBEXAMPLE

Texto sem formatação do certificado digital:

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

Chave privada em texto não criptografado:

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

c. Em Encryption key (Chave de criptografia), escolha aws/secretsmanager para usar a Chave gerenciada pela AWS para Secrets Manager. Não há custo para o uso dessa chave. Você também pode usar sua própria chave gerenciada pelo cliente, por exemplo, para acessar o segredo de outra Conta da AWS. Para obter mais informações sobre os custos do uso de uma chave gerenciada pelo cliente, consulte Definição de preço.

d. Escolha Próximo.

4. Na página Choose secret type (Selecionar tipo de segredo), faça o seguinte:

   a. Insira um Secret name (Nome de segredo) descritivo e uma Description (Descrição).

b. Em Resource permissions (Permissões do recurso), escolha Edit permissions (Editar permissões). Cole a política a seguir, que *RoleToRetrieveSecretAtRuntime*permite recuperar o segredo, e escolha Salvar.

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

c. Na parte inferior da página, selecione a opção Próximo.

5. Na página Configure rotation (Configurar alternância), mantenha a alternância ligada. Escolha Próximo.
6. Na página Review (Revisar), revise os detalhes do segredo e escolha Store (Armazenar).

Etapa 2: atualizar o código

Seu código deve assumir a função *RoleToRetrieveSecretAtRuntime*do IAM para poder recuperar o segredo. Para obter mais informações, consulte Mudar para uma função do IAM (AWS API).

Em seguida, atualize o código para recuperar o segredo do Secrets Manager usando o código de exemplo fornecido pelo Secrets Manager.

Como encontrar o código de exemplo

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na página Secrets (Segredos), escolha o segredo.
3. Role para baixo até Sample code (Código de exemplo). Escolha a linguagem de programação e copie o trecho de código.

Em sua aplicação, remova o segredo codificado e cole o trecho de código. Conforme o idioma do código, talvez seja necessário adicionar uma chamada ao perfil ou método no trecho de código.

Teste se sua aplicação funciona conforme o esperado com o segredo no lugar do segredo codificado.

Etapa 3: atualizar o segredo

A última etapa é revogar e atualizar o segredo codificado. Consulte a origem do segredo para encontrar instruções para revogá-lo e atualizá-lo. Por exemplo, talvez seja necessário desativar o segredo atual e gerar um novo segredo.

Para atualizar o segredo com o novo valor

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Escolha Secrets (Segredos) e escolha o segredo.
3. Na página Secret details (Detalhes do segredo), role para baixo e escolha Retrieve secret value (Recuperar valor do segredo) e depois Edit (Editar).
4. Atualize o segredo e escolha Save (Salvar).

Em seguida, teste se a aplicação funciona conforme o esperado com o novo segredo.

Próximas etapas

Depois de remover um segredo codificado de seu código, há algumas ideias a serem consideradas em seguida:

- Para encontrar segredos codificados em seus aplicativos Java e Python, recomendamos o Amazon Reviewer. CodeGuru
- É possível melhorar a perfomance e reduzir custos armazenando segredos em cache. Para ter mais informações, consulte Obtenha segredos.
- Para segredos acessados de várias regiões, você pode replicar seu segredo para melhorar a latência. Para ter mais informações, consulte Replique segredos em todas as regiões.
- Neste tutorial, você concedeu *RoleToRetrieveSecretAtRuntime*somente a permissão para recuperar o valor secreto. Para conceder mais permissões ao perfil, por exemplo, para obter metadados sobre o segredo ou para visualizar uma lista de segredos, consulte the section called “Exemplos de política de permissões”.

- Neste tutorial, você concedeu permissão *RoleToRetrieveSecretAtRuntime*usando a política de recursos do segredo. Para saber outras formas de conceder permissão, consulte the section called “Anexar uma política de permissões a uma identidade”.

Mova as credenciais codificadas do banco de dados para AWS Secrets Manager

Caso você tenha credenciais de banco de dados em texto simples no código, recomendamos mover as credencias para o Secrets Manager e alterná-las imediatamente. Mover as credenciais para o Secrets Manager resolverá o problema de deixar as credenciais visíveis para qualquer pessoa que visualizar o código, porque, a partir de então, seu código recuperará as credenciais diretamente do Secrets Manager. Alternar o segredo atualizará a senha e revogará a senha codificada atual para que não seja mais válida.

Para bancos de dados do Amazon RDS, Amazon Redshift e Amazon DocumentDB, use as etapas desta página para mover credenciais codificadas para o Secrets Manager. Para outros tipos de credenciais e outros segredos, consulte the section called “Substituir segredos codificados ”.

Antes de começar, é necessário determinar quem precisa acessar o segredo. Recomendamos usar dois perfis do IAM para gerenciar a permissão para seu segredo:

- Um perfil que gerencia os segredos da organização. Para ter mais informações, consulte the section called “Permissões de administrador do Secrets Manager”. Você criará e alternará o segredo usando esse perfil.
- Uma função que pode usar as credenciais em tempo de execução, *RoleToRetrieveSecretAtRuntime*neste tutorial. Seu código assume esse perfil para recuperar o segredo.

Etapas:

- Etapa 1: criar o segredo
- Etapa 2: atualizar o código
- Etapa 3: alternar o segredo
- Próximas etapas

Etapa 1: criar o segredo

A primeira etapa é copiar as credenciais codificadas existentes para o Secrets Manager. Para obter a latência mais baixa, armazene o segredo na mesma região do banco de dados.

Para criar um segredo

1. Abra o console Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Selecione Armazenar um novo segredo.
3. Na página Choose secret type (Selecionar tipo de segredo), faça o seguinte:
   a. Em Secret type (Tipo de segredo), escolha o tipo de credenciais de banco de dados a armazenar:
      - Bancos de dados do Amazon RDS
      - Amazon DocumentDB database (Bancos de dados do Amazon DocumentDB)
      - Armazém de dados do Amazon Redshift.
      - Para outros tipos de segredos, consulte Replace hardcoded secrets (Substituir segredos codificados).
   b. Em Credenciais, insira as credenciais codificadas existentes para o banco de dados.
   c. Em Encryption key (Chave de criptografia), escolha aws/secretsmanager para usar a Chave gerenciada pela AWS para Secrets Manager. Não há custo para o uso dessa chave. Você também pode usar sua própria chave gerenciada pelo cliente, por exemplo, para acessar o segredo de outra Conta da AWS. Para obter mais informações sobre os custos do uso de uma chave gerenciada pelo cliente, consulte Definição de preço.
   d. Em Database (Banco de dados), escolha seu banco de dados.
   e. Escolha Próximo.
4. Na página Configure secret (Configurar segredo), faça o seguinte:
   a. Insira um Secret name (Nome de segredo) descritivo e uma Description (Descrição).
   b. Em Resource permissions (Permissões do recurso), escolha Edit permissions (Editar permissões). Cole a política a seguir, que *RoleToRetrieveSecretAtRuntime*permite recuperar o segredo, e escolha Salvar.

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

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

c. Na parte inferior da página, selecione a opção Próximo.

5. Na página Configure rotation (Configurar alternância), mantenha a alternância desligada por enquanto. Você a ativará depois. Escolha Próximo.
6. Na página Review (Revisar), revise os detalhes do segredo e escolha Store (Armazenar).

Etapa 2: atualizar o código

Seu código deve assumir a função *RoleToRetrieveSecretAtRuntime*do IAM para poder recuperar o segredo. Para obter mais informações, consulte Mudar para uma função do IAM (AWS API).

Em seguida, atualize o código para recuperar o segredo do Secrets Manager usando o código de exemplo fornecido pelo Secrets Manager.

Como encontrar o código de exemplo

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na página Secrets (Segredos), escolha o segredo.
3. Role para baixo até Sample code (Código de exemplo). Escolha a linguagem e copie o trecho de código.

Em sua aplicação, remova as credenciais codificadas e cole o trecho de código. Conforme o idioma do código, talvez seja necessário adicionar uma chamada ao perfil ou método no trecho de código.

Teste se sua aplicação funciona conforme o esperado com o segredo no lugar das credenciais codificadas.

Etapa 3: alternar o segredo

A última etapa é revogar as credenciais codificadas alternando o segredo. Alternância é o processo de atualizar periodicamente um segredo. Quando o Secrets Manager alterna um segredo, você atualiza as credenciais tanto no segredo como no banco de dados. O Secrets Manager pode alternar automaticamente um segredo para você em uma programação que você estabelecer.

Parte da configuração da alternância é garantir que a função de alternância do Lambda possa acessar o Secrets Manager e seu banco de dados. Quando você ativa a alternância automática, o Secrets Manager cria a função de alternância do Lambda na mesma VPC do banco de dados para acessar o banco de dados pela rede. A função de alternância do Lambda também deve conseguir fazer chamadas ao Secrets Manager para atualizar o segredo. Recomendamos que você crie um endpoint do Secrets Manager na VPC para que as chamadas do Lambda para o Secrets Manager não saiam da infraestrutura. AWS Para obter instruções, consulte Endpoint da VPC.

Para ativar a alternância

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na página Secrets (Segredos), escolha o segredo.
3. Na página Secret details (Detalhes do segredo), na seção Rotation configuration (Configuração da alternância), escolha Edit rotation (Editar alternância).
4. Na caixa de diálogo Edit rotation configuration (Editar configuração da alternância), siga estas etapas:
    a. Ative a Automatic rotation (Alternância automática).
    b. Em Rotation schedule (Programação de alternância), insira a sua programação no fuso horário UTC.
    c. Escolha Rotate immediately when the secret is stored (Alternar imediatamente quando o segredo for armazenado) para alternar o segredo assim que suas alterações forem salvas.
    d. Em Rotation function (Função de alternância), escolha Create a new Lambda function (Criar uma nova função Lambda) e insira um nome para a nova função. O Secrets Manager adiciona "SecretsManager" no início do nome da função.
    e. Em Estratégia de alternância, escolha Usuário único.
    f. Selecione Salvar.

Para verificar se o segredo foi alternado

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Escolha Secrets (Segredos) e escolha o segredo.
3. Na página Secret details (Detalhes do segredo), role para baixo e escolha Retrieve secret value (Recuperar valor do segredo).

   Se o valor do segredo foi alterado, a alternância funcionou. Se o valor secreto não mudou, você precisa Solução de problemas de alternância do examinar os CloudWatch registros da função de rotação.

Teste se a aplicação funciona conforme o esperado com o segredo alternado.

Próximas etapas

Depois de remover um segredo codificado de seu código, há algumas ideias a serem consideradas em seguida:

- É possível melhorar a perfomance e reduzir custos armazenando segredos em cache. Para ter mais informações, consulte Obtenha segredos.
- É possível escolher uma programação de alternância diferente. Para ter mais informações, consulte the section called “Cronogramas de rotação”.
- Para encontrar segredos codificados em seus aplicativos Java e Python, recomendamos o Amazon Reviewer. CodeGuru

Configure a rotação alternada de usuários para AWS Secrets Manager

Neste tutorial, você aprenderá como configurar a alternância de usuários alternados para um segredo que contenha credenciais de banco de dados. Alternância de usuários alternados é uma estratégia de alternância em que o Secrets Manager clona o usuário e, em seguida, alterna quais credenciais do usuário são atualizadas. Essa estratégia é uma boa opção se você precisar de alta disponibilidade para seu segredo, porque um dos usuários alternados tem credenciais atuais para o banco de dados enquanto o outro está sendo atualizado. Para ter mais informações, consulte the section called “Usuários alternados”.

Para configurar a alternância de usuários alternados, você precisa de dois segredos:

- Um segredo com as credenciais que você deseja alternar.
- Um segundo segredo que tem credenciais de administrador.

  Esse usuário tem permissões para clonar o primeiro usuário e alterar a senha do primeiro usuário. Neste tutorial, o Amazon RDS cria esse segredo para um usuário administrador. O Amazon RDS também gerencia a alternância da senha do administrador. Para ter mais informações, consulte the section called “Alternância gerenciada”.

A primeira parte deste tutorial está configurando um ambiente realista. Para mostrar como funciona a alternância, este tutorial usa um exemplo de banco de dados MySQL do Amazon RDS. Por segurança, o banco de dados está em uma VPC que não permite acesso de entrada à internet. Para se conectar ao banco de dados do computador local pela Internet, você usa um bastion host, um servidor na VPC que pode se conectar ao banco de dados, mas que também permite conexões SSH pela internet. O bastion host neste tutorial é uma instância do Amazon EC2, e os grupos de segurança da instância impedem outros tipos de conexões.

Depois de concluir o tutorial, recomendamos limpar os recursos do tutorial. Não os use em um ambiente de produção.

A rotação do Secrets Manager usa uma AWS Lambda função para atualizar o segredo e o banco de dados. Para obter mais informações sobre os custos do uso de uma função do Lambda, consulte Definição de preço.

Tutorial:

- Permissões
- Pré-requisitos
- Etapa 1: criar um usuário do banco de dados do Amazon RDS
- Etapa 2: criar um segredo para as credenciais do usuário
- Etapa 3: testar o segredo alternado
- Etapa 4: Limpar os recursos
- Próximas etapas

Permissões

Para os pré-requisitos do tutorial, você precisa de permissões administrativas para sua Conta da AWS. Em uma configuração de produção, é uma prática recomendada usar funções diferentes para

cada uma das etapas. Por exemplo, uma função com permissões de administrador de banco de dados criaria o banco de dados do Amazon RDS, e uma função com permissões de administrador de rede configuraria a VPC e os grupos de segurança. Para as etapas do tutorial, recomendamos que você continue usando a mesma identidade.

Para obter mais informações sobre como configurar permissões em um ambiente de produção, consulte Autenticação e controle de acesso.

Pré-requisitos

Para este tutorial, você precisa do seguinte:

- Pré-requisito A: Amazon VPC
- Pré-requisito B: instância do Amazon EC2
- Pré-requisito C: banco de dados do Amazon RDS e um segredo do Secrets Manager para as credenciais do administrador
- Pré-requisito D: Permita que seu computador local se conecte à instância do EC2

Pré-requisito A: Amazon VPC

Nesta etapa, crie uma VPC na qual você pode iniciar um banco de dados do Amazon RDS e uma instância do Amazon EC2. Em uma etapa posterior, você usará seu computador para se conectar pela Internet ao bastion e depois ao banco de dados, portanto, precisará permitir que o tráfego saia da VPC. Para fazer isso, o Amazon VPC conecta um gateway da Internet à VPC e adiciona uma rota na tabela de rotas de modo que o tráfego com destino para fora da VPC seja enviado para o gateway da Internet.

Dentro da VPC, você cria um endpoint do Secrets Manager e um endpoint do Amazon RDS. Quando você configura a alternância automática em uma etapa posterior, o Secrets Manager cria uma função de alternância do Lambda dentro da VPC para que ela possa acessar o banco de dados. A função de alternância do Lambda também chama o Secrets Manager para atualizar o segredo e chama o Amazon RDS para obter as informações de conexão do banco de dados. Ao criar endpoints na VPC, você garante que as chamadas da função Lambda para o Secrets Manager e o Amazon RDS não saiam da infraestrutura. AWS Em vez disso, elas são encaminhadas para os endpoints dentro da VPC.

Para criar uma VPC

1. Abra o console do Amazon VPC em https://console.aws.amazon.com/vpc/.

2. Escolha Criar VPC.
3. Na página Create VPC (Criar VPC), escolha VPC and more (VPC e muito mais).
4. Em Name tag auto-generation (Geração automática de tags de nome), em Auto-generate (Gerar automaticamente), insira **`SecretsManagerTutorial`**.
5. Em DNS options (Opções de DNS), escolha **`Enable DNS hostnames`** e **`Enable DNS resolution`**.
6. Escolha Criar VPC.

Para criar um endpoint do Secrets Manager na VPC

1. No console do Amazon VPC, em Endpoints, escolha Create Endpoint (Criar endpoint).
2. Em Endpoint settings (Configurações de endpoint), em API Name (Nome da API), insira **`SecretsManagerTutorialEndpoint`**.
3. Em Services (Serviços), insira **`secretsmanager`** para filtrar a lista e, em seguida, selecione o endpoint do Secrets Manager na sua Região da AWS. Por exemplo, no Leste dos EUA (Norte da Virgínia), escolha `com.amazonaws.us-east-1.secretsmanager`.
4. Em VPC, escolha **`vpc**** (SecretsManagerTutorial)`**.
5. Em Sub-redes, selecione todas as Availability Zones (Zonas de disponibilidade) e, em seguida, para cada uma, escolha um Subnet ID (ID de sub-rede) para incluir.
6. Em IP address type (Tipo de endereço IP), escolha **`IPv4`**.
7. Em Security groups (Grupos de segurança), escolha o grupo de segurança padrão.
8. Em Policy (Política), selecione **`Full access`**.
9. Escolha Criar endpoint.

Para criar um endpoint do Amazon RDS na VPC

1. No console do Amazon VPC, em Endpoints, escolha Create Endpoint (Criar endpoint).
2. Em Endpoint settings (Configurações de endpoint), em API Name (Nome da API), insira **`RDSTutorialEndpoint`**.
3. Em Services (Serviços), insira **`rds`** para filtrar a lista e, em seguida, selecione o endpoint do Amazon RDS na sua Região da AWS. Por exemplo, no Leste dos EUA (Norte da Virgínia), escolha `com.amazonaws.us-east-1.rds`.
4. Em VPC, escolha **`vpc**** (SecretsManagerTutorial)`**.

5. Em Sub-redes, selecione todas as Availability Zones (Zonas de disponibilidade) e, em seguida, para cada uma, escolha um Subnet ID (ID de sub-rede) para incluir.
6. Em IP address type (Tipo de endereço IP), escolha **`IPv4`**.
7. Em Security groups (Grupos de segurança), escolha o grupo de segurança padrão.
8. Em Policy (Política), selecione **`Full access`**.
9. Escolha Criar endpoint.

Pré-requisito B: instância do Amazon EC2

O banco de dados do Amazon RDS que você criar em uma etapa posterior estará na VPC, portanto, para acessá-lo, você precisa de um bastion host. O bastion host também está na VPC, mas em uma etapa posterior, você configura um grupo de segurança de modo que seu computador local se conecte ao bastion host com SSH.

Para criar uma instância do EC2 para um bastion host

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Escolha Instances (Instâncias) e, em seguida, escolha Launch Instances (Iniciar instâncias).
3. Em Name and tags (Nome e etiquetas), em Name (Nome), insira **`SecretsManagerTutorialInstance`**.
4. Em Application and OS Images (Imagens do aplicativo e do sistema operacional), mantenha o padrão **`Amazon Linux 2 AMI (HMV) Kernel 5.10`**.
5. Em Instance type (Tipo de instância), mantenha o padrão **`t2.micro`**.
6. Em Key pair (Par de chaves), escolha Create key pair (Criar par de chaves).

   Na caixa de diálogo Create Key Pair (Criar par de chaves), em Key pair name (Nome do par de chaves), insira **`SecretsManagerTutorialKeyPair`** e selecione Create key pair (Criar par de chaves).

   O download do par de chaves será realizado automaticamente.
7. Em Network settings (Configurações da rede), escolha Edit (Editar), e faça o seguinte:

   a. Em VPC, escolha **`vpc-**** SecretsManagerTutorial`**.
   b. Em Auto-assign Public IP (Atribuir IP público automaticamente), selecione **`Enable`**.
   c. Em Firewall, escolha Select existing security group (Selecionar grupo de segurança existente).

d. Em Common security groups (Grupos de segurança comuns), escolha **`default`**.

8. Escolha Iniciar instância.

Pré-requisito C: banco de dados do Amazon RDS e um segredo do Secrets Manager para as credenciais do administrador

Nesta etapa, você cria um banco de dados do Amazon RDS MySQL e o configura de modo que o Amazon RDS crie um segredo para conter as credenciais do administrador. Em seguida, o Amazon RDS gerencia automaticamente a alternância do segredo do administrador para você. Para ter mais informações, consulte Alternância gerenciada.

Como parte da criação do seu banco de dados, você especifica o bastion host que criou na etapa anterior. Em seguida, o Amazon RDS configura grupos de segurança para que o banco de dados e a instância possam se acessar. Você adiciona uma regra ao grupo de segurança anexado à instância para permitir que seu computador local também se conecte a ela.

Para criar um banco de dados Amazon RDS com um segredo do Secrets Manager que contém as credenciais de administrador

1. No console do Amazon RDS, escolha Create database (Criar banco de dados).
2. Na seção Engine options (Opções de mecanismo), em Engine type (Tipo de mecanismo), escolha **`MySQL`**.
3. Na seção Templates (Modelos), escolha **`Free tier`**.
4. Na seção Settings (Configurações), faça o seguinte:

   a. Para DB instance identifier (Identificador de instância de banco de dados), insira **`SecretsManagerTutorial`**.

   b. Em Configurações de credenciais, selecione Gerenciar credenciais mestras em. AWS Secrets Manager

5. Na seção Connectivity (Conectividade), em Computer resource (Recurso do computador), escolha Connect to an EC2 computer resource (Conectar a um recurso de computador do EC2) e, em seguida, em EC2 Instance (Instância do EC2), escolha **`SecretsManagerTutorialInstance`**.
6. Selecione Criar banco de dados.

Pré-requisito D: Permita que seu computador local se conecte à instância do EC2

Nesta etapa, você configura a instância do EC2 criada no Pré-requisito B para permitir que seu computador local se conecte a ela. Para fazer isso, você edita o grupo de segurança que o Amazon RDS adicionou no Pré-requisito C para incluir uma regra que permite que o endereço IP do seu computador se conecte ao SSH. A regra permite que seu computador local (identificado pelo seu endereço IP atual) se conecte ao bastion host usando SSH pela Internet.

Para permitir que seu computador local se conecte à instância do EC2

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Na instância EC2 SecretsManagerTutorialInstance, na guia Segurança, em Grupos de segurança, escolha**`sg-*** (ec2-rds-X)`**.
3. Em Input Rules (Regras de entrada), selecione Edit inbound rules (Editar regras de entrada).
4. Selecione Add Rule (Adicionar regra) e, na regra, e faça o seguinte:
   a. Em Type (Tipo), escolha **`SSH`**.
   b. Em Source type (Tipo de fonte), selecione **`My IP`**.

Etapa 1: criar um usuário do banco de dados do Amazon RDS

Primeiro, você precisa de um usuário cujas credenciais serão armazenadas no segredo. Para criar o usuário, faça login no banco de dados do Amazon RDS com credenciais de administrador. Para simplificar, no tutorial, você cria um usuário com permissão total para um banco de dados. Em um ambiente de produção, isso não é típico. Recomendamos que você siga o princípio de privilégio mínimo.

Para se conectar ao banco de dados, você usa uma ferramenta cliente de MySQL. Neste tutorial, você usa o MySQL Workbench, uma aplicação baseada em GUI. Baixe o MySQL Workbench em Download MySQL Workbench (Baixar MySQL Workbench).

Para se conectar ao banco de dados, crie uma configuração de conexão no MySQL Workbench. Para a configuração, você precisa de algumas informações do Amazon EC2 e do Amazon RDS.

Para criar uma conexão de banco de dados no MySQL Workbench

1. No MySQL Workbench, ao lado de MySQL Connections (Conexões do MySQL), escolha o botão (+).

2. Na caixa de diálogo Setup New Connection (Configurar uma nova conexão), faça o seguinte:

   a. Em Connection Name (Nome da conexão), insira **`SecretsManagerTutorial`**.

   b. Em Connection Method (Método de conexão), escolha **`Standard TCP/IP over SSH`**.

   c. Na guia Parameters (Parâmetros), faça o seguinte:

      i. Em SSH Hostname (Nome do host de SSH), insira o endereço IP público da instância do Amazon EC2.

         Você pode encontrar o endereço IP no console do Amazon EC2 escolhendo a instância. SecretsManagerTutorialInstance Copie o endereço IP em Public IPv4 DNS (DNS IPv4 público).

      ii. Em SSH Username (Nome de usuário do SSH), insira **`ec2-user`**.

      iii. Para Arquivo de chave SSH, escolha o arquivo de par de chaves SecretsManagerTutorialKeyPair.pem que você baixou no pré-requisito anterior.

      iv. Em MySQL Hostname (Nome do host do MySQL), insira o endereço do endpoint do Amazon RDS.

         Você pode encontrar o endereço do endpoint no console do Amazon RDS escolhendo a instância do banco de dados secretsmanagertutorialdb. Copie o endereço em Endpoint.

      v. Em Username (Nome do usuário), insira **`admin`**.

   d. Escolha OK.

Para recuperar a senha do administrador

1. No Console do Amazon RDS, navegue até o seu banco de dados.
2. Na guia Configuration (Configuração), em Master Credentials ARN (ARN das credenciais principais), escolha Manage in Secrets Manager (Gerenciar no Secrets Manager).

   O Console do Secrets Manager é aberto.
3. Na página de detalhes do seu segredo, escolha Retrieve secret value (Recuperar valor do segredo).
4. A senha aparece na seção Secret value (Valor do segredo).

Para criar um usuário de banco de dados

1. No MySQL Workbench, escolha a conexão. SecretsManagerTutorial
2. Digite a senha de administrador que você recuperou do segredo.
3. Em MySQL Workbench, na janela Query (Consultar), insira os seguintes comandos (incluindo uma senha forte) e, depois, escolha Execute (Executar).

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

   Na janela Output (Saída), você verá os comandos com êxito.

Etapa 2: criar um segredo para as credenciais do usuário

Em seguida, crie um segredo para armazenar as credenciais do usuário que acabou de criar. Este é o segredo que você estará alternando. Você ativa a alternância automática e, para indicar a estratégia de usuários alternados, você escolhe um segredo de superusuário separado que tenha permissão para alterar a senha do primeiro usuário.

1. Abra o console Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Selecione Armazenar um novo segredo.
3. Na página Choose secret type (Selecionar tipo de segredo), faça o seguinte:
   a. Em Secret type (Tipo de segredo), escolha Credentials for Amazon RDS database (Credenciais para o banco de dados do Amazon RDS).
   b. Em Credentials (Credenciais), insira o nome do usuário **`appuser`** e a senha inserida para o usuário do banco de dados que você criou usando o MySQL Workbench.
   c. Em Database (Banco de dados), escolha secretsmanagertutorialdb.
   d. Escolha Próximo.
4. Na página Configure secret (Configurar segredo) , em Secret name (Nome de segredo), insira **`SecretsManagerTutorialAppuser`** e, depois, escolha Next (Próximo).
5. Na página Configure rotation (Configurar alternância), faça o seguinte:
   a. Ative a Automatic rotation (Alternância automática).

b. Em Rotation schedule (Programação da alternância), defina uma programação de Days (Dias):**2** dias com Duration (Duração): **2h**. Mantenha Rotate immediately (Alternar imediatamente) selecionado.

c. Em Rotation function (Função de alternância), escolha Create a rotation function (Criar uma função de alternância), e, em seguida, para nome da função, insira **`tutorial-alternating-users-rotation`**.

d. Em Estratégia de alternância, escolha Usuários alternados e, em Segredo da credencial do administrador, escolha o segredo com o nome rds!cluster... e cuja Descrição inclui o nome do banco de dados que você criou neste tutorial **`secretsmanagertutorial`**, por exemplo, `Secret associated with primary RDS DB instance: arn:aws:rds:`*`Region`*`:`*`AccountId`*`:db:secretsmanagertutorial`.

e. Escolha Próximo.

6. Na página Review (Análise), escolha Store (Armazenar).

   Secrets Manager retorna à página de detalhes secretos. Na parte superior da página, você pode ver o status da configuração de alternância. O Secrets Manager usa CloudFormation para criar recursos como a função de rotação Lambda e uma função de execução que executa a função Lambda. Quando CloudFormation terminar, o banner muda para Segredo programado para rotação. A primeira alternância está completa.

Etapa 3: testar o segredo alternado

Agora que o segredo está alternado, você pode verificar se o segredo contém credenciais válidas. A senha no segredo mudou a partir das credenciais originais.

Para recuperar a nova senha do segredo

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Escolha Secrets (Segredos) e, em seguida, escolha o segredo **`SecretsManagerTutorialAppuser`**.
3. Na página Secret details (Detalhes do segredo), role para baixo e escolha Retrieve secret value (Recuperar valor do segredo).
4. Na tabela Key/value (Chave/valor), copie o Secret value (Valor do segredo) para **`password`**.

Para testar as credenciais

1. No MySQL Workbench, clique com o botão direito do mouse na conexão SecretsManagerTutoriale escolha Editar conexão.
2. Na caixa de diálogo Manage Server Connections (Gerenciar conexões do servidor), em Username (Nome do usuário), insira **`appuser`** e, depois, escolha Close (Fechar).
3. De volta ao MySQL Workbench, escolha a conexão. SecretsManagerTutorial
4. Na caixa de diálogo Open SSH Connection (Abrir conexão de SSH), em Password (Senha), cole a senha que você recuperou do segredo e escolha OK.

   Se as credenciais forem válidas, o MySQL Workbench será aberto na página de design do banco de dados.

Isso mostra que a alternância secreta é bem-sucedida. As credenciais no segredo foram atualizadas, e ele é uma senha válida para se conectar ao banco de dados.

Etapa 4: Limpar os recursos

Se você quiser tentar outra estratégia de alternância, a single user rotation (alternância de usuários alternados), pule a limpeza de recursos e vá para the section called “Alternâncias de usuário único”.

Ou então, para evitar possíveis cobranças e remover a instância do EC2 que tem acesso à Internet, exclua os seguintes recursos criados neste tutorial e seus pré-requisitos:

- Instância de bancos de dados do Amazon RDS. Para obter instruções, consulte Deleting a DB instance (Como excluir uma instância de banco de dados) no Amazon RDS User Guide (Guia do usuário do Amazon RDS).
- Instância do Amazon EC2. Para obter instruções, consulte Encerrar uma instância no Guia do usuário do Amazon EC2.
- Segredo `SecretsManagerTutorialAppuser` do Secrets Manager. Para obter instruções, consulte the section called “Excluir um segredo”.
- Endpoint do Secrets Manager. Para obter instruções, consulte Delete a VPC endpoint (Excluir um endpoint da VPC) no AWS PrivateLink Guide (Guia da).
- VPC endpoint. Para obter instruções, consulte Delete your VPC (Excluir sua VPC) no AWS PrivateLink Guide (Guia da).

Próximas etapas

- Saiba como recuperar segredos em suas aplicações.
- Saiba mais sobre outras programações de alternância.

Configurar a alternância de usuário único para o AWS Secrets Manager

Neste tutorial, você aprenderá como configurar a alternância de usuário único para um segredo que contenha credenciais de banco de dados. Alternância de usuário único é uma estratégia de alternância na qual o Secrets Manager atualiza as credenciais de um usuário no segredo e no banco de dados. Para ter mais informações, consulte the section called “Usuário único”.

Depois de concluir o tutorial, recomendamos limpar os recursos do tutorial. Não os use em um ambiente de produção.

A rotação do Secrets Manager usa uma AWS Lambda função para atualizar o segredo e o banco de dados. Para obter mais informações sobre os custos do uso de uma função do Lambda, consulte Definição de preço.

Sumário

Permissões

Para os pré-requisitos do tutorial, você precisa de permissões administrativas para sua Conta da AWS. Em uma configuração de produção, é uma prática recomendada usar funções diferentes para cada uma das etapas. Por exemplo, uma função com permissões de administrador de banco de dados criaria o banco de dados do Amazon RDS, e uma função com permissões de administrador de

rede configuraria a VPC e os grupos de segurança. Para as etapas do tutorial, recomendamos que você continue usando a mesma identidade.

Para obter mais informações sobre como configurar permissões em um ambiente de produção, consulte Autenticação e controle de acesso.

Pré-requisitos

O pré-requisito para este tutorial é the section called “Alternância de usuários alternados”. Não limpe os recursos no final do primeiro tutorial. Depois desse tutorial, você tem um ambiente realista com um banco de dados do Amazon RDS e um segredo do Secrets Manager que contém credenciais de administrador para o banco de dados. Você também tem um segundo segredo que contém credenciais para um usuário do banco de dados, mas você não usa esse segredo neste tutorial.

Você também tem uma conexão configurada no MySQL Workbench para se conectar ao banco de dados com as credenciais de administrador.

Etapa 1: criar um usuário do banco de dados do Amazon RDS

Primeiro, você precisa de um usuário cujas credenciais serão armazenadas no segredo. Para criar o usuário, faça login no banco de dados do Amazon RDS com credenciais de administrador que são armazenadas em um segredo. Para simplificar, no tutorial, você cria um usuário com permissão total para um banco de dados. Em um ambiente de produção, isso não é típico. Recomendamos que você siga o princípio de privilégio mínimo.

Para recuperar a senha do administrador

1. No Console do Amazon RDS, navegue até o seu banco de dados.
2. Na guia Configuration (Configuração), em Master Credentials ARN (ARN das credenciais principais), escolha Manage in Secrets Manager (Gerenciar no Secrets Manager).

   O Console do Secrets Manager é aberto.
3. Na página de detalhes do seu segredo, escolha Retrieve secret value (Recuperar valor do segredo).
4. A senha aparece na seção Secret value (Valor do segredo).

Para criar um usuário de banco de dados

1. No MySQL Workbench, clique com o botão direito do mouse na conexão SecretsManagerTutoriale escolha Editar conexão.
2. Na caixa de diálogo Manage Server Connections (Gerenciar conexões do servidor), em Username (Nome do usuário), insira **admin** e, depois, escolha Close (Fechar).
3. De volta ao MySQL Workbench, escolha a conexão. SecretsManagerTutorial
4. Digite a senha de administrador que você recuperou do segredo.
5. Em MySQL Workbench, na janela Query (Consultar), insira os seguintes comandos (incluindo uma senha forte) e, depois, escolha Execute (Executar).

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

   Na janela Output (Saída), você verá os comandos com êxito.

Etapa 2: criar um segredo para as credenciais do usuário do banco de dados

Em seguida, crie um segredo para armazenar as credenciais do usuário que acabou de criar e ative a alternância automática, incluindo uma alternância imediata. O Secrets Manager alterna o segredo, o que significa que a senha é gerada programaticamente, ou seja, nenhum ser humano viu essa nova senha. O início imediato da alternância também pode ajudar a determinar se a alternância está configurada corretamente.

1. Abra o console Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Selecione Armazenar um novo segredo.
3. Na página Choose secret type (Selecionar tipo de segredo), faça o seguinte:
   a. Em Secret type (Tipo de segredo), escolha Credentials for Amazon RDS database (Credenciais para o banco de dados do Amazon RDS).
   b. Em Credentials (Credenciais), insira o nome do usuário **dbuser** e a senha inserida para o usuário do banco de dados que você criou usando o MySQL Workbench.
   c. Em Database (Banco de dados), escolha secretsmanagertutorialdb.
   d. Escolha Próximo.

4. Na página Configure secret (Configurar segredo) , em Secret name (Nome de segredo), insira **`SecretsManagerTutorialDbuser`** e, depois, escolha Next (Próximo).
5. Na página Configure rotation (Configurar alternância), faça o seguinte:
   a. Ative a Automatic rotation (Alternância automática).
   b. Em Rotation schedule (Programação da alternância), defina uma programação de Days (Dias):**2** dias com Duration (Duração): **2h**. Mantenha Rotate immediately (Alternar imediatamente) selecionado.
   c. Em Rotation function (Função de alternância), escolha Create a rotation function (Criar uma função de alternância), e, em seguida, para nome da função, insira **`tutorial-single-user-rotation`**.
   d. Em Estratégia de alternância, escolha Usuário único.
   e. Escolha Próximo.
6. Na página Review (Análise), escolha Store (Armazenar).

   Secrets Manager retorna à página de detalhes secretos. Na parte superior da página, você pode ver o status da configuração de alternância. O Secrets Manager usa CloudFormation para criar recursos como a função de rotação Lambda e uma função de execução que executa a função Lambda. Quando CloudFormation terminar, o banner muda para Segredo programado para rotação. A primeira alternância está completa.

Etapa 3: testar a senha alternada

Após a primeira alternância de segredo, que pode levar alguns segundos, você pode verificar se o segredo ainda contém credenciais válidas. A senha no segredo mudou a partir das credenciais originais.

Para recuperar a nova senha do segredo

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Escolha Secrets (Segredos) e, em seguida, escolha o segredo **`SecretsManagerTutorialDbuser`**.
3. Na página Secret details (Detalhes do segredo), role para baixo e escolha Retrieve secret value (Recuperar valor do segredo).
4. Na tabela Key/value (Chave/valor), copie o Secret value (Valor do segredo) para **`password`**.

Para testar as credenciais

1. No MySQL Workbench, clique com o botão direito do mouse na conexão SecretsManagerTutoriale escolha Editar conexão.
2. Na caixa de diálogo Manage Server Connections (Gerenciar conexões do servidor), em Username (Nome do usuário), insira **`dbuser`** e, depois, escolha Close (Fechar).
3. De volta ao MySQL Workbench, escolha a conexão. SecretsManagerTutorial
4. Na caixa de diálogo Open SSH Connection (Abrir conexão de SSH), em Password (Senha), cole a senha que você recuperou do segredo e escolha OK.

   Se as credenciais forem válidas, o MySQL Workbench será aberto na página de design do banco de dados.

Etapa 4: Limpar os recursos

Para evitar possíveis cobranças, exclua o segredo que você criou neste tutorial. Para obter instruções, consulte the section called “Excluir um segredo”.

Para limpar os recursos criados no tutorial anterior, consulte the section called “Etapa 4: Limpar os recursos”.

Próximas etapas

- Saiba como recuperar segredos em suas aplicações. Consulte Obtenha segredos.
- Saiba mais sobre outras programações de alternância. Consulte the section called “Cronogramas de rotação”.

Autenticação e controle de acesso para AWS Secrets Manager

O Secrets Manager usa o AWS Identity and Access Management (IAM) para proteger o acesso a segredos. O IAM fornece autenticação e controle de acesso. Autenticação verifica a identidade das solicitações das pessoas. O Secrets Manager usa um processo de login com senhas, chaves de acesso e tokens de autenticação multifator (MFA) para verificar a identidade dos usuários. Consulte Como fazer login em AWS. Controle de acesso garante que apenas pessoas aprovadas possam executar operações em recursos da AWS , como segredos. O Secrets Manager usa políticas para definir quem tem acesso a quais recursos e quais ações a identidade pode executar nesses recursos. Consulte Políticas e permissões no IAM.

Permissões de administrador do Secrets Manager

Para conceder permissões de administrador ao Secrets Manager, siga as instruções de Adicionar e remover permissões de identidade do IAM e anexe as seguintes políticas:

- SecretsManagerReadWrite
- IAMFullAccess

Recomendamos que você não conceda permissões de administrador a usuários finais. Embora isso permita que os usuários criem e gerenciem os segredos, a permissão necessária para habilitar a alternância (IAMFullAccess) concede permissões significativas que não são adequadas para usuários finais.

Permissões para acessar segredos

Ao usar as políticas de permissão do IAM, você pode controlar quais usuários ou serviços têm acesso aos seus segredos. Uma política de permissões descreve quem pode executar quais ações em quais recursos. É possível:

- the section called “Anexar uma política de permissões a uma identidade”
- the section called “Anexar uma política de permissões a um segredo”

Permissões para funções de alternância do Lambda

O Secrets Manager usa AWS Lambda funções para alternar segredos. A função Lambda deve ter acesso ao segredo e também ao banco de dados ou serviço para o qual o segredo contém credenciais. Consulte Permissões para alternância.

Permissões para chaves de criptografia

O Secrets Manager usa chaves AWS Key Management Service (AWS KMS) para criptografar segredos. O tem Chave gerenciada pela AWS `aws/secretsmanager` automaticamente as permissões corretas. Se você usar uma chave do KMS diferente, o Secrets Manager necessitará de permissões para essa chave. Consulte the section called “Permissões para a chave do KMS”.

Permissões para replicação

Ao usar as políticas de permissão do IAM, você controla quais usuários ou serviços podem replicar seus segredos para outras regiões. Consulte the section called “Evite a replicação”.

Anexar uma política de permissões a uma identidade

É possível anexar políticas de permissões a identidades do IAM: usuários, grupos de usuários e funções. Em uma política baseada em identidades, especifique que segredos a identidade pode acessar e que ações a identidade pode executar nos segredos. Para obter mais informações, consulte Adicionar e remover permissões de identidade do IAM.

É possível conceder permissões a um perfil que representa uma aplicação ou um usuário em outro serviço. Por exemplo, uma aplicação em execução em uma instância do Amazon EC2 pode precisar de acesso a um banco de dados. Você pode criar um perfil do IAM anexado ao perfil de instância do EC2 e usar uma política de permissões para conceder à função acesso ao segredo que contém credenciais para o banco de dados. Para obter mais informações, consulte Uso de um perfil do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2. Outros serviços aos quais você pode anexar perfis incluem o Amazon Redshift, o AWS Lambda e o Amazon ECS.

Você também pode conceder permissões a usuários autenticados por um sistema de identidade diferente do IAM. Por exemplo, você pode associar funções do IAM a usuários de aplicações móveis que fazem login usando o Amazon Cognito. A função concede ao aplicativo credenciais temporárias

com as permissões na política de permissões da função. Em seguida, você pode usar uma política de permissões para conceder à função acesso ao segredo. Para obter mais informações, consulte Provedores de identidade e federação.

Você pode usar políticas baseadas em identidades para:

- Conceder um acesso de identidade a vários segredos.
- Controlar quem pode criar novos segredos e quem pode acessar segredos que ainda não foram criados.
- Conceder a um grupo do IAM acesso a segredos.

Para obter mais informações, consulte the section called “Exemplos de política de permissões”.

Anexo de uma política de permissões a um segredo do AWS Secrets Manager

Em uma política baseada em recursos, especifique quem pode acessar o segredo e as ações que essa pessoa pode executar no segredo. Você pode usar políticas baseadas em recursos para:

- Conceder acesso a vários usuários e funções a um único segredo.
- Conceda acesso a usuários ou funções em outras AWS contas.

Consulte the section called “Exemplos de política de permissões”.

Quando você anexa uma política baseada em recursos a um segredo no console, o Secrets Manager usa o mecanismo de raciocínio automatizado Zelkova e a API `ValidateResourcePolicy` para impedir que você conceda acesso a seus segredos a uma ampla gama de entidades principais do IAM. Você também pode chamar a API `PutResourcePolicy` com o parâmetro `BlockPublicPolicy` da CLI ou do SDK.

> **Important**
>
> A validação da política de recursos e o `BlockPublicPolicy` parâmetro ajudam a proteger seus recursos, impedindo que o acesso público seja concedido por meio das políticas de recursos diretamente associadas aos seus segredos. Além de usar esses recursos, inspecione cuidadosamente as políticas a seguir para confirmar se elas não concedem acesso público:

- Políticas baseadas em identidade vinculadas aos AWS diretores associados (por exemplo, funções do IAM)
- Políticas baseadas em recursos anexadas aos AWS recursos associados (por exemplo, chaves AWS Key Management Service (AWS KMS))

Para revisar as permissões de seus segredos, consulteDeterminação de quem tem permissões para seus segredos do .

Para visualizar, alterar ou excluir a política de recursos de um segredo (console)

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na lista de segredos, escolha o segredo.
3. Na página de detalhes do segredo, na guia de Visão geral da seção Permissões de recursos, escolha Editar permissões.
4. No campo do código, siga um dos procedimentos a seguir e escolha Save (Salvar):
   - Para anexar ou modificar uma política de recursos, insira a política.
   - Para excluir a política, limpe o campo do código.

AWS CLI

Example Recuperar uma política de recursos

O exemplo de `get-resource-policy` a seguir recupera a política baseada em recurso anexada a um segredo.

```
aws secretsmanager get-resource-policy \
    --secret-id MyTestSecret
```

Example Excluir uma política de recurso

O exemplo de `delete-resource-policy` a seguir exclui a política baseada em recurso anexada a um segredo.

```
aws secretsmanager delete-resource-policy \
```

```
    --secret-id MyTestSecret
```

Example Adicionar uma política de recurso

O exemplo de `put-resource-policy` a seguir adiciona uma política de permissões a um segredo, verificando primeiro se a política não fornece acesso amplo ao segredo. A política é lida de um arquivo. Para obter mais informações, consulte Carregando AWS CLI parâmetros de um arquivo no Guia AWS CLI do usuário.

```
aws secretsmanager put-resource-policy \
    --secret-id MyTestSecret \
    --resource-policy file://mypolicy.json \
    --block-public-policy
```

Conteúdo de `mypolicy.json`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:role/MyRole"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "*"
        }
    ]
}
```

AWS SDK

Para recuperar a política anexada a um segredo, use `GetResourcePolicy`.

Para excluir a política anexada a um segredo, use `DeleteResourcePolicy`.

Para anexar uma política a um segredo, use `PutResourcePolicy`. Se já houver uma política anexada, o comando a substituirá pela nova política. O documento da política deve estar formatado como texto JSON estruturado. Consulte Estrutura de documento de política JSON. Use os the section called “Exemplos de política de permissões” para começar a criar sua política.

Para ter mais informações, consulte the section called “AWS SDKs”.

AWS política gerenciada para AWS Secrets Manager

Uma política AWS gerenciada é uma política autônoma criada e administrada por AWS. AWS as políticas gerenciadas são projetadas para fornecer permissões para muitos casos de uso comuns, para que você possa começar a atribuir permissões a usuários, grupos e funções.

Lembre-se de que as políticas AWS gerenciadas podem não conceder permissões de privilégio mínimo para seus casos de uso específicos porque elas estão disponíveis para uso de todos os AWS clientes. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo cliente específicas para seus casos de uso.

Você não pode alterar as permissões definidas nas políticas AWS gerenciadas. Se AWS atualizar as permissões definidas em uma política AWS gerenciada, a atualização afetará todas as identidades principais (usuários, grupos e funções) às quais a política está anexada. AWS é mais provável que atualize uma política AWS gerenciada quando uma nova AWS service (Serviço da AWS) é lançada ou novas operações de API são disponibilizadas para serviços existentes.

Para mais informações, consulte Políticas gerenciadas pela AWS no Manual do usuário do IAM.

AWS política gerenciada: SecretsManagerReadWrite

Essa política fornece acesso de leitura/gravação aos recursos do Amazon RDS AWS Secrets Manager, Amazon Redshift e Amazon DocumentDB, incluindo permissão para descrever recursos do Amazon RDS, Amazon Redshift e Amazon DocumentDB, além de permissão para usar para criptografar e descriptografar segredos. AWS KMS Essa política também fornece permissão para criar conjuntos de AWS CloudFormation alterações, obter modelos de rotação de um bucket do Amazon S3 gerenciado por AWS, listar AWS Lambda funções e descrever as VPCs do Amazon EC2. Essas permissões são exigidas pelo console para configurar a rotação com as funções de rotação existentes.

Para criar novas funções de rotação, você também deve ter permissão para criar AWS CloudFormation pilhas e funções de AWS Lambda execução. Você pode atribuir a política FullAccess gerenciada do IAM. Consulte Permissões para alternância.

Detalhes das permissões

Esta política inclui as seguintes permissões:

- `secretsmanager`: permite que entidades principais realizem todas as ações do Secrets Manager.
- `cloudformation`— Permite que os diretores criem AWS CloudFormation pilhas. Isso é necessário para que os diretores que usam o console para ativar a rotação possam criar funções AWS CloudFormation de rotação do Lambda por meio de pilhas. Para ter mais informações, consulte the section called “Como o Secrets Manager usa AWS CloudFormation”.
- `ec2`: permite que as entidades principais descrevam as VPCs do Amazon EC2. Isso é necessário para que as entidades principais que usam o console possam criar funções de rotação na mesma VPC do banco de dados das credenciais que estão armazenando em um segredo.
- `kms`— Permite que os diretores usem AWS KMS chaves para operações criptográficas. Isso é necessário para que o Secrets Manager possa criptografar e descriptografar segredos. Para ter mais informações, consulte the section called “Criptografia e descriptografia de segredos”.
- `lambda`: permite que as entidades principais listem as funções de rotação do Lambda. Isso é necessário para que as entidades principais que usam o console possam escolher as funções de rotação existentes.
- `rds`: permite que as entidades principais descrevam clusters e instâncias no Amazon RDS. Isso é necessário para que as entidades principais que usam o console possam escolher clusters ou instâncias do Amazon RDS.
- `redshift`: permite que as entidades principais descrevam clusters no Amazon Redshift. Isso é necessário para que as entidades principais que usam o console possam escolher clusters do Amazon Redshift.
- `redshift-serverless`— Permite que os diretores descrevam namespaces no Amazon Redshift Serverless. Isso é necessário para que os diretores que usam o console possam escolher namespaces sem servidor do Amazon Redshift.
- `docdb-elastic`: permite que as entidades principais descrevam clusters elásticos no Amazon DocumentDB. Isso é necessário para que as entidades principais que usam o console possam escolher clusters elásticos do Amazon DocumentDB.
- `tag`: permite que as entidades principais obtenham todos os recursos marcados na conta.
- `serverlessrepo`— Permite que os diretores criem conjuntos de AWS CloudFormation mudanças. Isso é necessário para que as entidades principais que usam o console possam criar funções de rotação do Lambda. Para ter mais informações, consulte the section called “Como o Secrets Manager usa AWS CloudFormation”.
- `s3`— Permite que os diretores obtenham objetos de um bucket do Amazon S3 que é gerenciado pelo. AWS Esse bucket contém Modelos de função de alternância do Lambda. Essa permissão é

necessária para que as entidades principais que usam o console possam criar funções de rotação do Lambda com base nos modelos no bucket. Para ter mais informações, consulte the section called “Como o Secrets Manager usa AWS CloudFormation”.

Para ver a política, consulte o documento de política SecretsManagerReadWrite JSON.

Atualizações do Secrets Manager para políticas AWS gerenciadas

Veja detalhes sobre as atualizações das políticas AWS gerenciadas do Secrets Manager.

Alteração	Descrição	Data
SecretsManagerReadWrite: atualizar para uma política existente	Essa política foi atualizada para permitir descrever o acesso ao Amazon Redshift Serverless para que os usuários do console possam escolher um namespace do Amazon Redshift Serverless ao criarem um segredo do Amazon Redshift.	12 de março de 2024
SecretsManagerReadWrite: atualização para uma política existente	Essa política foi atualizada para permitir descrever o acesso aos clusters elásticos do Amazon DocumentDB para que os usuários do console possam escolher um cluster elástico ao criar um segredo do Amazon DocumentDB.	12 de setembro de 2023
SecretsManagerReadWrite: atualização para uma política existente	Essa política foi atualizada para permitir descrever o acesso ao Amazon Redshift para que os usuários do console possam escolher um cluster do Amazon Redshift ao criarem um segredo do	24 de junho de 2020

Alteração	Descrição	Data
	Amazon Redshift. A atualizaç ão também adicionou novas permissões para permitir acesso de leitura a um bucket do Amazon S3 gerenciad o pela empresa AWS que armazena os modelos de função de rotação do Lambda.	
SecretsManagerReadWrite: atualização para uma política existente	Essa política foi atualizad a para permitir descrever o acesso aos clusters do Amazon RDS para que os usuários do console possam escolher um cluster ao criar um segredo do Amazon RDS.	3 de maio de 2018
SecretsManagerReadWrite – Nova política	O Secrets Manager criou uma política para conceder as permissões necessárias para usar o console com todo o acesso de leitura e gravação ao Secrets Manager.	04 de abril de 2018
O Secrets Manager começou a rastrear as alterações	O Secrets Manager começou a monitorar as mudanças em suas políticas AWS gerenciad as.	04 de abril de 2018

Determinação de quem tem permissões para seus segredos do AWS Secrets Manager

Por padrão, as identidades do IAM não têm permissão para acessar segredos. Ao autorizar o acesso a um segredo, o Secrets Manager avalia a política baseada em recursos anexada ao segredo e

todas as políticas baseadas em identidades anexadas ao usuário ou à função do IAM que está enviando a solicitação. Para fazer isso, o Secrets Manager usa um processo semelhante ao descrito em Determinar se uma solicitação é permitida ou negada no Manual do usuário do IAM.

Quando várias políticas se aplicarem a uma solicitação, o Secrets Manager usará uma hierarquia para controlar as permissões:

1. Se uma declaração em qualquer política com uma `deny` explícita corresponder à ação de solicitação e ao recurso:

   A `deny` explícita substituirá todo o resto e bloqueará a ação.
2. Se não houver qualquer `deny` explícita, mas uma declaração com uma `allow` explícita corresponder à ação de solicitação e ao recurso:

   A `allow` explícita concederá à ação de solicitação acesso aos recursos da declaração.

   Se a identidade e o segredo estiverem em duas contas diferentes, deverá haver uma `allow` na política de recursos para o segredo e na política anexada à identidade, caso contrário, a AWS negará a solicitação. Para obter mais informações, consulte Acesso entre contas.
3. Se não houver qualquer declaração com uma `allow` explícita que corresponda à ação de solicitação e ao recurso:

   A AWS negará a solicitação por padrão, o que é denominado negação implícita.

Para visualizar a política baseada em recursos de um segredo

- Faça um dos seguintes procedimentos:
  - Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/. Na página de detalhes do segredo, na seção Resource permissions (Permissões de recursos), escolha Edit permissions (Editar permissões).
  - Use a AWS CLI para chamar `get-resource-policy` ou o AWS SDK para chamar `GetResourcePolicy`.

Para determinar quem tem acesso por meio de políticas baseadas em identidades

- Use o simulador de políticas do IAM. Consulte Testar políticas do IAM com o simulador de políticas do IAM

Acesse AWS Secrets Manager segredos de uma conta diferente

Para permitir que os usuários de uma conta acessem segredos em outra conta (acesso entre contas), você deve permitir o acesso em uma política de recursos e em uma política de identidade. Isso é diferente de conceder acesso a identidades na mesma conta do segredo.

Você também deve permitir que a identidade use a chave do KMS com a qual o segredo está criptografado. Isso ocorre porque você não pode usar o Chave gerenciada pela AWS (`aws/secretsmanager`) para acesso entre contas. Em vez disso, você precisa criptografar o segredo com uma chave do KMS que criar e, em seguida, anexar uma política de chave a ele. Existe uma cobrança pela criação de chaves do KMS. Para alterar a chave de criptografia de um segredo, consulte the section called “Modificar um segredo”.

Os exemplos de políticas a seguir supõem que você tenha um segredo e uma chave de criptografia na Conta1 e uma identidade na Conta2, à qual você quer permitir acesso ao valor do segredo.

Etapa 1: anexar uma política de recursos ao segredo na Conta1

- A política a seguir permite que *ApplicationRole*na *Conta 2* acesse o segredo na *Conta* 1. Para usar essa política, consulte the section called “Anexar uma política de permissões a um segredo”.

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

Etapa 2: adicionar uma instrução à política de chaves para a chave KMS na Conta1

- *A declaração de política chave a seguir permite que ApplicationRolena Conta 2 use a chave KMS na Conta 1 para decifrar o segredo na Conta 1.*

Para usar essa declaração, adicione-a à política de chaves para sua chave do KMS. Para obter mais informações, consulte Alterar uma política de chaves.

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::Account2:role/ApplicationRole"
  },
  "Action": [
    "kms:Decrypt",
    "kms:DescribeKey"
  ],
  "Resource": "*"
}
```

Etapa 3: anexar uma política de identidade à identidade na Conta2

- *A política a seguir permite que ApplicationRolena Conta 2 acesse o segredo na Conta 1 e descriptografe o valor secreto usando a chave de criptografia que também está na Conta 1.* Para usar essa política, consulte the section called “Anexar uma política de permissões a uma identidade”. Você pode encontrar o ARN do seu segredo no console do Secrets Manager na página de detalhes do segredo em ARN do segredo. Como alternativa, você pode chamar `describe-secret`.

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

Acesse segredos de um ambiente local

Você pode usar o AWS Identity and Access Management Roles Anywhere para obter credenciais de segurança temporárias no IAM para cargas de trabalho, como servidores, contêineres e aplicativos executados fora do. AWS Suas cargas de trabalho podem usar as mesmas políticas e funções do IAM que você usa com AWS aplicativos para acessar AWS recursos. Com o IAM Roles Anywhere, você pode usar o Secrets Manager para armazenar e gerenciar credenciais que podem ser acessadas por recursos e dispositivos locais, como servidores de aplicativos. AWS Para obter mais informações, consulte o Guia do usuário do IAM Roles Anywhere.

Exemplos de políticas de permissões para AWS Secrets Manager

Uma política de permissões é um texto estruturado JSON. Consulte Estrutura de documento de política JSON.

As políticas de permissões que você anexa a recursos e identidades são muito semelhantes. Alguns elementos que você inclui em uma política de acesso a segredos incluem:

- `Principal`: a quem conceder acesso. Consulte Especificar um elemento principal no Manual do usuário do IAM. Ao anexar uma política a uma identidade, você não inclui um elemento `Principal` na política.
- `Action`: o que eles podem fazer. Consulte the section called “Ações do Secrets Manager”.
- `Resource`: quais segredos eles podem acessar. Consulte the section called “Recursos do Secrets Manager”.

  O caractere curinga (*) tem significados diferentes, que dependem de onde a política será anexada:

  - Em uma política anexada a um segredo, * significa que a política se aplica a este segredo.
  - Em uma política anexada a uma identidade, * significa que a política se aplica a todos os recursos, incluindo segredos, na conta.

Para anexar uma política a um segredo, consulte the section called “Anexar uma política de permissões a um segredo”.

Para anexar uma política a uma identidade, consulte the section called “Anexar uma política de permissões a uma identidade”.

Tópicos

Exemplo: permissão para recuperar valores de segredos individuais

Para conceder permissão para recuperar valores de segredos, você pode anexar políticas a segredos ou identidades. Para obter ajuda na determinação do tipo de política a ser usado, consulte Políticas baseadas em identidades e políticas baseadas em recursos. Para obter informações sobre como anexar uma política, consulte the section called “Anexar uma política de permissões a um segredo” e the section called “Anexar uma política de permissões a uma identidade”.

Os exemplos a seguir mostram duas formas diferentes de conceder acesso a um segredo. O primeiro exemplo é uma política baseada em recursos que pode ser anexada a um segredo. Esse exemplo é útil quando você quer conceder acesso a um único segredo a vários usuários ou funções. O segundo exemplo é uma política baseada em identidades que pode ser anexada a um usuário ou função no IAM. Esse exemplo é útil quando você deseja conceder acesso a um grupo do IAM. Para conceder permissão para recuperar um grupo de segredos em uma chamada de API em lote, consulte the section called “Exemplo: permissão para recuperar um grupo de valores secretos em um lote”.

Example Ler um segredo (anexar a um segredo)

É possível conceder acesso a um segredo anexando a seguinte política ao segredo. Para usar essa política, consulte the section called “Anexar uma política de permissões a um segredo”.

```
{
```

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

Example Ler um segredo criptografado usando uma chave gerenciada pelo cliente (anexar à identidade)

Se um segredo for criptografado usando uma chave gerenciada pelo cliente, você poderá conceder acesso para ler o segredo anexando a política a seguir a uma identidade. Para usar essa política, consulte the section called “Anexar uma política de permissões a uma identidade”.

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

Exemplo: permissão para ler e descrever segredos individuais

Example Leia e descreva um segredo (anexe a uma identidade)

É possível conceder acesso a um segredo anexando a seguinte política a uma identidade. Para usar essa política, consulte the section called “Anexar uma política de permissões a uma identidade”.

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

Exemplo: permissão para recuperar um grupo de valores secretos em um lote

Example Ler um grupo de segredos em um lote (anexar à identidade)

É possível conceder acesso para recuperar um grupo de segredos em uma chamada de API em lote anexando a seguinte política a uma identidade. A política restringe o chamador para que ele só possa recuperar os segredos especificados por *SecretARN1*, *SecretARN2* e *SecretARN3*, mesmo que a chamada em lote inclua outros segredos. Se o chamador também solicitar outros segredos na chamada de API em lote, o Secrets Manager não os retornará. Para obter mais informações, consulte `BatchGetSecretValue.` . Para usar essa política, consulte the section called “Anexar uma política de permissões a uma identidade”.

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

```
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "SecretARN1",
        "SecretARN2",
        "SecretARN3"
        ]
    }
  ]
}
```

Exemplo: curingas

Você pode usar curingas para incluir um conjunto de valores em um elemento de política.

Example Acessar todos os segredos em um caminho (anexar à identidade)

A política a seguir concede acesso para recuperar todos os segredos com um nome que começa com "*TestEnv/*". Para usar essa política, consulte the section called “Anexar uma política de permissões a uma identidade”.

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

Example Acessar metadados em todos os segredos (anexar à identidade)

A política a seguir concede `DescribeSecret` e permissões, começando com `List`: `ListSecrets` e `ListSecretVersionIds`. Para usar essa política, consulte the section called “Anexar uma política de permissões a uma identidade”.

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

```
      "secretsmanager:DescribeSecret",
      "secretsmanager:List*"
    ],
    "Resource": "*"
  }
}
```

Example Corresponder nome do segredo (anexar à identidade)

A política a seguir concede todas as permissões do Secrets Manager para um segredo por nome. Para usar essa política, consulte the section called “Anexar uma política de permissões a uma identidade”.

Para corresponder um nome de segredo, crie o ARN para o segredo reunindo a região, o ID da conta, o nome do segredo e o caractere curinga (?) para corresponder caracteres aleatórios individuais. O Secrets Manager anexa seis caracteres aleatórios a nomes de segredos como parte do ARN para que você possa usar esse curinga para corresponder a esses caracteres. Se você usar a sintaxe `"another_secret_name-*"`, o Secrets Manager corresponderá não apenas ao segredo previsto com os seis caracteres aleatórios, mas também a `"another_secret_name-<anything-here>a1b2c3"`.

Como você pode prever todas as partes do ARN de um segredo, exceto os seis caracteres aleatórios, o uso da sintaxe `'??????'` do caractere curinga permite que você conceda com segurança permissões a um segredo que ainda não existe. Mas observe que, se você excluir o segredo e recriá-lo com o mesmo nome, o usuário receberá automaticamente permissão para o novo segredo, mesmo que os 6 caracteres tenham sido alterados.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:*",
      "Resource": [
        "arn:aws:secretsmanager:Region:AccountId:secret:a_specific_secret_name-a1b2c3",
        "arn:aws:secretsmanager:Region:AccountId:secret:another_secret_name-??????"
      ]
    }
  ]
}
```

Exemplo: Permissão para criar segredos

Para conceder permissões a um usuário para criar um segredo, recomendamos anexar uma política de permissões a um grupo do IAM ao qual o usuário pertence. Consulte Grupos de usuários do IAM.

Example Criar segredos (anexar à identidade)

A política a seguir concede permissão para a criação de segredos e a visualização de uma lista de segredos. Para usar essa política, consulte the section called “Anexar uma política de permissões a uma identidade”.

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

Exemplo: negar uma AWS KMS chave específica para criptografar segredos

> **Important**
>
> Para negar uma chave gerenciada pelo cliente, recomendamos que você restrinja o acesso usando uma política de chaves ou uma concessão de chaves. Para obter mais informações, consulte Autenticação e controle de acesso AWS KMS no Guia do AWS Key Management Service desenvolvedor.

Example Negar a chave AWS gerenciada **`aws/secretsmanager`** (anexar à identidade)

A política a seguir mostra como negar o uso da chave AWS gerenciada `aws/secretsmanager` para criar ou atualizar segredos. Isso significa que os segredos devem ser criptografados usando

uma chave gerenciada pelo cliente. Se a `aws/secretsmanager` chave existir, você também deverá incluir o ID da chave. Você também inclui a string vazia porque o Secrets Manager a interpreta como a chave AWS `aws/secretsmanager` gerenciada. A segunda declaração nega solicitações para criar segredos que não incluam uma chave KMS, porque o Secrets Manager interpreta isso como a AWS chave gerenciada. `aws/secretsmanager`

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Sid": "RequireCustomerManagedKeysOnSecrets",
          "Effect": "Deny",
          "Action": [
              "secretsmanager:CreateSecret",
              "secretsmanager:UpdateSecret"
          ],
          "Resource": "*",
          "Condition": {
              "ForAnyValue:StringLikeIfExists": {
                  "secretsmanager:KmsKeyId": [
                      "*alias/aws/secretsmanager",
                      "*<key_ID_of_the_AWS_managed_key>",
                      ""
                  ]
              }
          }
      },
      {
          "Sid": "RequireKmsKeyIdParameterOnCreate",
          "Effect": "Deny",
          "Action": "secretsmanager:CreateSecret",
          "Resource": "*",
          "Condition": {
              "Null": {
                  "secretsmanager:KmsKeyId": "true"
              }
          }
      }
  ]
}
```

Exemplo: Permissões e VPCs

Se você precisar acessar o Secrets Manager de uma VPC, verifique se as solicitações para o Secrets Manager vêm da VPC ao incluir uma condição nas políticas de permissões. Para obter mais informações, consulte Condições do endpoint da VPC e Endpoint da VPC.

Certifique-se de que as solicitações para acessar o segredo de outros AWS serviços também venham da VPC, caso contrário, essa política negará o acesso a elas.

Example Exigir que as solicitações sejam enviadas por meio de um endpoint da VPC (anexado ao segredo)

A política a seguir permite que um usuário execute operações do Secrets Manager somente quando a solicitação é fornecida por meio do endpoint do VPC *vpce-1234a5678b9012c*. Para usar essa política, consulte the section called “Anexar uma política de permissões a um segredo”.

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

Example Exigir que as solicitações sejam provenientes de uma VPC (anexar ao segredo)

A política a seguir permite que os comandos criem e gerenciem segredos somente quando eles são provenientes da *vpc-12345678*. Além disso, a política permite operações que usam o acesso ao valor criptografado do segredo somente quando as solicitações são recebidas de `vpc-2b2b2b2b`. Você poderá usar uma política como essa se um aplicativo for executado em uma VPC, mas você

usa uma segunda VPC isolada para funções de gerenciamento. Para usar essa política, consulte the section called “Anexar uma política de permissões a um segredo”.

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

```
  ]
}
```

Exemplo: Controlar o acesso a segredos usando etiquetas

Você pode usar etiquetas para controlar o acesso aos segredos. O uso de etiquetas para controlar permissões é útil em ambientes que estão crescendo rapidamente e ajuda em situações em que o gerenciamento de políticas se torna um problema. Uma estratégia é anexar etiquetas a segredos e, em seguida, conceder permissões a uma identidade quando um segredo tem uma etiqueta específica.

Example Permitir acesso a segredos com uma etiqueta específica (anexar a uma identidade)

A política a seguir permite `DescribeSecret` segredos com uma tag com a chave "*ServerName*" e o valor "*ServerABC*". Para usar essa política, consulte the section called “Anexar uma política de permissões a uma identidade”.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "secretsmanager:DescribeSecret",
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "secretsmanager:ResourceTag/ServerName": "ServerABC"
      }
    }
  }
}
```

Exemplo: Limitar o acesso a identidades com etiquetas que correspondam às etiquetas dos segredos

Uma estratégia é anexar etiquetas a segredos e a identidades do IAM. Em seguida, crie políticas de permissões para permitir operações quando a etiqueta da identidade corresponder à etiqueta do segredo. Para obter um tutorial completo, consulte Definir permissões para acessar segredos com base em etiquetas.

O uso de etiquetas para controlar permissões é útil em ambientes que estão crescendo rapidamente e ajuda em situações em que o gerenciamento de políticas se torna um problema. Para obter mais informações, consulte O que é ABAC para a AWS?

Example Permitir acesso a funções que têm as mesmas etiquetas que os segredos (anexar a um segredo)

A política a seguir só concederá `GetSecretValue` à conta *123456789012* se a etiqueta *AccessProject* tiver o mesmo valor para o segredo e para a função. Para usar essa política, consulte the section called “Anexar uma política de permissões a um segredo”.

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

Exemplo: entidade principal de serviço

Se a política de recursos anexada ao seu segredo incluir um diretor de AWS serviço, recomendamos que você use as chaves de condição SourceAccount globais aws: SourceArn e aws:. O ARN e os valores da conta só são incluídos no contexto de autorização quando uma solicitação chega ao Secrets Manager diretamente de outro serviço da AWS . Essa combinação de condições evita um potencial confused deputy scenario (cenário de substituto confuso).

Se um ARN de recurso incluir caracteres que não sejam permitidos em uma política de recurso, você não poderá usar esse ARN de recurso no valor da chave de condição `aws:SourceArn`. Em vez disso, use a chave de condição `aws:SourceAccount`. Para obter mais informações, consulte os requisitos do IAM.

Os diretores de serviços normalmente não são usados como diretores em uma política anexada a um segredo, mas alguns AWS serviços exigem isso. Para obter informações sobre as políticas de recursos que um serviço exige que você anexe a um segredo, consulte a documentação do serviço.

Example Permitir que um serviço acesse um segredo usando uma entidade principal de serviço (anexar a um segredo)

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

    }
  ]
}
```

Referência de permissões para AWS Secrets Manager

Para ver os elementos que compõem uma política de permissões, consulte Estrutura de documento de política JSON e Referência de elementos de política JSON do IAM.

Para começar a escrever sua própria política de permissões, consulte the section called “Exemplos de política de permissões”.

A coluna Tipos de recursos na tabela Ações indica se cada ação é compatível com permissões no nível do recurso. Se não houver valor para essa coluna, você deverá especificar todos os recursos

("*") aos quais a política se aplica no elemento `Resource` de sua declaração de política. Se a coluna incluir um tipo de recurso, você poderá especificar um ARN desse tipo em uma instrução com essa ação. Se a ação tiver um ou mais recursos necessários, o chamador deverá ter permissão para usar a ação com esses recursos. Os recursos obrigatórios são indicados na tabela com um asterisco (*). Se você limitar o acesso aos recursos com o elemento `Resource` em uma política do IAM, deverá incluir um ARN ou padrão para cada tipo de recurso necessário. Algumas ações oferecem suporte a vários tipos de recursos. Se o tipo de recurso for opcional (não indicado como obrigatório), você poderá optar por usar um dos tipos de recurso opcionais.

A coluna Chaves de condição na tabela Ações inclui chaves que você pode especificar em um elemento `Condition` da declaração de política. Para obter mais informações sobre as chaves de condição associadas aos recursos do serviço, consulte a coluna Chaves de condição da tabela Tipos de recursos.

Ações do Secrets Manager

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
BatchGetSecretValue	Concede permissão para recuperar e descriptografar uma lista de segredos	Listar			
CancelRotateSecret	Concede permissão para cancelar uma alternância do segredo em andamento	Escrever	Secret*	secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
				secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
CreateSecret	Concede permissão para criar um segredo que armazena dados criptografados que podem ser consultados e roteados	Escrever	Secret*	secretsmanager:Name secretsmanager:Description secretsmanager:KmsKeyId aws:RequestTag/${TagKey} aws:ResourceTag/${TagKey} aws:TagKeys secretsmanager:ResourceTag/tag-key secretsmanager:AddReplicaRegions	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
				secretsmanager:ForceOverwriteReplicaSecret	
DeleteResourcePolicy	Concede permissão para excluir a política de recursos anexada a um segredo	Gerenciamento de permissões	Secret*	secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
DeleteSecret	Concede permissão para excluir um segredo	Escrever	Secret*		

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:RecoveryWindowInDays secretsmanager:ForceDeleteWithoutRecovery secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:Sec	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
				retPrimaryRegion	
DescribeSecret	Concede permissão para recuperar os metadados sobre um segredo, mas não os dados criptografados	Leitura	Secret*		
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
GetRandomPassword	Concede permissão para gerar uma string aleatória para uso na criação de senha	Leitura			

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
GetResourcePolicy	Concede permissão para obter a política de recursos anexada a um segredo	Leitura	Secret*		
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
GetSecretValue	Concede permissão para recuperar e descriptografar os dados criptografados	Leitura	Secret*		

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
				secretsmanager:SecretId secretsmanager:VersionId secretsmanager:VersionStage secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
ListSecretVersionIds	Concede permissão para listar as versões disponíveis de um segredo	Leitura	Secret*		
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
ListSecrets	Concede permissão para listar os segredos disponíveis	Listar			
PutResourcePolicy	Concede permissão para anexar uma política de recursos a um segredo	Gerenciamento de permissões	Secret*		

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:BlockPublicPolicy secretsmanager:SecretPrimaryRegion	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
PutSecretValue	Concede permissão para criar uma nova versão do segredo com novos dados criptografados	Escrever	Secret*		
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
RemoveRegionsFromReplication	Concede permissão para remover regiões da replicação	Escrever	Secret*		

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
ReplicateSecretToRegions	Concede permissão para converter um segredo existente em um segredo multirregional e começar a replicar o segredo em uma lista de novas regiões	Escrever	Secret*	secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion secretsmanager:AddReplicaRegions secretsmanager:For	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
				ceOverwriteReplicaSecret	
RestoreSecret	Concede permissão para cancelar a exclusão de um segredo	Escrever	Secret*		
				secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	
RotateSecret	Concede permissão para iniciar a alternância de um segredo	Escrever	Secret*		

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
				secretsmanager:SecretId secretsmanager:RotationLambdaARN secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion secretsmanager:Mod	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
				ifyRotationRules secretsmanager:RotateImmediately	
StopReplicationToReplica	Concede permissão para remover o segredo da replicação e promove o segredo para um segredo regional na Região da réplica	Escrever	Secret*	secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
TagResource	Concede permissão para adicionar etiquetas a um segredo	Tags	Secret*	secretsmanager:SecretId aws:RequestTag/${TagKey} aws:TagKeys secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
UntagResource	Concede permissão para remover etiquetas de uma segredo	Tags	Secret*	secretsmanager:SecretId aws:TagKeys secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
UpdateSecret	Concede permissão para atualizar um segredo com novos metadados ou com uma nova versão dos dados criptografados	Escrever	Secret*	secretsmanager:SecretId secretsmanager:Description secretsmanager:KmsKeyId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:Sec	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
				retPrimaryRegion	
UpdateSecretVersionStage	Concede permissão para mover um estágio de um segredo para outro	Escrever	Secret*	secretsmanager:SecretId secretsmanager:VersionStage secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	

Ações	Descrição	Nível de acesso	Tipos de recursos (*necessários)	Chaves de condição	Ações dependentes
ValidateResourcePolicy	Concede permissão para validar uma política de recursos antes de anexar uma política	Gerenciamento de permissões	Secret*	secretsmanager:SecretId secretsmanager:resource/AllowRotationLambdaArn secretsmanager:ResourceTag/tag-key aws:ResourceTag/${TagKey} secretsmanager:SecretPrimaryRegion	

Recursos do Secrets Manager

Tipos de recursos	ARN	Chaves de condição
Secret	`arn:${Partition}:secretsmanager:${Region}:${Account}:secret:${SecretId}`	aws:RequestTag/${TagKey} aws:ResourceTag/${TagKey} aws:TagKeys secretsmanager:ResourceTag/tag-key secretsmanager:resource/AllowRotationLambdaArn

O Secrets Manager cria a última parte do ARN do segredo acrescentando um hífen e seis caracteres alfanuméricos aleatórios ao final do nome do segredo. Se você excluir um segredo e recriar outro com o mesmo nome, essa formatação ajudará a garantir que as pessoas com permissões para o segredo original não tenham acesso automaticamente ao novo segredo, pois o Secrets Manager gera seis novos caracteres aleatórios.

Você pode encontrar o ARN do segredo no console do Secrets Manager na página de detalhes do segredo ou chamando `DescribeSecret`.

Chaves de condição

Se você incluir alguma condição de string da tabela a seguir na sua política de permissões, os chamadores do Secrets Manager deverão transmitir o parâmetro correspondente ou o acesso será negado. Para evitar que os chamadores sejam negados em função de um parâmetro ausente, adicione `IfExists` ao final do nome do operador da condição, por exemplo `StringLikeIfExists`. Para obter mais informações, consulte Elementos da política JSON do IAM: operadores de condição.

Chaves de condição	Descrição	Tipo
aws:RequestTag/${TagKey}	Filtra o acesso por uma chave que está presente na solicitação feita pelo usuário para o serviço Secrets Manager	String
aws:ResourceTag/${TagKey}	Filtra o acesso pelas etiquetas associadas ao recurso	String
aws:TagKeys	Filtra o acesso por uma lista de todos os nomes de chaves de etiquetas presentes na solicitação feita pelo usuário para o serviço Secrets Manager	ArrayOfString
secretsmanager:AddReplicaRegions	Filtra o acesso pela lista de Regiões nas quais replicar o segredo	ArrayOfString
secretsmanager:BlockPublicPolicy	Filtra o acesso de acordo com o fato de a política de recursos bloquear o Conta da AWS acesso amplo	Bool
secretsmanager:Description	Filtra o acesso pelo texto da descrição na solicitação	String
secretsmanager:ForceDeleteWithoutRecovery	Filtra o acesso pela necessidade do segredo ser excluído imediatamente sem uma janela de recuperação	Bool
secretsmanager:ForceOverwriteReplicaSecret	Filtra o acesso por sobrescrever um segredo com o mesmo nome na Região de destino	Bool

Chaves de condição	Descrição	Tipo
secretsmanager:KmsKeyId	Filtra o acesso pelo ARN da chave do KMS na solicitação	String
secretsmanager:ModifyRotationRules	Filtra o acesso de acordo com a necessidade de as regras de alternância do segredo serem modificadas	Bool
secretsmanager:Name	Filtra o acesso pelo nome amigável do segredo na solicitação	String
secretsmanager:RecoveryWindowInDays	Filtra o acesso pelo número de dias que o Secrets Manager aguarda antes de excluir o segredo	Numérico
secretsmanager:ResourceTag/tag-key	Filtra o acesso por uma chave da etiqueta e par de valores	String
secretsmanager:RotateImmediately	Filtra o acesso pela necessidade do segredo ser girado imediatamente	Bool
secretsmanager:RotationLambdaARN	Filtra o acesso pelo ARN da função de alternância do Lambda na solicitação	ARN
secretsmanager:SecretId	Filtra o acesso pelo valor de SecretID na solicitação	ARN
secretsmanager:SecretPrimaryRegion	Filtra o acesso pela região primária na qual o segredo é criado	String

Chaves de condição	Descrição	Tipo
secretsmanager:VersionId	Filtra o acesso pelo identificador exclusivo da versão do segredo na solicitação	String
secretsmanager:VersionStage	Filtra o acesso pela lista de estágios de versão na solicitação	String
secretsmanager:resource/AllowRotationLambdaArn	Filtra o acesso pelo ARN da função de alternância do Lambda associada ao segredo	ARN

Bloqueie o acesso aos segredos com a condição **`BlockPublicPolicy`**

Em políticas de identidade que permitem a ação `PutResourcePolicy`, recomendamos usar `BlockPublicPolicy: true`. Essa condição significa que os usuários só poderão anexar uma política de recursos a um segredo se a política não permitir acesso amplo.

O Secrets Manager usa o raciocínio automatizado Zelkova para analisar políticas de recursos para acesso amplo. Para obter mais informações sobre Zelkova, consulte Como AWS usa o raciocínio automatizado para ajudar você a obter segurança em grande escala no AWS Blog de Segurança.

O exemplo a seguir mostra como usar `BlockPublicPolicy`.

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

```
}
```

Condições de endereço IP

Tome cuidado ao especificar os operadores de condição do endereço IP ou a chave de condição `aws:SourceIp` em uma declaração de política que permite ou nega acesso ao Secrets Manager. Por exemplo, se você anexar uma política que restringe AWS ações às solicitações do intervalo de endereços IP da sua rede corporativa a um segredo, suas solicitações como usuário do IAM que invoca a solicitação da rede corporativa funcionarão conforme o esperado. No entanto, se você permitir que outros serviços acessem o segredo em seu nome, como ao ativar a rotação com uma função Lambda, essa função chamará as operações do Secrets Manager a partir de um espaço AWS de endereço interno. As solicitações afetadas pela política com o filtro de endereços IP falharão.

Além disso, a chave de condição `aws:sourceIP` é menos efetiva quando a solicitação se origina em um endpoint da Amazon VPC. Para restringir solicitações a um endpoint da VPC específico, use the section called “Condições do endpoint da VPC”.

Condições do endpoint da VPC

Para conceder ou negar acesso a solicitações de uma determinada VPC ou de um endpoint da VPC, use `aws:SourceVpc` para limitar o acesso a solicitações da VPC especificada ou `aws:SourceVpce` para limitar o acesso a solicitações do endpoint da VPC especificado. Consulte the section called “Exemplo: Permissões e VPCs”.

- `aws:SourceVpc` limita o acesso a solicitações da VPC especificado.
- `aws:SourceVpce` limita o acesso a solicitações do endpoint da VPC especificado.

Se você usar essas chaves de condição em uma declaração de política de recurso que permite ou nega o acesso a segredos do Secrets Manager, poderá inadvertidamente negar acesso a serviços que usam o Secrets Manager para acessar segredos em seu nome. Somente alguns AWS serviços podem ser executados com um endpoint em sua VPC. Se você restringir as solicitações de um segredo a uma VPC ou a um endpoint da VPC, poderão ocorrer falhas nas chamadas para o Secrets Manager de um serviço não configurado para o serviço.

Consulte Endpoint da VPC.

Crie e gerencie segredos com AWS Secrets Manager

Um segredo pode ser uma senha, um conjunto de credenciais, como um nome de usuário e senha, um token OAuth ou outras informações secretas que você armazena em formato criptografado no Secrets Manager.

Tópicos

Crie um segredo AWS Secrets Manager de banco de dados

Após criar um usuário no Amazon RDS, Amazon Aurora, Amazon Redshift ou Amazon DocumentDB, é possível armazenar as credenciais no Secrets Manager seguindo estas etapas. Ao usar o AWS CLI ou um dos SDKs para armazenar o segredo, você deve fornecer o segredo na estrutura JSON correta. Quando você usa o console para armazenar um segredo de banco de dados, o Secrets Manager o cria automaticamente na estrutura JSON correta.

> **Tip**
>
> Para credenciais de usuário administrador do Amazon RDS e do Amazon Redshift, recomendamos que você use segredos gerenciados. Você cria o segredo gerenciado por meio do serviço de gerenciamento e, em seguida, pode usar a rotação gerenciada.

Quando você armazena credenciais de banco de dados para um banco de dados de origem replicado para outras regiões, o segredo contém informações de conexão para o banco de dados de origem. Se você replicar o segredo, as réplicas serão cópias do segredo de origem e conterão as mesmas informações de conexão. É possível adicionar pares de chave-valor ao segredo para informações de conexão regional.

Para criar um segredo, você precisa das permissões concedidas pelo SecretsManagerReadWriteAWS políticas gerenciadas.

O Secrets Manager gera uma entrada de CloudTrail registro quando você cria um segredo. Para ter mais informações, consulte the section called “Faça login com AWS CloudTrail ”.

Para criar um segredo (console)

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Selecione Armazenar um novo segredo.
3. Na página Choose secret type (Selecionar tipo de segredo), faça o seguinte:
   a. Em Secret type (Tipo de segredo), escolha o tipo de credenciais de banco de dados a armazenar:
      - Amazon RDS database (Banco de dados do Amazon RDS, inclui Aurora)
      - Amazon DocumentDB database (Bancos de dados do Amazon DocumentDB)
      - Armazém de dados do Amazon Redshift
   b. Em Credentials (Credenciais), insira as credenciais do banco de dados.
   c. Em Chave de criptografia, escolha a AWS KMS key que o Secrets Manager usa para criptografar o valor secreto. Para ter mais informações, consulte Criptografia e descriptografia de segredos.
      - Para a maioria dos casos, escolha aws/secretsmanager para usar a Chave gerenciada pela AWS para o Secrets Manager. Não há custo para o uso dessa chave.
      - Se você precisar acessar o segredo de outra pessoa Conta da AWS ou se quiser usar sua própria chave KMS para poder alterná-la ou aplicar uma política de chaves a ela, escolha uma chave gerenciada pelo cliente na lista ou escolha Adicionar nova chave para criar uma. Para obter mais informações sobre os custos do uso de uma chave gerenciada pelo cliente, consulte Definição de preço.

É necessário ter the section called “Permissões para a chave do KMS”. Para obter informações sobre o acesso entre contas, consulte the section called “Acesso entre contas”.

d. Em Database (Banco de dados), escolha seu banco de dados.

e. Escolha Próximo.

4. Na página Configure secret (Configurar segredo), faça o seguinte:

a. Insira um Secret name (Nome de segredo) descritivo e uma Description (Descrição). Os nomes de segredos devem conter de 1 a 512 caracteres Unicode.

b. (Opcional) Na seção Tags (Etiquetas), adicione etiquetas ao segredo. Para conhecer as estratégias de marcação, consulte the section called “Marcação de segredos do ”. Não armazene informações sigilosas em etiquetas porque elas não são criptografadas.

c. (Opcional) Em Resource permissions (Permissões do recurso), para adicionar uma política de recursos ao segredo, escolha Edit permissions (Editar permissões). Para ter mais informações, consulte the section called “Anexar uma política de permissões a um segredo”.

d. (Opcional) Em Replicar segredo, para replicar seu segredo para outro Região da AWS, escolha Replicar segredo. Você pode replicar seu segredo agora ou voltar e replicá-lo mais tarde. Para ter mais informações, consulte Replique segredos em todas as regiões.

e. Escolha Próximo.

5. (Opcional) Na página Configure rotation (Configurar alternância), habilite alternância automática para os segredos. Você também pode manter a alternância desabilitada por enquanto e habilitá-la mais tarde. Para ter mais informações, consulte Alternar segredos . Escolha Próximo.

6. Na página Review (Revisar), revise os detalhes do segredo e escolha Store (Armazenar).

O Secrets Manager retorna para a lista de segredos. Se o segredo não aparecer, escolha Refresh (Atualizar).

AWS CLI

Quando você insere comandos em um shell de comando, existe o risco de o histórico de comandos ser acessado ou de utilitários terem acesso aos seus parâmetros de comando. Consulte the section called “Mitigação de riscos do uso da AWS CLI para armazenar segredos do AWS Secrets Manager”.

Example Criar um segredo com base nas credenciais em um arquivo JSON

O exemplo de `create-secret` a seguir cria um segredo com base em credenciais em um arquivo. Para obter mais informações, consulte Carregando AWS CLI parâmetros de um arquivo no Guia AWS CLI do usuário.

Para que o Secrets Manager possa alternar o segredo, você deve se certificar de que o JSON corresponde a Estrutura JSON de um segredo.

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

Conteúdo de mycreds.json:

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

AWS SDK

Para criar um segredo usando um dos AWS SDKs, use a `CreateSecret`ação. Para ter mais informações, consulte the section called “AWS SDKs”.

Estrutura JSON de segredos AWS Secrets Manager

Você pode armazenar qualquer texto ou binário nos segredos do Secrets Manager. Se você quiser ativar a alternância automática para um segredo do Secrets Manager, ele deve estar na estrutura JSON correta. Durante a alternância, o Secrets Manager usa as informações no segredo para se conectar à origem da credencial e atualizar as credenciais nela. Os nomes das chaves JSON diferenciam maiúsculas de minúsculas.

Observe que ao usar o console para armazenar um segredo de banco de dados, o Secrets Manager o cria automaticamente na estrutura JSON correta.

É possível adicionar mais pares de chave/valor a um segredo (p. ex., a um segredo de banco de dados), para conter informações de conexão para bancos de dados de réplica em outras regiões.

Tópicos

Estrutura de segredos do Db2 do Amazon RDS

Para instâncias do Db2 do Amazon RDS, como os usuários não podem alterar suas próprias senhas, você deve fornecer credenciais de administrador em outro segredo.

```
{
  "engine": "db2",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 3306>,
  "masterarn": "<the ARN of the elevated secret>"
}
```

Estrutura do segredo MariaDB do Amazon RDS

```
{
  "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>
}
```

Para usar othe section called “Usuários alternados”, você inclui o `masterarn` para o segredo que contém credenciais de administrador ou superusuário.

```
{
  "engine": "mariadb",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 3306>,
  "masterarn": "<the ARN of the elevated secret>"
}
```

Estrutura de segredos do Amazon RDS e do Amazon Aurora 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>
}
```

Para usar othe section called “Usuários alternados”, você inclui o `masterarn` para o segredo que contém credenciais de administrador ou superusuário.

```
{
  "engine": "mysql",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 3306>,
  "masterarn": "<the ARN of the elevated secret>"
```

```
}
```

Estrutura do segredo do Oracle do Amazon RDS

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

Para usar othe section called “Usuários alternados”, você inclui o `masterarn` para o segredo que contém credenciais de administrador ou superusuário.

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

Estrutura de segredos do Amazon RDS e do Amazon Aurora PostgreSQL

```
{
  "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>
}
```

Para usar othe section called “Usuários alternados”, você inclui o `masterarn` para o segredo que contém credenciais de administrador ou superusuário.

```
{
```

```
  "engine": "postgres",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to 'postgres'>",
  "port": <TCP port number. If not specified, defaults to 5432>,
  "masterarn": "<the ARN of the elevated secret>"
}
```

Estrutura do segredo do Microsoft SQLServer do Amazon RDS

```
{
  "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>
}
```

Para usar othe section called “Usuários alternados”, você inclui o `masterarn` para o segredo que contém credenciais de administrador ou superusuário.

```
{
  "engine": "sqlserver",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to 'master'>",
  "port": <TCP port number. If not specified, defaults to 1433>,
  "masterarn": "<the ARN of the elevated secret>"
}
```

Estrutura do segredo do Amazon DocumentDB

```
{
  "engine": "mongo",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
```

```
  "port": <TCP port number. If not specified, defaults to 27017>,
  "ssl": <true|false. If not specified, defaults to false>
}
```

Para usar othe section called “Usuários alternados”, você inclui o `masterarn` para o segredo que contém credenciais de administrador ou superusuário.

```
{
  "engine": "mongo",
  "host": "<instance host name/resolvable DNS name>",
  "username": "<username>",
  "password": "<password>",
  "dbname": "<database name. If not specified, defaults to None>",
  "port": <TCP port number. If not specified, defaults to 27017>,
  "masterarn": "<the ARN of the elevated secret>",
  "ssl": <true|false. If not specified, defaults to false>
}
```

Estrutura do segredo do Amazon Redshift

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

Para usar othe section called “Usuários alternados”, você inclui o `masterarn` para o segredo que contém credenciais de administrador ou superusuário.

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

Estrutura secreta sem servidor do Amazon Redshift

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

Para usar othe section called “Usuários alternados”, você inclui o `masterarn` para o segredo que contém credenciais de administrador ou superusuário.

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

Estrutura ElastiCache secreta da Amazon

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

Para obter mais informações, consulte Rotação automática de senhas para usuários no Guia do ElastiCache usuário da Amazon.

Estruturas secretas do Active Directory

AWS Directory Service usa segredos para armazenar credenciais do Active Directory. Para obter mais informações, consulte Unir perfeitamente uma instância Linux do Amazon EC2 ao seu AD

Active Directory gerenciado AWS Directory Service no Guia de Administração. A associação perfeita ao domínio requer os nomes de chave nos exemplos a seguir. Se você não usar a junção de domínio perfeita, poderá alterar os nomes das chaves no segredo usando variáveis de ambiente, conforme descrito no código do modelo da função de rotação.

Para girar segredos do Active Directory, você pode usar os modelos de rotação do Active Directory.

Estrutura secreta de credenciais do Active Directory

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

Se você quiser alternar o segredo, inclua o ID do diretório do domínio.

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

Se o segredo for usado em conjunto com um segredo que contém um keytab, você inclui os ARNs secretos do keytab.

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

Estrutura secreta do Active Directory keytab

Para obter informações sobre o uso de arquivos keytab para autenticação em contas do Active Directory no Amazon EC2, consulte Implantação e configuração da autenticação do Active Directory com o SQL Server 2017 no 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
}
```

Crie um AWS Secrets Manager segredo

Para armazenar chaves de API, tokens de acesso, credenciais que não sejam para bancos de dados e outros segredos no Secrets Manager, siga estas etapas. Para um ElastiCache segredo da Amazon, se você quiser ativar a rotação, deverá armazenar o segredo na estrutura JSON esperada.

Para criar um segredo, você precisa das permissões concedidas pelo SecretsManagerReadWriteAWS políticas gerenciadas.

O Secrets Manager gera uma entrada de CloudTrail registro quando você cria um segredo. Para ter mais informações, consulte the section called “Faça login com AWS CloudTrail ”.

Para criar um segredo (console)

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Selecione Armazenar um novo segredo.
3. Na página Choose secret type (Selecionar tipo de segredo), faça o seguinte:

    a. Em Secret type (Tipo de segredo), escolha Other type of secret (Outro tipo de segredo).

b. Em Key/value pairs (Pares de chave/valor), ou insira seu segredo no JSON Key/value (Chave/valor), ou escolha a guia Plaintext (Texto simples) e insira o segredo em qualquer formato. É possível armazenar até 65536 bytes no segredo. Alguns exemplos:

Os pares de chave-valor da API:

ClientID: ***my_client_id***

ClientSecret : ***WJALRXUTNFEMI/K7MDENG/CYEXAMPLEKEY bPxRfi***

Credenciais de pares de chave-valor:

Username: *saanvis*

Password: *EXAMPLE-PASSWORD*

Texto não criptografado do token OAuth:

AKIAI44QH8DHBEXAMPLE

Texto sem formatação do certificado digital:

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

Chave privada em texto não criptografado:

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

c. Em Chave de criptografia, escolha a AWS KMS key que o Secrets Manager usa para criptografar o valor secreto. Para ter mais informações, consulte Criptografia e descriptografia de segredos.

- Para a maioria dos casos, escolha aws/secretsmanager para usar a Chave gerenciada pela AWS para o Secrets Manager. Não há custo para o uso dessa chave.
- Se você precisar acessar o segredo de outra pessoa Conta da AWS ou se quiser usar sua própria chave KMS para poder alterná-la ou aplicar uma política de chaves a ela,

escolha uma chave gerenciada pelo cliente na lista ou escolha Adicionar nova chave para criar uma. Para obter mais informações sobre os custos do uso de uma chave gerenciada pelo cliente, consulte Definição de preço.

É necessário ter the section called “Permissões para a chave do KMS”. Para obter informações sobre o acesso entre contas, consulte the section called “Acesso entre contas”.

d. Escolha Próximo.

4. Na página Configure secret (Configurar segredo), faça o seguinte:

   a. Insira um Secret name (Nome de segredo) descritivo e uma Description (Descrição). Os nomes de segredos devem conter de 1 a 512 caracteres Unicode.

   b. (Opcional) Na seção Tags (Etiquetas), adicione etiquetas ao segredo. Para conhecer as estratégias de marcação, consulte the section called “Marcação de segredos do ”. Não armazene informações sigilosas em etiquetas porque elas não são criptografadas.

   c. (Opcional) Em Resource permissions (Permissões do recurso), para adicionar uma política de recursos ao segredo, escolha Edit permissions (Editar permissões). Para ter mais informações, consulte the section called “Anexar uma política de permissões a um segredo”.

   d. (Opcional) Em Replicar segredo, para replicar seu segredo para outro Região da AWS, escolha Replicar segredo. Você pode replicar seu segredo agora ou voltar e replicá-lo mais tarde. Para ter mais informações, consulte Replique segredos em todas as regiões.

   e. Escolha Próximo.

5. (Opcional) Na página Configure rotation (Configurar alternância), habilite alternância automática para os segredos. Você também pode manter a alternância desabilitada por enquanto e habilitá-la mais tarde. Para ter mais informações, consulte Alternar segredos . Escolha Próximo.

6. Na página Review (Revisar), revise os detalhes do segredo e escolha Store (Armazenar).

   O Secrets Manager retorna para a lista de segredos. Se o segredo não aparecer, escolha Refresh (Atualizar).

AWS CLI

Quando você insere comandos em um shell de comando, existe o risco de o histórico de comandos ser acessado ou de utilitários terem acesso aos seus parâmetros de comando. Consulte the section called “Mitigação de riscos do uso da AWS CLI para armazenar segredos do AWS Secrets Manager”.

Example Criar um segredo

O seguinte exemplo de `create-secret` cria um segredo com dois pares de chave/valor.

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

Example Criar um segredo com base nas credenciais em um arquivo JSON

O exemplo de `create-secret` a seguir cria um segredo com base em credenciais em um arquivo. Para obter mais informações, consulte Carregando AWS CLI parâmetros de um arquivo no Guia AWS CLI do usuário.

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

Conteúdo de mycreds.json:

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

AWS SDK

Para criar um segredo usando um dos AWS SDKs, use a `CreateSecret`ação. Para ter mais informações, consulte the section called “AWS SDKs”.

Atualize o valor de um AWS Secrets Manager segredo

Para atualizar o valor da sua chave secreta, você pode usar o console, a CLI ou um SDK. Quando você atualiza o valor do segredo, o Secrets Manager cria uma nova versão do segredo com uma rótulo de preparação `AWSCURRENT`. Você ainda pode acessar a versão antiga, que tem o rótulo da `AWSPREVIOUS`. Você também pode adicionar seus próprios rótulos. Para obter mais informações, consulte Secrets Manager versioning (Versionamento do Secrets Manager).

Para atualizar o valor do segredo (console)

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na lista de segredos, escolha o segredo.
3. Na página de detalhes do segredo, na guia Visão geral, na seção Valor do segredo, escolha Recuperar o valor do segredo e, em seguida, Editar.

AWS CLI

Para atualizar o valor do segredo (AWS CLI)

- Quando você insere comandos em um shell de comando, existe o risco de o histórico de comandos ser acessado ou de utilitários terem acesso aos seus parâmetros de comando. Consulte the section called “Mitigação de riscos do uso da AWS CLI para armazenar segredos do AWS Secrets Manager”.

  O seguinte `put-secret-value` cria uma nova versão de um segredo com dois pares de chave/valor.

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

  O seguinte `put-secret-value` cria uma nova versão com um rótulo de teste personalizado. A nova versão terá os rótulos `MyLabel` e `AWSCURRENT`.

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

AWS SDK

Recomendamos que você evite chamar `PutSecretValue` ou `UpdateSecret` em uma frequência sustentada de mais de uma vez a cada dez minutos. Quando você chama `PutSecretValue` ou `UpdateSecret` para atualizar o valor do segredo, o Secrets Manager cria uma nova versão do segredo. O Secrets Manager remove versões sem rótulo quando há mais de 100, mas não remove

versões criadas há menos de 24 horas. Se você atualizar o valor do segredo mais de uma vez a cada dez minutos, criará mais versões do que o Secrets Manager remove e atingirá a cota de versões de segredos.

Para atualizar o valor de um segredo, utilize as seguintes ações: `UpdateSecret` ou `PutSecretValue`. Para obter mais informações, consulte the section called “AWS SDKs”.

Gere uma senha com o Secrets Manager

Um padrão comum para usar o Secrets Manager é gerar uma senha no Secrets Manager e depois usar essa senha em seu banco de dados ou serviço. Você pode fazer isso usando os seguintes métodos:

- AWS CloudFormation — VejaAWS CloudFormation.
- AWS CLI — Veja `get-random-password`.
- AWS SDKs — Veja `GetRandomPassword`.

Reverter um segredo para uma versão anterior

Você pode reverter um segredo para uma versão anterior movendo os rótulos anexados às versões secretas usando o. AWS CLI Para obter informações sobre como o Secrets Manager armazena versões de segredos, consultethe section called “Versões secretas”.

O `update-secret-version-stage`exemplo a seguir move o rótulo AWSCURRENT de teste para a versão anterior de um segredo, o que reverte o segredo para a versão anterior. Para encontrar o ID da versão anterior, use `list-secret-version-ids`ou visualize as versões no console do Secrets Manager.

Neste exemplo, a versão com o rótulo é A1B2C3D4-5678-90AB-CDEF-Example11111 e a versão com o AWSCURRENT rótulo é A1B2C3D4-5678-90AB-CDEF-Example22222. AWSPREVIOUS Neste exemplo, você move o AWSCURRENT rótulo da versão 11111 para 22222. Como o AWSCURRENT rótulo é removido de uma versão, move `update-secret-version-stage` automaticamente o AWSPREVIOUS rótulo para essa versão (11111). O efeito é que as AWSPREVIOUS versões AWSCURRENT e são trocadas.

```
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
```

Troque a chave de criptografia por um AWS Secrets Manager segredo

O Secrets Manager usa criptografia de envelope com AWS KMS chaves e chaves de dados para proteger cada valor secreto. Para cada segredo, você pode escolher qual chave KMS usar. Você pode usar o Chave gerenciada pela AWS aws/secretsmanager ou usar uma chave gerenciada pelo cliente. Na maioria dos casos, recomendamos usar aws/secretsmanager, e não há custo para usá-la. Se você precisar acessar o segredo de outra pessoa Conta da AWS ou se quiser usar sua própria chave KMS para poder alterná-la ou aplicar uma política de chaves a ela, use a. chave gerenciada pelo clienteÉ necessário ter the section called “Permissões para a chave do KMS”. Para obter mais informações sobre os custos do uso de uma chave gerenciada pelo cliente, consulte Definição de preço.

Você pode alterar a chave de criptografia de um segredo. Por exemplo, se você quiser acessar o segredo de outra conta e o segredo estiver atualmente criptografado usando a chave AWS gerenciadaaws/secretsmanager, você pode mudar para uma chave gerenciada pelo cliente.

> **Tip**
>
> Se você quiser girar seu chave gerenciada pelo cliente, recomendamos usar a rotação AWS KMS automática de chaves. Para obter mais informações, consulte AWS KMS Chaves giratórias.

Quando você altera a chave de criptografia, o Secrets Manager AWSCURRENT criptografa novamente e AWSPENDING cria AWSPREVIOUS versões com a nova chave. Para evitar que você fique sem o segredo, o Secrets Manager mantém todas as versões existentes criptografadas com a chave anterior. Isso significa que você pode descriptografar AWSCURRENTAWSPENDING, e AWSPREVIOUS versões com a chave anterior ou a nova chave.

Para que isso só AWSCURRENT possa ser descriptografado pela nova chave de criptografia, crie uma nova versão do segredo com a nova chave. Então, para poder decifrar a versão AWSCURRENT secreta, você deve ter permissão para a nova chave.

Se você desativar a chave de criptografia anterior, não poderá descriptografar nenhuma versão secreta, exceto `AWSCURRENT`, `AWSPENDING` e `AWSPREVIOUS`. Se você tiver outras versões secretas rotuladas às quais deseja manter o acesso, precisará recriar essas versões com a nova chave de criptografia usando o the section called “AWS CLI”.

Para alterar a chave de criptografia de um segredo (console)

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na lista de segredos, escolha o segredo.
3. Na página de detalhes do segredo, na seção Secrets details (Detalhes dos segredos), selecione Actions (Ações) e, em seguida, selecione Edit encryption key (Editar chave de criptografia).

AWS CLI

Se você alterar a chave de criptografia de um segredo e, em seguida, desativar a chave de criptografia anterior, você não conseguirá descriptografar nenhuma versão do segredo, exceto `AWSCURRENT`, `AWSPENDING` e `AWSPREVIOUS`. Se você tiver outras versões secretas rotuladas às quais deseja manter o acesso, precisará recriar essas versões com a nova chave de criptografia usando o the section called “AWS CLI”.

Para alterar a chave de criptografia de um segredo, consulte (AWS CLI)

1. O exemplo de `update-secret` a seguir atualiza a chave do KMS usada para criptografar o valor secreto. A chave do KMS precisa estar na mesma do segredo.

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

2. (Opcional) Se você tiver versões de segredos com rótulos personalizados, para recriptografá-las usando a nova chave, será necessário recriar essas versões.

   Quando você insere comandos em um shell de comando, existe o risco de o histórico de comandos ser acessado ou de utilitários terem acesso aos seus parâmetros de comando. Consulte the section called “Mitigação de riscos do uso da AWS CLI para armazenar segredos do AWS Secrets Manager”.

   a. Obtenha o valor da versão secreta.

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

Anote o valor do segredo.

b. Crie uma nova versão com esse valor.

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

Modificar um AWS Secrets Manager segredo

Dependendo de quem criou o segredo, você pode modificar os metadados de um segredo depois que ele for criado. Para segredos criados por outros serviços, talvez seja necessário usar o outro serviço para atualizar ou alternar o segredo.

Para determinar quem gerencia um segredo, você pode revisar o nome do segredo. Os segredos gerenciados por outros serviços são prefixados com o ID do respectivo serviço. Ou, no AWS CLI, chame describe-secret e revise o campo. `OwningService` Para ter mais informações, consulte Segredos gerenciados.

Para os segredos que você gerencia, é possível modificar a descrição, a política baseada em recursos, a chave de criptografia e as etiquetas. Você também pode alterar o valor do segredo criptografado; entretanto, recomendamos usar alternância para atualizar os valores dos segredo que contêm credenciais. A alternância atualiza o segredo no Secrets Manager e as credenciais no banco de dados ou serviço. Isso mantém o segredo sincronizado automaticamente para que, quando solicitarem um valor de segredo, os clientes sempre obtenham um conjunto de credenciais que funcionam. Para ter mais informações, consulte Alternar segredos .

O Secrets Manager gera uma entrada de CloudTrail registro quando você modifica um segredo. Para ter mais informações, consulte the section called “Faça login com AWS CloudTrail ”.

Para atualizar um segredo que você gerencia (console)

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.

2. Na lista de segredos, escolha o segredo.
3. Na página de detalhes do segredo, siga um destes procedimentos:

   Observe que você não pode alterar o nome ou o ARN de um segredo.

   - Para atualizar a descrição, na seção Secrets details (Detalhes dos segredos), escolha Actions (Ações) e, em seguida, Edit description (Editar descrição).
   - Para atualizar a chave de criptografia, consulte the section called “Altere a chave de criptografia de um segredo”.
   - Para atualizar etiquetas, na guia Etiquetas, escolha Editar etiquetas. Consulte the section called “Marcação de segredos do ”.
   - Para atualizar o valor do segredo, consulte the section called “Atualize o valor do segredo”.
   - Para atualizar permissões do seu segredo, na guia Visão geral, escolha Editar permissões. Consulte the section called “Anexar uma política de permissões a um segredo”.

   - Para atualizar a alternância do seu segredo, na guia Alternância, escolha Editar alternância. Consulte Alternar segredos .
   - Para replicar seu segredo para outras regiões, consulte Replique segredos em todas as regiões.
   - Se seu segredo tiver réplicas, você pode trocar a chave de criptografia de uma réplica. Na guia Replicação, selecione o botão de opção para a réplica e, em seguida, no menu Ações, selecione Editar chave de criptografia. Consulte the section called “Criptografia e descriptografia de segredos”.
   - Para alterar um segredo de modo que ele seja gerenciado por outro serviço, você precisa recriar o segredo no respectivo serviço. Consulte Segredos gerenciados.

AWS CLI

Example Atualizar descrição do segredo

O exemplo de `update-secret` a seguir retorna a descrição de um segredo.

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

AWS SDK

Recomendamos que você evite chamar `PutSecretValue` ou `UpdateSecret` em uma frequência sustentada de mais de uma vez a cada dez minutos. Quando você chama `PutSecretValue` ou `UpdateSecret` para atualizar o valor do segredo, o Secrets Manager cria uma nova versão do segredo. O Secrets Manager remove versões sem rótulo quando há mais de 100, mas não remove versões criadas há menos de 24 horas. Se você atualizar o valor do segredo mais de uma vez a cada dez minutos, criará mais versões do que o Secrets Manager remove e atingirá a cota de versões de segredos.

Para atualizar um segredo, use as seguintes ações: `UpdateSecret` ou `ReplicateSecretToRegions`. Para ter mais informações, consulte the section called “AWS SDKs”.

Encontre segredos em AWS Secrets Manager

Quando você pesquisa por segredos sem um filtro, o Secrets Manager faz correspondências com base em palavras-chave com o nome do segredo, descrição, chave de etiqueta e valor da etiqueta. Pesquisar sem filtros não diferencia maiúsculas de minúsculas e ignora caracteres especiais, como espaço, /, _, =, #, e usa apenas números e letras. Quando você pesquisa sem um filtro, o Secrets Manager analisa a string de pesquisa para convertê-la em palavras separadas. As palavras são separadas por qualquer alteração de maiúsculas para minúsculas, de letra para número ou de número/letra para pontuação. Por exemplo, inserir o termo de pesquisa `credsDatabase#892` gera a pesquisa por `creds`, `Database` e `892` no nome, descrição e chave de tag e valor.

O Secrets Manager gera uma entrada de CloudTrail registro quando você lista segredos. Para ter mais informações, consulte the section called “Faça login com AWS CloudTrail ”.

É possível aplicar os seguintes filtros para sua pesquisa:

Nome

> Encontra correspondências com o início dos nomes de segredo; diferencia maiúsculas de minúsculas. Por exemplo, Name: **`Data`** retorna um segredo nomeado como `DatabaseSecret`, mas não `databaseSecret` ou `MyData`.

Descrição

Encontra correspondência nas palavras em descrições dos segredos, não diferencia letras maiúsculas de minúsculas. Por exemplo, Description: **`My Description`** encontra correspondências entre segredos com as seguintes descrições:

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

Gerenciado por

Encontra segredos gerenciados por serviços externos AWS, por exemplo, CyberArk ou HashiCorp.

Serviço de propriedade

Faz correspondências com o início do prefixo do ID do serviço de gerenciamento, não diferencia maiúsculas de minúsculas. Por exemplo, **`my-ser`** faz a correspondência com segredos gerenciados por serviços com os prefixos `my-serv` e `my-service`. Para ter mais informações, consulte Segredos gerenciados.

Segredos replicados

Você pode filtrar segredos primários, réplicas de segredo ou segredos que não estão replicados.

Chaves de etiqueta

Encontra correspondências com o início das chaves de etiqueta; diferencia maiúsculas de minúsculas. Por exemplo, Tag key: **`Prod`** retorna segredos com as etiquetas `Production` e `Prod1`, mas não segredos com as etiquetas `prod` ou `1 Prod`.

Valores de etiqueta

Encontra correspondências com o início dos valores de etiqueta; diferencia maiúsculas de minúsculas. Por exemplo, Tag value: **`Prod`** retorna segredos com as etiquetas `Production` e `Prod1`, mas não segredos com o valor de etiqueta `prod` ou `1 Prod`.

O Secrets Manager é um serviço regional e apenas segredos na região selecionada são retornados.

AWS CLI

Example Listar os segredos em sua conta

O exemplo de `list-secrets` a seguir mostra uma lista dos segredos em sua conta.

```
aws secretsmanager list-secrets
```

Example Filtrar a lista de segredos em sua conta

O exemplo de `list-secrets` a seguir obtém uma lista dos segredos em sua conta que têm Test no nome. A filtragem por nome diferencia maiúsculas de minúsculas.

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

Example Encontre segredos que são gerenciados por outros AWS serviços

O exemplo de `list-secrets` a seguir obtém uma lista de segredos gerenciados por um serviço. Você especifica o ID do serviço. Para ter mais informações, consulte Segredos gerenciados.

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

AWS SDK

Para encontrar segredos usando um dos AWS SDKs, use `ListSecrets`. Para ter mais informações, consulte the section called “AWS SDKs”.

Excluir um AWS Secrets Manager segredo

Devido à natureza crítica dos segredos, dificulta AWS Secrets Manager intencionalmente a exclusão de um segredo. O Secrets Manager não exclui segredos de imediato. Em vez disso, o Secrets Manager torna os segredos imediatamente inacessíveis e programados para exclusão após uma janela de recuperação de, no mínimo, sete dias. Até que a janela de recuperação termine, é possível recuperar um segredo excluído anteriormente. Não há cobrança para segredos que você marcou para exclusão.

Você não pode excluir um segredo primário se ele for replicado para outras regiões. Primeiramente, exclua as réplicas e, em seguida, exclua o segredo primário. Quando você exclui uma réplica, ela é excluída imediatamente.

Não é possível excluir diretamente uma versão de um segredo. Em vez disso, você remove todos os rótulos de teste da versão usando o AWS CLI ou AWS SDK. Isso marca a versão como defasada e permite que o Secrets Manager exclua automaticamente a versão em segundo plano.

Se você não sabe se um aplicativo ainda usa um segredo, você pode criar um CloudWatch alarme da Amazon para alertá-lo sobre qualquer tentativa de acessar um segredo durante a janela de recuperação. Para ter mais informações, consulte Monitore quando AWS Secrets Manager os segredos programados para exclusão são acessados.

Para excluir um segredo, você precisa ter permissões do `secretsmanager:ListSecrets` e do `secretsmanager:DeleteSecret`.

O Secrets Manager gera uma entrada de CloudTrail registro quando você exclui um segredo. Para ter mais informações, consulte the section called “Faça login com AWS CloudTrail ”.

Para excluir um segredo (console)

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na lista de segredos, escolha o segredo que você quer excluir.
3. Na seção Secrets details (Detalhes dos segredos), escolha Actions (Ações) e, em seguida, escolha Delete secret (Excluir segredo).
4. Na caixa de diálogo Disable secret and schedule deletion (Desabilitar segredo e programar exclusão), em Waiting period (Período de espera), insira o número de dias de espera antes que a exclusão se torne permanente. O Secrets Manager anexa um campo denominado `DeletionDate` e o define como a data e hora atual e soma o número de dias especificado para a janela de recuperação.
5. Escolha Schedule deletion.

Para visualizar segredos excluídos

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na página Secrets (Segredos), escolha Preferences (Preferências) ( ).

3. Na caixa de diálogo Preferências, selecione Mostrar segredos programados para exclusão e, em seguida, escolha Salvar.

Para excluir um segredo de réplica

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Selecione o segredo primário.
3. Na seção Replicate Secret (Replicar segredo), escolha o segredo de réplica.
4. No menu Actions (Ações), escolha Delete Replica (Excluir réplica).

AWS CLI

Example Excluir um segredo

O exemplo de `delete-secret` a seguir exclui um segredo. Você pode recuperar o segredo `restore-secret`até a data e a hora no campo de DeletionDate resposta. Para excluir um segredo que está replicado em outras regiões, primeiro remova suas réplicas com `remove-regions-from-replication` e então chame `delete-secret`.

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

Example Excluir um segredo imediatamente

O exemplo de `delete-secret` a seguir exclui imediatamente um segredo sem uma janela de recuperação. Não é possível recuperar esse segredo.

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

Example Excluir um segredo de réplica

O exemplo de `remove-regions-from-replication` a seguir exclui um segredo de réplica em eu-west-3. Para excluir um segredo primário que está replicado em outras regiões, primeiro remova as réplicas e então chame `delete-secret`.

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

AWS SDK

Para excluir um segredo, use o comando `DeleteSecret`. Para excluir uma versão de um segredo, use o comando `UpdateSecretVersionStage`. Para excluir uma réplica, use o comando `StopReplicationToReplica`. Para ter mais informações, consulte the section called “AWS SDKs”.

Restaurar um AWS Secrets Manager segredo

O Secrets Manager considera um segredo programado para exclusão como defasado e não pode mais ser acessado diretamente. Depois que a janela de recuperação passar, o Secrets Manager excluirá o segredo permanentemente. Depois que o Secrets Manager apaga o segredo, não é possível recuperá-lo. Antes do final da janela de recuperação, você pode recuperar o segredo e torná-lo acessível novamente. Isso remove o campo `DeletionDate`, que cancela a exclusão permanente programada.

Para restaurar um segredo e os metadados no console, é necessário ter as permissões `secretsmanager:ListSecrets` e `secretsmanager:RestoreSecret`.

O Secrets Manager gera uma entrada de CloudTrail registro quando você restaura um segredo. Para ter mais informações, consulte the section called “Faça login com AWS CloudTrail ”.

Para restaurar um segredo (console)

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na lista de segredos, escolha o segredo que você quer restaurar.

   Se os segredos excluídos não constarem da lista de segredos, escolha Preferences (Preferências)
   ().
   Na caixa de diálogo Preferências, selecione Mostrar segredos programados para exclusão e, em seguida, escolha Salvar.
3. Na página Secret details (Detalhes do segredo), escolha Cancel deletion (Cancelar exclusão).

4. Na caixa de diálogo Cancel secret deletion (Cancelar exclusão do segredo), selecione Cancel deletion (Cancelar exclusão).

AWS CLI

Example Restaurar um segredo excluído anteriormente

O exemplo de `restore-secret` a seguir restaura um segredo que estava previamente programado para exclusão.

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

AWS SDK

Para restaurar um segredo marcado para exclusão, use o comando `RestoreSecret`. Para ter mais informações, consulte the section called “AWS SDKs”.

Marcação de segredos do AWS Secrets Manager

O Secrets Manager define uma etiqueta como um rótulo que consiste em uma chave definida por você e um valor opcional. É possível usar etiquetas para facilitar o gerenciamento, a pesquisa e a filtragem dos segredos e outros recursos em sua conta da AWS. Ao etiquetar seus segredos, use um esquema de nomenclatura padronizado em todos os seus recursos. Para obter mais informações, consulte o whitepaper Tagging Best Practices (Práticas recomendadas de marcação).

Você pode conceder ou negar acesso a um segredo ao marcar as etiquetas anexadas ao segredo. Para obter mais informações, consulte the section called “Exemplo: Controlar o acesso a segredos usando etiquetas”.

Você pode localizar segredos por etiquetas no console, na AWS CLI e nos SDKs. A AWS também fornece a ferramenta Resource Groups (Grupos de recursos) para a criação de um console personalizado que consolida e organiza seus recursos com base em suas etiquetas. Para encontrar segredos com uma etiqueta específica, consulte the section called “Localizar segredos”. O Secrets Manager não oferece suporte à alocação de custos baseada em etiquetas.

Nunca armazene informações confidenciais de um segredo em uma etiqueta.

Para cotas de tags e restrições de nomenclatura, consulte Service Quotas para marcação no Guia de referência geral da AWS. As tags diferenciam letras maiúsculas de minúsculas.

O Secrets Manager gera uma entrada de log do CloudTrail quando você marca ou desmarca um segredo. Para obter mais informações, consulte the section called “Faça login com AWS CloudTrail ”.

Para alterar as etiquetas do segredo (console)

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na lista de segredos, escolha o segredo.
3. Na página de detalhes do segredo, na guia Etiquetas, escolha Editar etiquetas. Os nomes e valores da chave de etiqueta diferenciam maiúsculas de minúsculas e as chaves de etiqueta devem ser exclusivas.

AWS CLI

Example Adicionar um tag a um segredo

O exemplo de `tag-resource` a seguir mostra como anexar um tag com uma sintaxe abreviada.

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

Example Adicionar várias tags a um segredo

O exemplo de `tag-resource` a seguir anexa duas tags de chave/valor a um segredo.

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

Example Remover tags de um segredo

O exemplo de `untag-resource` a seguir remove duas tags de um segredo. Para cada tag, tanto a chave quanto o valor são removidos.

```
aws secretsmanager untag-resource \
```

```
            --secret-id MyTestSecret \
            --tag-keys '[ "FirstTag", "SecondTag"]'
```

AWS SDK

Para alterar as etiquetas do seu segredo use `TagResource` ou `UntagResource`. Para obter mais informações, consulte the section called “AWS SDKs”.

Replique AWS Secrets Manager segredos em todas as regiões

Você pode replicar seus segredos em vários Regiões da AWS para oferecer suporte a aplicativos espalhados por essas regiões e atender aos requisitos regionais de acesso e baixa latência. Se precisar mais tarde, você pode promover uma réplica secreta para uma cópia autônoma e depois configurá-la para replicação independente. O Secrets Manager replica todos os dados de segredos e metadados criptografados, como etiquetas, políticas de recursos em todas as regiões especificadas.

O ARN de um segredo replicado é o mesmo do segredo primário, exceto pela região, por exemplo:

- Segredo primário:
  `arn:aws:secretsmanager:Region1:123456789012:secret:MySecret-a1b2c3`
- Segredo de réplica:
  `arn:aws:secretsmanager:Region2:123456789012:secret:MySecret-a1b2c3`

Para obter informações sobre preços de segredos de réplicas, consulte Definição de preços de AWS Secrets Manager.

Quando você armazena credenciais de banco de dados para um banco de dados de origem replicado para outras regiões, o segredo contém informações de conexão para o banco de dados de origem. Se você replicar o segredo, as réplicas serão cópias do segredo de origem e conterão as mesmas informações de conexão. É possível adicionar pares de chave-valor ao segredo para informações de conexão regional.

Se você habilitar a alternância para seu segredo primário, o Secrets Manager alternará o segredo na região primária e o novo valor do segredo se propagará para todos os segredos de réplica associados. Você não precisa gerenciar a alternância individualmente para todos os segredos de réplica.

Você pode replicar segredos em todas as AWS regiões habilitadas. No entanto, se você usar o Secrets Manager em AWS regiões especiais, como AWS GovCloud (US) regiões da China, só poderá configurar segredos e réplicas dentro dessas AWS regiões especializadas. Você não pode replicar um segredo em suas AWS regiões habilitadas para uma região especializada ou replicar segredos de uma região especializada para uma região comercial.

Antes que possa replicar um segredo para outra região, você deve habilitar essa região. Para obter mais informações, consulte Gerenciar regiões da AWS .

É possível usar um segredo em várias regiões sem replicá-lo chamando o endpoint do Secrets Manager na região onde o segredo está armazenado. Para uma lista de endpoints , consulte the section called “Endpoint do Secrets Manager”. Para usar a replicação para melhorar a resiliência de sua carga de trabalho, consulte Arquitetura de recuperação de desastres (DR) em AWS, Parte I: Estratégias para recuperação na nuvem.

O Secrets Manager gera uma entrada de CloudTrail registro quando você replica um segredo. Para ter mais informações, consulte the section called “Faça login com AWS CloudTrail ”.

Para replicar um segredo para outras regiões (console)

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na lista de segredos, escolha o segredo.
3. Na página de detalhes do segredo, na guia Replicação, siga um destes procedimentos:
   - Se seu segredo não for replicado, selecione Replicate secret (Replicar segredo).
   - Se seu segredo for replicado, na seção Replicate secret (Replicar segredo), selecione Add Region (Adicionar região).
4. Na caixa de diálogo Add replica regions (Adicionar regiões para replicação), faça o seguinte:
   a. Em Região da AWS , escolha a região na qual deseja replicar o segredo.
   b. (Opcional) Para Encryption key (Chave de criptografia), escolha uma chave do KMS para criptografar o segredo. A chave deve ser criada na mesma região da réplica.
   c. (Opcional) Para adicionar outra região, selecione Add more regions (Adicionar mais regiões).
   d. Selecione Replicate (Replicar).

   Você retorna à página de detalhes do segredo. Na seção Replicate Secret (Replicar segredo), o Replication status (Status de replicação) é exibido para cada região.

AWS CLI

Example Replicar um segredo para outra região

O exemplo de `replicate-secret-to-regions` a seguir replica um segredo para eu-west-3. A réplica é criptografada com a chave AWS gerenciada aws/secretsmanager.

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

Example Crie um segredo e replique-o

O exemplo a seguir cria um segredo e o replica para eu-west-3. A réplica é criptografada com a chave AWS gerenciada 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

Para replicar um segredo, use o comando `ReplicateSecretToRegions`. Para ter mais informações, consulte the section called “AWS SDKs”.

Promoção de um segredo de réplica para um segredo autônomo no AWS Secrets Manager

Um segredo de réplica é um segredo que é replicado de um primário em outra Região da AWS. Ele tem o mesmo valor e metadados do segredo primário, mas pode ser criptografado com uma chave diferente do KMS. Uma réplica de segredo não pode ser atualizada independentemente de seu segredo primário, exceto pela sua chave de criptografia. Promover uma réplica de segredo desconecta a réplica do segredo primário e transforma a réplica do segredo em um segredo autônomo. Alterações no segredo primário não serão replicadas para o segredo autônomo.

Pode ser útil promover uma réplica de segredo para um segredo autônomo como uma solução de recuperação de desastres se o segredo primário ficar indisponível. Ou talvez promover uma réplica a um segredo autônomo se você quiser ativar a alternância para a réplica.

Se promover uma réplica, certifique-se de atualizar as aplicações correspondentes para usar o segredo autônomo.

O Secrets Manager gera uma entrada de log do CloudTrail quando você promove um segredo. Para obter mais informações, consulte the section called “Faça login com AWS CloudTrail ”.

Para promover um segredo de réplica (console)

1. Faça login no Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Navegue até a região da réplica.
3. Na página Secrets (Segredos), selecione o segredo de réplica.
4. Na página de detalhes da réplica de segredo, escolha Promote to standalone secret (Promover a segredo autônomo).
5. Na caixa de diálogo Promote replica to standalone secret (Promover réplica a segredo autônomo), insira a região e escolha Promote replica (Promover réplica).

AWS CLI

Example Promover um segredo de réplica a um segredo primário

O exemplo de `stop-replication-to-replica` a seguir remove o link entre um segredo de réplica e o primário. O segredo de réplica é promovido a um segredo primário na região da réplica. É necessário chamar `stop-replication-to-replica` diretamente da região da réplica.

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

SDK da AWS

Para promover uma réplica a um segredo autônomo, use o comando `StopReplicationToReplica`. Você deve chamar esse comando da região da réplica de segredo. Para obter mais informações, consulte the section called “AWS SDKs”.

Evite a AWS Secrets Manager replicação

Como os segredos podem ser replicados usando `ReplicateSecretToRegions`ou quando são criados usando `CreateSecret`, se você quiser impedir que os usuários repliquem segredos, recomendamos que você evite ações que contenham o `AddReplicaRegions` parâmetro. Você pode usar uma `Condition` declaração em suas políticas de permissão para permitir somente ações que não adicionem regiões de réplica. Veja os exemplos de políticas a seguir para ver as declarações de condição que você pode usar.

Example Impedir a permissão de replicação

O exemplo de política a seguir mostra como permitir todas as ações que não adicionam regiões de réplica. Isso impede que os usuários repliquem segredos por meio de `ReplicateSecretToRegions` e. `CreateSecret`

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

Example Permitir permissão de replicação somente para regiões específicas

A política a seguir mostra como permitir tudo o que segue:

- Crie segredos sem replicação
- Crie segredos com replicação para regiões somente nos Estados Unidos e Canadá
- Replique segredos para regiões somente nos Estados Unidos e Canadá

```
{
```

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

Solucionar problemas de AWS Secrets Manager replicação

A seguir, estão alguns motivos pelos quais a replicação pode falhar.

Existe um segredo com o mesmo nome na região selecionada

Para resolver esse problema, você pode substituir o segredo com nome duplicado na região da réplica. Repita a replicação e, na caixa de diálogo Tentar replicação novamente, escolha Substituir.

Não há permissões disponíveis na chave do KMS para concluir a replicação

O Secrets Manager primeiro descriptografa o segredo antes de criptografá-lo novamente com a nova chave do KMS na região da réplica. Se você não tiver permissão do `kms:Decrypt` para a chave de criptografia na região primária, você encontrará esse erro. Para criptografar o segredo replicado com uma chave do KMS diferente de `aws/secretsmanager`, você precisa de `kms:GenerateDataKey` e `kms:Encrypt` para a chave. Consulte the section called “Permissões para a chave do KMS”.

A chave do KMS foi desativada ou não foi encontrada

Se a chave de criptografia na região primária for desativada ou excluída, o Secrets Manager não poderá replicar o segredo. Esse erro pode ocorrer mesmo se você tiver alterado a chave de criptografia, se o segredo tiver versões personalizadas rotuladas que foram criptografadas com a chave de criptografia desativada ou excluída. Para obter informações sobre como o Secrets Manager faz criptografia, consulte the section called “Criptografia e descriptografia de segredos”. Para contornar esse problema, você pode recriar as versões secretas para que o Secrets Manager as criptografe com a chave de criptografia atual. Para obter mais informações, consulte Altere a chave de criptografia para um secredo. Em seguida, tente novamente a replicação.

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

Você não habilitou a região onde a replicação ocorre

Para obter informações sobre como habilitar uma região, consulte Gerenciando AWS regiões. no Guia de referência de gerenciamento de AWS contas.

Obtenha segredos de AWS Secrets Manager

O Secrets Manager gera uma entrada de CloudTrail registro quando você recupera um segredo. Para ter mais informações, consulte the section called “Faça login com AWS CloudTrail ”.

Você pode recuperar valores secretos usando:

- Obtenha um valor secreto do Secrets Manager usando Java
- Obtenha um valor secreto do Secrets Manager usando Python
- Obtenha um valor secreto do Secrets Manager usando o.NET
- Obtenha um valor secreto do Secrets Manager usando Go
- Obtenha um valor secreto do Secrets Manager usando o SDK para C++ AWS
- Obtenha um valor secreto do Secrets Manager usando o JavaScript AWS SDK
- Obtenha um valor secreto do Secrets Manager usando o Kotlin SDK AWS
- Obtenha um valor secreto do Secrets Manager usando o AWS SDK do PHP
- Obtenha um valor secreto do Secrets Manager usando o SDK Ruby AWS
- Obtenha um valor secreto do Secrets Manager usando o Rust SDK AWS
- Obtenha um valor secreto usando o AWS CLI
- Obtenha um valor secreto usando o AWS console
- Use AWS Secrets Manager segredos em AWS Batch
- Obtenha um AWS Secrets Manager segredo em um AWS CloudFormation recurso
- Use AWS Secrets Manager segredos no Amazon Elastic Kubernetes Service
- Use AWS Secrets Manager segredos em GitHub empregos
- Usar segredos do AWS Secrets Manager no AWS IoT Greengrass
- Use AWS Secrets Manager segredos em AWS Lambda funções
- Usar segredos do AWS Secrets Manager no Parameter Store

Obtenha um valor secreto do Secrets Manager usando Java

Nos aplicativos, você pode recuperar seus segredos chamando `GetSecretValue` ou `BatchGetSecretValue` em qualquer um dos AWS SDKs. No entanto, recomendamos armazenar em cache seus valores de segredos usando o cache do lado do cliente. Armazenar segredos em cache melhora a velocidade e reduz os seus custos.

Para se conectar a um banco de dados usando as credenciais em um segredo, você pode usar os drivers de conexão SQL do Secrets Manager, que agrupam o driver JDBC básico. Isso também usa o cache do lado do cliente, para reduzir o custo de chamar as APIs do Secrets Manager.

Tópicos

- Obtenha um valor secreto do Secrets Manager usando Java com armazenamento em cache do lado do cliente
- Conecte-se a um banco de dados SQL usando JDBC com credenciais em um segredo AWS Secrets Manager
- Obtenha um valor secreto do Secrets Manager usando o Java AWS SDK

Obtenha um valor secreto do Secrets Manager usando Java com armazenamento em cache do lado do cliente

Ao recuperar um segredo, você pode usar o componente de cache baseado em Java do Secrets Manager para armazená-lo em cache para uso futuro. Recuperar um segredo armazenado em cache é mais rápido do que recuperá-lo do Secrets Manager. Como há um custo para chamar APIs do Secrets Manager, usar um cache pode reduzir seus custos. Para ver todas as formas pelas quais você pode recuperar segredos, consulte Obtenha segredos.

A política de caches é a do menos usado recentemente (LRU). Assim, quando o cache precisar descartar um segredo, escolherá o segredo menos usado recentemente. Por padrão, o cache atualiza segredos a cada hora. É possível configurar com qual frequência o segredo será atualizado no cache e você poderá se conectar à recuperação do segredo para adicionar mais funcionalidades.

O cache não obriga a coleta de resíduos depois que as referências de cache são liberadas. A implementação do cache não inclui a invalidação do cache. A implementação do cache é centrada em torno do próprio cache, portanto, não é reforçada ou tem ênfase em segurança. Se você necessitar de mais segurança, como criptografia de itens no cache, use as interfaces e os métodos abstratos fornecidos.

Para usar o componente, é necessário ter o seguinte:

- Um ambiente de desenvolvimento Java 8 ou superior. Consulte Java SE Downloads (Downloads do Java SE) no site da Oracle.
- O AWS SDK 1.x para Java. Você pode usar as duas versões do AWS SDK for Java em seus projetos. Para obter mais informações, consulte Usando o SDK for Java 1.x e 2.x. side-by-side

Para baixar o código-fonte, consulte o componente cliente de cache baseado em Java do Secrets Manager em. GitHub

Para adicionar o componente ao seu projeto, inclua a seguinte dependência no arquivo do Maven pom.xml. Para obter mais informações sobre o Maven, consulte o Getting Started Guide (Guia de primeiros passos) no site do projeto Maven do Apache.

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

Permissões obrigatórias:

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

Para ter mais informações, consulte Referência de permissões.

Referência

- SecretCache
- SecretCacheConfiguration
- SecretCacheHook

Example Recuperar segredos

O exemplo de código a seguir mostra uma função Lambda que recupera uma string do segredo. Ele segue a prática recomendada de instanciar o cache fora do manipulador de funções para que ele não continue chamando a API se você chamar a função Lambda novamente.

```
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

Um cache na memória para segredos solicitados no Secrets Manager. Você usa the section called "getSecretString" ou the section called "getSecretBinary" para recuperar um segredo do cache. Você pode definir as configurações de cache executando-as em um objeto the section called "SecretCacheConfiguration" no construtor.

Para obter mais informações, incluindo exemplos, consulte the section called "Java com cache do lado do cliente".

Construtores

`public SecretCache()`

Construtor padrão para um objeto `SecretCache`.

`public SecretCache(AWSSecretsManagerClientBuilder builder)`

Constrói um novo cache usando um cliente do Secrets Manager criado usando o `AWSSecretsManagerClientBuilder` fornecido. Use esse construtor para personalizar o cliente Secrets Manager, por exemplo, para usar uma região ou endpoint específico.

`public SecretCache(AWSSecretsManager client)`

Constrói um novo cache de segredo usando o `AWSSecretsManagerClient` fornecido. Use esse construtor para personalizar o cliente Secrets Manager, por exemplo, para usar uma região ou endpoint específico.

`public SecretCache(SecretCacheConfiguration config)`

Constrói um novo cache de segredo usando a `the section called "SecretCacheConfiguration"` fornecida.

Métodos

getSecretString

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

Recupera um segredo de string do Secrets Manager. Retorna um `String`.

getSecretBinary

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

Recupera um segredo de binário do Secrets Manager. Retorna um `ByteBuffer`.

refreshNow

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

Impõe a atualização do cache. Retorna `true` se a atualização for concluída sem erro; caso contrário, `false`.

feche

```
public void close()
```

Fecha o cache.

SecretCacheConfiguration

Opções de configuração de cache para um the section called “SecretCache”, como o tamanho máximo do cache e a vida útil (TTL) para segredos armazenados em cache.

Construtor

```
public SecretCacheConfiguration
```

Construtor padrão para um objeto `SecretCacheConfiguration`.

Métodos

getClient

```
public AWSSecretsManager getClient()
```

Ele retorna o `AWSSecretsManagerClient` de onde o cache recupera segredos.

setClient

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

Configura o cliente do `AWSSecretsManagerClient` de onde o cache recupera segredos.

getCacheHook

```
public SecretCacheHook getCacheHook()
```

Retorna a interface do the section called “SecretCacheHook” usada para conectar atualizações de cache.

setCacheHook

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

Define a interface do the section called “SecretCacheHook” usada para conectar atualizações de cache.

getMaxCacheTamanho

```
public int getMaxCacheSize()
```

Retorna o tamanho máximo do cache. O padrão é de 1.024 segredos.

setMaxCacheTamanho

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

Define o tamanho máximo do cache. O padrão é de 1.024 segredos.

getCacheItemTTL

```
public long getCacheItemTTL()
```

Retorna o TTL em milissegundos para os itens armazenados em cache. Quando um segredo armazenado em cache excede esse TTL, o cache recupera uma nova cópia do segredo do `AWSSecretsManagerClient`. O padrão é 1 hora em milissegundos.

O cache atualizará o segredo de forma síncrona quando ele for solicitado após o TTL. Se a atualização síncrona falhar, o cache retornará o segredo obsoleto.

setCacheItemTTL

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

Define o TTL em milissegundos para os itens armazenados em cache. Quando um segredo armazenado em cache excede esse TTL, o cache recupera uma nova cópia do segredo do `AWSSecretsManagerClient`. O padrão é 1 hora em milissegundos.

getVersionStage

```
public String getVersionStage()
```

Retorna a versão dos segredos que você deseja armazenar em cache. Para obter mais informações, consulte Secret versions (Versões de segredos). O padrão é `"AWSCURRENT"`.

setVersionStage

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

Define a versão dos segredos que você deseja armazenar em cache. Para obter mais informações, consulte Secret versions (Versões de segredos). O padrão é `"AWSCURRENT"`.

SecretCacheConfiguration Com o cliente

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

Define o `AWSSecretsManagerClient` de onde os segredos serão recuperados. Retorna o objeto da `SecretCacheConfiguration` atualizado com a nova configuração.

SecretCacheConfiguration withCacheHook

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

Define a interface usada para conectar o cache na memória. Retorna o objeto da `SecretCacheConfiguration` atualizado com a nova configuração.

SecretCacheConfiguration withMaxCacheTamanho

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

Define o tamanho máximo do cache. Retorna o objeto da `SecretCacheConfiguration` atualizado com a nova configuração.

SecretCacheConfiguration withCacheItemTTL

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

Define o TTL em milissegundos para os itens armazenados em cache. Quando um segredo armazenado em cache excede esse TTL, o cache recupera uma nova cópia do segredo do `AWSSecretsManagerClient`. O padrão é 1 hora em milissegundos. Retorna o objeto da `SecretCacheConfiguration` atualizado com a nova configuração.

SecretCacheConfiguration withVersionStage

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

Define a versão dos segredos que você deseja armazenar em cache. Para obter mais informações, consulte Secret versions (Versões de segredos). Retorna o objeto da `SecretCacheConfiguration` atualizado com a nova configuração.

SecretCacheHook

Uma interface para conectar a um the section called “SecretCache” para executar ações nos segredos sendo armazenados no cache.

put

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

Prepare o objeto para o armazenamento no cache.

Retorna o objeto a ser armazenado no cache.

get

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

Derive o objeto do objeto sendo armazenado em cache.

Retorna o objeto a ser retornado do cache

Conecte-se a um banco de dados SQL usando JDBC com credenciais em um segredo AWS Secrets Manager

Em aplicativos Java, você pode usar os drivers de conexão SQL do Secrets Manager para se conectar aos bancos de dados MySQL, PostgreSQL, Oracle, MSSQLServer, Db2 e Redshift usando credenciais armazenadas no Secrets Manager. Cada driver empacota o driver JDBC base para que você possa usar chamadas JDBC para acessar o seu banco de dados. No entanto, em

vez de passar um nome de usuário e senha para a conexão, você fornece o ID de um segredo. O driver chama o Secrets Manager para recuperar o valor do segredo e usa as credenciais no segredo para se conectar ao banco de dados. O driver também armazena em cache as credenciais usando a biblioteca em cache no lado do cliente de Java. Assim, as conexões futuras não exigem uma chamada para o Secrets Manager. Por padrão, o cache é atualizado a cada hora e também quando o segredo é alternado. Para configurar o cache, consulte the section called "SecretCacheConfiguration".

Você pode baixar o código-fonte em GitHub.

Para usar os drivers de conexão SQL do Secrets Manager:

- A sua aplicação deve estar em Java 8 ou superior.
- O seu segredo deve ser um dos seguintes:
  - Um segredo de banco de dados na estrutura JSON esperada. Para verificar o formato, no console do Secrets Manager, veja o seu segredo e escolha Retrieve secret value (Recuperar o valor do segredo). Como alternativa, no AWS CLI, ligue get-secret-value.
  - Um segredo gerenciado pelo Amazon RDS. Para esse tipo de segredo, é necessário especificar um endpoint e uma porta ao estabelecer a conexão.
  - Um segredo gerenciado pelo Amazon Redshift. Para esse tipo de segredo, é necessário especificar um endpoint e uma porta ao estabelecer a conexão.

Se o banco de dados for replicado para outras regiões, para se conectar a uma réplica de banco de dados em outra região, especifique o endpoint regional e a porta ao criar a conexão. Você pode armazenar informações de conexão regional no segredo como pares de chave-valor adicionais, nos parâmetros do SSM Parameter Store ou na configuração do código.

Para adicionar o driver ao seu projeto, no arquivo de compilação do Maven `pom.xml`, adicione a seguinte dependência para o driver. Para obter mais informações, consulte Secrets Manager SQL Connection Library (Biblioteca de conexões SQL do Secrets Manager) no site Maven Central Repository.

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

O driver usa a cadeia de fornecedores de credenciais padrão. Se você executar o driver no Amazon EKS, ele poderá retirar as credenciais do nó em que está sendo executado, em vez do perfil da conta de serviço. Para resolver isso, adicione a versão 1 de `com.amazonaws:aws-java-sdk-sts` ao seu arquivo de projeto Gradle ou Maven como uma dependência.

Para definir uma URL de endpoint de AWS PrivateLink DNS e uma região no `secretsmanager.properties` arquivo:

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

Para substituir a região primária, defina a variável de ambiente `AWS_SECRET_JDBC_REGION` ou faça a seguinte alteração no arquivo `secretsmanager.properties`:

```
drivers.region = region
```

Permissões obrigatórias:

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

Para ter mais informações, consulte Referência de permissões.

Exemplos:

- Estabeleça uma conexão com um banco de dados
- Estabelecer uma conexão especificando o endpoint e a porta
- Use o grupo de conexões c3p0 para estabelecer uma conexão
- Usar o agrupamento de conexões c3p0 para estabelecer uma conexão especificando o endpoint e a porta

Estabeleça uma conexão com um banco de dados

O exemplo a seguir mostra como estabelecer uma conexão com um banco de dados usando as credenciais e informações sobre a conexão em um segredo. Assim que você tiver uma conexão, será possível usar chamadas JDBC para acessar o banco de dados. Para obter mais informações, consulte JDBC Basics (Conceitos básicos de JDBC) no site da documentação de 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);
```

Estabelecer uma conexão especificando o endpoint e a porta

O exemplo a seguir mostra como estabelecer uma conexão com um banco de dados usando as credenciais em um segredo com um endpoint e uma porta que você especifica.

Os segredos gerenciados pelo Amazon RDS não incluem o endpoint e a porta do banco de dados. Para se conectar a um banco de dados usando credenciais principais em um segredo gerenciado pelo Amazon RDS, você as especifica em seu código.

Segredos que são replicados para outras regiões podem melhorar a latência da conexão com o banco de dados regional, mas não contêm informações diferentes de conexão em relação ao segredo da origem. Cada réplica é uma cópia do segredo de origem. Para armazenar informações de conexão regional no segredo, adicione mais pares de chave-valor para as informações de endpoint e porta das regiões.

Assim que você tiver uma conexão, será possível usar chamadas JDBC para acessar o banco de dados. Para obter mais informações, consulte JDBC Basics (Conceitos básicos de JDBC) no site da documentação de 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);
```

Use o grupo de conexões c3p0 para estabelecer uma conexão

O exemplo a seguir mostra como estabelecer um pool de conexões com um arquivo `c3p0.properties` que usa o driver para recuperar credenciais e informações de conexão do segredo. Em `user` e `jdbcUrl`, insira o ID do segredo para configurar o grupo de conexões. Em seguida, você pode recuperar conexões do grupo e usá-las como qualquer outra conexão de banco de dados. Para obter mais informações, consulte JDBC Basics (Conceitos básicos de JDBC) no site da documentação de Java.

Para obter mais informações sobre c3p0, consulte c3p0 no site 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
```

Usar o agrupamento de conexões c3p0 para estabelecer uma conexão especificando o endpoint e a porta

O exemplo a seguir mostra como estabelecer um pool de conexões com um `c3p0.properties` arquivo que usa o driver para recuperar credenciais em um segredo com um endpoint e uma porta especificados por você. Em seguida, você pode recuperar conexões do grupo e usá-las como qualquer outra conexão de banco de dados. Para obter mais informações, consulte JDBC Basics (Conceitos básicos de JDBC) no site da documentação de Java.

Os segredos gerenciados pelo Amazon RDS não incluem o endpoint e a porta do banco de dados. Para se conectar a um banco de dados usando credenciais principais em um segredo gerenciado pelo Amazon RDS, você as especifica em seu código.

Segredos que são replicados para outras regiões podem melhorar a latência da conexão com o banco de dados regional, mas não contêm informações diferentes de conexão em relação ao segredo da origem. Cada réplica é uma cópia do segredo de origem. Para armazenar informações de conexão regional no segredo, adicione mais pares de chave-valor para as informações de endpoint e porta das regiões.

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
```

Obtenha um valor secreto do Secrets Manager usando o Java AWS SDK

Nos aplicativos, você pode recuperar seus segredos chamando `GetSecretValue` ou `BatchGetSecretValue` em qualquer um dos AWS SDKs. No entanto, recomendamos armazenar em cache seus valores de segredos usando o cache do lado do cliente. Armazenar segredos em cache melhora a velocidade e reduz os seus custos.

- Se você armazenar credenciais de banco de dados no segredo, use os drivers de conexão SQL do Secrets Manager para se conectar a um banco de dados usando as credenciais no segredo.

- Para outros tipos de segredos, use o componente de cache baseado em Java do Secrets Manager ou chame o SDK diretamente com ou. `GetSecretValue` `BatchGetSecretValue`

Os exemplos de códigos a seguir mostram como usar `GetSecretValue`.

Permissões obrigatórias: `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);
        }
    }
}
```

Obtenha um valor secreto do Secrets Manager usando Python

Nos aplicativos, você pode recuperar seus segredos chamando `GetSecretValue` ou `BatchGetSecretValue` em qualquer um dos AWS SDKs. No entanto, recomendamos armazenar em cache seus valores de segredos usando o cache do lado do cliente. Armazenar segredos em cache melhora a velocidade e reduz os seus custos.

Tópicos

- Obtenha um valor secreto do Secrets Manager usando Python com armazenamento em cache do lado do cliente

- Obtenha um valor secreto do Secrets Manager usando o Python SDK AWS
- Obtenha um lote de valores secretos do Secrets Manager usando o Python SDK AWS

Obtenha um valor secreto do Secrets Manager usando Python com armazenamento em cache do lado do cliente

Ao recuperar um segredo, você pode usar o componente de cache baseado em Python do Secrets Manager para armazená-lo em cache para uso futuro. Recuperar um segredo armazenado em cache é mais rápido do que recuperá-lo do Secrets Manager. Como há um custo para chamar APIs do Secrets Manager, usar um cache pode reduzir seus custos. Para ver todas as formas pelas quais você pode recuperar segredos, consulte Obtenha segredos.

A política de caches é a do menos usado recentemente (LRU). Assim, quando o cache precisar descartar um segredo, escolherá o segredo menos usado recentemente. Por padrão, o cache atualiza segredos a cada hora. É possível configurar com qual frequência o segredo será atualizado no cache e você poderá se conectar à recuperação do segredo para adicionar mais funcionalidades.

O cache não obriga a coleta de resíduos depois que as referências de cache são liberadas. A implementação do cache não inclui a invalidação do cache. A implementação do cache é centrada em torno do próprio cache, portanto, não é reforçada ou tem ênfase em segurança. Se você necessitar de mais segurança, como criptografia de itens no cache, use as interfaces e os métodos abstratos fornecidos.

Para usar o componente, é necessário ter o seguinte:

- Python 3.6 ou superior.
- botocore 1.12 ou superior. Consulte AWS SDK para Python e Botocore.
- setuptools_scm 3.2 ou superior. Consulte https://pypi.org/project/setuptools-scm/.

Para baixar o código-fonte, consulte o componente cliente de cache baseado em Python do Secrets Manager em. GitHub

Para instalar o componente, use o seguinte comando.

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

Permissões obrigatórias:

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

Para ter mais informações, consulte Referência de permissões.

Referência

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

Example Recuperar segredos

O exemplo a seguir mostra como obter o valor de um segredo nomeado como *`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

Um cache na memória para segredos recuperados no Secrets Manager. Você usa the section called “get_secret_string” ou the section called “get_secret_binary” para recuperar um segredo do cache. Você pode definir as configurações de cache executando-as em um objeto the section called “SecretCacheConfig” no construtor.

Para obter mais informações, incluindo exemplos, consulte the section called “Python com armazenamento em cache do lado do cliente”.

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

```
    client = client
)
```

Estes são os métodos disponíveis:

- get_secret_string
- get_secret_binary

get_secret_string

Recupera o valor da string do segredo.

Sintaxe da solicitação

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

Parâmetros

- `secret_id` (string): [obrigatório] o nome ou o ARN do segredo.
- `version_stage` (string): a versão dos segredos que você quer recuperar. Para obter mais informações, consulte as versões secretas. O padrão é 'AWSCURRENT'.

Tipo de retorno

string

get_secret_binary

Recupera o valor do binário do segredo.

Sintaxe da solicitação

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

Parâmetros

- `secret_id` (string): [obrigatório] o nome ou o ARN do segredo.

- `version_stage` (string): a versão dos segredos que você quer recuperar. Para obter mais informações, consulte as versões secretas. O padrão é 'AWSCURRENT'.

Tipo de retorno

String codificada em base64

SecretCacheConfig

Opções de configuração de cache para um the section called “SecretCache”, como o tamanho máximo do cache e a vida útil (TTL) para segredos armazenados em cache.

Parâmetros

`max_cache_size` (int)

O tamanho máximo do cache. O padrão é de `1024` segredos.

`exception_retry_delay_base` (int)

O número de segundos a aguardar ao encontrar uma exceção e antes de repetir a solicitação. O padrão é `1`.

`exception_retry_growth_factor` (int)pur

O fator de crescimento a ser usado para calcular o tempo de espera entre novas tentativas de solicitações com falha. O padrão é `2`.

`exception_retry_delay_max` (int)

O tempo máximo em segundos a aguardar entre solicitações com falha. O padrão é `3600`.

`default_version_stage` (str)

A versão de segredos que você deseja armazenar em cache. Para obter mais informações, consulte Secret versions (Versões de segredos). O padrão é `'AWSCURRENT'`.

`secret_refresh_interval` (int)

O número de segundos a aguardar entre a atualização de informações de segredos armazenados em cache. O padrão é `3600`.

`secret_cache_hook` (SecretCacheHook)

Uma implementação da classe abstrata de `SecretCacheHook`. O valor padrão é `None`.

SecretCacheHook

Uma interface para conectar a um the section called “SecretCache” para executar ações nos segredos sendo armazenados no cache.

Estes são os métodos disponíveis:

- put
- get

put

Prepara o objeto para armazenamento no cache.

Sintaxe da solicitação

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

Parâmetros

- `obj` (objeto): [obrigatório] o segredo ou objeto que contém o segredo.

Tipo de retorno

objeto

get

Deriva o objeto do objeto armazenado em cache.

Sintaxe da solicitação

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

Parâmetros

- `obj` (objeto): [obrigatório] o segredo ou objeto que contém o segredo.

Tipo de retorno

objeto

@InjectSecretString

Este decorador espera uma string de ID de segredo e the section called “SecretCache” como o primeiro e o segundo argumentos. O decorador retorna o valor da string do segredo. O segredo deve conter uma string.

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

cache = SecretCache()

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

@InjectKeywordedSecretString

Este decorador espera uma string de ID de segredo e the section called “SecretCache” como o primeiro e o segundo argumentos. Os argumentos restantes mapeiam parâmetros da função empacotada para chaves JSON no segredo. O segredo deve conter uma string na estrutura JSON.

Para um segredo que contém esse JSON:

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

O exemplo a seguir mostra como extrair os valores JSON para `username` e `password` do segredo.

```
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')
```

Obtenha um valor secreto do Secrets Manager usando o Python SDK AWS

Nos aplicativos, você pode recuperar seus segredos chamando `GetSecretValue` ou `BatchGetSecretValue` em qualquer um dos AWS SDKs. No entanto, recomendamos armazenar em cache seus valores de segredos usando o cache do lado do cliente. Armazenar segredos em cache melhora a velocidade e reduz os seus custos.

Para aplicações Python, use o componente de cache baseado em Python do Secrets Manager ou chame o SDK diretamente com `get_secret_value` ou `batch_get_secret_value`.

Os exemplos de códigos a seguir mostram como usar `GetSecretValue`.

Permissões obrigatórias: `secretsmanager:GetSecretValue`

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


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

        :param secret_name: The name of the secret fetched.
        :type secret_name: str
        """
        try:
            get_secret_value_response = self.client.get_secret_value(
                SecretId=secret_name
            )
            logging.info("Secret retrieved successfully.")
            return get_secret_value_response["SecretString"]
```

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

Obtenha um lote de valores secretos do Secrets Manager usando o Python SDK AWS

O exemplo de código a seguir mostra como obter um lote de valores de segredo do Secrets Manager.

Permissões obrigatórias:

- `secretsmanager:BatchGetSecretValue`
- `secretsmanager:GetSecretValue`permissão para cada segredo que você deseja recuperar.
- Se você usa filtros, também deve ter `secretsmanager:ListSecrets`.

Para um exemplo de política de permissões, consulte the section called “Exemplo: permissão para recuperar um grupo de valores secretos em um lote”.

> **Important**
>
> Se você tiver uma política de VPCE que nega permissão para recuperar um segredo individual no grupo que você está recuperando, `BatchGetSecretValue` não retornará nenhum valor de segredo e retornará um erro.

```
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
```

Obtenha um valor secreto do Secrets Manager usando o.NET

Nos aplicativos, você pode recuperar seus segredos chamando `GetSecretValue` ou `BatchGetSecretValue` em qualquer um dos AWS SDKs. No entanto, recomendamos armazenar em cache seus valores de segredos usando o cache do lado do cliente. Armazenar segredos em cache melhora a velocidade e reduz os seus custos.

Tópicos

- Obtenha um valor secreto do Secrets Manager usando o.NET com armazenamento em cache do lado do cliente
- Obtenha um valor secreto do Secrets Manager usando o AWS SDK.NET

Obtenha um valor secreto do Secrets Manager usando o.NET com armazenamento em cache do lado do cliente

Ao recuperar um segredo, você pode usar o componente de cache baseado em .NET do Secrets Manager para armazená-lo em cache para uso futuro. Recuperar um segredo armazenado em cache é mais rápido do que recuperá-lo do Secrets Manager. Como há um custo para chamar APIs do Secrets Manager, usar um cache pode reduzir seus custos. Para ver todas as formas pelas quais você pode recuperar segredos, consulte Obtenha segredos.

A política de caches é a do menos usado recentemente (LRU). Assim, quando o cache precisar descartar um segredo, escolherá o segredo menos usado recentemente. Por padrão, o cache atualiza segredos a cada hora. É possível configurar com qual frequência o segredo será atualizado no cache e você poderá se conectar à recuperação do segredo para adicionar mais funcionalidades.

O cache não obriga a coleta de resíduos depois que as referências de cache são liberadas. A implementação do cache não inclui a invalidação do cache. A implementação do cache é centrada em torno do próprio cache, portanto, não é reforçada ou tem ênfase em segurança. Se você necessitar de mais segurança, como criptografia de itens no cache, use as interfaces e os métodos abstratos fornecidos.

Para usar o componente, é necessário ter o seguinte:

- .NET Framework 4.6.2 ou superior, ou .NET Standard 2.0 ou superior. Consulte Download .NET (Baixe .NET) no site da Microsoft .NET.
- O AWS SDK para.NET. Consulte the section called “AWS SDKs”.

Para baixar o código-fonte, consulte Cliente de armazenamento em cache para.NET on GitHub.

Para usar o cache, primeiro instancie-o e, em seguida, recupere o seu segredo usando `GetSecretString` ou `GetSecretBinary`. Em recuperações sucessivas, o cache retorna a cópia armazenada em cache do segredo.

Para obter o pacote de cache

- Execute um destes procedimentos:
  - Execute o seguinte comando CLI do .NET em seu diretório de projeto.

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

  - Adicione a seguinte referência de pacote ao seu arquivo `.csproj`.

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

Permissões obrigatórias:

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

Para ter mais informações, consulte Referência de permissões.

Referência

- SecretsManagerCache
- SecretCacheConfiguration
- Eu SecretCacheHook

Example Recuperar segredos

O exemplo de código a seguir mostra um método que recupera um segredo chamado *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 Configuração da duração da atualização do cache de vida útil (TTL)

O exemplo de código a seguir mostra um método que recupera um segredo chamado *MySecret*e define a duração da atualização do cache TTL como 24 horas.

```
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

Um cache na memória para segredos solicitados no Secrets Manager. Você usa the section called "GetSecretString" ou the section called "GetSecretBinary" para recuperar um segredo do cache. Você pode definir as configurações de cache executando-as em um objeto the section called "SecretCacheConfiguration" no construtor.

Para obter mais informações, incluindo exemplos, consulte the section called ".NET com armazenamento em cache do lado do cliente".

Construtores

```
public SecretsManagerCache()
```

Construtor padrão para um objeto `SecretsManagerCache`.

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

Constrói um novo cache usando um cliente Secrets Manager criado usando o fornecido AmazonSecretsManagerClient. Use esse construtor para personalizar o cliente do Secrets Manager, por exemplo, para usar uma região ou endpoint específico.

Parâmetros

secretsManager

O AmazonSecretsManagerClientpara recuperar segredos de.

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

Constrói um novo cache de segredo usando a the section called "SecretCacheConfiguration" fornecida. Use esse construtor para configurar o cache, por exemplo, o número de segredos a serem armazenados em cache e com que frequência ele é atualizado.

Parâmetros

config

Uma the section called "SecretCacheConfiguration" que contém informações de configuração para o cache.

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

Constrói um novo cache usando um cliente Secrets Manager criado usando o fornecido AmazonSecretsManagerCliente umthe section called "SecretCacheConfiguration". Use este

construtor para personalizar o cliente do Secrets Manager, por exemplo, para usar uma região ou endpoint específico, assim como configurar o cache, por exemplo, o número de segredos a serem armazenados em cache e com que frequência ele será atualizado.

Parâmetros

secretsManager

O AmazonSecretsManagerClientpara recuperar segredos de.

config

Uma the section called “SecretCacheConfiguration” que contém informações de configuração para o cache.

Métodos

GetSecretString

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

Recupera um segredo de string do Secrets Manager.

Parâmetros

secretId

O ARN ou o nome do segredo a ser recuperado.

GetSecretBinary

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

Recupera um segredo de binário do Secrets Manager.

Parâmetros

secretId

O ARN ou o nome do segredo a ser recuperado.

RefreshNowAsync

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

Solicita o valor do segredo do Secrets Manager e atualiza o cache com quaisquer alterações. Se não houver entrada de cache existente, ele criará uma nova. Se a atualização for bem-sucedida, ele retornará `true`.

Parâmetros

secretId

O ARN ou o nome do segredo a ser recuperado.

GetCachedSecret

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

Retorna a entrada de cache para o segredo especificado, se ele existir no cache. Caso contrário, ele recupera o segredo do Secrets Manager e cria uma nova entrada de cache.

Parâmetros

secretId

O ARN ou o nome do segredo a ser recuperado.

SecretCacheConfiguration

Opções de configuração de cache para um the section called “SecretsManagerCache”, como o tamanho máximo do cache e a vida útil (TTL) para segredos armazenados em cache.

Propriedades

CacheItemTTL

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

O TTL de um item de cache em milissegundos. O padrão é de `3600000` minutos ou 1 hora. O máximo é de `4294967295` milissegundos, que é aproximadamente 49,7 dias.

MaxCacheSize

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

O tamanho máximo do cache. O padrão é de 1.024 segredos. O máximo é 65.535.

VersionStage

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

A versão de segredos que você deseja armazenar em cache. Para obter mais informações, consulte Secret versions (Versões de segredos). O padrão é `"AWSCURRENT"`.

Cliente

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

O AmazonSecretsManagerClientpara recuperar segredos de. Se for `null`, o cache instancia um novo cliente. O padrão é `null`.

CacheHook

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

O the section called “Eu SecretCacheHook”.

Eu SecretCacheHook

Uma interface para conectar a um the section called “SecretsManagerCache” para executar ações nos segredos sendo armazenados no cache.

Métodos

Put

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

Prepare o objeto para o armazenamento no cache.

Retorna o objeto a ser armazenado no cache.

Obtenção

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

Derive o objeto do objeto sendo armazenado em cache.

Retorna o objeto a ser retornado do cache

Obtenha um valor secreto do Secrets Manager usando o AWS SDK.NET

Nos aplicativos, você pode recuperar seus segredos chamando `GetSecretValue` ou `BatchGetSecretValue` em qualquer um dos AWS SDKs. No entanto, recomendamos armazenar em cache seus valores de segredos usando o cache do lado do cliente. Armazenar segredos em cache melhora a velocidade e reduz os seus custos.

Para aplicações .NET, use o componente de cache baseado em .NET do Secrets Manager ou chame o SDK diretamente com `GetSecretValue` ou `BatchGetSecretValue`.

Os exemplos de códigos a seguir mostram como usar `GetSecretValue`.

Permissões obrigatórias: `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;
            }
        }
    }
```

Obtenha um valor secreto do Secrets Manager usando Go

Nos aplicativos, você pode recuperar seus segredos chamando `GetSecretValue` ou `BatchGetSecretValue` em qualquer um dos AWS SDKs. No entanto, recomendamos armazenar

em cache seus valores de segredos usando o cache do lado do cliente. Armazenar segredos em cache melhora a velocidade e reduz os seus custos.

Tópicos

- Obtenha um valor secreto do Secrets Manager usando Go com cache do lado do cliente
- Obtenha um valor secreto do Secrets Manager usando o Go AWS SDK

Obtenha um valor secreto do Secrets Manager usando Go com cache do lado do cliente

Ao recuperar um segredo, você pode usar o componente de cache baseado em Go do Secrets Manager para armazená-lo em cache para uso futuro. Recuperar um segredo armazenado em cache é mais rápido do que recuperá-lo do Secrets Manager. Como há um custo para chamar APIs do Secrets Manager, usar um cache pode reduzir seus custos. Para ver todas as formas pelas quais você pode recuperar segredos, consulte Obtenha segredos.

A política de caches é a do menos usado recentemente (LRU). Assim, quando o cache precisar descartar um segredo, escolherá o segredo menos usado recentemente. Por padrão, o cache atualiza segredos a cada hora. É possível configurar com qual frequência o segredo será atualizado no cache e você poderá se conectar à recuperação do segredo para adicionar mais funcionalidades.

O cache não obriga a coleta de resíduos depois que as referências de cache são liberadas. A implementação do cache não inclui a invalidação do cache. A implementação do cache é centrada em torno do próprio cache, portanto, não é reforçada ou tem ênfase em segurança. Se você necessitar de mais segurança, como criptografia de itens no cache, use as interfaces e os métodos abstratos fornecidos.

Para usar o componente, é necessário ter o seguinte:

- AWS SDK para Go. Consulte the section called “AWS SDKs”.

Para baixar o código-fonte, consulte Cliente de cache do Secrets Manager Go ativado GitHub.

Para configurar um ambiente de desenvolvimento Go, consulte Golang Getting Started (Conceitos básicos de Golang) no site da Go Programming Language.

Permissões obrigatórias:

- `secretsmanager:DescribeSecret`

- `secretsmanager:GetSecretValue`

Para ter mais informações, consulte Referência de permissões.

Referência

- tipo Cache
- tipo CacheConfig
- tipo CacheHook

Example Recuperar segredos

O exemplo de código a seguir mostra uma função Lambda que recupera um segredo.

```
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)
}
```

tipo Cache

Um cache na memória para segredos solicitados no Secrets Manager. Você usa the section called "GetSecretString" ou the section called "GetSecretBinary" para recuperar um segredo do cache.

O exemplo a seguir mostra como definir as configurações de 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 },
)
```

Para obter mais informações, incluindo exemplos, consulte the section called “Use o armazenamento em cache do lado do cliente”.

Métodos

Novo

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

O New constrói um cache do segredo usando opções funcionais. Caso contrário, usa opções padrão. Inicializa um SecretsManager cliente a partir de uma nova sessão. Inicializa CacheConfig com os valores padrão. Inicializa o cache LRU com um tamanho máximo padrão.

GetSecretString

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

GetSecretString obtém o valor da string secreta do cache para determinado ID secreto. Caso a operação falhe, ele retorna a string do segredo e um erro.

GetSecretStringWithStage

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

GetSecretStringWithStage obtém o valor da string secreta do cache para determinado ID secreto e estágio de versão. Caso a operação falhe, ele retorna a string do segredo e um erro.

GetSecretBinary

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

GetSecretBinary obtém o valor binário secreto do cache para determinado ID secreto. Caso a operação falhe, ele retorna o binário do segredo e um erro.

GetSecretBinaryWithStage

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

GetSecretBinaryWithStage obtém o valor binário secreto do cache para determinado ID secreto e estágio de versão. Caso a operação falhe, ele retorna o binário do segredo e um erro.

tipo CacheConfig

Opções de configuração de cache para o Cache, como o tamanho máximo do cache, a etapa de versão padrão e a vida útil (TTL) para segredos armazenados em 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

Uma interface para conectar a um Cache para executar ações no segredo que está sendo armazenado no cache.

Métodos

Put

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

Prepara o objeto para armazenamento no cache.

Obtenção

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

Deriva o objeto do objeto armazenado em cache.

Obtenha um valor secreto do Secrets Manager usando o Go AWS SDK

Nos aplicativos, você pode recuperar seus segredos chamando `GetSecretValue` ou `BatchGetSecretValue` em qualquer um dos AWS SDKs. No entanto, recomendamos armazenar em cache seus valores de segredos usando o cache do lado do cliente. Armazenar segredos em cache melhora a velocidade e reduz os seus custos.

Para aplicações Go, use o componente de cache baseado em Go do Secrets Manager ou chame o SDK diretamente com `GetSecretValue` ou `BatchGetSecretValue`.

O exemplo de código a seguir mostra como obter um valor de segredo do Secrets Manager.

Permissões obrigatórias: `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.
  }
```

Obtenha um valor secreto do Secrets Manager usando o SDK para C++ AWS

Para aplicativos C++, chame o SDK diretamente com GetSecretValueou. BatchGetSecretValue

O exemplo de código a seguir mostra como obter um valor de segredo do Secrets Manager.

Permissões obrigatórias: `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();
}
```

Obtenha um valor secreto do Secrets Manager usando o JavaScript AWS SDK

Para JavaScript aplicativos, chame o SDK diretamente com `getSecretValue`ou `batchGetSecretValue`.

O exemplo de código a seguir mostra como obter um valor de segredo do Secrets Manager.

Permissões obrigatórias: `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;
  }
};
```

Obtenha um valor secreto do Secrets Manager usando o Kotlin SDK AWS

Para aplicativos Kotlin, chame o SDK diretamente com ou. GetSecretValueBatchGetSecretValue

O exemplo de código a seguir mostra como obter um valor de segredo do Secrets Manager.

Permissões obrigatórias: `secretsmanager:GetSecretValue`

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

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

Obtenha um valor secreto do Secrets Manager usando o AWS SDK do PHP

Para aplicações PHP, chame o SDK diretamente com `GetSecretValue` ou `BatchGetSecretValue`.

O exemplo de código a seguir mostra como obter um valor de segredo do Secrets Manager.

Permissões obrigatórias: `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
```

Obtenha um valor secreto do Secrets Manager usando o SDK Ruby AWS

Para aplicações Ruby, chame o SDK diretamente com `get_secret_value` ou `batch_get_secret_value`.

O exemplo de código a seguir mostra como obter um valor de segredo do Secrets Manager.

Permissões obrigatórias: `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
```

Obtenha um valor secreto do Secrets Manager usando o Rust SDK AWS

Para aplicativos Rust, chame o SDK diretamente com GetSecretValueou. BatchGetSecretValue

O exemplo de código a seguir mostra como obter um valor de segredo do Secrets Manager.

Permissões obrigatórias: `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(())
}
```

Obtenha um valor secreto usando o AWS CLI

Permissões obrigatórias: `secretsmanager:GetSecretValue`

Example Recuperar o valor secreto criptografado de um segredo

O exemplo de `get-secret-value` a seguir obtém o valor atual do segredo.

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

Example Recuperar o valor secreto anterior

O exemplo de `get-secret-value` a seguir recupera o valor secreto anterior.

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

Obtenha um grupo de segredos em um lote usando o AWS CLI

Permissões obrigatórias:

- `secretsmanager:BatchGetSecretValue`
- `secretsmanager:GetSecretValue`permissão para cada segredo que você deseja recuperar.
- Se você usa filtros, também deve ter `secretsmanager:ListSecrets`.

Para um exemplo de política de permissões, consulte the section called “Exemplo: permissão para recuperar um grupo de valores secretos em um lote”.

> **Important**
>
> Se você tiver uma política de VPCE que nega permissão para recuperar um segredo individual no grupo que você está recuperando, `BatchGetSecretValue` não retornará nenhum valor de segredo e retornará um erro.

Example Recuperar o valor do segredo de um grupo de segredos listados por nome

O exemplo de `batch-get-secret-value` a seguir obtém o valor do segredo para três segredos.

```
aws secretsmanager batch-get-secret-value \
          --secret-id-list MySecret1 MySecret2 MySecret3
```

Example Recuperar o valor do segredo de um grupo de segredos selecionados pelo filtro

O `batch-get-secret-value` de exemplo a seguir obtém o valor dos segredos que têm uma tag chamada “Test”.

```
aws secretsmanager batch-get-secret-value \
          --filters Key="tag-key",Values="Test"
```

Obtenha um valor secreto usando o AWS console

Para recuperar um segredo (console)

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na lista de segredos, escolha o segredo que você deseja recuperar.
3. Na seção Valor secreto, selecione Recuperar valor secreto.

   Secrets Manager exibe a versão atual (`AWSCURRENT`) do segredo. Para ver outras versões do segredo, como `AWSPREVIOUS` versões com rótulos personalizados, use o the section called “AWS CLI”.

Use AWS Secrets Manager segredos em AWS Batch

AWS Batch ajuda você a executar cargas de trabalho de computação em lote no Nuvem AWS. Com AWS Batch, você pode injetar dados confidenciais em seus trabalhos armazenando seus dados confidenciais em AWS Secrets Manager segredos e, em seguida, referenciando-os na definição de seu trabalho. Para mais informações, consulte Especificar dados sigilosos usando segredos do Secrets Manager.

Obtenha um AWS Secrets Manager segredo em um AWS CloudFormation recurso

Com AWS CloudFormation isso, você pode recuperar um segredo para usar em outro AWS CloudFormation recurso. Um cenário comum é primeiro criar um segredo com uma senha gerada

pelo Secrets Manager e, em seguida, recuperar o nome de usuário e a senha do segredo para usar como credenciais para um novo banco de dados. Para obter informações sobre como criar segredos com AWS CloudFormation, consulteAWS CloudFormation.

Para recuperar um segredo em um AWS CloudFormation modelo, você usa uma referência dinâmica. Quando você cria a pilha, a referência dinâmica extrai o valor secreto para o AWS CloudFormation recurso, para que você não precise codificar as informações secretas. Em vez disso, é necessário fazer referência ao segredo pelo nome ou ARN. Você pode usar uma referência dinâmica para um segredo em qualquer propriedade do recurso. Você não pode usar uma referência dinâmica para um segredo em metadados do recurso, por exemplo, `AWS::CloudFormation::Init`, pois isso tornaria o valor secreto visível no console.

Uma referência dinâmica para um segredo tem o seguinte padrão:

```
{{resolve:secretsmanager:secret-id:SecretString:json-key:version-stage:version-id}}
```

secret-id

O nome ou ARN completo do segredo. Para acessar um segredo em sua AWS conta, você pode usar o nome secreto. Para acessar um segredo em uma AWS conta diferente, use o ARN do segredo.

json-key (opcional)

O nome da chave do par de chave-valor cujo valor você deseja recuperar. Se você não especificar um`json-key`, AWS CloudFormation recuperará todo o texto secreto. Esse segmento não pode incluir o caractere de dois pontos (`:`).

version-stage (opcional)

A versão do segredo a ser usado. O Secrets Manager usa rótulos de preparação para acompanhar diferentes versões durante o processo de alternância. Se você usar `version-stage`, não especifique `version-id`. Se você não especificar `version-stage` ou `version-id`, o padrão é a versão `AWSCURRENT`. Esse segmento não pode incluir o caractere de dois pontos (`:`).

version-id (opcional)

O identificador exclusivo da versão do segredo a usar. Se você especificar `version-id`, não especifique `version-stage`. Se você não especificar `version-stage` ou `version-id`, o padrão é a versão `AWSCURRENT`. Esse segmento não pode incluir o caractere de dois pontos (`:`).

Para obter mais informações, consulte Uso de referências dinâmicas para especificar segredos do Secrets Manager.

> **Note**
>
> Não crie uma referência dinâmica usando uma barra invertida (\) como valor final. AWS CloudFormation não consigo resolver essas referências, o que causa uma falha no recurso.

Use AWS Secrets Manager segredos no Amazon Elastic Kubernetes Service

Para mostrar segredos do Secrets Manager como arquivos montados em pods do Amazon EKS, você pode usar o AWS Secrets and Configuration Provider (ASCP) para o driver CSI do Kubernetes Secrets Store. O ASCP funciona com o Amazon Elastic Kubernetes Service (Amazon EKS) 1.17+ executando um grupo de nós do Amazon EC2. AWS Fargate grupos de nós não são suportados. Com o ASCP, você pode armazenar e gerenciar segredos no Secrets Manager e, em seguida, recuperá-los por meio de workloads executadas no Amazon EKS. Se seu segredo contiver vários pares de chave-valor no formato JSON, você poderá escolher quais montará no Amazon EKS. O ASCP usa Sintaxe JMESPath para consultar os pares de chave-valor no seu segredo. O ASCP também funciona com parâmetros do Parameter Store.

Se você usar um cluster privado do Amazon EKS, garanta que a VPC na qual o cluster está tenha um endpoint do Secrets Manager. O driver CSI do Secrets Store usa o endpoint para fazer chamadas para o Secrets Manager. Para obter mais informações sobre como criar um endpoint, consulte Endpoint da VPC.

Se você usar a alternância automática do Secrets Manager para seus segredos, também poderá usar o recurso de reconciliador de alternância do driver da CSI do armazenamento de segredos para garantir que está recuperando o segredo mais recente do Secrets Manager. Para obter mais informações, consulte Alternância automática de conteúdos montados e segredos do Kubernetes sincronizados.

Tópicos

- Etapa 1: configurar o controle de acesso
- Etapa 2: Instalar e configurar o ASCP
- Etapa 3: identificar quais segredos montar

- Etapa 4: montar os segredos como arquivos no pod Amazon EKS
- Solução de problemas
- SecretProviderClass

Etapa 1: configurar o controle de acesso

O ASCP recupera a identidade do pod do Amazon EKS e a troca por uma função do IAM. Você define permissões em uma política do IAM para essa função do IAM. Quando o ASCP assume a função do IAM, ele obtém acesso aos segredos que você autorizou. Outros contêineres não podem acessar os segredos, a menos que você também os associe à função do IAM.

Se as chamadas do ASCP para pesquisar a região e a função do IAM associadas ao pod forem limitadas pelo Kubernetes, você poderá alterar as cotas de limitação usando, conforme mostrado na Etapa 2. `helm install`

Para conceder ao seu pod Amazon EKS acesso aos segredos no Secrets Manager

1. Crie uma política de permissões que `secretsmanager:GetSecretValue` conceda `secretsmanager:DescribeSecret` permissões aos segredos que o pod precisa acessar. Para visualizar um exemplo de política, consulte the section called “Exemplo: permissão para ler e descrever segredos individuais”.
2. Crie um provedor IAM OIDC Connect (OIDC) para o cluster, se você ainda não tiver um. Para obter mais informações, consulte Criar um provedor IAM OIDC para seu cluster no Guia do usuário do Amazon EKS.
3. Crie uma função do IAM para a conta de serviço e anexe a política a ela. Para obter mais informações, consulte Criar uma função do IAM para uma conta de serviço no Guia do usuário do Amazon EKS.
4. Se você usa um cluster privado do Amazon EKS, certifique-se de que a VPC na qual o cluster está tenha um AWS STS endpoint. Para obter informações sobre a criação de um endpoint, consulte Interface VPC endpoints no AWS Identity and Access Management Guia do usuário.

Etapa 2: Instalar e configurar o ASCP

O ASCP está disponível GitHub no repositório secrets-store-csi-provider-aws. O repositório também contém arquivos YAML de exemplo para criar e montar um segredo.

Durante a instalação, você pode configurar o ASCP para usar um endpoint FIPS. Para uma lista de endpoints , consulte the section called “Endpoint do Secrets Manager”.

Para instalar o ASCP usando o Helm

1. Para garantir que o repositório esteja apontando para os gráficos mais recentes, use `helm repo update.`
2. Adicione o gráfico de drivers CSI da Secrets Store.

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

3. Instale o chart. Para configurar a limitação, adicione a seguinte sinalização: `--set-json 'k8sThrottlingParams={"qps": "`*`<number of queries per second>`*`", "burst": "`*`<number of queries per second>`*`"}'`

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

4. Adicione o gráfico ASCP.

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

5. Instale o chart. Para usar um endpoint FIPS, adicione a seguinte sinalização: `--set useFipsEndpoint=true`

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

Para instalar usando o YAML no repositório

- Use os seguintes comandos.

```
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
```

Etapa 3: identificar quais segredos montar

Para determinar quais segredos o ASCP monta no Amazon EKS como arquivos no sistema de arquivos, você cria um arquivo YAML the section called "`SecretProviderClass`". `SecretProviderClass`Lista os segredos a serem montados e o nome do arquivo como montá-los. O`SecretProviderClass`deve estar no mesmo namespace que o pod Amazon EKS que ele faz referência.

Os exemplos a seguir mostram como usar o `SecretProviderClass` para descrever os segredos que você deseja montar e como nomear os arquivos montados no pod do Amazon EKS.

Exemplos:

- Exemplo: montar segredos por nome ou ARN
- Exemplo: montar pares de chave/valor com base em um segredo
- Exemplo: definir uma região de failover para um segredo multirregional
- Exemplo: escolher um segredo de failover para montar

Exemplo: montar segredos por nome ou ARN

O exemplo a seguir mostra um `SecretProviderClass` que monta três arquivos no Amazon EKS:

1. Um segredo especificado por um ARN completo.
2. Um segredo especificado pelo nome.
3. Uma versão específica de um segredo.

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    objects: |
```

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

Exemplo: montar pares de chave/valor com base em um segredo

O exemplo a seguir mostra um `SecretProviderClass` que monta três arquivos no Amazon EKS:

1. Um segredo especificado por um ARN completo.
2. O par de chave-valor `username` do mesmo segredo.
3. O par de chave-valor `password` do mesmo segredo.

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    objects: |
      - objectName: "arn:aws:secretsmanager:us-east-2:111122223333:secret:MySecret-
a1b2c3"
        jmesPath:
            - path: username
              objectAlias: dbusername
            - path: password
              objectAlias: dbpassword
```

Exemplo: definir uma região de failover para um segredo multirregional

Para proporcionar disponibilidade durante interrupções de conectividade ou para configurações de recuperação de desastres, o ASCP oferece suporte a um recurso de failover automatizado para recuperar segredos de uma região secundária.

O exemplo a seguir mostra um `SecretProviderClass` que recupera um segredo que é replicado em várias regiões. Neste exemplo, o ASCP tenta recuperar o segredo de `us-east-1` e `us-east-2`.

Se qualquer uma das regiões retornar um erro 4xx (p. ex., para um problema de autenticação), o ASCP não montará nenhum segredo. Se o segredo for recuperado com êxito de `us-east-1`, o ASCP montará o valor desse segredo. Se o segredo não for recuperado com êxito de `us-east-1`, mas for recuperado com êxito de `us-east-2`, o ASCP montará o valor desse segredo.

```
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"
```

Exemplo: escolher um segredo de failover para montar

O exemplo a seguir mostra um `SecretProviderClass` que especifica qual segredo montar em caso de failover. O segredo de failover não é uma réplica. Neste exemplo, o ASCP tenta recuperar os dois segredos especificados por `objectName`. Se qualquer um deles retornar um erro 4xx (p. ex., para um problema de autenticação), o ASCP não montará nenhum segredo. Se o segredo for recuperado com êxito de `us-east-1`, o ASCP montará o valor desse segredo. Se o segredo não for recuperado com êxito de `us-east-1`, mas for recuperado com êxito de `us-east-2`, o ASCP montará o valor desse segredo. O arquivo montado no Amazon EKS tem o nome `MyMountedSecret`.

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-secrets
spec:
  provider: aws
  parameters:
    region: us-east-1
    failoverRegion: us-east-2
    objects: |
      - objectName: "arn:aws:secretsmanager:us-east-1:111122223333:secret:MySecret-
a1b2c3"
        objectAlias: "MyMountedSecret"
```

```
        failoverObject:
          - objectName: "arn:aws:secretsmanager:us-
east-2:111122223333:secret:MyFailoverSecret-d4e5f6"
```

Etapa 4: montar os segredos como arquivos no pod Amazon EKS

Para montar segredos no Amazon EKS

1. Aplique o `SecretProviderClass` ao pod com o comando`kubectl apply -f ExampleSecretProviderClass.yaml`.
2. Implante seu pod com o comando`kubectl apply -f ExampleDeployment.yaml`.
3. O ASCP monta os arquivos.

Solução de problemas

Você pode visualizar a maioria dos erros ao descrever a implantação do pod.

Para ver mensagens de erro para o contêiner

1. Obtenha uma lista de nomes de pods com o comando a seguir. Se você não estiver usando o namespace padrão, use `-n <NAMESPACE>`.

   ```
   kubectl get pods
   ```

2. Para descrever o pod, no comando a seguir, para*<PODID>*use o ID do pod dos pods encontrados na etapa anterior. Se você não estiver usando o namespace padrão, use `-n <NAMESPACE>`.

   ```
   kubectl describe pod/<PODID>
   ```

Para ver erros para o ASCP

- Para encontrar mais informações nos registros do provedor, <PODID>use o ID do pod csi-secrets-store-provider-aws no comando a seguir.

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

SecretProviderClass

Você usa o YAML para descrever quais segredos montar no Amazon EKS usando o ASCP. Para ver exemplos, consulte the section called “Montar segredos por nome ou ARN”.

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

O campo `parameters` contém os detalhes da solicitação de montagem:

região

(Opcional) O Região da AWS do segredo. Se você não usar esse campo, o ASCP procurará a Região a partir da anotação no nó. Essa pesquisa adiciona sobrecarga para solicitações de montagem, portanto, recomendamos que você forneça a Região para clusters que usam um grande número de pods.

Se você também especificar `failoverRegion`, o ASCP tentará recuperar o segredo de ambas as regiões. Se qualquer uma das regiões retornar um erro 4xx (p. ex., para um problema de autenticação), o ASCP não montará nenhum segredo. Se o segredo for recuperado com êxito de `region`, o ASCP montará o valor desse segredo. Se o segredo não for recuperado com êxito de `region`, mas for recuperado com êxito de `failoverRegion`, o ASCP montará o valor desse segredo.

failoverRegion

(Opcional) Se você incluir esse campo, o ASCP tentará recuperar o segredo das regiões definidas em `region` e nesse campo. Se qualquer uma das regiões retornar um erro 4xx (p. ex., para um problema de autenticação), o ASCP não montará nenhum segredo. Se o segredo for recuperado com êxito de `region`, o ASCP montará o valor desse segredo. Se o segredo não for recuperado com êxito de `region`, mas for recuperado com êxito de `failoverRegion`, o ASCP

montará o valor desse segredo. Para obter um exemplo de como usar esse campo, consulte Definir uma região de failover para um segredo multirregional.

pathTranslation

(Opcional) Um só caractere de substituição para usar se o nome de arquivo no Amazon EKS vá conter o caractere separador de caminho, como barra (/) no Linux. O ASCP não poderá criar um arquivo montado que contenha um caractere separador de caminho. Em vez disso, o ASCP substituirá o caractere separador de caminho por outro caractere. Se você não usar esse campo, o caractere substituto será um sublinhado (_), portanto, por exemplo,`My/Path/Secret` será montado como `My_Path_Secret`.

Para impedir a substituição de caracteres, digite a string`False`.

objects

Uma string contendo uma declaração do YAML dos segredos a serem montados. Recomendamos o uso de uma string com várias linhas ou um caractere pipe (|) no YAML.

objectName

O nome ou ARN completo do segredo. Se você usar o ARN, poderá omitir o `objectType`. A menos que você especifique `objectAlias`, esse campo passará a ser o nome de arquivo do segredo no pod do Amazon EKS. Se você usar um ARN, a região no ARN deverá corresponder ao campo `region`. Se você incluir um `failoverRegion`, esse campo representará o `objectName` primário.

objectType

Necessário se você não usar um ARN do Secrets Manager para`objectName`. Pode ser `secretsmanager` ou `ssmparameter`.

objectAlias

(Opcional) O nome do arquivo do segredo no pod do Amazon EKS. Se você não especificar esse campo, o `objectName` aparece como o nome do arquivo.

objectVersion

(Opcional) O ID da versão do segredo. Não recomendado porque você deve atualizar o ID da versão sempre que atualizar o segredo. Por padrão, a versão mais recente é usada. Se você incluir um `failoverRegion`, esse campo representará o `objectVersion` primário.

objectVersionLabel

(Opcional) O alias da versão. O padrão é a versão mais recente AWSCURRENT. Para ter mais informações, consulte the section called “Versões secretas”. Se você incluir um `failoverRegion`, esse campo representará o `objectVersionLabel` primário.

jmesPath

(Opcional) Um mapa das chaves no segredo para os arquivos a serem montados no Amazon EKS. Para usar esse campo, o valor do segredo deve estar no formato JSON. Se você usar esse campo, deverá incluir os subcampos `path` e `objectAlias`.

caminho

Uma chave de um par de chave-valor no JSON do valor do segredo. Se o campo contiver um hífen, use aspas simples para delimitá-lo, por exemplo: `path: '"hyphenated-path"'`

objectAlias

O nome do arquivo a ser montado no pod Amazon EKS. Se o campo contiver um hífen, use aspas simples para delimitá-lo, por exemplo: `objectAlias: '"hyphenated-alias"'`

failoverObject

(Opcional) Se você especificar esse campo, o ASCP tentará recuperar o segredo especificado no `objectName` primário e o segredo especificado no subcampo `objectName` de `failoverObject`. Se qualquer um deles retornar um erro 4xx (p. ex., para um problema de autenticação), o ASCP não montará nenhum segredo. Se o segredo for recuperado com êxito do `objectName` primário, o ASCP montará o valor desse segredo. Se o segredo não for recuperado com êxito do `objectName` primário, mas for recuperado com êxito do `objectName` do failover, o ASCP montará o valor desse segredo. Se incluir esse campo, você deverá incluir o campo `objectAlias`. Para obter um exemplo de como usar esse campo, consulte Escolher um segredo de failover para montar.

Normalmente, você usa esse campo quando o segredo de failover não for uma réplica. Para obter um exemplo de como especificar uma réplica, consulte Definir uma região de failover para um segredo multirregional.

objectName

O nome ou ARN completo do segredo de failover. Se você usar um ARN, a região no ARN deverá corresponder ao campo `failoverRegion`.

objectVersion

(Opcional) O ID da versão do segredo. Deve corresponder ao `objectVersion` primário. Não recomendado porque você deve atualizar o ID da versão sempre que atualizar o segredo. Por padrão, a versão mais recente é usada.

objectVersionLabel

(Opcional) O alias da versão. O padrão é a versão mais recente AWSCURRENT. Para ter mais informações, consulte the section called “Versões secretas”.

Use AWS Secrets Manager segredos em GitHub empregos

Para usar um segredo em um GitHub trabalho, você pode usar uma GitHub ação para recuperar segredos AWS Secrets Manager e adicioná-los como variáveis de ambiente mascaradas em seu GitHub fluxo de trabalho. Para obter mais informações sobre GitHub ações, consulte Entendendo GitHub ações nos GitHub documentos.

Quando você adiciona um segredo ao seu GitHub ambiente, ele fica disponível para todas as outras etapas do seu GitHub trabalho. Siga as orientações em Fortalecimento de segurança para GitHub ações para ajudar a evitar que segredos em seu ambiente sejam usados indevidamente.

Você pode definir a cadeia de caracteres inteira no valor do segredo como o valor da variável de ambiente ou, caso a cadeia de caracteres seja JSON, pode analisar o JSON a fim de definir variáveis de ambiente individuais para cada par chave-valor JSON. Se o valor do segredo for um binário, a ação o converterá em uma cadeia de caracteres.

Para visualizar as variáveis de ambiente criadas utilizando seus segredos, ative o log de depuração. Para obter mais informações, consulte Habilitando o registro de depuração no GitHub Docs.

Para usar as variáveis de ambiente criadas a partir de seus segredos, consulte Variáveis de ambiente na GitHub documentação.

Pré-requisitos

Para usar essa ação, primeiro você precisa configurar AWS as credenciais e defini-las Região da AWS em seu GitHub ambiente usando a `configure-aws-credentials` etapa. Siga as instruções em Configurar ações de AWS credenciais para que GitHub as ações assumam a função diretamente usando o provedor GitHub OIDC. Isso permite que você use credenciais de curta duração e evite armazenar chaves de acesso adicionais externas ao Secrets Manager.

O perfil do IAM que a ação assume deve ter as seguintes permissões:

- `GetSecretValue` sobre os segredos que você deseja recuperar.
- `ListSecrets` em todos os segredos.
- (Opcional) `Decrypt` KMS key se os segredos estiverem criptografados com um chave gerenciada pelo cliente.

Para ter mais informações, consulte Autenticação e controle de acesso.

Uso

Para usar a ação, adicione uma etapa ao fluxo de trabalho que use a sintaxe a seguir.

```
- 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
```

Parâmetros

`secret-ids`

ARNS, nomes e prefixos de nomes dos segredos.

Para definir o nome da variável de ambiente, insira-o antes do ID do segredo e acrescente uma vírgula. Por exemplo, o `ENV_VAR_1, secretId` cria uma variável de ambiente chamada ENV_VAR_1 utilizando o `secretId` do segredo. O nome da variável de ambiente pode consistir em letras maiúsculas, números e sublinhados.

Para usar um prefixo, digite pelo menos três caracteres seguidos de um asterisco. Por exemplo, `dev*` corresponde a todos os segredos com um nome que começa em dev. O número máximo de segredos correspondentes que podem ser recuperados é 100. Se você definir o nome da variável e o prefixo corresponder a diversos segredos, a ação falhará.

`name-transformation`

Por padrão, a etapa cria cada nome de variável de ambiente utilizando o nome do segredo, o qual é transformado para incluir apenas letras maiúsculas, números e sublinhados, bem como não começar com um número. Para as letras no nome, você pode configurar a etapa para usar letras minúsculas com `lowercase` ou não alterar as maiúsculas e minúsculas das letras com. `none` O valor padrão é `uppercase`.

`parse-json-secrets`

(Opcional) Por padrão, a ação define o valor da variável de ambiente para a cadeia de caracteres JSON inteira no valor do segredo. `parse-json-secrets`Defina como `true` para criar variáveis de ambiente para cada par de valores-chave no JSON.

Observe que, se o JSON usar chaves que diferenciam maiúsculas de minúsculas, como “nome” e “Nome”, a ação apresentará conflitos de nomes duplicados. Nesse caso, defina os `parse-json-secrets` como `false` e analise o valor do segredo JSON separadamente.

Nomeação de variáveis de ambiente

As variáveis de ambiente criadas pela ação são nomeadas da mesma forma que os segredos de onde elas vêm. As variáveis de ambiente têm requisitos de nomenclatura mais rígidos do que os segredos, então a ação transforma os nomes secretos para atender a esses requisitos. Por exemplo, a ação transforma letras minúsculas em letras maiúsculas. Se você analisar o JSON do segredo, o nome da variável de ambiente incluirá tanto o nome secreto quanto o nome da chave JSON, por exemplo. `MYSECRET_KEYNAME` Você pode configurar a ação para não transformar letras minúsculas.

Se duas variáveis de ambiente terminarem com o mesmo nome, a ação falhará. Nesse caso, você deve especificar os nomes que deseja usar para as variáveis de ambiente como aliases.

Exemplos de quando os nomes podem entrar em conflito:

- Um segredo chamado "MySecret" e um segredo chamado “mysecret” se tornariam variáveis de ambiente chamadas “MYSECRET”.
- Um segredo chamado “secret_keyname” e um segredo analisado em JSON chamado “Secret” com uma chave chamada “keyname” se tornariam variáveis de ambiente chamadas “SECRET_KEYNAME”.

Você pode definir o nome da variável de ambiente especificando um alias, conforme mostrado no exemplo a seguir, que cria uma variável chamada. `ENV_VAR_NAME`

```
secret-ids: |
  ENV_VAR_NAME, secretId2
```

Aliases em branco

- Se você definir `parse-json-secrets: true` e inserir um alias em branco, seguido por uma vírgula e depois pelo ID secreto, a ação nomeará a variável de ambiente da mesma forma que as chaves JSON analisadas. Os nomes das variáveis não incluem o nome secreto.

  Se o segredo não contiver um JSON válido, a ação criará uma variável de ambiente e a nomeará da mesma forma que o nome do segredo.
- Se você definir `parse-json-secrets: false` e inserir um alias em branco, seguido por uma vírgula e o ID secreto, a ação nomeará as variáveis de ambiente como se você não tivesse especificado um alias.

O exemplo a seguir mostra um alias em branco.

```
,secret2
```

Exemplos

Example 1. Obtenha segredos por nome e por ARN

O exemplo a seguir cria variáveis de ambiente para segredos identificados por nome e por 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
```

Variáveis de ambiente criadas:

```
EXAMPLESECRETNAME: secretValue1
TEST1: secretValue2
_0_TEST_SECRET: secretValue3
_PROD_EXAMPLE_SECRET: secretValue4
SECRET_ALIAS_1: secretValue5
SECRET_ALIAS_2: secretValue6
SECRET2: secretValue7
```

Example 2. Obtenha todos os segredos que começam com determinado prefixo

O exemplo a seguir cria variáveis de ambiente para todos os segredos com nomes que começam com *beta*.

```
- name: Get Secret Names by Prefix
  uses: 2
  with:
    secret-ids: |
      beta*    # Retrieves all secrets that start with 'beta'
```

Variáveis de ambiente criadas:

```
BETASECRETNAME: secretValue1
BETATEST: secretValue2
BETA_NEWSECRET: secretValue3
```

Example 3. Análise do JSON no segredo

O exemplo a seguir cria variáveis de ambiente ao analisar o JSON no segredo.

```
- 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
```

O segredo `test/secret` tem o seguinte valor de segredo.

```
{
```

```
  "api_user": "user",
  "api_key": "key",
  "config": {
    "active": "true"
  }
}
```

O segredo `secret2` tem o seguinte valor de segredo.

```
{
  "myusername": "alejandro_rosalez",
  "mypassword": "EXAMPLE_PASSWORD"
}
```

Variáveis de ambiente criadas:

```
TEST_SECRET_API_USER: "user"
TEST_SECRET_API_KEY: "key"
TEST_SECRET_CONFIG_ACTIVE: "true"
MYUSERNAME: "alejandro_rosalez"
MYPASSWORD: "EXAMPLE_PASSWORD"
```

Example 4 Use letras minúsculas para nomes de variáveis de ambiente

O exemplo a seguir cria uma variável de ambiente com um nome em minúsculas.

```
- name: Get secrets
  uses: aws-actions/aws-secretsmanager-get-secrets@v2
  with:
    secret-ids: exampleSecretName
    name-transformation: lowercase
```

Variável de ambiente criada:

```
examplesecretname: secretValue
```

Usar segredos do AWS Secrets Manager no AWS IoT Greengrass

O AWS IoT Greengrass é um software que amplia os recursos da nuvem para dispositivos locais. Isso permite que os dispositivos coletem e analisem dados mais próximos da fonte de informações,

reajam de maneira autônoma a eventos locais e se comuniquem com segurança uns com os outros em redes locais.

O AWS IoT Greengrass permite que você se autentique com serviços e aplicativos de dispositivos do Greengrass sem codificar senhas, tokens ou outros segredos. Você pode usar o AWS Secrets Manager para armazenar e gerenciar seus segredos na nuvem. O AWS IoT Greengrass estende o Secrets Manager para dispositivos de núcleo do Greengrass, de modo que seus conectores e funções do Lambda possam usar segredos locais para interagir com serviços e aplicações.

Para integrar um segredo a um grupo do Greengrass, crie um recurso de grupo que faça referência ao segredo do Secrets Manager. Esse recurso de segredo faz referência ao segredo da nuvem usando o ARN associado. Para saber como criar, gerenciar e usar recursos de segredo, consulte Trabalhar com recursos de segredo no Guia do desenvolvedor da AWS IoT.

Para Implantar segredos no núcleo do AWS IoT Greengrass, consulte Implantar segredos no núcleo do AWS IoT Greengrass.

Use AWS Secrets Manager segredos em AWS Lambda funções

Você pode usar a extensão Lambda de AWS parâmetros e segredos para recuperar e armazenar segredos AWS Secrets Manager em funções do Lambda sem usar um SDK. Recuperar um segredo armazenado em cache é mais rápido do que recuperá-lo do Secrets Manager. Como há um custo para chamar APIs do Secrets Manager, usar um cache pode reduzir seus custos. A extensão pode recuperar segredos do Secrets Manager e parâmetros do Parameter Store. Para obter informações sobre o Parameter Store, consulte Parameter Store integration with Lambda extensions (Integração do Parameter Store com as extensões do Lambda) no Guia do usuário do AWS Systems Manager .

Uma extensão do Lambda corresponde a um processo complementar que aumenta os recursos de uma função Lambda. Para obter mais informações, consulte Lambda extensions (Extensões do Lambda) no Guia do desenvolvedor do Lambda. Para obter informações sobre o uso da extensão em uma imagem de contêiner, consulte Working with Lambda layers and extensions in container images. O Lambda registra informações de execução sobre a extensão junto com a função usando o Amazon CloudWatch Logs. Por padrão, a extensão registra uma quantidade mínima de informações em CloudWatch. Para registrar mais detalhes em log, defina a variável de ambiente `PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL` para `debug`.

Para fornecer o cache na memória para parâmetros e segredos, a extensão expõe um endpoint HTTP local, a porta 2773 do host local, ao ambiente do Lambda. Você pode configurar a porta ao definir a variável de ambiente `PARAMETERS_SECRETS_EXTENSION_HTTP_PORT`.

O Lambda cria instâncias separadas correspondentes ao nível de simultaneidade requerido por sua função. Cada instância é isolada e mantém o próprio cache local dos dados de configuração. Para obter mais informações sobre as instâncias e a simultaneidade do Lambda, consulte Managing concurrency for a Lambda function (Gerenciamento da simultaneidade para uma função Lambda) no Guia do desenvolvedor do Lambda.

Para adicionar a extensão para ARM, você deve usar a arquitetura `arm64` para sua função Lambda. Para obter mais informações, consulte Arquiteturas de conjuntos de instruções do Lambda no Guia do desenvolvedor do Lambda. A extensão oferece suporte para ARM nas seguintes regiões: Ásia-Pacífico (Mumbai), Leste dos EUA (Ohio), Europa (Irlanda), Europa (Frankfurt), Europa (Zurique), Leste dos EUA (Norte da Virgínia), Europa (Londres), Europa (Espanha), Ásia-Pacífico (Tóquio), Oeste dos EUA (Oregon), Ásia-Pacífico (Singapura), Ásia-Pacífico (Haiderabade) e Ásia-Pacífico (Sydney).

A extensão usa um AWS cliente. Para obter informações sobre como configurar o AWS cliente, consulte a referência de configurações no Guia de referência do AWS SDK e das ferramentas. Se sua função Lambda for executada em uma VPC, você precisará criar um VPC endpoint para que a extensão possa fazer chamadas para o Secrets Manager. Para ter mais informações, consulte Endpoint da VPC.

Permissões obrigatórias:

- A função de execução do Lambda deve ter `secretsmanager:GetSecretValue` permissão para o segredo.
- Se o segredo for criptografado com uma chave gerenciada pelo cliente em vez de Chave gerenciada pela AWS `aws/secretsmanager`, a função de execução também precisará de `kms:Decrypt` permissão para a chave KMS.

Para usar a extensão Lambda de AWS parâmetros e segredos

1. Adicione a AWS camada chamada AWS Parameters and Secrets Lambda Extension à sua função. Para obter instruções, consulte Adicionar camadas às funções no Lambda Developer Guide. Se você usar o AWS CLI para adicionar a camada, precisará do ARN da extensão. Para obter uma lista de ARNs, consulte AWS Parameters and Secrets Lambda Extension (Parâmetros e extensão do Secrets Lambda) ARNs no AWS Systems Manager Guia do usuário.
2. Conceda permissões à função de execução do Lambda para poder acessar segredos:
   - Permissão `secretsmanager:GetSecretValue` para o segredo. Consulte the section called “Exemplo: permissão para recuperar valores de segredos individuais”.

- (Opcional) Se o segredo for criptografado com uma chave gerenciada pelo cliente em vez da Chave gerenciada pela AWS `aws/secretsmanager`, a função de execução também precisará de `kms:Decrypt` permissão para a chave KMS.
- Você pode usar o controle de acesso por atributo (ABAC) com a função Lambda para permitir um acesso mais granular aos segredos na conta. Para obter mais informações, consulte the section called “Exemplo: Controlar o acesso a segredos usando etiquetas” e the section called “Exemplo: Limitar o acesso a identidades com etiquetas que correspondam às etiquetas dos segredos”.

3. Configure o cache com variáveis de ambiente do Lambda.
4. Para recuperar segredos do cache de extensão, primeiro você precisa adicionar o `X-AWS-Parameters-Secrets-Token` ao cabeçalho da solicitação. Defina o token como `AWS_SESSION_TOKEN`, fornecido pelo Lambda para todas as funções em execução. O uso desse cabeçalho indica que o chamador está no ambiente do Lambda.

   O exemplo de Python, a seguir, mostra como adicionar o cabeçalho.

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

5. Para recuperar um segredo na função Lambda, use uma das seguintes solicitações GET do HTTP:

   - Para recuperar um segredo, use o ARN ou nome completo do segredo em `secretId`.

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

   - Para recuperar o valor secreto anterior ou uma versão específica por meio de um rótulo de preparação, use o ARN ou o nome do segredo em `secretId` e use o rótulo de preparação em `versionStage`.

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

   - Para recuperar uma versão específica do segredo por ID, use o ARN ou o nome do segredo em `secretId` e use o ID da versão em `versionId`.

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

Example Recuperar um segredo (Python)

O exemplo de Python, a seguir, mostra como recuperar um segredo e analisar o resultado usando `json.loads`.

```
secrets_extension_endpoint = "http://localhost:" + \
    secrets_extension_http_port + \
    "/secretsmanager/get?secretId=" + \
    <secret_name>

  r = requests.get(secrets_extension_endpoint, headers=headers)

  secret = json.loads(r.text)["SecretString"] # load the Secrets Manager response
 into a Python dictionary, access the secret
```

AWS Parâmetros e segredos Variáveis de ambiente da extensão Lambda

Você pode configurar a extensão com as seguintes variáveis de ambiente.

Para obter informações sobre como usar variáveis de ambiente, consulte Usar variáveis de ambiente do Lambda no Guia do desenvolvedor do Lambda.

`PARAMETERS_SECRETS_EXTENSION_CACHE_ENABLED`

Defina como true (verdadeiro) para armazenar parâmetros e segredos em cache. Defina como false (falso) para não realizar armazenamento em cache. O padrão é true (verdadeiro).

`PARAMETERS_SECRETS_EXTENSION_CACHE_SIZE`

O número máximo de segredos e parâmetros a serem armazenados em cache. Deve ser um valor de zero a mil. Um valor de zero significa que não há armazenamento em cache. Essa variável será ignorada se `SSM_PARAMETER_STORE_TTL` e `SECRETS_MANAGER_TTL` forem zero. O padrão é mil.

`PARAMETERS_SECRETS_EXTENSION_HTTP_PORT`

A porta para o servidor HTTP local. A padrão é 2773.

`PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL`

O nível de registro em log que a extensão fornece: `debug`, `info`, `warn`, `error` ou `none`. Defina como `debug` para visualizar a configuração do cache. O padrão é `info`.

`PARAMETERS_SECRETS_EXTENSION_MAX_CONNECTIONS`

Número máximo de conexões para clientes HTTP que a extensão usa para fazer solicitações ao Parameter Store ou ao Secrets Manager. Essa é uma configuração por cliente. O padrão é três.

`SECRETS_MANAGER_TIMEOUT_MILLIS`

Tempo limite para solicitações ao Secrets Manager em milissegundos. Um valor de zero significa que não há tempo limite. O padrão é 0.

`SECRETS_MANAGER_TTL`

A TTL de um segredo no cache em segundos. Um valor de zero significa que não há armazenamento em cache. O máximo é 300 segundos. Essa variável será ignorada se `PARAMETERS_SECRETS_CACHE_SIZE` for zero. O padrão é 300 segundos.

`SSM_PARAMETER_STORE_TIMEOUT_MILLIS`

Tempo limite para solicitações ao Parameter Store em milissegundos. Um valor de zero significa que não há tempo limite. O padrão é 0.

`SSM_PARAMETER_STORE_TTL`

A TTL de um parâmetro no cache em segundos. Um valor de zero significa que não há armazenamento em cache. O máximo é 300 segundos. Essa variável será ignorada se `PARAMETERS_SECRETS_CACHE_SIZE` for zero. O padrão é 300 segundos.

Usar segredos do AWS Secrets Manager no Parameter Store

O Parameter Store do Systems Manager da AWS oferece armazenamento hierárquico seguro para o gerenciamento de dados de configuração e gerenciamento de segredos. Você pode armazenar dados, como senhas, strings de banco de dados e códigos de licença como valores de parâmetro. No entanto, o repositório de parâmetros não fornece serviços de rotação automática para os segredos armazenados. Em vez disso, o Parameter Store permite que você armazene o segredo no Secrets Manager e faça referência ao segredo como um parâmetro do Parameter Store.

Quando você configura o Parameter Store com o Secrets Manager, o Parameter Store do `secret-id` requer uma barra (/) antes da string do nome.

Para obter mais informações, consulte Fazer referência aos segredos do AWS Secrets Manager dos parâmetros do Parameter Store no Manual do usuário do AWS Systems Manager.

Gire segredos AWS Secrets Manager

Alternância é o processo de atualizar periodicamente um segredo. Quando o Secrets Manager alterna um segredo, ele atualiza as credenciais tanto no segredo quanto no banco de dados ou serviço. No Secrets Manager, você pode configurar alternância automática para seus segredos. Há duas formas de rotação:

- Alternância gerenciada— Para a maioria dos segredos gerenciados, você usa a rotação gerenciada, na qual o serviço configura e gerencia a rotação para você. A rotação gerenciada não usa uma função Lambda.
- the section called “Rotação por função Lambda”— Para outros tipos de segredos, a rotação do Secrets Manager usa uma função Lambda para atualizar o segredo e o banco de dados ou serviço.

Rotação gerenciada para AWS Secrets Manager segredos

Alguns serviços oferecem alternância gerenciada, na qual o serviço configura e gerencia a alternância para você. Com a rotação gerenciada, você não usa uma AWS Lambda função para atualizar o segredo e as credenciais no banco de dados.

Os seguintes serviços oferecem alternância gerenciada:

- O Amazon Aurora oferece rotação gerenciada para credenciais de usuário mestre. Para obter mais informações, consulte Gerenciamento de senhas com Amazon Aurora e o AWS Secrets Manager no Guia do usuário do Amazon Aurora.
- O Amazon ECS Service Connect oferece rotação gerenciada para certificados AWS Private Certificate Authority TLS. Para obter mais informações, consulte TLS with Service Connect no Amazon Elastic Container Service Developer Guide.
- O Amazon RDS oferece rotação gerenciada para credenciais de usuário mestre. Para obter mais informações, consulte Gerenciamento de senhas com Amazon RDS e o AWS Secrets Manager no Guia do usuário do Amazon RDS.
- O Amazon Redshift oferece rotação gerenciada para senhas de administrador. Para obter mais informações, consulte Gerenciamento de senhas de administrador do Amazon Redshift usando o Guia de gerenciamento AWS Secrets Manager de clusters do Amazon Redshift.

> **Tip**
>
> Para todos os outros tipos de segredos, consulte the section called “Rotação por função Lambda”.

A rotação de segredos gerenciados normalmente é concluída em um minuto. Durante a rotação, novas conexões que recuperam o segredo podem obter a versão anterior das credenciais. Em aplicações, é altamente recomendado que você siga a prática recomendada de utilizar um usuário de banco de dados criado com os privilégios mínimos necessários para sua aplicação, em vez de usar o usuário principal. Para usuários de aplicativos, para maior disponibilidade, você pode usar a Estratégia de rotação de usuários alternados.

Para alterar a programação da rotação gerenciada

1. Abra o segredo gerenciado no console do Secrets Manager. Você pode seguir um link do serviço de gerenciamento ou pesquisar o segredo no console do Secrets Manager.
2. Em Rotation schedule (Programação da alternância), insira sua programação no fuso horário UTC no Schedule expression builder (Desenvolvedor de expressão programada) ou como uma Schedule expression (Expressão programada). O Secrets Manager armazena sua programação como uma expressão `rate()` ou `cron()`. A janela de alternância começa automaticamente à 0h, a menos que você especifique um horário de início. Você pode alternar um segredo com intervalos a partir de quatro horas. Para ter mais informações, consulte Cronogramas de rotação.
3. (Opcional) Em Window duration (Duração da janela), escolha a duração da janela em que deseja que o Secrets Manager alterne o seu segredo, por exemplo, **3h**, por uma janela de três horas. A janela não pode se estender até a próxima janela de alternância. Se você não especificar Window duration (Duração da janela) para uma programação de alternância em horas, a janela será automaticamente encerrada após uma hora. Para uma programação de alternância em dias, a janela terminará automaticamente no final do dia.
4. Escolha Salvar.

Para alterar o cronograma de alternância gerenciada (AWS CLI)

- Chame `rotate-secret`. O exemplo a seguir alterna o segredo entre 16h e 18h UTC no 1.º e no 15.º dias do mês. Para ter mais informações, consulte Cronogramas de rotação.

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

Rotação por função Lambda

Para muitos tipos de segredos, o Secrets Manager usa uma AWS Lambda função para atualizar o segredo e o banco de dados ou serviço. Para obter mais informações sobre os custos do uso de uma função do Lambda, consulte Definição de preço.

Para alguns Segredos gerenciados, você usa alternância gerenciada. Para usar Alternância gerenciada, primeiro você cria o segredo por meio do serviço de gerenciamento.

Durante a alternância, o Secrets Manager registra eventos de logs que indicam o estado de alternância. Para ter mais informações, consulte the section called “Faça login com AWS CloudTrail ”.

Para alternar um segredo, o Secrets Manager chama uma função Lambda de acordo com o cronograma de rotação que você configurou. Se você também atualizar manualmente seu valor secreto enquanto a alternância automática estiver configurada, o Secrets Manager considerará essa alternância válida ao calcular a próxima data de alternância.

Durante a alternância, o Secrets Manager chama a mesma função várias vezes, cada vez com parâmetros diferentes. O Secrets Manager invoca a função com a seguinte estrutura de parâmetros de solicitação JSON:

```
{
    "Step" : "request.type",
    "SecretId" : "string",
    "ClientRequestToken" : "string"
    }
```

Se alguma etapa de alternância falhar, o Secrets Manager tentará novamente todo o processo de alternância várias vezes.

Tópicos

- Configure a rotação automática para segredos do Amazon RDS, Amazon Aurora, Amazon Redshift ou Amazon DocumentDB
- Configure a rotação automática para segredos que não sejam do banco de dados AWS Secrets Manager
- Configure a rotação automática usando o AWS CLI
- Estratégias de rotação da função Lambda
- Funções de rotação Lambda
- AWS Secrets Manager modelos de função de rotação
- Permissões da função de execução da função de rotação Lambda para AWS Secrets Manager
- Acesso à rede para a função de rotação Lambda
- Solucionar problemas de rotação AWS Secrets Manager

Configure a rotação automática para segredos do Amazon RDS, Amazon Aurora, Amazon Redshift ou Amazon DocumentDB

Este tutorial descreve como configurar segredos the section called “Rotação por função Lambda” de banco de dados. Alternância é o processo de atualizar periodicamente um segredo. Quando o Secrets Manager alterna um segredo, você atualiza as credenciais tanto no segredo como no banco de dados. No Secrets Manager, você pode configurar a alternância automática para seus segredos de banco de dados.

Para configurar a alternância usando o console, você precisa primeiro escolher uma estratégia de alternância. Em seguida, você configura o segredo para a alternância, o qual cria uma função de alternância do Lambda, caso você ainda não tenha uma. O console também define permissões para a função de execução da função Lambda. A última etapa é garantir que a função de alternância do Lambda possa acessar o Secrets Manager e seu banco de dados utilizando a rede.

> **Warning**
>
> Para ativar a rotação automática, você deve ter permissão para criar uma função de execução do IAM para a função de rotação do Lambda e anexar uma política de permissão a ela. Ambas as permissões `iam:CreateRole` e `iam:AttachRolePolicy` são necessárias. A concessão dessas permissões permite que uma identidade conceda a si mesma qualquer permissão.

Etapas:

Etapa 1: escolher uma estratégia de alternância e (opcionalmente) criar um segredo de superusuário

Para obter informações sobre as estratégias oferecidas pelo Secrets Manager, consultethe section called “Estratégias de rotação da função Lambda”.

Se você escolher a estratégia de usuários alternados, deverá Criar um segredo de banco de dados e armazenar nele as credenciais de superusuário do banco de dados. Você precisa de um segredo com credenciais de superusuário porque a alternância clona o primeiro usuário e a maioria dos usuários não tem essa permissão. Observe que o Amazon RDS Proxy não suporta a estratégia de usuários alternados.

Etapa 2: configurar a alternância e criar uma função de alternância

Para ativar a alternância para um segredo do Amazon RDS, do Amazon DocumentDB ou do Amazon Redshift

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na página Secrets (Segredos), escolha o segredo.
3. Na página Secret details (Detalhes do segredo), na seção Rotation configuration (Configuração da alternância), escolha Edit rotation (Editar alternância).
4. Na caixa de diálogo Edit rotation configuration (Editar configuração da alternância), siga estas etapas:

    a. Ative a Automatic rotation (Alternância automática).

    b. Em Rotation schedule (Programação da alternância), insira sua programação no fuso horário UTC no Schedule expression builder (Desenvolvedor de expressão programada) ou como uma Schedule expression (Expressão programada). O Secrets Manager armazena

sua programação como uma expressão `rate()` ou `cron()`. A janela de alternância começa automaticamente à 0h, a menos que você especifique um horário de início. Você pode alternar um segredo com intervalos a partir de quatro horas. Para ter mais informações, consulte Cronogramas de rotação.

c. (Opcional) Em Window duration (Duração da janela), escolha a duração da janela em que deseja que o Secrets Manager alterne o seu segredo, por exemplo, **3h**, por uma janela de três horas. A janela não pode se estender até a próxima janela de alternância. Se você não especificar Window duration (Duração da janela) para uma programação de alternância em horas, a janela será automaticamente encerrada após uma hora. Para uma programação de alternância em dias, a janela terminará automaticamente no final do dia.

d. (Opcional) Escolha Rotate immediately when the secret is stored (Alternar imediatamente quando o segredo for armazenado) para alternar o seu segredo assim que as suas alterações forem salvas. Se você desmarcar a caixa de seleção, a primeira alternância começará no cronograma definido.

   Se a alternância falhar, por exemplo, porque as etapas 3 e 4 ainda não foram concluídas, o Secrets Manager repetirá o processo de alternância várias vezes.

e. Em Rotation function (Função de alternância), execute uma das ações a seguir:

   - Selecione Create a new Lambda function (Criar uma nova função Lambda) e insira um nome para sua nova função. O Secrets Manager adiciona `SecretsManager` ao início do nome da função. O Secrets Manager cria a função com base no modelo apropriado e define as permissões necessárias para a função de execução do Lambda.
   - Escolha Use an existing Lambda function (Usar uma função Lambda existente) para reutilizar uma função de alternância usada para outro segredo. As funções de alternância listadas em Recommended VPC configurations (Configurações de VPC recomendadas) têm a mesma VPC e o mesmo grupo de segurança que o banco de dados, o que ajuda a função a acessar o banco de dados.

f. Em Estratégia de alternância, escolha a estratégia de Usuário único ou de Usuários alternados. Para ter mais informações, consulte the section called “Etapa 1: escolher uma estratégia de alternância e (opcionalmente) criar um segredo de superusuário”.

5. Escolha Salvar.

Etapa 3: (Opcional) Defina condições de permissões adicionais na função de alternância

Na política de recursos para sua função de alternância, recomendamos incluir a chave de contexto `aws:SourceAccount` para ajudar a evitar que o Lambda seja usado como um confused deputy. Para alguns AWS serviços, para evitar o cenário confuso de substituto, AWS recomenda que você use as chaves de condição `aws:SourceArn` e as chaves de condição `aws:SourceAccount` globais. No entanto, se você incluir a condição `aws:SourceArn` em sua política de função de alternância, a função de alternância só poderá ser usada para alternar o segredo especificado por esse ARN. Recomendamos que inclua apenas a chave de contexto `aws:SourceAccount`, de modo que possa usar a função de alternância para vários segredos.

Para atualizar sua política de recursos da função de alternância

1. No console do Secrets Manager, escolha seu segredo e, em seguida, na página de detalhes, em Rotation configuration (Configuração de alternância), escolha a função de alternância do Lambda. Abra o console do Lambda.
2. Siga as instruções em Usar políticas baseadas em recursos para o Lambda para adicionar uma condição `aws:sourceAccount`.

```
"Condition": {
    "StringEquals": {
        "AWS:SourceAccount": "123456789012"
    }
},
```

Se o segredo for criptografado com uma chave KMS diferente da Chave gerenciada pela AWS `aws/secretsmanager`, o Secrets Manager concede permissão à função de execução do Lambda para usar a chave. Você pode usar o contexto de criptografia SecretARN para limitar o uso da função de descriptografia, de modo que a função de alternância tenha acesso apenas para descriptografar o segredo que é responsável pela alternância.

Para atualizar o papel de execução da função de alternância

1. Na função de alternância do Lambda, escolha Configuração e, em Função de execução, escolha o Nome da função.

2. Siga as instruções em Modifying a role permissions policy (Modificar uma política de permissões de função) para adicionar uma condição `kms:EncryptionContext:SecretARN`.

```
"Condition": {
    "StringEquals": {
        "kms:EncryptionContext:SecretARN": "SecretARN"
    }
},
```

Etapa 4: configurar acesso à rede para a função de alternância

Para ter mais informações, consulte the section called “Acesso à rede para a função de rotação Lambda”.

Próximas etapas

Consulte the section called “Solução de problemas de alternância do ”.

Configure a rotação automática para segredos que não sejam do banco de dados AWS Secrets Manager

Este tutorial descreve como configurar segredos que não sejam the section called “Rotação por função Lambda” de banco de dados. Alternância é o processo de atualizar periodicamente um segredo. Ao alternar um segredo, você atualiza as credenciais no segredo e no banco de dados ou serviço para o qual o segredo se destina.

Para obter informações sobre segredos do banco de dados, consulte Alternância automática para segredos de banco de dados (console).

> **Warning**
>
> Para ativar a rotação automática, você deve ter permissão para criar uma função de execução do IAM para a função de rotação do Lambda e anexar uma política de permissão a ela. Ambas as permissões `iam:CreateRole` e `iam:AttachRolePolicy` são necessárias. A concessão dessas permissões permite que uma identidade conceda a si mesma qualquer permissão.

Etapas:

Etapa 1: criar uma função de rotação genérica

Para começar, crie uma função de rotação Lambda. Ele não terá o código para alternar seu segredo, então você escreverá isso em uma etapa posterior. Para obter informações sobre como uma função de rotação funciona, consultethe section called “Funções de rotação Lambda”.

Nas regiões suportadas, você pode usar AWS Serverless Application Repository para criar a função a partir de um modelo. Para obter uma lista das regiões suportadas, consulte as AWS Serverless Application Repository perguntas frequentes. Em outras regiões, você cria a função do zero e copia o código do modelo na função.

Para criar uma função de rotação genérica

1. Para determinar se AWS Serverless Application Repository é compatível com sua região, consulte AWS Serverless Application Repository endpoints e cotas na Referência AWS geral.
2. Execute um destes procedimentos:
   - Se AWS Serverless Application Repository houver suporte em sua região:
     a. No console Lambda, escolha Aplicativos e, em seguida, escolha Criar aplicativo.
     b. Na página Criar aplicativo, escolha a guia Aplicativo sem servidor.
     c. Na caixa de pesquisa, em Aplicativos públicos, digite**`SecretsManagerRotationTemplate`**.
     d. Selecione Mostrar aplicativos que criam funções personalizadas do IAM ou políticas de recursos.
     e. Escolha o SecretsManagerRotationTemplateladrilho.

f. Na página Revisar, configurar e implantar, no quadro Configurações do aplicativo, preencha os campos obrigatórios.

- Para endpoint, insira o endpoint da sua região, inclusive. **`https://`** Para uma lista de endpoints , consulte the section called “Endpoint do Secrets Manager”.
- Para colocar a função Lambda em uma VPC, inclua Ids e. vpcSecurityGroup vpcSubnetIds

g. Escolha Implantar.

- Se AWS Serverless Application Repository não for suportado em sua região:

a. No console Lambda, escolha Funções e, em seguida, escolha Criar função.

b. Na página Create function (Criar função), faça o seguinte:

i. Escolha Criar do zero.

ii. Em Function name (Nome da função), insira um nome para sua função de alternância.

iii. Em Runtime, escolha Python 3.9.

iv. Escolha a opção Criar função.

Etapa 2: programar o código da função de alternância

Nesta etapa, você escreve o código que atualiza o segredo e o serviço ou banco de dados para o qual o segredo se destina. Para obter informações sobre o que uma função de rotação faz, incluindo dicas sobre como escrever sua própria função de rotação, consultethe section called “Funções de rotação Lambda”. Você também pode usar o Modelos de função de alternância como referência.

Etapa 3: configurar o segredo para rotação

Nesta etapa, você define um cronograma de rotação para seu segredo e conecta a função de rotação ao segredo.

Para configurar a alternância e criar uma função de alternância em branco

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Na página Secrets (Segredos), escolha o segredo.

3. Na página Secret details (Detalhes do segredo), na seção Rotation configuration (Configuração da alternância), escolha Edit rotation (Editar alternância). Na caixa de diálogo Edit rotation configuration (Editar configuração da alternância), siga estas etapas:

   a. Ative a Automatic rotation (Alternância automática).

   b. Em Rotation schedule (Programação da alternância), insira sua programação no fuso horário UTC no Schedule expression builder (Desenvolvedor de expressão programada) ou como uma Schedule expression (Expressão programada). O Secrets Manager armazena sua programação como uma expressão `rate()` ou `cron()`. A janela de alternância começa automaticamente à 0h, a menos que você especifique um horário de início. Você pode alternar um segredo com intervalos a partir de quatro horas. Para ter mais informações, consulte Cronogramas de rotação.

   c. (Opcional) Em Window duration (Duração da janela), escolha a duração da janela em que deseja que o Secrets Manager alterne o seu segredo, por exemplo, **3h**, por uma janela de três horas. A janela não pode se estender até a próxima janela de alternância. Se você não especificar Window duration (Duração da janela) para uma programação de alternância em horas, a janela será automaticamente encerrada após uma hora. Para uma programação de alternância em dias, a janela terminará automaticamente no final do dia.

   d. (Opcional) Escolha Rotate immediately when the secret is stored (Alternar imediatamente quando o segredo for armazenado) para alternar o seu segredo assim que as suas alterações forem salvas. Se você desmarcar a caixa de seleção, a primeira alternância começará no cronograma definido.

   e. Em Função de rotação, escolha a função Lambda que você criou na Etapa 1.

   f. Escolha Salvar.

Etapa 4: Permitir que a função de rotação acesse o Secrets Manager e seu banco de dados ou serviço

A função de alternância do Lambda necessita de permissão para acessar o segredo no Secrets Manager e precisa de permissão para acessar seu banco de dados ou serviço. Nesta etapa, você concede essas permissões à função de execução do Lambda. Se o segredo for criptografado com uma chave KMS diferente da Chave gerenciada pela AWS `aws/secretsmanager`, você precisará conceder à função de execução do Lambda permissão para usar a chave. Você pode usar o contexto de criptografia SecretARN para limitar o uso da função de descriptografia, de modo que a função de rotação tenha acesso apenas para descriptografar o segredo que é responsável pela rotação. Para obter exemplos de políticas, consulte Permissões para alternância.

Para obter instruções, consulte Função de execução do Lambda no Guia do desenvolvedor do AWS Lambda .

Etapa 5: Permitir que o Secrets Manager invoque a função de rotação

Para permitir que o Secrets Manager invoque a função de rotação na programação de rotação que você configurou, você precisa conceder `lambda:InvokeFunction` permissão ao diretor do serviço Secrets Manager na política de recursos da função Lambda.

Na política de recursos para sua função de alternância, recomendamos incluir a chave de contexto `aws:SourceAccount` para ajudar a evitar que o Lambda seja usado como um confused deputy. Para alguns AWS serviços, para evitar o cenário confuso de substituto, AWS recomenda que você use as chaves de condição `aws:SourceArn` e as chaves de condição `aws:SourceAccount` globais. No entanto, se você incluir a condição `aws:SourceArn` em sua política de função de alternância, a função de alternância só poderá ser usada para alternar o segredo especificado por esse ARN. Recomendamos que inclua apenas a chave de contexto `aws:SourceAccount`, de modo que possa usar a função de alternância para vários segredos.

Para anexar uma política de recursos a uma função Lambda, consulte Usar políticas baseadas em recursos para o Lambda.

A política a seguir permite que o Secrets Manager invoque uma função Lambda.

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

```
}
```

Etapa 6: configurar o acesso à rede para a função de rotação

Nesta etapa, você permite que a função de rotação se conecte ao Secrets Manager e ao serviço ou banco de dados ao qual o segredo se destina. A função de rotação deve ter acesso a ambas para poder girar o segredo. Consulte the section called “Acesso à rede para a função de rotação Lambda”.

Próximas etapas

Ao configurar a rotação na Etapa 3, você define um cronograma para a rotação do segredo. Se a rotação falhar quando estiver programada, o Secrets Manager tentará a rotação várias vezes. Você também pode iniciar uma rotação imediatamente seguindo as instruções emAlternar um segredo imediatamente.

Se a rotação falhar, consulteSolução de problemas de alternância do .

Configure a rotação automática usando o AWS CLI

Este tutorial descreve como configurar the section called “Rotação por função Lambda” usando AWS CLI o. Ao alternar um segredo, você atualiza as credenciais no segredo e no banco de dados ou serviço para o qual o segredo se destina.

Você também pode configurar a rotação usando o console. Para obter informações sobre segredos do banco de dados, consulte Alternância automática para segredos de banco de dados (console). Para todos os outros tipos de segredos, consulte Rotação automática para segredos que não são do banco de dados (console).

Para configurar a rotação usando o AWS CLI, se você estiver rotacionando um segredo de banco de dados, primeiro você precisa escolher uma estratégia de rotação. Se você escolher a estratégia de usuários alternados, deverá armazenar um segredo separado com credenciais para um superusuário do banco de dados. Em seguida, você programa o código da função de alternância. O Secrets Manager fornece modelos nos quais você pode basear sua função. Em seguida, você cria uma função Lambda com seu código e define permissões para a função Lambda e para a função de execução do Lambda. A próxima etapa é garantir que a função Lambda possa acessar o Secrets Manager e seu banco de dados ou serviço pela rede. Por fim, você configura o segredo para a alternância.

Etapas:

Pré-requisito para segredos do banco de dados: escolha uma estratégia de rotação

Para obter informações sobre as estratégias oferecidas pelo Secrets Manager, consultethe section called “Estratégias de rotação da função Lambda”.

Opção 1: estratégia de usuário único

Se você escolher a estratégia de usuário único, poderá continuar com a Etapa 1.

Opção 2: estratégia de usuários alternados

Se você escolher a estratégia de usuários alternados, deverá:

- Crie um segredo de banco de dados e armazene nele as credenciais de superusuário do banco de dados. Você precisa de um segredo com as credenciais de superusuário porque a rotação alternada de usuários clona o primeiro usuário, e a maioria dos usuários não tem essa permissão.
- Adicione o ARN do segredo do superusuário ao segredo original. Para ter mais informações, consulte the section called “Estrutura JSON de um segredo”.

Observe que o Amazon RDS Proxy não suporta a estratégia de usuários alternados.

Etapa 1: escrever o código da função de rotação

Para alternar um segredo, você precisa de uma função de alternância. Uma função de alternância corresponde a uma função Lambda a qual o Secrets Manager chama para alternar o segredo. Para ter mais informações, consulte the section called “Rotação por função Lambda”. Nesta etapa, você escreve o código que atualiza o segredo e o serviço ou banco de dados para o qual o segredo se destina.

O Secrets Manager fornece modelos para os segredos dos bancos de dados Amazon RDS, Amazon Aurora, Amazon Redshift e Amazon DocumentDB em. Modelos de função de alternância

Para escrever o código da função de rotação

1. Execute um destes procedimentos:
   - Verifique a lista de modelos de função de rotação. Se houver uma que corresponda à sua estratégia de serviço e rotação, copie o código.
   - Para outros tipos de segredos, você escreve sua própria função de rotação. Para obter instruções, consulte the section called “Funções de rotação Lambda”.
2. Salve o arquivo em um arquivo ZIP *my-function.zip* junto com todas as dependências necessárias.

Etapa 2: Criar a função do Lambda

Nesta etapa, você cria a função Lambda usando o arquivo ZIP criado na Etapa 1. Você também define a função de execução do Lambda, que é a função que o Lambda assume quando a função é invocada.

Para criar uma função de alternância e uma função de execução do Lambda

1. Crie uma política de confiança para a função de execução do Lambda e salve-a como um arquivo JSON. Para obter exemplos e mais informações, consultethe section called “Permissões para alternância”. A política deve:
   - Permitir que a função chame as operações do Secrets Manager no segredo.
   - Permita que a função chame o serviço ao qual o segredo se destina, por exemplo, para criar uma nova senha.
2. Crie a função de execução do Lambda e aplique a política de confiança que você criou na etapa anterior chamando. `iam create-role`

```
aws iam create-role \
    --role-name rotation-lambda-role \
    --assume-role-policy-document file://trust-policy.json
```

3. Crie a função Lambda do arquivo ZIP chamando `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. Defina uma política de recursos na função Lambda para permitir que o Secrets Manager a invoque chamando `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
```

Etapa 3: configurar o acesso à rede

Para ter mais informações, consulte the section called “Acesso à rede para a função de rotação Lambda”.

Etapa 4: configurar o segredo para rotação

Para ativar a alternância automática do seu segredo, determine `rotate-secret`. Você pode definir uma programação de alternância com uma expressão de programação `cron()` ou `rate()` e pode definir a duração da janela de alternância. Para ter mais informações, consulte the section called “Cronogramas de rotação”.

```
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\"}"
```

Próximas etapas

Consulte the section called “Solução de problemas de alternância do ”.

Estratégias de rotação da função Lambda

Poisthe section called “Rotação por função Lambda”, para segredos de banco de dados, o Secrets Manager oferece duas estratégias de rotação.

Estratégia de alternância: usuário único

Essa estratégia atualiza as credenciais de um único usuário em um segredo. Para instâncias do Db2 do Amazon RDS, como os usuários não podem alterar suas próprias senhas, você deve fornecer credenciais de administrador em outro segredo. Essa é a estratégia de alternância mais simples, e é apropriada para a maioria dos casos de uso. Em particular, recomendamos que você use essa estratégia para obter credenciais para usuários únicos (ad hoc) ou interativos.

Quando o segredo é alternado, as conexões de banco de dados abertas não são descartadas. Enquanto a alternância acontece, há um curto período entre a alteração da senha no banco de dados e a atualização do segredo. Durante esse período, há um baixo risco de o banco de dados recusar chamadas que usam as credenciais alternadas. Você pode mitigar esse risco com uma estratégia de nova tentativa apropriada. Após a alternância, as novas conexões usam as novas credenciais.

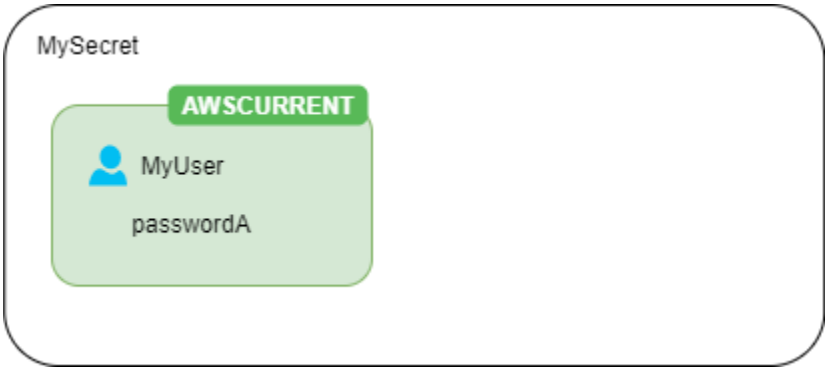
Estratégia de alternância: usuários alternados

Essa estratégia atualiza as credenciais de dois usuários em um segredo. Você cria o primeiro usuário e, durante a primeira alternância, a função de alternância o clona para criar o segundo usuário. Toda vez que o segredo é alternado, a função de alternância alterna a senha do usuário que ela atualiza. Como a maioria dos usuários não tem permissão para se clonar, você deve fornecer as credenciais para um `superuser` em outro segredo. Recomendamos o uso da estratégia de alternância de usuário único quando os usuários clonados em seu banco de dados não tiverem as mesmas permissões que o usuário original, e para credenciais de usuários únicos (ad hoc) ou interativos.

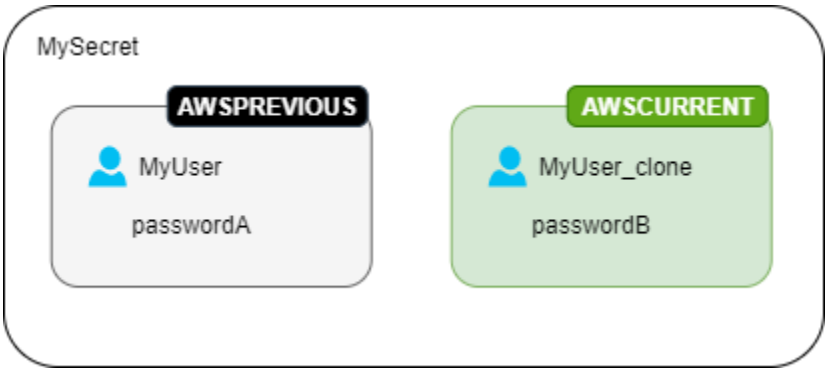
Essa estratégia é apropriada para bancos de dados com modelos de permissão em que uma função possui as tabelas de banco de dados e uma segunda função tem permissão para acessar as tabelas de banco de dados. Ela também é apropriada para aplicações que requerem alta disponibilidade. Se uma aplicação recuperar o segredo durante a alternância, a aplicação ainda obterá um conjunto válido de credenciais. Após a alternância, as credenciais do `user` e do `user_clone` são válidas. Há ainda menos chances de aplicações serem recusadas durante esse tipo de alternância em comparação com a alternância de usuário único. Se o banco de dados estiver hospedado em um farm de servidores em que a alteração de senha demora algum tempo para se propagar para todos os servidores, existe o risco de o banco de dados recusar chamadas que usam as novas credenciais. Você pode mitigar esse risco com uma estratégia de nova tentativa apropriada.

O Secrets Manager cria o usuário clonado com as mesmas permissões do usuário original. Se você alterar as permissões do usuário original após a criação do clone, também deverá alterar as permissões do usuário clonado.

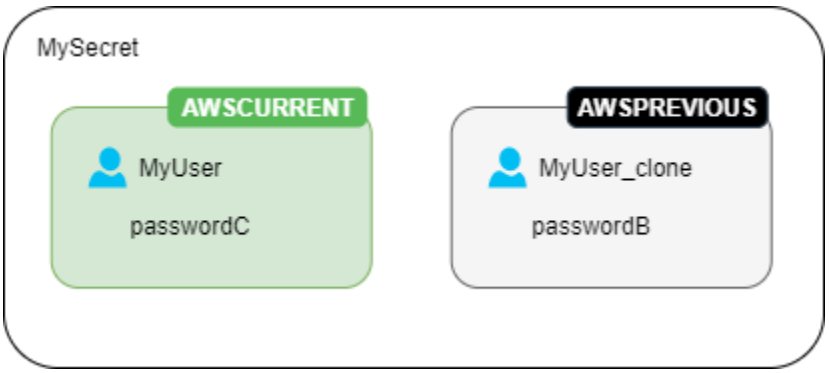
Por exemplo, se você criar um segredo com as credenciais de um usuário do banco de dados, o segredo conterá uma versão com essas credenciais.



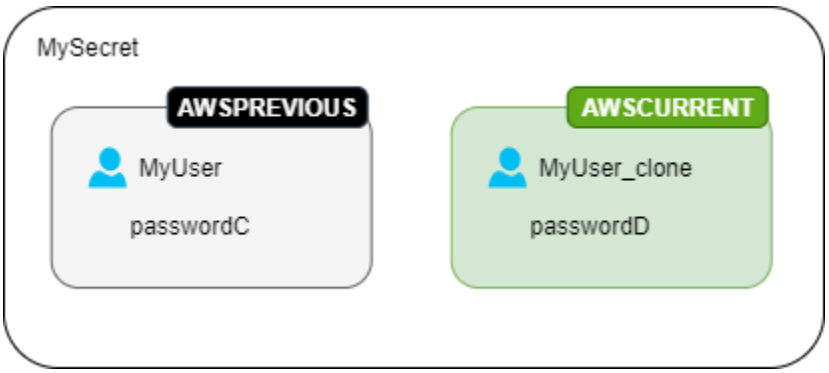
Primeira alternância: a função de alternância cria um clone do seu usuário com uma senha gerada e essas credenciais se tornam a versão secreta atual.



Segunda alternância: a função de alternância atualizada a senha do usuário original.



Terceira alternância: a função de alternância atualizada a senha do usuário clonado.



Funções de rotação Lambda

Emthe section called “Rotação por função Lambda”, uma função Lambda faz o trabalho de girar o segredo. O Secrets Manager usa rótulos de preparação prévia para rotular as versões de um segredo durante a alternância.

Se o Secrets Manager não fornecer um modelo de função de rotação para seu tipo de segredo, você poderá criar uma função de rotação. Ao escrever uma função de rotação, siga as orientações de cada etapa.

Dicas para escrever sua própria função de rotação

- Use o modelo de rotação genérico como ponto de partida para escrever sua própria função de rotação.
- Ao escrever sua função, tenha cuidado quanto à inclusão de instruções de depuração ou registro em log. Essas declarações podem fazer com que as informações em sua função sejam gravadas na Amazon CloudWatch, então você precisa garantir que o registro não inclua nenhuma informação confidencial coletada durante o desenvolvimento.

  Para exemplos de instruções de log, consulte o código-fonte dos the section called “Modelos de função de alternância”.
- Por motivos de segurança, o Secrets Manager só permite que uma função de alternância do Lambda alterne o segredo diretamente. A função de alternância não pode chamar uma segunda função do Lambda para alternar o segredo.
- Para sugestões de depuração, consulte Testing and debugging serverless applications (Teste e depuração de aplicações com tecnologia sem servidor).
- Se você usa binários e bibliotecas externos, por exemplo, para se conectar a um recurso, deverá gerenciar a aplicação de patches e a manutenção deles. up-to-date

- Salve sua função de alternância em um arquivo ZIP denominado *my-function.zip* junto com quaisquer dependências necessárias.

Quatro etapas em uma função de rotação

Tópicos

`create_secret`: Crie uma nova versão do segredo

O método `create_secret` primeiro verifica se existe um segredo chamando `get_secret_value`com o passado. `ClientRequestToken` Se não houver nenhum segredo, ele cria um novo segredo com `create_secret`e o token como `VersionId` o. Em seguida, ele gera um novo valor secreto com `get_random_password`. Em seguida, ele liga `put_secret_value`para armazená-lo com a etiqueta `AWSPENDING` de teste. Armazenar o novo valor do segredo em `AWSPENDING` ajuda a garantir a idempotência. Se a alternância falhar por qualquer motivo, você poderá consultar esse valor de segredo em chamadas subsequentes. Consulte How do I make my Lambda function idempotent? (Como torno minha função Lambda idempotente?).

Dicas para escrever sua própria função de rotação

- Certifique-se de que o novo valor secreto inclua somente caracteres válidos para o banco de dados ou serviço. Exclua caracteres usando o parâmetro `ExcludeCharacters`.
- Ao testar sua função, use o AWS CLI para ver os estágios da versão: chame `describe-secret`e examine`VersionIdsToStages`.
- Para o Amazon RDS MySQL, alternando a rotação de usuários, o Secrets Manager cria um usuário clonado com um nome de no máximo 16 caracteres. Você pode modificar a função de rodízio para permitir nomes de usuário mais longos. A versão 5.7 e superior do MySQL é compatível com nomes de usuário de até 32 caracteres. No entanto, o Secrets Manager acrescenta “_clone” (seis caracteres) ao final do nome de usuário, portanto, você deve manter o nome de usuário com no máximo 26 caracteres.

`set_secret`: altere as credenciais no banco de dados ou serviço

O método `set_secret` altera a credencial no banco de dados ou serviço para corresponder ao novo valor secreto na `AWSPENDING` versão do segredo.

Dicas para escrever sua própria função de rotação

- Se você passar instruções para um serviço que interpreta instruções, como um banco de dados, use a parametrização de consultas. Para obter mais informações, consulte a Folha de dicas de parametrização de consultas no site da OWASP.
- A função de alternância é um representante privilegiado que tem autorização para acessar e modificar as credenciais do cliente no segredo do Secrets Manager e no recurso de destino. Para evitar um possível ataque "confused deputy", você precisa garantir que um invasor não possa usar a função para acessar outros recursos. Antes de atualizar a credencial:
  - Verifique se a credencial na versão `AWSCURRENT` do segredo é válida. Se a credencial `AWSCURRENT` não for válida, abandone a tentativa de alternância.
  - Verifique se os valores dos segredos `AWSCURRENT` e `AWSPENDING` são para o mesmo recurso. Para obter um nome de usuário e uma senha, verifique se os nomes de usuário `AWSCURRENT` e `AWSPENDING` são os mesmos.
  - Verifique se o recurso do serviço de destino é o mesmo. Para um banco de dados, verifique se os nomes de host `AWSCURRENT` e `AWSPENDING` são os mesmos.
- Em casos raros, talvez você queira personalizar uma função de rotação existente para um banco de dados. Por exemplo, com o rodízio de usuários em alternância, o Secrets Manager cria o usuário clonado copiando os parâmetros de configuração de runtime do primeiro usuário. Se você quiser incluir mais atributos ou alterar quais são concedidos ao usuário clonado, é necessário atualizar o código na função `set_secret`.

`test_secret`: Teste a nova versão secreta

Em seguida, a função de alternância do Lambda testará a versão `AWSPENDING` do segredo usando-a para acessar o banco de dados ou serviço. As funções de alternância baseadas em Modelos de função de alternância testam o novo segredo usando o acesso de leitura.

`finish_secret`: Concluir a rotação

Por fim, a função de alternância do Lambda move o rótulo `AWSCURRENT` da versão anterior do segredo para a atual, o que também remove o rótulo `AWSPENDING` na mesma chamada de API. O

Secrets Manager adiciona o rótulo de preparação prévia AWSPREVIOUS na versão anterior, para que você retenha a última versão boa conhecida do segredo.

O método finish_secret usado update_secret_version_stagepara mover o rótulo de teste AWSCURRENT da versão secreta anterior para a nova versão secreta. O Secrets Manager adiciona automaticamente o rótulo de preparação AWSPREVIOUS à versão anterior para reter a última versão válida do segredo.

Dicas para escrever sua própria função de rotação

- Não remova AWSPENDING antes desse ponto e não a remova usando uma chamada de API separada, pois isso pode indicar ao Secrets Manager que a rotação não foi concluída com êxito. O Secrets Manager adiciona o rótulo de preparação prévia AWSPREVIOUS na versão anterior, para que você retenha a última versão boa conhecida do segredo.

Quando a alternância for bem-sucedida, talvez o rótulo de preparação de AWSPENDING seja anexado à mesma versão da versão AWSCURRENT ou talvez não seja anexado a nenhuma versão. Se o rótulo de preparação de AWSPENDING estiver presente, mas não estiver anexado à mesma versão de AWSCURRENT, qualquer invocação posterior de alternância vai pressupor que uma solicitação de alternância anterior ainda está em andamento e retornará um erro. Quando a alternância não for bem-sucedida, o rótulo de preparação de AWSPENDING poderá ser anexado a uma versão vazia de segredo. Para ter mais informações, consulte Solução de problemas de alternância do .

AWS Secrets Manager modelos de função de rotação

Parathe section called “Rotação por função Lambda”, o Secrets Manager fornece vários modelos de funções de rotação. Para usar os modelos, consulte:

- Alternância automática para segredos de banco de dados (console)
- Rotação automática para segredos que não são do banco de dados (console)

Os modelos suportam o Python 3.9.

Para escrever sua própria função de rotação, consulte Escrever uma função de rotação.

Modelos

- Amazon RDS e Amazon Aurora
  - Usuário único do Db2 do Amazon RDS

- Usuários em alternância do Db2 do Amazon RDS
- Usuário único do MariaDB do Amazon RDS
- Usuários alternados do MariaDB do Amazon RDS
- Usuário único do Amazon RDS e do Amazon Aurora MySQL
- Usuários em alternância do Amazon RDS e do Amazon Aurora MySQL
- Usuário único do Oracle do Amazon RDS
- Usuários alternados do Oracle do Amazon RDS
- Usuário único do Amazon RDS e do Amazon Aurora PostgreSQL
- Usuários em alternância do Amazon RDS e do Amazon Aurora PostgreSQL
- Usuário único do Microsoft SQLServer do Amazon RDS
- Usuários alternados do Microsoft SQLServer do Amazon RDS
- Amazon DocumentDB (compatível com MongoDB)
  - Usuário único do Amazon DocumentDB
  - Usuários alternados do Amazon DocumentDB
- Amazon Redshift
  - Usuário único do Amazon Redshift
  - Usuários alternados do Amazon Redshift
- Amazon ElastiCache
- Active Directory
  - Credenciais do Active Directory
  - Tecla do Active Directory
- Outros tipos de segredo

Amazon RDS e Amazon Aurora

Usuário único do Db2 do Amazon RDS

- Nome do modelo: SecretsManager RDSdb2 RotationSingleUser
- Estratégia de alternância: Estratégia de alternância: usuário único.
- Estrutura da **`SecretString`**: the section called “Estrutura de segredos do Db2 do Amazon RDS”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/master/ RdsDb2 /lambda_function.py SecretsManager RotationSingleUser

- Dependência: python-ibmdb

Usuários em alternância do Db2 do Amazon RDS

- Nome do modelo: SecretsManager RDSdb2 RotationMultiUser
- Estratégia de alternância: the section called “Usuários alternados”.
- Estrutura da **`SecretString`**: the section called “Estrutura de segredos do Db2 do Amazon RDS”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ RdsDb2 /lambda_function.py SecretsManager RotationMultiUser
- Dependência: python-ibmdb

Usuário único do MariaDB do Amazon RDS

- Nome do modelo: SecretsManager RDSMariaDB RotationSingleUser
- Estratégia de alternância: Estratégia de alternância: usuário único.
- Estrutura da **`SecretString`**: the section called “Estrutura do segredo MariaDB do Amazon RDS”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ RDSMariaDB /lambda_function.py SecretsManager RotationSingleUser
- Dependência: PyMy SQL 1.0.2. Se você usar a senha sha256 para autenticação, PyMy SQL [rsa]. Para obter informações sobre o uso de pacotes com código compilado em um tempo de execução do Lambda, consulte Como faço para adicionar pacotes Python com binários compilados ao meu pacote de implantação e tornar o pacote compatível com o Lambda? no Centro de AWS Conhecimento.

Usuários alternados do MariaDB do Amazon RDS

- Nome do modelo: SecretsManager RDSMariaDB RotationMultiUser
- Estratégia de alternância: the section called “Usuários alternados”.
- Estrutura da **`SecretString`**: the section called “Estrutura do segredo MariaDB do Amazon RDS”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ RDSMariaDB /lambda_function.py SecretsManager RotationMultiUser
- Dependência: PyMy SQL 1.0.2. Se você usar a senha sha256 para autenticação, PyMy SQL [rsa]. Para obter informações sobre o uso de pacotes com código compilado em um tempo de execução

do Lambda, consulte Como faço para adicionar pacotes Python com binários compilados ao meu pacote de implantação e tornar o pacote compatível com o Lambda? no Centro de AWS Conhecimento.

Usuário único do Amazon RDS e do Amazon Aurora MySQL

- Nome do modelo: SecretsManager RDSMySQL RotationSingleUser
- Estratégia de alternância: the section called “Usuário único”.
- Estrutura esperada da **`SecretString`**: the section called “Estrutura de segredos do Amazon RDS e do Amazon Aurora MySQL”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/master/ RDSMySQL /lambda_function.py SecretsManager RotationSingleUser
- Dependência: PyMy SQL 1.0.2. Se você usar a senha sha256 para autenticação, PyMy SQL [rsa]. Para obter informações sobre o uso de pacotes com código compilado em um tempo de execução do Lambda, consulte Como faço para adicionar pacotes Python com binários compilados ao meu pacote de implantação e tornar o pacote compatível com o Lambda? no Centro de AWS Conhecimento.

Usuários em alternância do Amazon RDS e do Amazon Aurora MySQL

- Nome do modelo: SecretsManager RDSMySQL RotationMultiUser
- Estratégia de alternância: the section called “Usuários alternados”.
- Estrutura esperada da **`SecretString`**: the section called “Estrutura de segredos do Amazon RDS e do Amazon Aurora MySQL”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/master/ RDSMySQL /lambda_function.py SecretsManager RotationMultiUser
- Dependência: PyMy SQL 1.0.2. Se você usar a senha sha256 para autenticação, PyMy SQL [rsa]. Para obter informações sobre o uso de pacotes com código compilado em um tempo de execução do Lambda, consulte Como faço para adicionar pacotes Python com binários compilados ao meu pacote de implantação e tornar o pacote compatível com o Lambda? no Centro de AWS Conhecimento.

Usuário único do Oracle do Amazon RDS

- Nome do modelo: SecretsManager RDS OracleRotationSingleUser

- Estratégia de alternância: the section called “Usuário único”.
- Estrutura esperada da **`SecretString`**: the section called “Estrutura do segredo do Oracle do Amazon RDS”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManager - lambdas/tree/master/ RDS /lambda_function.py OracleRotationSingleUser
- Dependência: python-oracledb 2.0.1

Usuários alternados do Oracle do Amazon RDS

- Nome do modelo: SecretsManager RDS OracleRotationMultiUser
- Estratégia de alternância: the section called “Usuários alternados”.
- Estrutura esperada da **`SecretString`**: the section called “Estrutura do segredo do Oracle do Amazon RDS”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManager - lambdas/tree/master/ RDS /lambda_function.py OracleRotationMultiUser
- Dependência: python-oracledb 2.0.1

Usuário único do Amazon RDS e do Amazon Aurora PostgreSQL

- Nome do modelo: SecretsManager RDSPostgreSQL RotationSingleUser
- Estratégia de alternância: Estratégia de alternância: usuário único.
- Estrutura esperada da **`SecretString`**: the section called “Estrutura de segredos do Amazon RDS e do Amazon Aurora PostgreSQL”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/ master/ RDSPostgreSQL /lambda_function.py SecretsManager RotationSingleUser
- Dependência: PyGre SQL 5.0.7

Usuários em alternância do Amazon RDS e do Amazon Aurora PostgreSQL

- Nome do modelo: SecretsManager RDSPostgreSQL RotationMultiUser
- Estratégia de alternância: the section called “Usuários alternados”.
- Estrutura esperada da **`SecretString`**: the section called “Estrutura de segredos do Amazon RDS e do Amazon Aurora PostgreSQL”.

- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/master/ RDSPostgreSQL /lambda_function.py SecretsManager RotationMultiUser
- Dependência: PyGre SQL 5.0.7

Usuário único do Microsoft SQLServer do Amazon RDS

- Nome do modelo: SecretsManager RDSSQL ServerRotationSingleUser
- Estratégia de alternância: the section called “Usuário único”.
- Estrutura esperada da **`SecretString`**: the section called “Estrutura do segredo do Microsoft SQLServer do Amazon RDS”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/master/ RDSSQL /lambda_function.py SecretsManager ServerRotationSingleUser
- Dependência: Pymssql 2.2.2

Usuários alternados do Microsoft SQLServer do Amazon RDS

- Nome do modelo: SecretsManager RDSSQL ServerRotationMultiUser
- Estratégia de alternância: the section called “Usuários alternados”.
- Estrutura esperada da **`SecretString`**: the section called “Estrutura do segredo do Microsoft SQLServer do Amazon RDS”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation -lambdas/tree/master/ RDSSQL /lambda_function.py SecretsManager ServerRotationMultiUser
- Dependência: Pymssql 2.2.2

Amazon DocumentDB (compatível com MongoDB)

Usuário único do Amazon DocumentDB

- Nome do modelo: SecretsManagerMongo DB RotationSingleUser
- Estratégia de alternância: the section called “Usuário único”.
- Estrutura esperada da **`SecretString`**: the section called “Estrutura do segredo do Amazon DocumentDB”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerMongo -lambdas/tree/master/ DB /lambda_function.py RotationSingleUser

- Dependência: Pymongo 3.2

Usuários alternados do Amazon DocumentDB

- Nome do modelo: SecretsManagerMongo DB RotationMultiUser
- Estratégia de alternância: the section called “Usuários alternados”.
- Estrutura esperada da **`SecretString`**: the section called “Estrutura do segredo do Amazon DocumentDB”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerMongo -lambdas/tree/master/ DB /lambda_function.py RotationMultiUser
- Dependência: Pymongo 3.2

Amazon Redshift

Usuário único do Amazon Redshift

- Nome do modelo: SecretsManagerRedshiftRotationSingleUser
- Estratégia de alternância: the section called “Usuário único”.
- **`SecretString`**Estrutura esperada: the section called “Estrutura do segredo do Amazon Redshift” outhe section called “Estrutura secreta sem servidor do Amazon Redshift”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerRedshiftRotationSingleUser -lambdas/tree/master/ /lambda_function.py
- Dependência: PyGre SQL 5.0.7

Usuários alternados do Amazon Redshift

- Nome do modelo: SecretsManagerRedshiftRotationMultiUser
- Estratégia de alternância: the section called “Usuários alternados”.
- **`SecretString`**Estrutura esperada: the section called “Estrutura do segredo do Amazon Redshift” outhe section called “Estrutura secreta sem servidor do Amazon Redshift”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerRedshiftRotationMultiUser -lambdas/tree/master/ /lambda_function.py
- Dependência: PyGre SQL 5.0.7

Amazon ElastiCache

Para usar esse modelo, consulte Rotação automática de senhas para usuários no Guia do ElastiCache usuário da Amazon.

- Nome do modelo: SecretsManagerElasticacheUserRotation
- Estrutura esperada da **`SecretString`**: the section called “Estrutura ElastiCache secreta da Amazon”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerElasticacheUserRotation -lambdas/tree/master/ /lambda_function.py

Active Directory

Credenciais do Active Directory

- Nome do modelo: SecretsManagerActiveDirectoryRotationSingleUser
- Estrutura esperada da **`SecretString`**: the section called “Estrutura secreta de credenciais do Active Directory”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerActiveDirectoryRotationSingleUser -lambdas/tree/master/ /lambda_function.py

Tecla do Active Directory

- Nome do modelo: SecretsManagerActiveDirectoryAndKeytabRotationSingleUser
- Estrutura esperada da **`SecretString`**: the section called “Estruturas secretas do Active Directory”.
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerActiveDirectoryAndKeytabRotationSingleUser -lambdas/tree/master/ / lambda_function.py
- Dependências: msktutil

Outros tipos de segredo

O Secrets Manager fornece esse modelo como ponto de partida para você criar uma função de alternância para qualquer tipo de segredo.

- Nome do modelo: SecretsManagerRotationTemplate
- Código fonte: https://github.com/aws-samples/ aws-secrets-manager-rotation SecretsManagerRotationTemplate -lambdas/tree/master/ /lambda_function.py

Permissões da função de execução da função de rotação Lambda para AWS Secrets Manager

Poisthe section called “Rotação por função Lambda”, quando o Secrets Manager usa uma função Lambda para alternar um segredo, o Lambda assume uma função de execução do IAM e fornece essas credenciais para o código da função Lambda. Para obter instruções sobre como configurar a rotação automática, consulte:

- Alternância automática para segredos de banco de dados (console)
- Rotação automática para segredos que não são do banco de dados (console)
- Alternância automática (AWS CLI)

Os exemplos a seguir mostram políticas em linha para funções de execução da função de alternância do Lambda. Para criar uma função de execução e anexá-la a uma política de permissões, consulte AWS Lambda Função de execução.

Exemplos:

- Política para uma função de execução da função de alternância do Lambda
- Declaração de política para chaves gerenciadas pelo cliente
- Declaração de política para a estratégia de usuários alternados

Política para uma função de execução da função de alternância do Lambda

A política de exemplo a seguir permite que a função de alternância:

- Execute operações do Secrets Manager para o *SecretARN*
- Criar uma nova senha.
- Definir a configuração necessária se seu banco de dados ou serviço for executado em uma VPC. Consulte Configuring a Lambda function to access resources in a VPC (Configurar uma função do Lambda para acessar recursos em uma VPC).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecretVersionStage"
            ],
            "Resource": "SecretARN"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword"
            ],
            "Resource": "*"
        },
        {
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DetachNetworkInterface"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Declaração de política para chaves gerenciadas pelo cliente

Se o segredo for criptografado com uma chave KMS diferente da Chave gerenciada pela AWS `aws/secretsmanager`, você precisará conceder à função de execução do Lambda permissão para usar a chave. Você pode usar o contexto de criptografia SecretARN para limitar o uso da função de descriptografia, para que a função de alternância tenha acesso apenas para descriptografar o segredo que é responsável pela rotação. O exemplo a seguir mostra uma instrução a ser adicionada à política de função de execução para descriptografar o segredo usando a chave KMS.

```
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey",
                "kms:GenerateDataKey"
            ],
            "Resource": "KMSKeyARN"
            "Condition": {
                "StringEquals": {
                    "kms:EncryptionContext:SecretARN": "SecretARN"
                }
            }
        }
```

Para usar a função de alternância para vários segredos criptografados com uma chave gerenciada pelo cliente, adicione uma declaração como o exemplo a seguir para permitir que o perfil de execução decifre o segredo.

```
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey",
                "kms:GenerateDataKey"
            ],
            "Resource": "KMSKeyARN"
            "Condition": {
                "StringEquals": {
                    "kms:EncryptionContext:SecretARN": [
                        "arn1",
                        "arn2"
                    ]
                }
            }
        }
```

Declaração de política para a estratégia de usuários alternados

Para obter informações sobre a estratégia de alternância de usuários alternados, consulte the section called “Estratégias de rotação da função Lambda”.

Para um segredo que contém credenciais do Amazon RDS, se você estiver usando a estratégia de usuários alternativos e o segredo do superusuário for gerenciado pelo Amazon RDS, você também deverá permitir que a função de alternância chame APIs somente de leitura no Amazon RDS para que ela possa obter as informações de conexão do banco de dados. Recomendamos que você anexe a política AWS gerenciada ReadOnlyAccessAmazonRDS.

O exemplo de política a seguir permite que a função:

- Execute operações do Secrets Manager para o *SecretARN*
- Recupere as credenciais no segredo do superusuário. O Secrets Manager usa as credenciais no segredo do superusuário para atualizar as credenciais no segredo que está sendo alternado.
- Criar uma nova senha.
- Definir a configuração necessária se seu banco de dados ou serviço for executado em uma VPC. Para obter mais informações, consulte Configuração de uma função do Lambda para acessar recursos em uma VPC.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecretVersionStage"
            ],
            "Resource": "SecretARN"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": "SuperuserSecretARN"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword"
```

```
            ],
            "Resource": "*"
        },
        {
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DetachNetworkInterface"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Acesso à rede para a função de rotação Lambda

Poisthe section called “Rotação por função Lambda”, quando o Secrets Manager usa uma função Lambda para girar um segredo, a função de rotação do Lambda deve ser capaz de acessar o segredo. Se seu segredo contiver credenciais, a função Lambda também deverá poder acessar a fonte dessas credenciais, como um banco de dados ou serviço.

Para acessar um segredo

A função de alternância do Lambda deve ser capaz de acessar um endpoint do Secrets Manager. Se sua função do Lambda puder acessar a Internet, você pode usar um endpoint público. Para localizar um endpoint, consulte the section called “Endpoint do Secrets Manager”.

Se a função do Lambda for executada em uma VPC que não tem acesso à Internet, recomendamos que você configure endpoints privados de serviço do Secrets Manager dentro de sua VPC. Assim, sua VPC pode interceptar solicitações endereçadas ao endpoint regional público e redirecioná-las para o endpoint privado. Para ter mais informações, consulte Endpoint da VPC.

Como alternativa, você pode habilitar a função Lambda para acessar um endpoint público do Secrets Manager adicionando um gateway NAT ou um gateway da Internet à VPC, o que permite que o tráfego da VPC alcance o endpoint público. Isso expõe a VPC a um risco maior, pois um endereço IP do gateway pode ser atacado a partir da Internet pública.

(Opcional) para acessar o banco de dados ou o serviço

Para segredos como chaves de API, não há banco de dados ou serviço de origem que você precise atualizar junto com o segredo.

Se seu banco de dados ou serviço estiver sendo executado em uma instância do Amazon EC2 em uma VPC, é recomendável que você configure a função do Lambda para ser executada na mesma VPC. Assim, a função de alternância pode se comunicar diretamente com seu serviço. Para obter mais informações, consulte Configuring VPC access (Configurar o acesso à VPC).

Para permitir que a função do Lambda acesse o banco de dados ou serviço, certifique-se de que os grupos de segurança anexados à sua função de alternância do Lambda permitam conexões de saída com o banco de dados ou serviço. Você também deve garantir que os grupos de segurança anexados ao seu banco de dados ou serviço permitam conexões de entrada a partir da função de alternância do Lambda.

Solucionar problemas de rotação AWS Secrets Manager

Em muitos serviços, o Secrets Manager usa uma função do Lambda para alternar segredos. Para ter mais informações, consulte the section called “Rotação por função Lambda”. A função de alternância do Lambda interage com o banco de dados ou serviço para o qual o segredo se destina, bem como o Secrets Manager. Quando a rotação não funciona da maneira esperada, você deve primeiro verificar os CloudWatch registros.

> **Note**
>
> Alguns serviços podem gerenciar segredos para você, incluindo o gerenciamento da alternância automática. Para ter mais informações, consulte the section called “Alternância gerenciada”.

Para visualizar os CloudWatch registros da sua função Lambda

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Escolha seu segredo e, em seguida, na página de detalhes, em Rotation configuration (Configuração de alternância), escolha a função de alternância do Lambda. Abra o console do Lambda.
3. Na guia Monitorar, escolha Registros e, em seguida, selecione Exibir registros CloudWatch.

O CloudWatch console é aberto e exibe os registros da sua função.

Para interpretar os logs

- Nenhuma atividade após “Found credentials in environment variables” (Credenciais encontradas em variáveis de ambiente)
- Nenhuma atividade após “createSecret”
- Erro: “O acesso ao KMS não é permitido”
- Erro: “Key is missing from secret JSON” (A chave do segredo JSON está ausente)
- Erro: “setSecret: Unable to log into database” (setSecret: não é possível fazer login no banco de dados)
- Erro: “Não é possível importar o módulo ‘lambda_function’"
- Atualize uma função de alternância existente do Python 3.7 para 3.9

Nenhuma atividade após “Found credentials in environment variables” (Credenciais encontradas em variáveis de ambiente)

Se não houver atividade após “Found credentials in environment variables” (Credenciais encontradas em variáveis de ambiente) e a duração da tarefa for longa, por exemplo, o tempo limite padrão do Lambda de 30 mil milissegundos, a função Lambda pode estar expirando ao tentar alcançar o endpoint do Secrets Manager.

A função de alternância do Lambda deve ser capaz de acessar um endpoint do Secrets Manager. Se sua função do Lambda puder acessar a Internet, você pode usar um endpoint público. Para localizar um endpoint, consulte the section called “Endpoint do Secrets Manager”.

Se a função do Lambda for executada em uma VPC que não tem acesso à Internet, recomendamos que você configure endpoints privados de serviço do Secrets Manager dentro de sua VPC. Assim, sua VPC pode interceptar solicitações endereçadas ao endpoint regional público e redirecioná-las para o endpoint privado. Para ter mais informações, consulte Endpoint da VPC.

Como alternativa, você pode habilitar a função Lambda para acessar um endpoint público do Secrets Manager adicionando um gateway NAT ou um gateway da Internet à VPC, o que permite que o tráfego da VPC alcance o endpoint público. Isso expõe a VPC a um risco maior, pois um endereço IP do gateway pode ser atacado a partir da Internet pública.

Nenhuma atividade após “createSecret”

A seguir, estão os problemas que podem fazer com que a alternância pare após createSecret:

As ACLs de rede VPC não permitem o tráfego HTTPS de entrada e saída.

Para obter mais informações, consulte Controlar o tráfego para sub-redes com ACLs de rede no Guia do usuário da Amazon VPC.

A configuração de tempo limite da função Lambda é muito curta para executar a tarefa.

Para obter mais informações, consulte Configurar as opções da função Lambda no Guia do desenvolvedor do AWS Lambda .

O endpoint da VPC do Secrets Manager não permite a entrada de CIDRs da VPC nos grupos de segurança atribuídos.

Para obter mais informações, consulte Controle o tráfego para recursos usando grupos de segurança no Guia do usuário da Amazon VPC.

A política de endpoint da VPC do Secrets Manager não permite que o Lambda use o endpoint da VPC.

Para ter mais informações, consulte Endpoint da VPC.

O segredo usa a alternância de usuários alternados, o segredo do superusuário é gerenciado pelo Amazon RDS e a função do Lambda não pode acessar a API do RDS.

Para alternar a rotação de usuários em que o segredo do superusuário é gerenciado por outro AWS serviço, a função de rotação do Lambda deve ser capaz de chamar o endpoint do serviço para obter as informações de conexão do banco de dados. Recomendamos que você configure um endpoint da VPC para o serviço de banco de dados. Para obter mais informações, consulte:

- API do Amazon RDS e endpoints da VPC de interface no Guia do usuário do Amazon RDS.
- Trabalhar com endpoints da VPC no Guia de gerenciamento do Amazon Redshift.

Erro: “O acesso ao KMS não é permitido”

Se você vir `ClientError: An error occurred (AccessDeniedException) when calling the GetSecretValue operation: Access to KMS is not allowed`, a função de alternância não tem permissão para descriptografar o segredo usando a chave KMS usada para criptografar o segredo. Pode haver uma condição na política de permissões que limita o contexto

de criptografia a um segredo específico. Para obter informações sobre as permissões necessárias, consulte the section called “Declaração de política para chaves gerenciadas pelo cliente”.

Erro: “Key is missing from secret JSON” (A chave do segredo JSON está ausente)

Uma função de alternância do Lambda requer que o valor do segredo esteja em uma estrutura JSON específica. Se você vir esse erro, o JSON pode estar sem uma chave que a função de alternância tentou acessar. Para obter informações sobre a estrutura JSON para cada tipo de segredo, consulte the section called “Estrutura JSON de um segredo”.

Erro: “setSecret: Unable to log into database” (setSecret: não é possível fazer login no banco de dados)

A seguir, estão os problemas que podem causar esse erro:

A função de alternância não pode acessar o banco de dados.

- Se a duração da tarefa for longa, por exemplo, mais de 5 milissegundos, a função de alternância do Lambda pode não conseguir acessar o banco de dados utilizando a rede.

  Se seu banco de dados ou serviço estiver sendo executado em uma instância do Amazon EC2 em uma VPC, é recomendável que você configure a função do Lambda para ser executada na mesma VPC. Assim, a função de alternância pode se comunicar diretamente com seu serviço. Para obter mais informações, consulte Configuring VPC access (Configurar o acesso à VPC).

  Para permitir que a função do Lambda acesse o banco de dados ou serviço, certifique-se de que os grupos de segurança anexados à sua função de alternância do Lambda permitam conexões de saída com o banco de dados ou serviço. Você também deve garantir que os grupos de segurança anexados ao seu banco de dados ou serviço permitam conexões de entrada a partir da função de alternância do Lambda.

As credenciais no segredo estão incorretas.

- Se a duração da tarefa for curta, a função de alternância do Lambda pode não conseguir autenticar com as credenciais no segredo. Verifique as credenciais fazendo login manualmente com as informações nas `AWSPREVIOUS` versões `AWSCURRENT` e do segredo usando o AWS CLI comando `get-secret-value`.

O banco de dados usa **`scram-sha-256`** para criptografar senhas.

- Se seu banco de dados for Aurora PostgreSQL versão 13 ou posterior e usar `scram-sha-256` para criptografar senhas, mas a função de alternância usar `libpq` versão 9 ou anterior que não

seja compatível com `scram-sha-256`, a função de alternância não poderá estabelecer conexão com o banco de dados.

Para determinar quais usuários do banco de dados usam criptografia **`scram-sha-256`**

- Consulte Checking for users with non-SCRAM passwords (Verificação de usuários com senhas não SCRAM) na postem de blog SCRAM Authentication in RDS for PostgreSQL 13 (Autenticação SCRAM no RDS para PostgreSQL 13).

Para determinar a versão que sua função de alternância **`libpq`** usa

1. Em um computador baseado em Linux, no console do Lambda, acesse sua função de alternância e baixe o pacote de implantação. Descompacte o arquivo zip em um diretório de trabalho.
2. Usando uma linha de comando, no diretório de trabalho, execute:

   ```
   readelf -a libpq.so.5 | grep RUNPATH
   ```
3. Se você visualizar a string *`PostgreSQL-9.4.x`* ou qualquer versão principal inferior à versão 10, a função de alternância não será compatível com `scram-sha-256`.
   - Saída para uma função de alternância incompatível com `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]
     ```
   - Saída para uma função de alternância compatível com `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]
     ```

> **Note**
>
> Se tiver configurado a alternância automática de segredos antes de 30 de dezembro de 2021, sua função de alternância incluiu uma versão mais antiga de `libpq` que não é compatível com `scram-sha-256`. Para oferecer suporte a `scram-sha-256`, você precisa recriar sua função de alternância.

O banco de dados requer acesso SSL/TLS.

Se seu banco de dados requer uma conexão SSL/TLS, mas a função de alternância usa uma conexão não criptografada, a função de alternância não pode se conectar ao banco de dados. As funções de rodízio do Amazon RDS (exceto Oracle e Db2) e o Amazon DocumentDB usam automaticamente o Secure Socket Layer (SSL) ou o Transport Layer Security (TLS) para se conectar ao seu banco de dados, se ele estiver disponível. Caso contrário, utilizarão uma conexão não criptografada.

> **Note**
>
> Se você configurou a alternância automática de segredos antes de 20 de dezembro de 2021, a sua função de alternância pode ser baseada em um modelo mais antigo que é incompatível com SSL/TLS. Para oferecer suporte a conexões que usam SSL/TLS, você precisa recriar a sua função de alternância.

Para determinar quando a sua função de alternância foi criada

1. No console do Secrets Manager https://console.aws.amazon.com/secretsmanager/, abra o seu segredo. Na seção Rotation configuration (Configuração de alternância), em Lambda rotation function (Função de alternância do Lambda), você verá o Lambda function ARN (ARN da função Lambda), por exemplo, `arn:aws:lambda:`*`aws-region`*`:`*`123456789012`*`:function:`*`SecretsManagerMyRotationFunction`*. Copie o nome da função do final do ARN. Neste exemplo, é *`SecretsManagerMyRotationFunction`* .
2. No AWS Lambda console https://console.aws.amazon.com/lambda/, em Funções, cole o nome da função Lambda na caixa de pesquisa, escolha Enter e escolha a função Lambda.
3. Na página de detalhes da função, na guia Configuration (Configuração), em Tags (Etiquetas), copie o valor ao lado da chave aws:cloudformation:stack-name.

4. No AWS CloudFormation console https://console.aws.amazon.com/cloudformation, em Pilhas, cole o valor da chave na caixa de pesquisa e escolha Enter.
5. A lista de filtros de pilhas para que somente a pilha que criou a função de alternância do Lambda apareça. Na coluna Created date (Data da criação), visualize a data em que a pilha foi criada. Esta é a data em que a função de alternância do Lambda foi criada.

Erro: “Não é possível importar o módulo ‘lambda_function’"

Você pode receber esse erro se estiver executando uma função Lambda anterior que foi atualizada automaticamente do Python 3.7 para uma versão mais recente do Python. Para solucionar o erro, você pode alterar a versão da função do Lambda de volta para Python 3.7 e, em seguida, para the section called “Atualize uma função de alternância existente do Python 3.7 para 3.9”. Para obter mais informações, consulte Why did my Secrets Manager Lambda function rotation fail with a “pg module not found“ error? em AWS re:Post.

Atualize uma função de alternância existente do Python 3.7 para 3.9

Algumas funções de alternância criadas antes de novembro de 2022 usavam o Python 3.7. O AWS SDK para Python deixou de oferecer suporte ao Python 3.7 em dezembro de 2023. Para obter mais informações, consulte Atualizações da política de suporte do Python para AWS SDKs e ferramentas. Para mudar para uma nova função de alternância que usa o Python 3.9, é possível adicionar uma propriedade de runtime a uma função de alternância existente ou recriá-la.

Para descobrir quais funções de alternância do Lambda usam o Python 3.7

1. Faça login no AWS Management Console e abra o AWS Lambda console em https://console.aws.amazon.com/lambda/.
2. Na lista de Funções, filtre por **`SecretsManager`**.
3. Na lista filtrada de funções, em Runtime, busque o Python 3.7.

Para atualizar para o Python 3.9:

- Opção 1: Recriar a função de rotação usando AWS CloudFormation
- Opção 2: atualizar o tempo de execução da função de rotação existente usando AWS CloudFormation
- Opção 3: Para AWS CDK usuários, atualize a biblioteca CDK

Opção 1: Recriar a função de rotação usando AWS CloudFormation

Quando você usa o console do Secrets Manager para ativar a rotação, o Secrets Manager usa AWS CloudFormation para criar os recursos necessários, incluindo a função de rotação do Lambda. Se você usou o console para ativar a rotação ou criou a função de rotação usando uma AWS CloudFormation pilha, poderá usar a mesma AWS CloudFormation pilha para recriar a função de rotação com um novo nome. A nova função usa a versão mais recente do Python.

Para encontrar a AWS CloudFormation pilha que criou a função de rotação

- Na página de detalhes da função do Lambda, na guia Configuração, escolha Tags. Visualize o ARN próximo à aws:cloudformation:stack-id.

  O nome da pilha está incorporado no ARN, conforme exibido no exemplo a seguir.

  - ARN: `arn:aws:cloudformation:us-west-2:408736277230:stack/`**`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**`-3CUDHZMDMBO8/79fc9050-2eef-11ed-`
  - Nome da pilha: **`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**

Para recriar uma função de alternância (AWS CloudFormation)

1. Em AWS CloudFormation, pesquise a pilha pelo nome e escolha Atualizar.

   Se aparecer uma caixa de diálogo recomendando a atualização da pilha raiz, escolha Ir para a pilha raiz e, em seguida, escolha Atualizar.
2. Na página Atualizar pilha, escolha Editar modelo no Designer e, em seguida, escolha Visualizar no Designer.
3. No Designer, no código do modelo, em `SecretRotationScheduleHostedRotationLambda`, substitua o valor por `"functionName": "SecretsManagerTestRotationRDS"` com um novo nome de função. Por exemplo, em JSON, **`"functionName": "SecretsManagerTestRotationRDSupdated"`**.
4. Continue com o fluxo de trabalho da AWS CloudFormation pilha e escolha Enviar.

Opção 2: atualizar o tempo de execução da função de rotação existente usando AWS CloudFormation

Quando você usa o console do Secrets Manager para ativar a rotação, o Secrets Manager usa AWS CloudFormation para criar os recursos necessários, incluindo a função de rotação do Lambda. Se você usou o console para ativar a rotação ou criou a função de rotação usando uma AWS CloudFormation pilha, poderá usar a mesma AWS CloudFormation pilha para atualizar o tempo de execução da função de rotação.

Para encontrar a AWS CloudFormation pilha que criou a função de rotação

- Na página de detalhes da função do Lambda, na guia Configuração, escolha Tags. Visualize o ARN próximo à aws:cloudformation:stack-id.

  O nome da pilha está incorporado no ARN, conforme exibido no exemplo a seguir.

  - ARN: `arn:aws:cloudformation:us-west-2:408736277230:stack/`**`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**`-3CUDHZMDMBO8/79fc9050-2eef-11ed-`
  - Nome da pilha: **`SecretsManagerRDSMySQLRotationSingleUser5c2-SecretRotationScheduleHostedRotationLambda`**

Para atualizar o runtime de uma função de alternância (AWS CloudFormation)

1. Em AWS CloudFormation, pesquise a pilha pelo nome e escolha Atualizar.

   Se aparecer uma caixa de diálogo recomendando a atualização da pilha raiz, escolha Ir para a pilha raiz e, em seguida, escolha Atualizar.
2. Na página Atualizar pilha, escolha Editar modelo no Designer e, em seguida, escolha Visualizar no Designer.
3. No designer, no modelo JSON, para`SecretRotationScheduleHostedRotationLambda`, abaixo `PropertiesParameters`, adicione **`"runtime": "python3.9"`**
4. Continue com o fluxo de trabalho da AWS CloudFormation pilha e escolha Enviar.

Opção 3: Para AWS CDK usuários, atualize a biblioteca CDK

Se você usou a versão AWS CDK anterior à v2.94.0 para configurar a rotação do seu segredo, você pode atualizar a função Lambda fazendo o upgrade para a v2.94.0 ou posterior. Para obter mais informações, consulte o Guia do desenvolvedor do AWS Cloud Development Kit (AWS CDK) v2.

Gire um AWS Secrets Manager segredo imediatamente

Você só pode alternar um segredo que tenha a alternância configurada. Para determinar se um segredo foi configurado para alternância, no console, visualize o segredo e role para baixo até a seção Rotation configuration (Configuração de alternância). Se o Rotation status (Estado de alternância) for Enabled (Habilitado), o segredo será configurado para alternância. Se não, consulte Alternar segredos .

Para alternar um segredo imediatamente (console)

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Escolha seu segredo.
3. Na página de detalhes do segredo, em Rotation configuration (Configuração de alternância), escolha Rotate secret immediately (Alternar o segredo imediatamente).
4. Na caixa de diálogo Rotate secret (Alternar segredo), escolha Rotate (Alternar).

AWS CLI

Example Alternar um segredo imediatamente

O exemplo de `rotate-secret` a seguir inicia uma alternância imediata. O segredo já deve ter a alternância configurada.

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

Cronogramas de rotação

Quando você ativa a alternância automática, é possível usar uma expressão cron() ou rate() para definir o cronograma para a alternância do seu segredo. Com uma expressão rate, você pode criar uma programação de alternância que se repetirá com base em um determinado intervalo de horas ou dias. Com uma expressão cron, você pode criar programações de alternância mais detalhadas do

que um intervalo de alternância. Os horários de alternância do Secrets Manager usam o fuso horário UTC. Você pode alternar um segredo com intervalos a partir de quatro horas. O Secrets Manager alterna seu segredo a qualquer momento durante a janela de alternância.

Para ativar a alternância, consulte:

- the section called “Alternância gerenciada”
- the section called “Alternância automática para segredos de banco de dados (console)”
- the section called “Rotação automática para segredos que não são do banco de dados (console)”

Expressões rate

As expressões rate do Secrets Manager têm o seguinte formato, com *`Value`* (Valor) indicando um número inteiro positivo e *`Unit`* (Unidade) podendo ser `hour`, `hours`, `day` ou `days`:

```
rate(Value Unit)
```

Você pode alternar um segredo com intervalos a partir de quatro horas. Exemplos:

- `rate(4 hours)` significa que o segredo é alternado a cada 4 horas.
- `rate(1 day)` significa que o segredo é alternado a cada dia.
- `rate(10 days)` significa que o segredo é alternado a cada 10 dias.

Para um rate (ritmo) em horas, a janela padrão de alternância começa à meia-noite e se encerra após uma hora. É possível definir a Window duration (Duração de janela) para modificar a janela de alternância. A janela de alternância não pode se estender até a próxima janela de alternância. Uma maneira de verificar isso é confirmar se a janela de alternância é menor ou igual ao número de horas entre as alternâncias.

Para um rate (ritmo) em dias, a janela de alternância padrão começa à meia-noite e se encerra ao final do dia. É possível definir a Window duration (Duração de janela) para modificar a janela de alternância. A janela de alternância não pode se estender até o próximo dia UTC. Uma forma de verificar isso é confirmar se o horário inicial somado à duração da janela é menor ou igual a 24 horas.

Expressão cron

As expressões cron têm o formato a seguir:

```
cron(Minutes Hours Day-of-month Month Day-of-week Year)
```

Uma expressão cron que inclui incrementos de redefinições de horas a cada dia. Por exemplo, `cron(0 4/12 * * ? *)` significa 4h, 16h e, no dia seguinte, 4h, 16h. Os horários de alternância do Secrets Manager usam o fuso horário UTC.

Para uma programação em horas, a janela padrão de alternância se encerra após uma hora. É possível definir a Window duration (Duração de janela) para modificar a janela de alternância. A janela de alternância não pode adentrar na próxima janela de alternância. Você pode alternar um segredo com intervalos a partir de quatro horas.

Exemplo de programação	Expressão
A cada 8 horas a partir da meia-noite.	`cron(0 /8 * * ? *)`
A cada 8 horas a partir das 8h.	`cron(0 8/8 * * ? *)`
A cada 10 horas a partir das 2h. As janelas de alternância começarão às 2h, 12h e 22h, e no dia seguinte, às 2h, 12h e 22h.	`cron(0 2/10 * * ? *)`
Todos os dias às 10h.	`cron(0 10 * * ? *)`
Todos os sábados às 18h.	`cron(0 18 ? * SAT *)`
O primeiro dia de cada mês, às 8h.	`cron(0 8 1 * ? *)`
A cada três meses, no primeiro domingo, à 1 hora da manhã.	`cron(0 1 ? 1/3 SUN#1 *)`
O último dia de cada mês, às 17h.	`cron(0 17 L * ? *)`
De segunda-feira a sexta-feira, às 8h.	`cron(0 8 ? * MON-FRI *)`
Primeiro e 15.º dia de cada mês às 16h.	`cron(0 16 1,15 * ? *)`
Primeiro domingo de cada mês à 0h.	`cron(0 0 ? * SUN#1 *)`

Requisitos de expressão cron no Secrets Manager

O Secrets Manager tem algumas restrições quanto ao que pode ser usado em expressões cron. Uma expressão cron para o Secrets Manager deve ter 0 no campo de minutos, pois as janelas de alternância do Secrets Manager começam na hora indicada. É necessário que um * esteja no campo de ano, porque o Secrets Manager não oferece suporte a cronogramas de alternância com mais de um ano de intervalo. A tabela a seguir mostra as opções que você pode utilizar.

Campos	Valores	Curingas
Minutos	Deve ser 0	Nenhum
Horas	0–23	Use / (barra) para especific ar incrementos. Por exemplo, `2/10` significa a cada 10 horas a partir das 2h. Você pode alternar um segredo com intervalos a partir de quatro horas.
D ay-of-month	1–31	Use , (vírgula) para incluir valores adicionais. Por exemplo, `1,15` significa o 1.º e o 15.º dia do mês. Use - (traço) para especificar um intervalo. Por exemplo, `1-15` significa do dia 1 ao dia 15 do mês. Use * (asterisco) para incluir todos os valores no campo. Por exemplo, `*` significa todos os dias do mês. O curinga ? (interrogação) especifica um ou outro. Não é possível especificar os

Campos	Valores	Curingas
		campos `Day-of-month` e `Day-of-week` na mesma expressão cron. Se você especificar um valor em um dos campos, deverá usar um ? (ponto de interrogação) no outro. Use / (barra) para especificar incrementos. Por exemplo, `1/2` significa a cada 2 dias a partir do dia 1, ou seja, dia 1, 3, 5 e assim por diante. Use L para especificar o último dia do mês. Use ***DAY***L para especificar o último dia nomeado do mês. Por exemplo, `SUNL` significa o último domingo do mês.

Campos	Valores	Curingas
Mês	1-12 ou JAN-DEZ	Use , (vírgula) para incluir valores adicionais. Por exemplo, `JAN,APR,JUL,OCT` significa janeiro, abril, julho e outubro. Use - (traço) para especificar um intervalo. Por exemplo, `1-3` significa do mês 1 ao mês 3 do ano. Use * (asterisco) para incluir todos os valores no campo. Por exemplo, * significa todos os meses. Use / (barra) para especificar incrementos. Por exemplo, `1/3` significa a cada 3 meses, começando no mês 1, ou seja, mês 1, 4, 7 e 10.

Campos	Valores	Curingas
D ay-of-week	1-7 ou DOM-SÁB	Use # para especificar o dia da semana em um mês. Por exemplo, `TUE#3` significa a terceira terça-feira do mês. Use , (vírgula) para incluir valores adicionais. Por exemplo, `1,4` significa o 1.º e o 4.º dia da semana. Use - (traço) para especificar um intervalo. Por exemplo, `1-4` significa do dia 1 ao dia 4 da semana. Use * (asterisco) para incluir todos os valores no campo. Por exemplo, `*` significa todos os dias da semana. O curinga ? (interrogação) especifica um ou outro. Não é possível especificar os campos `Day-of-month` e `Day-of-week` na mesma expressão cron. Se você especificar um valor em um dos campos, deverá usar um ? (ponto de interrogação) no outro. Use / (barra) para especific ar incrementos. Por exemplo, `1/2` significa cada 2.º dia da semana, começando no

Campos	Valores	Curingas
		primeiro dia, ou seja, dias 1, 3, 5 e 7. Use L para especificar o último dia da semana.
Ano	Deve ser *	Nenhum

Encontre segredos que não são alterados

Você pode usar AWS Config para avaliar seus segredos para ver se eles estão alternando de acordo com seus padrões. Você define seus requisitos internos de segurança e conformidade para segredos usando AWS Config regras. Em seguida, AWS Config pode identificar segredos que não estão de acordo com suas regras. Você também pode rastrear alterações em metadados de segredos, configuração de alternância, na chave KMS usada para criptografia de segredos, função de alternância do Lambda e etiquetas associadas a um segredo.

Se você tiver segredos em vários Contas da AWS e Regiões da AWS em sua organização, poderá agregar esses dados de configuração e conformidade. Para obter mais informações, consulte Agregação de dados multirregionais de várias contas.

Para avaliar se os segredos estão girando

1. Siga as instruções em Como avaliar seus recursos com AWS Config regras e escolha uma das seguintes regras:

   - `secretsmanager-rotation-enabled-check`: verifica se a alternância está configurada para segredos armazenados no Secrets Manager.
   - `secretsmanager-scheduled-rotation-success-check`: verifica se a última alternância bem-sucedida está dentro da frequência de alternância configurada. A frequência mínima para a verificação é diária.
   - `secretsmanager-secret-periodic-rotation`: verifica se os segredos foram alternados dentro do número de dias especificado.
2. Opcionalmente, configure AWS Config para notificá-lo quando os segredos não estiverem em conformidade. Para obter mais informações, consulte Notificações AWS Config enviadas para um tópico do Amazon SNS.

Cancelar a rotação automática no Secrets Manager

Se você configurou a rotação automática para um segredo e deseja parar de girá-lo, poderá cancelar a rotação.

Para cancelar a rotação automática

1. Abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Escolha seu segredo.
3. Na página de detalhes secretos, em Configuração de rotação, escolha Editar rotação.
4. Na caixa de diálogo Editar configuração de rotação, desative a rotação automática e escolha Salvar.

   O Secrets Manager retém as informações de configuração de rotação para que você possa usá-las no futuro, caso decida reativar a rotação.

AWS Secrets Manager segredos gerenciados por outros AWS serviços

Muitos AWS serviços armazenam e usam segredos em AWS Secrets Manager. Em alguns casos, esses segredos são segredos gerenciados, o que significa que o serviço que os criou ajuda a gerenciá-los. Por exemplo, alguns segredos gerenciados incluem alternância gerenciada para que você não precise configurar a alternância por conta própria. O serviço de gerenciamento também pode impedir você de atualizar segredos ou excluí-los sem um período de recuperação, o que ajuda a evitar interrupções, uma vez que o serviço de gerenciamento depende do segredo.

Os segredos gerenciados usam uma convenção de nomenclatura que inclui o ID do serviço de gerenciamento para ajudar a identificá-los.

```
Secret name: ServiceID!MySecret
Secret ARN : arn:aws:us-east-1:ServiceID!MySecret-a1b2c3
```

IDs de serviços que gerenciam segredos

- `appflow` – the section called “Amazon AppFlow”
- `databrew` – the section called “AWS Glue DataBrew”
- `datasync` – the section called “AWS DataSync”
- `directconnect` – the section called “AWS Direct Connect”
- `ecs-sc` – the section called “Amazon Elastic Container Service”
- `events` – the section called “Amazon EventBridge”
- `marketplace-deployment` – the section called “AWS Marketplace”
- `opsworks-cm` – the section called “AWS OpsWorks for Chef Automate”
- `rds` – the section called “Amazon RDS”
- `redshift` – the section called “Amazon Redshift”
- `sqlworkbench` – the section called “Editor de Consultas do Amazon Redshift v2”

Para encontrar segredos gerenciados por outros AWS serviços, consulte Localizar segredos gerenciados.

Para obter uma lista completa dos serviços que usam segredos, consulteServiços que usam segredos.

AWS serviços que usam AWS Secrets Manager segredos

Obtenha informações sobre como cada um dos itens a seguir Serviços da AWS se integra ao Secrets Manager.

- Como AWS App Runner usa AWS Secrets Manager
- Como o AWS App2Container usa AWS Secrets Manager
- Como AWS AppConfig usa AWS Secrets Manager
- Como a Amazon AppFlow usa AWS Secrets Manager
- Como AWS AppSync usa AWS Secrets Manager
- Como o Amazon Athena usa o AWS Secrets Manager
- Como o Amazon Aurora usa AWS Secrets Manager
- Como AWS CodeBuild usa AWS Secrets Manager
- Como o Amazon Data Firehose usa AWS Secrets Manager
- Como AWS DataSync usa AWS Secrets Manager
- Como a Amazon DataZone usa AWS Secrets Manager
- Como AWS Direct Connect usa AWS Secrets Manager
- Como AWS Directory Service usa AWS Secrets Manager
- Como o Amazon DocumentDB (com compatibilidade com o MongoDB) usa o AWS Secrets Manager
- Como AWS Elastic Beanstalk usa AWS Secrets Manager
- Como o Amazon Elastic Container Registry usa AWS Secrets Manager
- Amazon Elastic Container Service
- Como a Amazon ElastiCache usa AWS Secrets Manager
- Como AWS Elemental Live usa AWS Secrets Manager
- Como AWS Elemental MediaConnect usa AWS Secrets Manager
- Como AWS Elemental MediaConvert usa AWS Secrets Manager
- Como AWS Elemental MediaLive usa AWS Secrets Manager
- Como AWS Elemental MediaPackage usa AWS Secrets Manager
- Como AWS Elemental MediaTailor usa AWS Secrets Manager

- Como o Amazon EMR usa o Secrets Manager
- Como a Amazon EventBridge usa AWS Secrets Manager
- Como o Amazon FSx usa segredos AWS Secrets Manager
- Como AWS Glue DataBrew usa AWS Secrets Manager
- Como o AWS Glue Studio usa AWS Secrets Manager
- Como AWS IoT SiteWise usa AWS Secrets Manager
- Como a Amazon Kendra usa AWS Secrets Manager
- Como o Amazon Kinesis Video Streams usa AWS Secrets Manager
- Como AWS Launch Wizard usa AWS Secrets Manager
- Como o Amazon Lookout for Metrics usa o AWS Secrets Manager
- Como o Amazon Managed Grafana usa AWS Secrets Manager
- Como AWS Managed Services usa AWS Secrets Manager
- Como o Amazon Managed Streaming for Apache Kafka usa o AWS Secrets Manager
- Como o Amazon Managed Workflows for Apache Airflow usa AWS Secrets Manager
- AWS Marketplace
- Como AWS Migration Hub usa AWS Secrets Manager
- Como AWS Panorama usa o Secrets Manager
- Como AWS ParallelCluster usa AWS Secrets Manager
- Como o Amazon Q usa o Secrets Manager
- Como AWS OpsWorks for Chef Automate usa AWS Secrets Manager
- Como a Amazon QuickSight usa AWS Secrets Manager
- Amazon RDS
- Como o Amazon Redshift usa AWS Secrets Manager
- Editor de Consultas do Amazon Redshift v2
- Como a Amazon SageMaker usa AWS Secrets Manager
- Como AWS Schema Conversion Tool usa AWS Secrets Manager
- Como AWS Toolkit for JetBrains usa AWS Secrets Manager
- Como AWS Transfer Family usa AWS Secrets Manager segredos

- Como AWS Wickr usa segredos AWS Secrets Manager

Como AWS App Runner usa AWS Secrets Manager

AWS App Runner é um AWS serviço que fornece uma maneira rápida, simples e econômica de implantar a partir do código-fonte ou de uma imagem de contêiner diretamente em um aplicativo web escalável e seguro na AWS nuvem. Você não precisa aprender novas tecnologias, decidir qual serviço de computação usar ou saber como provisionar e configurar AWS recursos.

Com o App Runner, você pode fazer referência a segredos e configurações como variáveis de ambiente em seu serviço ao criar um serviço ou atualizar a configuração do serviço. Para obter mais informações, consulte Referencing environment variables (Fazer referência a variáveis de ambiente) e Managing environment variables (Gerenciar variáveis de ambiente) no Guia do desenvolvedor do AWS App Runner .

Como o AWS App2Container usa AWS Secrets Manager

AWS App2Container é uma ferramenta de linha de comando para ajudá-lo a elevar e deslocar aplicativos executados em seus datacenters locais ou em máquinas virtuais, para que sejam executados em contêineres gerenciados pelo Amazon ECS, Amazon EKS ou. AWS App Runner

O App2Container usa o Secrets Manager para gerenciar as credenciais para conectar sua máquina de trabalho a servidores de aplicações para executar comandos remotos. Para obter mais informações, consulte Gerenciar segredos do AWS App2Container no Guia do usuário do AWS App2Container.

Como AWS AppConfig usa AWS Secrets Manager

AWS AppConfig é um recurso AWS Systems Manager que você pode usar para criar, gerenciar e implantar rapidamente configurações de aplicativos. Uma configuração pode conter dados de credenciais ou outras informações confidenciais armazenadas no Secrets Manager. Ao criar um perfil de configuração de formato livre, você pode escolher o Secrets Manager como fonte de seus dados de configuração. Para obter mais informações, consulte Criar um perfil de configuração de formato livre no Guia do usuário do AWS AppConfig . Para obter informações sobre como AWS AppConfig lidar com segredos que têm a rotação automática ativada, consulte a rotação de chaves do Secrets Manager no Guia AWS AppConfig do Usuário.

Como a Amazon AppFlow usa AWS Secrets Manager

AppFlow A Amazon é um serviço de integração totalmente gerenciado que permite que você troque dados com segurança entre aplicativos de software como serviço (SaaS), como o Salesforce, e, como o Amazon Simple Storage Service (Amazon S3) e Serviços da AWS o Amazon Redshift.

Na Amazon AppFlow, ao configurar um aplicativo SaaS como origem ou destino, você cria uma conexão. Isso inclui informações necessárias para se conectar às aplicações SaaS, como tokens de autenticação, nomes de usuário e senhas. A Amazon AppFlow armazena seus dados de conexão em um segredo gerenciado pelo Secrets Manager com o prefixo`appflow`. O custo de armazenar o segredo está incluído na cobrança da Amazon AppFlow. Para obter mais informações, consulte Proteção de dados na Amazon AppFlow no Guia AppFlow do usuário da Amazon.

Como AWS AppSync usa AWS Secrets Manager

AWS AppSync fornece uma interface GraphQL robusta e escalável para desenvolvedores de aplicativos combinarem dados de várias fontes, incluindo Amazon DynamoDB e AWS Lambda APIs HTTP.

AWS AppSync usa o comando CLI `rds execute-statement`para se conectar ao Amazon RDS usando as credenciais em um segredo. Para obter mais informações, consulte Tutorial: Aurora Serverless (Tutorial: Aurora Serverless) no Guia do desenvolvedor do AWS AppSync .

Como o Amazon Athena usa o AWS Secrets Manager

O Amazon Athena é um serviço de consultas interativas que facilita a análise de dados diretamente no Amazon Simple Storage Service (Amazon S3) usando SQL padrão.

Os conectores da fonte de dados do Amazon Athena podem usar o atributo de consulta federada do Athena com segredos do Secrets Manager para consultar dados. Para obter mais informações, consulte Usar a consulta federada do Amazon Athena no Guia do usuário do Amazon Athena.

Como o Amazon Aurora usa AWS Secrets Manager

O Amazon Aurora é um mecanismo de banco de dados relacional gerenciado compatível com o MySQL e o PostgreSQL.

Para gerenciar as credenciais do usuário principal do Aurora, o Aurora pode criar um segredo gerenciado para você. Haverá uma cobrança por esse segredo. O Aurora também gerencia a rotação dessas credenciais. Para obter mais informações, consulte Gerenciamento de senhas com Amazon Aurora e o AWS Secrets Manager no Guia do usuário do Amazon Aurora.

Para outras credenciais da Aurora, consulte. the section called “Criar um segredo de banco de dados”

Ao chamar a API de dados do Amazon RDS, você pode passar credenciais para o banco de dados usando um segredo no Secrets Manager. Para obter mais informações, consulte Usar a tecnologia sem servidor da API Data for Aurora no Guia do usuário do Amazon Aurora.

Ao usar o editor de consultas do Amazon RDS para se conectar a um banco de dados, você pode armazenar credenciais para o banco de dados no Secrets Manager. Para obter mais informações, consulte Usar o editor de consultas no Guia do usuário do Amazon RDS.

Como AWS CodeBuild usa AWS Secrets Manager

AWS CodeBuild é um serviço de criação totalmente gerenciado na nuvem. CodeBuild compila seu código-fonte, executa testes de unidade e produz artefatos prontos para serem implantados.

É possível armazenar as credenciais de registro privadas usando o Secrets Manager. Para obter mais informações, consulte Registro privado com AWS Secrets Manager amostra CodeBuild no Guia do AWS CodeBuild usuário.

Como o Amazon Data Firehose usa AWS Secrets Manager

Você pode usar o Amazon Data Firehose para fornecer dados de streaming em tempo real para vários destinos de streaming. Quando o destino exige uma chave ou credenciais, o Firehose recupera um segredo do Secrets Manager em tempo de execução para se conectar ao destino. Para obter mais informações, consulte Autenticar com o Amazon Data Firehose AWS Secrets Manager no Guia do desenvolvedor do Amazon Data Firehose.

Como AWS DataSync usa AWS Secrets Manager

AWS DataSync é um serviço de transferência de dados on-line que simplifica, automatiza e acelera a movimentação de dados entre sistemas e serviços de armazenamento. DataSync O Discovery ajuda você a acelerar sua migração para AWS o.

Para coletar informações sobre um sistema de armazenamento local, o DataSync Discovery usa as credenciais da interface de gerenciamento do sistema de armazenamento. DataSync armazena essas credenciais em um segredo gerenciado do Secrets Manager com o prefixo`datasync`. Haverá uma cobrança por esse segredo. Para obter mais informações, consulte Adicionar seu sistema de armazenamento local ao DataSync Discovery no Guia do AWS DataSync usuário.

Como a Amazon DataZone usa AWS Secrets Manager

DataZone A Amazon é um serviço de gerenciamento de dados que permite catalogar, descobrir, controlar, compartilhar e analisar seus dados. Você pode usar ativos de dados de tabelas e visualizações de um cluster do Amazon Redshift que é rastreado usando um trabalho. Crawler do AWS Glue Para se conectar ao Amazon Redshift, você fornece as DataZone credenciais da Amazon em um segredo do Secrets Manager. Para obter mais informações, consulte Criar uma fonte de dados para um banco de dados do Amazon Redshift usando uma nova AWS Glue conexão no Guia DataZone do usuário da Amazon.

Como AWS Direct Connect usa AWS Secrets Manager

AWS Direct Connect conecta sua rede interna a um AWS Direct Connect local por meio de um cabo de fibra óptica Ethernet padrão. Com essa conexão, você pode criar interfaces virtuais diretamente para o público Serviços da AWS.

AWS Direct Connect armazena um nome de chave de associação de conectividade e um par de chaves de associação de conectividade (par CKN/CAK) em um segredo gerenciado com o prefixo. `directconnect` O custo do segredo está incluído na cobrança do AWS Direct Connect. Para atualizar o segredo, você deve usar AWS Direct Connect em vez do Secrets Manager. Para obter mais informações, consulte Associate a MACsec CKN/CAK with a LAG (Associar uma chave MacSec CKN/CAK a um LAG) no Guia do usuário do AWS Direct Connect .

Como AWS Directory Service usa AWS Secrets Manager

AWS Directory Service fornece várias maneiras de usar o Microsoft Active Directory (AD) com outros AWS serviços. Você pode unir uma instância do Amazon EC2 ao seu diretório usando segredos para credenciais: Para obter mais informações, no Guia do usuário do AWS Direct Connect , consulte:

- Associe perfeitamente uma instância Linux EC2 ao seu diretório gerenciado AWS do Microsoft AD

- Ingressar perfeitamente uma instância do EC2 do Linux ao seu diretório do AD Connector
- Ingressar perfeitamente uma instância do EC2 do Linux ao seu diretório do Simple AD

Como o Amazon DocumentDB (com compatibilidade com o MongoDB) usa o AWS Secrets Manager

No Amazon DocumentDB, os usuários se autenticam em um cluster em conjunto com uma senha. Com o AWS Secrets Manager, é possível substituir credenciais codificadas, incluindo senhas, por uma chamada de API ao Secrets Manager para recuperar o segredo por programação. Para obter mais informações, consulte the section called “Criar um segredo de banco de dados” e Managing Amazon DocumentDB Users (Gestão de usuários do Amazon DocumentDB) no Guia do desenvolvedor do Amazon DocumentDB.

Como AWS Elastic Beanstalk usa AWS Secrets Manager

Com AWS Elastic Beanstalk, você pode implantar e gerenciar rapidamente aplicativos na AWS nuvem sem precisar aprender sobre a infraestrutura que executa esses aplicativos. O Elastic Beanstalk pode iniciar ambientes do Docker compilando uma imagem descrita em um Dockerfile ou usando uma imagem do Docker remota. Para se autenticar com o registro on-line que hospeda o repositório privado, o Elastic Beanstalk usa um segredo do Secrets Manager. Para obter mais informações, consulte Configuração do Docker no AWS Elastic Beanstalk Guia do desenvolvedor.

Como o Amazon Elastic Container Registry usa AWS Secrets Manager

O Amazon Elastic Container Registry (Amazon ECR) é AWS um serviço gerenciado de registro de imagens de contêineres que é seguro, escalável e confiável. Você pode usar a CLI do Docker, ou seu cliente preferido, para enviar e extrair imagens dos seus repositórios. Para cada registro upstream que contém imagens que você deseja armazenar em cache em seu registro privado do Amazon ECR, é necessário criar uma regra de cache de pull-through. Para registros de upstream que exigem autenticação, você deve armazenar as credenciais em um segredo do Secrets Manager. Você pode criar o segredo do Secrets Manager nos consoles Amazon ECR ou Secrets Manager. Para obter mais informações, consulte Criação de uma regra de cache pull through no Guia do usuário do Amazon ECR.

Amazon Elastic Container Service

O Amazon Elastic Container Service (Amazon ECS) é um serviço totalmente gerenciado de orquestração de contêineres ajuda a implantar, gerenciar e dimensionar facilmente aplicações conteinerizadas. Você pode injetar dados confidenciais em seus contêineres fazendo referência aos segredos do Secrets Manager. Para obter mais informações, consulte as seguintes páginas no Guia do desenvolvedor do Amazon Elastic Container Service:

- Tutorial: Especificar dados sigilosos usando segredos do Secrets Manager
- Recuperar segredos programaticamente por meio da sua aplicação
- Recuperar segredos por meio de variáveis de ambiente
- Recuperar segredos para a configuração de registro em log

O Amazon ECS oferece suporte a volumes FSx for Windows File Server para contêineres. O Amazon ECS usa as credenciais armazenadas em um segredo do Secrets Manager para ingressar no domínio do Active Directory e anexar o sistema de arquivos FSx for Windows File Server. Para obter mais informações, consulte Tutorial: Usando FSx for Windows File Server com volumes do Amazon ECS e FSx for Windows File Server no Guia do desenvolvedor do Amazon Elastic Container Service.

Você pode referenciar imagens de contêiner em registros privados AWS que não exijam autenticação usando um segredo do Secrets Manager com as credenciais do registro. Para obter mais informações, consulte Autenticação de registro privado para tarefas no Guia do desenvolvedor do Amazon Elastic Container Service.

Quando você usa o Amazon ECS Service Connect, o Amazon ECS usa segredos gerenciados do Secrets Manager para armazenar certificados AWS Private Certificate Authority TLS. O custo de armazenar o segredo está incluído nas cobranças do Amazon ECS. Para atualizar o segredo, você deve usar o Amazon ECS em vez do Secrets Manager. Para obter mais informações, consulte TLS with Service Connect no Amazon Elastic Container Service Developer Guide.

Como a Amazon ElastiCache usa AWS Secrets Manager

Em ElastiCache você pode usar um recurso chamado Controle de Acesso Baseado em Funções (RBAC) para proteger o cluster. É possível armazenar essas credenciais no Secrets Manager. O Secrets Manager fornece um modelo de alternância para esse tipo de segredo. Para obter mais

informações, consulte Rotação automática de senhas para usuários no Guia do ElastiCache usuário da Amazon.

Como AWS Elemental Live usa AWS Secrets Manager

AWS Elemental Live é um serviço de vídeo em tempo real que permite criar saídas ao vivo para transmissão e entrega de streaming.

AWS Elemental Live usa um ARN secreto para obter um segredo que contém uma chave de criptografia do Secrets Manager. O Elemental Live usa a chave de criptografia para criptografar/descriptografar o vídeo. Para obter mais informações, consulte Como a entrega de AWS Elemental Live para MediaConnect funciona em tempo de execução no Guia do usuário do Elemental Live.

Como AWS Elemental MediaConnect usa AWS Secrets Manager

AWS Elemental MediaConnect é um serviço que torna mais fácil para emissoras e outros provedores de vídeo premium ingerir vídeo ao vivo de forma confiável Nuvem AWS e distribuí-lo para vários destinos dentro ou fora do. Nuvem AWS

É possível usar criptografia de chave estática para proteger suas fontes, saídas e direitos e armazenar sua chave de criptografia no AWS Secrets Manager. Para obter mais informações, consulte Criptografia de chave estática AWS Elemental MediaConnect no Guia AWS Elemental MediaConnect do usuário.

Como AWS Elemental MediaConvert usa AWS Secrets Manager

AWS Elemental MediaConvert é um serviço de processamento de vídeo baseado em arquivos que fornece processamento de vídeo escalável para proprietários e distribuidores de conteúdo com bibliotecas de mídia de qualquer tamanho. Para MediaConvert codificar as marcas d'água da Kantar, você usa o Secrets Manager para armazenar suas credenciais da Kantar. Para obter mais informações, consulte Usando o Kantar para marcas d'água de áudio nas AWS Elemental MediaConvert saídas no Guia do AWS Elemental MediaConvert usuário.

Como AWS Elemental MediaLive usa AWS Secrets Manager

AWS Elemental MediaLive é um serviço de vídeo em tempo real que permite criar saídas ao vivo para transmissão e entrega de streaming. Se sua organização usa AWS Elemental Link dispositivos

com AWS Elemental MediaLive ou AWS Elemental MediaConnect, você deve implantar o dispositivo e configurá-lo. Para obter mais informações, consulte Configurando MediaLive como uma entidade confiável no Guia do MediaLive usuário.

Como AWS Elemental MediaPackage usa AWS Secrets Manager

AWS Elemental MediaPackage é um serviço de embalagem e originação de just-in-time vídeos executado no Nuvem AWS. Com MediaPackage, você pode fornecer streams de vídeo altamente seguros, escaláveis e confiáveis para uma ampla variedade de dispositivos de reprodução e redes de entrega de conteúdo (CDNs). Para obter mais informações, consulte Acesso ao Secrets Manager para autorização de CDN no Guia do AWS Elemental MediaPackage Usuário.

Como AWS Elemental MediaTailor usa AWS Secrets Manager

AWS Elemental MediaTailor é um serviço escalável de inserção de anúncios e montagem de canais executado no. Nuvem AWS

MediaTailor suporta a autenticação de token de acesso do Secrets Manager em seus locais de origem. Com a autenticação do token de acesso do Secrets Manager, MediaTailor usa um segredo do Secrets Manager para autenticar solicitações em sua origem. Para obter mais informações, consulte Configurando a autenticação do token de AWS Secrets Manager acesso no Guia do AWS Elemental MediaTailor usuário.

Como o Amazon EMR usa o Secrets Manager

O Amazon EMR é uma plataforma que simplifica a execução de estruturas de big data, como Apache Hadoop e Apache Spark, para processar e analisar grandes quantidades de dados. AWS Ao usar essas estruturas e projetos de código aberto relacionados, como Apache Hive e Apache Pig, você pode processar dados para análise e workloads de business intelligence. Você também pode usar o Amazon EMR para transformar e mover grandes quantidades de dados para dentro e para fora de outros bancos de dados e bancos de AWS dados, como o Amazon S3 e o Amazon DynamoDB.

Como o Amazon EMR executado no Amazon EC2 usa o Secrets Manager

Ao criar um cluster no Amazon EMR, você pode fornecer dados de configuração de aplicações usando um segredo no Secrets Manager. Para obter mais informações, consulte Armazenar dados de configuração confidenciais no Secrets Manager no Guia de gerenciamento do Amazon EMR.

Além disso, quando você cria um bloco de anotações do EMR, é possível armazenar as credenciais de registro privadas baseadas no Git usando o Secrets Manager. Para obter mais informações, consulte Add a Git-based Repository to Amazon EMR (Adicionar um repositório baseado em Git ao Amazon EMR) no Guia de gerenciamento do Amazon EMR.

Como o EMR Serverless usa o Secrets Manager

O EMR Serverless fornece um ambiente de runtime com tecnologia sem servidor para simplificar a operação de aplicações de análise para que você não precise configurar, otimizar, proteger ou operar clusters.

Você pode armazenar seus dados AWS Secrets Manager e depois usar o ID secreto em suas configurações do EMR Serverless. Dessa forma, você não passa dados de configuração confidenciais em texto simples e os expõe a APIs externas.

Para obter mais informações, consulte Secrets Manager para proteção de dados com o EMR Serverless no Guia do usuário do Amazon EMR Serverless.

Como a Amazon EventBridge usa AWS Secrets Manager

EventBridge A Amazon é um serviço de barramento de eventos sem servidor que você pode usar para conectar seus aplicativos a dados de várias fontes.

Quando você cria um destino de EventBridge API da Amazon, EventBridge armazena a conexão para ele em um segredo gerenciado do Secrets Manager com o prefixo`events`. O custo de armazenamento do segredo está incluído na cobrança pelo uso de um destino de API. Para atualizar o segredo, você deve usar EventBridge em vez do Secrets Manager. Para obter mais informações, consulte Destinos de API no Guia EventBridge do usuário da Amazon.

Como o Amazon FSx usa segredos AWS Secrets Manager

O Amazon FSx para Windows File Server fornece servidores de arquivos Microsoft Windows totalmente gerenciados, baseados em um sistema de arquivos Windows totalmente nativo. Ao criar ou gerenciar compartilhamentos de arquivos, você pode passar credenciais de um AWS Secrets Manager segredo. Para obter mais informações, consulte File shares (Compartilhamentos de arquivos) e Migrating file share configurations to Amazon FSx (Migração de configurações de compartilhamento de arquivos para o Amazon FSx) no Guia do usuário do Amazon FSx para Windows File Server.

Como AWS Glue DataBrew usa AWS Secrets Manager

AWS Glue DataBrew é uma ferramenta visual de preparação de dados que você pode usar para limpar e normalizar dados sem escrever nenhum código. Em DataBrew, um conjunto de etapas de transformação de dados é chamado de receita. AWS Glue DataBrew fornece as etapas de `DETERMINISTIC_DECRYPT`, `DETERMINISTIC_ENCRYPT`, e `CRYPTOGRAPHIC_HASH`receita para realizar transformações em informações de identificação pessoal (PII) em um conjunto de dados, que usa uma chave de criptografia armazenada em um segredo do Secrets Manager. Se você usar o segredo DataBrew padrão para armazenar a chave de criptografia, DataBrew cria um segredo gerenciado com o prefixo`databrew`. O custo de armazenar o segredo está incluído na taxa de uso DataBrew. Se você criar um novo segredo para armazenar a chave de criptografia, DataBrew cria um segredo com o prefixo`AwsGlueDataBrew`. Haverá uma cobrança por esse segredo.

Como o AWS Glue Studio usa AWS Secrets Manager

AWS Glue Studio é uma interface gráfica que facilita a criação, execução e monitoramento de trabalhos de extração, transformação e carregamento (ETL) em AWS Glue. Você pode usar o Amazon OpenSearch Service como um armazenamento de dados para suas tarefas de extração, transformação e carregamento (ETL) configurando o Elasticsearch Spark Connector em. AWS Glue Studio Para se conectar ao OpenSearch cluster, você pode usar um segredo no Secrets Manager. Para obter mais informações, consulte Tutorial: Usando o conector AWS Glue para Elasticsearch no Guia do AWS Glue desenvolvedor.

Como AWS IoT SiteWise usa AWS Secrets Manager

AWS IoT SiteWise é um serviço gerenciado que permite coletar, modelar, analisar e visualizar dados de equipamentos industriais em grande escala. Você pode usar o AWS IoT SiteWise console para criar um gateway. Em seguida, adicione origens dos dados, servidores locais ou equipamentos industriais conectados a gateways. Se a sua origem exigir autenticação, use um segredo para autenticar. Para obter mais informações, consulte Configuring data source authentication (Configurar a autenticação da fonte de dados) Guia do usuário do AWS IoT SiteWise .

Como a Amazon Kendra usa AWS Secrets Manager

O Amazon Kendra é um serviço de pesquisa altamente preciso e inteligente que habilita seus usuários a pesquisarem dados não estruturados e estruturados usando processamento de linguagem natural e algoritmos de pesquisa avançados.

Você pode indexar documentos armazenados em um banco de dados especificando um segredo que contenha credenciais para o banco de dados. Para obter mais informações, consulteUsing a database data source (Usar uma fonte de dados de banco de dados) no Guia do usuário do Amazon Kendra.

Como o Amazon Kinesis Video Streams usa AWS Secrets Manager

Você pode usar o Amazon Kinesis Video Streams para se conectar a câmeras IP nas dependências do cliente, gravar e armazenar localmente vídeos das câmeras e transmitir vídeos para a nuvem para fins de armazenamento, reprodução e processamento analítico de longo prazo. Para gravar e carregar mídia de câmeras IP, você deve implantar o Kinesis Video Streams Edge Agent no AWS IoT Greengrass. Você armazena as credenciais necessárias para acessar os arquivos de mídia que são transmitidos para a câmera em um segredo do Secrets Manager. Para obter mais informações, consulte Implantar o Amazon Kinesis Video Streams Edge Agent no AWS IoT Greengrass no Guia do desenvolvedor do Amazon Kinesis Video Streams.

Como AWS Launch Wizard usa AWS Secrets Manager

AWS Launch Wizard for Active Directory é um serviço que aplica as melhores práticas de Nuvem AWS aplicativos para orientá-lo na configuração de uma nova infraestrutura do Active Directory ou na adição de controladores de domínio a uma infraestrutura existente, no local Nuvem AWS ou no local.

AWS Launch Wizard exige que as credenciais do administrador de domínio sejam adicionadas ao Secrets Manager para unir seus controladores de domínio ao Active Directory. Para obter mais informações, consulte Configurar AWS Launch Wizard para o Active Directory no Guia AWS Launch Wizard do Usuário.

Como o Amazon Lookout for Metrics usa o AWS Secrets Manager

O Amazon Lookout for Metrics é um serviço que localiza anomalias nos seus dados, determina suas causas raiz e permite tomar medidas rapidamente. O Amazon Redshift ou o Amazon RDS podem ser usados como fontes de dados para um detector do Lookout for Metrics. Para configurar a fonte de dados, você usa um segredo que contém a senha do banco de dados. Para obter mais informações, consulte Using Amazon RDS with Lookout for Metrics (Uso do Amazon RDS com o Lookout for Metrics) e Using Amazon Redshift with Lookout for Metrics (Uso do Amazon Redshift com o Lookout for Metrics) no Guia do desenvolvedor do Amazon Lookout for Metrics.

Como o Amazon Managed Grafana usa AWS Secrets Manager

O Amazon Managed Grafana é um serviço de visualização de dados totalmente gerenciado e seguro que você pode usar para consultar, correlacionar e visualizar instantaneamente métricas operacionais, logs e rastreamentos de várias fontes. Ao usar o Amazon Redshift como fonte de dados, você pode fornecer credenciais do Amazon Redshift usando um segredo. AWS Secrets Manager Para obter mais informações, consulte Configurando Amazon Redshift no Guia de ususário do Amazon Managed Grafana.

Como AWS Managed Services usa AWS Secrets Manager

AWS Managed Services é um serviço corporativo que fornece gerenciamento contínuo de sua AWS infraestrutura. O modo AMS Self-Service Provisioning (SSP) fornece acesso total aos recursos nativos AWS service (Serviço da AWS) e de API nas contas gerenciadas pelo AMS. Para obter informações sobre como solicitar acesso ao Secrets Manager no AMS, consulte AWS Secrets Manager (provisionamento de autoatendimento do AMS) no Guia avançado do usuário do AMS.

Como o Amazon Managed Streaming for Apache Kafka usa o AWS Secrets Manager

O Amazon Managed Streaming for Apache Kafka (Amazon MSK) é um serviço totalmente gerenciado que o habilita a criar e executar aplicações que usam o Apache Kafka para processar dados de transmissões. Você pode controlar o acesso aos seus clusters do Amazon MSK usando nomes de usuário e senhas armazenados e protegidos com o AWS Secrets Manager. Para obter mais informações, consulteautenticação de nome de usuário e senha com AWS Secrets Manager no Guia do desenvolvedor do Amazon Managed Streaming for Apache Kafka.

Como o Amazon Managed Workflows for Apache Airflow usa AWS Secrets Manager

O Amazon Managed Workflows for Apache Airflow é um serviço gerenciado de orquestração para o Apache Airflow que facilita a configuração e a operação de pipelines de end-to-end dados na nuvem em grande escala.

É possível configurar uma conexão do Apache Airflow usando um segredo do Secrets Manager. Para obter mais informações, consulte Configurando uma conexão do Apache Airflow usando um

segredo do Secrets Manager e Usando uma chave secreta AWS Secrets Manager para uma variável do Apache Airflow no Guia do usuário do Amazon Managed Workflows for Apache Airflow.

AWS Marketplace

Quando você usa o AWS Marketplace Quick Launch, AWS Marketplace distribui seu software junto com a chave de licença. AWS Marketplace armazena a chave de licença em sua conta como um segredo gerenciado pelo Secrets Manager. O custo de armazenar o segredo está incluído nas cobranças de AWS Marketplace. Para atualizar o segredo, você deve usar AWS Marketplace em vez do Secrets Manager. Para obter mais informações, consulte Configuração rápida no Guia do vendedor do AWS Marketplace .

Como AWS Migration Hub usa AWS Secrets Manager

AWS Migration Hub fornece um único local para rastrear as tarefas de migração em várias AWS ferramentas e soluções de parceiros.

AWS Migration Hub O Orchestrator simplifica e automatiza a migração de servidores e aplicativos corporativos para o. AWS O Migration Hub Orchestrator usa um segredo para as informações de conexão com o servidor de origem. Para obter mais informações, no Guia do usuário do AWS Migration Hub Orchestrator, consulte:

- Migre NetWeaver aplicativos SAP para AWS
- Redefinir a hospedagem de aplicações no Amazon EC2

O Migration Hub Strategy Recommendations oferece recomendações de estratégia de migração e modernização para caminhos de transformação viáveis para suas aplicações. O Strategy Recommendations pode analisar bancos de dados do SQL Server, usando um segredo para as informações de conexão. Para obter mais informações, consulte Análise de banco de dados do Strategy Recommendations.

Como AWS Panorama usa o Secrets Manager

AWS Panorama é um serviço que traz a visão computacional para sua rede de câmeras local. Você usa AWS Panorama para registrar um equipamento, atualizar seu software e implantar aplicativos nele. Quando você registra um stream de vídeo como fonte de dados para seu aplicativo, se o stream estiver protegido por senha, AWS Panorama armazena as credenciais dele em um segredo

do Secrets Manager. Para obter mais informações, consulte Gerenciar fluxo de câmeras no AWS Panorama no AWS Panorama Guia do desenvolvedor .

Como AWS ParallelCluster usa AWS Secrets Manager

AWS ParallelCluster é uma ferramenta de gerenciamento de cluster de código aberto que você pode usar para implantar e gerenciar clusters de computação de alto desempenho (HPC) no Nuvem AWS. Você pode criar um ambiente de vários usuários que inclua um AWS ParallelCluster que esteja integrado a um Microsoft AD AWS gerenciado (Active Directory). O AWS ParallelCluster usa um segredo do Secrets Manager para validar logins no Active Directory. Para obter mais informações, consulte Integrar o Active Directory no Guia do usuário do AWS ParallelCluster .

Como o Amazon Q usa o Secrets Manager

Para autenticar o Amazon Q para acessar sua fonte de dados, você fornece suas credenciais de acesso à fonte de dados para o Amazon Q usando um segredo do Secrets Manager. Se você usa o console, pode escolher criar um novo segredo ou usar um existente. Para obter mais informações, consulte Conceitos - Autenticação no Amazon Q Developer Guide.

Como AWS OpsWorks for Chef Automate usa AWS Secrets Manager

AWS OpsWorks é um serviço de gerenciamento de configuração que ajuda você a configurar e operar aplicativos em uma empresa em nuvem usando o OpsWorks Puppet Enterprise ou AWS OpsWorks for Chef Automate.

Quando você cria um novo servidor em AWS OpsWorks CM, o OpsWorks CM armazena informações do servidor em um segredo gerenciado do Secrets Manager com o prefixo`opsworks-cm`. O custo do segredo está incluído na cobrança do AWS OpsWorks. Para obter mais informações, consulte Integration with AWS Secrets Manager (Integração com o) no Guia do usuário do AWS OpsWorks .

Como a Amazon QuickSight usa AWS Secrets Manager

QuickSight A Amazon é um serviço de inteligência de negócios (BI) em escala de nuvem que você pode usar para análises, visualização de dados e relatórios. Você pode usar uma variedade de fontes de dados na Amazon QuickSight. Se você armazenar as credenciais do banco de dados nos

segredos do Secrets Manager, a Amazon QuickSight poderá usar esses segredos para se conectar aos bancos de dados. Para obter mais informações, consulte Usando AWS Secrets Manager segredos no lugar de credenciais de banco de dados na Amazon QuickSight no Guia do QuickSight usuário da Amazon.

Amazon RDS

O Amazon Relational Database Service (Amazon RDS) é um serviço da Web que facilita a configuração, a operação e escalabilidade de um banco de dados relacional na Nuvem AWS.

Para gerenciar as credenciais de usuário mestre do Amazon Relational Database Service (Amazon RDS), incluindo o Aurora, o Amazon RDS pode criar um segredo gerenciado para você. Haverá uma cobrança por esse segredo. O Amazon RDS também gerencia a alternância dessas credenciais. Para obter mais informações, consulte Gerenciamento de senhas com Amazon RDS e o AWS Secrets Manager no Guia do usuário do Amazon RDS.

Para outras credenciais do Amazon RDS, consulte the section called “Criar um segredo de banco de dados”.

Ao usar o editor de consultas do Amazon RDS para se conectar a um banco de dados, você pode armazenar credenciais para o banco de dados no Secrets Manager. Para obter mais informações, consulte Usar o editor de consultas no Guia do usuário do Amazon RDS.

Como o Amazon Redshift usa AWS Secrets Manager

O Amazon Redshift é um serviço de data warehouse totalmente gerenciado e em escala de petabytes na Nuvem .

Para gerenciar as credenciais de administrador do Amazon Redshift, o Amazon Redshift pode criar um segredo gerenciado para você. Haverá uma cobrança por esse segredo. O Amazon Redshift também gerencia a rotação dessas credenciais. Para obter mais informações, consulte Gerenciamento de senhas de administrador do Amazon Redshift usando o Guia de gerenciamento AWS Secrets Manager de clusters do Amazon Redshift.

Para outras credenciais do Amazon Redshift, consulte the section called “Criar um segredo de banco de dados”.

Ao chamar a API de dados do Amazon Redshift, você pode passar credenciais para o cluster usando um segredo no Secrets Manager. Para obter mais informações, consulte Uso da API de dados do Amazon Redshift.

Quando você usa o editor de consultas do Amazon Redshift para se conectar a um banco de dados, o Amazon Redshift pode armazenar suas credenciais em um segredo do Secrets Manager com o prefixo `redshiftqueryeditor`. Haverá uma cobrança por esse segredo. Para obter mais informações, consulte Consultar um banco de dados usando o editor de consultas no Guia de gerenciamento do Amazon Redshift.

Para o editor de consultas v2, consulte the section called “Editor de Consultas do Amazon Redshift v2”.

Editor de Consultas do Amazon Redshift v2

O editor de consultas v2 do Amazon Redshift é uma aplicação de cliente SQL baseada na Web que você pode usar para criar e executar consultas no data warehouse do Amazon Redshift. Quando você usa o editor de consultas v2 do Amazon Redshift para se conectar a um banco de dados, o Amazon Redshift pode armazenar suas credenciais em um segredo gerenciado do Secrets Manager com o prefixo. `sqlworkbench` O custo de armazenamento do segredo está incluído na cobrança pelo uso do Amazon Redshift. Para atualizar o segredo, você deve usar o Amazon Redshift em vez do Secrets Manager. Para obter mais informações, consulte Trabalhar com o editor de consultas v2 no Guia de gerenciamento do Amazon Redshift.

Para a versão anterior do editor de consultas, consulte the section called “Amazon Redshift”.

Como a Amazon SageMaker usa AWS Secrets Manager

SageMaker é um serviço de aprendizado de máquina totalmente gerenciado. Com isso SageMaker, cientistas e desenvolvedores de dados podem criar e treinar modelos de aprendizado de máquina com rapidez e facilidade e, em seguida, implantá-los diretamente em um ambiente hospedado pronto para produção. O serviço oferece uma instância de notebook de autoria Jupyter integrado para facilitar o acesso a fontes de dados para fins de exploração e análise, sem necessidade de gerenciar servidores.

Você pode associar repositórios do Git a suas instâncias do caderno Jupyter para salvar seus cadernos de anotações em um ambiente de controle de origem que persista mesmo se você interromper ou excluir sua instância de bloco de anotações. É possível gerenciar suas credenciais de repositórios privados usando o Secrets Manager. Para obter mais informações, consulte Associar repositórios Git às instâncias do Amazon SageMaker Notebook no Amazon Developer Guide. SageMaker

Para importar dados do Databricks, o Data Wrangler armazena sua URL do JDBC no Secrets Manager. Para obter mais informações, consulte Importar dados do Databricks (JDBC).

Para importar dados do snowflake, o Data Wrangler armazena suas credenciais em um segredo do Secrets Manager. Para obter mais informações, consulte Importar dados do Snowflake.

Como AWS Schema Conversion Tool usa AWS Secrets Manager

Você pode usar o AWS Schema Conversion Tool (AWS SCT) para converter seu esquema de banco de dados existente de um mecanismo de banco de dados para outro. Você pode converter o esquema OLTP relacional ou o esquema de data warehouse. Seu esquema convertido é adequado para um Amazon Relational Database Service (Amazon RDS) MySQL, MariaDB, Oracle, SQL Server, PostgreSQL DB, um cluster Amazon Aurora DB ou um cluster Amazon Redshift. O esquema convertido também pode ser usado com um banco de dados em uma instância do Amazon Elastic Compute Cloud ou armazenado como dados em um bucket do S3.

Quando você converte um esquema de banco de dados, AWS SCT pode usar as credenciais do banco de dados que você armazena. AWS Secrets Manager Para obter mais informações, consulte Usando AWS Secrets Manager na interface AWS SCT do usuário no Guia do AWS Schema Conversion Tool usuário.

Como AWS Toolkit for JetBrains usa AWS Secrets Manager

O AWS Toolkit for JetBrains é um plug-in de código aberto para os ambientes de desenvolvimento integrado (IDEs) do JetBrains. Esse toolkit facilita o desenvolvimento, a depuração e a implantação de aplicações sem servidor para desenvolvedores que usam AWS. Ao conectar a um cluster do Amazon Redshift usando o toolkit, você pode se autenticar usando um segredo do Secrets Manager. Para obter mais informações, consulte Accessing Amazon Redshift clusters (Acesso a clusters do Amazon Redshift) no Guia do usuário do AWS Toolkit for JetBrains .

Como AWS Transfer Family usa AWS Secrets Manager segredos

AWS Transfer Family é um serviço de transferência segura que permite transferir arquivos para dentro e para fora dos serviços de AWS armazenamento.

O Transfer Family agora oferece suporte ao uso da autenticação básica para servidores que usam o protocolo Applicability Statement 2 (AS2). Você pode criar um novo segredo do Secrets Manager ou

escolher um segredo existente para suas credenciais. Para obter mais informações, consulte Basic authentication for AS2 connectors (Autenticação básica para conectores AS2) no AWS Transfer Family Guia do usuário.

Para autenticar usuários do Transfer Family, você pode usar AWS Secrets Manager como provedor de identidade. Para obter mais informações, consulte Trabalhando com provedores de identidade personalizados no Guia do AWS Transfer Family usuário e no artigo do blog Habilitar a autenticação por senha para AWS Transfer Family uso AWS Secrets Manager.

Você pode usar a decodificação Pretty Good Privacy (PGP) com os arquivos que o Transfer Family processa com fluxos de trabalho. Para usar a descriptografia em uma etapa do fluxo de trabalho, você fornece uma chave PGP que gerencia no Secrets Manager. Para obter mais informações, consulte Generate and manage PGP keys (Geração e gerenciamento de chaves PGP) no AWS Transfer Family Guia do usuário.

Como AWS Wickr usa segredos AWS Secrets Manager

AWS Wickr é um serviço end-to-end criptografado que ajuda organizações e agências governamentais a se comunicarem com segurança por meio one-to-one de mensagens em grupo, chamadas de voz e vídeo, compartilhamento de arquivos, compartilhamento de tela e muito mais. Você pode automatizar fluxos de trabalho usando bots de retenção de dados Wickr. Se o bot tiver acesso Serviços da AWS, você deverá criar um segredo do Secrets Manager para armazenar as credenciais do bot. Para obter mais informações, consulte Iniciar o bot de retenção de dados no AWS Guia de administração do Wickr.

Uso de um endpoint da VPC do AWS Secrets Manager

Sempre que possível, recomendamos que você execute o máximo da infraestrutura em redes privadas que não sejam acessíveis pela Internet pública. É possível estabelecer uma conexão privada entre a sua VPC e o Secrets Manager criando um interface VPC endpoint (Endpoint da VPC de interface). Os endpoints de interface são habilitados por AWS PrivateLink, uma tecnologia que permite acessar de forma privada as APIs diretas do Secrets Manager sem um gateway da Internet, um dispositivo NAT, uma conexão VPN ou uma conexão do AWS Direct Connect. As instâncias na VPC não precisam de endereços IP públicos para a comunicação com APIs do Secrets Manager. O tráfego entre sua VPC e o Secrets Manager não sai da rede da AWS. Para obter mais informações, consulte Endpoints da VPC da interface (AWS PrivateLink) no Manual do Usuário do Amazon VPC.

Quando o Secrets Manager alterna um segredo usando uma função de alternância do Lambda, por exemplo, um segredo que contém credenciais de banco de dados, a função do Lambda faz solicitações para o banco de dados e para o Secrets Manager. Quando você ativa a alternância automática usando o console, o Secrets Manager cria a função do Lambda na mesma VPC do banco de dados. Recomendamos que você crie um endpoint do Secrets Manager na mesma VPC para que as solicitações da função de alternância do Lambda para o Secrets Manager não saiam da rede da Amazon.

Se você habilitar o DNS privado para o endpoint, poderá fazer solicitações de API para o Secrets Manager usando seu nome DNS padrão para a região, por exemplo, `secretsmanager.us-east-1.amazonaws.com`. Para obter mais informações, consulte Acessar um serviço por um endpoint de interface no Guia do usuário da Amazon VPC.

Você pode verificar se as solicitações para o Secrets Manager vêm do acesso à VPC ao incluir uma condição nas políticas de permissões. Para ter mais informações, consulte the section called “Exemplo: Permissões e VPCs”.

Você pode usar os logs do AWS CloudTrail para auditar o uso de segredos por meio do endpoint da VPC.

Para criar um endpoint da VPC do Secrets Manager

1. Consulte Criação de um endpoint de interface no Guia do usuário da Amazon VPC. Use o nome do serviço: com.amazonaws.*region*.secretsmanager
2. Para controlar o acesso ao endpoint, consulte Controle o acesso aos endpoints da VPC usando políticas de endpoint.

Sub-redes compartilhadas

Você não pode criar, descrever, modificar ou excluir endpoints da VPC em sub-redes que são compartilhadas com você. No entanto, você pode usar os endpoints da VPC em sub-redes que são compartilhadas com você. Para obter informações sobre o compartilhamento de VPC, consulte Compartilhar sua VPC com outras contas no Guia do usuário do Amazon Virtual Private Cloud.

Criação de segredos do AWS Secrets Manager no AWS CloudFormation

Você pode criar segredos em uma pilha do CloudFormation usando o recurso `AWS::SecretsManager::Secret` em um modelo do CloudFormation, como mostrado no Criar um segredo.

Para criar um segredo de administrador para Amazon RDS ou Aurora, recomendamos que você use `ManageMasterUserPassword` em `AWS::RDS::DBCluster`. Em seguida, o Amazon RDS cria o segredo e gerencia a alternância para você. Para obter mais informações, consulte Alternância gerenciada.

Para credenciais do Amazon Redshift e do Amazon DocumentDB, primeiro, crie um segredo com uma senha gerada pelo Secrets Manager e, em seguida, usar uma referência dinâmica para recuperar o nome de usuário e a senha do segredo para usar como credenciais para um novo banco de dados. Em seguida, use o recurso `AWS::SecretsManager::SecretTargetAttachment` para adicionar detalhes sobre o banco de dados ao segredo de que o Secrets Manager precisa para alternar o segredo. Por fim, para ativar a alternância automática, use o recurso `AWS::SecretsManager::RotationSchedule` e forneça uma função de rotação e um cronograma. Veja os exemplos a seguir:

- Criar um segredo com credenciais do Amazon Redshift
- Criar um segredo com credenciais do Amazon DocumentDB

Para anexar uma política de recursos ao seu segredo, use o recurso `AWS::SecretsManager::ResourcePolicy`.

Para obter informações sobre como criar recursos com o AWS CloudFormation, consulte Saiba mais sobre noções básicas de modelo no Guia do usuário do AWS CloudFormation. Você também pode usar o AWS Cloud Development Kit (AWS CDK) Para obter mais informações, consulte Biblioteca de construções do AWS Secrets Manager.

Criar um segredo do AWS Secrets Manager usando o AWS CloudFormation

Este exemplo cria um segredo denominado **`CloudFormationCreatedSecret-`*`a1b2c3d4e5f6`***. O valor do segredo é o JSON seguinte, com uma senha de 32 caracteres gerada na criação do segredo.

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

Este exemplo usa os seguintes recursos do CloudFormation:

- `AWS::SecretsManager::Secret`

Para obter informações sobre como criar recursos com o AWS CloudFormation, consulte Saiba mais sobre noções básicas de modelo no Guia do usuário do AWS CloudFormation.

JSON

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

YAML

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

Criação de segredos do AWS Secrets Manager com alternância automática e uma instância de banco de dados MySQL do Amazon RDS com o AWS CloudFormation

Para criar um segredo de administrador para Amazon RDS ou Aurora, recomendamos que você use `ManageMasterUserPassword`, conforme mostrado no exemplo Criar um segredo do Secrets Manager para uma senha mestre em `AWS::RDS::DBCluster`. Em seguida, o Amazon RDS cria o segredo e gerencia a alternância para você. Para obter mais informações, consulte Alternância gerenciada.

Crie um AWS Secrets Manager segredo e um cluster do Amazon Redshift com AWS CloudFormation

Para criar um segredo administrativo para o Amazon Redshift, recomendamos que você use os exemplos em e. `AWS::Redshift::ClusterAWS::RedshiftServerless::Namespace`

Crie um AWS Secrets Manager segredo e uma instância do Amazon DocumentDB com AWS CloudFormation

Este exemplo cria um segredo e uma instância do Amazon DocumentDB usando as credenciais no segredo como usuário e senha. O segredo tem uma política baseada em recursos anexada que define quem pode acessar o segredo. O modelo também cria uma função de alternância do Lambda a partir de Modelos de função de alternância e configura o segredo para alternar automaticamente

entre às 8h e 10h UTC do primeiro dia de cada mês. Como uma prática recomendada de segurança, a instância está em uma Amazon VPC.

Este exemplo usa os seguintes CloudFormation recursos para o Secrets Manager:

- AWS::SecretsManager::Secret
- AWS::SecretsManager::SecretTargetAttachment
- AWS::SecretsManager::RotationSchedule

Para obter informações sobre a criação de recursos com AWS CloudFormation, consulte Aprenda os conceitos básicos do modelo no Guia do AWS CloudFormation usuário.

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 * ? *)
```

Como o Secrets Manager usa AWS CloudFormation

Quando você usa o console para ativar a rotação, o Secrets Manager usa AWS CloudFormation para criar recursos para rotação. Se você criar uma nova função de rotação durante esse processo, o AWS CloudFormation cria um `AWS::Serverless::Function` com base no Modelos de função de alternância apropriado. Em seguida, o AWS CloudFormation define a `RotationSchedule`, que estabelece a função de alternância e as regras de alternância para o segredo. Você pode visualizar a pilha de AWS CloudFormation escolhendo View stack (Visualizar pilha) no banner depois de ativar a rotação automática.

Para obter informações sobre como ativar a rotação automática, consulte Alternar segredos .

Crie AWS Secrets Manager segredos em AWS Cloud Development Kit (AWS CDK)

Para criar, gerenciar e recuperar segredos em um aplicativo CDK, você pode usar a AWS Secrets Manager Construct Library, que contém construções `ResourcePolicy`,, `RotationScheduleSecretSecretRotation` e `SecretTargetAttachment`.

Uma boa prática para usar segredos em aplicativos CDK é primeiro criar o segredo usando o console ou a CLI e, em seguida, importar o segredo para seu aplicativo CDK.

Para ver exemplos, consulte:

- Criar um segredo
- Importar um segredo
- Recuperar segredos
- Conceder permissão para usar o segredo
- Alternar um segredo
- Alternar um segredo de banco de dados
- Replicar um segredo para outras regiões

Para obter mais informações sobre o CDK, consulte o AWS Cloud Development Kit (AWS CDK) Guia do desenvolvedor v2.

Monitore AWS Secrets Manager segredos

AWS fornece ferramentas de monitoramento para monitorar os segredos do Secrets Manager, relatar quando algo está errado e realizar ações automáticas quando apropriado. Você pode usar os logs se precisar investigar qualquer uso ou alteração inesperada e, em seguida, reverter as alterações indesejadas. Você também pode estabelecer verificações automatizadas para o uso inadequado de segredos e qualquer tentativa de exclusão de segredos.

Tópicos

AWS Secrets Manager Registre eventos com AWS CloudTrail

AWS CloudTrail registra todas as chamadas de API para o Secrets Manager como eventos, incluindo chamadas do console do Secrets Manager, bem como vários outros eventos para rotação e exclusão de versões secretas. Para obter uma lista das entradas de registro nos registros do Secrets Manager, consulteCloudTrail entradas.

Você pode usar o CloudTrail console para ver os últimos 90 dias de eventos gravados. Para um registro contínuo de eventos em sua AWS conta, incluindo eventos para o Secrets Manager, crie uma trilha para CloudTrail entregar arquivos de log para um bucket do Amazon S3. Consulte Criação de uma trilha para sua AWS conta. Você também pode configurar CloudTrail para receber arquivos de CloudTrail log de vários Contas da AWSRegiões da AWSe.

Você pode configurar outros AWS serviços para analisar e agir com base nos dados coletados nos CloudTrail registros. Veja as integrações de AWS serviços com CloudTrail registros. Você também pode receber notificações ao CloudTrail publicar novos arquivos de log em seu bucket do Amazon S3. Consulte Configuração de notificações do Amazon SNS para. CloudTrail

Para recuperar eventos do Secrets Manager dos CloudTrail registros (console)

1. Abra o CloudTrail console em https://console.aws.amazon.com/cloudtrail/.
2. Certifique-se de que o console aponte para a região onde seus eventos ocorreram. O console mostra somente os eventos que ocorreram na região selecionada. Escolha a região na lista suspensa no canto superior direito do console.
3. No painel de navegação à esquerda, escolha Histórico de eventos.
4. Escolha o critério Filter (Filtrar) e/ou um Time range (Período) para ajudar a encontrar o evento que você está procurando. Por exemplo: .

    a. Para ver todos os eventos do Secrets Manager, em Atributos de pesquisa, escolha Origem do evento. Em seguida, em Enter envent source (Inserir origem do evento), escolha **`secretsmanager.amazonaws.com`**.

    b. Para ver todos os eventos de um segredo, em Atributos de pesquisa, escolha Nome do recurso. Em seguida, em Inserir um nome de recurso, insira o nome do segredo.
5. Para ver detalhes adicionais, escolha a seta de expansão ao lado do evento. Para ver todas as informações disponíveis, escolha View evento (Visualizar evento).

AWS CLI

Example Recupere eventos do Secrets Manager dos registros CloudTrail

O exemplo de `lookup-events` a seguir procura eventos do Secrets Manager.

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

AWS CloudTrail entradas para Secrets Manager

AWS Secrets Manager grava entradas em seu AWS CloudTrail log para todas as operações do Secrets Manager e para outros eventos relacionados à rotação e exclusão. Para obter informações sobre como agir nesses eventos, consulte Combine eventos do Secrets Manager com EventBridge.

Tipos de entrada de log

- Entradas de log para operações do Secrets Manager

- Entradas de log para exclusão
- Entradas de log para replicação
- Entradas de log para alternância

Entradas de log para operações do Secrets Manager

Os eventos que são gerados por chamadas para as operações do Secrets Manager têm `"detail-type": ["AWS API Call via CloudTrail"]`.

> **Note**
>
> Antes de fevereiro de 2024, algumas operações do Secrets Manager relataram eventos que continham “ARN” em vez de “arn” para o ARN secreto. Para obter mais informações, consulte AWS re:Post.

A seguir estão as CloudTrail entradas geradas quando você ou um serviço chamam as operações do Secrets Manager por meio da API, SDK ou CLI.

BatchGetSecretValue

Gerado pela BatchGetSecretValueoperação. Para obter informações sobre a recuperação de segredos, consulte Obtenha segredos.

CancelRotateSecret

Gerado pela CancelRotateSecretoperação. Para obter informações sobre alternância, consulte Alternar segredos .

CreateSecret

Gerado pela CreateSecretoperação. Para obter informações sobre a criação de segredos, consulte Criar e gerenciar segredos.

DeleteResourcePolicy

Gerado pela DeleteResourcePolicyoperação. Para obter mais informações sobre permissões, consulte Autenticação e controle de acesso.

DeleteSecret

Gerado pela DeleteSecretoperação. Para obter informações sobre como excluir segredos, consulte the section called “Excluir um segredo”.

DescribeSecret

Gerado pela DescribeSecretoperação.

GetRandomPassword

Gerado pela GetRandomPasswordoperação.

GetResourcePolicy

Gerado pela GetResourcePolicyoperação. Para obter mais informações sobre permissões, consulte Autenticação e controle de acesso.

GetSecretValue

Gerado pelas BatchGetSecretValueoperações GetSecretValuee. Para obter informações sobre a recuperação de segredos, consulte Obtenha segredos.

ListSecrets

Gerado pela ListSecretsoperação. Para obter informações sobre a listagem de segredos, consulte the section called “Localizar segredos”.

ListSecretVersionIds

Gerado pela ListSecretVersionIdsoperação.

PutResourcePolicy

Gerado pela PutResourcePolicyoperação. Para obter mais informações sobre permissões, consulte Autenticação e controle de acesso.

PutSecretValue

Gerado pela PutSecretValueoperação. Para obter informações sobre como atualizar um segredo, consulte the section called “Modificar um segredo”.

RemoveRegionsFromReplication

Gerado pela RemoveRegionsFromReplicationoperação. Para obter informações sobre a replicação de um segredo, consulte Replique segredos em todas as regiões.

ReplicateSecretToRegions

Gerado pela ReplicateSecretToRegionsoperação. Para obter informações sobre a replicação de um segredo, consulte Replique segredos em todas as regiões.

RestoreSecret

Gerado pela RestoreSecretoperação. Para obter informações sobre como restaurar um segredo excluído, consulte the section called “Restaurar um segredo”.

RotateSecret

Gerado pela RotateSecretoperação. Para obter informações sobre alternância, consulte Alternar segredos .

StopReplicationToReplica

Gerado pela StopReplicationToReplicaoperação. Para obter informações sobre a replicação de um segredo, consulte Replique segredos em todas as regiões.

TagResource

Gerado pela TagResourceoperação. Para obter informações sobre como marcar um segredo, consulte the section called “Marcação de segredos do ”.

UntagResource

Gerado pela UntagResourceoperação. Para obter informações sobre como desmarcar um segredo, consulte the section called “Marcação de segredos do ”.

UpdateSecret

Gerado pela UpdateSecretoperação. Para obter informações sobre como atualizar um segredo, consulte the section called “Modificar um segredo”.

UpdateSecretVersionStage

Gerado pela UpdateSecretVersionStageoperação. Para obter informações sobre estágios de versão, consulte the section called “Versões secretas”.

ValidateResourcePolicy

Gerado pela ValidateResourcePolicyoperação. Para obter mais informações sobre permissões, consulte Autenticação e controle de acesso.

Entradas de log para exclusão

Além dos eventos das operações do Secrets Manager, o Secrets Manager gera os eventos a seguir relacionados à exclusão. Esses eventos têm `"detail-type": ["AWS Service Event via CloudTrail"]`.

CancelSecretVersionDelete

Gerado pelo serviço do Secrets Manager. Se você chamar `DeleteSecret` em um segredo que tenha versões e, posteriormente, chamar `RestoreSecret`, o Secrets Manager registra esse evento para cada versão de segredo que foi restaurada. Para obter informações sobre como restaurar um segredo excluído, consulte the section called “Restaurar um segredo”.

EndSecretVersionDelete

Gerado pelo serviço do Secrets Manager quando uma versão do segredo é excluída. Para ter mais informações, consulte the section called “Excluir um segredo”.

StartSecretVersionDelete

Gerado pelo serviço do Secrets Manager quando ele começa a exclusão da versão de um segredo. Para obter informações sobre como excluir segredos, consulte the section called “Excluir um segredo”.

SecretVersionDeletion

Gerado pelo serviço do Secrets Manager quando ele começa a exclusão da versão de um segredo. Para obter mais informações, consulte Secret versions (Versões de segredos).

Entradas de log para replicação

Além dos eventos das operações do Secrets Manager, o Secrets Manager gera os eventos a seguir relacionados à replicação. Esses eventos têm `"detail-type": ["AWS Service Event via CloudTrail"]`.

ReplicationFailed

Gerado pelo serviço do Secrets Manager quando a replicação falha. Para obter informações sobre a replicação de um segredo, consulte Replique segredos em todas as regiões.

ReplicationStarted

Gerado pelo serviço Secrets Manager quando ele começa a replicar um segredo. Para obter informações sobre a replicação de um segredo, consulte Replique segredos em todas as regiões.

ReplicationSucceeded

Gerado pelo serviço do Secrets Manager quando um segredo é replicado com sucesso. Para obter informações sobre a replicação de um segredo, consulte Replique segredos em todas as regiões.

Entradas de log para alternância

Além dos eventos das operações do Secrets Manager, o Secrets Manager gera os eventos a seguir relacionados à alternância. Esses eventos têm `"detail-type": ["AWS Service Event via CloudTrail"]`.

RotationStarted

Gerado pelo serviço Secrets Manager quando ele começa a alternar um segredo. Para obter informações sobre alternância, consulte Alternar segredos .

RotationAbandoned

Gerado pelo serviço do Secrets Manager quando ele abandona uma tentativa de alternância e remove o rótulo `AWSPENDING` de uma versão existente de um segredo. O Secrets Manager abandona a rotação quando você cria uma nova versão de um segredo durante a alternância. Para obter informações sobre alternância, consulte Alternar segredos .

RotationFailed

Gerado pelo serviço do Secrets Manager quando a alternância falha. Para obter informações sobre alternância, consulte the section called “Solução de problemas de alternância do ”.

RotationSucceeded

Gerado pelo serviço do Secrets Manager quando um segredo é alternado com êxito. Para obter informações sobre alternância, consulte Alternar segredos .

TestRotationStarted

Gerado pelo serviço do Secrets Manager quando ele começa a testar a alternância de um segredo que não está programado para alternância imediata. Para obter informações sobre alternância, consulte Alternar segredos .

TestRotationSucceeded

Gerado pelo serviço do Secrets Manager quando ele testa com êxito a alternância de um segredo que não está programado para alternância imediata. Para obter informações sobre alternância, consulte Alternar segredos .

TestRotationFailed

Gerado pelo serviço do Secrets Manager quando ele testa a alternância de um segredo que não está programado para alternância imediata e a alternância falhou. Para obter informações sobre alternância, consulte the section called “Solução de problemas de alternância do ”.

Monitore AWS Secrets Manager com a Amazon CloudWatch

Usando a Amazon CloudWatch, você pode monitorar AWS serviços e criar alarmes para avisar quando as métricas mudam. CloudWatch mantém essas estatísticas por 15 meses, para que você possa acessar informações históricas e ter uma melhor perspectiva sobre o desempenho de seu aplicativo ou serviço web. Pois AWS Secrets Manager, você pode monitorar o número de segredos em sua conta, incluindo segredos marcados para exclusão e chamadas de API para o Secrets Manager, incluindo chamadas feitas por meio do console. Para obter informações sobre como monitorar métricas, consulte Usar CloudWatch métricas no Guia do CloudWatch usuário.

Para encontrar métricas do Secrets Manager

1. No CloudWatch console, em Métricas, escolha Todas as métricas.
2. Na caixa Pesquisa de métricas, digite`secret`.
3. Faça o seguinte:
   - Para monitorar o número de segredos em sua conta, escolha AWS/eSecretsManager, em seguida, selecione SecretCount. Essa métrica é publicada de hora em hora.
   - Para monitorar chamadas de API para o Secrets Manager, incluindo chamadas feitas pelo console, escolha Uso > Por AWS recurso e selecione as chamadas de API a serem monitoradas. Para obter uma lista das APIs do Secrets Manager, consulte Operações do Secrets Manager.
4. Faça o seguinte:
   - Para criar um gráfico da métrica, consulte Representação gráfica de métricas no Guia do CloudWatch usuário da Amazon.

- Para detectar anomalias, consulte Usando a detecção de CloudWatch anomalias no Guia do usuário da Amazon CloudWatch .
- Para obter estatísticas de uma métrica, consulte Obter estatísticas de uma métrica no Guia CloudWatch do usuário da Amazon.

CloudWatch alarmes

Você pode criar um CloudWatch alarme que envia uma mensagem do Amazon SNS quando o valor de uma métrica muda e faz com que o alarme mude de estado. Você pode definir um alarme na métrica do Secrets Manager`ResourceCount`, que é o número de segredos em sua conta. Você também pode definir alarmes em Um alarme observa uma métrica durante um período de tempo especificado e executa ações com base no valor da métrica em relação a um determinado limite em vários períodos de tempo. Os alarmes invocam ações somente para mudanças de estado sustentadas. CloudWatch os alarmes não invocam ações simplesmente porque estão em um determinado estado; o estado deve ter sido alterado e mantido por um determinado número de períodos.

Para obter mais informações, consulte Usando CloudWatch alarmes da Amazon e Criar um CloudWatch alarme com base na detecção de anomalias no Guia do CloudWatch usuário.

Você também pode definir alarmes que observam determinados limites e enviam notificações ou realizam ações quando esses limites são atingidos. Para obter mais informações, consulte o Guia CloudWatch do usuário da Amazon.

Combine AWS Secrets Manager eventos com a Amazon EventBridge

Na Amazon EventBridge, você pode combinar eventos do Secrets Manager a partir de entradas de CloudTrail registro. Você pode configurar EventBridge regras que procurem esses eventos e, em seguida, enviem novos eventos gerados a um alvo para agir. Para obter uma lista das CloudTrail entradas que o Secrets Manager registra, consulteCloudTrail entradas. Para obter instruções de configuração EventBridge, consulte Introdução EventBridge no Guia do EventBridge usuário.

Corresponder todas as alterações com um segredo especificado

> **Note**
>
> Como alguns eventos do Secrets Manager devolvem o ARN do segredo com uma capitalização diferente, em padrões de eventos que correspondem a mais de uma ação, para especificar um segredo por ARN, talvez seja necessário incluir ambas as chaves `arn` e `aRN`. Para obter mais informações, consulte AWS re:Post.

O exemplo a seguir mostra um padrão de EventBridge evento que corresponde às entradas de registro para alterações em um segredo.

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

Corresponder eventos quando um valor do segredo é alternado

O exemplo a seguir mostra um padrão de EventBridge evento que corresponde às entradas de CloudTrail registro para alterações de valores secretos que ocorrem a partir de atualizações manuais ou rotação automática. Como alguns desses eventos são provenientes de operações do Secrets Manager e outros são gerados pelo serviço do Secrets Manager, você deve incluir o `detail-type` para ambos.

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

Monitore quando AWS Secrets Manager os segredos programados para exclusão são acessados

Você pode usar uma combinação de AWS CloudTrail Amazon CloudWatch Logs e Amazon Simple Notification Service (Amazon SNS) para criar um alarme que notifique você sobre qualquer tentativa de acessar uma exclusão secreta pendente. Se você receber uma notificação de um alarme, talvez queira cancelar a exclusão do segredo para ter mais tempo para decidir se deseja realmente excluí-lo. Sua investigação talvez resulte na restauração do segredo porque ainda precisa dele. Como alternativa, talvez seja necessário atualizar o usuário com detalhes do novo segredo a ser usado.

Os procedimentos a seguir explicam como receber uma notificação quando uma solicitação para a `GetSecretValue` operação resulta em uma mensagem de erro específica gravada em seus arquivos de CloudTrail log. É possível executar outras operações de API no segredo sem acionar o alarme. Esse CloudWatch alarme detecta um uso que pode indicar que uma pessoa ou aplicativo está usando credenciais desatualizadas.

Antes de iniciar esses procedimentos, você deve ativar a conta Região da AWS e CloudTrail na qual pretende monitorar as solicitações de AWS Secrets Manager API. Para obter instruções, consulte Criar uma trilha pela primeira vez no AWS CloudTrail Manual do usuário.

Etapa 1: configurar a entrega do arquivo de CloudTrail log para o CloudWatch Logs

Você deve configurar a entrega de seus arquivos de CloudTrail log para o CloudWatch Logs. Você faz isso para que o CloudWatch Logs possa monitorá-los em busca de solicitações da API Secrets Manager para recuperar um segredo pendente de exclusão.

Para configurar a entrega do arquivo de CloudTrail log para o CloudWatch Logs

1. Abra o CloudTrail console em https://console.aws.amazon.com/cloudtrail/.
2. Na barra de navegação superior, escolha a opção Região da AWS para monitorar segredos.

3. No painel de navegação esquerdo, escolha Trilhas e, em seguida, escolha o nome da trilha a ser configurada. CloudWatch
4. Na página Configuração de trilhas, role para baixo até a seção CloudWatch Registros e escolha o ícone de edição (✏️).
5. Em New or existing log group, digite um nome para o grupo de log, como **`CloudTrail/MyCloudWatchLogGroup`**.
6. Para a função do IAM, você pode usar a função padrão chamada CloudTrail_ CloudWatchLogs _Role. Essa função tem uma política de função padrão com as permissões necessárias para entregar CloudTrail eventos ao grupo de registros.
7. Escolha Continue para salvar suas configurações.
8. Em AWS CloudTrail Entregará CloudTrail eventos associados à atividade da API em sua conta na página do grupo de CloudWatch registros de registros, escolha Permitir.

Etapa 2: criar o CloudWatch alarme

Para receber uma notificação quando uma operação `GetSecretValue` da API Secrets Manager solicitar o acesso a uma exclusão secreta pendente, você deve criar um CloudWatch alarme e configurar a notificação.

Para criar um CloudWatch alarme

1. Faça login no CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. Na barra de navegação superior, escolha a AWS região em que você deseja monitorar os segredos.
3. No painel de navegação esquerdo, selecione Logs.
4. Na lista de grupos de registros, marque a caixa de seleção ao lado do grupo de registros que você criou no procedimento anterior, como CloudTrail/MyCloudWatchLogGroup. Escolha Create Metric Filter (Criar filtro de métrica).
5. Para Filter Pattern, digite ou cole o seguinte:

   ```
   { $.eventName = "GetSecretValue" && $.errorMessage = "*secret because it was marked
    for deletion*" }
   ```

   Escolha Assign Metric (Atribuir métrica).

6. Na página Create Metric Filter and Assign a Metric (Criar filtro de métrica e atribuir uma métrica), faça o seguinte:

   a. Em Metric Namespace (Namespace da métrica), digite **`CloudTrailLogMetrics`**.
   b. Para Metric Name (Nome da métrica), digite **`AttemptsToAccessDeletedSecrets`**.
   c. Escolha Show advanced metric settings, em seguida, se necessário, para Metric Value, digite **1**.
   d. Selecione Create Filter (Criar filtro).
7. Na caixa de filtro, selecione Create Alarm (Criar alarme).
8. Na janela Create Alarm (Criar alarme), faça o seguinte:

   a. Para Name (Nome), digite **`AttemptsToAccessDeletedSecretsAlarm`**.
   b. Em Whenever: (Sempre:), em is: (é:), escolha **>=** e digite **1**.
   c. Ao lado de Send notification to:, siga um destes procedimentos:

      - Para criar e usar um novo tópico do Amazon SNS, escolha New list (Nova lista) e digite o nome do novo tópico. Para Lista de e-mails:, digite pelo menos um endereço de e-mail. É possível digitar mais de um endereço de e-mail, basta separá-los com vírgulas.
      - Para usar um tópico existente do Amazon SNS, escolha o nome do tópico a ser usado. Se a lista não existir, escolha Select list (Selecionar lista).

   d. Escolha Create Alarm.

Etapa 3: testar o CloudWatch alarme

Para testar seu alarme, crie um segredo e, em seguida, programe sua exclusão. Em seguida, tente recuperar o valor do segredo. Em breve, você receberá um e-mail no endereço configurado no alarme. Ele alerta sobre o uso de um segredo com a exclusão programada.

Monitore AWS Secrets Manager segredos de conformidade usando AWS Config

Você pode usar AWS Config para avaliar seus segredos para ver se eles estão em conformidade com seus padrões. Você define seus requisitos internos de segurança e conformidade para segredos usando AWS Config regras. Em seguida, AWS Config pode identificar segredos que não estão de acordo com suas regras. Você também pode rastrear alterações nos metadados secretos, na

configuração de rotação, na chave KMS usada para criptografia secreta, na função de rotação do Lambda e nas tags associadas a um segredo.

Você pode configurar AWS Config para notificá-lo sobre alterações. Para obter mais informações, consulte Notificações AWS Config enviadas para um tópico do Amazon SNS.

Se você tiver segredos em vários Contas da AWS e Regiões da AWS em sua organização, poderá agregar esses dados de configuração e conformidade. Para obter mais informações, consulte Agregação de dados multirregionais de várias contas.

Para avaliar se os segredos estão em conformidade

- Siga as instruções em Avaliação de seus recursos com AWS Config regras e escolha uma das seguintes regras:
  - `secretsmanager-secret-unused`: verifica se os segredos foram acessados dentro do número de dias especificado.
  - `secretsmanager-using-cmk`— Verifica se os segredos são criptografados usando a chave gerenciada pelo cliente Chave gerenciada pela AWS `aws/secretsmanager` ou uma chave gerenciada pelo cliente que você criou AWS KMS.
  - `secretsmanager-rotation-enabled-check`: verifica se a alternância está configurada para segredos armazenados no Secrets Manager.
  - `secretsmanager-scheduled-rotation-success-check`: verifica se a última alternância bem-sucedida está dentro da frequência de alternância configurada. A frequência mínima para a verificação é diária.
  - `secretsmanager-secret-periodic-rotation`: verifica se os segredos foram alternados dentro do número de dias especificado.

Monitore os custos do Secrets Manager

Você pode usar CloudWatch a Amazon para monitorar as AWS Secrets Manager cobranças estimadas. Para obter mais informações, consulte Criação de um alarme de cobrança para monitorar suas AWS cobranças estimadas no Guia do CloudWatch usuário.

Outra opção para monitorar seus custos é a Detecção de Anomalias de AWS Custos. Para obter mais informações, consulte Detecção de gastos incomuns com a detecção de anomalias de AWS custo no Guia do usuário do gerenciamento de AWS custos.

Para obter informações sobre como monitorar o uso do Secrets Manager, consulte the section called “Monitor com CloudWatch” the section called “Faça login com AWS CloudTrail ” e.

Para obter informações sobre AWS Secrets Manager preços, consultethe section called “Definição de preço”.

Validação de conformidade para AWS Secrets Manager

Sua responsabilidade de conformidade ao usar o Secrets Manager é determinada pela confidencialidade de seus dados, pelos objetivos de conformidade de sua empresa e pelas leis e regulamentos aplicáveis. AWS fornece os seguintes recursos para ajudar na conformidade:

- Guias de início rápido de segurança e compatibilidade: estes guias de implantação abordam as considerações de arquitetura e fornecem etapas para implantação de ambientes de linha de base focados em compatibilidade e segurança na AWS.
- Documento técnico sobre arquitetura para segurança e conformidade com a HIPAA — Este whitepaper descreve como as empresas podem usar para criar aplicativos compatíveis com a HIPAA. AWS
- AWS Recursos de https://aws.amazon.com/compliance/resources/ de conformidade — Essa coleção de pastas de trabalho e guias pode ser aplicada ao seu setor e local.
- O AWS Config avalia até que ponto suas configurações de recursos atendem adequadamente às práticas internas e às diretrizes e regulamentações do setor. Para ter mais informações, consulte the section called “Monitore segredos para verificar a conformidade”.
- AWS Security Hubfornece uma visão abrangente do seu estado de segurança interno AWS que ajuda você a verificar sua conformidade com os padrões e as melhores práticas do setor de segurança. Para obter mais informações sobre como usar o Security Hub para avaliar os recursos do Secrets Manager, consulte Controles do AWS Secrets Manager no Guia do usuário do AWS Security Hub .
- O IAM Access Analyzer analisa políticas, incluindo instruções de condição em uma política, que permitem que uma entidade externa acesse um segredo. Para obter mais informações, consulte Pré-visualização de acesso com Access Analyzer.
- O AWS Systems Manager fornece runbooks predefinidos para o Secrets Manager. Para obter mais informações, consulte Referência do runbook Systems Manager Automation para o Secrets Manager.
- Você pode baixar relatórios de auditoria de terceiros usando AWS Artifact. Para obter mais informações, consulte Baixar relatórios em AWS Artifact .

Padrões de conformidade

AWS Secrets Manager passou por uma auditoria para os seguintes padrões e pode fazer parte de sua solução quando você precisar obter a certificação de conformidade.

- AWS HIPAA — expandiu seu programa de conformidade com a Lei de Portabilidade e Responsabilidade de Seguros de Saúde (HIPAA) para incluí-lo como um serviço qualificado pela HIPAA. AWS Secrets Manager Se você tiver um Acordo de Associado Comercial (BAA) assinado com AWS, você pode usar o Secrets Manager para ajudar a criar seus aplicativos compatíveis com HIPAA. AWS oferece um whitepaper com foco na HIPAA para clientes interessados em saber mais sobre como eles podem aproveitar o processamento e o armazenamento AWS de informações de saúde. Para obter mais informações, consulte Conformidade com a HIPAA.
- Organização participante do PIC — AWS Secrets Manager tem um Atestado de Conformidade para o Padrão de Segurança de Dados (DSS) do Setor de Cartões de Pagamento (PCI), versão 3.2, no nível 1 do provedor de serviços. Os clientes que usam AWS produtos e serviços para armazenar, processar ou transmitir dados do titular do cartão podem usá-los AWS Secrets Manager enquanto gerenciam sua própria certificação de conformidade com o PCI DSS. Para obter mais informações sobre o PCI DSS, incluindo como solicitar uma cópia do PCI AWS Compliance Package, consulte PCI DSS Nível 1.
- ISO — AWS Secrets Manager concluiu com sucesso a certificação de conformidade para ISO/IEC 27001, ISO/IEC 27017, ISO/IEC 27018 e ISO 9001. Para obter mais informações, consulte ISO 27001, ISO 27017, ISO 27018 e ISO 9001.
- Os relatórios AICPA SOC — System and Organization Control (SOC) são relatórios de exame independentes de terceiros que demonstram como o Secrets Manager alcança os principais objetivos e controles de conformidade. O objetivo desses relatórios é ajudar você e seus auditores a entender os AWS controles estabelecidos para apoiar as operações e a conformidade. Para obter mais informações, consulte Conformidade com o SOC.
- FedRAMP — O Programa Federal de Gerenciamento de Riscos e Autorizações (FedRAMP) é um programa governamental que fornece uma abordagem padronizada para avaliação de segurança, autorização e monitoramento contínuo de produtos e serviços em nuvem. O Programa FedRAMP também fornece autorizações provisórias para serviços e regiões do Leste/Oeste GovCloud e para consumir dados governamentais ou regulamentados. Para obter mais informações, consulte Conformidade com o FedRAMP.
- Departamento de Defesa — O Guia de Requisitos de Segurança de Computação em Nuvem (SRG) do Departamento de Defesa (DoD) fornece um processo padronizado de avaliação e autorização para que os provedores de serviços em nuvem (CSPs) obtenham uma autorização

provisória do DoD, para que possam atender aos clientes do DoD. Para obter mais informações, consulte Recursos de SRG do DoD

- IRAP — O Programa de Avaliadores Registrados de Segurança da Informação (IRAP) permite que os clientes do governo australiano validem se os controles apropriados estão em vigor e determinem o modelo de responsabilidade apropriado para atender aos requisitos do Manual de Segurança da Informação (ISM) do governo australiano produzido pelo Centro Australiano de Segurança Cibernética (ACSC). Para obter mais informações, consulte Recursos do IRAP.
- OSPAR — A Amazon Web Services (AWS) obteve o certificado Outsourced Service Provider's Audit Report (OSPAR). AWS o alinhamento com as Diretrizes da Associação de Bancos de Cingapura (ABS) sobre objetivos e procedimentos de controle para provedores de serviços terceirizados (Diretrizes ABS) demonstra aos clientes o AWS compromisso dos clientes em atender às altas expectativas dos provedores de serviços em nuvem estabelecidas pelo setor de serviços financeiros em Cingapura. Para obter mais informações, consulte Recursos do OSPAR.

Segurança em AWS Secrets Manager

A segurança na AWS é a nossa maior prioridade. Como cliente da AWS, você se beneficia de um datacenter e uma arquitetura de rede criados para atender os requisitos da maioria das organizações com exigências de segurança.

Você e a AWS compartilham a responsabilidade pela segurança. O modelo de responsabilidade compartilhada descreve a segurança da nuvem e a segurança na nuvem:

- Segurança da nuvem: a AWS é responsável pela proteção da infraestrutura que executa serviços da AWS na Nuvem AWS. A AWS também fornece serviços que podem ser usados com segurança. Auditores de terceiros testam e verificam regularmente a eficácia da nossa segurança como parte dos Programas de conformidade da AWS. Para saber mais sobre os programas de conformidade que se aplicam ao AWS Secrets Manager, consulte Serviços da AWS no escopo pelo programa de conformidade.
- Segurança na nuvem – seu serviço da AWS determina sua responsabilidade. Você também é responsável por outros fatores, incluindo a confidencialidade de seus dados, os requisitos da sua empresa e as leis e regulamentos aplicáveis.

Para obter mais recursos, consulte Pilar de segurança - AWS Well-Architected Framework.

Tópicos

Mitigação de riscos do uso da AWS CLI para armazenar segredos do AWS Secrets Manager

Quando você usa a AWS Command Line Interface (AWS CLI) para chamar operações da AWS, você insere esses comandos em um shell de comandos. Por exemplo, você pode usar o prompt

de comando do Windows ou o Windows PowerShell, Bash ou Z shell, entre outros. Muitos desses shells de comando incluem funcionalidade desenvolvida para aumentar a produtividade. Mas essa funcionalidade pode ser usada para comprometer seus segredos. Por exemplo, na maioria dos shells, você pode usar a tecla de seta para cima para ver o último comando inserido. O recurso histórico de comandos pode ser explorado por qualquer pessoa que acesse sua sessão não segura. Além disso, outros utilitários que funcionam em segundo plano podem ter acesso aos parâmetros de seus comandos, com o objetivo de ajudar você a realizar as tarefas com mais eficiência. Para mitigar esses riscos, siga as seguintes etapas:

- Sempre bloqueie seu computador ao sair do console.
- Desinstale ou desabilite os utilitários do console que você não precisa ou não usa mais.
- Verifique se o shell ou o programa de acesso remoto, caso esteja usando um, não registra em log os comandos digitados.
- Use técnicas para passar parâmetros não capturados pelo histórico de comandos do shell. O exemplo a seguir mostra como você pode digitar o texto do segredo em um arquivo de texto, passar o arquivo para o comando do AWS Secrets Manager e destruir o arquivo imediatamente. Isso significa que o histórico típico do shell não captura o texto do segredo.

  O exemplo a seguir mostra os comandos típicos do Linux, mas seu shell pode exigir comandos um pouco diferentes:

```
$ 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.
```

Depois de executar esses comandos, você poderá usar as setas para cima e para baixo para percorrer o histórico de comandos e ver se o texto do segredo está sendo exibido em alguma linha.

> **Important**
>
> Por padrão, não é possível executar uma técnica equivalente no Windows, a menos que você reduza primeiro o tamanho do buffer do histórico de comandos para 1.

Para configurar o prompt de comando do Windows para ter apenas 1 buffer de histórico de comando de 1 comando

1. Abra um prompt de comando como administrador (Executar como administrador).
2. Escolha o ícone no canto superior esquerdo e escolha Properties (Propriedades).
3. Na guia Options, defina Buffer Size e Number of Buffers como **1**, e escolha OK.
4. Sempre que precisar digitar um comando que você não deseja armazenar no histórico, insira outro comando imediatamente depois dele, como:

   ```
   echo.
   ```

   Isso garante que você descarrega o comando sensível.

No shell do Prompt de comando do Windows, você pode fazer download da ferramenta SysInternals SDelete e usar comandos semelhantes aos seguintes:

```
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.
```

Proteção de dados no AWS Secrets Manager

O modelo de responsabilidade compartilhada da AWS se aplica à proteção de dados no AWS Secrets Manager. Conforme descrito nesse modelo, a AWS é responsável por proteger a infraestrutura global que executa toda a Nuvem AWS. Você é responsável por manter o controle sobre seu conteúdo hospedado nessa infraestrutura. Esse conteúdo inclui as tarefas de configuração e gerenciamento de segurança dos Serviços da AWS que você usa. Para ter mais informações sobre a privacidade de dados, consulte as Perguntas frequentes sobre privacidade de dados. Para obter mais informações sobre a proteção de dados na Europa, consulte a postagem do blog AWS Shared Responsibility Model and GDPR no Blog de segurança da AWS.

Para fins de proteção de dados, recomendamos que você proteja as credenciais da conta da Conta da AWS e configure as contas de usuário individuais com o AWS Identity and Access Management (IAM). Dessa maneira, cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com os recursos da AWS. O Secrets Manager é compatível com o TLS 1.2 e 1.3 em todas as regiões. O Secrets Manager também é compatível com um protocolo híbrido de criptografia de rede com opção de troca de chave pós-quântica para TLS (PQTLS).
- Assine suas solicitações programáticas para o Secrets Manager usando um ID da chave de acesso e uma chave de acesso secreta associados a uma entidade principal do IAM. Você também pode usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.
- Configure o registro em log das atividades da API e do usuário com o AWS CloudTrail. Consulte the section called “Faça login com AWS CloudTrail ”.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-2 ao acessar a AWS por meio de uma interface de linha de comando ou uma API, use um endpoint do FIPS. Consulte the section called “Endpoint do Secrets Manager”.
- Se você usar a AWS CLI para acessar o Secrets Manager, the section called “Mitigação de riscos do uso da AWS CLI para armazenar segredos do AWS Secrets Manager”.

Criptografia em repouso

O Secrets Manager usa criptografia por meio do AWS Key Management Service (AWS KMS) para proteger a confidencialidade dos dados em repouso. O AWS KMS fornece um serviço de armazenamento e criptografia de chaves usado por muitos serviços da AWS. Cada segredo no Secrets Manager é criptografado com uma chave de dados exclusiva. Cada chave de dados é protegida por uma chave KMS. Você pode optar por usar a criptografia padrão com a Chave gerenciada pela AWS do Secrets Manager para a conta ou pode criar sua própria chave gerenciada pelo cliente no AWS KMS. O uso de uma chave gerenciada pelo cliente oferece controles de autorização mais granulares sobre as atividades da chave KMS. Para obter mais informações, consulte the section called “Criptografia e descriptografia de segredos”.

Criptografia em trânsito

O Secrets Manager fornece endpoints seguros e privados para criptografar dados em trânsito. Os endpoints seguros e privados permitem que a AWS proteja a integridade das solicitações de API ao Secrets Manager. A AWS exige que as chamadas de API sejam assinadas pela entidade responsável usando certificados X.509 ou uma chave de acesso secreta do Secrets Manager. Esse requisito é indicado no Processo de assinatura do Signature versão 4 (Sigv4).

Se você usar a AWS Command Line Interface (AWS CLI) ou qualquer um dos SDKs da AWS para fazer chamadas para a AWS, configure a chave de acesso a ser usada. Em seguida, essas ferramentas usam automaticamente a chave de acesso para assinar as solicitações para você. Consulte the section called “Mitigação de riscos do uso da AWS CLI para armazenar segredos do AWS Secrets Manager”.

Privacidade do tráfego entre redes

A AWS oferece opções para manter a privacidade ao rotear tráfego por meio de rotas de rede conhecidas e privadas.

Tráfego entre clientes de serviço e on-premises e as aplicações

Você tem duas opções de conectividade entre sua rede privada e a AWS Secrets Manager:

- Uma conexão VPN de local a local da AWS. Para obter mais informações, consulte O que é o AWS Site-to-Site VPN?
- Uma conexão AWS Direct Connect. Para obter mais informações, consulte O que é o AWS Direct Connect?

Tráfego entre recursos da AWS na mesma região

Para proteger o tráfego entre o Secrets Manager e os clientes de API na AWS, configure um PrivateLink da AWS para acessar com privacidade os endpoints de API do Secrets Manager.

Gerenciamento de chave de criptografia

Quando o Secrets Manager precisa criptografar uma nova versão dos dados protegidos do segredo, o Secrets Manager envia uma solicitação ao AWS KMS para gerar uma nova chave de dados da chave do KMS. O Secrets Manager usa essa chave de dados para criptografia de envelope. O Secrets Manager armazena a chave de dados criptografada com o segredo criptografado. Quando o segredo precisa ser descriptografado, o Secrets Manager pede ao AWS KMS para descriptografar a chave de dados. Em seguida, o Secrets Manager usa a chave de dados descriptografada para descriptografar o segredo criptografado. O Secrets Manager nunca armazena a chave de dados em formato não criptografado e remove a chave da memória o mais rápido possível. Para obter mais informações, consulte the section called “Criptografia e descriptografia de segredos”.

Criptografia e decodificação secretas em AWS Secrets Manager

O Secrets Manager usa criptografia de envelope com AWS KMS chaves e chaves de dados para proteger cada valor secreto. Sempre que o valor secreto em um segredo muda, o Secrets Manager solicita uma nova chave de dados AWS KMS para protegê-lo. A chave de dados é criptografada em uma chave do KMS e é armazenada nos metadados do segredo. Para descriptografar o segredo, o Secrets Manager primeiro descriptografa a chave de dados criptografada usando a chave KMS em. AWS KMS

O Secrets Manager não usa a chave do KMS para criptografar o valor do segredo diretamente. Em vez disso, ele usa a chave do KMS para gerar e criptografar uma chave de dados simétrica de Padrão de criptografia avançada (AES) de 256 bits e usa a chave de dados para criptografar o valor do segredo. O Secrets Manager usa a chave de dados de texto simples para criptografar o valor secreto externo e AWS KMS, em seguida, o remove da memória. Ele armazena a cópia criptografada da chave de dados nos metadados do segredo.

Tópicos

- Escolhendo uma AWS KMS chave
- O que é criptografado?
- Processos de criptografia e descriptografia

- Permissões para a chave do KMS
- Como o Secrets Manager usa a chave do KMS
- Política de chaves da Chave gerenciada pela AWS (aws/secretsmanager)
- Contexto de criptografia do Secrets Manager
- Monitore a interação do Secrets Manager com AWS KMS

Escolhendo uma AWS KMS chave

Ao criar um segredo, você pode escolher qualquer chave de criptografia simétrica gerenciada pelo cliente na região Conta da AWS e, ou você pode usar o Chave gerenciada pela AWS for Secrets Manager (`aws/secretsmanager`). Se você escolher o Chave gerenciada pela AWS `aws/secretsmanager` e ele ainda não existe, o Secrets Manager o cria e o associa ao segredo. É possível usar a mesma chave do KMS ou diferentes chaves do KMS para cada segredo na sua conta. Talvez você queira usar chaves KMS diferentes para definir permissões personalizadas nas chaves para um grupo de segredos ou se quiser auditar operações específicas para essas chaves. O Secrets Manager só oferece suporte a chaves de criptografia simétricas do KMS. Se você usar uma chave KMS em um armazenamento de chaves externas, as operações criptográficas na chave KMS podem levar mais tempo e ter menos confiabilidade e durabilidade, pois a solicitação precisa sair da AWS.

Para obter informações sobre alterar a chave de criptografia de um segredo, consulte the section called “Altere a chave de criptografia de um segredo”.

Quando você altera a chave de criptografia, o Secrets Manager `AWSCURRENT` criptografa novamente e `AWSPENDING` cria `AWSPREVIOUS` versões com a nova chave. Para evitar bloquear você do segredo, o Secrets Manager mantém todas as versões existentes criptografadas com a chave anterior. Isso significa que você pode descriptografar `AWSCURRENTAWSPENDING`, e `AWSPREVIOUS` versões com a chave anterior ou a nova chave.

Para que isso só `AWSCURRENT` possa ser descriptografado pela nova chave de criptografia, crie uma nova versão do segredo com a nova chave. Então, para poder decifrar a versão `AWSCURRENT` secreta, você deve ter permissão para a nova chave.

Você pode negar a permissão Chave gerenciada pela AWS `aws/secretsmanager` e exigir que os segredos sejam criptografados com uma chave gerenciada pelo cliente. Para ter mais informações, consulte the section called “Exemplo: negar uma AWS KMS chave específica para criptografar segredos”.

Para encontrar a chave KMS associada a um segredo, visualize o segredo no console ou ligue para ListSecretsou DescribeSecret. Quando o segredo é associado ao Chave gerenciada pela AWS for Secrets Manager (`aws/secretsmanager`), essas operações não retornam um identificador de chave KMS.

O que é criptografado?

O Secrets Manager criptografa o valor do segredo, mas não criptografa o seguinte:

- Nome e descrição do segredo
- Configurações de alternância
- ARN da chave do KMS associada ao segredo
- Qualquer AWS etiqueta anexada

Processos de criptografia e descriptografia

Para criptografar o valor de um segredo, o Secrets Manager usa o processo descrito a seguir.

1. O Secrets Manager chama a AWS KMS GenerateDataKeyoperação com o ID da chave KMS do segredo e uma solicitação de uma chave simétrica AES de 256 bits. AWS KMS retorna uma chave de dados em texto simples e uma cópia dessa chave de dados criptografada sob a chave KMS.
2. O Secrets Manager usa a chave de dados de texto simples e o algoritmo Advanced Encryption Standard (AES) para criptografar o valor secreto externo. AWS KMS Ele remove a chave de texto simples da memória o mais rápido possível após o uso.
3. O Secrets Manager armazena a chave de dados criptografada nos metadados do segredo para que fique disponível para descriptografar o valor do segredo. No entanto, nenhuma das APIs do Secrets Manager retorna o segredo criptografado ou a chave de dados criptografada.

Para descriptografar um valor do segredo criptografado:

1. O Secrets Manager chama a operação AWS KMS Decrypt e passa a chave de dados criptografada.
2. AWS KMS usa a chave KMS do segredo para descriptografar a chave de dados. Ele gera a chave de dados de texto simples.
3. O Secrets Manager usa a chave de dados de texto simples para descriptografar o valor do segredo. Então, ele remove a chave de dados da memória o mais rápido possível.

Permissões para a chave do KMS

Quando o Secrets Manager usa uma chave do KMS em operações de criptografia, ele atua em nome do usuário que está acessando ou atualizando o valor do segredo. É possível conceder permissões em uma política do IAM ou em uma política de chave. As seguintes operações do Secrets Manager exigem AWS KMS permissões.

- CreateSecret
- GetSecretValue
- PutSecretValue
- UpdateSecret
- ReplicateSecretToRegions

Para permitir que a chave KMS seja usada somente para solicitações originadas no Secrets Manager, na política de permissões, você pode usar a chave kms: ViaService condition com o valor. `secretsmanager.<Region>.amazonaws.com`

Também é possível usar as chaves ou valores no contexto de criptografia como condição para usar a chave do KMS em operações de criptografia. Por exemplo, você pode usar um operador de condição de string em um IAM ou documento de política de chaves ou usar uma restrição de concessão em uma concessão. A propagação da concessão da chave do KMS pode demorar até cinco minutos. Para obter mais informações, consulte CreateGrant.

Como o Secrets Manager usa a chave do KMS

O Secrets Manager chama as seguintes AWS KMS operações com sua chave KMS.

GenerateDataKey

O Secrets Manager chama a AWS KMS GenerateDataKeyoperação em resposta às seguintes operações do Secrets Manager.

- CreateSecret— Se o novo segredo incluir um valor secreto, o Secrets Manager solicitará uma nova chave de dados para criptografá-lo.
- PutSecretValue— O Secrets Manager solicita uma nova chave de dados para criptografar o valor secreto especificado.
- ReplicateSecretToRegions— Para criptografar o segredo replicado, o Secrets Manager solicita uma chave de dados para a chave KMS na região da réplica.

- UpdateSecret— Se você alterar o valor secreto ou a chave KMS, o Secrets Manager solicitará uma nova chave de dados para criptografar o novo valor secreto.

A RotateSecretoperação não chama `GenerateDataKey` porque não altera o valor secreto. No entanto, se o `RotateSecret` invoca a função Lambda que muda o valor do segredo, sua chamada para a operação `PutSecretValue` acionará uma solicitação `GenerateDataKey`.

Decrypt

O Secrets Manager chama a operação Decrypt em resposta às seguintes operações do Secrets Manager.

- GetSecretValuee BatchGetSecretValue— Secrets Manager decifra o valor secreto antes de devolvê-lo ao chamador. Para descriptografar um valor secreto criptografado, o Secrets Manager chama a operação AWS KMS Decrypt para descriptografar a chave de dados criptografada no segredo. Ele usa a chave de dados de texto simples para descriptografar o valor do segredo criptografado. Para comandos em lote, o Secrets Manager pode reutilizar a chave descriptografada; portanto, nem todas as chamadas resultam em uma solicitação `Decrypt`.
- PutSecretValuee UpdateSecret— A maioria das `UpdateSecret` solicitações de `PutSecretValue` e não aciona uma `Decrypt` operação. No entanto, quando uma solicitação `PutSecretValue` ou `UpdateSecret` tenta alterar o valor do segredo em uma versão existente de um segredo, o Secrets Manager descriptografa o valor do segredo existente e o compara com o valor do segredo da solicitação para confirmar se são iguais. Essa ação garante que as operações do Secrets Manager sejam idempotentes. Para descriptografar um valor secreto criptografado, o Secrets Manager chama a operação AWS KMS Decrypt para descriptografar a chave de dados criptografada no segredo. Ele usa a chave de dados de texto simples para descriptografar o valor do segredo criptografado.
- ReplicateSecretToRegions— O Secrets Manager primeiro descriptografa o valor secreto na região primária antes de criptografar novamente o valor secreto com a chave KMS na região de réplica.

Encrypt

O Secrets Manager chama a operação Encrypt em resposta às seguintes operações do Secrets Manager:

- UpdateSecret— Se você alterar a chave KMS, o Secrets Manager criptografará novamente a chave de dados que protege as versões `AWSCURRENTAWSPREVIOUS`, e `AWSPENDING` secretas com a nova chave.

- ReplicateSecretToRegions— O Secrets Manager criptografa novamente a chave de dados durante a replicação usando a chave KMS na região da réplica.

DescribeKey

O Secrets Manager chama a DescribeKeyoperação para determinar se a chave KMS deve ser listada ao criar ou editar um segredo no console do Secrets Manager.

Validar o acesso à chave do KMS

Ao estabelecer ou alterar a chave do KMS associada ao segredo, o Secrets Manager chama as operações `GenerateDataKey` e `Decrypt` com a chave do KMS especificada. Essas chamadas confirmam que o autor da chamada tem permissão para usar a chave do KMS para essas operações. O Secrets Manager descarta os resultados dessas operações; ele não os usa em qualquer operação de criptografia.

Você pode identificar essas chamadas de validação, pois o valor da chave `SecretVersionId` com contexto de criptografia nessas solicitações é `RequestToValidateKeyAccess`.

> **Note**
>
> No passado, as chamadas de validação do Secrets Manager não incluíam um contexto de criptografia. Você pode encontrar chamadas sem contexto de criptografia em AWS CloudTrail registros mais antigos.

Política de chaves da Chave gerenciada pela AWS (**`aws/secretsmanager`**)

A política de chaves do Chave gerenciada pela AWS for Secrets Manager (`aws/secretsmanager`) dá aos usuários permissão para usar a chave KMS para operações específicas somente quando o Secrets Manager faz a solicitação em nome do usuário. A política de chaves não permite que os usuários utilizem a chave do KMS diretamente.

Essa política de chaves, como as políticas de todas as Chaves gerenciadas pela AWS, é estabelecida pelo serviço. Não é possível alterar a política de chaves, mas é possível visualizá-la a qualquer momento. Para obter mais detalhes, consulte Visualizar uma política de chaves.

As declarações de política na política de chaves têm os seguintes efeitos:

- Permita que os usuários da conta usem a chave do KMS para operações de criptografia somente quando a solicitação for proveniente do Secrets Manager em seu nome. A chave de condição `kms:ViaService` impõe essa restrição.
- Permite que a AWS conta crie políticas do IAM que permitem aos usuários visualizar as propriedades da chave KMS e revogar concessões.
- Embora o Secrets Manager não use concessões para obter acesso à chave do KMS, a política também permite que o Secrets Manager crie concessões para a chave do KMS em nome do usuário e permite que a conta revogue qualquer concessão que permite que o Secrets Manager use a chave do KMS. Esses são elementos padrão do documento de política de uma Chave gerenciada pela AWS.

A seguir está uma política fundamental para um Chave gerenciada pela AWS exemplo de Secrets Manager.

```
{
  "Id": "auto-secretsmanager-2",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow access through AWS Secrets Manager for all principals in the
 account that are authorized to use AWS Secrets Manager",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "*"
        ]
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:CreateGrant",
        "kms:DescribeKey"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "111122223333",
          "kms:ViaService": "secretsmanager.us-west-2.amazonaws.com"
        }
```

```
      }
    },
    {
      "Sid": "Allow access through AWS Secrets Manager for all principals in the
 account that are authorized to use AWS Secrets Manager",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "*"
        ]
      },
      "Action": "kms:GenerateDataKey*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "111122223333"
        },
        "StringLike": {
          "kms:ViaService": "secretsmanager.us-west-2.amazonaws.com"
        }
      }
    },
    {
      "Sid": "Allow direct access to key metadata to the account",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:root"
        ]
      },
      "Action": [
        "kms:Describe*",
        "kms:Get*",
        "kms:List*",
        "kms:RevokeGrant"
      ],
      "Resource": "*"
    }
  ]
}
```

Contexto de criptografia do Secrets Manager

Um contexto de criptografia é um conjunto de pares de chave-valor que contêm dados arbitrários não secretos. Quando você inclui um contexto de criptografia em uma solicitação para criptografar dados, vincula AWS KMS criptograficamente o contexto de criptografia aos dados criptografados. Para descriptografar os dados, você deve passar o mesmo contexto de criptografia.

Em suas solicitações GenerateDataKeye Decrypt para, o Secrets AWS KMS Manager usa um contexto de criptografia com dois pares de nome e valor que identificam o segredo e sua versão, conforme mostrado no exemplo a seguir. Os nomes não variam, mas os valores de contexto de criptografia combinados serão diferentes para cada valor de segredo.

```
"encryptionContext": {
    "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-
a1b2c3",
    "SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
}
```

Você pode usar o contexto de criptografia para identificar essas operações criptográficas em registros e registros de auditoria, como AWS CloudTrailAmazon CloudWatch Logs, e como condição para autorização em políticas e concessões.

O contexto de criptografia do Secrets Manager consiste em dois pares de nome e valor.

- SecretARN: o primeiro par de nome e valor identifica o segredo. A chave é `SecretARN`. O valor é o nome de recurso da Amazon (ARN) do segredo.

  ```
  "SecretARN": "ARN of an Secrets Manager secret"
  ```

  Por exemplo, se o ARN do segredo fosse `arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-a1b2c3`, o contexto de criptografia incluiria o seguinte par.

  ```
  "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-
  a1b2c3"
  ```

- SecretVersionId— O segundo par nome-valor identifica a versão do segredo. A chave é `SecretVersionId`. O valor é o ID da versão.

```
"SecretVersionId": "<version-id>"
```

Por exemplo, se o ID de versão do segredo fosse `EXAMPLE1-90ab-cdef-fedc-ba987SECRET1`, o contexto de criptografia incluiria o seguinte par.

```
"SecretVersionId": "EXAMPLE1-90ab-cdef-fedc-ba987SECRET1"
```

Quando você estabelece ou altera a chave KMS de um segredo, o Secrets Manager envia GenerateDataKeye descriptografa solicitações para AWS KMS validar se o chamador tem permissão para usar a chave KMS para essas operações. Ele descarta as respostas; não as usa no valor do segredo.

Nessas solicitações de validação, o valor do `SecretARN` é o ARN real do segredo, mas o valor `SecretVersionId` é `RequestToValidateKeyAccess`, como mostrado no seguinte exemplo de contexto de criptografia. Esse valor especial ajuda você a identificar solicitações de validação em logs e trilhas de auditoria.

```
"encryptionContext": {
    "SecretARN": "arn:aws:secretsmanager:us-east-2:111122223333:secret:test-secret-
a1b2c3",
    "SecretVersionId": "RequestToValidateKeyAccess"
}
```

> **Note**
>
> No passado, as solicitações de validação do Secrets Manager não incluíam um contexto de criptografia. Você pode encontrar chamadas sem contexto de criptografia em AWS CloudTrail registros mais antigos.

Monitore a interação do Secrets Manager com AWS KMS

Você pode usar o AWS CloudTrail Amazon CloudWatch Logs para rastrear as solicitações que o Secrets Manager envia AWS KMS em seu nome. Para obter mais informações sobre o monitoramento do uso de segredos, consulte Monitorar segredos.

GenerateDataKey

Quando você cria ou altera o valor secreto em um segredo, o Secrets Manager envia uma GenerateDataKeysolicitação AWS KMS que especifica a chave KMS do segredo.

O evento que registra a operação `GenerateDataKey` é semelhante ao evento de exemplo a seguir. A solicitação foi invocada por `secretsmanager.amazonaws.com`. Os parâmetros incluem o nome do recurso da Amazon (ARN) da chave do KMS do segredo, um especificador de chaves que requer uma chave de 256 bits e o contexto de criptografia que identifica o segredo e a versão.

```
{
    "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 você obtém ou altera o valor secreto de um segredo, o Secrets Manager envia uma solicitação de descriptografia para descriptografar AWS KMS a chave de dados criptografada. Para comandos em lote, o Secrets Manager pode reutilizar a chave descriptografada; portanto, nem todas as chamadas resultam em uma solicitação `Decrypt`.

O evento que registra a operação `Decrypt` é semelhante ao evento de exemplo a seguir. O usuário é o principal da sua AWS conta que está acessando a tabela. Os parâmetros incluem a chave da tabela criptografada (como um blob de texto cifrado) e o contexto de criptografia que identifica a tabela e a conta. AWS AWS KMS deriva o ID da chave KMS do texto cifrado.

```
{
    "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"
}
```

Encrypt

Quando você altera a chave KMS associada a um segredo, o Secrets Manager envia uma solicitação Encrypt para AWS KMS recriptografar as versões`AWSCURRENT`,`AWSPREVIOUS`, e `AWSPENDING` secretas com a nova chave. Quando você replica um segredo para outra região, o Secrets Manager também envia uma solicitação Encrypt para o AWS KMS.

O evento que registra a operação `Encrypt` é semelhante ao evento de exemplo a seguir. O usuário é o principal da sua AWS conta que está acessando a tabela.

```
{
    "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"
}
```

Segurança da infraestrutura no AWS Secrets Manager

Por ser um serviço gerenciado, o AWS Secrets Manager é protegido pela segurança da rede global da AWS. Para obter informações sobre serviços de segurança da AWS e como a AWS protege a infraestrutura, consulte Segurança na Nuvem AWS. Para projetar seu ambiente da AWS usando as práticas recomendadas de segurança de infraestrutura, consulte Proteção de infraestrutura em Pilar segurança: AWS Well-Architected Framework.

O acesso ao Secrets Manager usando a rede é feito por meio de APIs publicadas pela AWS usando o TLS. As APIs do Secrets Manager podem ser chamadas de qualquer local de rede. No entanto, o Secrets Manager é compatível com políticas de acesso com base em recursos, que podem incluir restrições com base no endereço IP de origem. Você também pode usar políticas de recursos do Secrets Manager para controlar o acesso a segredos de endpoints específicos da nuvem privada virtual (VPC) ou de VPCs específicas. Efetivamente, isso isola o acesso à rede para um determinado segredo apenas da VPC específica dentro da rede da AWS. Para obter mais informações, consulte Endpoint da VPC.

Resiliência em AWS Secrets Manager

AWS constrói a infraestrutura global em torno de zonas Regiões da AWS de disponibilidade. Regiões da AWS fornecem várias zonas de disponibilidade fisicamente separadas e isoladas, que se conectam a redes de baixa latência, alta taxa de transferência e altamente redundantes. Com as zonas de disponibilidade, é possível projetar e operar aplicativos e bancos de dados que automaticamente executam o failover entre as zonas sem interrupção. As zonas de disponibilidade permitem que você seja mais altamente disponível, tolerante a falhas e escalável do que uma ou várias infraestruturas de datacenter tradicionais.

Para obter mais informações sobre resiliência e recuperação de desastres, consulte Reliability Pillar - Well-Architected AWS Framework.

Para obter mais informações sobre zonas de disponibilidade Regiões da AWS e zonas de disponibilidade, consulte Infraestrutura AWS global.

TLS pós-quântico

O Secrets Manager é compatível com uma opção de troca pós-quântica híbrida de chaves para o protocolo Transport Layer Security (TLS) de criptografia de rede. Você pode usar essa opção de TLS ao se conectar a endpoints de API do Secrets Manager. Estamos oferecendo esse recurso antes da padronização dos algoritmos pós-quânticos, permitindo que você comece a testar o efeito desses protocolos de troca de chaves em chamadas do Secrets Manager. Esses recursos opcionais de troca de chaves pós-quânticas híbridas são pelo menos tão seguros quanto a criptografia TLS que usamos atualmente e provavelmente fornecerão benefícios adicionais de segurança. No entanto, eles afetam a latência e o throughput em comparação com os protocolos clássicos de troca de chaves em uso atualmente.

Para proteger dados criptografados atualmente contra possíveis ataques futuros, a AWS está participando junto com a comunidade criptográfica no desenvolvimento de algoritmos quânticos ou pós-quânticos. Implementamos pacotes de criptografia de troca pós-quântica híbrida de chaves em endpoints do Secrets Manager. Esses conjuntos de cifras híbridas, que combinam elementos clássicos e pós-quânticos, garantem que sua conexão TLS seja pelo menos tão forte quanto seria com pacotes de criptografia clássica. No entanto, como as características de desempenho e os requisitos de largura de banda dos pacotes de criptografia híbrida são diferentes dos mecanismos clássicos de troca de chaves, recomendamos testá-los em suas chamadas de API.

O Secrets Manager é compatível com o PQTLS em todas as regiões, exceto nas da China.

Configurar o TLS pós-quântico híbrido

1. Adicione o cliente de runtime comum da AWS às suas dependências do Maven. Recomendamos usar a versão mais recente disponível. Por exemplo, esta instrução adiciona a versão 2.20.0.

```
<dependency>
  <groupId>software.amazon.awssdk</groupId>
  <artifactId>aws-crt-client</artifactId>
  <version>2.20.0</version>
</dependency>
```

2. Adicione o AWS SDK para Java 2.x ao seu projeto e inicialize-o. Habilite os pacotes de cifra pós-quântica híbrida em seu cliente HTTP.

```
SdkAsyncHttpClient awsCrtHttpClient = AwsCrtAsyncHttpClient.builder()
```

```
            .postQuantumTlsEnabled(true)
            .build();
```

3. Crie o cliente assíncrono do Secrets Manager.

```
SecretsManagerAsyncClient SecretsManagerAsync = SecretsManagerAsyncClient.builder()
            .httpClient(awsCrtHttpClient)
            .build();
```

   Agora quando você chama operações de API do Secrets Manager, suas chamadas são transmitidas ao endpoint do Secrets Manager ao usar o TLS pós-quântico híbrido.

Para obter mais informações sobre o uso de TLS pós-quântico híbrido, consulte:

- Guia do desenvolvedor do AWS SDK for Java 2.x e a publicação de blog released (Lançamento do AWS SDK for Java 2.x).
- Apresentação do s2n-tls, uma nova implementação de código aberto do TLS e Uso do s2n-tls.
- Post-Quantum Cryptography (Criptografia pós-quântica) no National Institute for Standards and Technology (NIST).
- Hybrid Post-Quantum Key Encapsulation Methods (PQ KEM – Métodos pós-quânticos híbridos de encapsulamento de chave) para Transport Layer Security 1.2 (TLS).

O TLS pós-quântico para o Secrets Manager está disponível em todas as regiões da Regiões da AWS, exceto nas da China.

Solução de problemas AWS Secrets Manager

Use estas informações para ajudá-lo a diagnosticar e corrigir problemas que você venha a encontrar ao trabalhar com o Secrets Manager.

Para problemas relacionados à alternância, consulte the section called “Solução de problemas de alternância do ”.

Tópicos

Mensagens de “Acesso negado”

Ao fazer uma chamada de API, como GetSecretValue ou CreateSecret para o Secrets Manager, você precisa ter permissões do IAM para fazer essa chamada. Quando você usa o console, ele faz as mesmas chamadas de API em seu nome, então você também precisa ter permissões do IAM. Um administrador pode conceder permissões anexando uma política do IAM ao seu usuário do IAM ou a um grupo do qual você é membro. Se as declarações de política que concedem essas permissões incluírem quaisquer condições, como time-of-day restrições de endereço IP, você também deverá atender a esses requisitos ao enviar a solicitação. Para obter informações sobre como visualizar ou modificar políticas para um usuário, grupo ou função do IAM, consulte Trabalho com políticas no Manual do usuário do IAM. Para obter mais informações sobre as permissões necessárias para o Secrets Manager, consulte Autenticação e controle de acesso.

Se você assinar solicitações de API manualmente, sem usar os SDKs da AWS, verifique se assinou a solicitação corretamente.

"Access denied" (Acesso negado) para credenciais de segurança temporárias

Verifique se o usuário ou a função do IAM que você está usando para fazer a solicitação tem as permissões corretas. As permissões de credenciais de segurança temporárias são originárias de um usuário ou função do IAM. Isso significa que as permissões são limitadas às concedidas ao usuário ou função do IAM. Para obter mais informações sobre como as permissões de credenciais de segurança temporárias são determinadas, consulte Controlar permissões para credenciais de segurança temporárias no Manual do usuário do IAM.

Verifique se suas solicitações são assinadas corretamente e bem-formada. Para obter detalhes, consulte a documentação do kit de ferramentas do SDK escolhido ou Como usar credenciais de segurança temporárias para solicitar acesso aos AWS recursos no Guia do usuário do IAM.

Verifique se suas credenciais de segurança temporárias não expiraram. Para obter mais informações, consulte Solicitação de credenciais de segurança temporárias no Manual do usuário do IAM.

Para obter mais informações sobre as permissões necessárias para o Secrets Manager, consulte Autenticação e controle de acesso.

As alterações que faço nem sempre ficam imediatamente visíveis.

O Secret Manager usa um modelo de computação distribuído chamado consistência eventual. Qualquer alteração que você fizer no Secrets Manager (ou em outros AWS serviços) leva tempo para se tornar visível em todos os endpoints possíveis. Parte do atraso resulta do tempo necessário para enviar os dados de um servidor para outro, de uma zona de replicação para outra e de uma região para outra em todo o mundo. O Secrets Manager também usa armazenamento em cache para melhorar a performance. Porém, em alguns casos, isso pode aumentar o tempo. A alteração talvez não fique visível enquanto os dados armazenados em cache anteriormente não atingirem o tempo limite.

Projete seus aplicações globais levando em conta esses possíveis atrasos. Além disso, garanta o funcionamento esperado, mesmo quando uma alteração feita em um local não fique imediatamente visível em outro.

Para obter mais informações sobre como alguns outros AWS serviços são afetados pela consistência eventual, consulte:

- Gerenciamento da consistência de dados no Guia do desenvolvedor do banco de dados do Amazon Redshift
- Modelo de consistência de dados do Amazon S3 no Manual do usuário do Amazon Simple Storage Service
- Garantir consistência ao usar o Amazon S3 e o Amazon EMR para fluxos de trabalho de ETL no blog sobre big data da AWS
- Consistência final do Amazon EC2 na Referência de API do Amazon EC2

“Cannot generate a data key with an asymmetric KMS key” (Não é possível gerar uma chave de dados com uma chave do KMS assimétrica) ao criar um segredo

O Secrets Manager usa uma chave de criptografia simétrica do KMS associada a um segredo para gerar uma chave de dados para cada valor de segredo. Não é possível usar uma chave do KMS assimétrica. Verifique se você está usando uma chave de criptografia do KMS simétrica em vez de uma chave do KMS assimétrica. Para obter instruções, consulte Identificação de chaves do KMS assimétricas.

Uma operação AWS do SDK AWS CLI ou não consegue encontrar meu segredo em um ARN parcial

Em muitos casos, o Secrets Manager pode encontrar seu segredo com parte de um ARN em vez do ARN completo. No entanto, se o nome do seu segredo terminar com um hífen seguido de seis caracteres, talvez o Secrets Manager não consiga encontrar o segredo exclusivamente com base em uma parte de um ARN. Em vez disso, recomendamos que você use o ARN completo ou o nome do segredo.

Mais detalhes

O Secrets Manager inclui seis caracteres aleatórios ao final do nome do segredo para ajudar a garantir que o ARN do segredo seja único. Se o segredo original for excluído e, então, for criado um novo com o mesmo nome, os dois segredos terão ARNs diferentes, em função desses caracteres. Os usuários com acesso ao segredo antigo não terão acesso automático ao novo segredo, uma vez que os ARNs são diferentes.

O Secrets Manager cria um ARN para um segredo com a região, conta, nome do segredo e, em seguida, um hífen e mais seis caracteres, da seguinte forma:

```
arn:aws:secretsmanager:us-east-2:111122223333:secret:SecretName-abcdef
```

Se o nome do segredo terminar com um hífen e seis caracteres, usar apenas parte do ARN pode dar a impressão para o Secrets Manager de que você está especificando um ARN completo. Por exemplo, você pode ter um segredo nomeado `MySecret-abcdef` com o ARN

`arn:aws:secretsmanager:us-east-2:111122223333:secret:MySecret-abcdef-nutBrk`

Se você chamar a seguinte operação, que usa apenas parte do ARN do segredo, talvez o Secrets Manager não encontre o segredo.

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

Esse segredo é gerenciado por um AWS serviço e você deve usar esse serviço para atualizá-lo.

Se você encontrar essa mensagem ao tentar modificar um segredo, o segredo só poderá ser atualizado usando o serviço de gerenciamento listado na mensagem. Para ter mais informações, consulte Segredos gerenciados.

Para determinar quem gerencia um segredo, você pode revisar o nome do segredo. Os segredos gerenciados por outros serviços são prefixados com o ID do respectivo serviço. Ou, no AWS CLI, chame describe-secret e revise o campo. `OwningService`

AWS Secrets ManagerCotas

As APIs de leitura do Secrets Manager têm cotas de TPS altas, e APIs de ambiente de gerenciamento que são chamadas com menos frequência têm cotas de TPS mais baixas. Recomendamos que você evite chamar `PutSecretValue` ou `UpdateSecret` em uma frequência sustentada de mais de uma vez a cada dez minutos. Quando você chama `PutSecretValue` ou `UpdateSecret` para atualizar o valor do segredo, o Secrets Manager cria uma nova versão do segredo. O Secrets Manager remove versões sem rótulo quando há mais de 100, mas não remove versões criadas há menos de 24 horas. Se você atualizar o valor do segredo mais de uma vez a cada dez minutos, criará mais versões do que o Secrets Manager remove e atingirá a cota de versões de segredos.

Você pode operar várias regiões na sua conta, e cada cota é específica para cada região.

Quando uma aplicação em uma Conta da AWS usa um segredo de propriedade de outra conta, isso é conhecido como uma solicitação entre contas. Nas solicitações entre contas, o Secrets Manager limita a conta da identidade que faz as solicitações, não a conta que possui o segredo. Por exemplo, se um aplicação na conta A usar um segredo na conta B, o uso do segredo será aplicado somente às cotas da conta A.

Cotas do Secrets Manager

Nome	Padrão	Ajustá	Descrição
Taxa combinada de solicitações de API de DeleteResourcePolicy, GetResourcePolicy, PutResourcePolicy e ValidateResourcePolicy	Cada região compatível: 50 por segundo	Não	O máximo de transaçõe s por segundo para solicitações de API combinadas de DeleteResourcePoli cy, GetResourcePolicy, PutResourcePolicy e ValidateResourcePolicy.
Taxa combinada de solicitações de API de DescribeSecret e GetSecretValue	Cada região compatíve l: 10.000 por segundo	Não	O máximo de transaçõe s por segundo para solicitações de API combinadas de

Nome	Padrão	Ajustá	Descrição
			DescribeSecret e GetSecretValue.
Taxa combinada de solicitações de API de PutSecretValue, RemoveRegionsFromReplication, ReplicateSecretToRegion, StopReplicationToReplica, UpdateSecret e UpdateSecretVersionStage	Cada região compatível: 50 por segundo	Não	O máximo de transações por segundo para solicitações de API combinadas de PutSecretValue, RemoveRegionsFromReplication, ReplicateSecretToRegion, StopReplicationToReplica, UpdateSecret e UpdateSecretVersionStage.
Taxa combinada de solicitações de API de RestoreSecret	Cada região compatível: 50 por segundo	Não	O máximo de transações por segundo para solicitações de API de RestoreSecret.
Taxa combinada de solicitações de API de RotateSecret e CancelRotateSecret	Cada região compatível: 50 por segundo	Não	O máximo de transações por segundo para solicitações de API combinadas de RotateSecret e CancelRotateSecret.
Taxa combinada de solicitações de API de TagResource e UntagResource	Cada região compatível: 50 por segundo	Não	O máximo de transações por segundo para solicitações de API combinadas de TagResource e UntagResource.

Nome	Padrão	Ajustá	Descrição
Taxa de solicitações de API de BatchGetSecretValue	Cada região compatível: 100 por segundo	Não	O máximo de transaçõe s por segundo para solicitações de API de BatchGetSecretValue.
Taxa de solicitações de API de CreateSecret	Cada região compatível: 50 por segundo	Não	O máximo de transaçõe s por segundo para solicitações de API de CreateSecret.
Taxa de solicitações de API de DeleteSecret	Cada região compatível: 50 por segundo	Não	O máximo de transaçõe s por segundo para solicitações de API de DeleteSecret.
Taxa de solicitações de API de GetRandomPassword	Cada região compatível: 50 por segundo	Não	O máximo de transaçõe s por segundo para solicitações de API combinadas de GetRandomPassword.
Taxa de solicitações de API de ListSecretVersionIds	Cada região compatível: 50 por segundo	Não	O máximo de transaçõe s por segundo para solicitações de API de ListSecretVersionIds.
Taxa de solicitações de API de ListSecrets	Cada região compatível: 100 por segundo	Não	O máximo de transaçõe s por segundo para solicitações de API de ListSecrets.
Comprimento de política baseada em recurso	Cada região compatível: 20.480	Não	O número máximo de caracteres em uma política de permissõe s baseada em recurso anexada a um segredo.

Nome	Padrão	Ajustá	Descrição
Tamanho de valor secreto	Cada região compatível: 65.536 Bytes	Não	O tamanho máximo de um valor secreto criptografado. Se o valor secreto for uma string, esse tamanho será referente ao número de caracteres permitidos no valor secreto.
Segredos	Cada região compatível: 500.000	Não	O número máximo de segredos em cada região da AWS dessa conta da AWS.
Preparação prévia de rótulos anexados em todas as versões de um segredo	Cada região compatível: 20	Não	Número máximo de rótulos de preparação anexados entre todas as versões de um segredo.
Versões por segredo	Cada região compatível: 100	Não	O número máximo de versões de um segredo.

Adicionar novas tentativas à aplicação

Seu cliente da AWS pode ver chamadas para o Secrets Manager falharem devido a problemas inesperados no lado do cliente. Ou as chamadas podem falhar devido à limitação da taxa pelo Secrets Manager. Quando você excede uma cota de solicitação de API, o Secrets Manager controla a utilização da solicitação. Ele rejeita uma solicitação que do contrário seria válida e retorna um erro de throttling. Para ambos os tipos de falhas, recomendamos que você tente a chamada novamente após um breve período de espera. Isso é chamado de estratégia de recuo e repetição.

Se ocorrerem os seguintes erros, adicione novas tentativas ao código da aplicação:

Erros e exceções transitórias

- `RequestTimeout`

- `RequestTimeoutException`
- `PriorRequestNotComplete`
- `ConnectionError`
- `HTTPClientError`

Erros e exceções de controle de utilização e limitação no lado do serviço

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

Para obter mais informações, além do código de exemplo, sobre novas tentativas, recuo exponencial e jitter, consulte os seguintes recursos:

- Recuo exponencial e jitter
- Tempos limite esgotado, novas tentativas e recuo com jitter
- Repetições de erro e recuo exponencial na AWS.

Histórico do documento

A tabela a seguir descreve as mudanças importantes na documentação desde a última versão do AWS Secrets Manager. Para receber notificações sobre atualizações dessa documentação, você poderá se inscrever em um feed RSS.

Alteração	Descrição	Data
Alteração do Secrets Manager para política AWS gerenciada	A política `SecretsManagerReadWrite` gerenciada agora inclui `redshift-serverless` permissão. Para obter mais informações, consulte a política AWS gerenciada para AWS Secrets Manager	12 de março de 2024

Atualizações anteriores

A tabela a seguir descreve mudanças importantes em cada versão do Guia do AWS Secrets Manager usuário antes de fevereiro de 2024.

Alteração	Descrição	Data
Disponibilidade geral	Este é o lançamento público inicial do Secrets Manager.	4 de abr de 2018

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.