😊 CloudHSM (Cloud Hardware Security Module)

CloudHSM (Cloud Hardware Security Module) is a cloud-based service that provides dedicated hardware security modules (HSMs) in the cloud. An HSM is a specialized hardware device designed to manage and protect cryptographic keys and perform cryptographic operations, such as encryption, decryption, and digital signatures, in a highly secure environment.

CloudHSM is a service provided by cloud platforms, such as AWS, that offers a hardware-based solution for cryptographic key management. Here's a deeper dive into its features, benefits, architecture, and typical use cases:

Detailed Features:

1. **Single-Tenant Architecture:**
    - Each CloudHSM instance is a dedicated appliance for a single customer, ensuring that resources are not shared with others, which enhances security and isolation.
2. **High Availability and Redundancy:**
    - CloudHSM supports the creation of clusters across multiple availability zones, ensuring that the service is highly available and fault-tolerant.
3. **Regulatory Compliance:**
    - CloudHSM is designed to help organizations meet compliance standards such as FIPS 140-2 Level 3, GDPR, HIPAA, and PCI DSS. This makes it suitable for industries like finance, healthcare, and government.
4. **Cryptographic Algorithms Supported:**
    - CloudHSM supports a wide range of cryptographic algorithms, including RSA, ECC (Elliptic Curve Cryptography), AES (Advanced Encryption Standard), and SHA (Secure Hash Algorithm).
5. **Multi-Region Support:**
    - You can deploy CloudHSM clusters in different regions, which is beneficial for global applications requiring consistent security policies across regions.
6. **Customizable Policies:**
    - Users can create and enforce custom security policies, including key usage restrictions and access controls, providing fine-grained control over how cryptographic keys are used.

Benefits:

1. **Enhanced Security:**

- By keeping cryptographic operations within a tamper-resistant hardware environment, CloudHSM reduces the risk of key exposure and ensures high security for sensitive operations.

2. **Performance:**
    - CloudHSM offers dedicated hardware resources, which ensures high-performance cryptographic operations without the overhead of shared resources.
3. **Full Key Control:**
    - Unlike other managed key services, CloudHSM gives customers full control over their keys. This includes the ability to import and export keys, and ensure that only authorized entities can access them.
4. **Ease of Integration:**
    - CloudHSM can be integrated with existing applications and services using industry-standard APIs like PKCS#11, Java JCA/JCE, and Microsoft CNG.
5. **Cost-Effectiveness:**
    - By using a cloud-based HSM service, organizations avoid the capital expenditure and maintenance costs associated with on-premises HSMs, while still meeting high security standards.

Architecture:

- **Cluster-Based Architecture:**
    - CloudHSM instances can be grouped into clusters, which distribute the cryptographic workload across multiple HSMs. Clusters can be set up across multiple availability zones to ensure high availability and resilience against hardware failures.
- **Client Software:**
    - AWS CloudHSM provides client software that interfaces with the HSM appliance, enabling applications to communicate securely with the HSM using standard cryptographic APIs.
- **VPC Integration:**
    - CloudHSM is typically deployed within a Virtual Private Cloud (VPC), allowing organizations to place the HSM within their network architecture, ensuring secure communication between their applications and the HSM.

Use Cases:

1. **Data Encryption:**
    - Encrypt sensitive data at rest and in transit using keys managed by CloudHSM. This is especially important for protecting sensitive customer data, financial information, and intellectual property.

2. **Digital Certificates and PKI:**
    - CloudHSM can be used to manage keys for Public Key Infrastructure (PKI) and issue digital certificates for secure communications.
3. **Database Encryption:**
    - Encrypt databases like Oracle, SQL Server, and others using CloudHSM to manage encryption keys securely.
4. **Code Signing:**
    - Securely sign software code to ensure its integrity and authenticity before deployment or distribution.
5. **Blockchain and Cryptocurrency:**
    - Manage private keys for blockchain and cryptocurrency applications in a highly secure environment.

Deployment Considerations:

- **Latency:**
    - Depending on the application, the latency introduced by the network communication between the HSM and the application should be considered, especially for real-time operations.
- **Scalability:**
    - Plan for the scalability requirements of your application. While CloudHSM can be scaled horizontally by adding more HSMs to a cluster, you need to ensure that the architecture can handle peak loads.
- **Cost:**
    - While CloudHSM offers a high level of security, it is typically more expensive than software-based key management services. Consider your security requirements and budget when deciding to use CloudHSM.

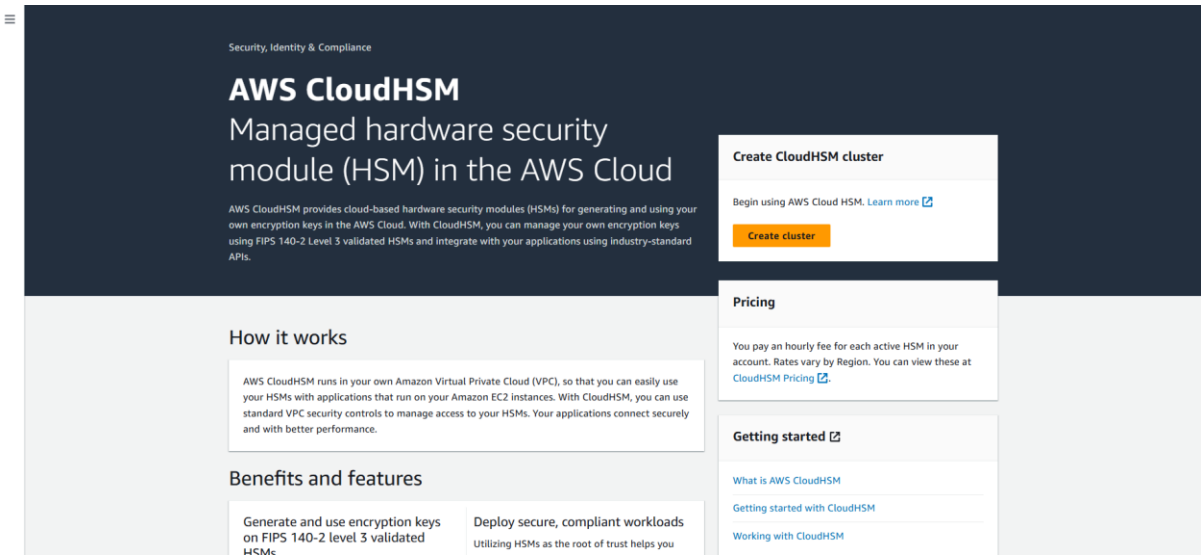
In summary, CloudHSM is a robust solution for organizations needing to perform cryptographic operations in a highly secure environment while maintaining full control over their keys. It is particularly well-suited for applications that must comply with strict regulatory requirements and need to protect highly sensitive data.

This guide walks you through setting up and managing an AWS CloudHSM cluster. The process involves creating a CloudHSM cluster, initializing and configuring it, and activating it. You'll create and configure an IAM user to manage the cluster, generate and sign certificates, and set up the CloudHSM client on an EC2 instance. After ensuring proper communication between the EC2 instance and the CloudHSM cluster, you'll log in to the HSM, change the password for the default user, and activate the cluster. Finally, an additional HSM is added for redundancy, and the resources are cleaned up after use.

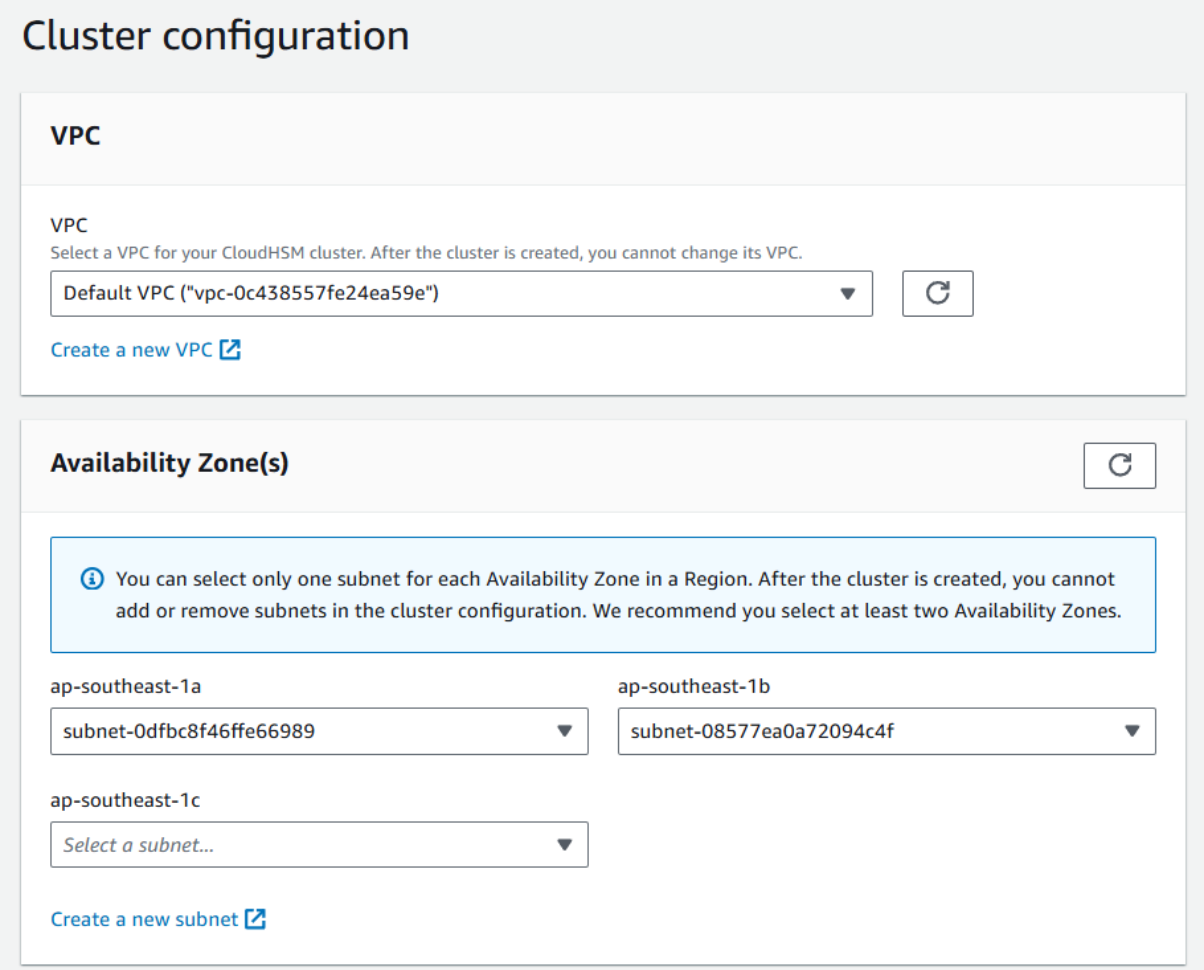
End Goal: Successfully set up a CloudHSM cluster with at least two HSMs in different availability zones, ensuring proper configuration, security, and activation.

😄 To begin with the Lab:

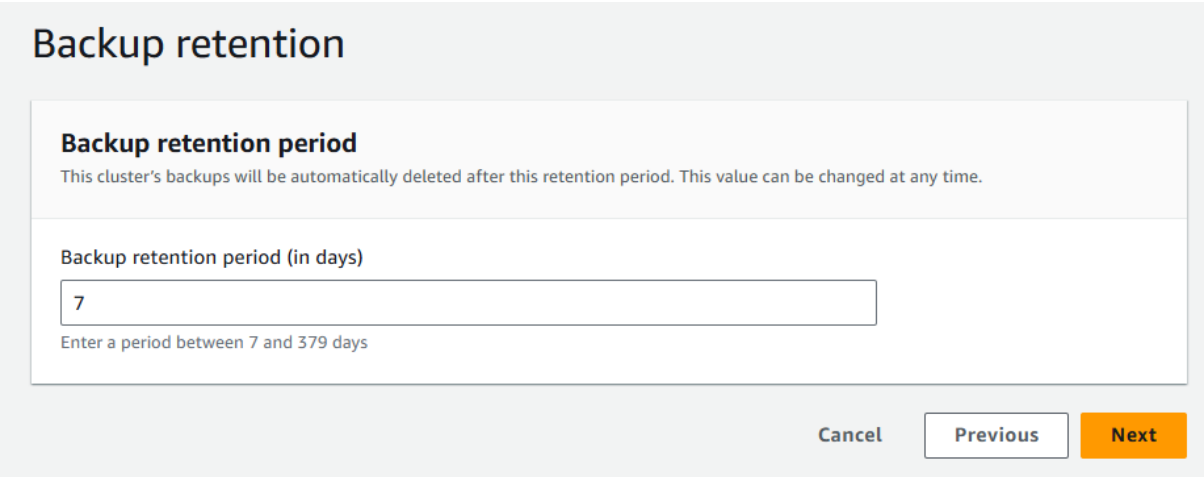
1. In your AWS Console search for CloudHSM, from its dashboard click on Create cluster.



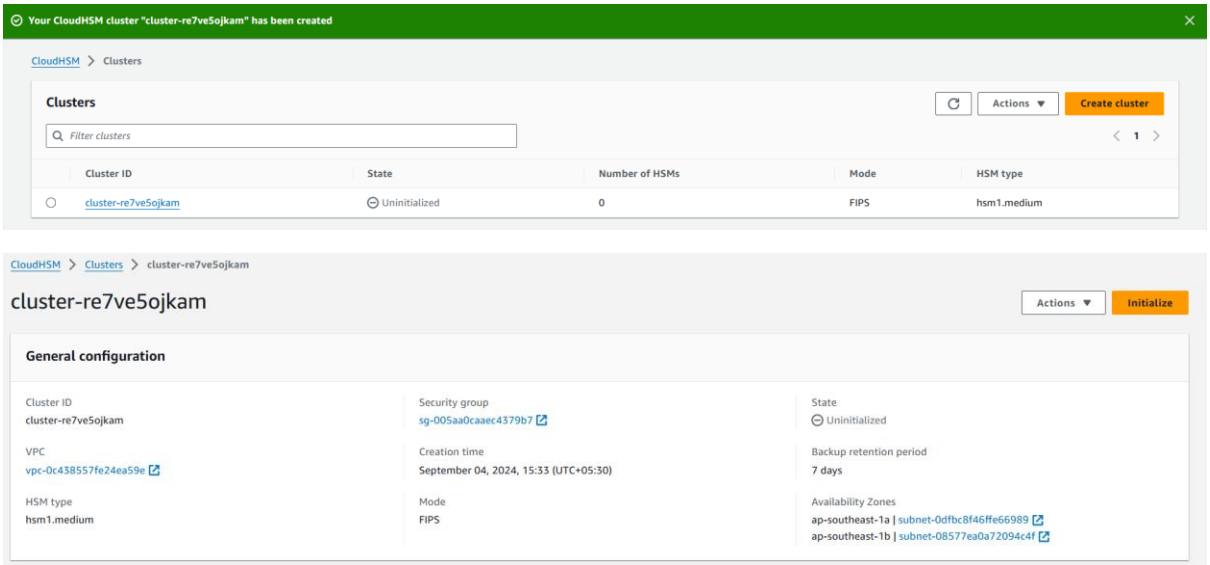
2. On step 1 choose your default VPC and then for the AZ choose any two subnets. Then keep rest of the settings to default and click on next.



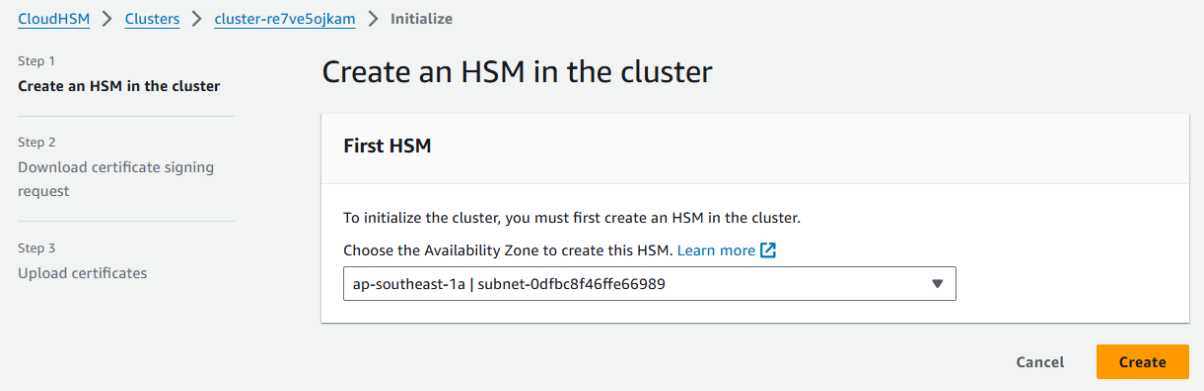
3. Now in the back retention choose 7 days and then move to the review page and create your cluster.
4. It will take at least 10 to 15 minutes so, you have to wait.



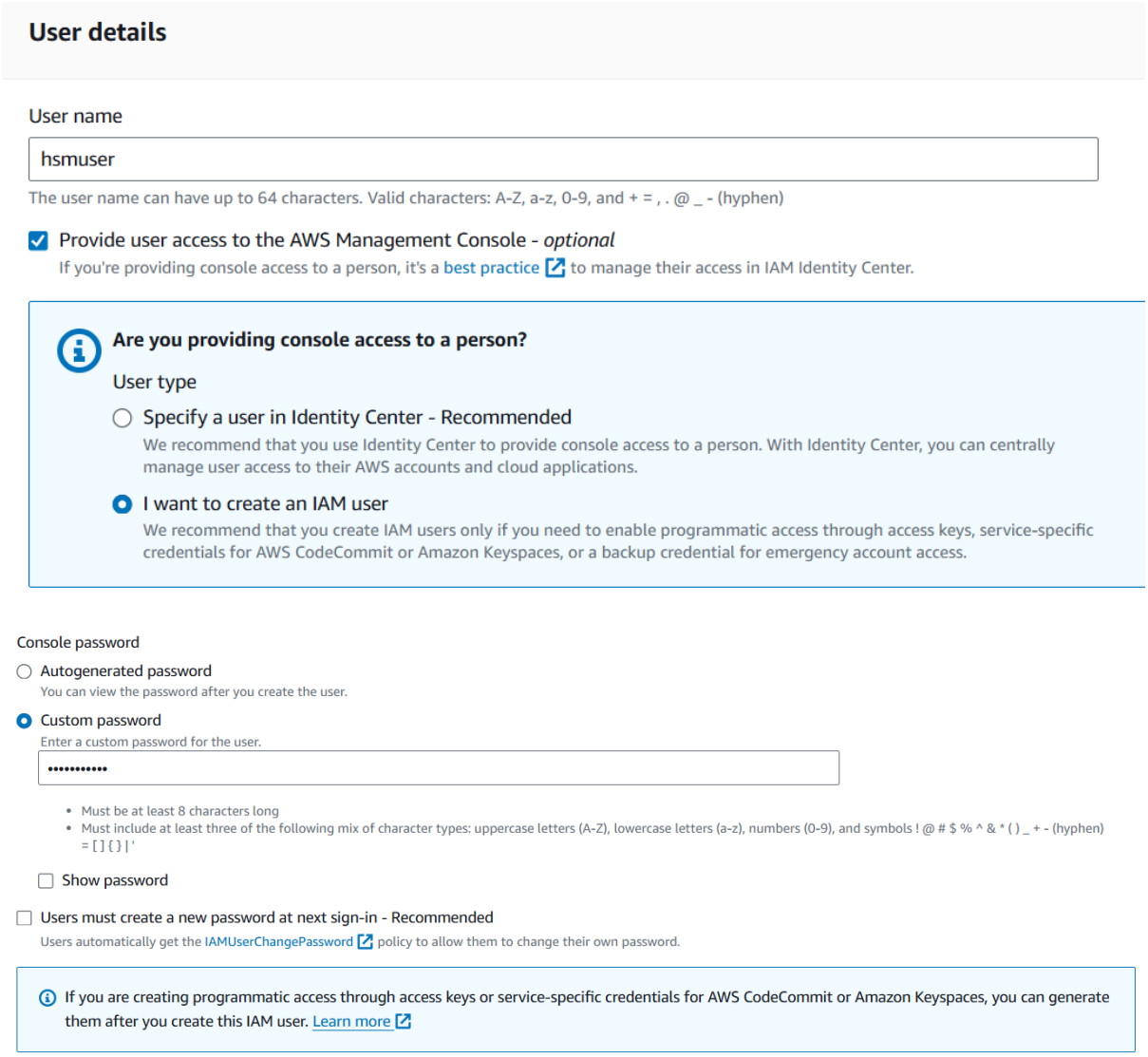
5. Below you can see that our cluster has been created but it is not initialized yet. So, click on it and inside you will see an option to initialize it.



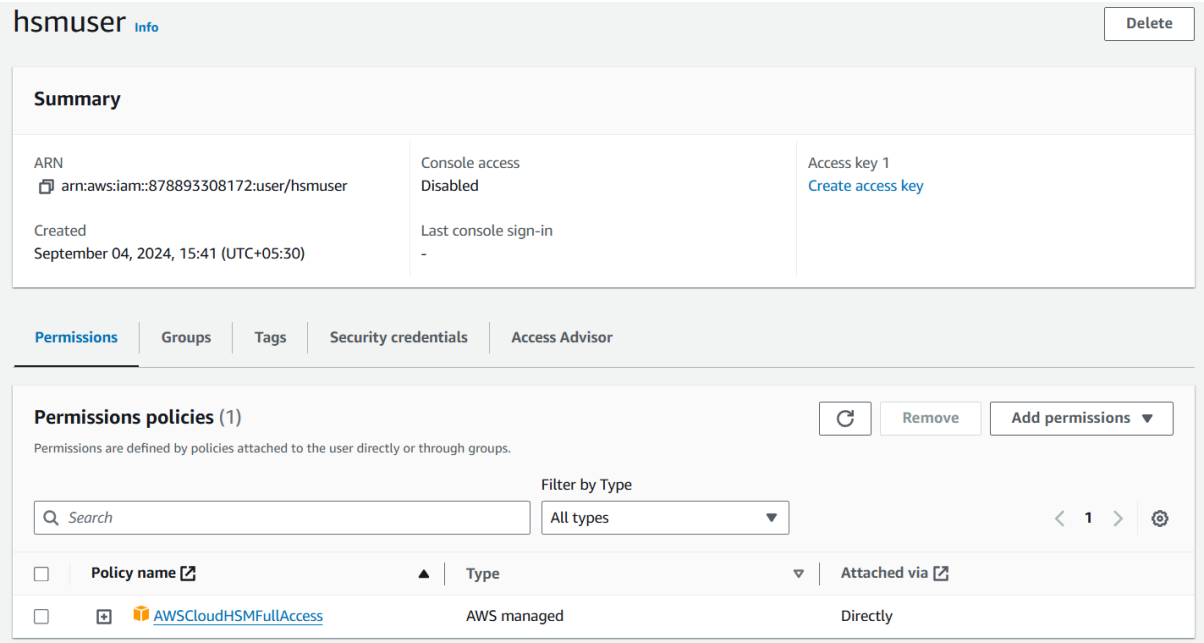
6. Then you have to choose one of the availability zones and click on create. Now wait until it gets initialized.



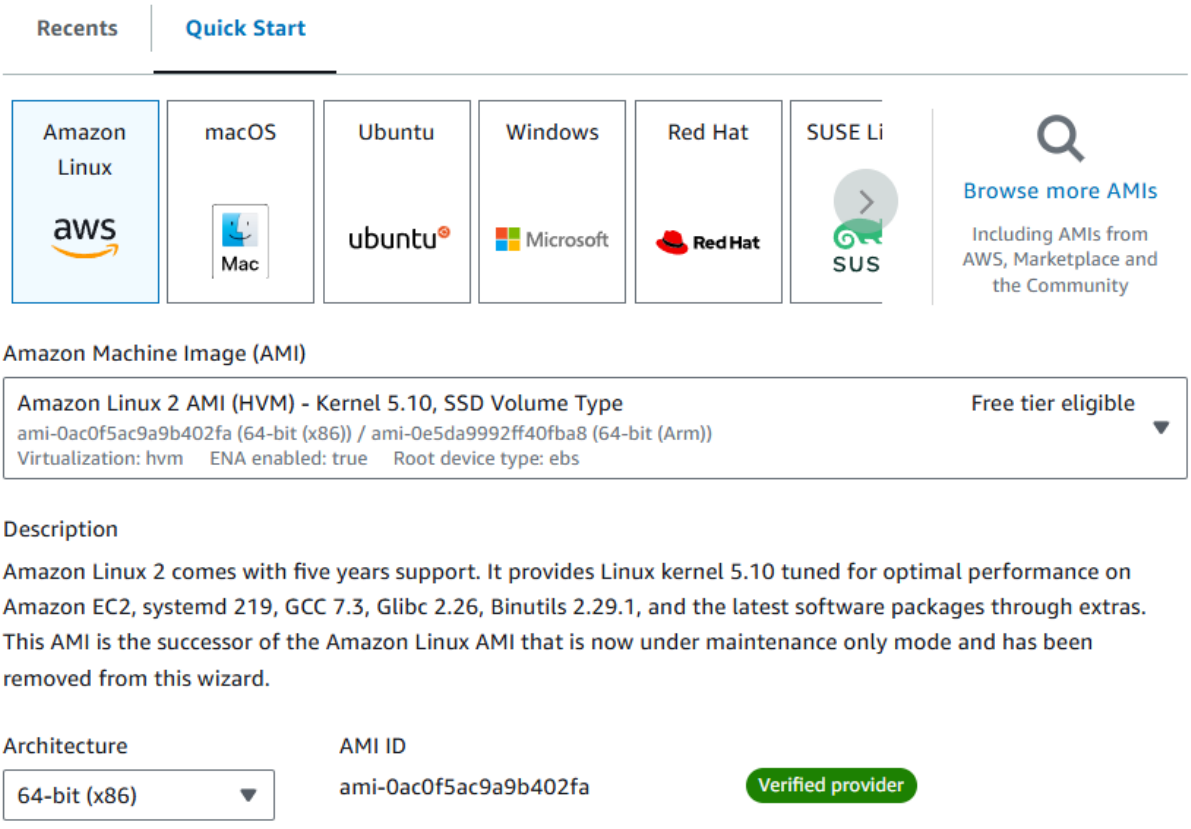
7. Now we need to create an IAM user named **hsmuser** with permission to initialize the cluster. You can leverage **CloudHSMFullAccess** existing policy for this lab.
8. Go to IAM and click on Create User. Give a name to your user and choose to give it access to the console then give a password of your choice. Also, disable not to change the password at the next login.



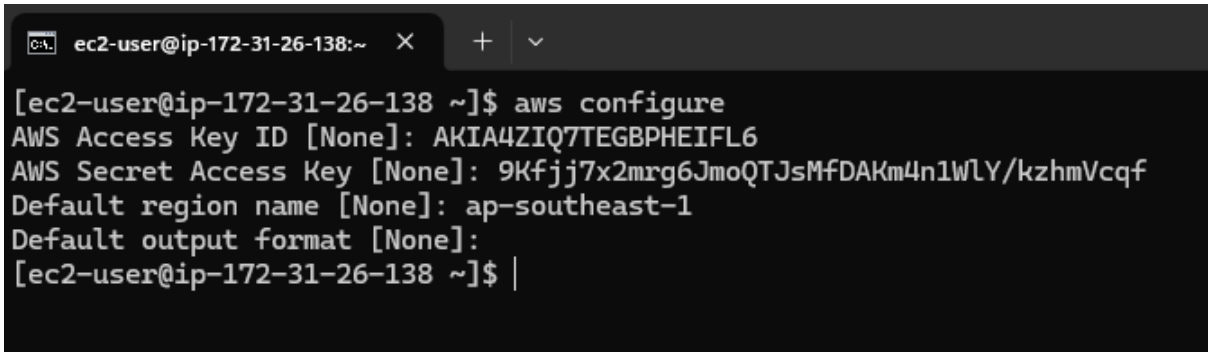
9. Below you can see that our user has been created. Now go to security credentials and create an access key and secret access key for CLI. Also, note the access key in a notepad or download them.



10. Now we are going to download the certificates using AWS CLI and an EC2 instance.
11. Go to EC2 and create an instance based on Amazon Linux 2.

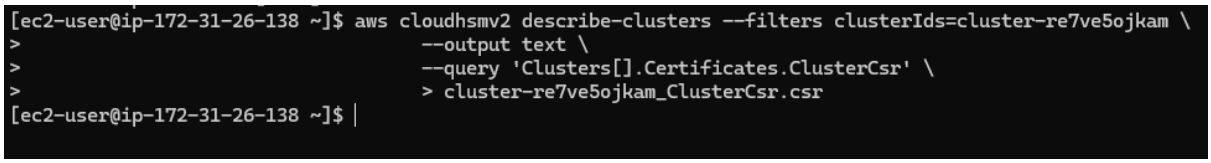


12. Once your instance is created then SSH into your instance. So, the first thing you need to do is configure your IAM hsmuser in the instance.

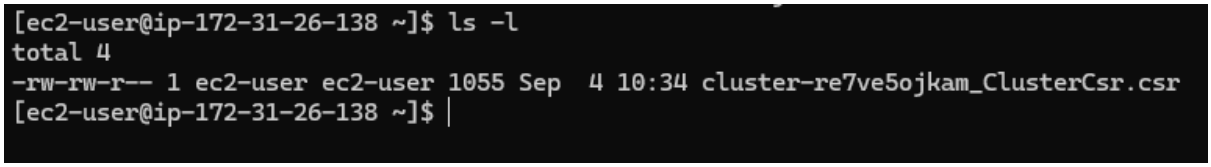


13. Then you need to run this command mentioned below to initialize the cluster.

    **aws cloudhsmv2 describe-clusters --filters clusterIds=<cluster ID> **
    **--output text **
    **--query 'Clusters[].Certificates.ClusterCsr' **
    > <cluster ID>_ClusterCsr.csr

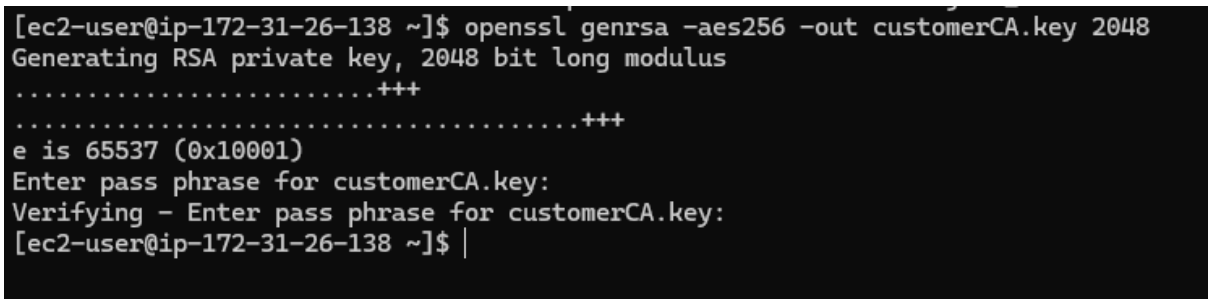


14. So, you can see that after running the above command we have downloaded the certificate which you can see below.



15. Next, we need to sign a certificate and for that, we need to provision a private key. For that, we need to run some commands. Below you can see that after running the commands it has asked us for the pass phrase. Note this pass phrase.

    openssl genrsa -aes256 -out customerCA.key 2048



16. We have created our certificate but we need to sign it also, for that run the command below. You can see in the snapshot that it asked you the pass phrase and then some basic information.

    openssl req -new -x509 -days 3652 -key customerCA.key -out customerCA.crt

```
[ec2-user@ip-172-31-26-138 ~]$ openssl req -new -x509 -days 3652 -key customerCA.key -out customerCA.crt
Enter pass phrase for customerCA.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:IN
State or Province Name (full name) []:DEL
Locality Name (eg, city) [Default City]:DEL
Organization Name (eg, company) [Default Company Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:
Email Address []:
[ec2-user@ip-172-31-26-138 ~]$
```

17. Now if you do a listing of certificates you can see all the certificates.

```
[ec2-user@ip-172-31-26-138 ~]$ ls -l
total 12
-rw-rw-r-- 1 ec2-user ec2-user 1055 Sep  4 10:34 cluster-re7ve5ojkam_ClusterCsr.csr
-rw-rw-r-- 1 ec2-user ec2-user 1233 Sep  4 10:42 customerCA.crt
-rw-rw-r-- 1 ec2-user ec2-user 1766 Sep  4 10:39 customerCA.key
[ec2-user@ip-172-31-26-138 ~]$
```

18. After that we need to run this command to sign the cluster certificate.

**openssl x509 -req -days 3652 -in <*cluster ID*>_ClusterCsr.csr **
**-CA customerCA.crt **
**-CAkey customerCA.key **
**-CAcreateserial **
-out <*cluster ID*>_CustomerHsmCertificate.crt

```
[ec2-user@ip-172-31-26-138 ~]$ openssl x509 -req -days 3652 -in cluster-re7ve5ojkam_ClusterCsr.csr \
>                               -CA customerCA.crt \
>                               -CAkey customerCA.key \
>                               -CAcreateserial \
>                               -out cluster-re7ve5ojkam_CustomerHsmCertificate.crt
Signature ok
subject=/C=US/ST=CA/O=Cavium/OU=N3FIPS/L=SanJose/CN=HSM:CED82B20CB33B2386BE880689CA25F:PARTN:15, for FIPS mode
Getting CA Private Key
Enter pass phrase for customerCA.key:
[ec2-user@ip-172-31-26-138 ~]$
```

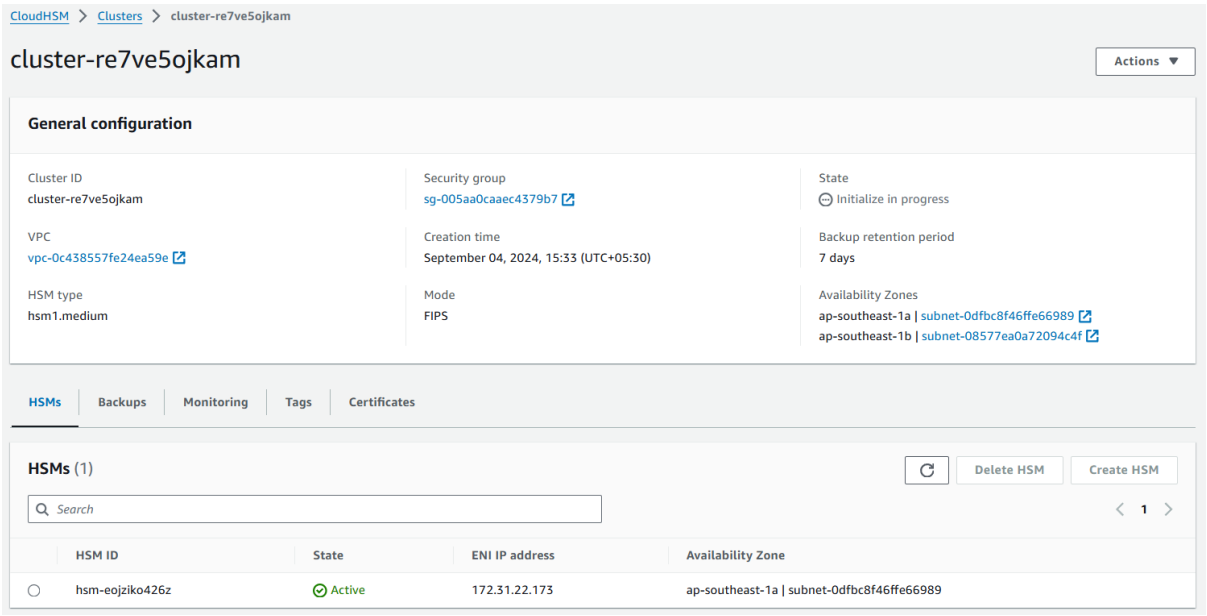
```
[ec2-user@ip-172-31-26-138 ~]$ ls -l
total 20
-rw-rw-r-- 1 ec2-user ec2-user 1055 Sep  4 10:34 cluster-re7ve5ojkam_ClusterCsr.csr
-rw-rw-r-- 1 ec2-user ec2-user 1208 Sep  4 10:45 cluster-re7ve5ojkam_CustomerHsmCertificate.crt
-rw-rw-r-- 1 ec2-user ec2-user 1233 Sep  4 10:42 customerCA.crt
-rw-rw-r-- 1 ec2-user ec2-user 1766 Sep  4 10:39 customerCA.key
-rw-rw-r-- 1 ec2-user ec2-user   17 Sep  4 10:45 customerCA.srl
[ec2-user@ip-172-31-26-138 ~]$
```

19. By using the command below, we will initialize our cluster.

**aws cloudhsmv2 initialize-cluster --cluster-id <*cluster ID*> **
**--signed-cert file://<*cluster ID*>_CustomerHsmCertificate.crt **
--trust-anchor file://customerCA.crt

```
[ec2-user@ip-172-31-26-138 ~]$ aws cloudhsmv2 initialize-cluster --cluster-id cluster-re7ve5ojkam \
>                                --signed-cert file://cluster-re7ve5ojkam_CustomerHsmCertificate.crt \
>                                --trust-anchor file://customerCA.crt
{
    "StateMessage": "Cluster is initializing. State will change to INITIALIZED upon completion.",
    "State": "INITIALIZE_IN_PROGRESS"
}
[ec2-user@ip-172-31-26-138 ~]$
```

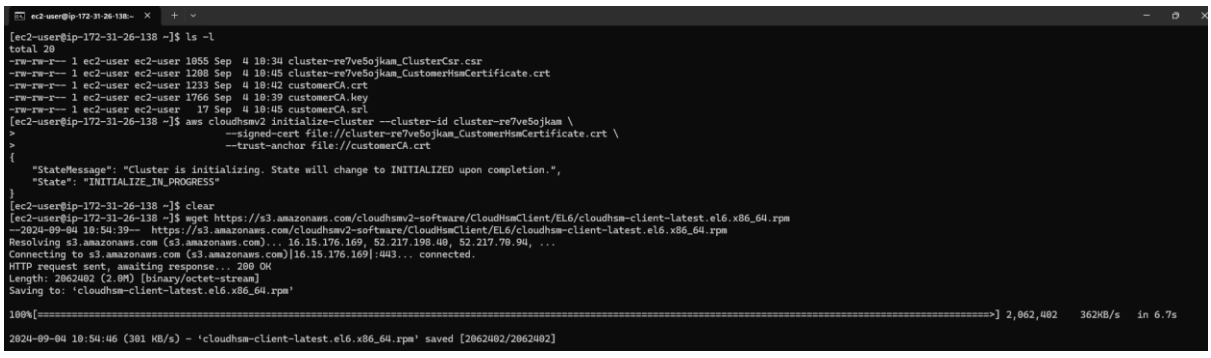
20. Now if you go to your CloudHSM cluster you will see that the initialize is in progress.



21. So, until now we have provisioned our cluster and initialized our cluster. Now we are going to activate our cluster.
22. First, we need to download and install AWS CloudHSM client on our EC2 instance using the below commands.

    wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-latest.el6.x86_64.rpm

    sudo yum install -y ./cloudhsm-client-latest.el6.x86_64.rpm

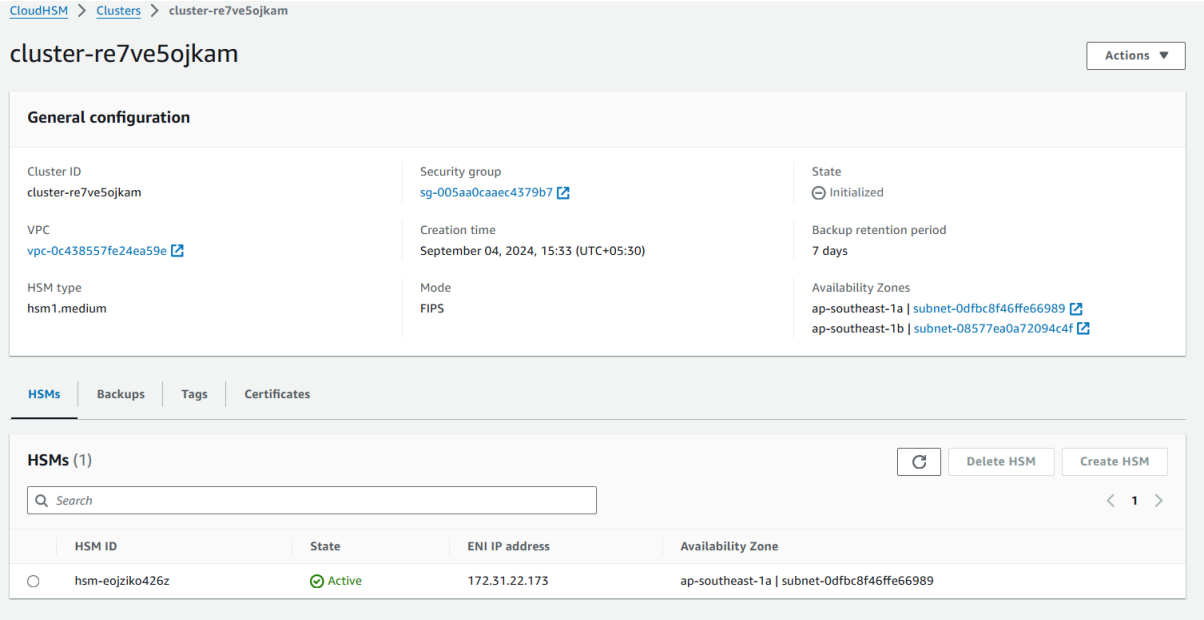


23. After the installation of the client, we need to edit the client configuration. For that, we need to switch to the root user.

sudo su
cp customerCA.crt /opt/cloudhsm/etc/customerCA.crt

```
[root@ip-172-31-26-138 ec2-user]# cp customerCA.crt /opt/cloudhsm/etc/customerCA.crt
[root@ip-172-31-26-138 ec2-user]# ls -l
total 2036
-rw-rw-r-- 1 ec2-user ec2-user 2062402 Jan  4  2022 cloudhsm-client-latest.el6.x86_64.rpm
-rw-rw-r-- 1 ec2-user ec2-user    1055 Sep  4 10:34 cluster-re7ve5ojkam_ClusterCsr.csr
-rw-rw-r-- 1 ec2-user ec2-user    1208 Sep  4 10:45 cluster-re7ve5ojkam_CustomerHsmCertificate.crt
-rw-rw-r-- 1 ec2-user ec2-user    1233 Sep  4 10:42 customerCA.crt
-rw-rw-r-- 1 ec2-user ec2-user    1766 Sep  4 10:39 customerCA.key
-rw-rw-r-- 1 ec2-user ec2-user      17 Sep  4 10:45 customerCA.srl
[root@ip-172-31-26-138 ec2-user]# cd /opt/cloudhsm/etc/
[root@ip-172-31-26-138 etc]# ls -l
total 20
drwxr-xr-x 2 root root   42 Sep  4 10:55 certs
-rwxr-xr-x 1 root root 1342 Dec 30  2021 client.crt
-rwxr-xr-x 1 root root 1704 Dec 30  2021 client.key
-rw-r--r-- 1 root root 1179 Dec 30  2021 cloudhsm_client.cfg
-rw-r--r-- 1 root root  600 Dec 30  2021 cloudhsm_mgmt_util.cfg
-rw-r--r-- 1 root root 1233 Sep  4 11:00 customerCA.crt
[root@ip-172-31-26-138 etc]#
```

24. Now modify the configuration files for the AWS CloudHSM client to update HSM's IP address.
25. So, go to CloudHSM and get the ENI IP address. Also, you will see that your cluster has been initialized.



26. Run the below command and you will see that the server is updating.

sudo /opt/cloudhsm/bin/configure -a YOURIP

```
[root@ip-172-31-26-138 etc]# sudo /opt/cloudhsm/bin/configure -a 172.31.22.173
Updating server config in /opt/cloudhsm/etc/cloudhsm_client.cfg
Updating server config in /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
[root@ip-172-31-26-138 etc]#
```

27. Now you need to log in HSM using the credentials of the precrypto officer (PRECO).

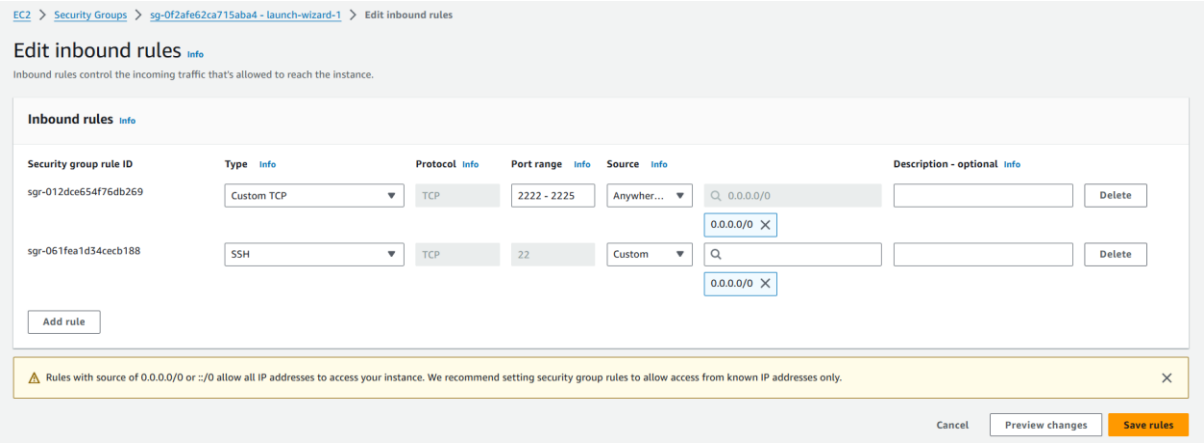
Note: The first HSM in a new cluster contains a PRECO user with a default username and password. When you change the password, the PRECO user becomes a crypto officer (CO).

28. You need to start the CloudHSM Management utility using the command below.

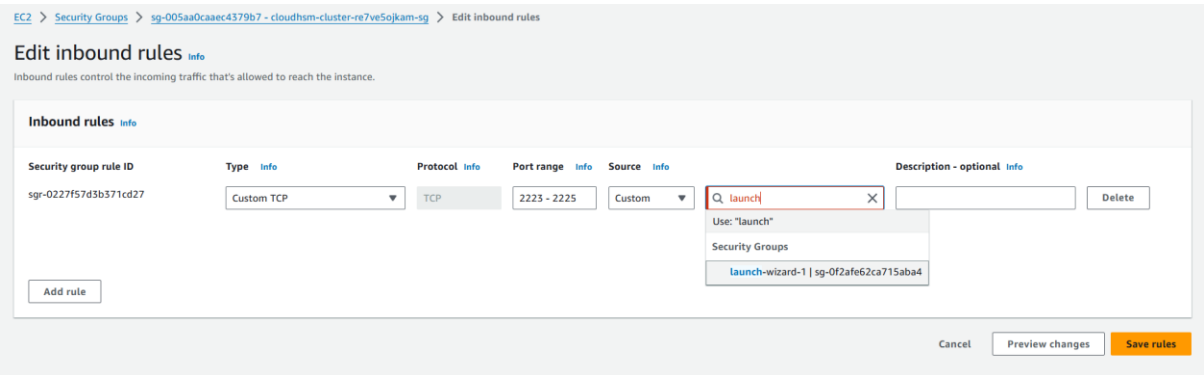
    /opt/cloudhsm/bin/cloudhsm_mgmt_util
    /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg

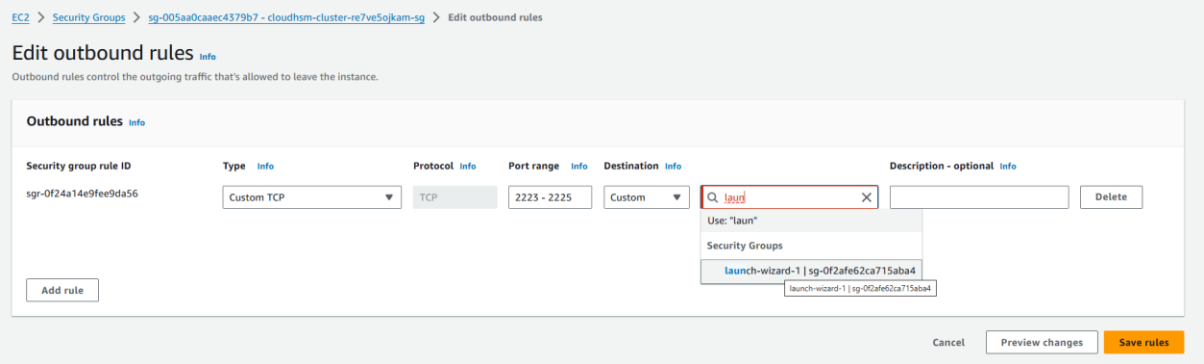
```
[root@ip-172-31-26-138 etc]# /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
Ignoring E2E enable flag in the configuration file

Connecting to the server(s), it may take time
depending on the server(s) load, please wait...

Connecting to server '172.31.22.173': hostname '172.31.22.173', port 2225...
```

29. But you can see that it wants to access port 2225 but we haven't provided this port to our security group of EC2. So, go to SG of our EC2 and add this port.



30. With that you also need to update the security group of your CloudHSM cluster. Go to CloudHSM, open the SG from there, and then go to inbound rules and click on edit. Here in the source, you must choose the security group of your EC2 instance. Then click on save.
31. You must do the same thing for outbound rules too.





32. Now run the command again and you will that our EC2 is communicating with CloudHSM over port 2225.

```
[root@ip-172-31-26-138 etc]# /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
Ignoring E2E enable flag in the configuration file

Connecting to the server(s), it may take time
depending on the server(s) load, please wait...

Connecting to server '172.31.22.173': hostname '172.31.22.173', port 2225...
Connected to server '172.31.22.173': hostname '172.31.22.173', port 2225.
E2E enabled on server 0(172.31.22.173)
aws-cloudhsm>
```

33. Use the below command to login as PRECO to change the password.

    loginHSM PRECO admin password
    listUsers

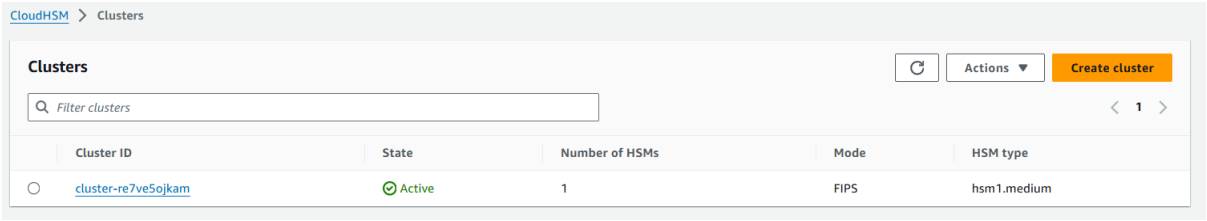
```
aws-cloudhsm>loginHSM PRECO admin password
loginHSM success on server 0(172.31.22.173)
aws-cloudhsm>listUsers
Users on server 0(172.31.22.173):
Number of users found:2

    User Id            User Type       User Name                      MofnPubKey    LoginFailureCnt         2FA
         1             PRECO           admin                              NO                  0          NO
         2             AU              app_user                           NO                  0          NO


aws-cloudhsm>
```

34. Now use the below command to change the password and you will see that the user type has been changed.

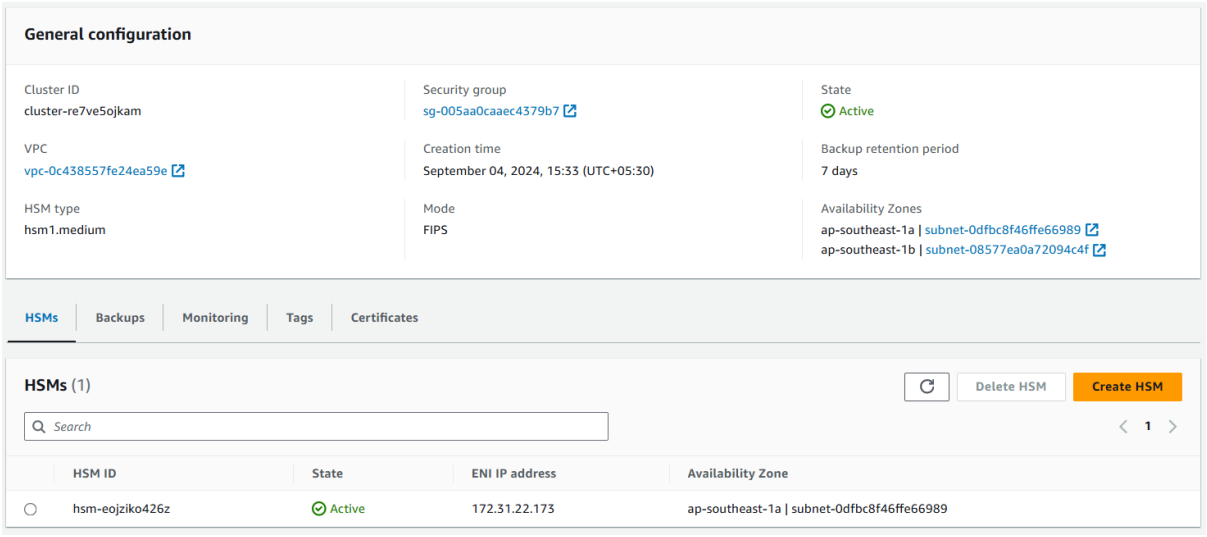
    changePswd PRECO admin NewPassword

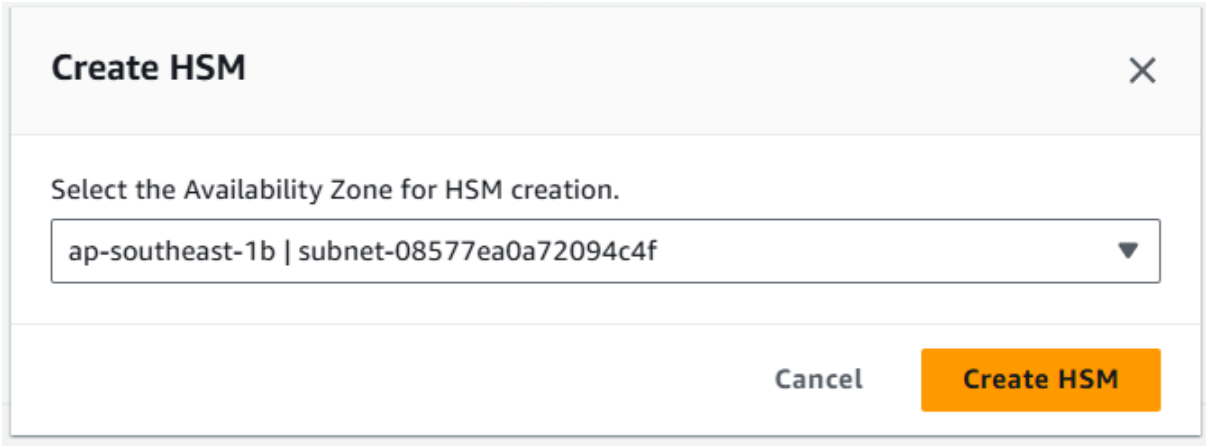
```
aws-cloudhsm>changePswd PRECO admin NewPassword
**********************CAUTION*****************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
**********************************************************

Do you want to continue(y/n)?y
Changing password for admin(PRECO) on 1 nodes
changePswd success on server 0(172.31.22.173)
aws-cloudhsm>listUsers
Users on server 0(172.31.22.173):
Number of users found:2

    User Id            User Type       User Name                      MofnPubKey    LoginFailureCnt         2FA
         1             CO              admin                          NO            0                       NO
         2             AU              app_user                       NO            0                       NO


aws-cloudhsm>
```

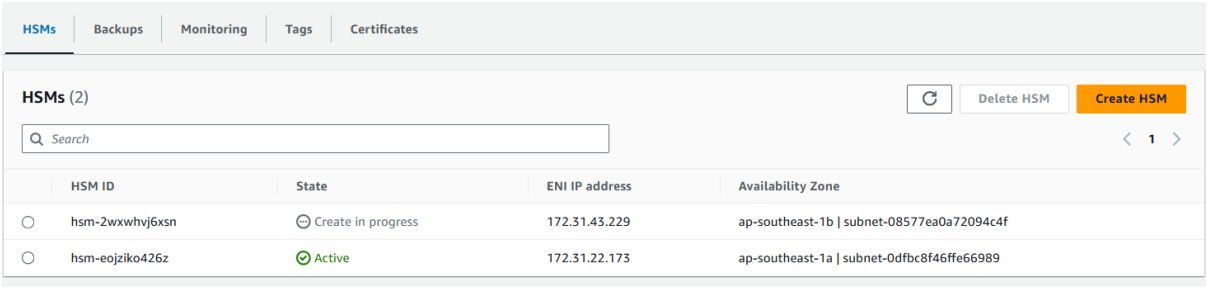
35. Now if you go to your CloudHSM cluster you will see that you cluster is now activated.



36. So, our cluster is active but only has one HSM, to complete this lab we need to create one more HSM in another availability zone.







37. Once you are done delete all the resources.