Sprint 3 Requirements: Enhanced Security with AWS Cloud HSM

Team Name: Data Defenders

Team Leader: Takala

Project Manager: Awa

Technical Lead: Jaelin

Technical Lead Support: Mariana

Technical Documentation:

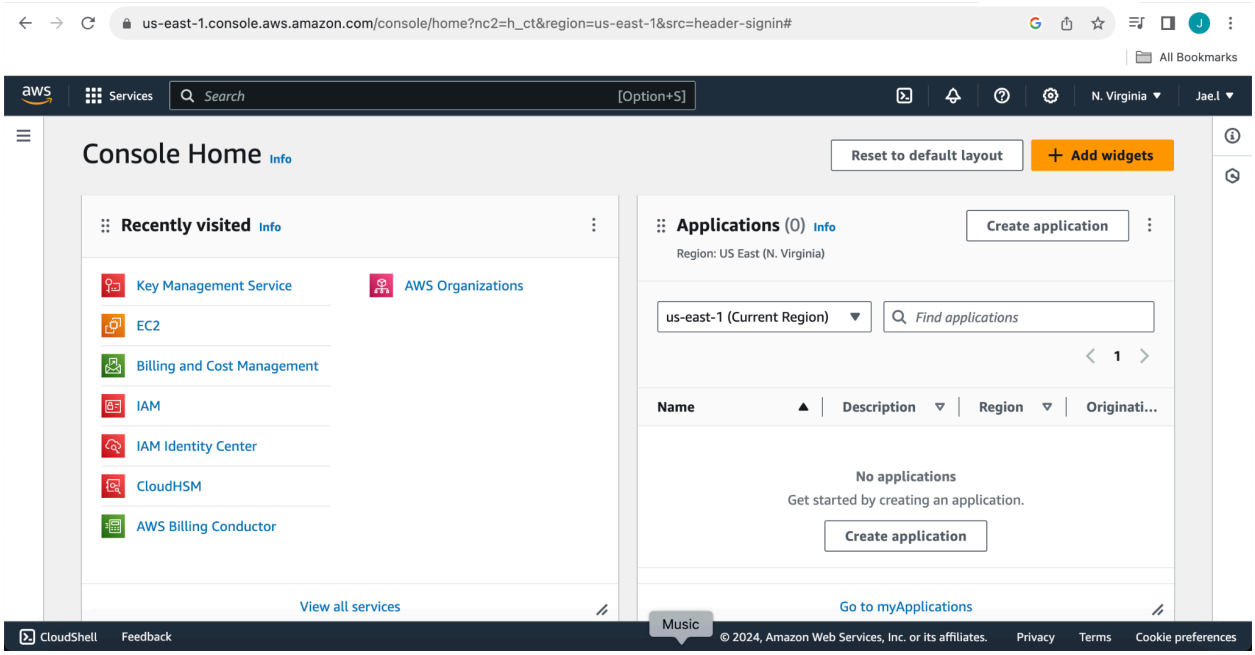
CloudHSM Deployment & Configuration

1. **Deploying an AWS CloudHSM Cluster:**
   - Choose appropriate CloudHSM instance type and configuration.
   - Create and initialize the CloudHSM cluster.
   - Configure networking and security groups.
   - Set up high availability and fault tolerance.

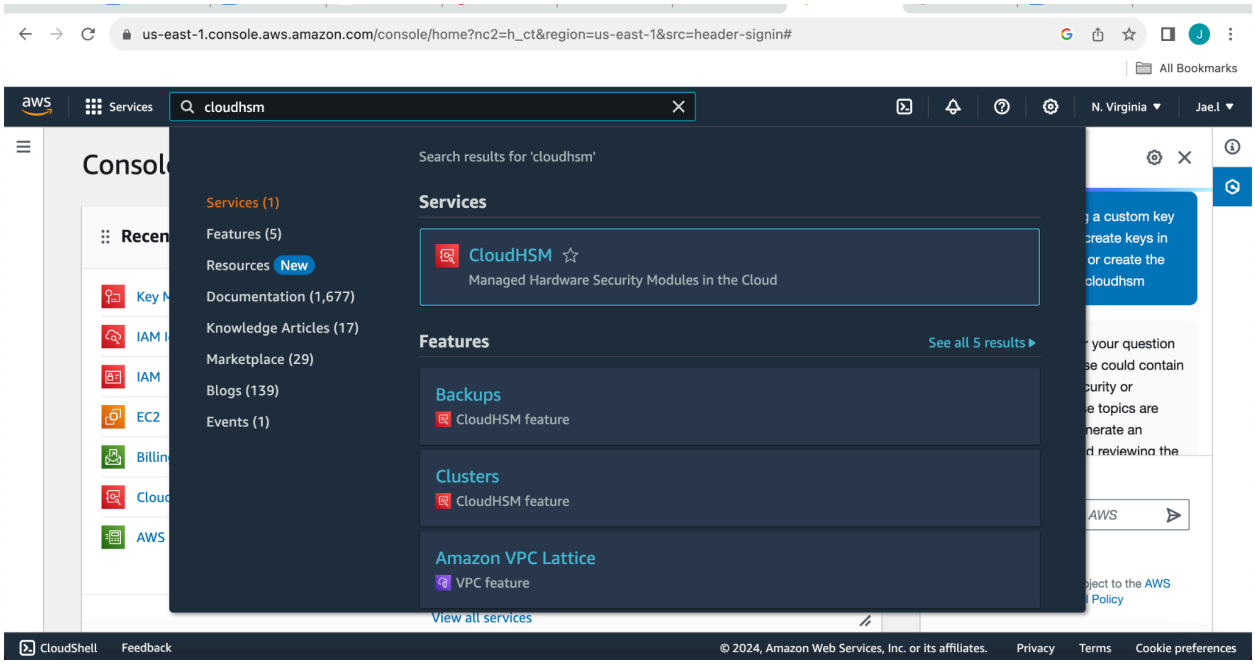
CloudHSM Cluster Deployment and Configuration

Creating a Cluster

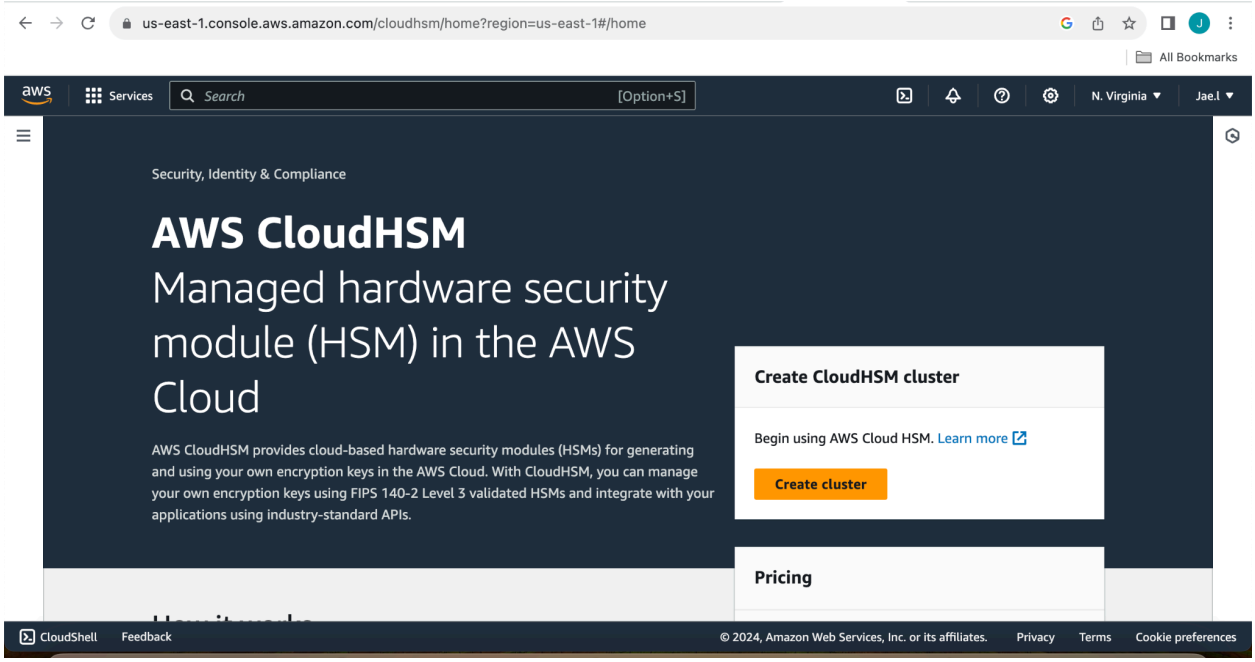
1. **Sign in to the AWS Management Console:** Go to the AWS Management Console and sign in to your AWS account. https://console.aws.amazon.com/



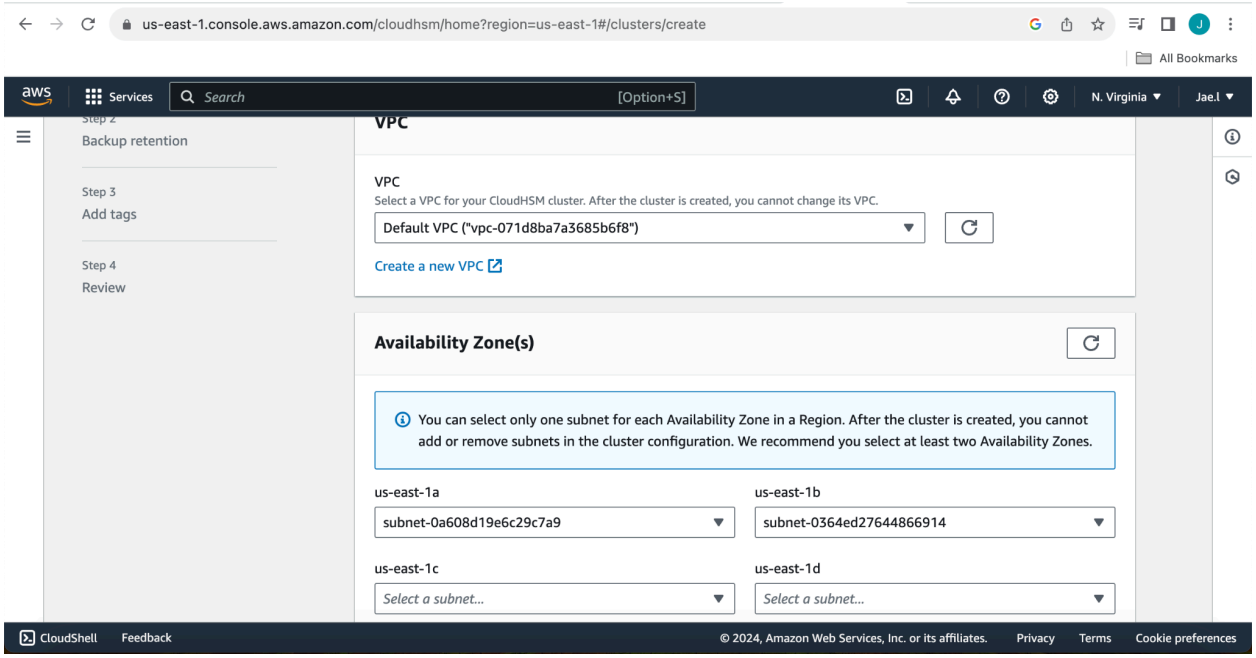
2. **Open CloudHSM in the Console:** Navigate to the CloudHSM service from the list of available AWS services.



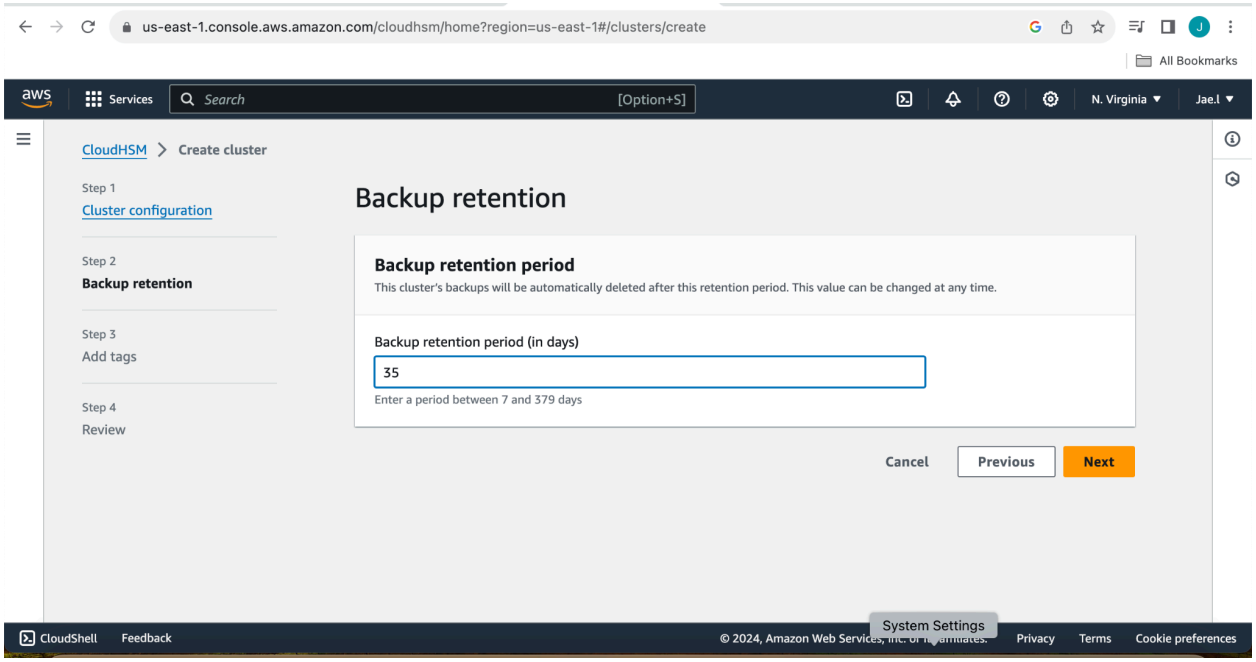
3. **Create a Cluster**: Click on "Create cluster" to start the process of creating a new cluster.



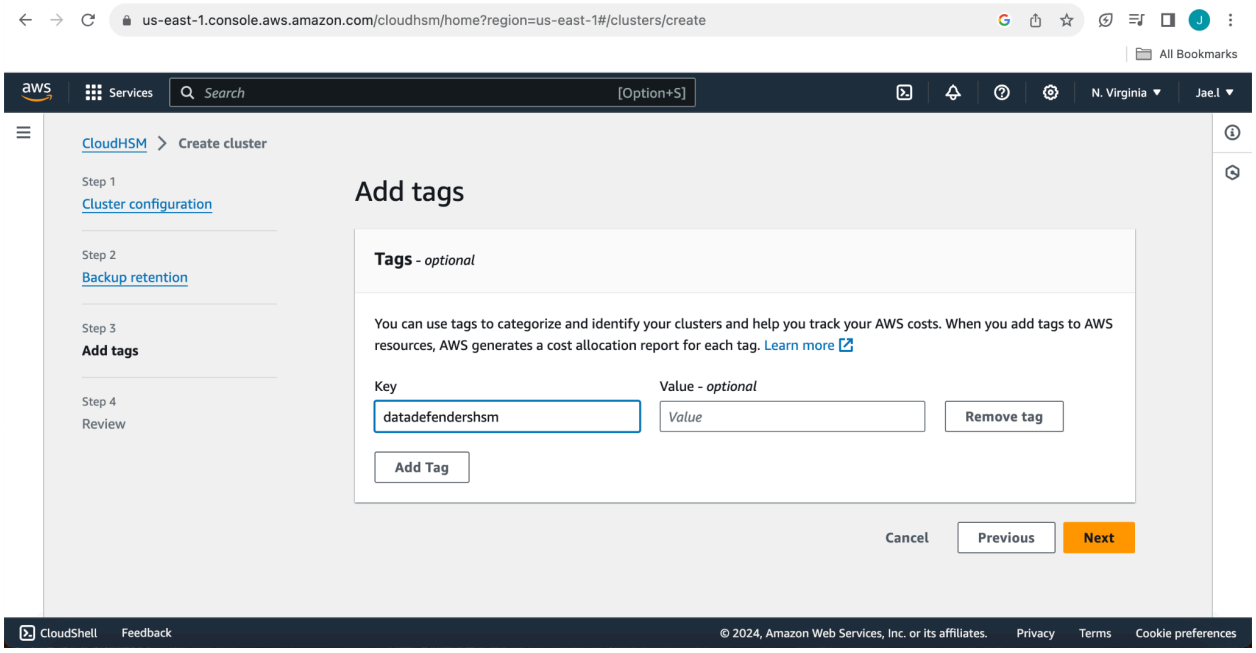
4. **Configure the cluster:**
   - Select the appropriate subnet and VPC (Virtual Private Cloud) for your HSMs.
   - Choose the AWS Region where you want to create the cluster.



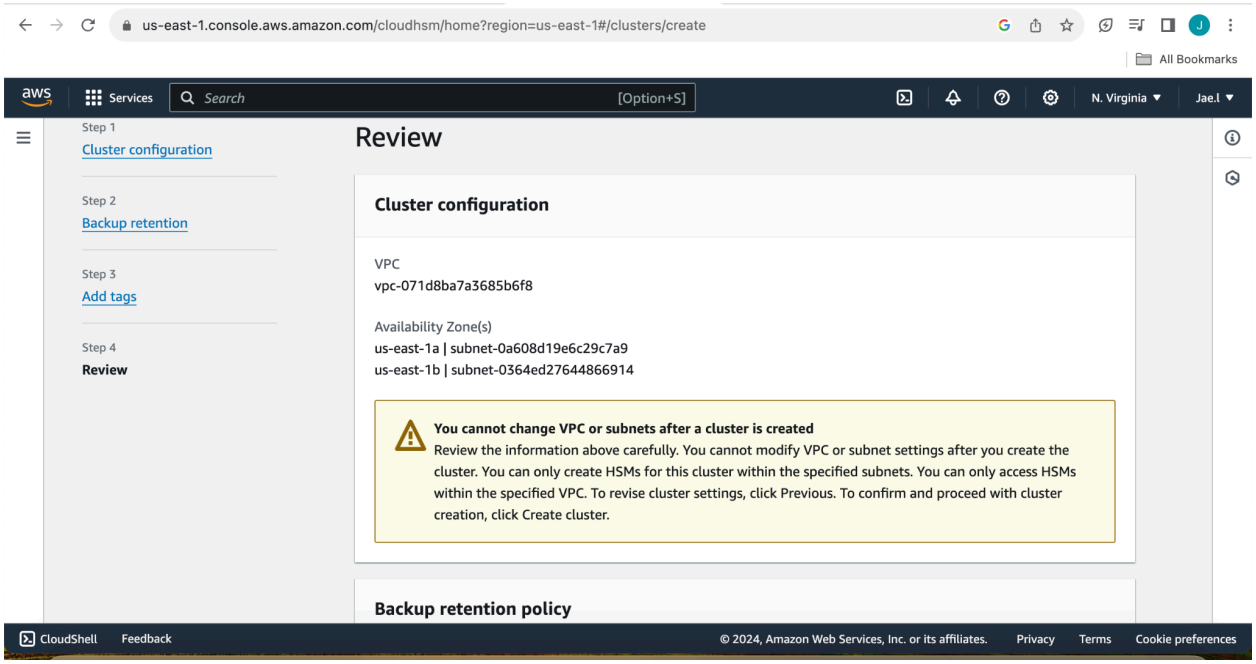
- Specify the "Cluster Backup Configuration" (Enable automatic backups or not).
- Specify the "Cluster Retention Period" : A 10 day backup retention period was selected



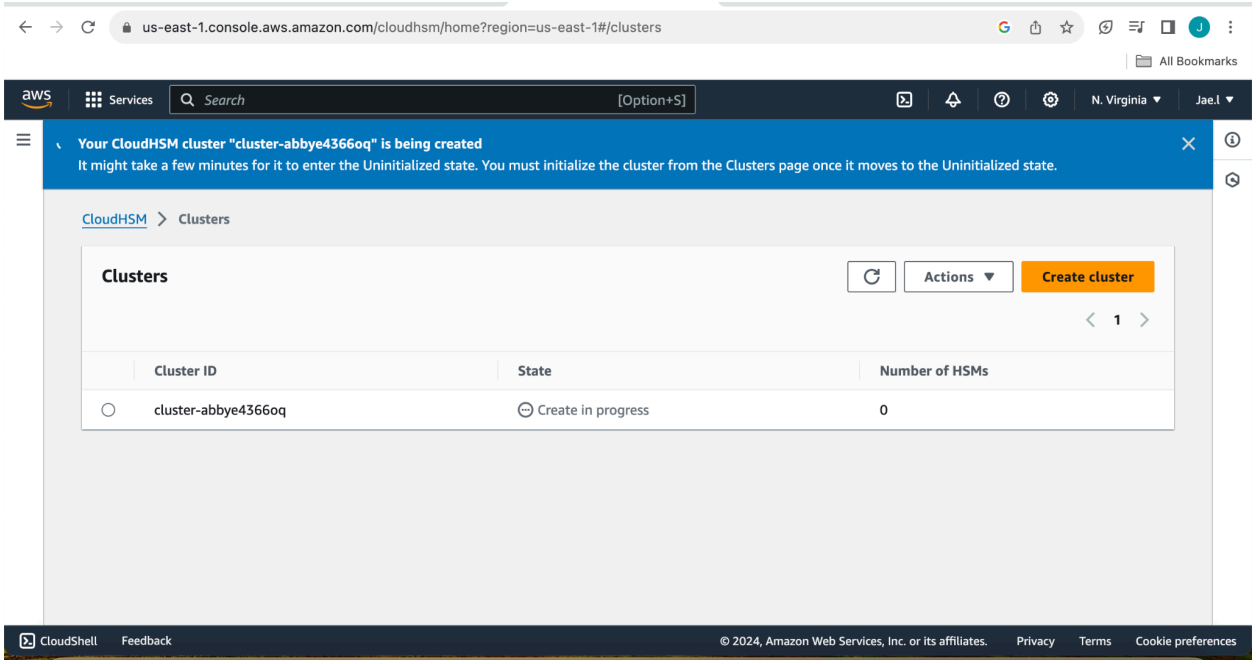
- Optionally, add tags to your cluster for easier identification.



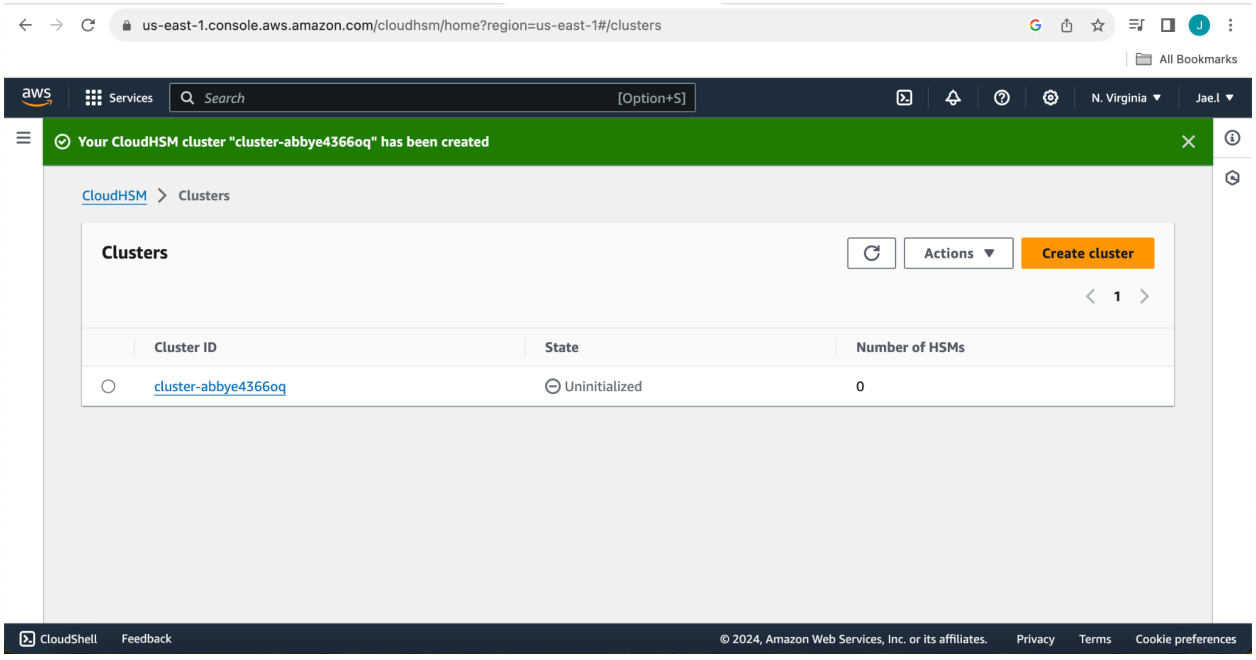
5. **Review and Create:**
    - Review the configuration details to ensure everything is correct.
    - If everything looks good, click on the "Create cluster" button to initiate the creation process.



6. **Wait for Cluster Creation:** AWS CloudHSM will now provision the resources and create the cluster. This process may take several minutes.

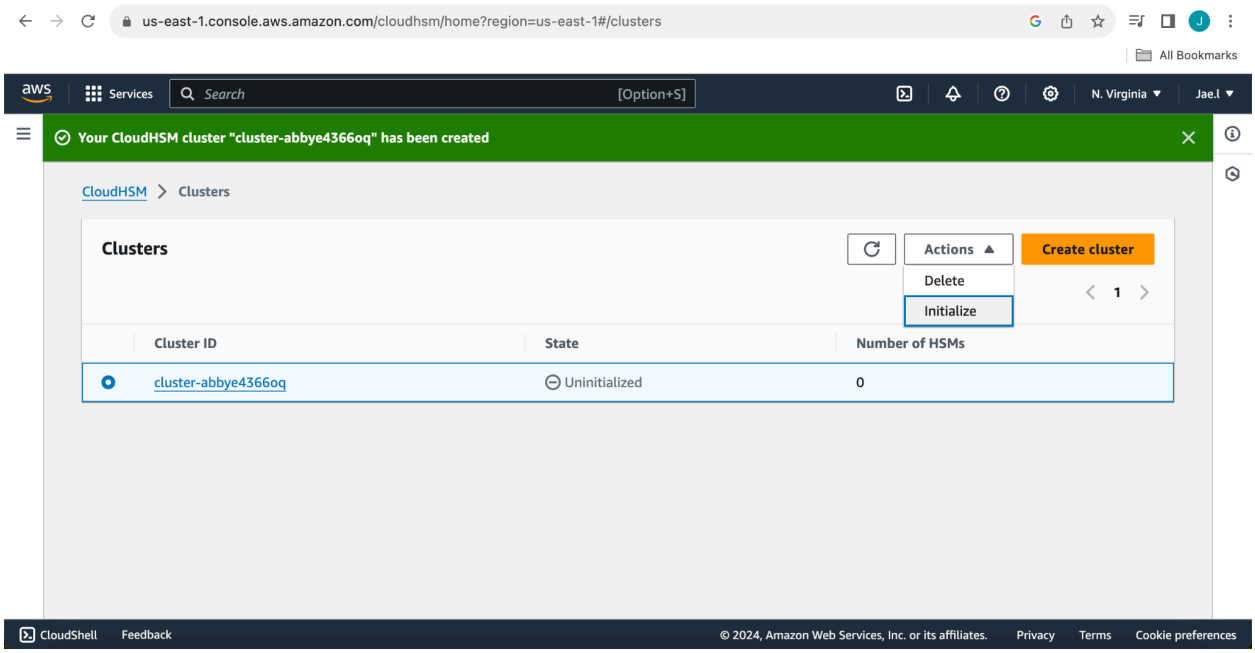


7. **Accessing the Cluster:** Once the cluster creation process is finished, you can access your HSMs using the CloudHSM client software or AWS SDKs/APIs.

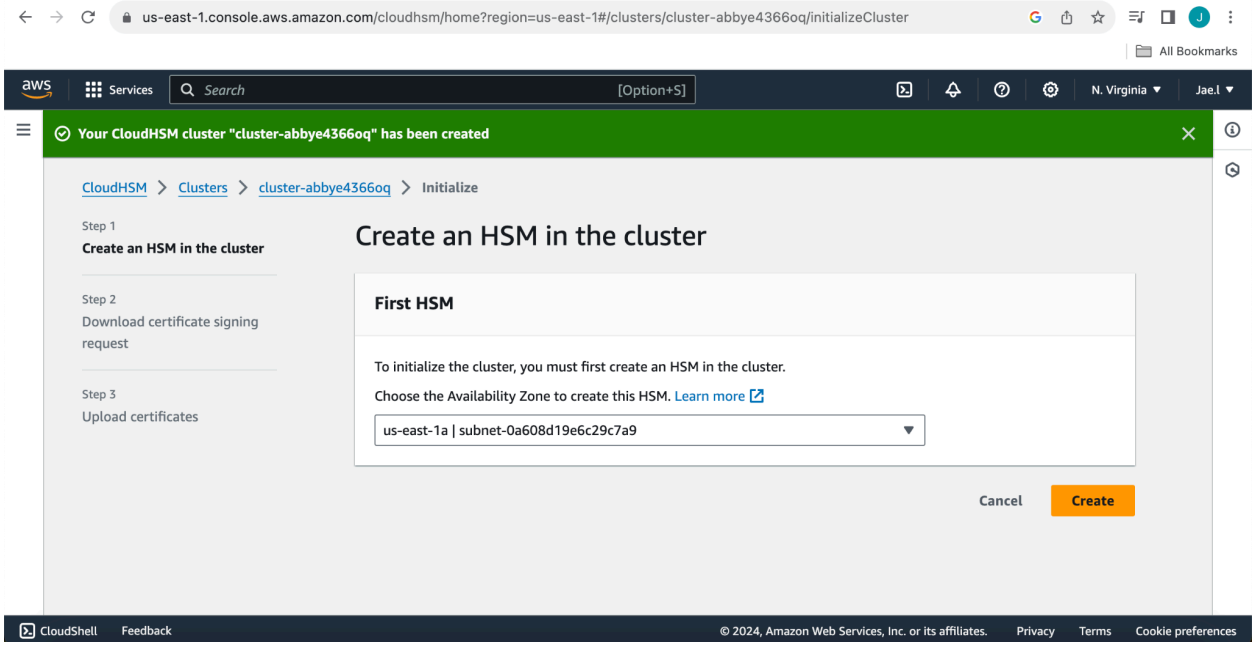


8. **Initialize HSMs**: Before you can start using your HSMs, you'll need to initialize them. This involves setting up the HSM administrator user and configuring other necessary settings.

- Select your cluster, select “initialize” from the actions menu



- Select your availability zone to provision your first HSM in the cluster



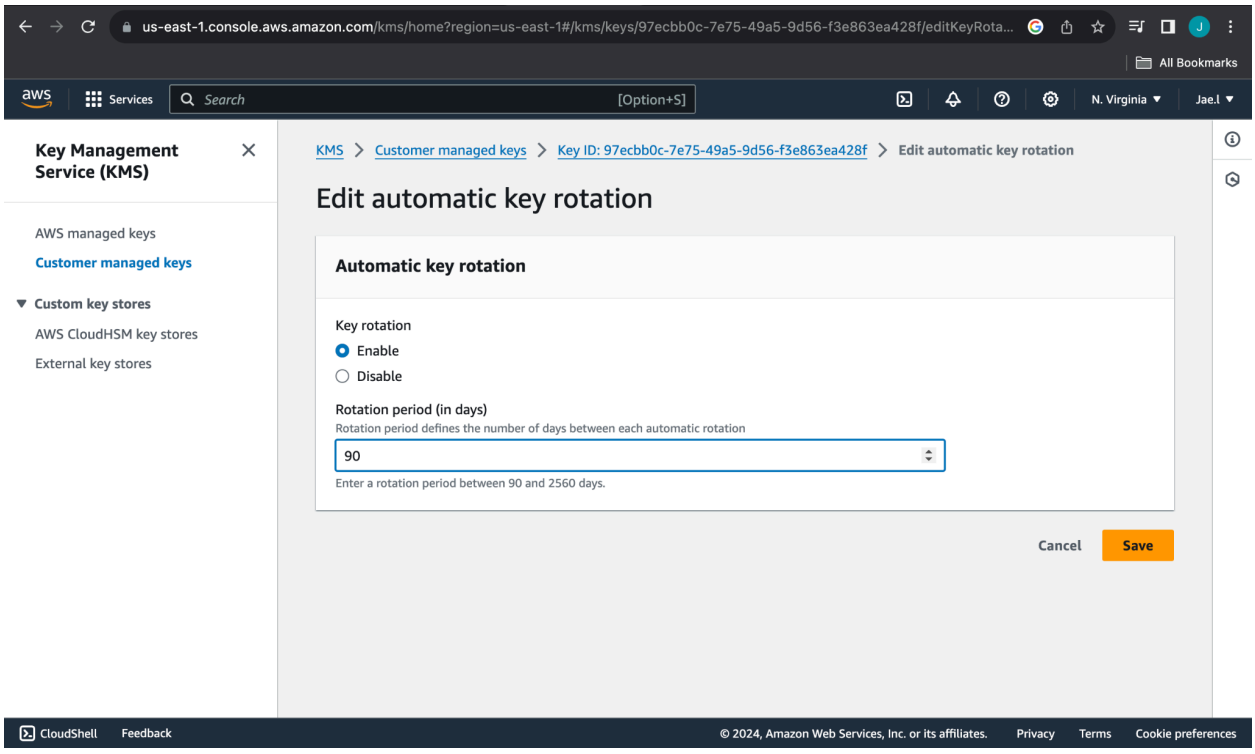
6. **Deploy Client Software:** Install the CloudHSM client software on the systems that need to interact with the HSMs. This software allows applications to communicate securely with the HSM cluster.

2. **Configuring CloudHSM**:
   - Partition and manage CloudHSM partitions.
   - Create and manage HSM users and roles.
   - Configure backup and restore procedures.
   - Explain key concepts: secure partition initialization, cryptographic officers, quorum authentication.
   - Discuss best practices for securing and hardening CloudHSM.
   - Provide examples of integrating CloudHSM with applications for cryptographic operations.

KMS Integration with CloudHSM

1. **Integrating AWS KMS with CloudHSM:**
   - Create a custom key store in KMS linked to your CloudHSM cluster.

- **Insert rotation period days and save**



- Generate and manage KMS customer master keys (CMKs) in CloudHSM.
- Configure KMS to use CloudHSM for cryptographic operations.

:

Benefits of Using CloudHSM:

- **Enhanced Security:** Dedicated hardware security modules (HSMs) ensure cryptographic keys never leave the hardware, safeguarding against unauthorized access.

- **Regulatory Compliance:** CloudHSM meets stringent compliance requirements like PCI DSS, HIPAA, and GDPR by offering tamper-resistant hardware and strict access controls.

- **Performance Improvements:** Offloading cryptographic operations to CloudHSM accelerates performance, reducing latency and enhancing application responsiveness.

Code examples for working with KMS keys integrated with Cloud HSM using AWS CLI commands:

1. Creating a Custom Key Store in KMS Linked to CloudHSM:
```bash
**aws kms create-custom-key-store --custom-key-store-name MyCustomKeyStore**
**--cloud-hsm-cluster-id <cloudHSM-cluster-id>**
```
2. Generating a KMS Customer Master Key (CMK) in CloudHSM:
```bash
**aws kms create-key --origin CLOUDHSM --custom-key-store-id <custom-key-store-id>**
```
3. Enabling Automatic Key Rotation for the CMK:
```bash
**aws kms enable-key-rotation --key-id <CMK-key-id>**
```
These commands demonstrate the creation of a custom key store in KMS linked to Cloud HSM, the generation of a CMK within Cloud HSM, and the enabling of automatic key rotation for the CMK to enhance security. Adjust the parameters accordingly to your AWS environment.

Key Considerations for Integration:

- Availability: Deploy cloud HSM clusters across multiple availability zones for fault tolerance.
- **Durability:** Regularly back up cloud HSM partitions to prevent data loss.
- Performance Implications: Optimize cloud HSM configurations based on application requirements to balance performance and cost.

Best Practices for Monitoring and Auditing KMS Key Usage:

1. **Enable AWS Cloud Trail Logging:** Log all KMS API calls to track key usage.
 Example:
  ```bash
  ```
2. **Set Up CloudWatch Alarms:** Monitor key usage metrics for anomalies.
Example:
  ```bash
  **aws cloudwatch put-metric-alarm --alarm-name kms-key-usage --metric-name KeyUsageCount --namespace AWS/KMS --statistic Sum --period 300 --threshold 100 --comparison-operator GreaterThanOrEqualToThreshold --evaluation-periods 1 --alarm-actions <SNS topic ARN>**
  ```
3. **Regularly Review Audit Logs:** Analyze CloudTrail logs for unauthorized access.
Example:
  ```bash
  **aws cloudtrail lookup-events --lookup-attributes AttributeKey=EventName,AttributeValue=Decrypt**
  ```
4. **Implement Least Privilege Access:** Restrict permissions to only necessary actions.
Example:
  ```bash
  **aws iam attach-role-policy --role-name <role-name> --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess**
  ```
5. **Encrypt CloudTrail Logs:** Encrypt logs using KMS-managed keys for data integrity.
Example:
  ```bash
  **aws cloudtrail update-trail --name my-kms-trail --kms-key-id <kms-key-id>**
  ```
6. **Enable Key Usage Tags**: Tag keys for categorization and tracking.
 Example:
  ```bash
  **aws kms tag-resource --key-id <key-id> --tags Key=Environment,Value=Production**
  ```
7. **Perform Regular Security Audits:** Review key usage, access policies, and configurations.
Example:
  ```bash
  **aws kms list-keys**
  ```
8. **Implement Multi-Factor Authentication (MFA):** Require MFA for critical operations.
Example:
  ```bash
  **aws kms enable-key-rotation --key-id <key-id> --multi-region**
  ```

```

By implementing these practices, you can ensure effective monitoring and auditing of KMS key usage, maintaining security and compliance standards.
```