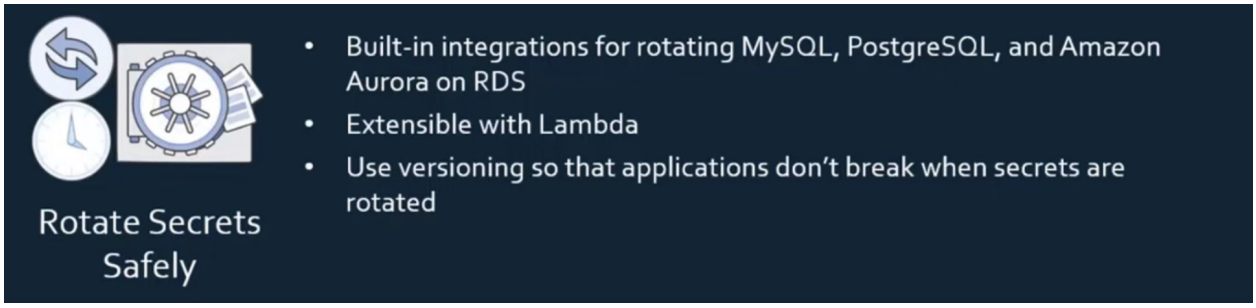
AWS secret manager

"Helps in managing, rotating (renewing) and retrieving database credentials, api keys, and other secrets with authorized access only. It also helps in auditing (when the secret was accessed, who is accessing the secret, etc)".

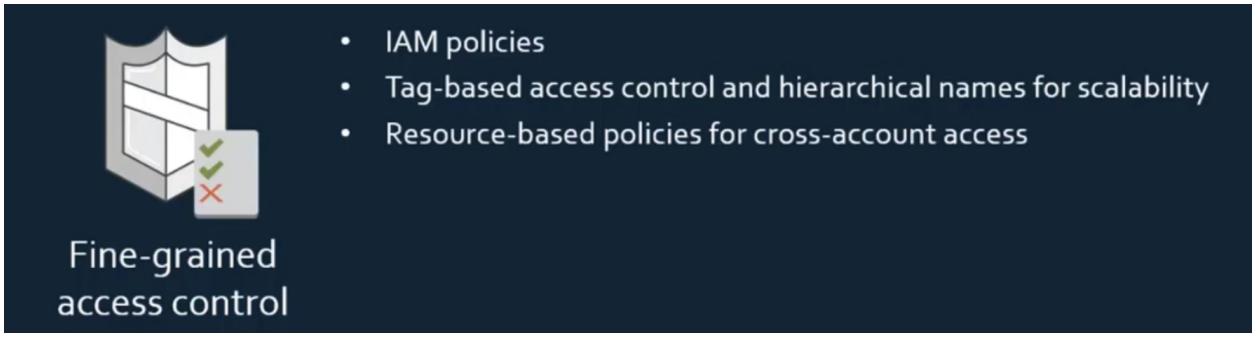
Features

1. Rotation



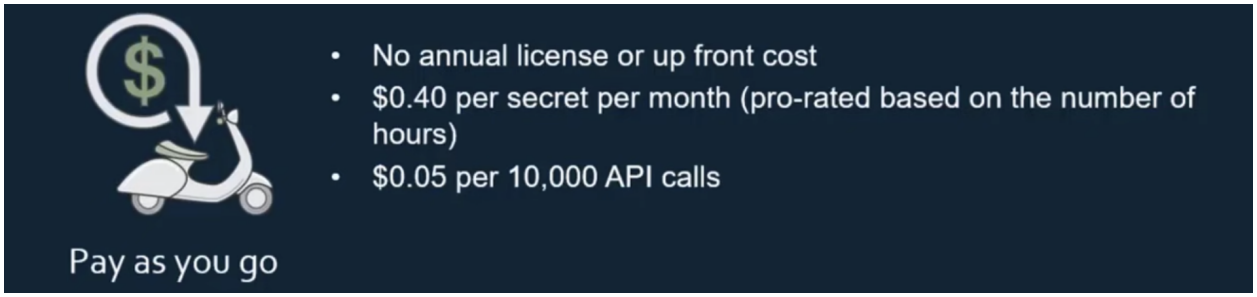
- Easy to rotate other secrets by writing your own lambda code for rotation.
- Versioning of secrets is also done so that applications do not break when secrets are rotated.

2. Fine-grained access

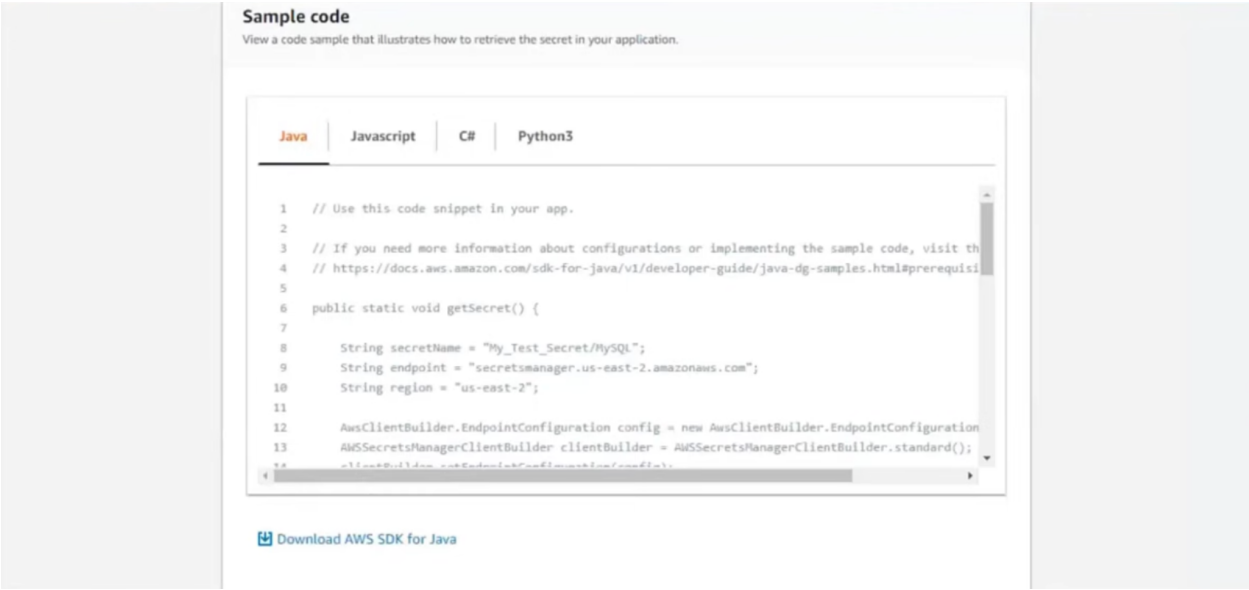


3. Security, Monitoring, Auditing

4. Pricing

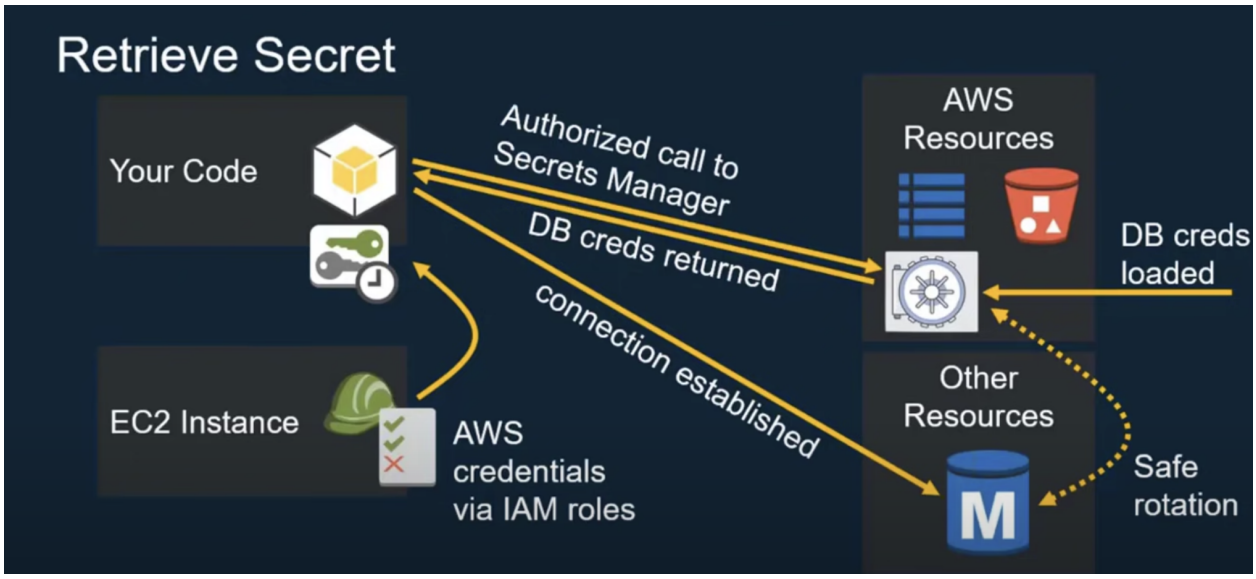


Note: The code to retrieve a secret in programming languages is provided by AWS at the time of creating the secret.



Command to get a secret,
aws secretsmanager get-secret-value --secret-id <secret_name_here>

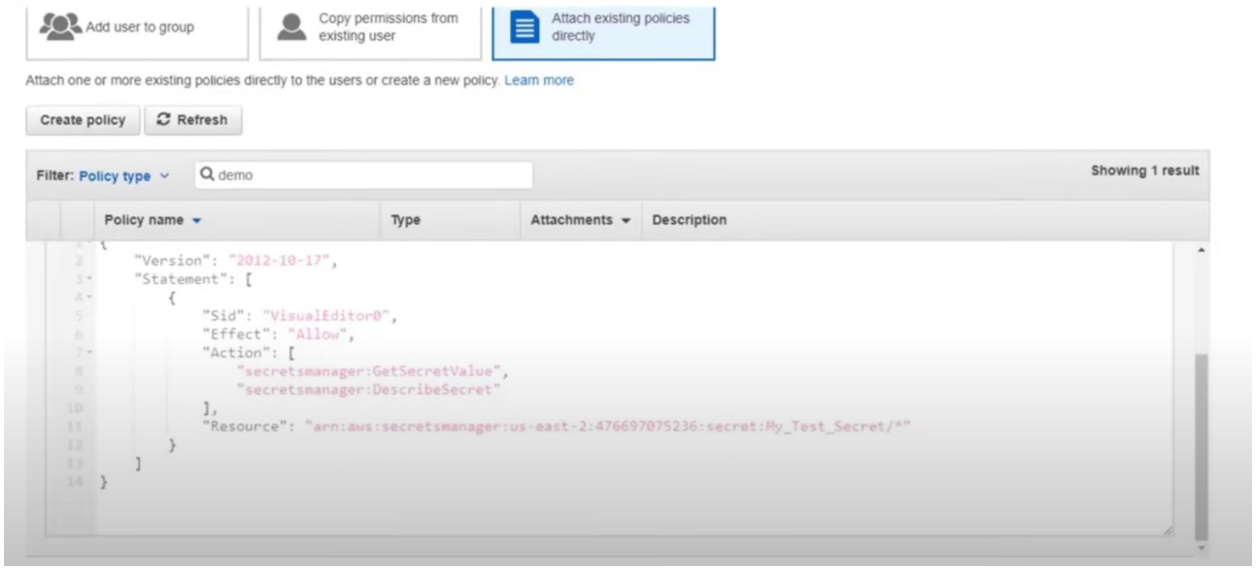
Flow for retrieving the secret

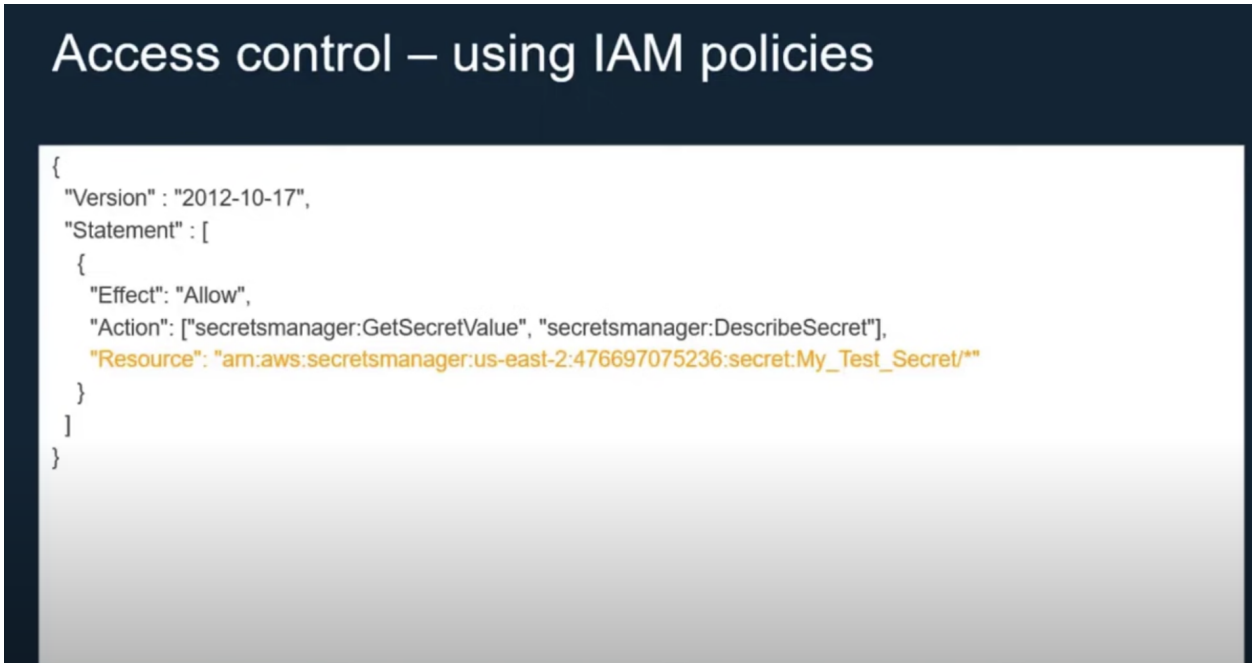


1. Give IAM permissions to your resource (where the application is running, EC2 in this case) to authenticate with an AWS secret manager.
2. Application will then request for secrets (db credentials in this case).
3. Applications can consume other AWS resources (RDS in this case) using those credentials.
4. If the rotation in the secret manager is enabled it will rotate/change the secrets with the resource (RDS in this case) for rotation.

Attaching policies to users for authorized access on secrets

This policy will allow user to perform get secret, and describe secret operations on the secrets having prefix “My_Test_Secrets/*”





We can also use tags in our policies,

Auditing, Monitoring

This can be done by integrating AWS cloudtrail with AWS secret manager (ASM). In cloudtrail, we can search for the events triggered on the secrets. We can see who tried to access the secrets, when, the request was denied or grant, etc