Practical: AWS Secrets Manager (Javascript)

Table of Contents

AWS Secrets Manager is a convenient way of storing secrets such as API keys, encryption keys, and private keys. It integrates with other services such as RDS to configure passwords in managed services that you can later retrieve programmatically. This saves a lot of effort in managing these passwords yourself. Secrets manager also has functionality for generating random passwords and periodically rotating secrets.

Prerequisites and references

- AWS CAB432 login (https://d-97671c4bd0.awsapps.com/start/#/?tab=accounts)
- AWS CAB432 Secrets manager (https://ap-southeast-2.console.aws.amazon.com/secretsmanager/listsecrets?region=ap-southeast-2)
- API reference for Secrets Manager (https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/client/secrets-manager/)
- Code samples for Secrets Manager (https://github.com/awsdocs/aws-doc-sdk-examples/tree/main/javascriptv3/example_code/secrets-manager)
- secretsdemo.zip (https://canvas.qut.edu.au/courses/20367/files/6583679/download)

Using Secrets Manager

There are two use cases that you are most likely to encounter in this unit:

- Passwords for managed services. In particular, RDS can use the Secrets Manager to manage the passwords for accessing managed databases. With this you no longer need to worry about setting passwords on the server. In your application code, you can retrieve the secret using the SDK and use it to connect to the database as a client.
- API keys. Typically you will use an API key, which should be treated as secret, to gain access to an external API. You can create secrets to store these keys manually using the AWS console and then retrieve them using the SDK in your application.

In these and other use cases one major advantage is that you don't need to build secrets into VM images used for automatically scaling out. New VMs can request the secrets as they start up. This

has a couple of advantages:

- Your secrets are stored encrypted with proper access control, as opposed to being buried in a VM or container image
- If you need to change the secret (eg. if an API key expires) then you can do this in one place without recreating VM or container images

Create a secret

- Go to the Secrets Manager on the AWS console
- Click *Store a new secret*
- At this point you may see several errors about lacking permissions. This is not a problem for our use cases.
- *Secret Type*:
  - Choose *Other type of secret*
- *Key/value pairs*:
  - A secret can have multiple values stored. For example use `mysecret` as the key and `supersecret` as the value.
- *Next* to go to the next page
- *Secret name and description*
  - *Secret Name*: we suggest something with your username, such as `n1234567-demosecret`
  - *Description*: optional
- *Tags*:
  - Key `qut-username` with value set to your username, eg `n1234567@qut.edu.au`
  - Key `purpose` with value set to `practical`
- *Next* and then *Next* again
- Note the *Sample code* at the bottom, which includes some Javascript code for accessing the secret.
- *Store* to create the secret

Access a secret with the SDK

- Create a new node app and install the Secrets Manager SDK module

```
mkdir secretsdemo
cd secretsdemo
npm init -y
npm i @aws-sdk/client-secrets-manager
```

- Create `index.js` in the app with the following contents (changing the username to your own):

```
SecretsManager = require("@aws-sdk/client-secrets-manager");

const secret_name = "n1234567-demosecret";
const client = new SecretsManager.SecretsManagerClient({ region: "ap-
southeast-2" });
```

```
async function main() {
   try {
      response = await client.send(
         new SecretsManager.GetSecretValueCommand({
            SecretId: secret_name
         })
      );
      const secret = response.SecretString;
      console.log(secret);
   } catch (error) {
      console.log(error);
   }
}

main();
```

- Run with `node index.js`. You should see the key/value pair stored in your secret printed out.

While it is also possible to create secrets and perform other actions with the SDK, this is not likely to be required for uses cases in this unit.

Delete secrets when done

When you are done, please use the AWS console to delete secrets that you are no longer using. These won't be deleted immediately as AWS requires a waiting period of at least 7 days before deletion.

Complete code: **secretsdemo.zip (https://canvas.qut.edu.au/courses/20367/files/6583679/download)**