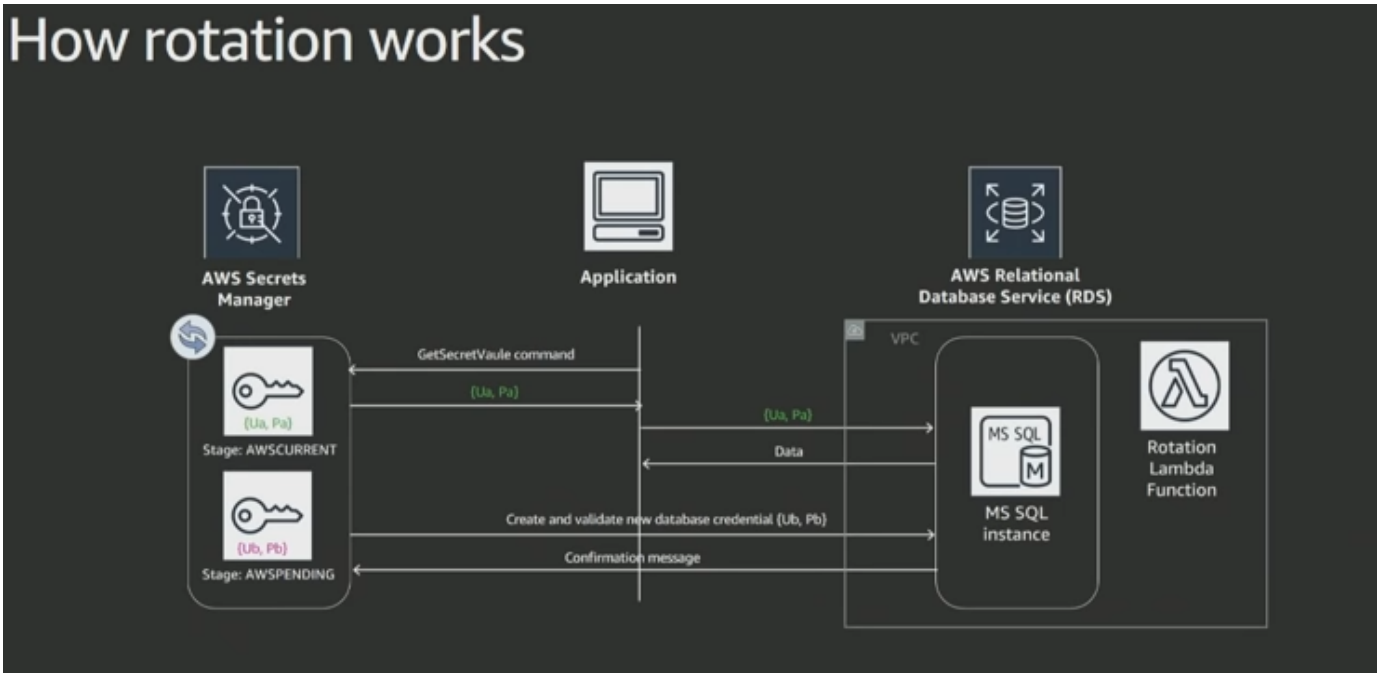
Rotation of secrets in Secrets Manager with Demo

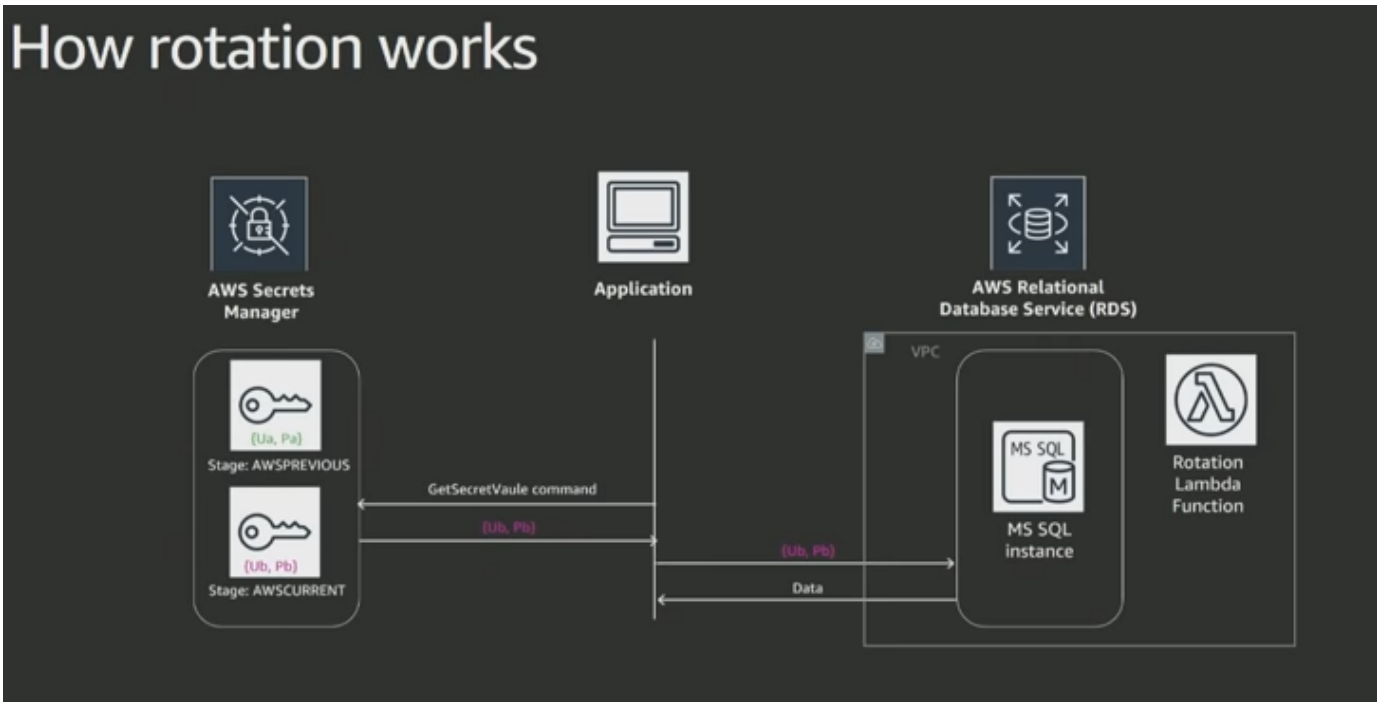
Rotation of secrets in Secrets Manager with Demo

You can configure AWS Secrets Manager to automatically rotate the secret for a secured service or database. Secrets Manager already natively knows how to rotate secrets for supported Amazon RDS databases. However, Secrets Manager also can enable you to rotate secrets for other databases or third-party services. Because each service or database can have a unique way of configuring its secrets, Secrets Manager uses a Lambda function that you can customize to work with whatever database or service that you choose. You customize the Lambda function to implement the service-specific details of how to rotate a secret.

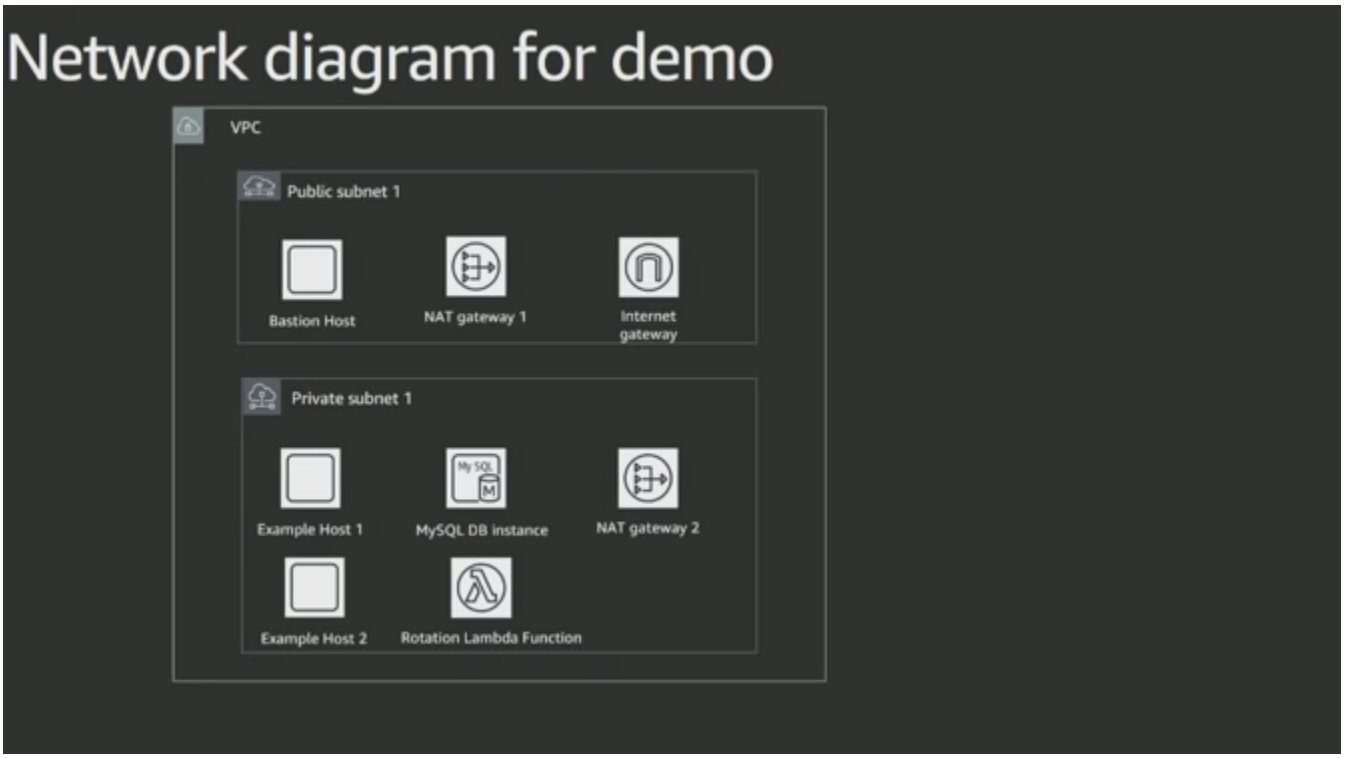
When you enable rotation for a secret by using the **Credentials for RDS database**, **Credentials for Redshift cluster**, or **Credentials for DynamoDB database** secret type, Secrets Manager provides a Lambda rotation function for you and populates the function's Amazon Resource Name (ARN) in the secret automatically. You typically don't need to do anything for this to work other than to provide a few details. For example, you specify which secret has permissions to rotate the credentials, and how often you want to rotate the secret.

When you enable rotation for a secret with **Credentials for other database** or some **Other type of secret**, you must provide the code for the Lambda function. The code includes the commands that are required to interact with your secured service to update or add credentials.





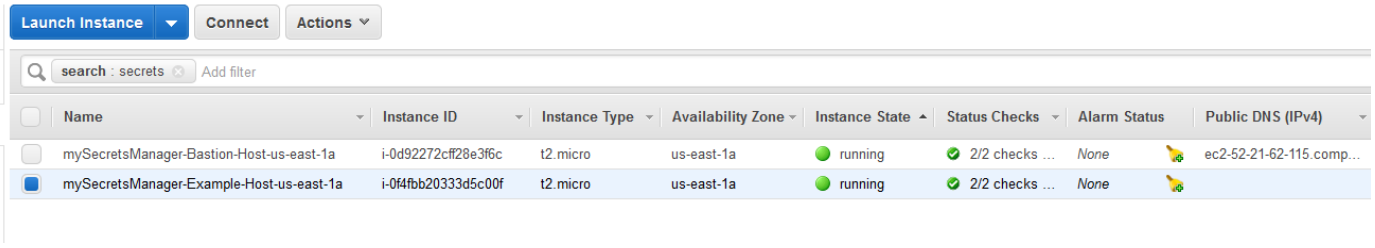
Demo -



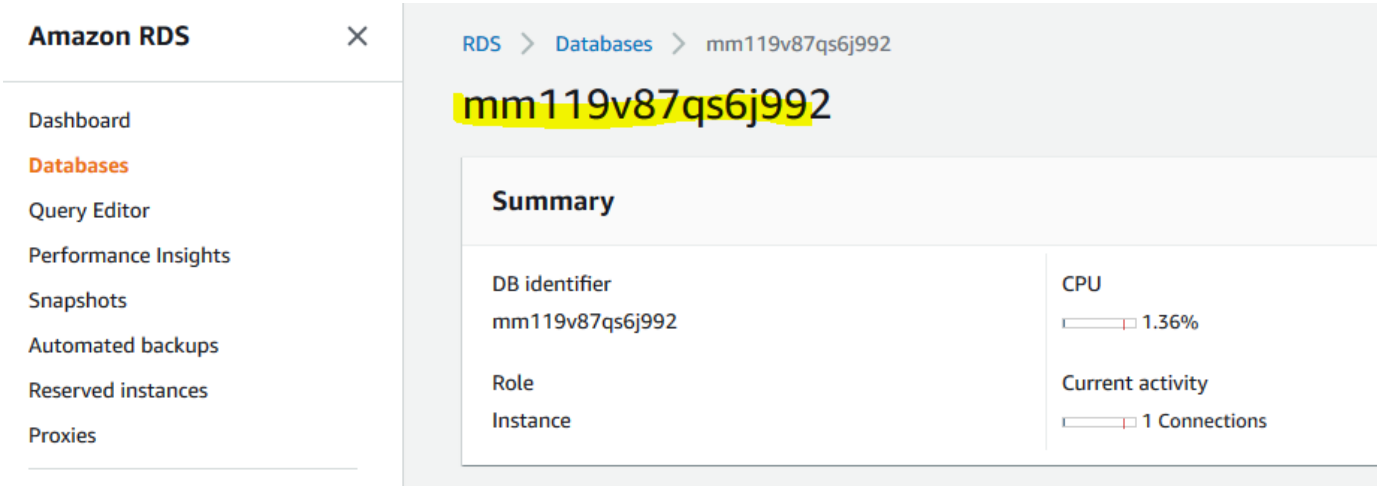
The below cloud formation template will do the above provisioning,

Once the stack is created,

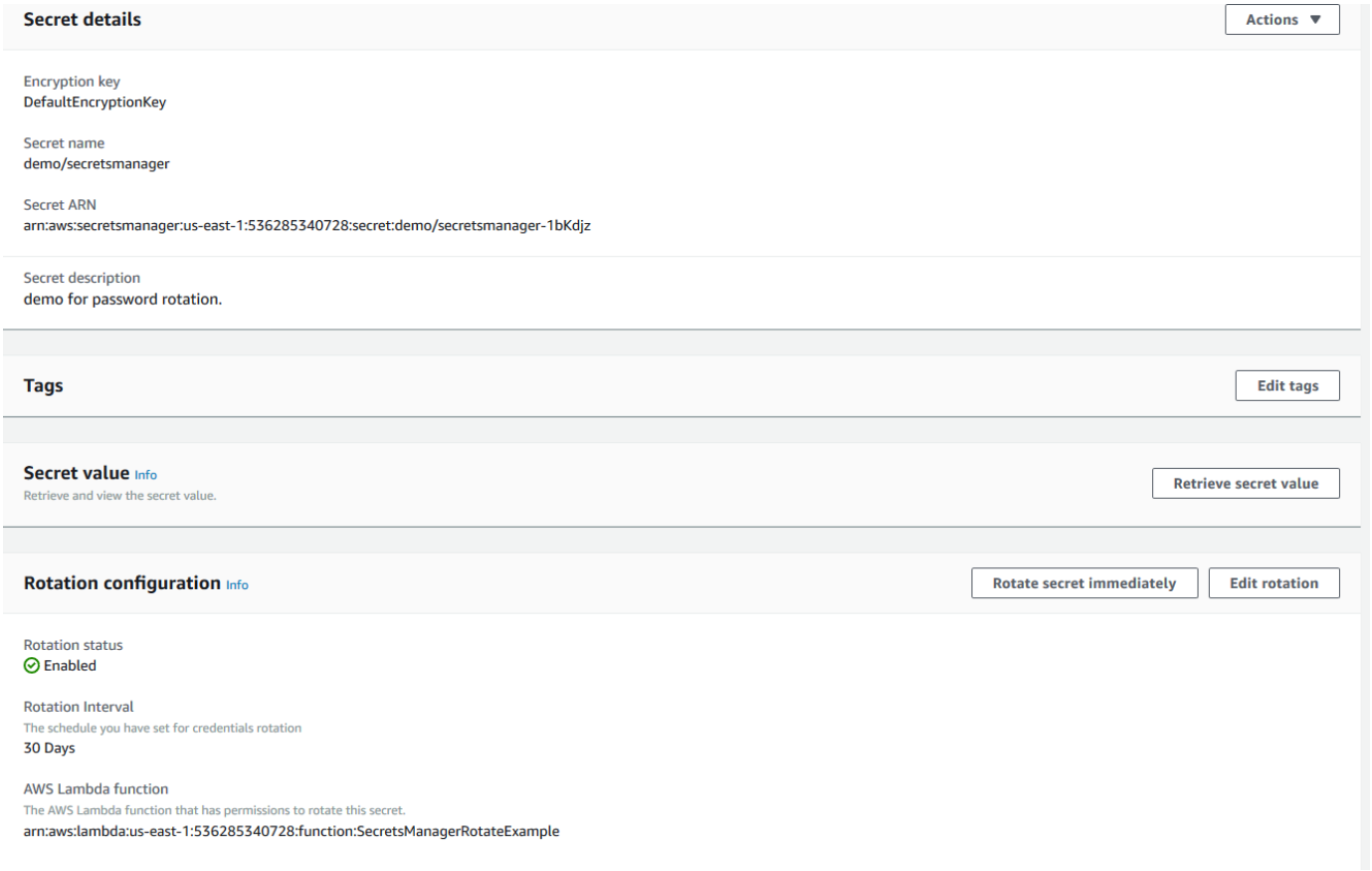
We can check for the EC2 instances created, one in public subnet and another in the private subnet.



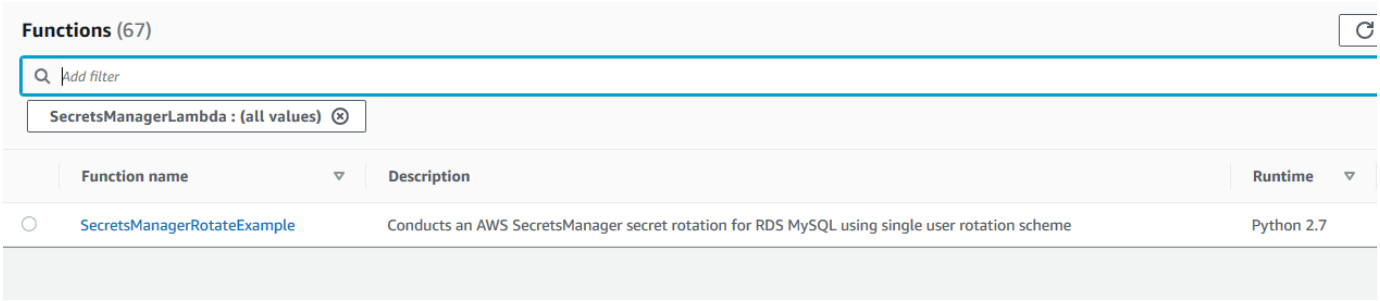
We can also see the RDS instance created with no internet access.



We need to go to the secrets manager create a secret for this database and enable the rotation policy.



A lambda function will automatically be created,



Now to login to this RDS database, 1st login to the Public Instance and ssh to the private instance.

Install aws2 cli and configure aws.

Run the below.

```
[ec2-user@ip-10-0-3-102 ~]$ HOST=$(aws2 secretsmanager get-secret-value --secret-id demo/secretsmanager | jq -r '.SecretString | fromjson | .host')

[ec2-user@ip-10-0-3-102 ~]$ echo $HOST

mm119v87qs6j992.ccac5zy4egop.us-east-1.rds.amazonaws.com

[ec2-user@ip-10-0-3-102 ~]$ USERNAME=$(aws2 secretsmanager get-secret-value --secret-id demo/secretsmanager | jq -r '.SecretString | fromjson | .username')

[ec2-user@ip-10-0-3-102 ~]$ echo $USERNAME

awsadmin

[ec2-user@ip-10-0-3-102 ~]$ PASSWORD=$(aws2 secretsmanager get-secret-value --secret-id demo/secretsmanager | jq -r '.SecretString | fromjson | .password')

[ec2-user@ip-10-0-3-102 ~]$ echo $PASSWORD

y>%O2iZk~nCF>(wHDz9td0O7tty,,Qee

[ec2-user@ip-10-0-3-102 ~]$ mysql -h $HOST -u $USERNAME -p"$PASSWORD"

Warning: Using a password on the command line interface can be insecure.

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 37

Server version: 5.7.22 Source distribution

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its

affiliates. Other names may be trademarks of their respective

owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```