Securing and Monitoring Resources with AWS

Presented by/

Mennatallah Abdelkareem Habashi Saleh St-ID:21046685

Supervised by/

DEPI Eng/Merihan Adel

Project overview and objectives:

In this project, I'm challenged to use familiar AWS services, to create resources in AWS and to implement security on them. I have used different AWS services and features to build a variety of solutions.

These specific sections of the project are meant to challenge me to practice skills that I have acquired throughout my learning experiences prior to this project.

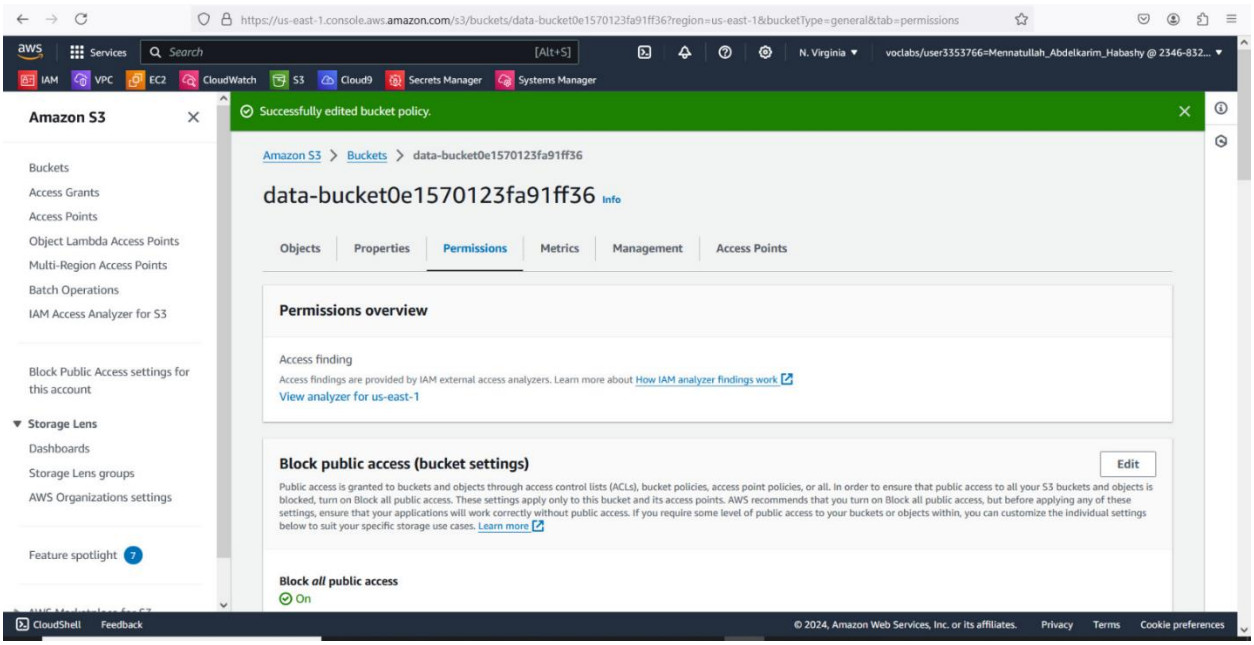
By the end of this project, I should be able to do the following:

- Secure access to objects in an Amazon Simple Storage Service (Amazon S3) bucket.
- Secure network access to my virtual network.
- Encrypt data at rest by using AWS Key Management Service (AWS KMS) on an Amazon Elastic Block Store (Amazon EBS) volume.
- Manage encryption keys by using AWS KMS.
- Create a monitoring and incident response system by using Amazon CloudWatch and AWS Config.

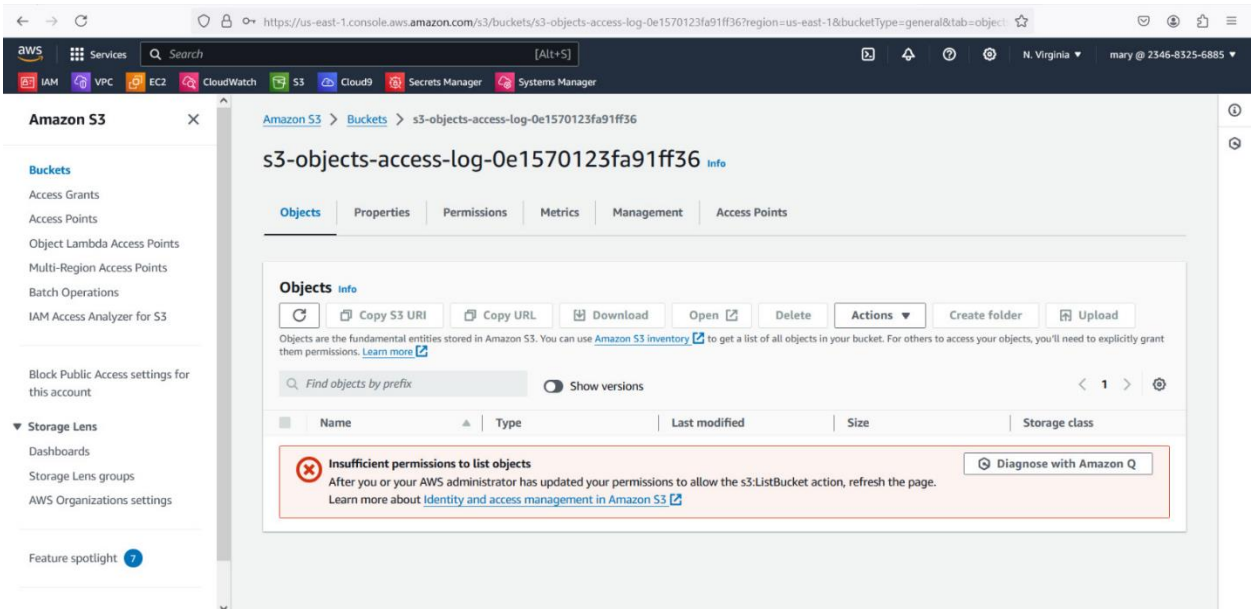
Phase 1: Securing data in Amazon S3:

In this phase, I begin implementing security settings in the AWS account. I have been asked to secure customer PII data that is stored in Amazon S3. The leadership team of Any Company Financial has heard about recent data breaches at other companies and wants to protect customer data from unauthorized access.

Task 1.1: Create a bucket, apply a bucket policy, and test access



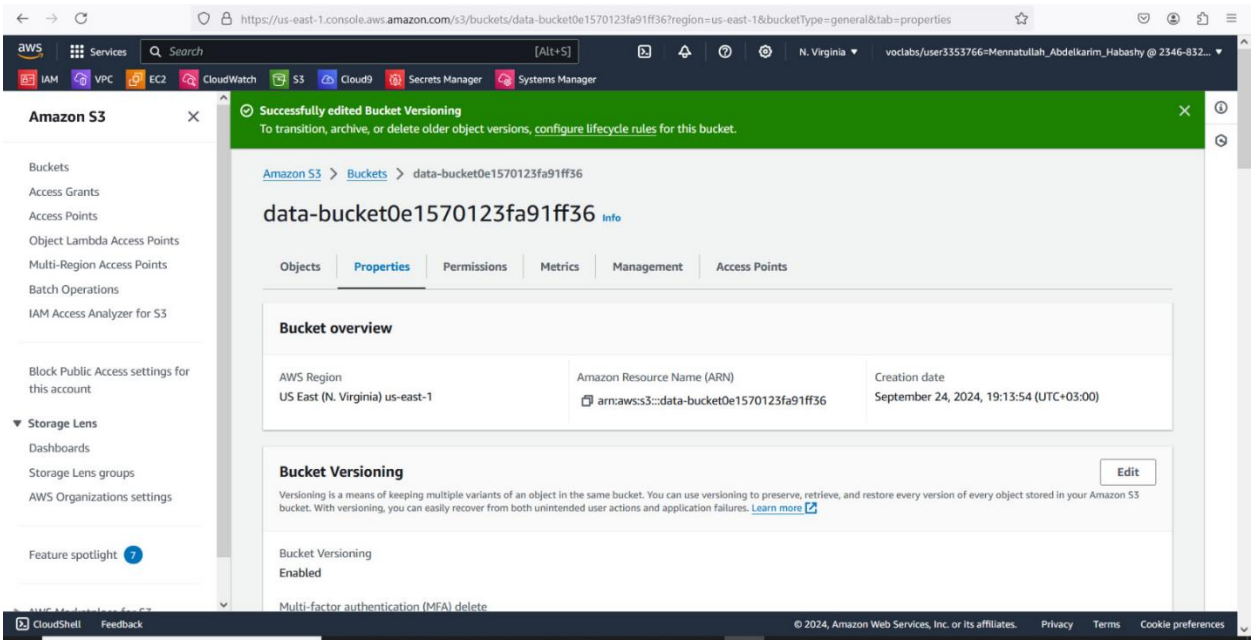
-*paulo* user's access to Amazon S3 but when Verify *Mary* user's access to Amazon S3, when she accesses any of the buckets, she sees an "Insufficient permissions to list objects" error , as the following:



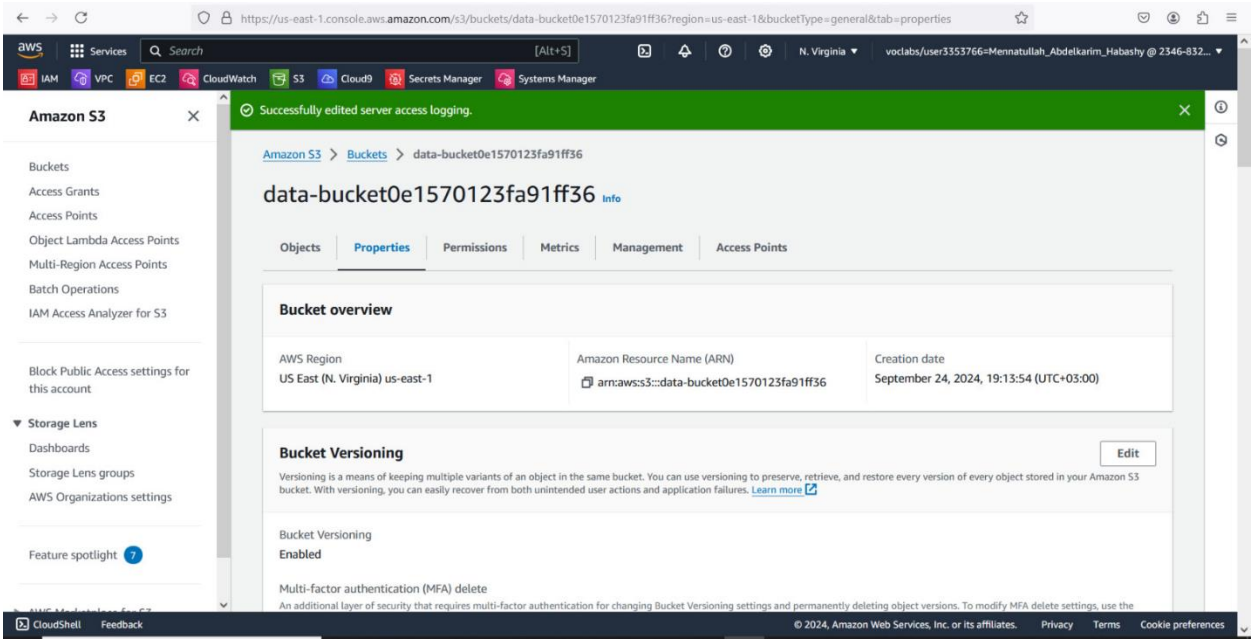
Task 1.2: Enable versioning and object-level logging on a bucket

In this task, I'm challenged to enable versioning and object-level logging on the *data-bucket*. With versioning enabled, I can track all changes to objects that are stored in the bucket and revert any object to a previous version if needed. Object-level logging creates a detailed audit trail of the objects that are stored in a bucket, which helps me to detect security incidents quickly.

-Enable versioning on a bucket:

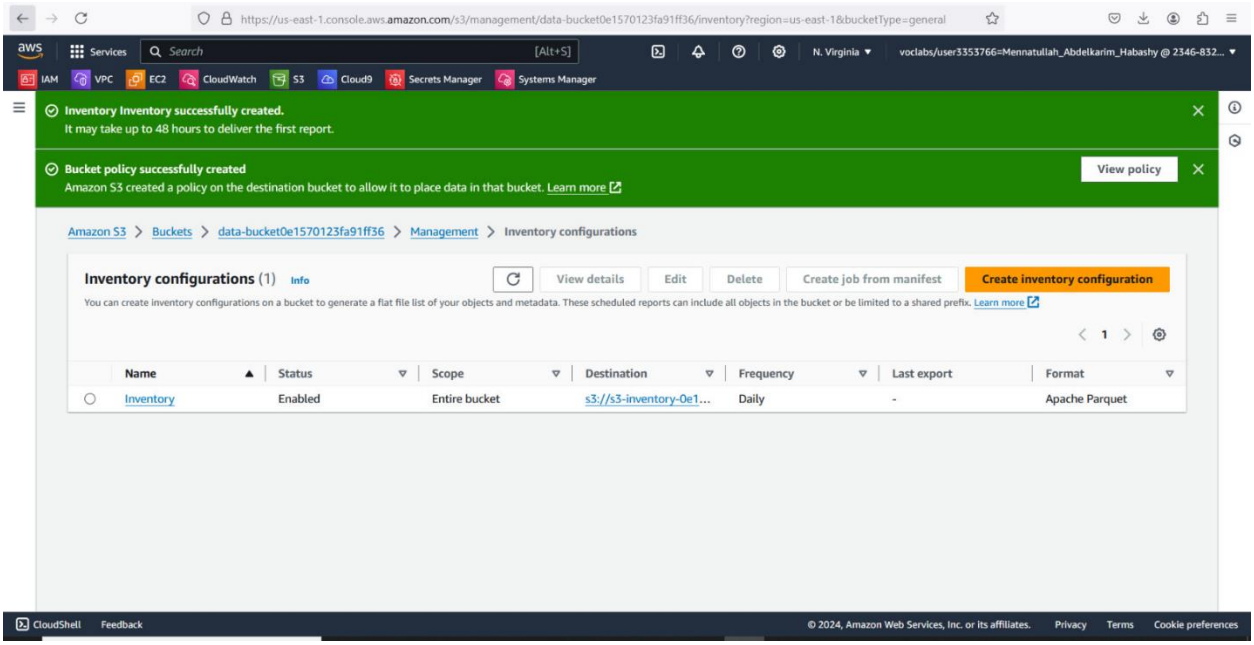


-Enable object-level logging on a bucket:



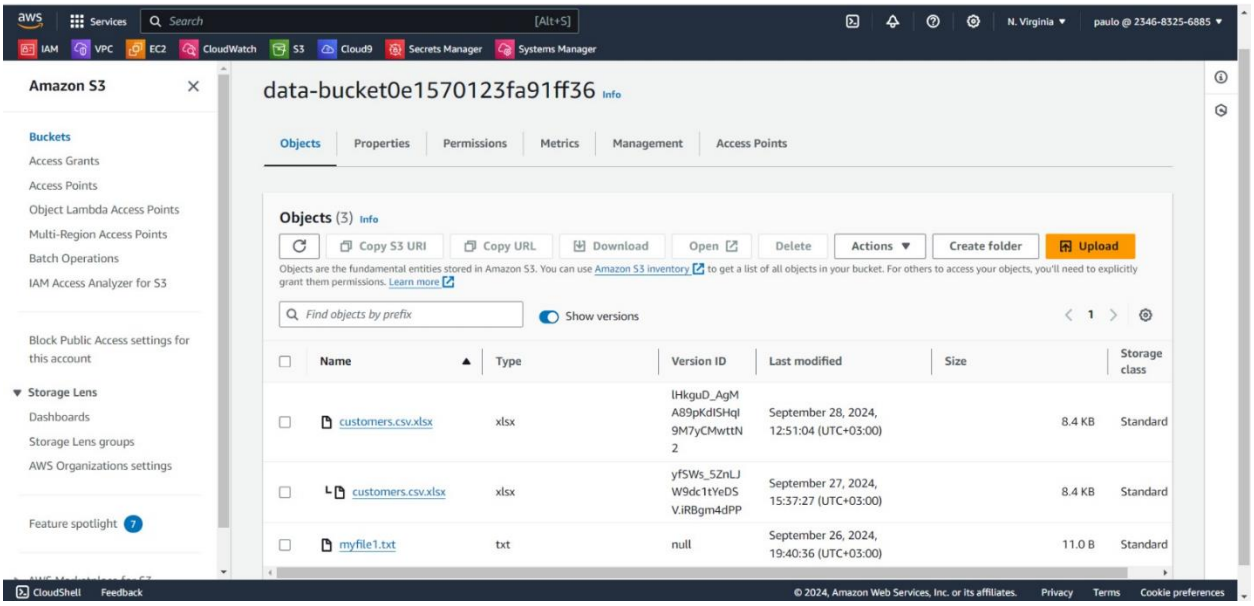
Task 1.3: Implement the S3 Inventory feature on a bucket

In this task, I'm challenged to enable the S3 Inventory feature to monitor changes to objects that are stored in an S3 bucket. S3 Inventory provides a scheduled report of the metadata and object-level changes to my S3 objects and buckets. By using the feature, I can track changes to the stored objects and detect potential security incidents.

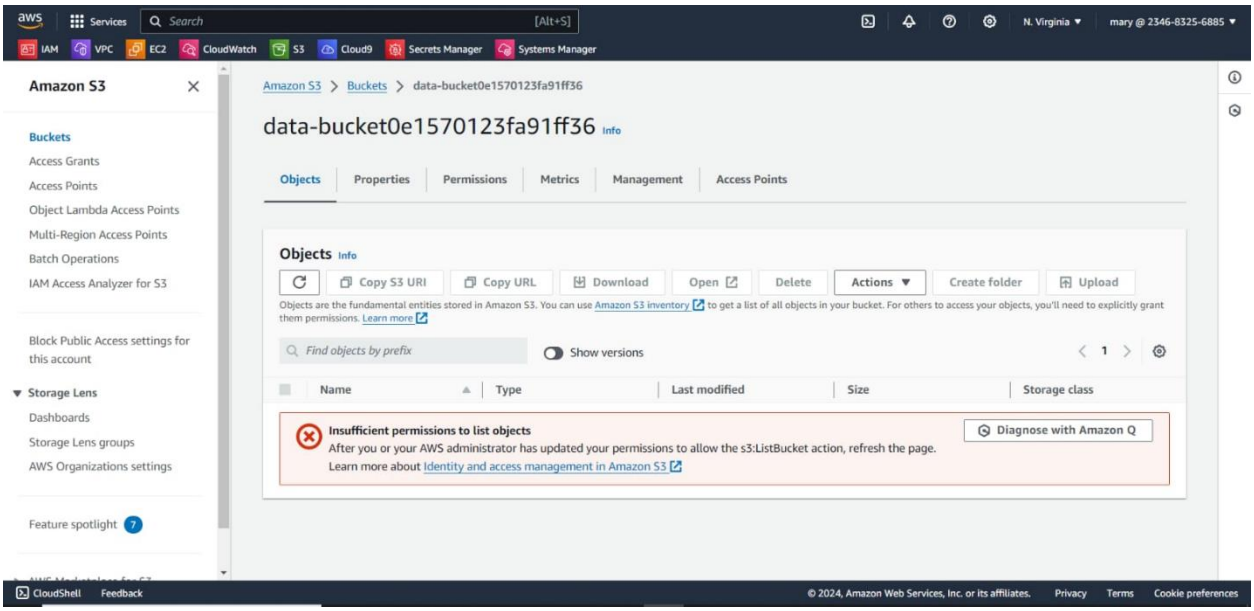


Task 1.4: Confirm that versioning works as intended

In this task, I'm challenged to access the AWS account as the *paulo* user and upload an object to the *data-bucket*. Then, you are challenged to confirm that versioning is enabled on the object. You are also challenged to test access as the *mary* user.



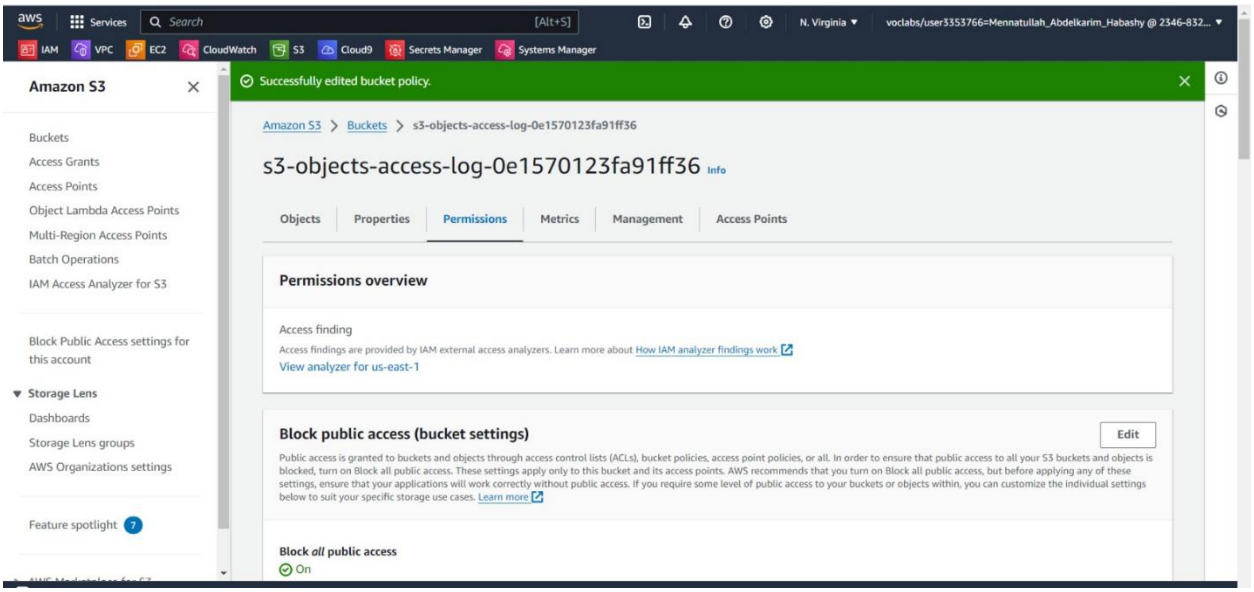
- Log in as the *mary* user and take action in the Amazon S3 console to generate log events After test versioning , notice that the *mary* user cannot access this bucket or its contents.



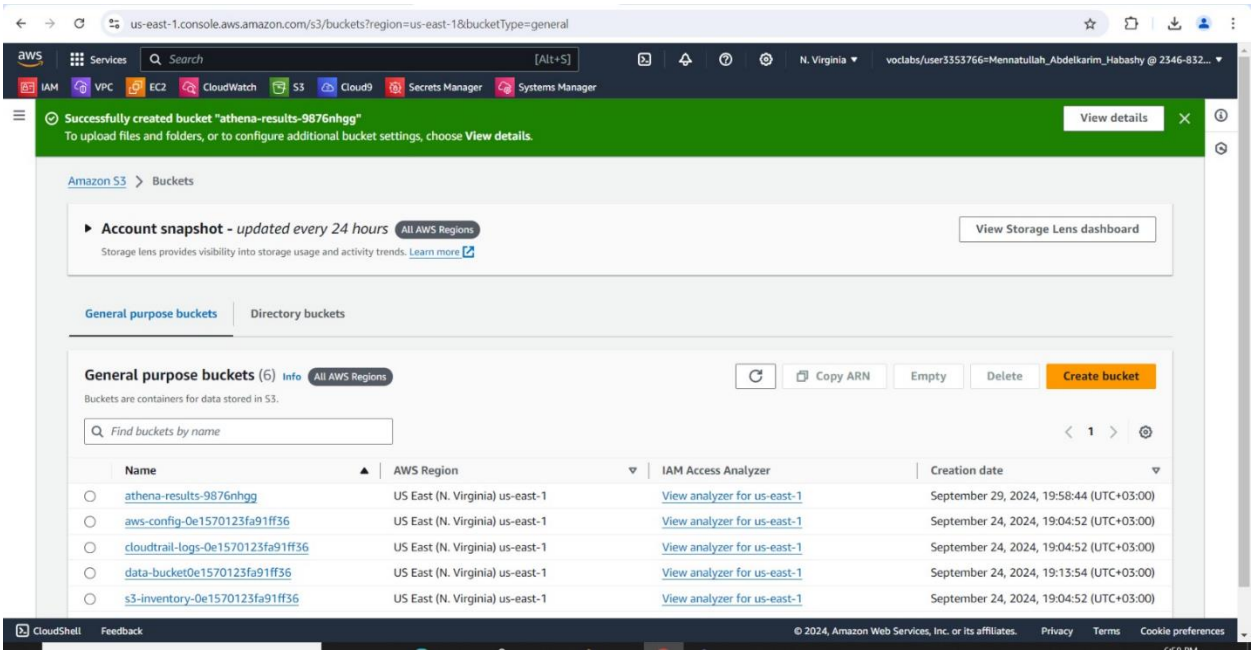
Task 1.5: Confirm object-level logging and query the access logs by using Athena

In this task, I'm challenged to confirm the S3 object-level logging I enabled earlier is successfully writing log data to S3. I will also use Athena to query these logs.

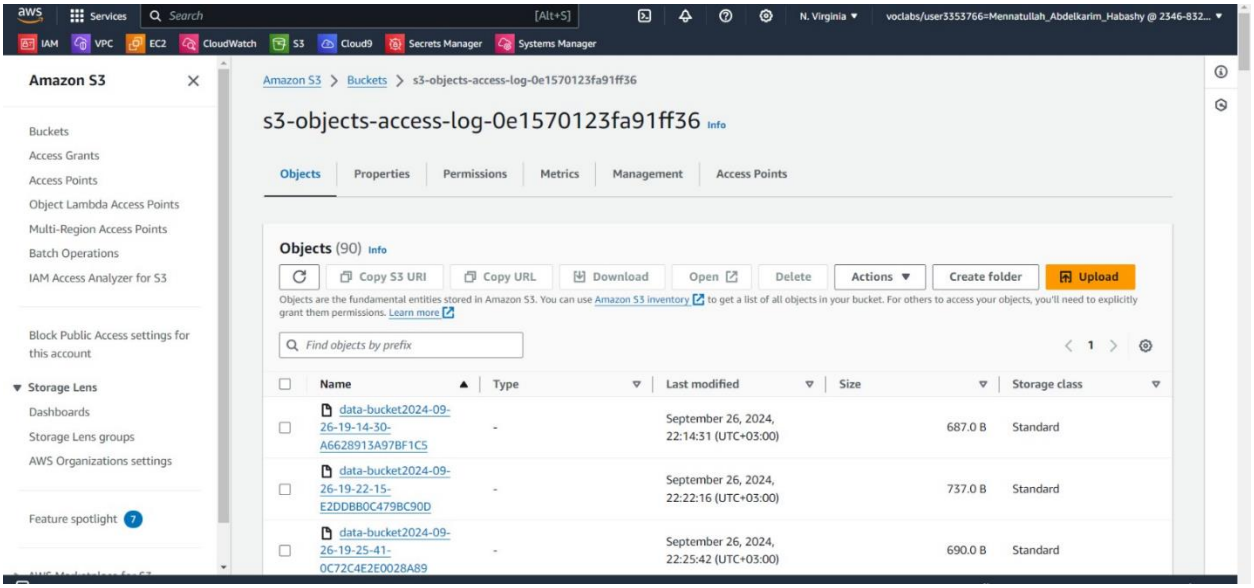
- Create s3 bucket to observe the objects that are stored in the *s3-objects-access-log* bucket:



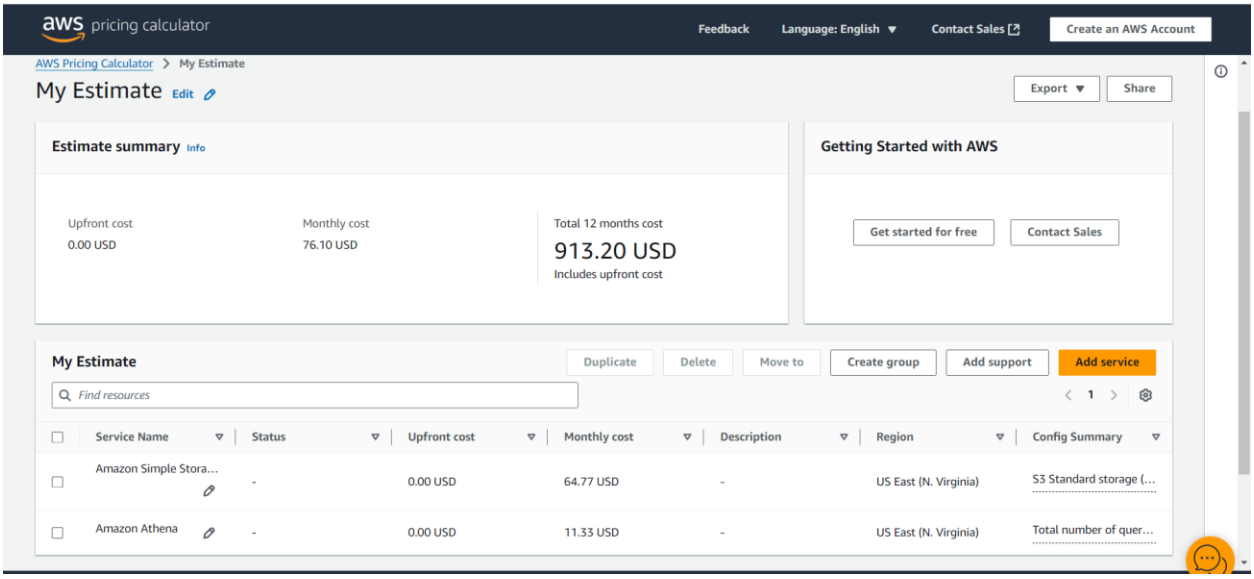
- Create an Athena table from the access logs:

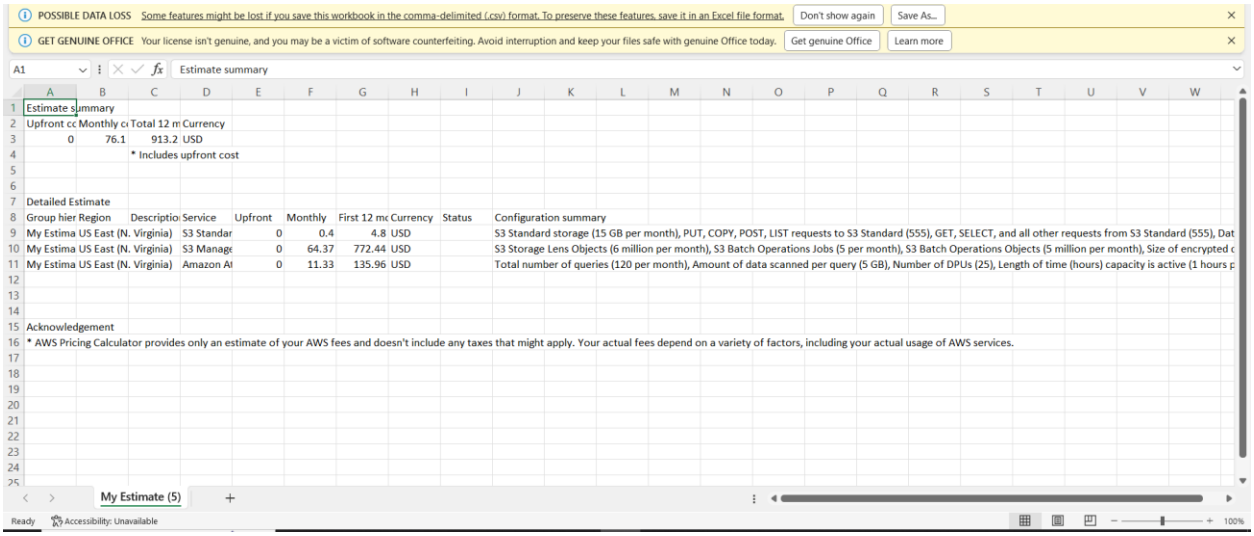


-Query the table to discover access log details:



-Cost assessment to secure Amazon S3:



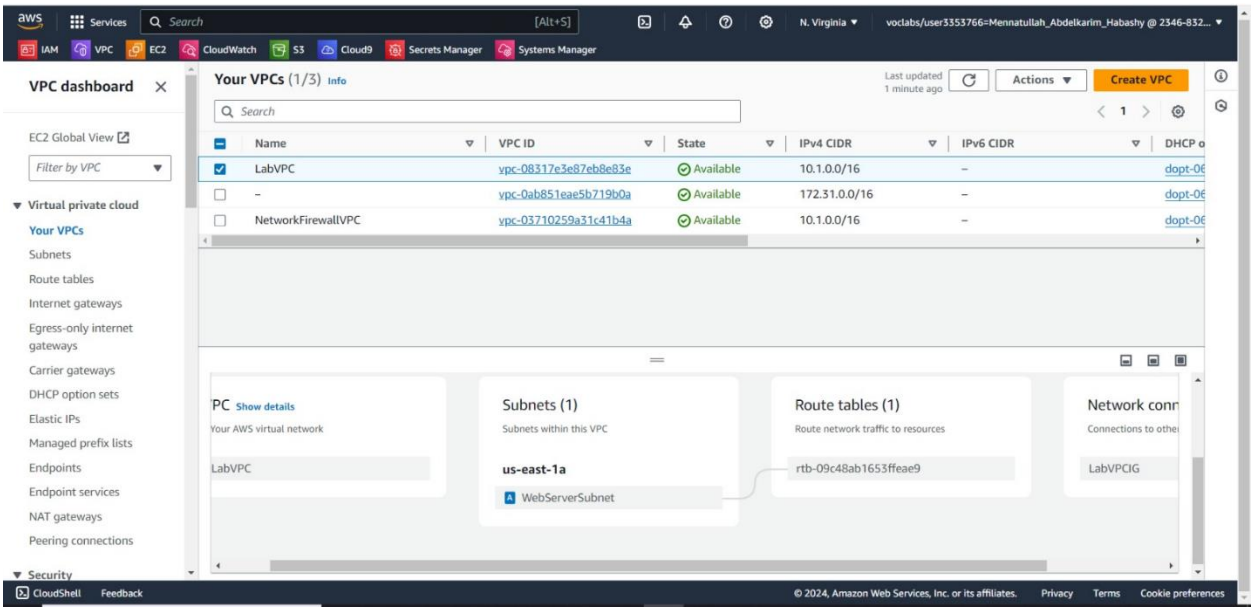


Phase 2: Securing VPCs:

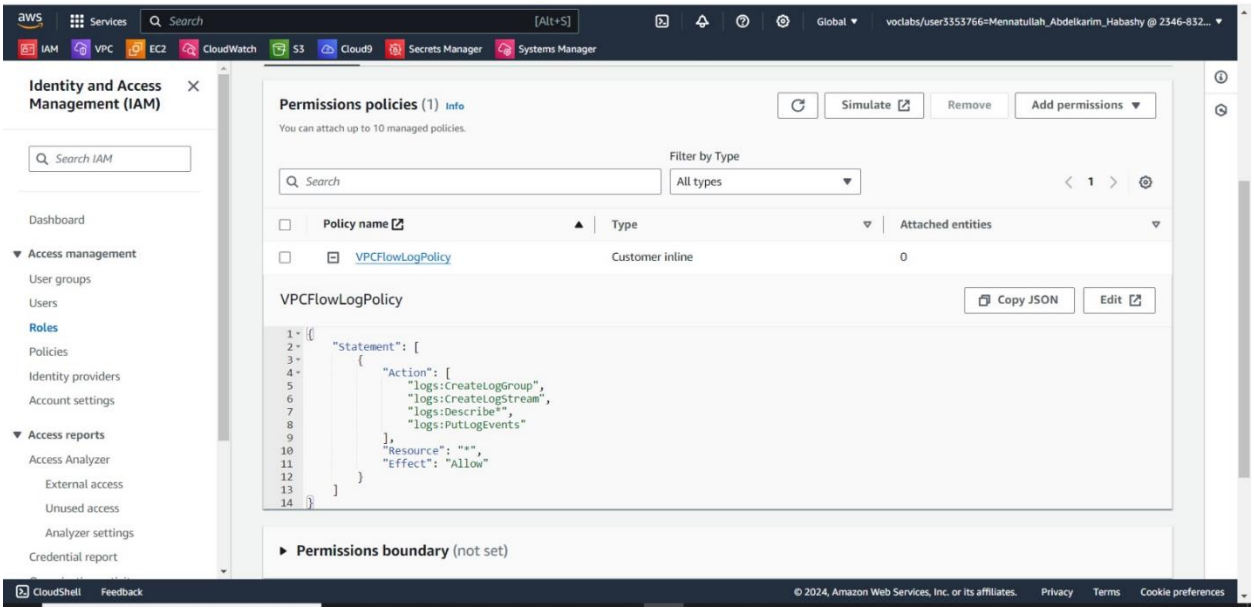
After securing the data in Amazon S3, the leadership team for Any Company Financial wants to focus on securing the *network* in the AWS Cloud that hosts the company's critical applications. They are aware of recent network security incidents and want to ensure that their network is protected from unauthorized access and attacks. So this task is to secure the virtual private clouds (VPCs) for the company's web servers.

Task 2.1: Review LabVPC and its associated resources:

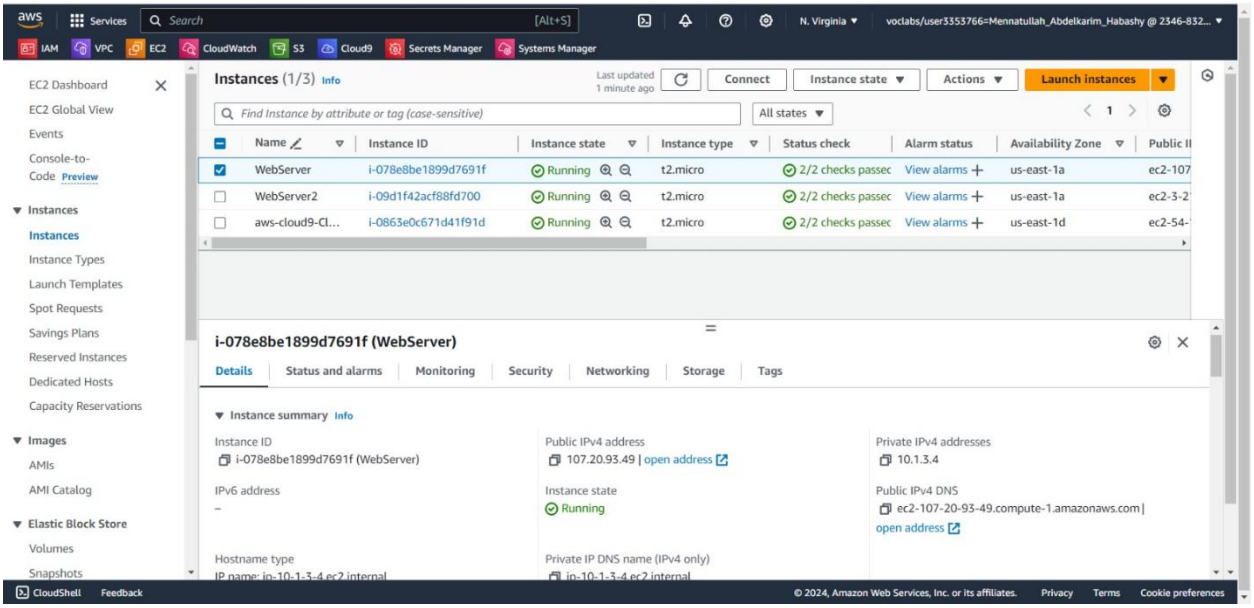
In this task, I will familiarize mySelf with resources that already exist in the lab environment.

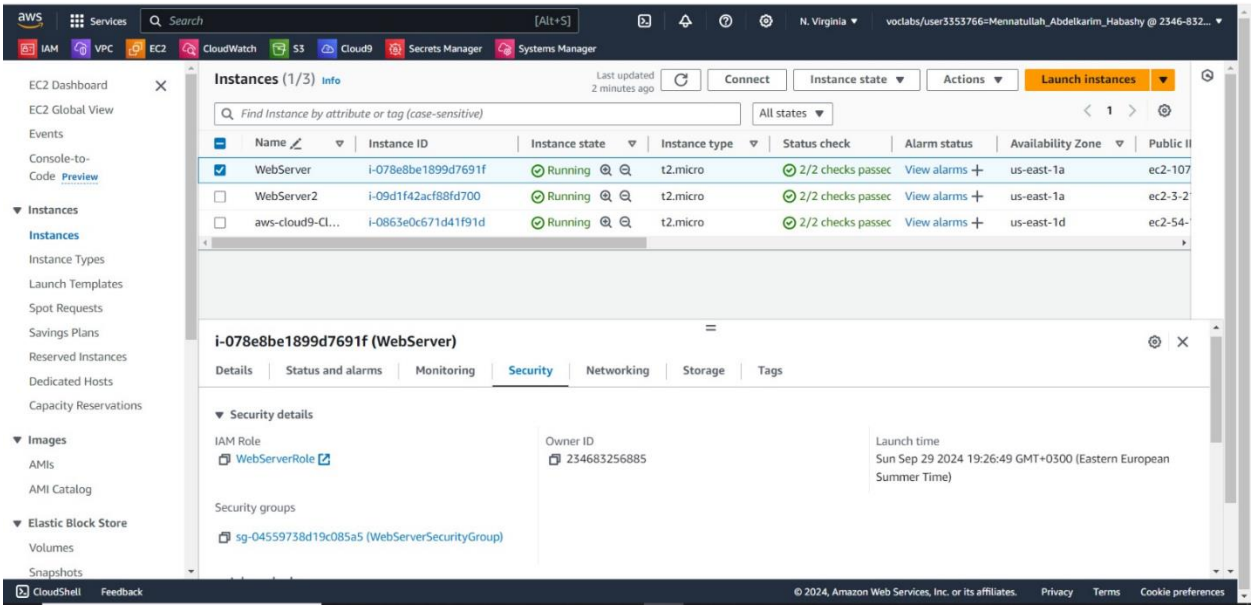


In the IAM console, locate the *VPCFlowLogsRole* IAM role. Review the permissions that are granted to this role.



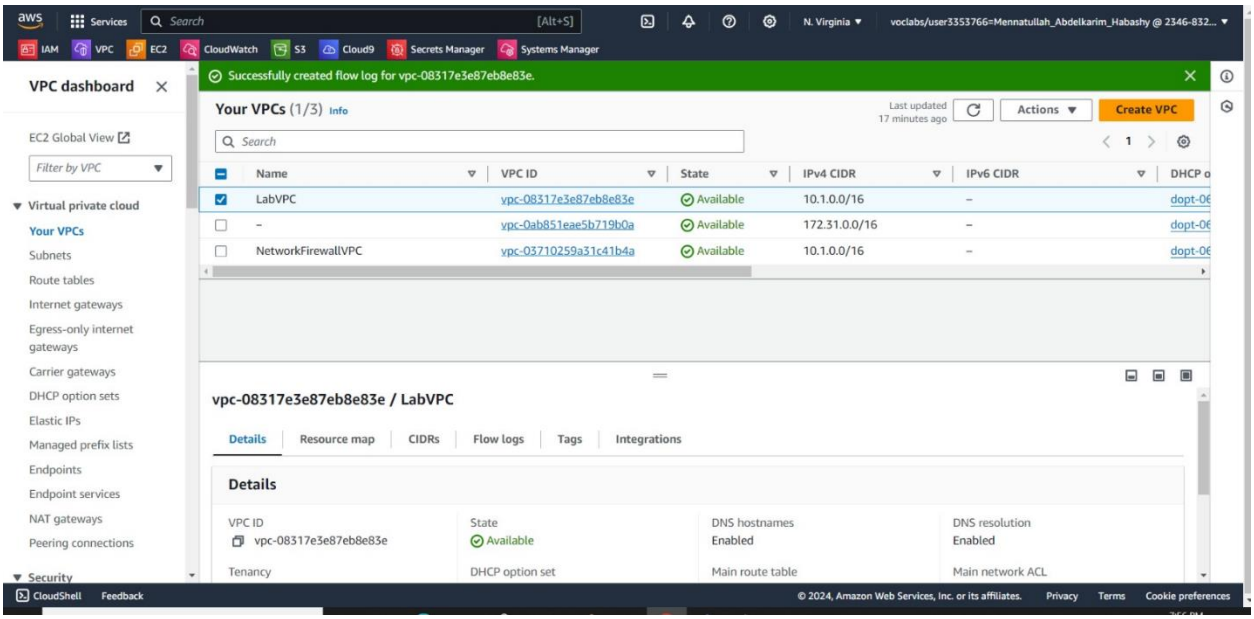
In the Amazon EC2 console, observe the details for the *WebServer* instance.

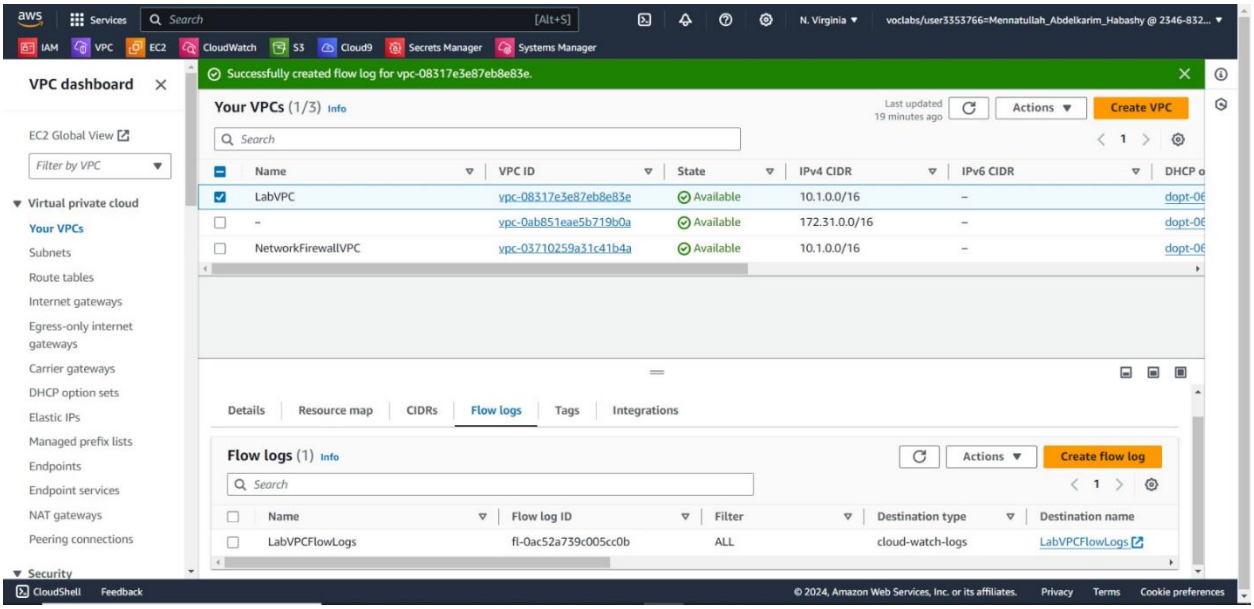




Task 2.2: Create a VPC flow log

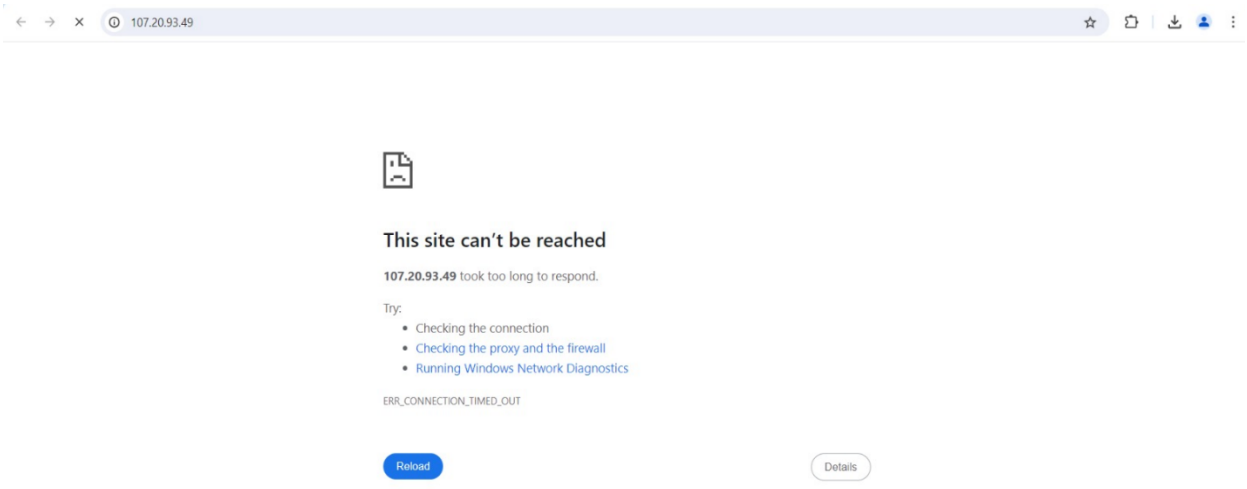
In this task, I'm challenged to create a VPC flow log for *LabVPC*. The VPC Flow Logs feature can help me understand how inbound and outbound traffic flows through the VPC.



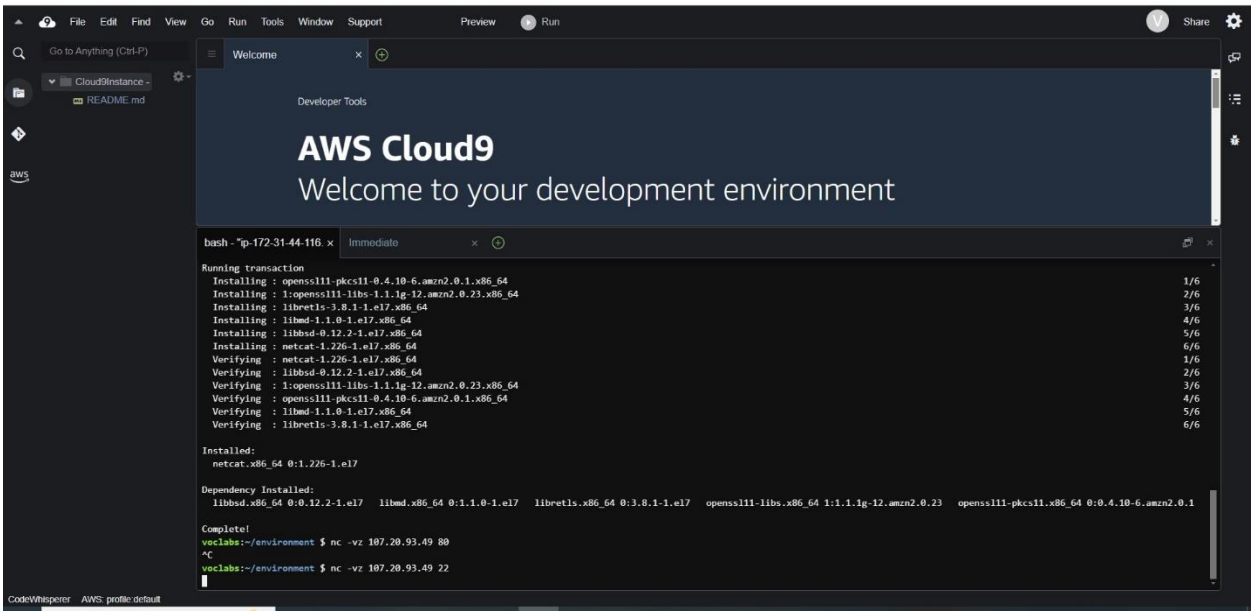


Task 2.3: Access the WebServer instance from the internet and review VPC flow logs in CloudWatch:

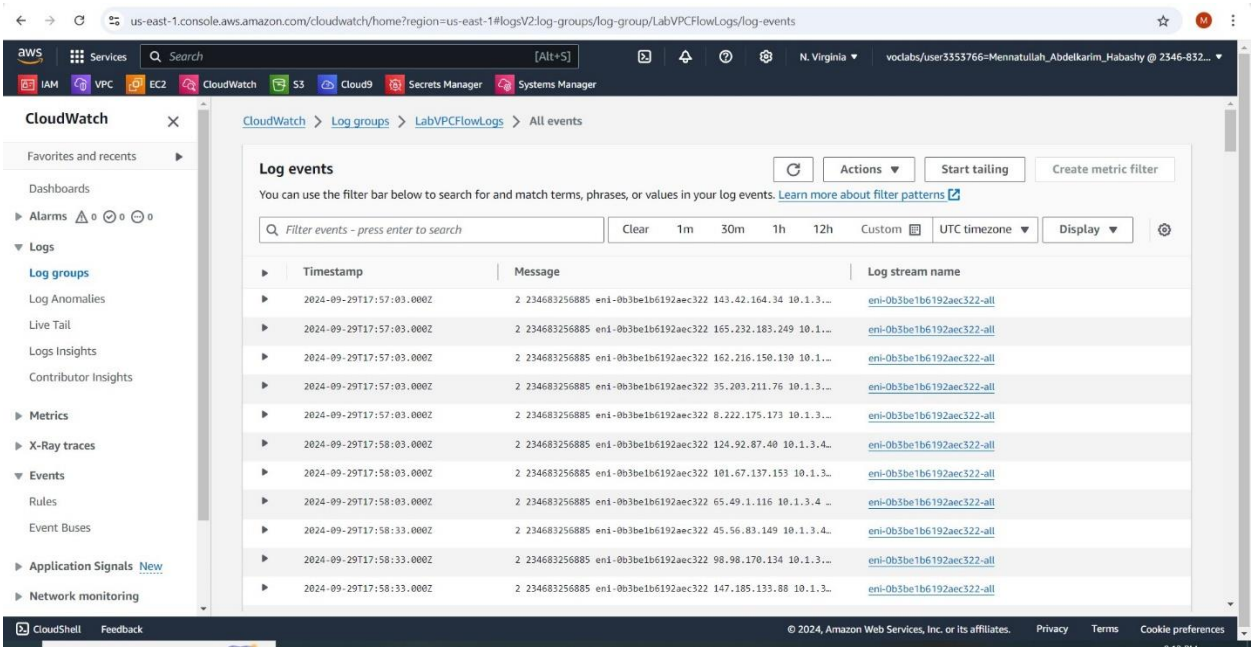
The page doesn't load. This is expected. Recall that an inexperienced employee of Any Company Financial made some mistakes and the result is that the network is not properly configured.



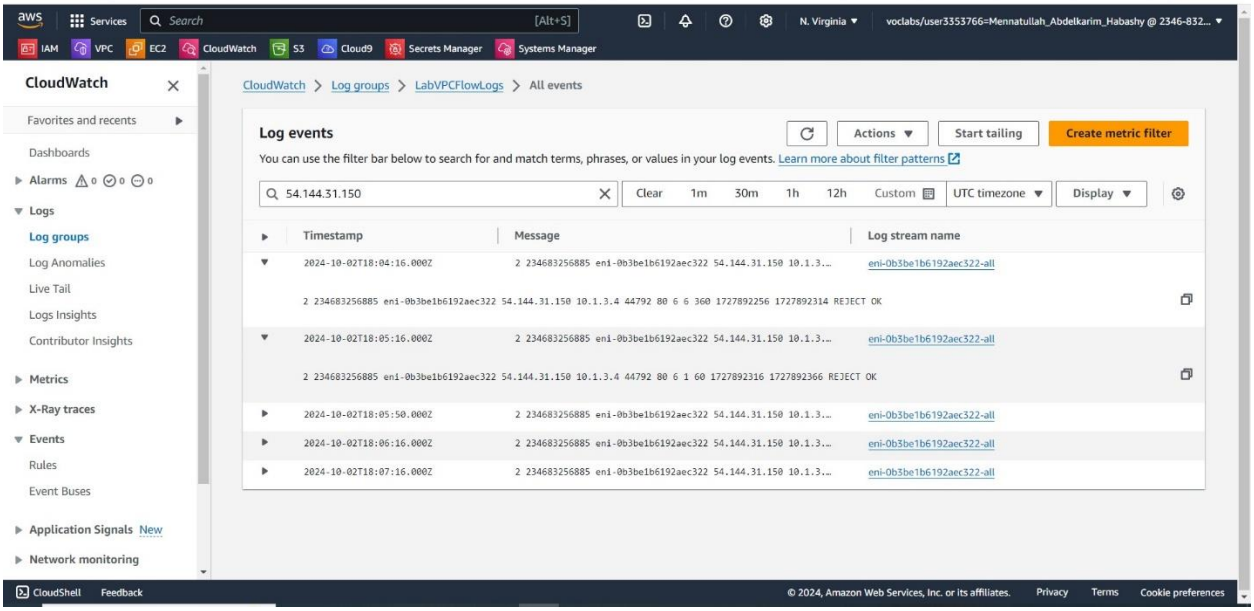
-Use the AWS Cloud9 Cloud9Instance IDE to test access to the *WebServer* instance over the standard HTTP port (80) After some time passes, the connection fails or times out, Next, try to access the same instance on standard SSH port (22) This connection also fails or times out.



look at the log stream for the *LabVPCFlowLogs* log group to verify logs were generated by my actions in the previous step.



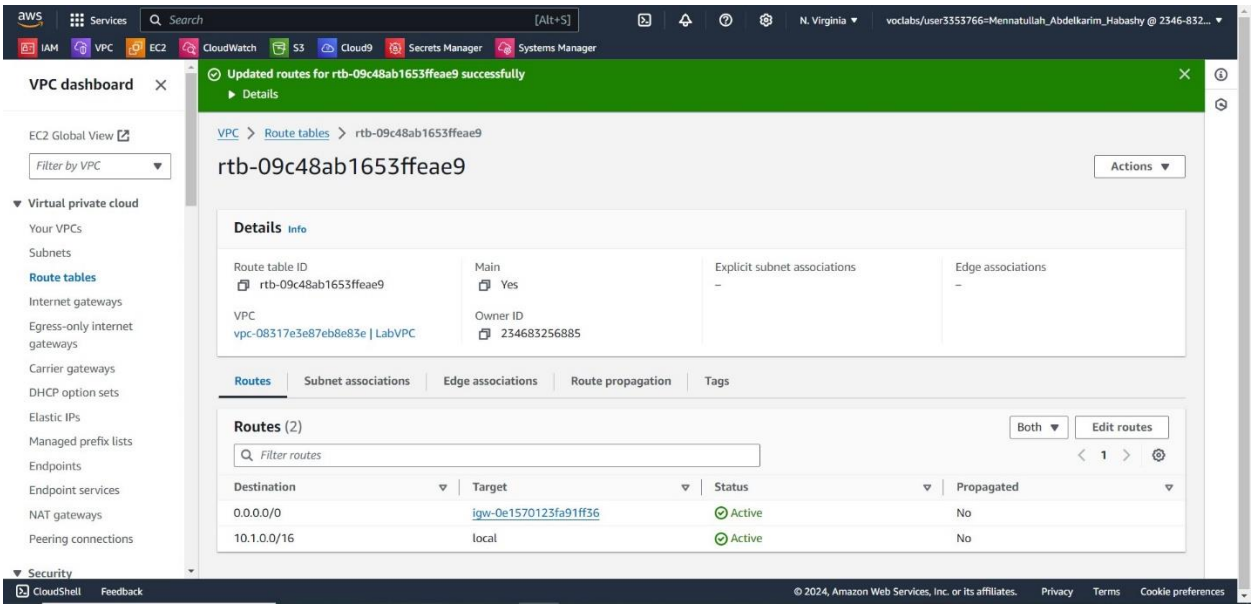
Filter the log entries to display only the events that originated from the public IP address of the AWS Cloud9 instance:



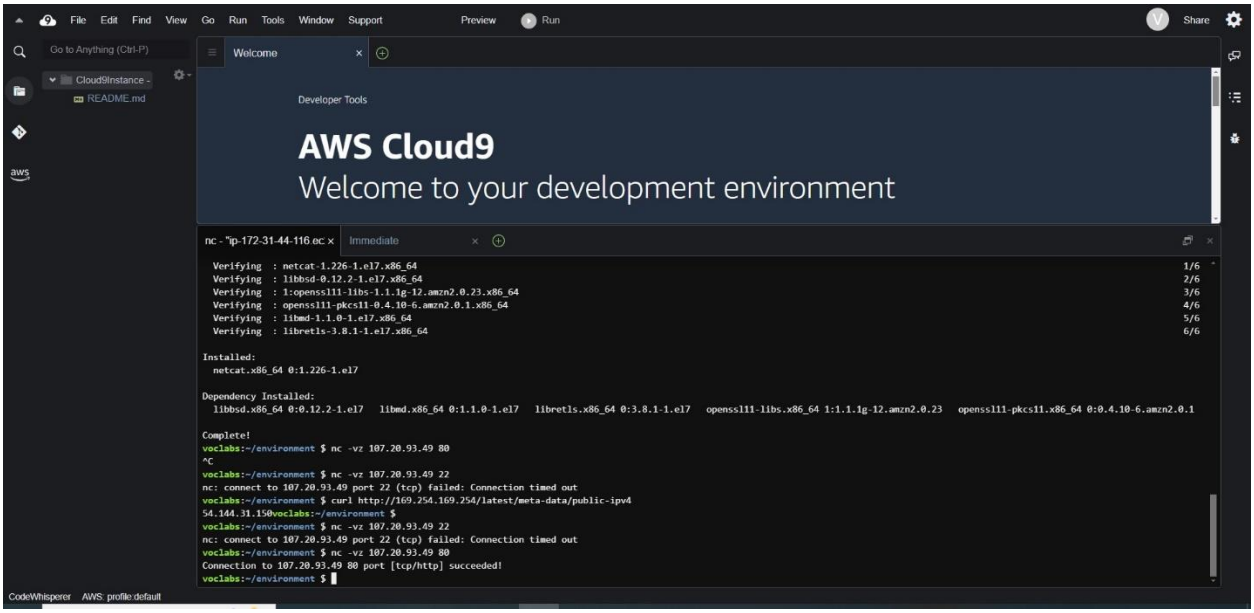
Task 2.4: Configure route table and security group settings:

In this task, I'm challenged to create a route for traffic from the internet to access the *WebServerSubnet* through an internet gateway. This will allow inbound HTTP traffic to be directed to the *WebServer* instance. I will also be challenged to modify the security group that is associated with the *WebServer* instance to allow inbound traffic on ports 22 (SSH) and 80 (HTTP).

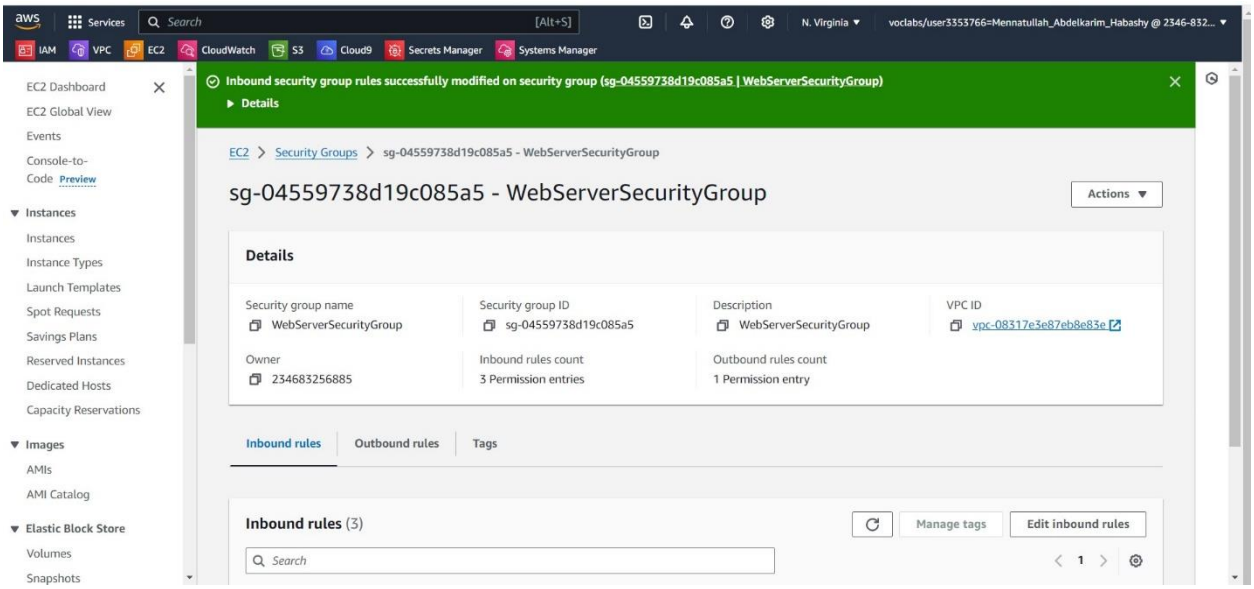
-Modify the route table that is associated with the *WebServerSubnet*.

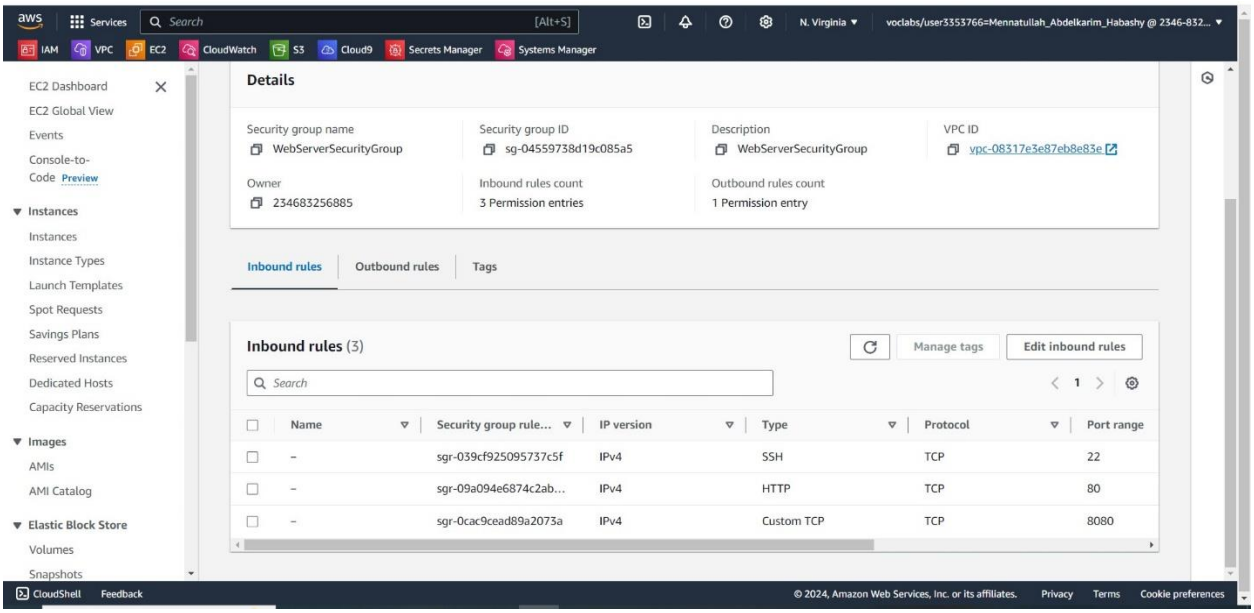


-Test access from port 80 to the *WebServer* instance again

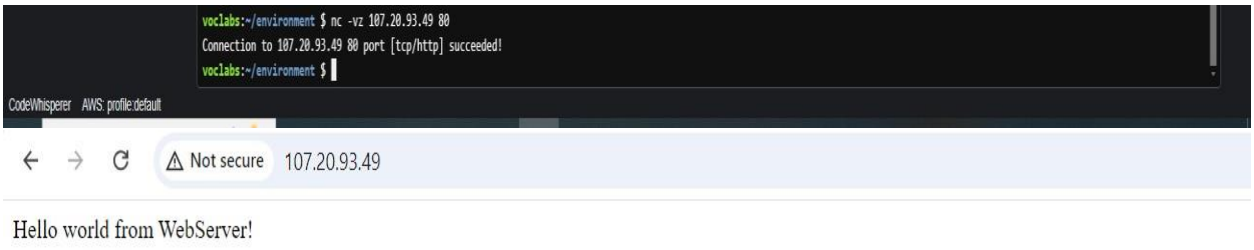


-Modify the inbound rules for the security group





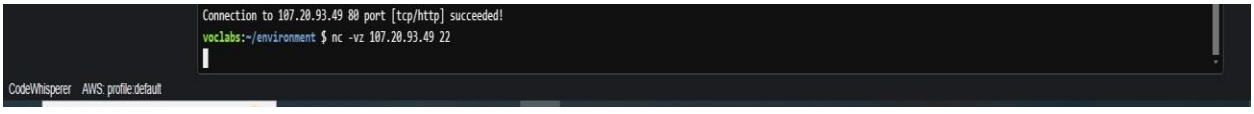
-test access from port 80 to the webpage on the *WebServer* instance.The test should be successful. The message "Hello world from WebServer!" displays:



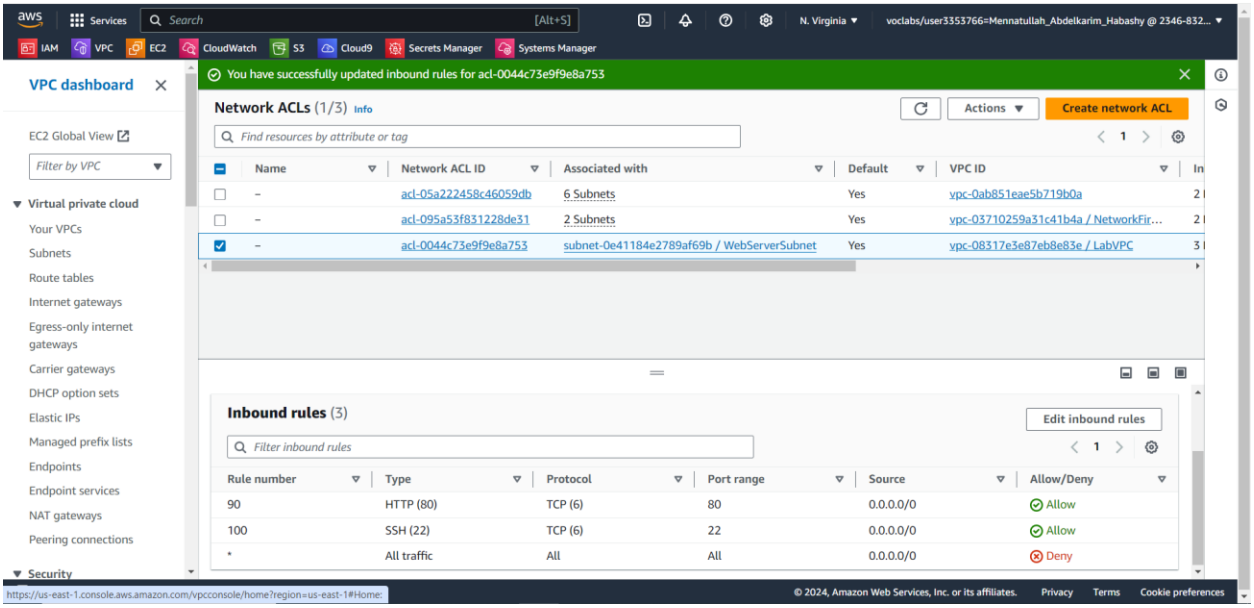
Task 2.5: Secure the WebServerSubnet with a network ACL

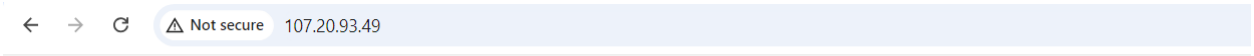
In this task, I'm challenged to configure a network access control list (ACL) to secure the subnet where the web server is running. The network ACL will provide an additional layer of security beyond the security group that i already configured.

-To test whether you can access port 22 on the *WebServer* instance, modify the *100* rule from *Allow* to *Deny*, **The connection fails or times out:**



-Modify rule number 100 to *allow* connections on port 22 only and Edit the inbound rules for the network ACL again. Add a *new* rule that allows *HTTP* traffic from *anywhere*. Use *90* as the rule number:



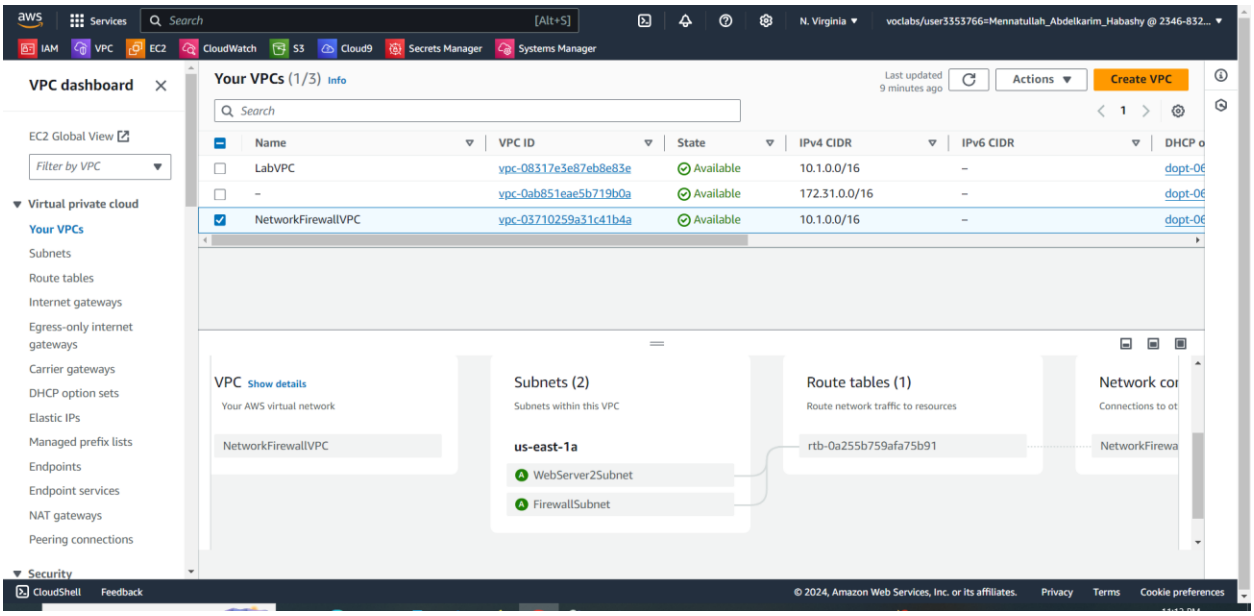


Hello world from WebServer!

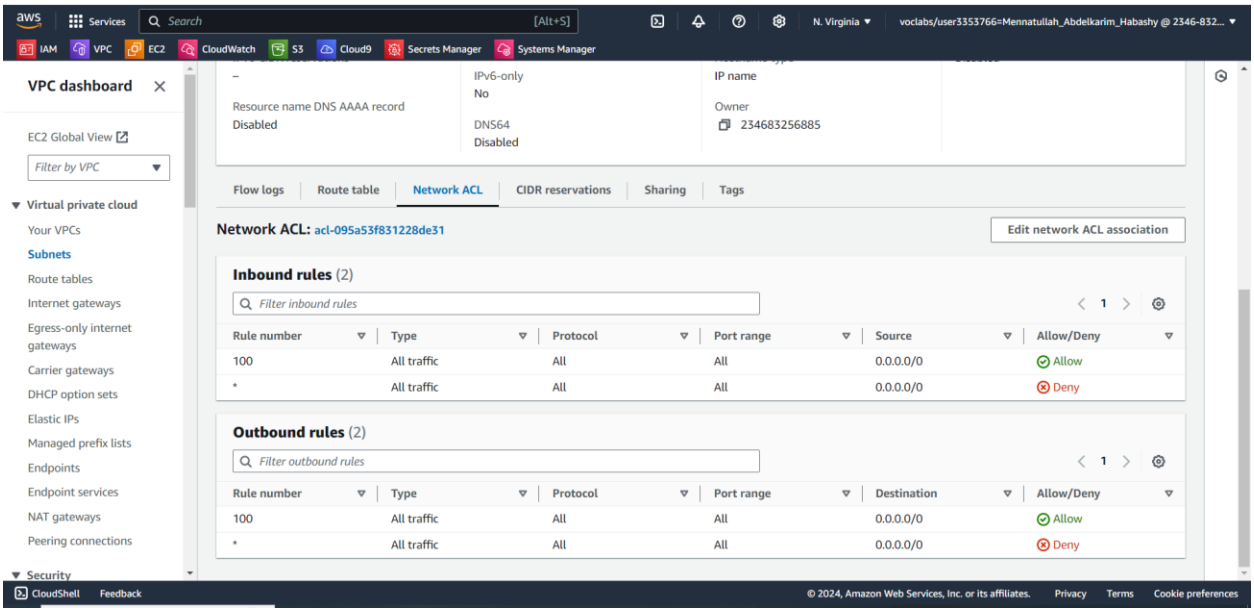
Task 2.6: Review NetworkFirewallVPC and its associated resources:

In this phase so far, I worked to secure *LabVPC* by updating a route table, network ACL, and security group.

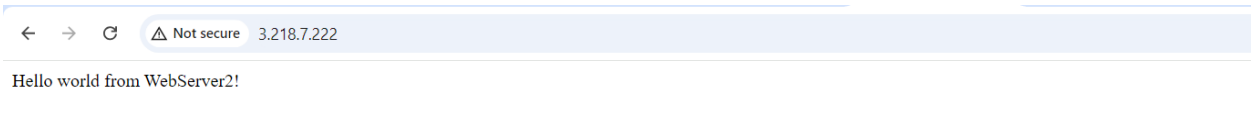
-Observe the existing NetworkFirewallVPC:



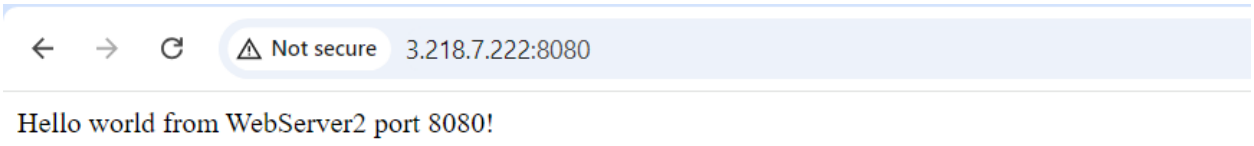
-observe the network ACL settings for *NetworkFirewallVPC*:



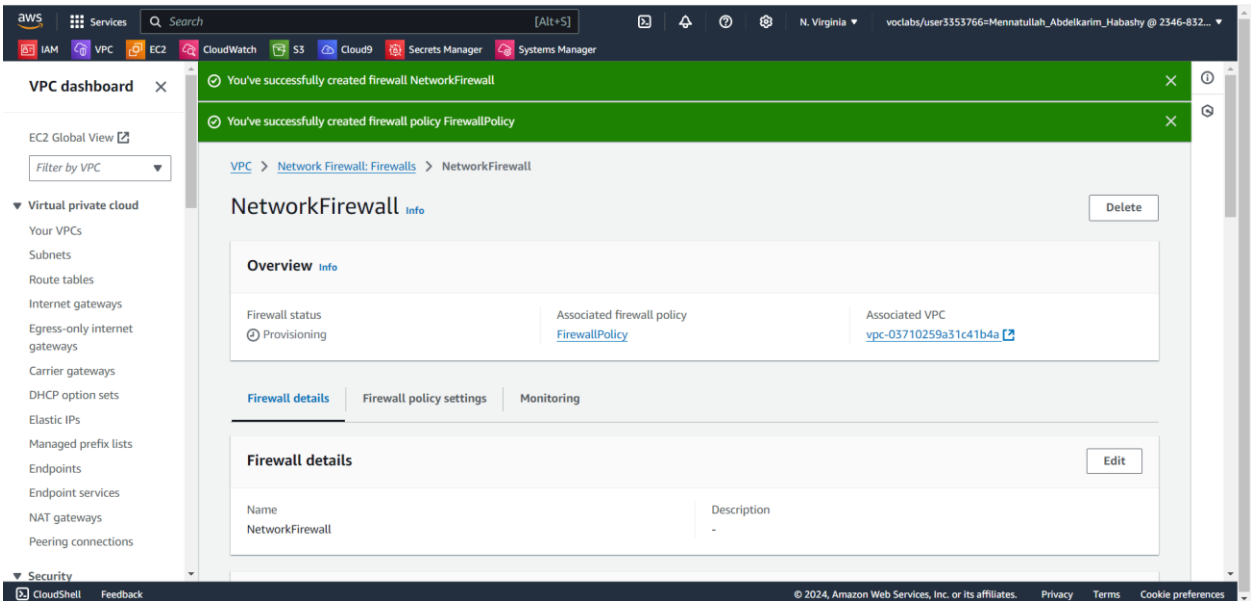
- -observe the settings for the *WebServer2* instance and Confirm access to the WebServer2 instance on ports 80 and 22, Verify HTTP access using a browser, should see the "Hello world from WebServer2!" message:



-Start an additional website that runs on *WebServer2* port 8080 and test access, The attempt should be successful, and should see the **"Hello world from WebServer2 port 8080!"** message:



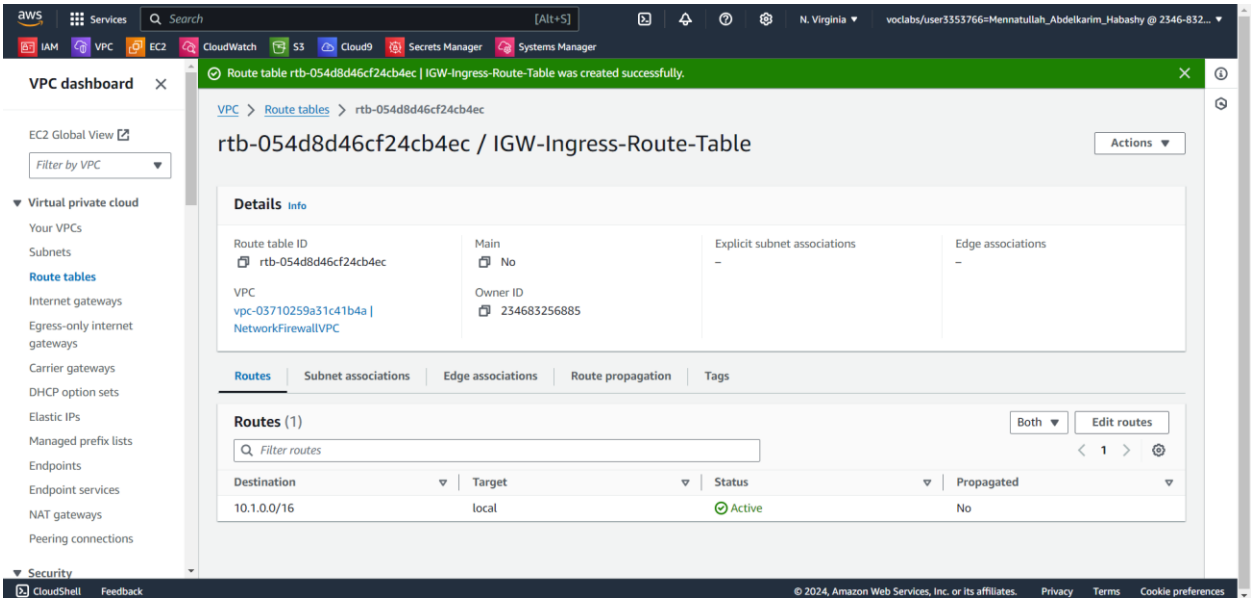
Task 2.7: Create a network firewall



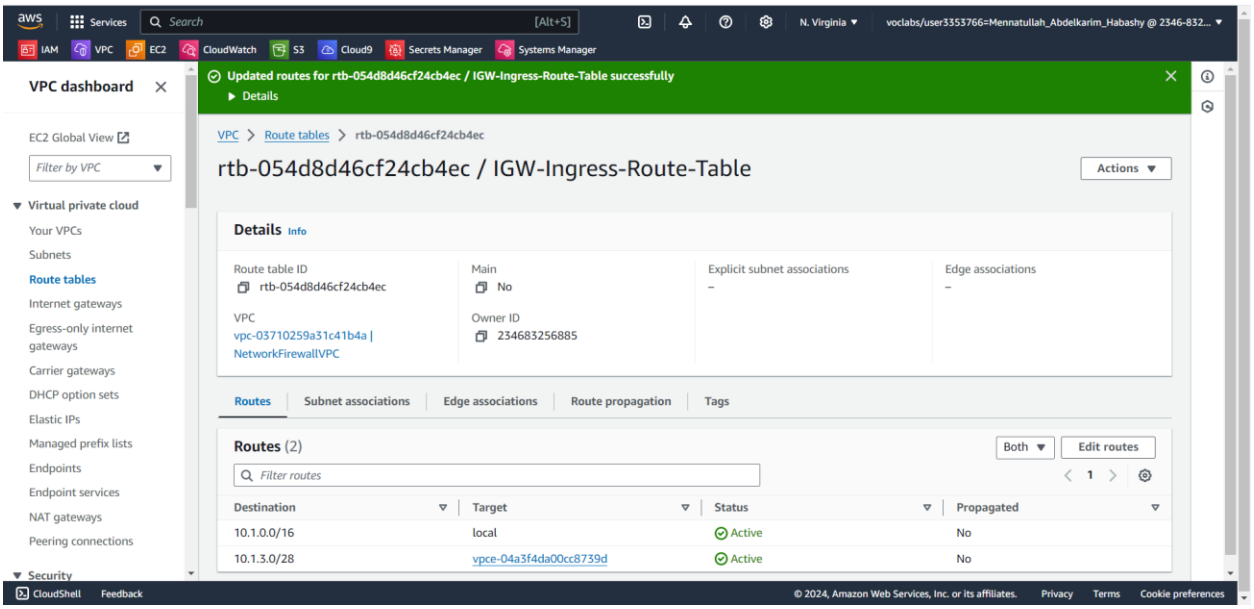
Task 2.8: Create route tables

In this task, I'm challenged to create and configure three new route tables, including one for each subnet in the *NetworkFirewallVPC* and one to handle inbound (ingress) traffic for the internet gateway in *NetworkFirewallVPC*.

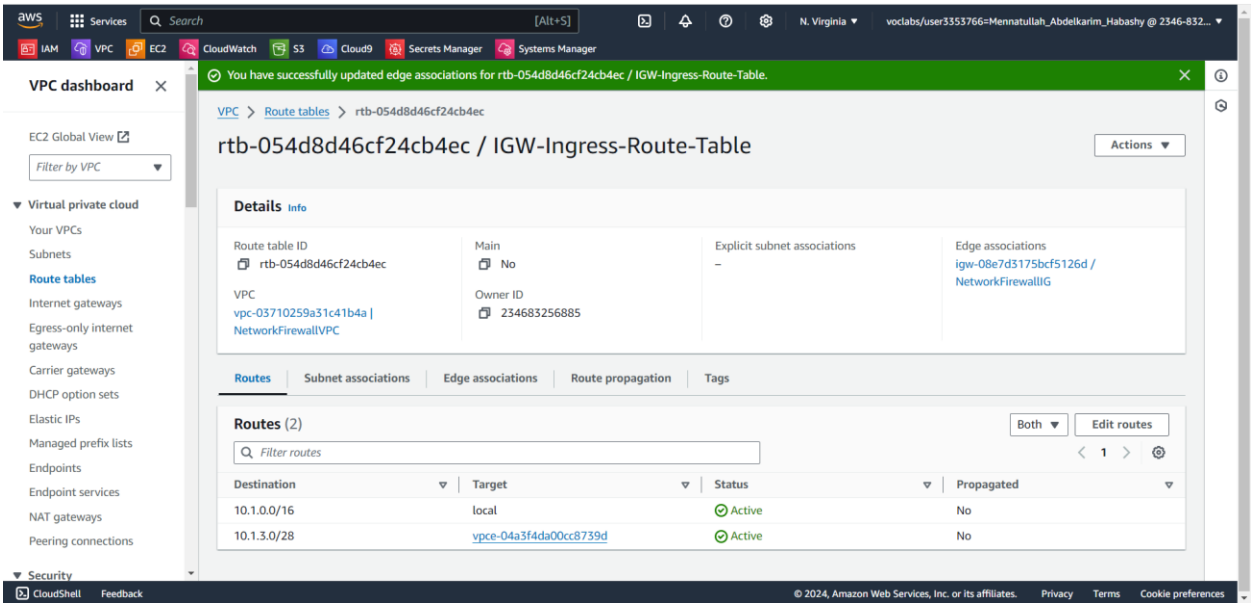
-Create a new route table in the *NetworkFirewallVPC*:



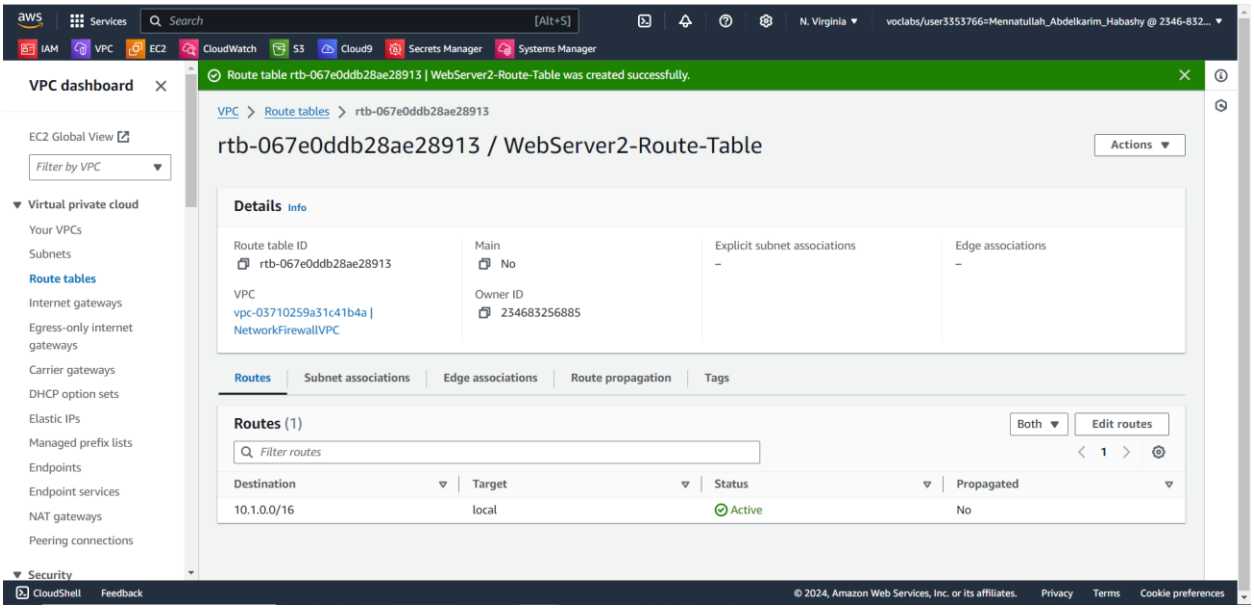
-Edit the route table to add a new route:



- Add an edge association to the *IGW-Ingress-Route-Table* so that the *NetworkFirewallIG* internet gateway is associated with the route table:



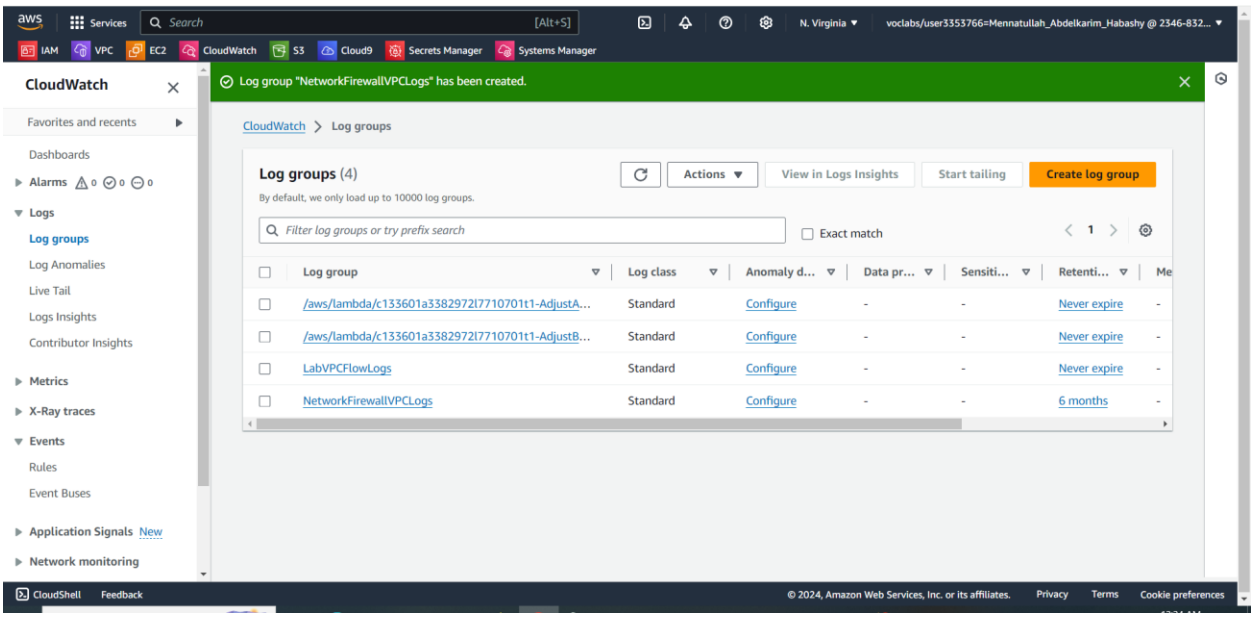
- Create another route table in *NetworkFirewallVPC* for the *Webserver2Subnet:*



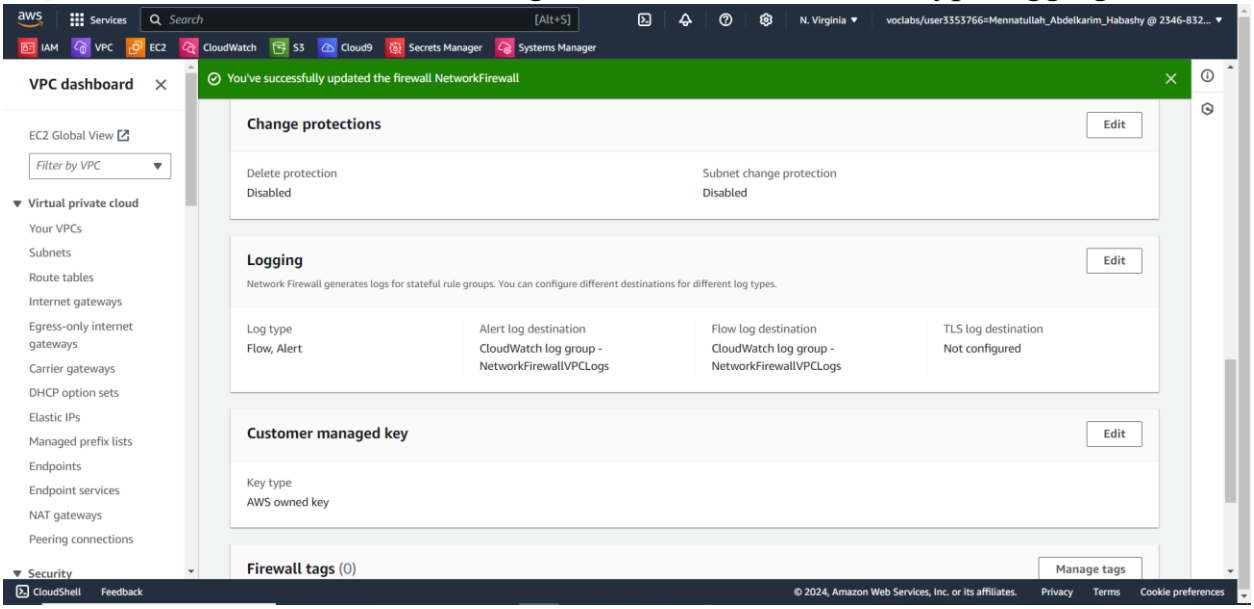
Task 2.9: Configure logging for the network firewall

In this task, I'm challenged to configure logging for the network firewall so that I can analyze details of network traffic requests.

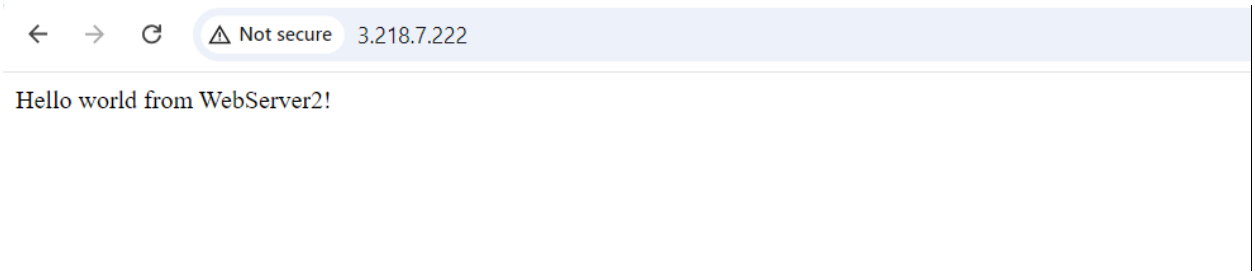
- Create a CloudWatch log group named NetworkFirewallVPCLogs with a retention setting of *6 months*:



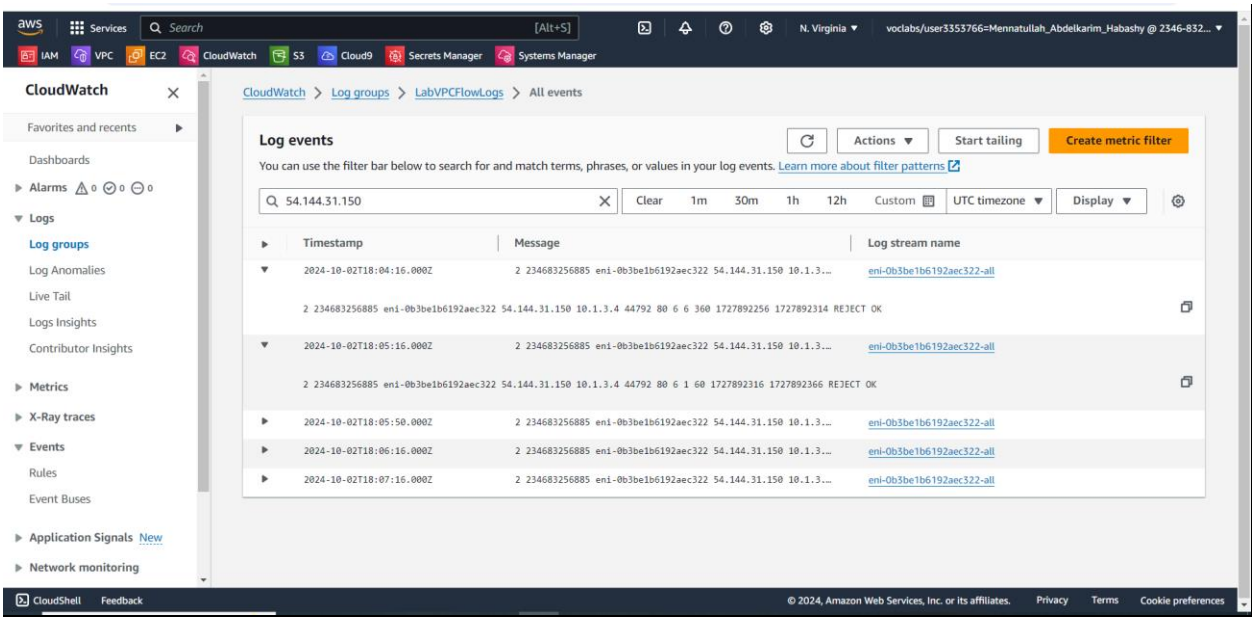
-in **Firewall details** area, and configure both **Alert** and **Flow** type logging:



-Test the logging configuration by attempting to load *WebServer2* instance's website on port 80 :

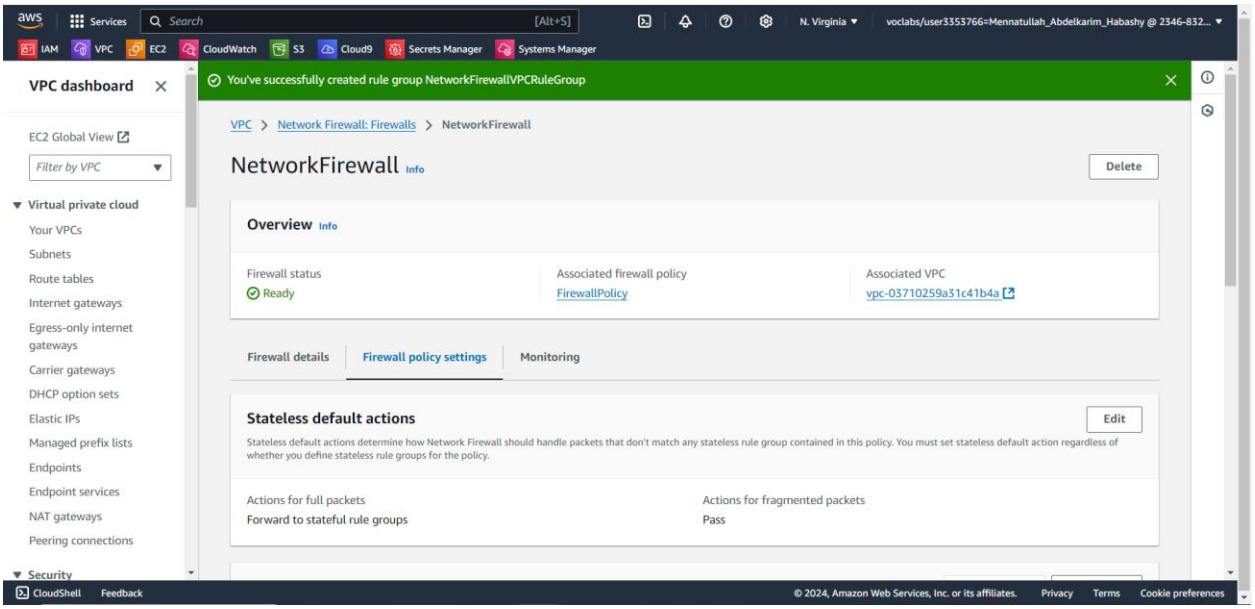


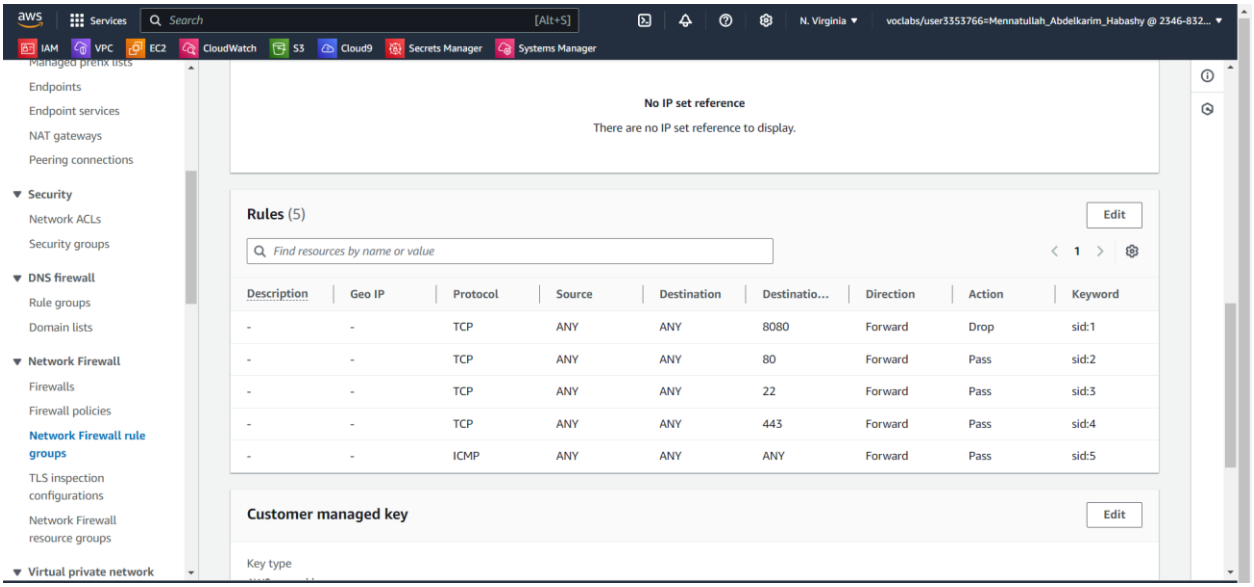
in the *NetworkFirewallVPCLogs* log group. Filter the log events by public IP address:



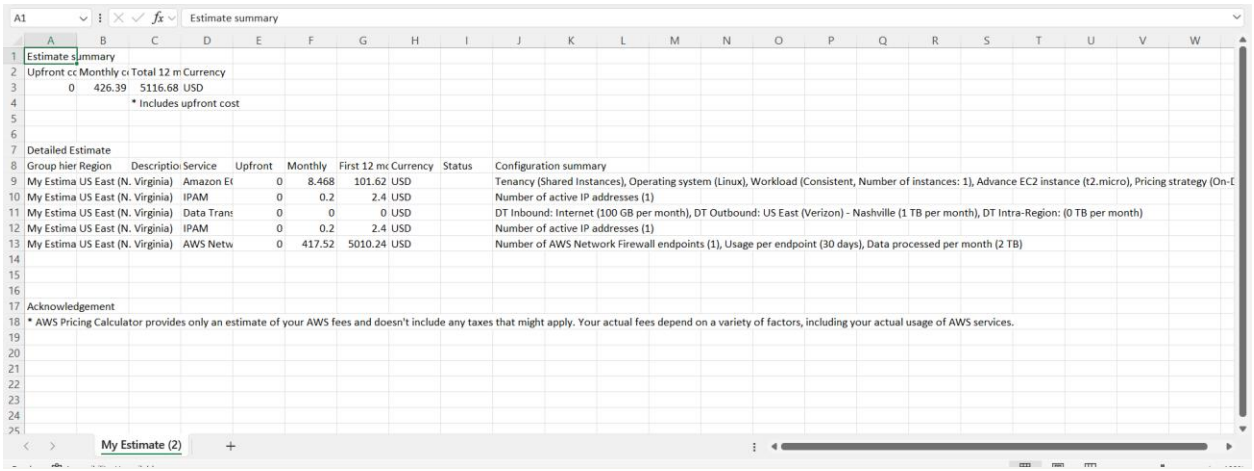
Task 2.10: Configure the firewall policy and test access:

In this task, I'm challenged to define and add a stateful rule group to the network firewall's policy.





- Cost estimate to secure a VPC with a network firewall:



Estimate summary

Upfront cc	Monthly c	Total 12 m	Currency
0	426.39	5116.68	USD

* Includes upfront cost

Detailed Estimate

Group hier	Region	Descriptio	Service	Upfront	Monthly	First 12 mc	Currency	Status	Configuration summary
My Estima	US East (N. Virginia)		Amazon E(	0	8.468	101.62	USD		Tenancy (Shared Instances), Operating system (Linux), Workload (Consistent, Number of instances: 1), Advance EC2 instance (t2.micro), Pricing strategy (On-D
My Estima	US East (N. Virginia)		IPAM	0	0.2	2.4	USD		Number of active IP addresses (1)
My Estima	US East (N. Virginia)		Data Trans	0	0	0	USD		DT Inbound: Internet (100 GB per month), DT Outbound: US East (Verizon) - Nashville (1 TB per month), DT Intra-Region: (0 TB per month)
My Estima	US East (N. Virginia)		IPAM	0	0.2	2.4	USD		Number of active IP addresses (1)
My Estima	US East (N. Virginia)		AWS Netw	0	417.52	5010.24	USD		Number of AWS Network Firewall endpoints (1), Usage per endpoint (30 days), Data processed per month (2 TB)

Acknowledgement

* AWS Pricing Calculator provides only an estimate of your AWS fees and doesn't include any taxes that might apply. Your actual fees depend on a variety of factors, including your actual usage of AWS services.

My Estimate (2)

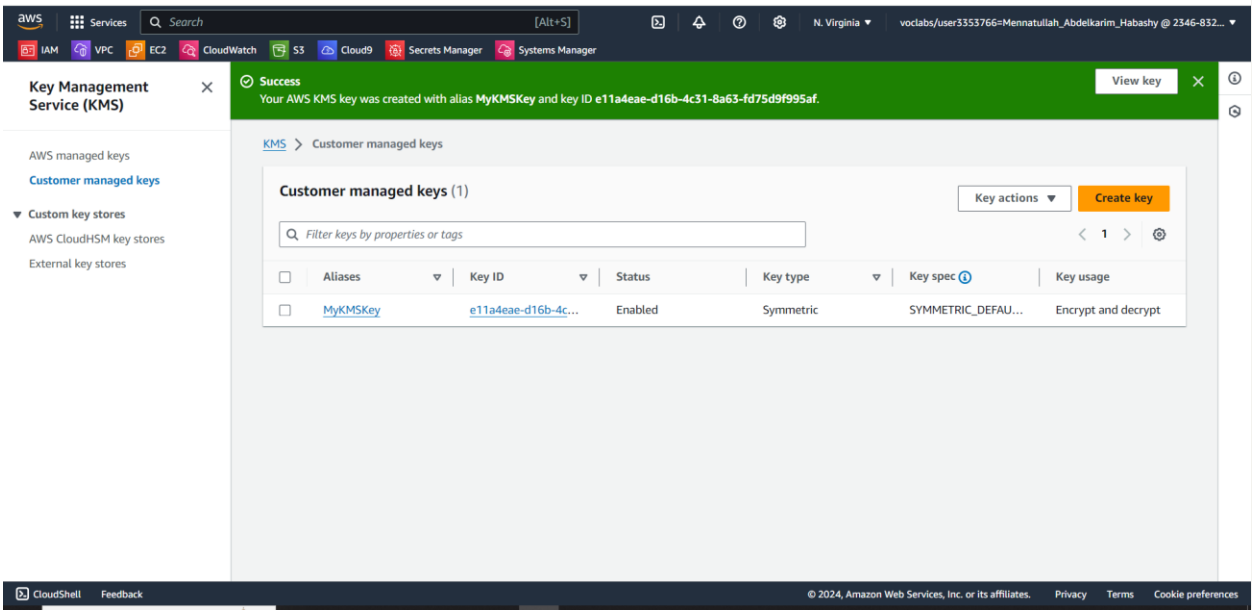
Phase 3: Securing AWS resources by using AWS KMS:

The legal department at Any Company Financial received notice from the U.S. Federal Deposit Insurance Corporation (FDIC) that the company needs to encrypt sensitive information, such as PII, credit card numbers, and social security numbers. The legal department contacted manager, the director of IT, and they tasked to implement encryption and tokenization to meet regulatory compliance standards.

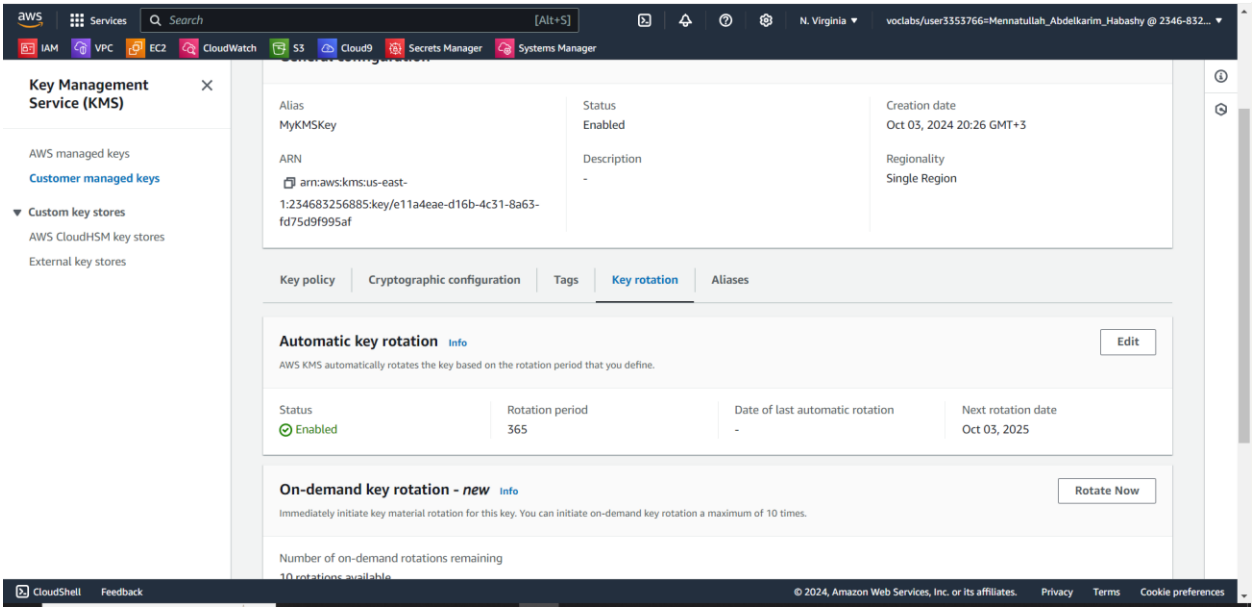
Task 3.1: Create a customer managed key and configure key rotation:

In this task, I will create an AWS KMS customer managed key. I will then configure automatic key rotation on the key.

- Create an AWS KMS customer managed key:



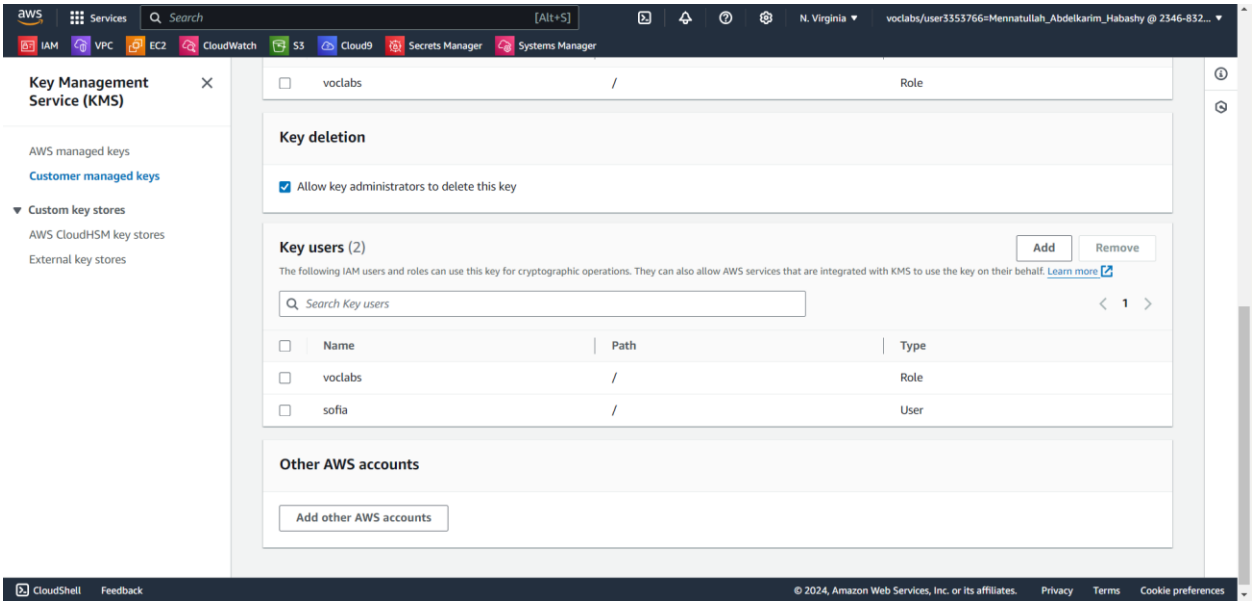
- Configure AWS KMS key rotation on the new key so that it is automatically rotated every year.



Task 3.2: Update the AWS KMS key policy and analyze an IAM policy

In this task, I'm challenged to modify the policy of the AWS KMS key that I created so that the *sofia* user will be authorized to use the key. I'm also challenged to analyze the IAM policy that controls what the *sofia* user can do in the AWS account.

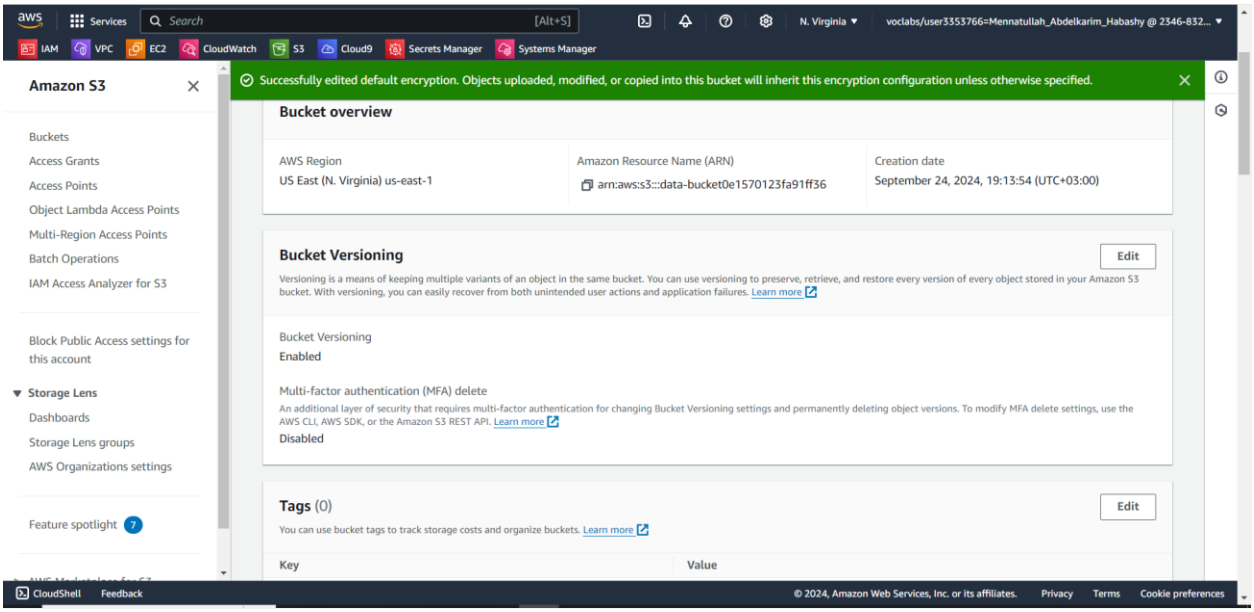
- Modify the key policy:



Task 3.3: Use AWS KMS to encrypt data in Amazon S3

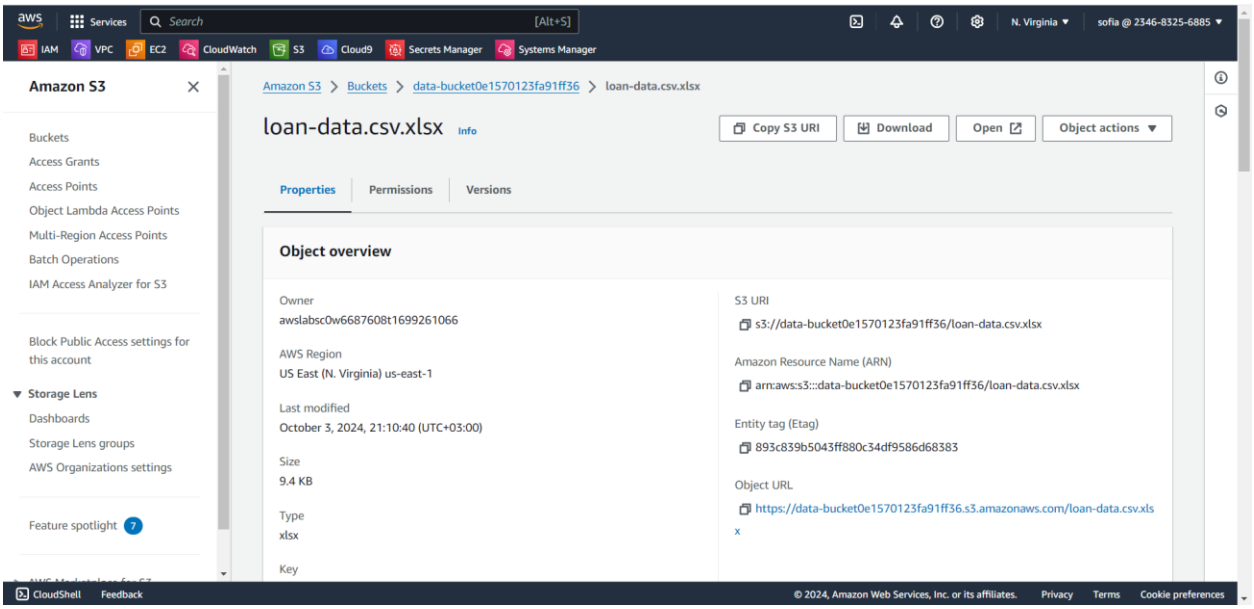
In this task, I'm challenged to use the AWS KMS key that I created to encrypt an object in the *data-bucket* S3 bucket. I will then test access to the object.

- Modify the encryption settings on the *data-bucket* S3 bucket:



- log in to the AWS Management Console as the *sofia* IAM user:

- Note that:The upload, open or download should be successful

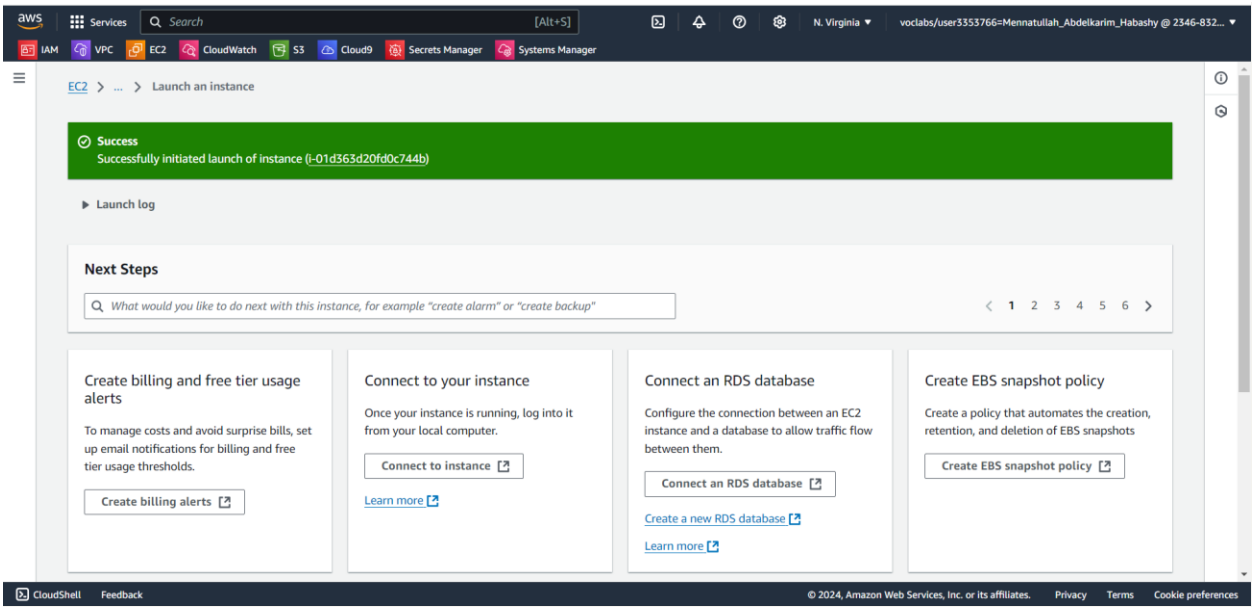


- Sign the *sofia* user out of the console, and then sign in as the *paulo* user. Then, try to open or download, The attempt should fail:

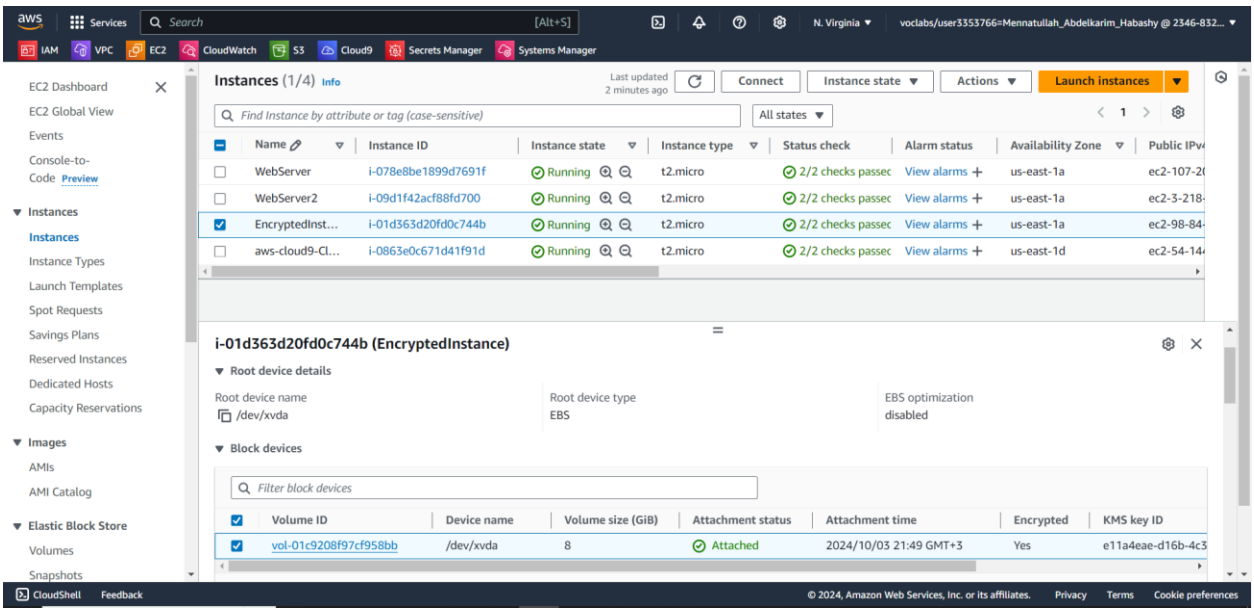
Task 3.4: Use AWS KMS to encrypt the root volume of an EC2 instance

In this task, I'm challenged to use the AWS KMS key again, but now I will use it to encrypt the root volume of a new EC2 instance.

- create a new EC2 instance:



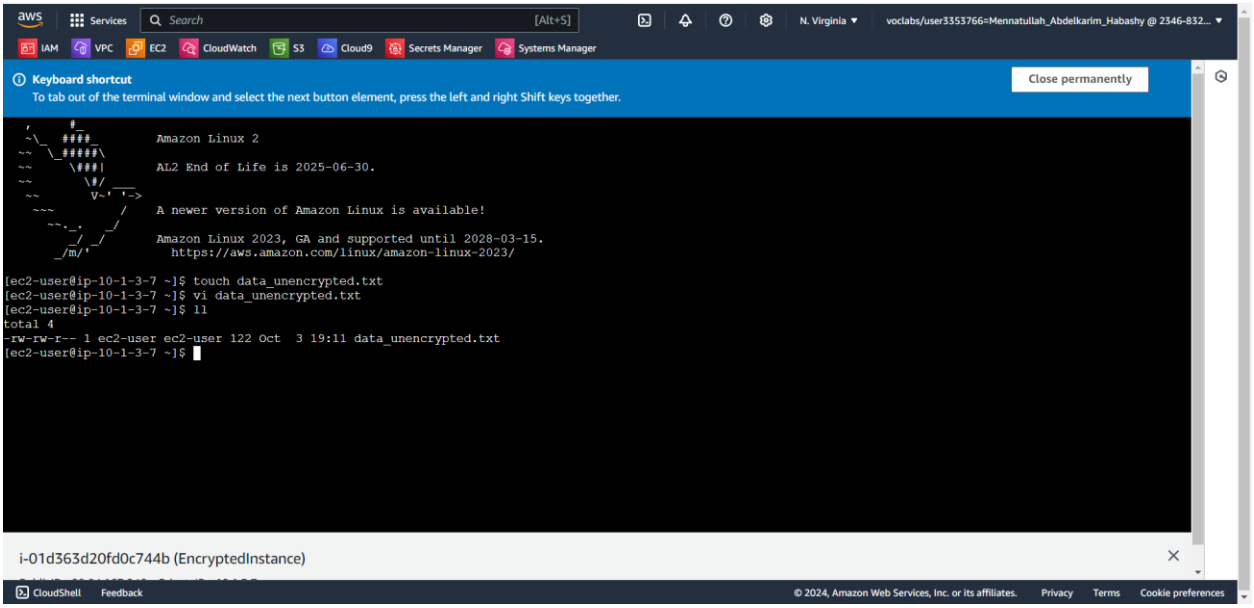
- verify that the instance root volume is encrypted:



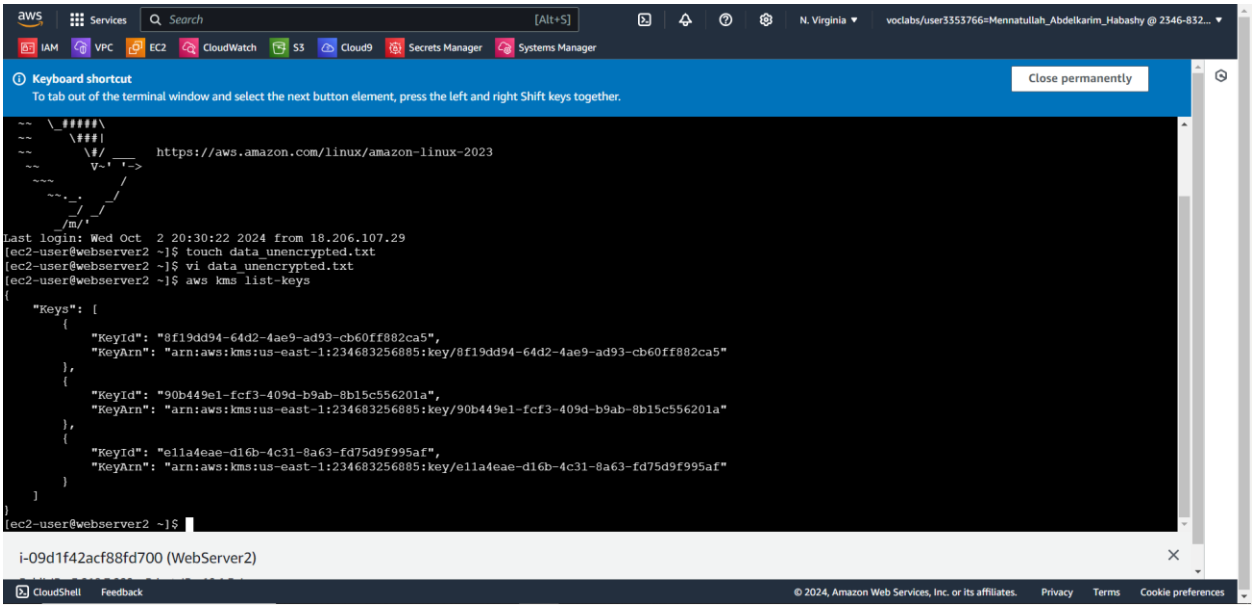
Task 3.5: Use AWS KMS envelope encryption to encrypt data in place

In this task, I'm challenged to use the AWS Command Line Interface (AWS CLI) to encrypt data in place by using the AWS KMS key. I'm also challenged to see how to decrypt the encrypted data.

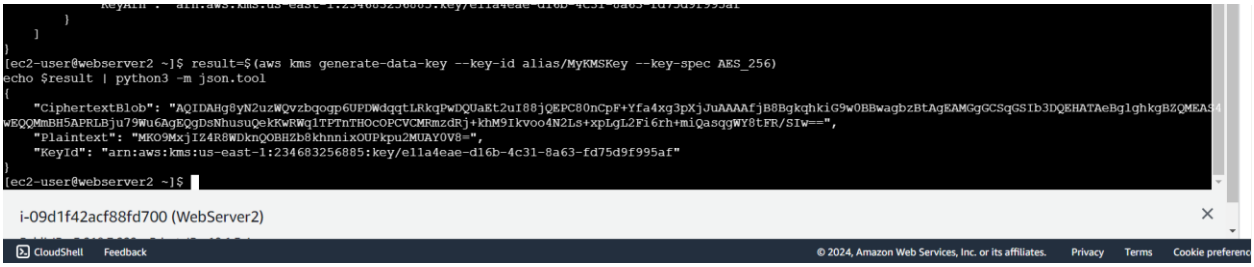
- create a file that you will later try to encrypt



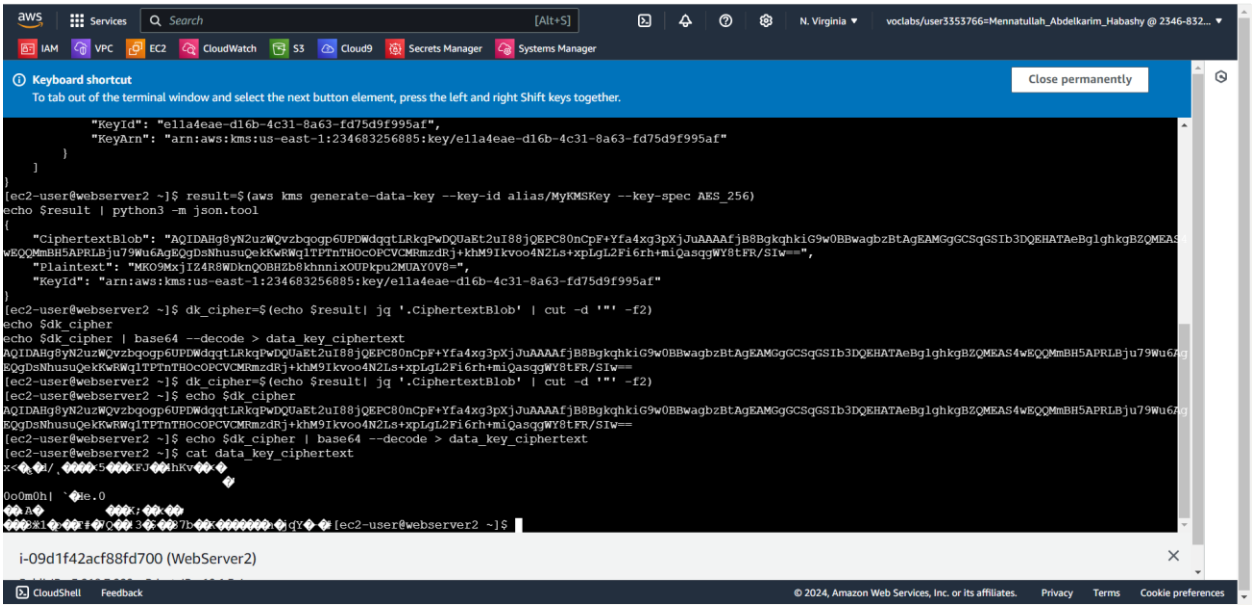
- generate a data key from the AWS KMS key:



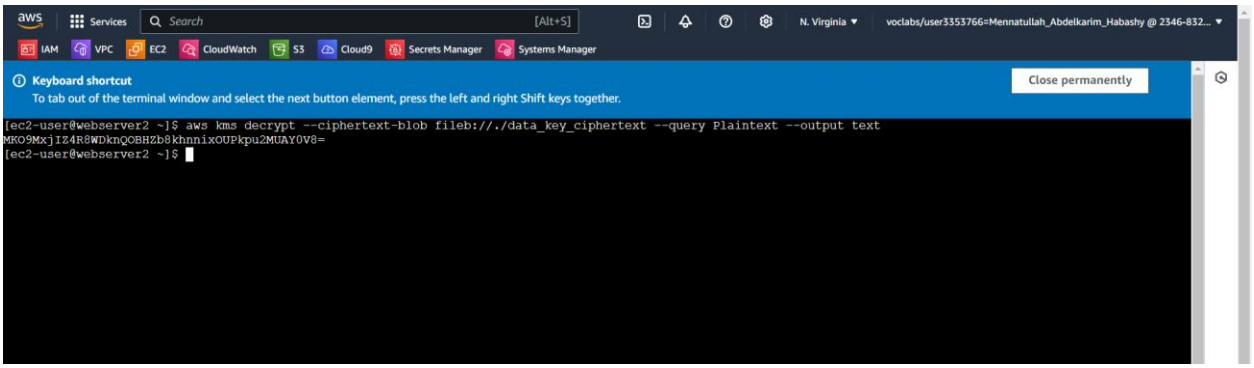
- generate a *data key* for the MyKMSKey, save it to a bash variable, and then echo the data key details:



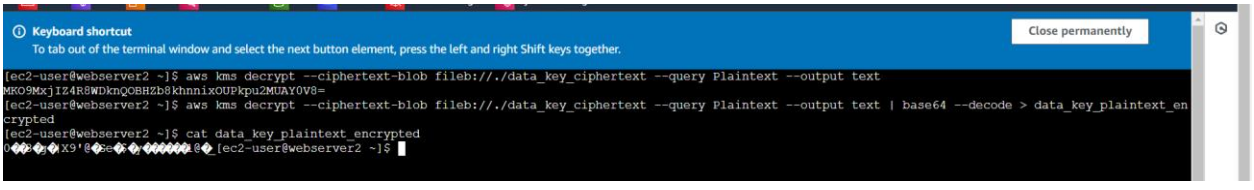
- Save the data key to disk to a text file in base64-encoded format:



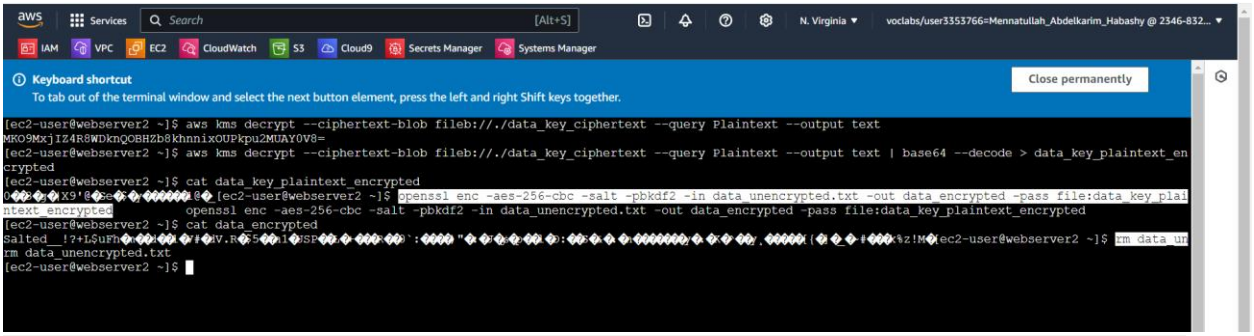
- regenerate the plaintext version of the data key from the base64-encoded ciphertext at anytime:



- also save the previous result to a file in base64-encoded format:

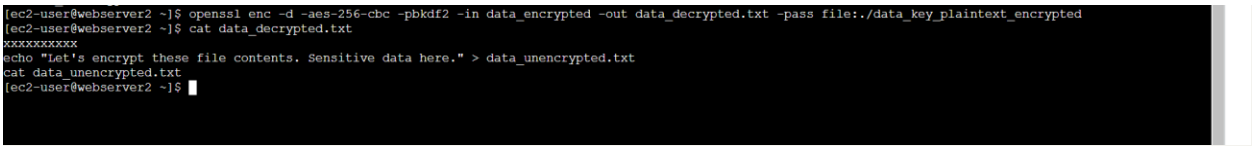


- To encrypt the *data_unencrypted.txt* file and try to read the file:



It's not human-readable. Then, Try To delete the unencrypted version of the file

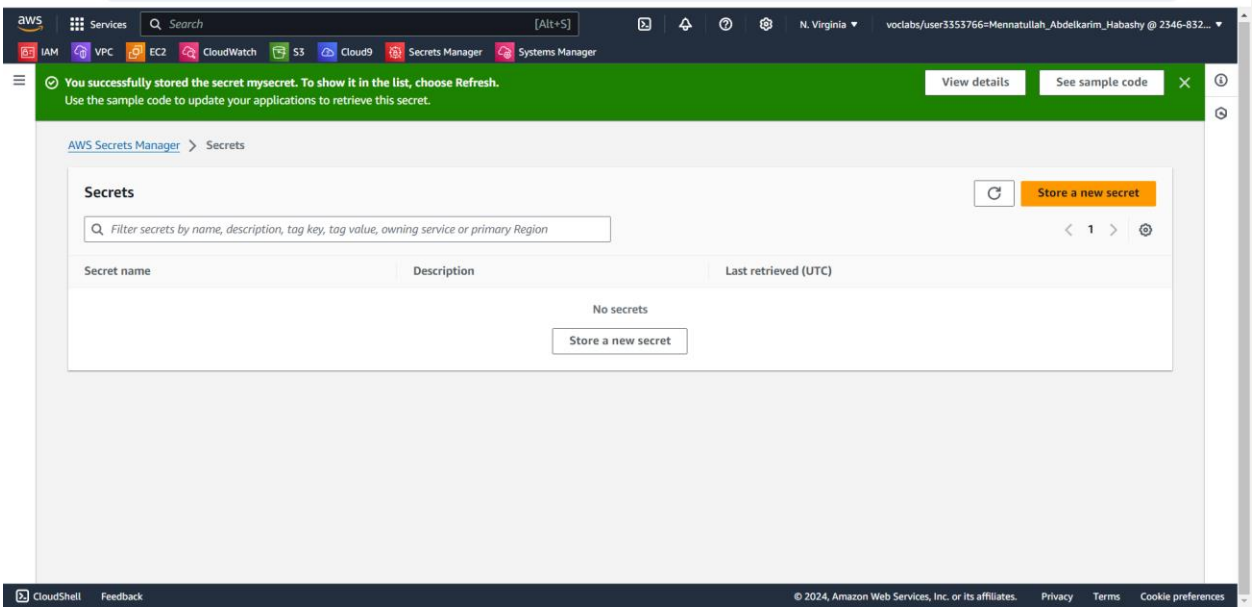
- Decrypt the file to prove that the data is retrievable and print the results to ensure the data is now readable:



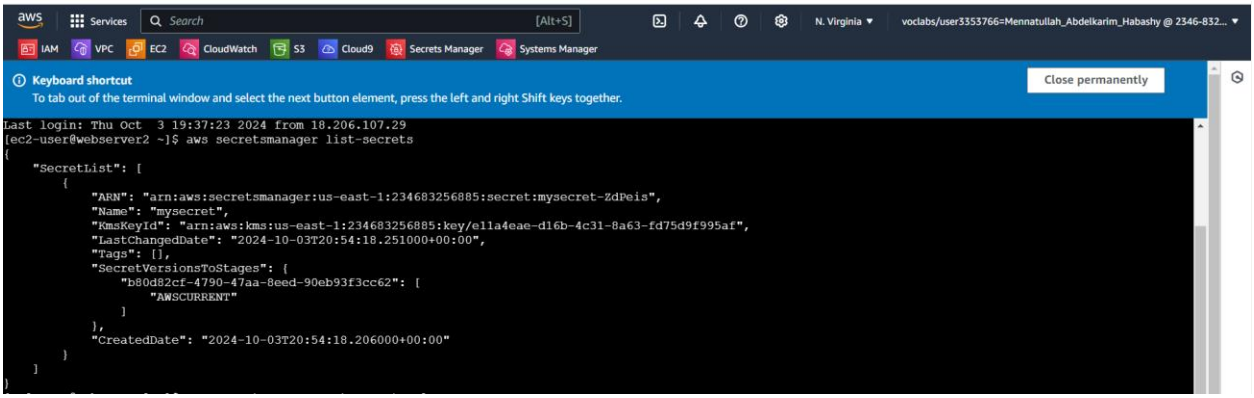
Task 3.6: Use AWS KMS to encrypt a Secrets Manager secret

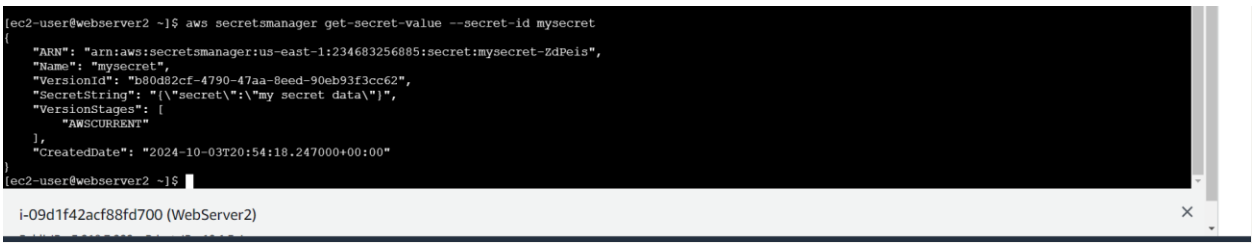
In this task, I'm challenged to create a key-value pair (a *secret*), which I will encrypt with my AWS KMS key and store in Secrets Manager. I'm then challenged to verify that I can retrieve the secret by using the AWS CLI.

- create a new secret of type *Other type of secret:*

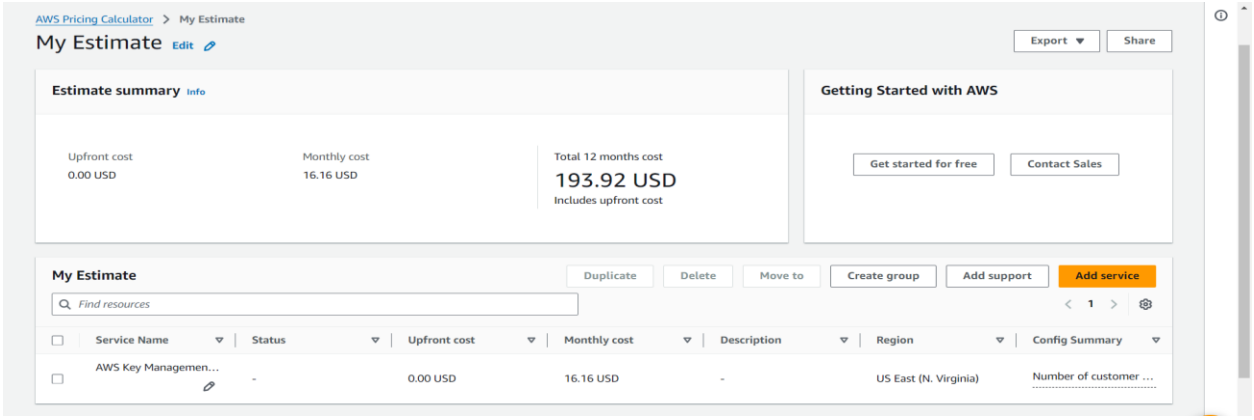


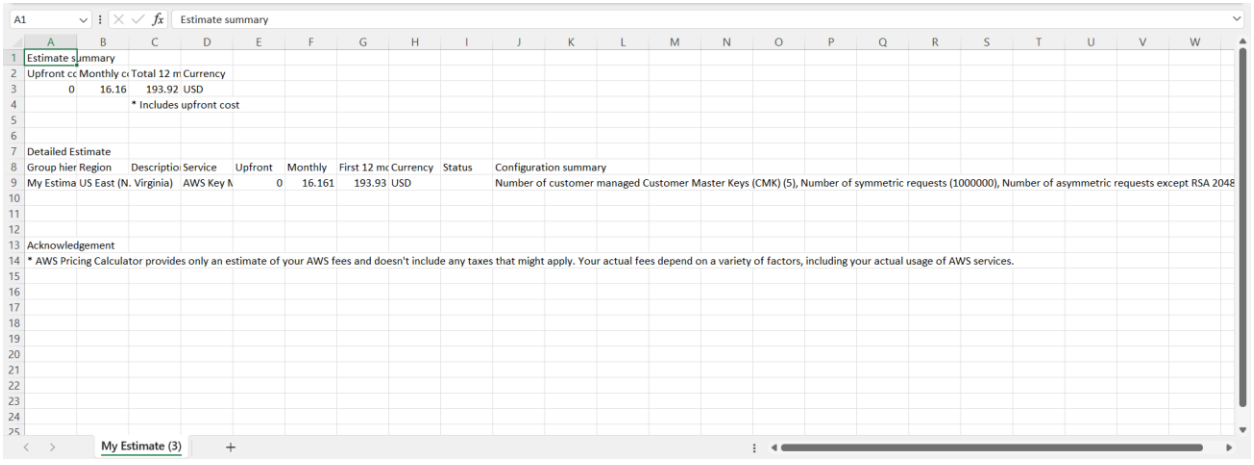
-Invoking and retrieve the secret:





Cost assessment for using AWS KMS:





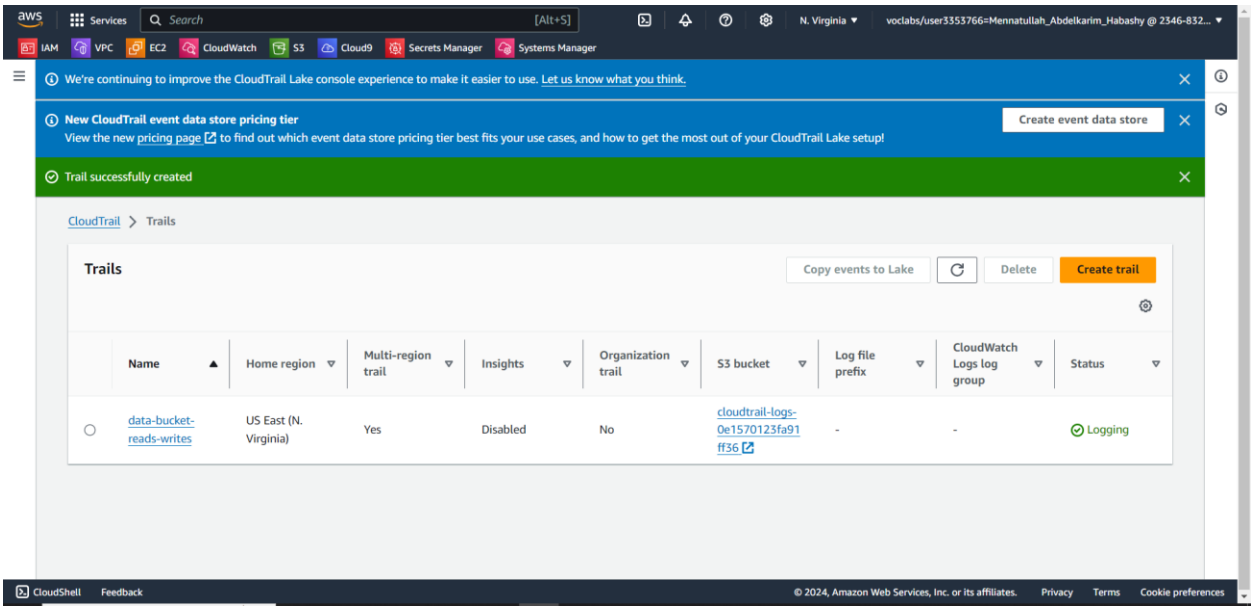
Phase 4: Monitoring and logging:

in this phase The solution needs to do the following:

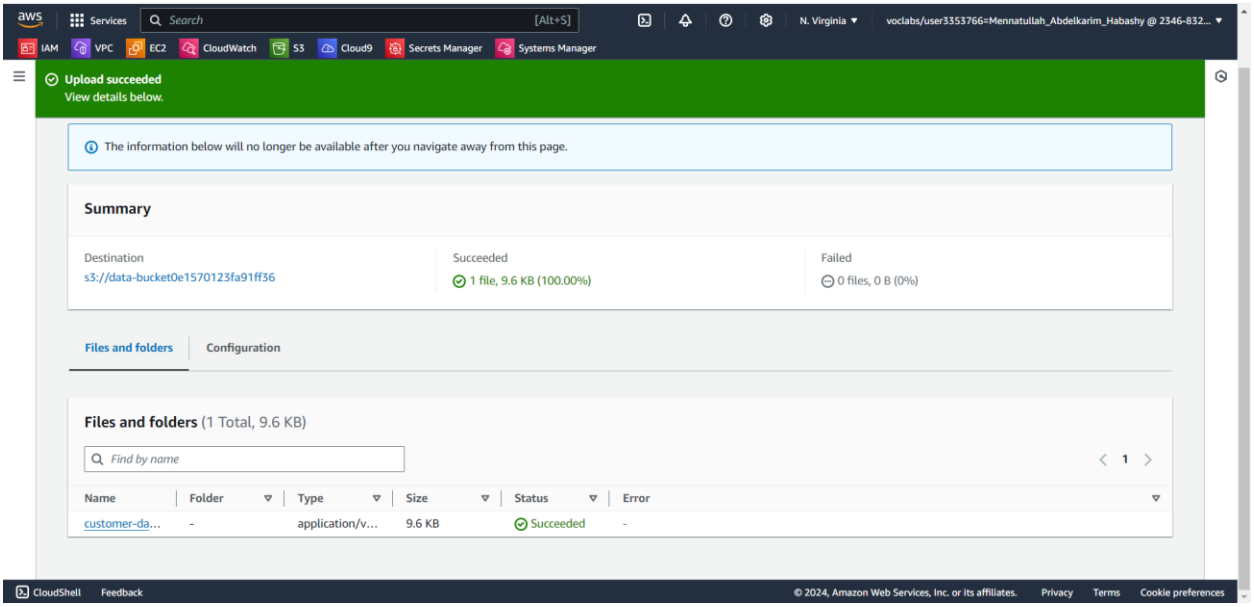
- Track all API calls to S3 buckets.
- Monitor application logs.
- Notify team members in case of security incidents.
- Monitor AWS resource configurations and automatically modify configurations that are out of compliance.

Task 4.1: Use CloudTrail to record Amazon S3 API calls:

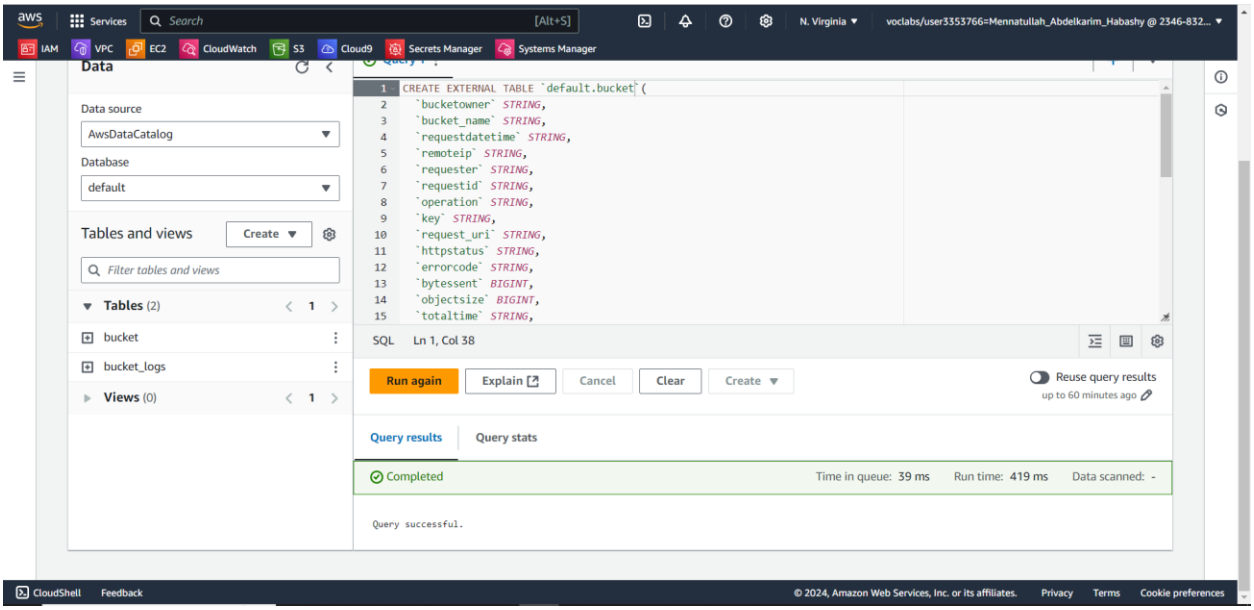
-Create a CloudTrail trail:



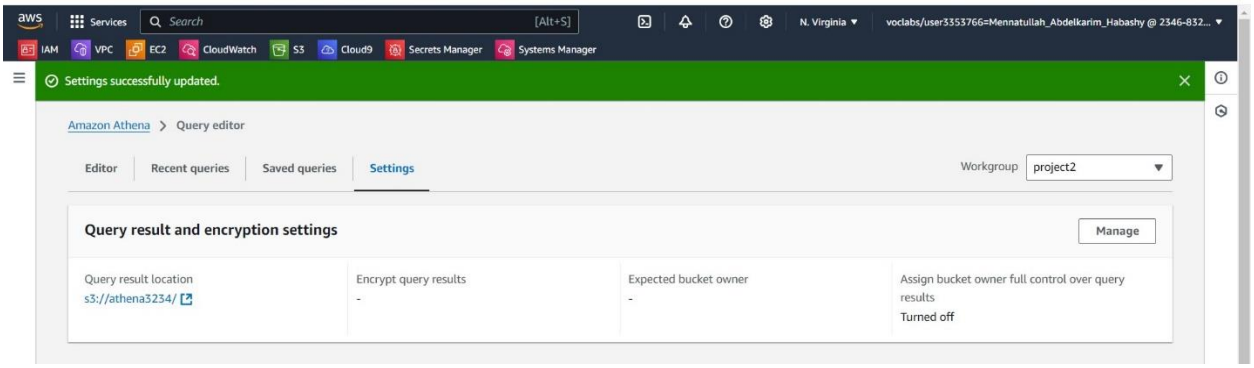
-Upload the *customer-data.csv* file to *data-bucket*:

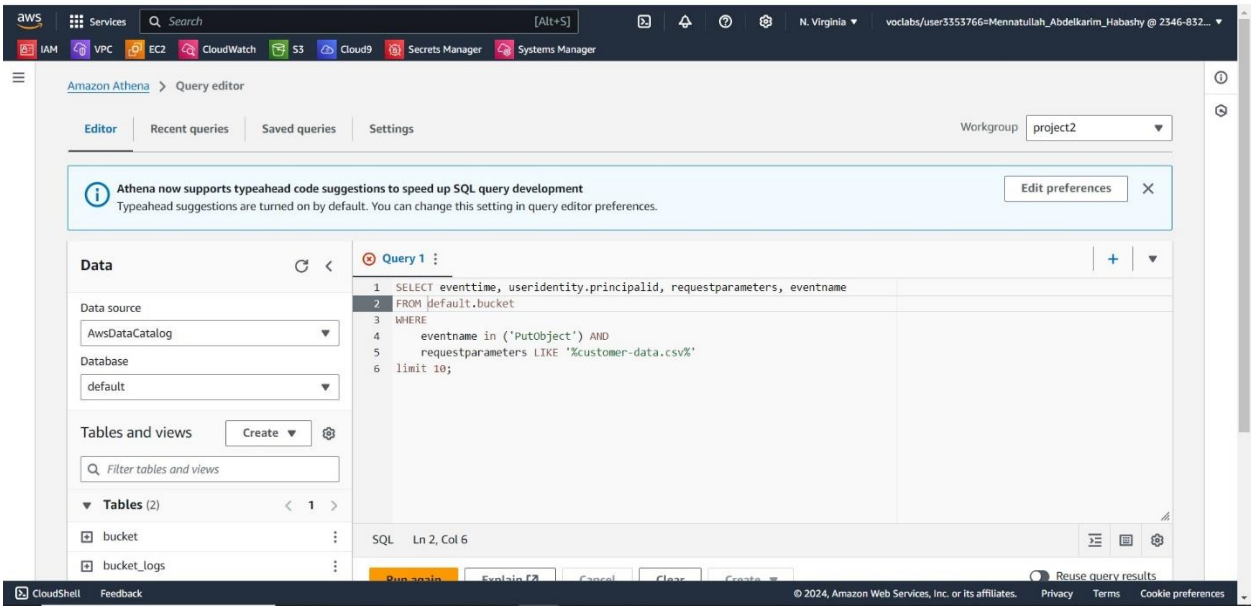


-Use the CloudTrail console to create an Athena table:



-Run an Athena query to retrieve the CloudTrail event log data in Result area:

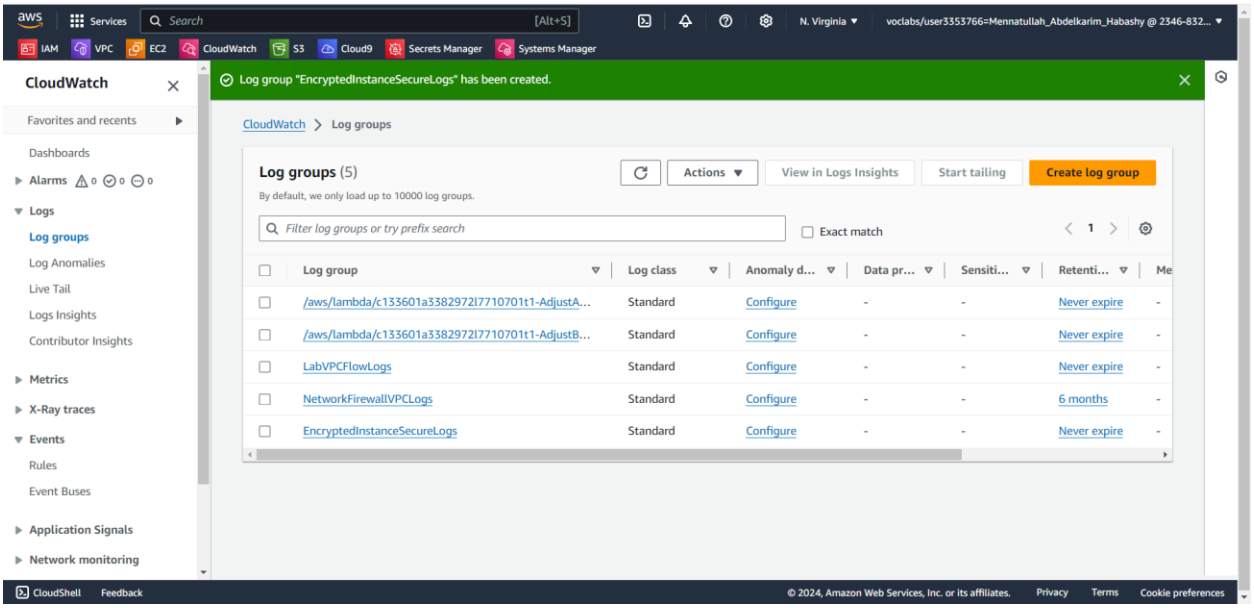




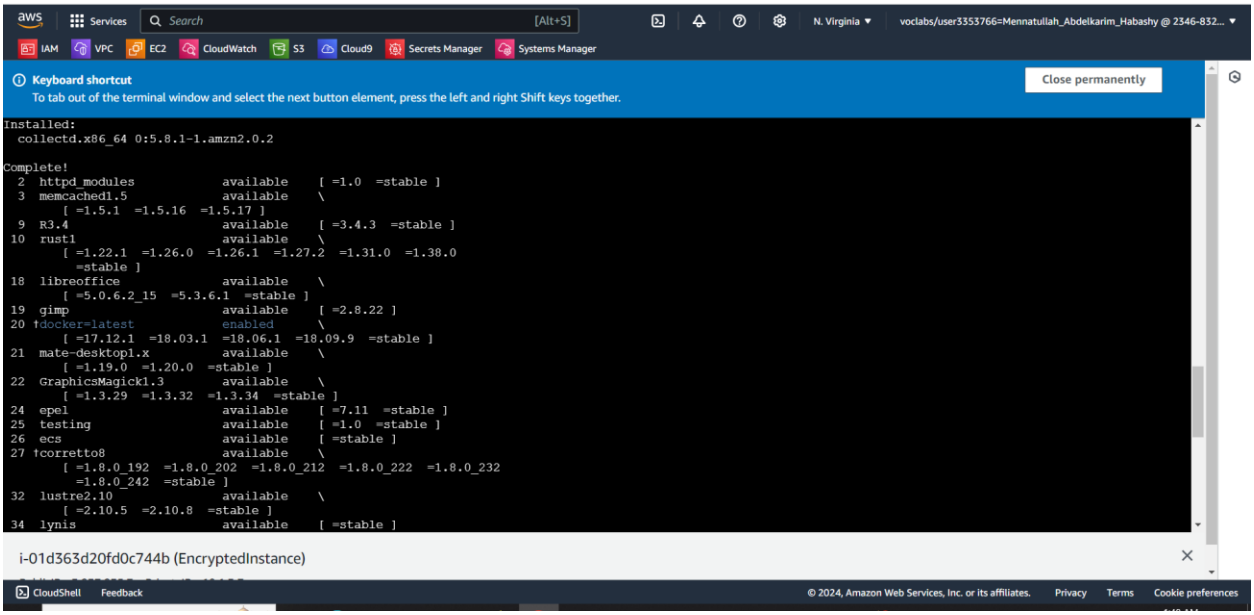
Task 4.2: Use CloudWatch Logs to monitor secure logs

In this task I have been create a solution to monitor access to *EncryptedInstance*.

-Create a CloudWatch log group with all default settings:

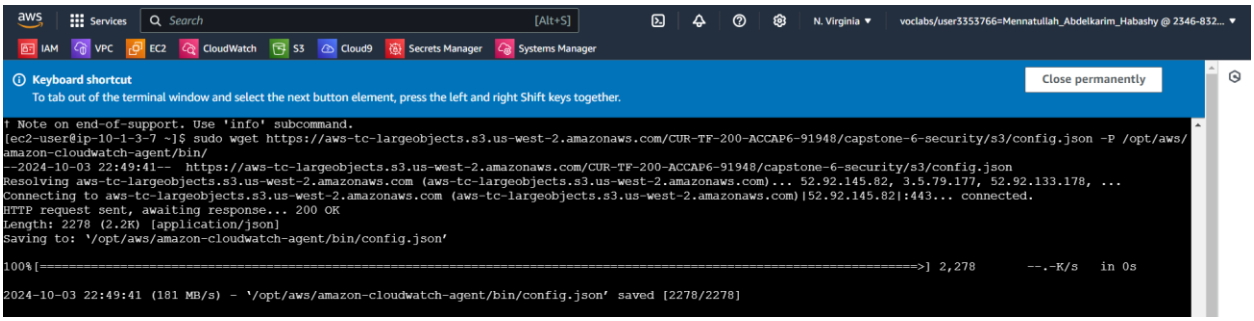


-install the CloudWatch agent and a Linux daemon named collectd, which the CloudWatch agent will use:

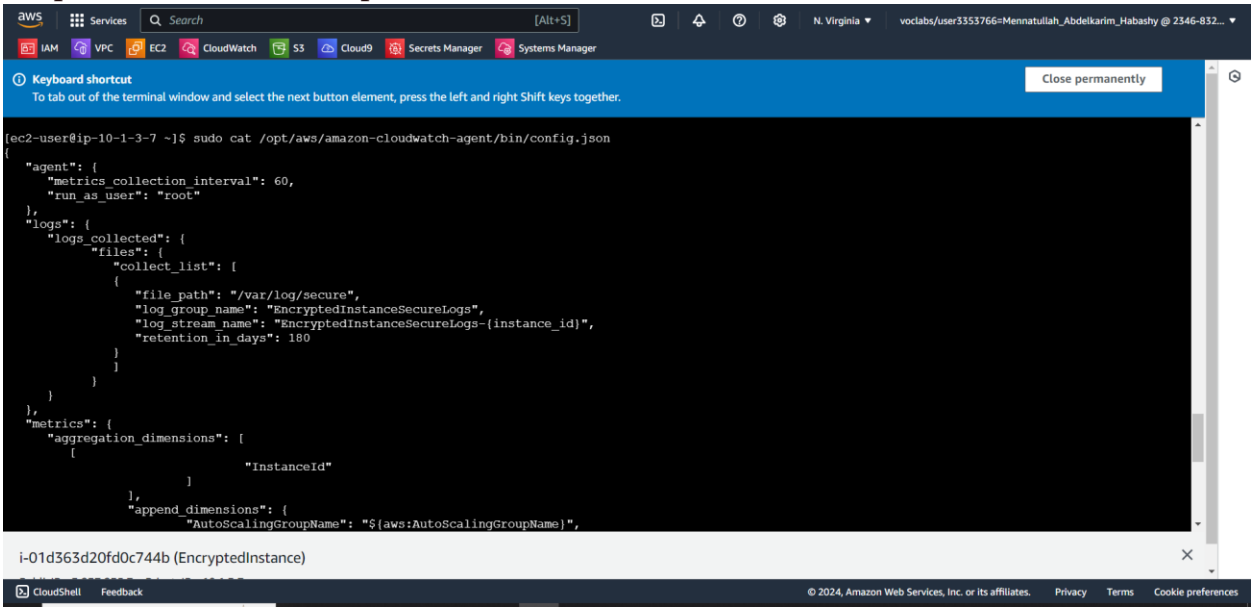


-Download and configure a JSON file that provides configuration details for the CloudWatch agent.

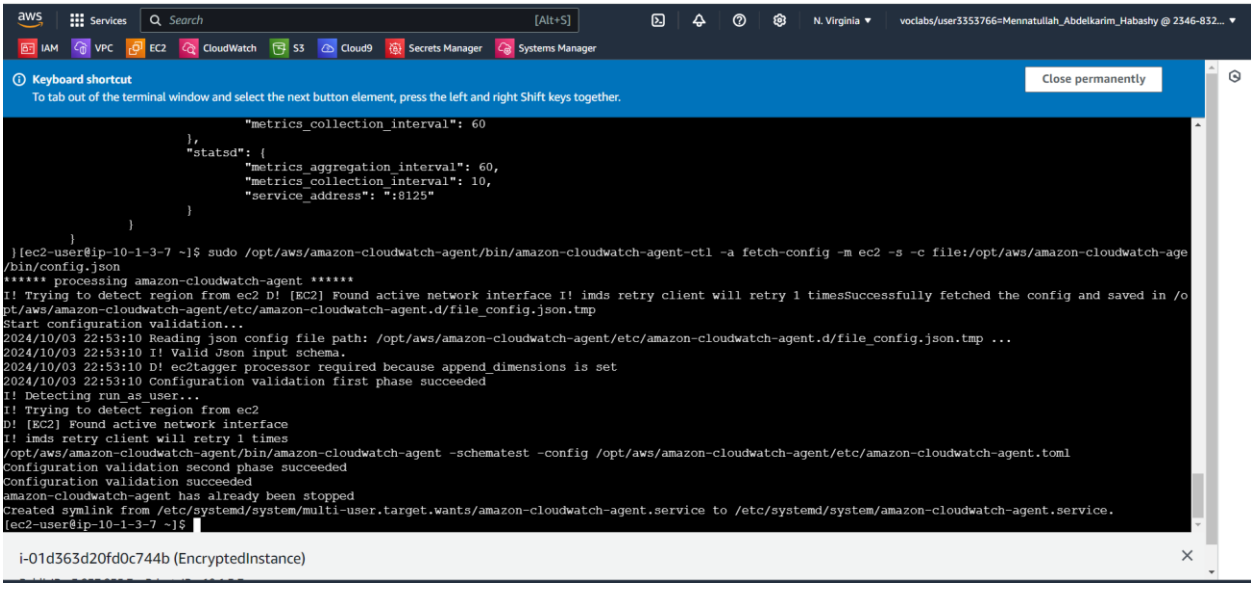
- To download the template file:



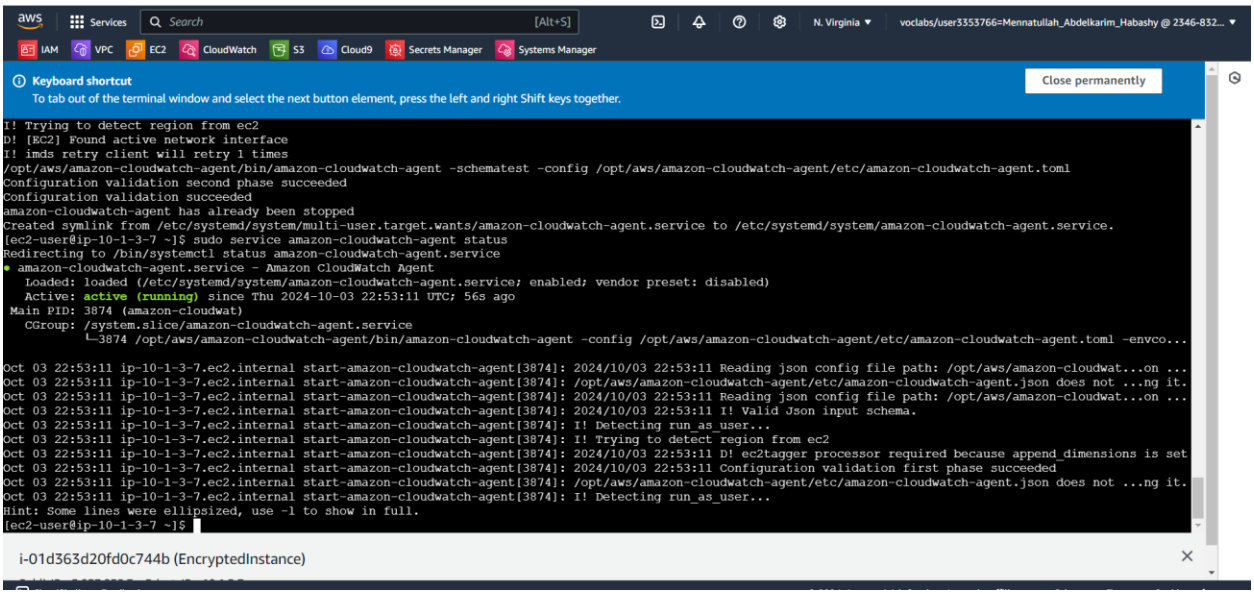
- To print out the file template:



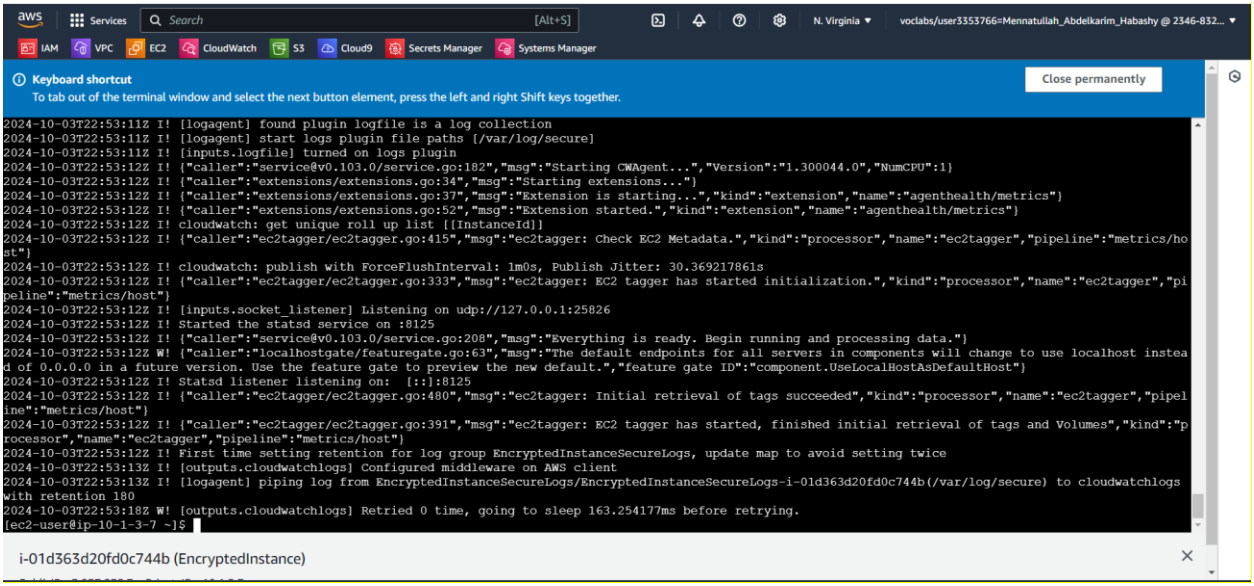
- Start the CloudWatch agent and confirm that it is running:



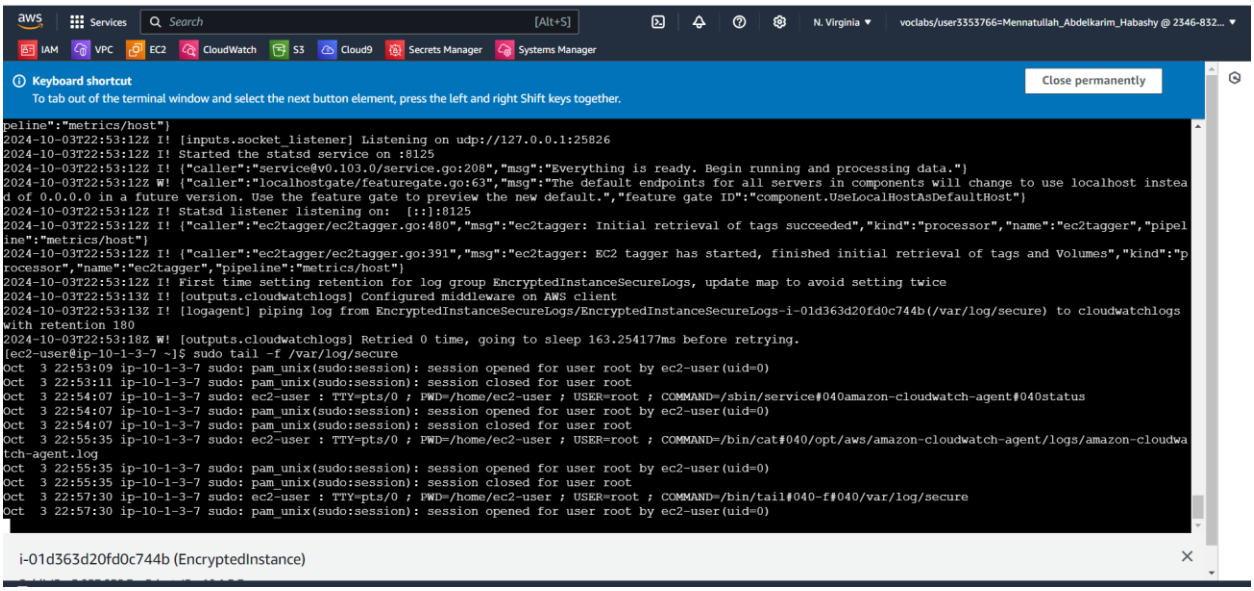
- To verify the CloudWatch agent status:



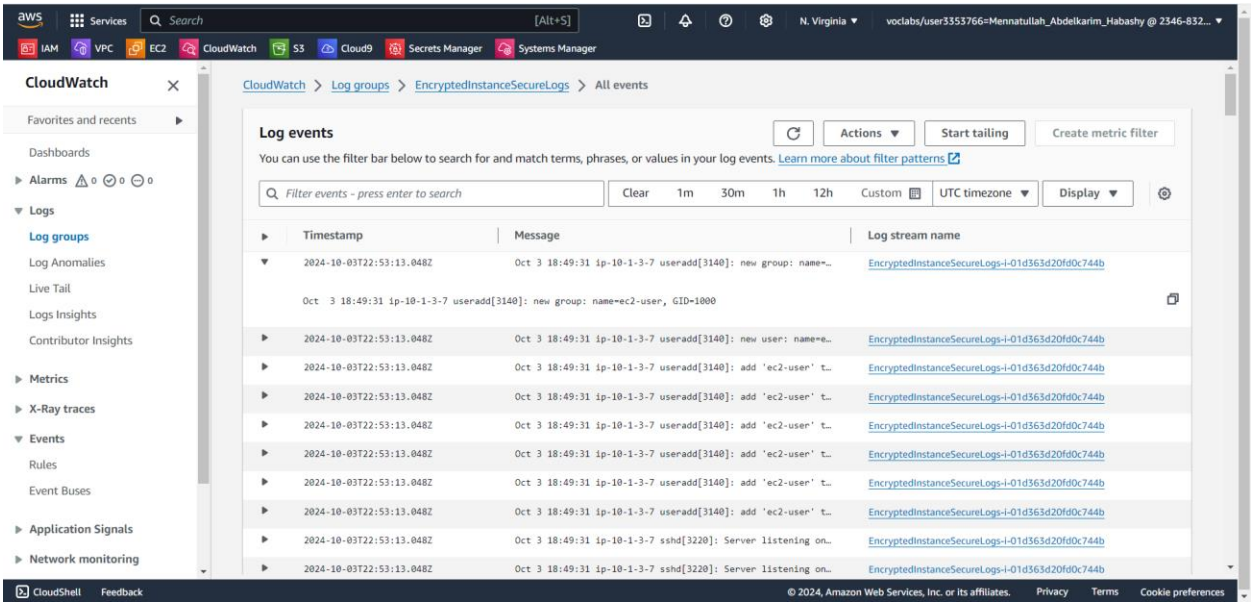
- To confirm that the CloudWatch agent is able to reach the CloudWatch service in my AWS account:



- To actively tail the */var/log/secure* file:



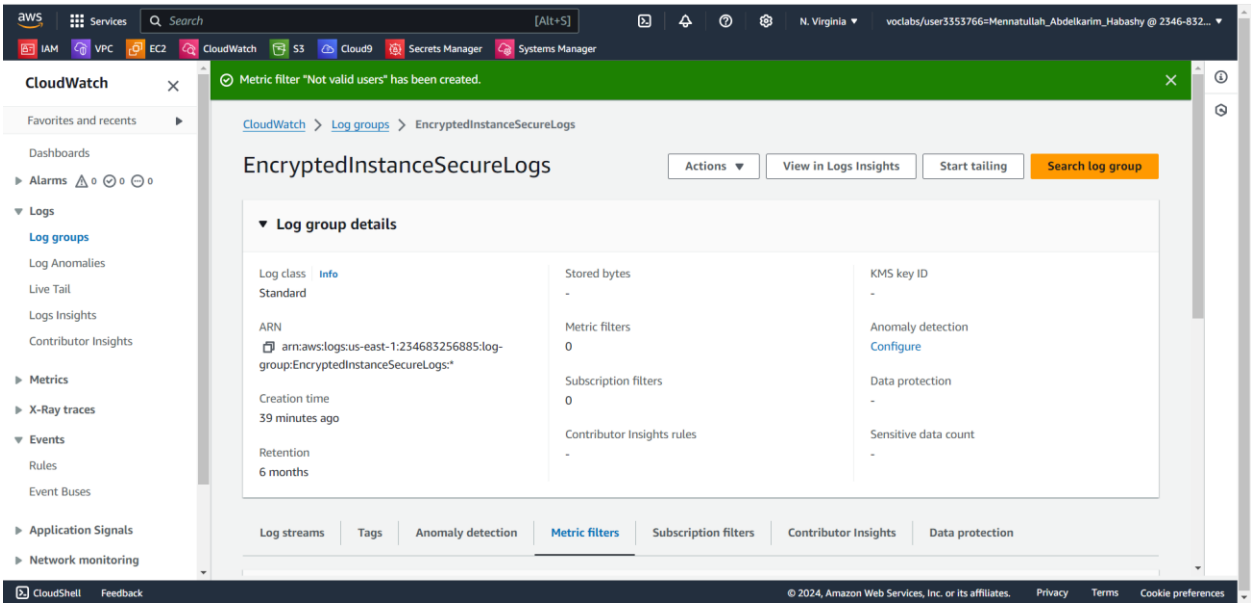
-Verify that the secure logs are being written to the CloudWatch log group:



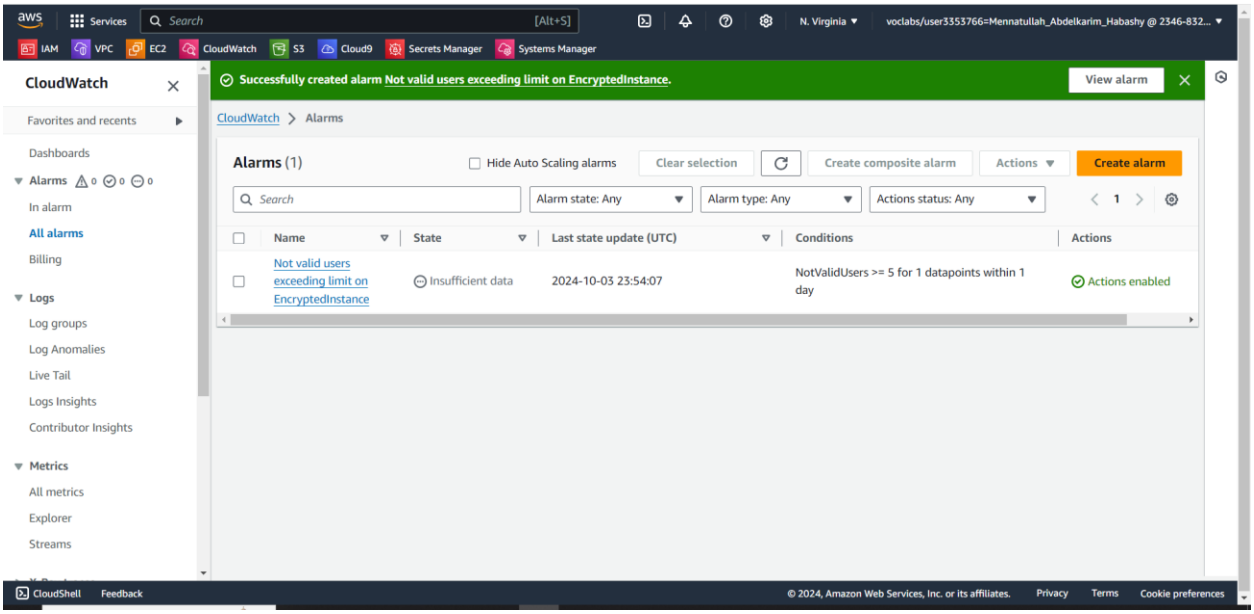
Task 4.3: Create a CloudWatch alarm to send notifications for security incidents

In this task, I will create a CloudWatch alarm to notify these team members when such an incident occurs.

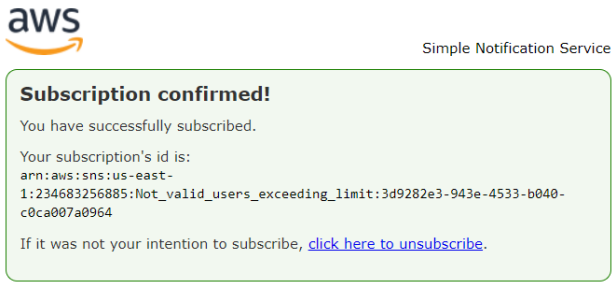
-create a metric filter:



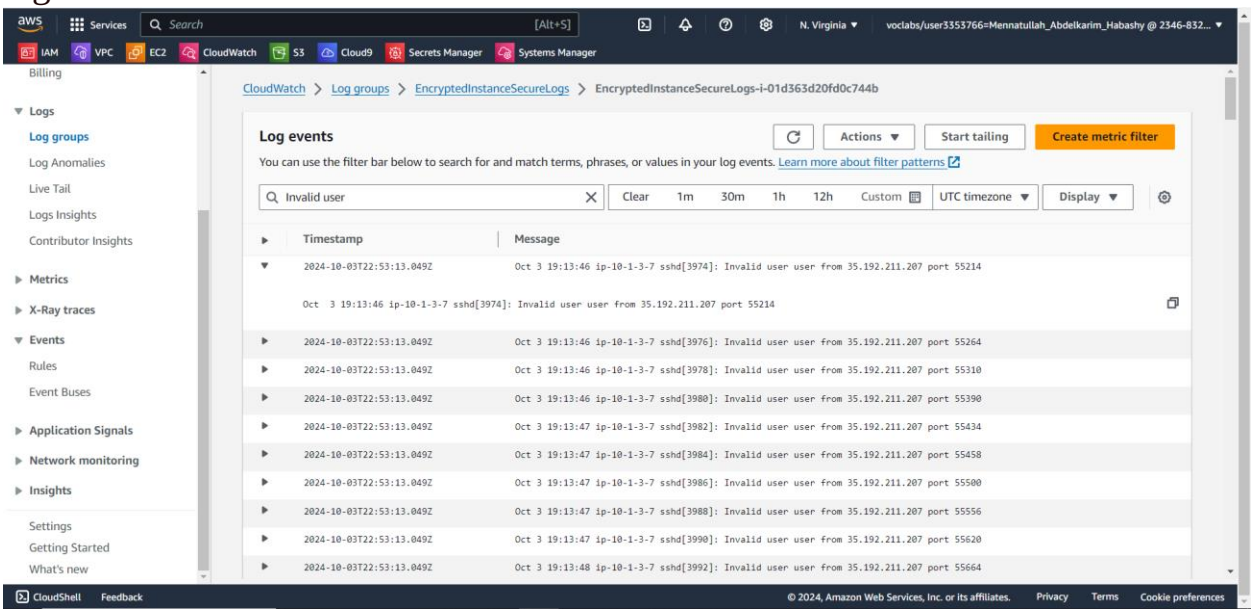
-Create a CloudWatch alarm from the metric filter:



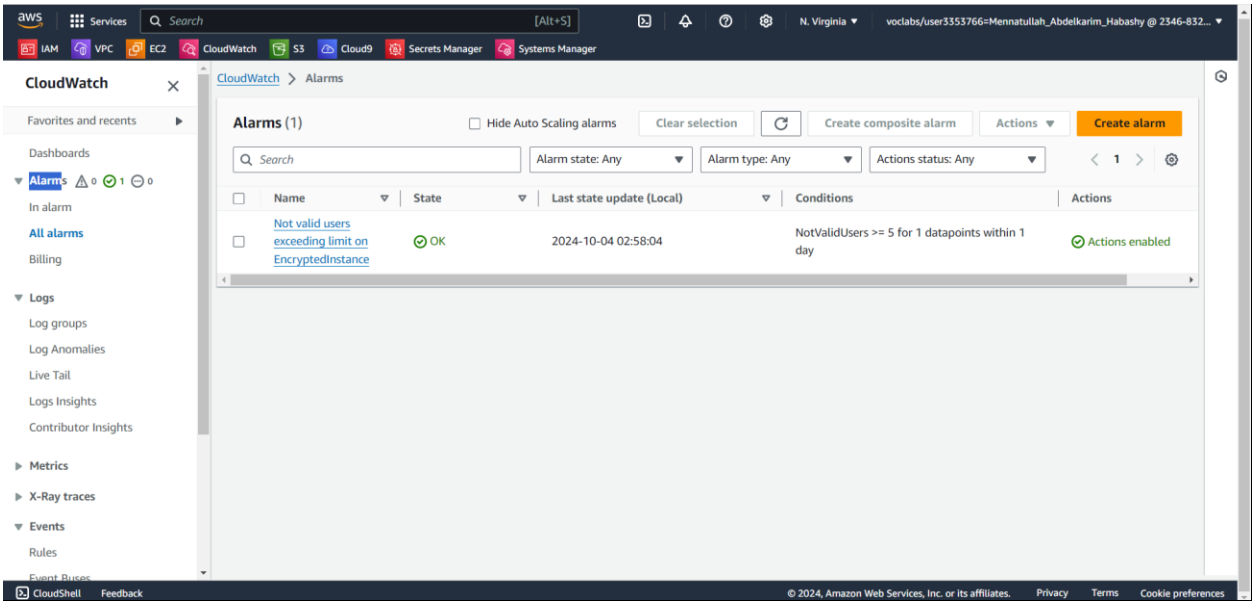
-**Confirm subscription** link.



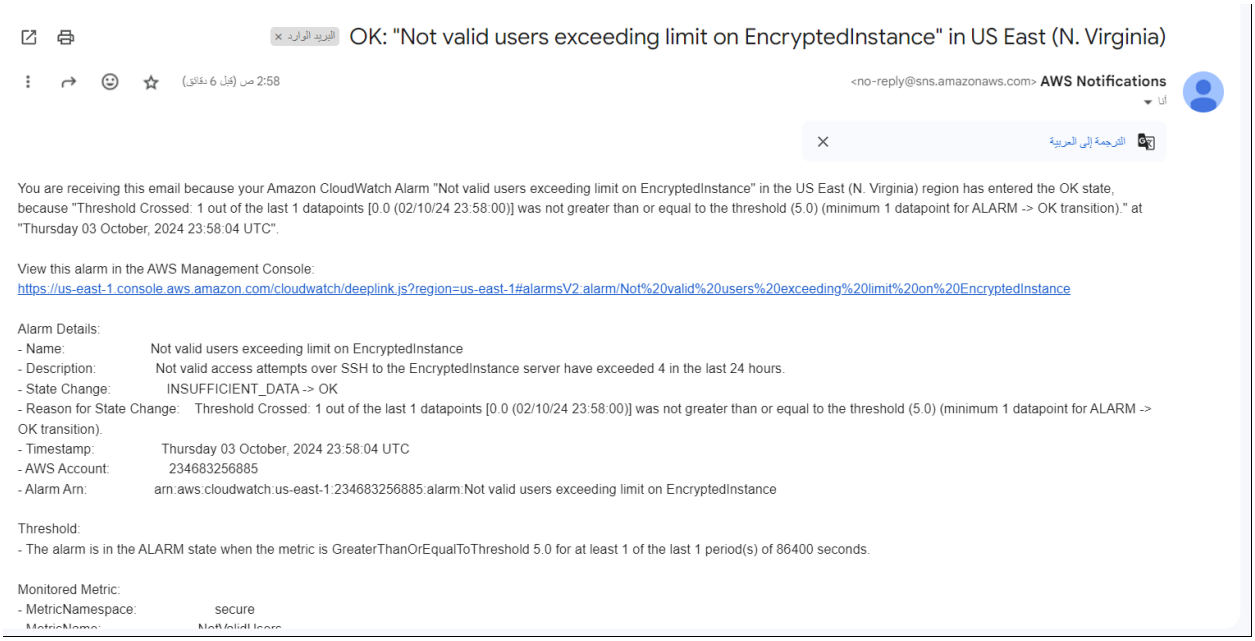
-in the latest log stream for the *EncryptedInstanceSecureLogs* log group, filter the log events for Invalid user:



-confirm that the *Invalid users* alarm has a state of *In alarm*:



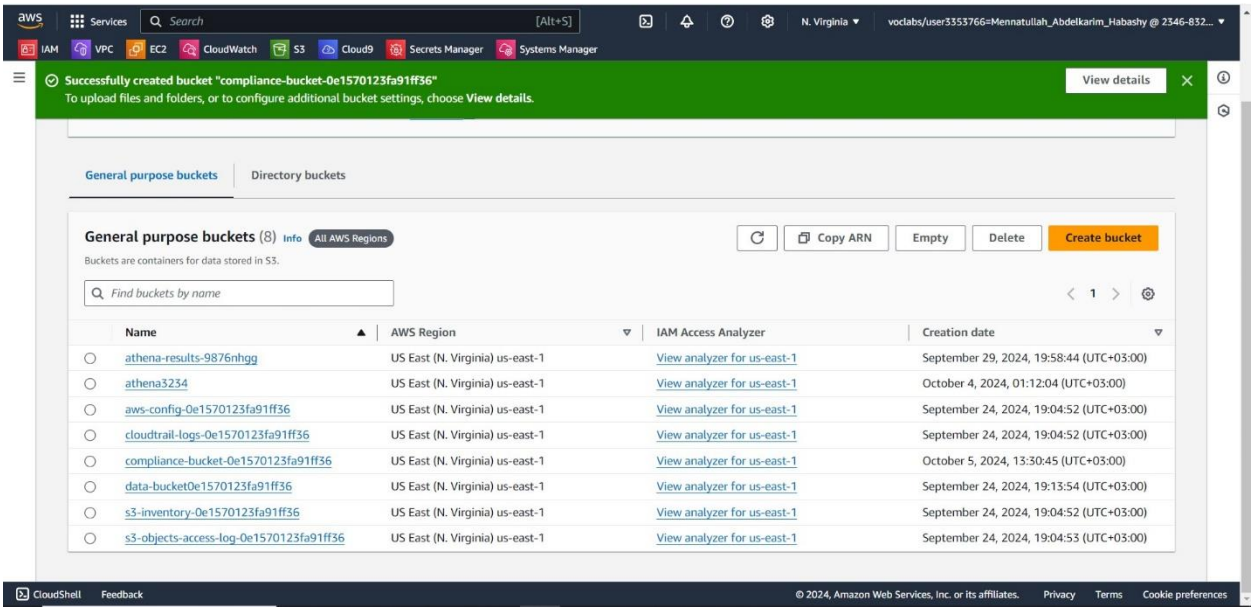
Checking email to confirm that I received a notification that the alarm threshold was crossed:



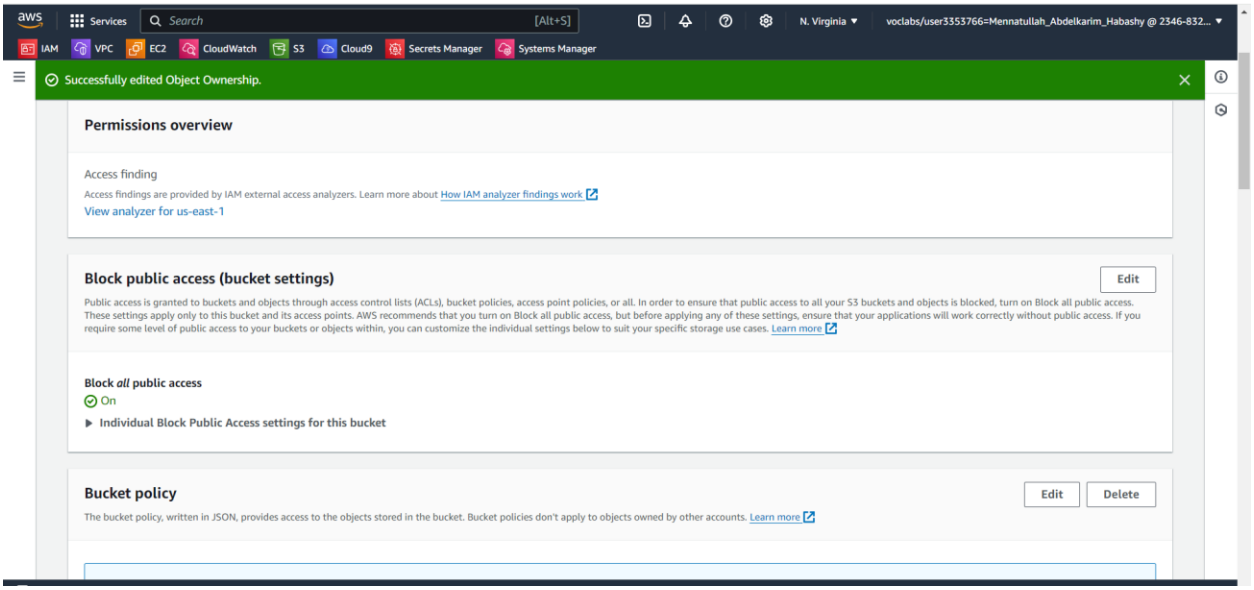
Task 4.4: Configure AWS Config to assess security settings and remediate the configuration of AWS resources.

In this task I'm challenged to use AWS Config to report on whether object logging is configured on the S3 buckets and configure an automation script to remediate noncompliance.

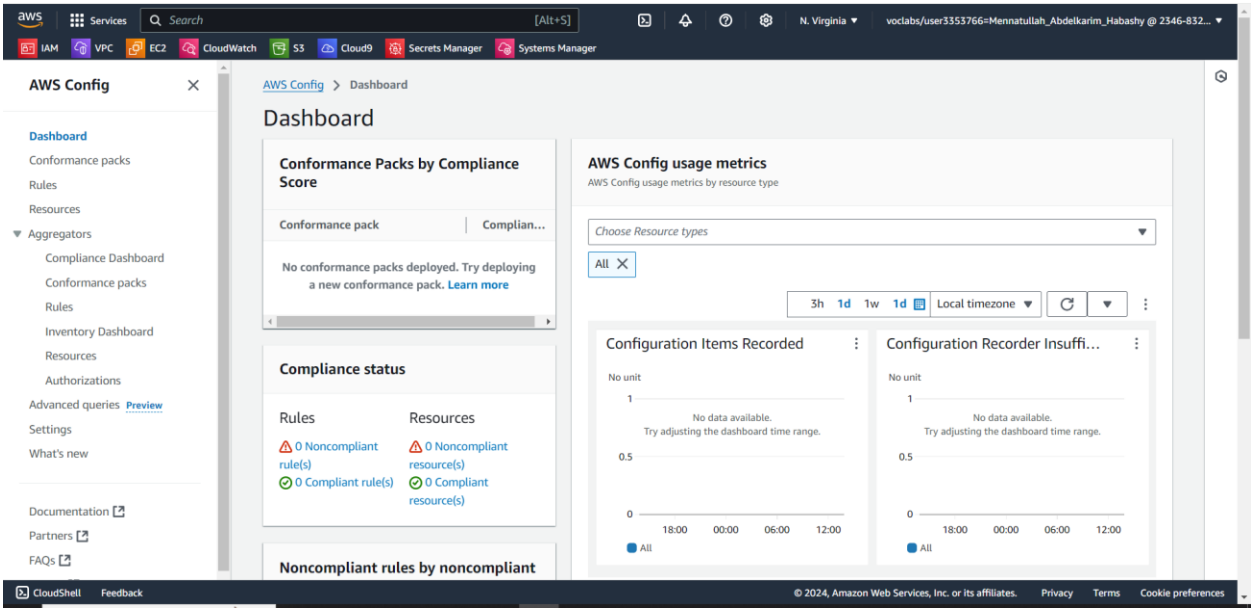
-Create a new S3 bucket named compliance-bucket-unique-ID:



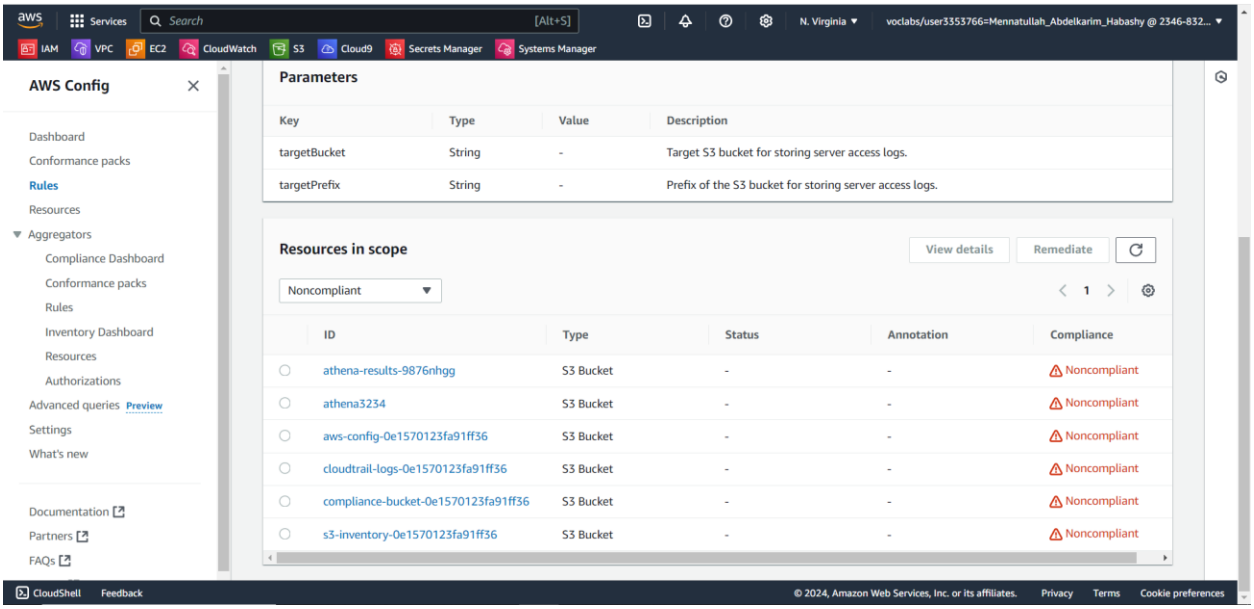
-Enable object ownership on the *s3-objects-access-log* bucket:



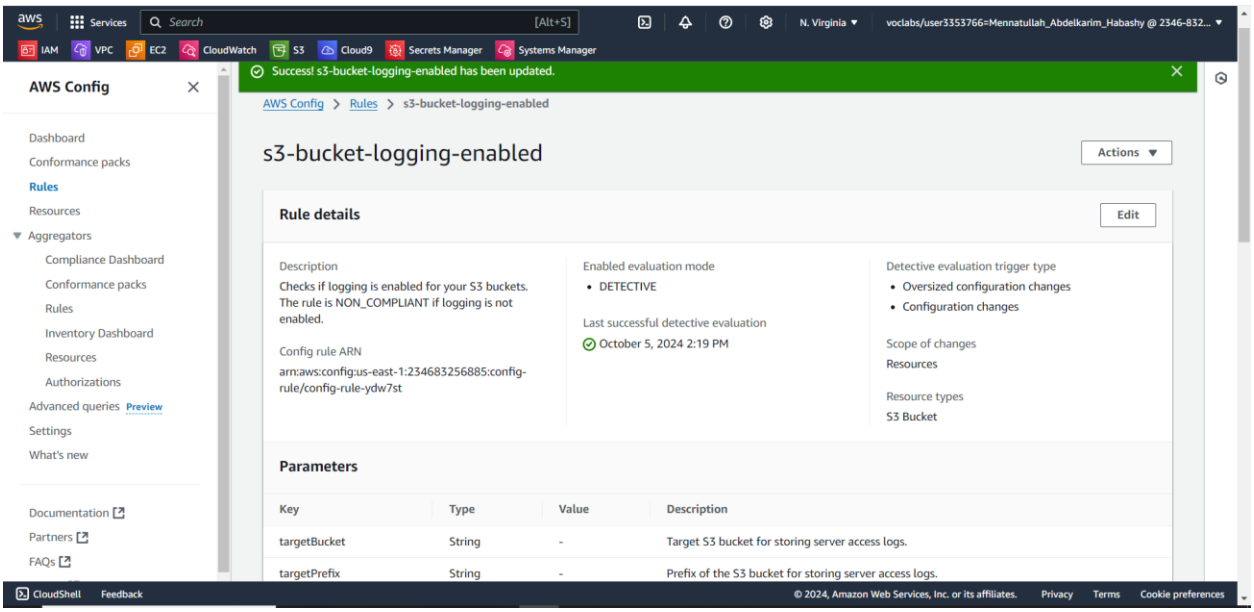
-Set up AWS Config:

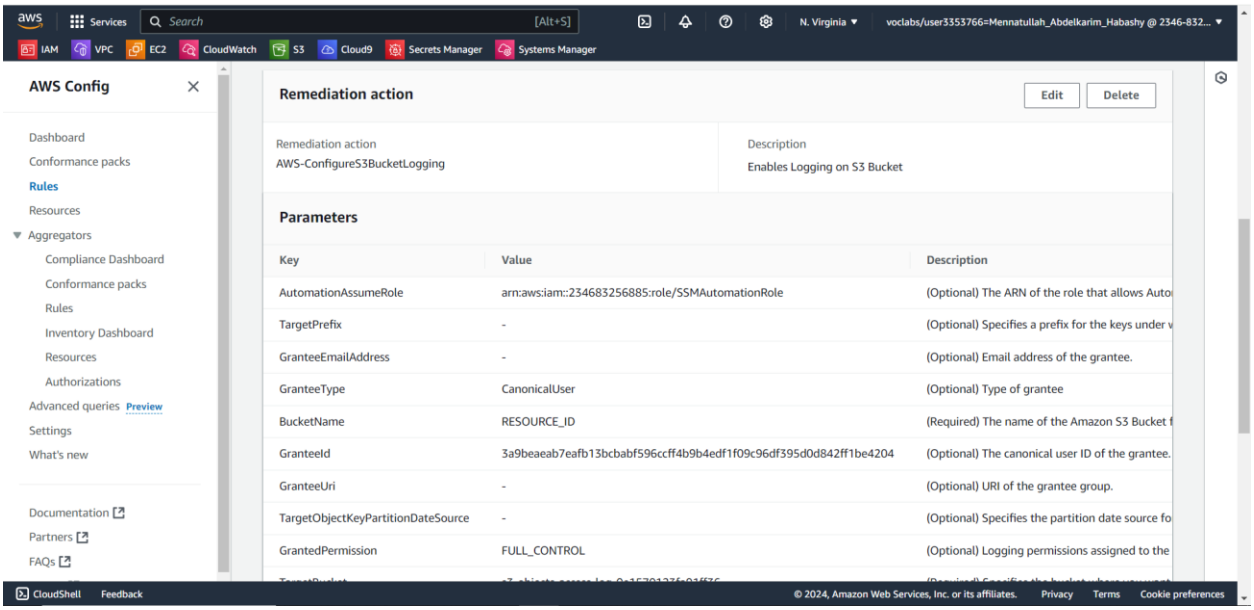


-Confirm that the *s3-bucket-logging-enabled* rule is finding resources that are in scope. Also, confirm that the *compliance-bucket* is flagged as noncompliant:

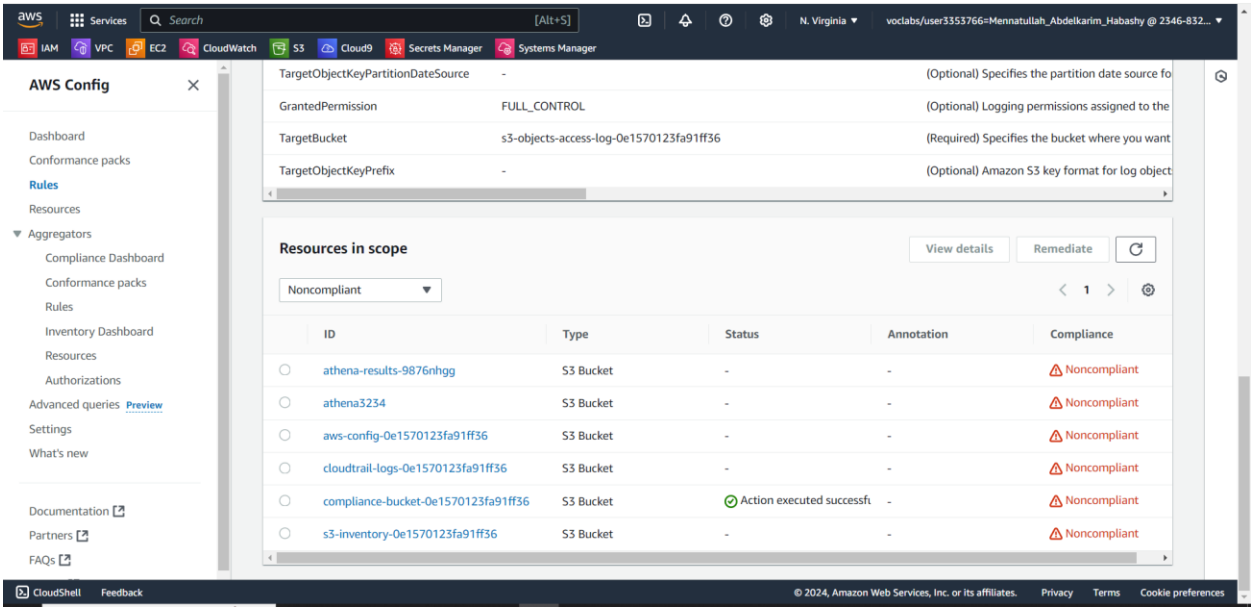


-Configure manual remediation settings for the *s3-bucket-logging-enable* rule:

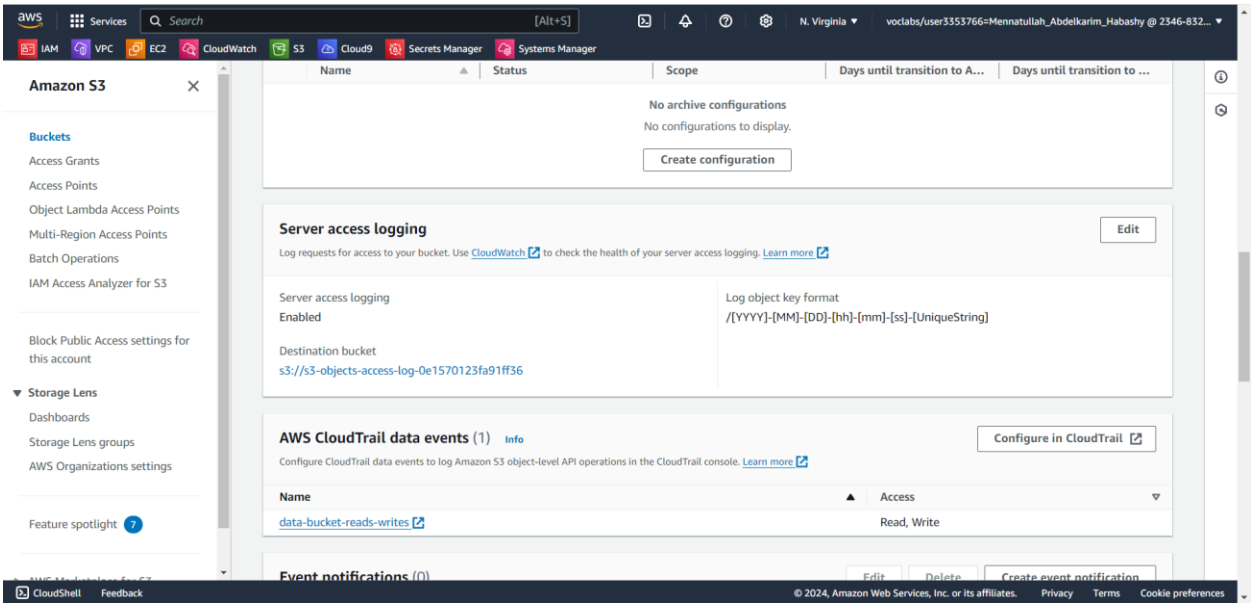




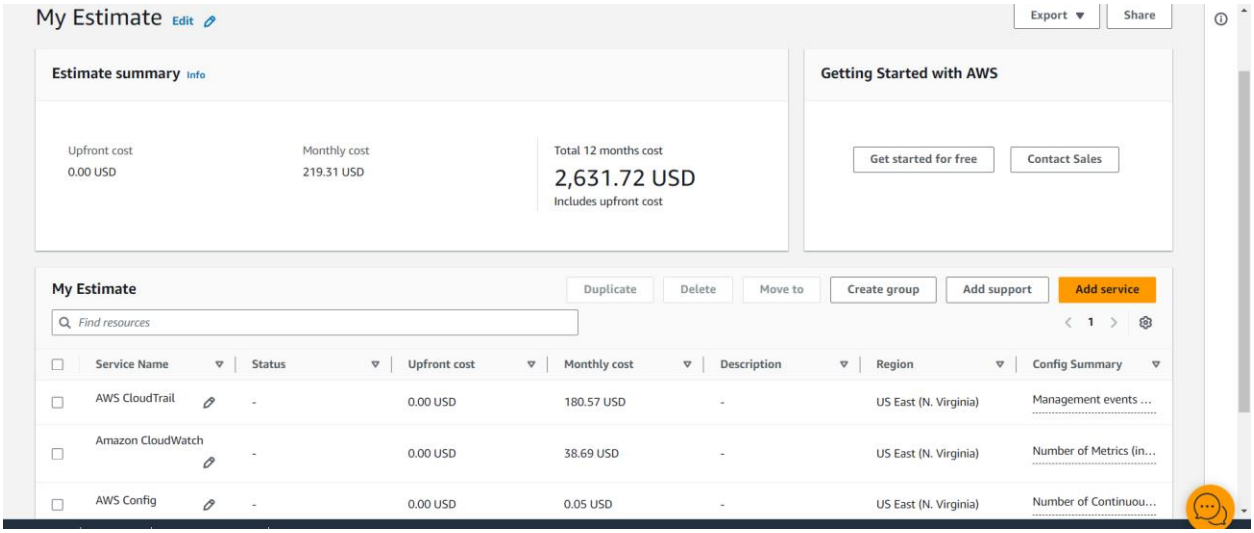
-Invoke the AWS Config remediation action:

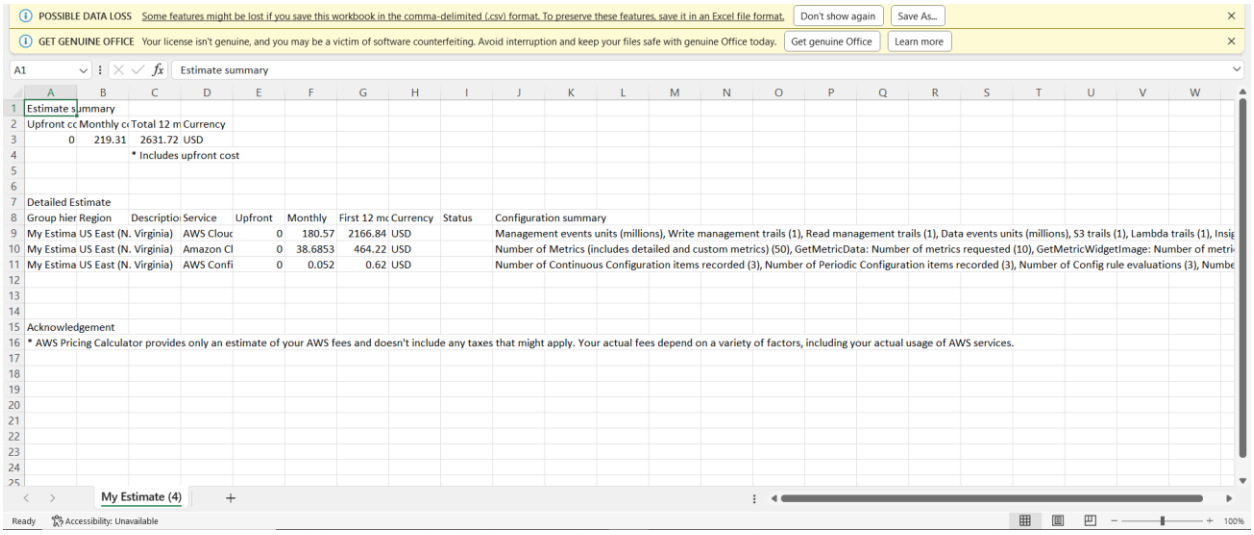


-Confirm that server access logging is now enabled on the *compliance-bucket*:



Cost assessment for monitoring and logging:





And Finally this is My badge from AWS Academy ,

https://www.credly.com/go/4ITNsuea