Rotating AWS Secrets Manager Secrets

Introduction

AWS Secrets Manager supports automatic rotation of secrets, helping you maintain credential hygiene and reduce the risk of unauthorized access. You can configure rotation using AWS-managed mechanisms or custom Lambda functions.

Overview

Secret rotation in AWS Secrets Manager automates the lifecycle of sensitive data like database credentials, API keys, and tokens. AWS offers two primary models:

- **Managed Rotation (No Lambda)**: Available for specific AWS services with built-in support.
- **Custom Rotation (With Lambda)**: For custom applications or unsupported services.

Pre-requisites

- Secrets Manager permissions (`secretsmanager:*` for full control)
- IAM roles for Lambda (if using custom rotation)
- Target service configured to support secret rotation

Benefits / Use Cases / Examples

- **Eliminates hardcoded secrets**
- **Meets compliance** (e.g., PCI-DSS, ISO 27001)
- **Automates key lifecycle**

Examples:

- RDS PostgreSQL credentials rotated automatically
- Third-party API keys rotated using custom logic in Lambda

How-To: Managed Rotation (No Lambda)

Supported Services

- Amazon RDS (MySQL, PostgreSQL, MariaDB, SQL Server, Oracle)
- Amazon DocumentDB

- Amazon Redshift

Steps

1. Create or store the secret in Secrets Manager.
2. Enable rotation and choose "Use a rotation function hosted by Secrets Manager".
3. Select the service type (e.g., RDS MySQL).
4. Choose the database instance and credentials.
5. Secrets Manager handles the rotation automatically using an internal AWS Lambda.

Rotation Schedule:

- Minimum: every 1 day
- Default: every 30 days

How-To: Custom Rotation (With Lambda)

Use Cases

- APIs
- Third-party systems
- Applications requiring custom credential update logic

Steps

1. Store the secret in Secrets Manager
2. Create a Lambda function with rotation logic
3. Enable rotation and select “Use a custom Lambda function”
4. Choose or create a Lambda function and define the rotation schedule
5. Ensure the Lambda implements four steps:
    - `createSecret`
    - `setSecret`
    - `testSecret`
    - `finishSecret`

IAM Role Permissions:

- Lambda must have `secretsmanager:GetSecretValue`, `secretsmanager:PutSecretValue`, `secretsmanager:UpdateSecretVersionStage`, etc.
- The secret must have a resource policy allowing the Lambda to access it

Considerations

- **Atomicity**: Custom Lambda must ensure secrets are updated in both the secret store and the target service
- **Testing**: Always use `testSecret` logic to validate credentials before promotion
- **Permissions**: Be careful with the IAM role assigned to Lambda
- **Logging**: Ensure Lambda logs are monitored for rotation failures (CloudWatch)
- **Backup**: Store previous versions for rollback if needed

Documentation

- AWS Secrets Manager Rotation Guide: https://docs.aws.amazon.com/secretsmanager/latest/userguide/rotating-secrets.html
- Rotation Lambda Template: https://docs.aws.amazon.com/secretsmanager/latest/userguide/rotate-secrets_lambda.html
- Supported Managed Rotation Services: https://docs.aws.amazon.com/secretsmanager/latest/userguide/rotating-secrets-required-permissions.html