What Is AWS Secrets Manager?

What Is AWS Secrets Manager?

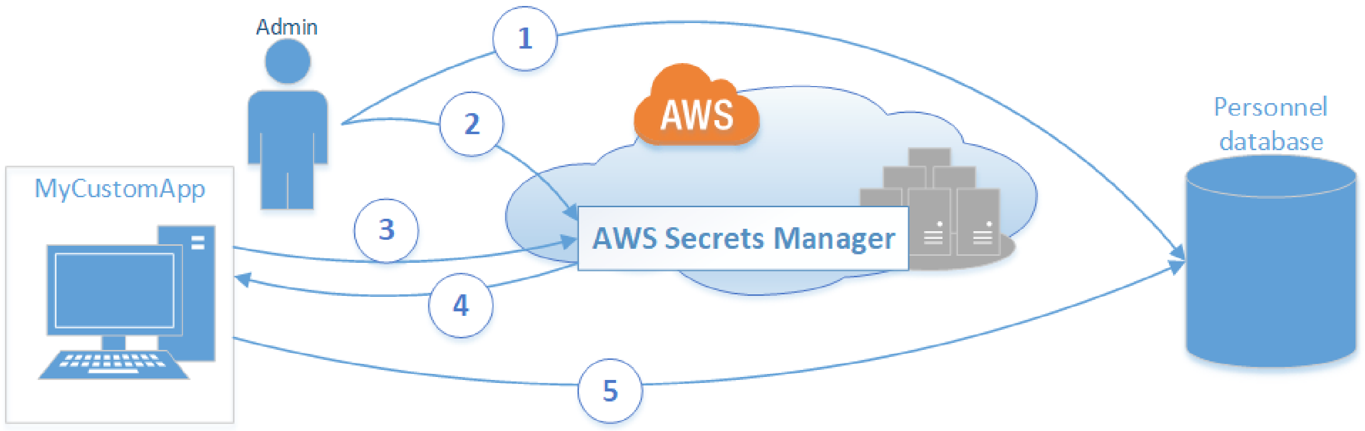
AWS provides a service, AWS Secrets Manager, to make it easier for you to manage application secrets. *Secrets* can be database credentials, passwords, third-party API keys, and even arbitrary text. You can centrally store and control access to these secrets by using the Secrets Manager console, the Secrets Manager command line interface (CLI), or the Secrets Manager API and SDKs.

In the past, when you created a custom application that retrieves information from a database, you typically embedded the credentials (the secret) for accessing the database directly into the application. When the time came to rotate the credentials, you had to do more than just create new credentials. You had to invest the time to update the application to use the new credentials. Then you had to distribute the updated application. If you had multiple applications with shared credentials and you missed updating one of them, the application would break. Because of this risk, many customers choose not to regularly rotate their credentials, which effectively substitutes one risk for another.

Secrets Manager enables you to replace hardcoded credentials in your code, including passwords, with an API call to Secrets Manager and retrieve the secret programmatically. This helps ensure the secret can't be compromised by someone examining your code, because the secret isn't in the code. Also, you can configure Secrets Manager to automatically rotate the secret for you according to a specified schedule. This enables you to replace long-term secrets with short-term ones, which significantly reduces the risk of compromise.

Basic Secrets Manager Scenario

The following diagram illustrates the most basic scenario. It displays how you can store credentials for a database in Secrets Manager, and then use those credentials in an application to access the database.



1. The database administrator creates a set of credentials on the Personnel database to use with an application called MyCustomApp. The administrator also configures those credentials with the required permissions to access the Personnel database.
2. The database administrator stores the credentials as a secret in Secrets Manager named *MyCustomAppCreds*. Secrets Manager encrypts and stores the credentials within the secret as the *protected secret text*.
3. When MyCustomApp needs to access the database, the application queries Secrets Manager for the secret named *MyCustomAppCreds*.
4. Secrets Manager retrieves the secret, decrypts the protected secret text, and returns it to the client application over a secureHTTPS with TLS channel.
5. The client application parses the credentials, connection string, and any other required information from the response and then uses the information to access the database server.

Features of Secrets Manager

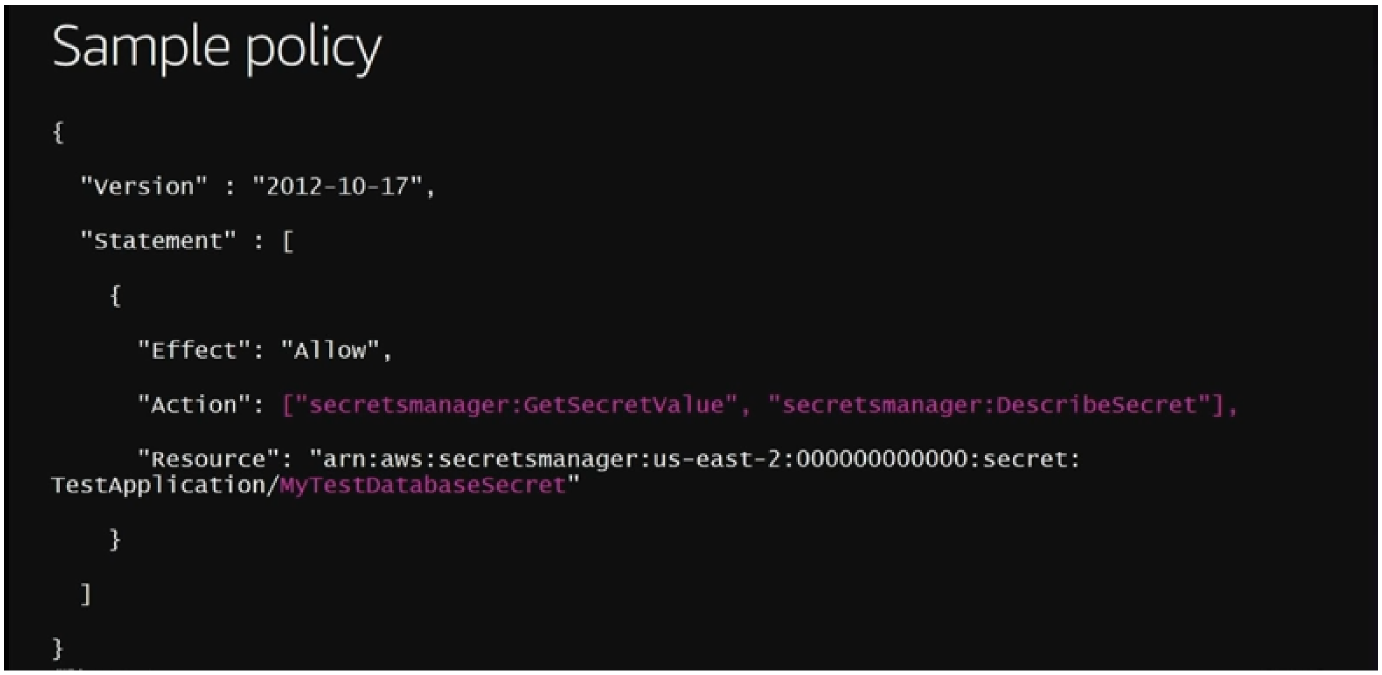
Programmatically Retrieving Encrypted Secret Values at Runtime Instead of Storing Them

Storing Just About Any Kind of Secret

Encrypting Your Secret Data

Automatically Rotating Your Secrets

Control Access to Secrets



Sample policy

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Effect": "Allow",
      "Action": ["secretsmanager:GetSecretValue", "secretsmanager:DescribeSecret"],
      "Resource": "arn:aws:secretsmanager:us-east-2:000000000000:secret:
TestApplication/MyTestDatabaseSecret"
    }
  ]
}
```

How can my application use these secrets?

First, you must write an AWS Identity and Access Management (IAM) policy permitting your application to access specific secrets. Then, in the application source code, you can replace secrets in plain text with code to retrieve these secrets programmatically using the Secrets Manager APIs.

Pricing

$0.40 per secret per month

$0.05 per 10,000 API calls

30-day free trial available