Cybersecurity Cohort Capstone: Technical Documentation (Sprints 1-5)

Project Goal

This project aims to establish a scalable and automated Security Operations Center (SOC) on AWS to centralize security monitoring, incident detection, and response capabilities. Secondary goals include improving security posture and reducing response times to security incidents.

Target Audience

This SOC platform is designed for security operations professionals, IT security teams, and security analysts within an organization.

Success Metrics

- Reduce alert fatigue by at least 30% within six months of SOC deployment.
- Increase incident response times by 20%.
- Achieve an average incident response time of less than 15 minutes for critical security events.
- Improve security visibility with centralized monitoring and reporting capabilities.
- Streamline incident response workflows through automation playbooks and improve overall response efficiency.

Architecture Design

Data Sources

- Network Security Devices (pfSense firewall)
- Cloud Infrastructure Logs (VPC Flow Logs, DNS Logs, CloudTrail Logs)

Log Ingestion and Preprocessing

- Amazon Kinesis Firehose: Collects logs from various sources and prepares them for further analysis by Splunk Cloud.

Log Management and Analysis

- Splunk Cloud (SIEM): Analyzes logs for security events, anomalies, and potential threats. Integrates with EOpen threat intelligence feed.

Threat Detection and Vulnerability Assessment

- Amazon GuardDuty: Monitors your AWS environment for malicious activity.
- Amazon Inspector: Scans EC2 instances for vulnerabilities and exposures.

Security Orchestration, Automation, and Response (SOAR)

- AWS Lambda (Automated Responses): Creates custom functions for repetitive tasks (e.g., quarantining infected instances).
- AWS Security Hub (Optional): Central repository for security findings from various AWS services.

Security Visualization

- Kibana (integrated with Splunk Cloud): Provides real-time dashboards and reports for security posture awareness.

Threat Intelligence Feed

- Emerging Threats Open (ETOpen) - Open Source Tool: Provides valuable threat intelligence data to Splunk Cloud.

IAM Roles with Least Privilege

- IAM ensures secure access control by assigning roles with least privilege to users and resources.

How They Work Together

1. **Log Collection:** Logs are generated from various sources and collected by Kinesis Firehose.
2. **Log Preprocessing and Delivery:** Kinesis Firehose preprocesses logs and delivers them to Splunk Cloud.
3. **Log Analysis and Threat Detection:** Splunk Cloud analyzes logs and leverages threat intelligence to identify potential threats.
4. **Alert Generation:** If suspicious activity is detected, Splunk Cloud generates security alerts.
5. **Investigation and Response:** Security analysts investigate alerts and determine appropriate responses.
6. **Automation (Optional):** AWS Lambda functions can be triggered for automated responses to high-severity alerts.
7. **Centralized Findings (Optional):** Security Hub can act as a central repository for findings, aiding in comprehensive investigation.
8. **Security Visualization:** Kibana provides real-time visualizations of security data for situational awareness.

Alternative: AWS OpenSearch Service

OpenSearch is an open-source alternative to Splunk Cloud for log analysis and works well with Kibana for visualization.

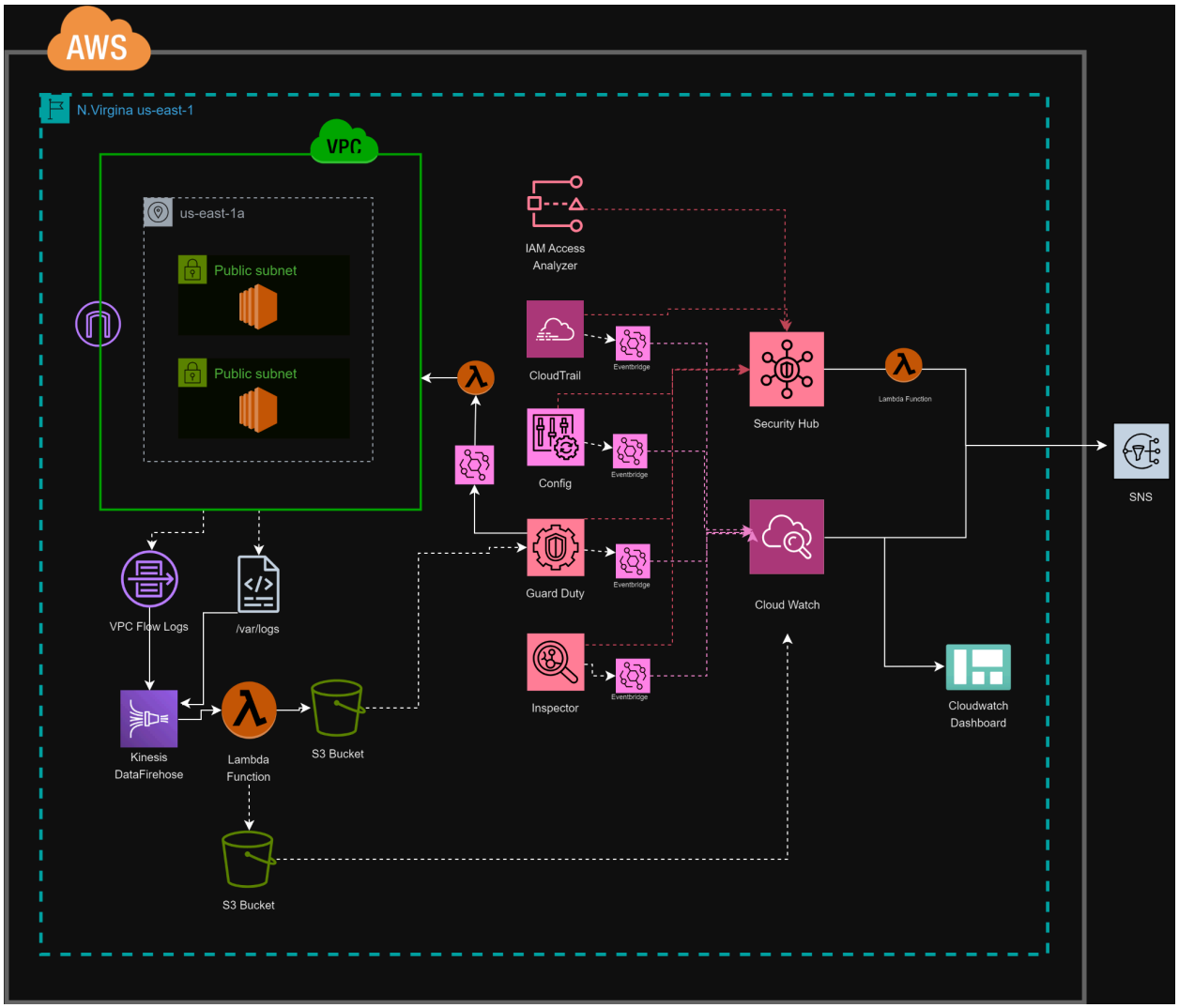
Research Highlights

- Optimizing data ingestion for threat coverage is crucial for effective SOC operations.
- Accurate data intake is essential for successful SOC functionality.
- Transitioning from basic log collection to true threat detection and response is important.

Additional Resources

- AWS re:Inforce 2022: Using AWS security services to build your cloud security operations baseline.
- SOCGUIDE: https://github.com/cyb3rxp/awesome-soc

Network Diagram



IAM Policies and Configuration

- IAM policies define access controls for users within the SOC, adhering to the principle of least privilege for security.
- Role-Based Access Control (RBAC) is implemented to assign permissions based on job responsibilities.

Security Data Ingestion Pipelines

This section outlines the security data ingestion pipelines for the SOC platform, covering data sources, ingestion architecture, automation and scalability, error handling, log management and normalization, and access control.

- **Data Source Identification:**
    - Cloud Service Logs (VPC Flow Logs, GuardDuty Findings)
    - Network Security Appliance Logs (Firewalls, IPS/IDS, WAF)
- **Data Ingestion Architecture:**
    - Log Forwarding Mechanisms (Amazon EventBridge, CloudTrail)
    - Data Ingestion Services (Amazon Kinesis Data Streams

Testing and Validation

This sprint focuses on testing and validating the functionalities developed in previous stages.

Testing Scenarios and Playbook Validation

- Develop test scenarios simulating security incidents. These scenarios could involve injecting test data into CloudTrail logs or VPC Flow Logs to trigger playbooks and validate their functionality.
- Ensure playbooks function as intended and produce the desired outcomes during simulated attacks.

Security of Playbooks

- Implement access controls and permissions within the SIEM solution to restrict unauthorized modification of playbooks. Regularly review and update playbooks to ensure they remain effective and don't introduce vulnerabilities.

Threat Detection & Alert Correlation

SIEM Solution and Data Sources

- **SIEM Solution:** We'll leverage Amazon CloudWatch Logs as the central log management solution. CloudTrail, VPC Flow Logs, and Inspector findings can all be integrated with CloudWatch Logs for centralized analysis.
- **Data Sources:** Data sources include CloudTrail logs (API activity), VPC Flow Logs (network traffic), and Inspector findings (vulnerability scans).

Normalization and Enrichment

- CloudWatch Logs offers log parsing capabilities to normalize logs from different sources into a common format.
- Additional enrichment might involve integrating threat intelligence feeds to add context to log data.

Correlation Rules and Threat Detection

- Develop rules within CloudWatch Logs that analyze log data from various sources to identify potential threats. These rules might look for specific patterns in CloudTrail logs indicating suspicious activity or correlate network traffic logs with potential vulnerabilities identified by Inspector.
- Techniques could include pattern matching (e.g., identifying known malicious commands in CloudTrail logs), statistical analysis (e.g., detecting unusual spikes in network traffic), or simple rule-based correlations.
- Integrate threat intelligence feeds from reputable providers to enrich correlation rules with the latest indicators of compromise (IoCs). This helps identify emerging threats and suspicious activity patterns.

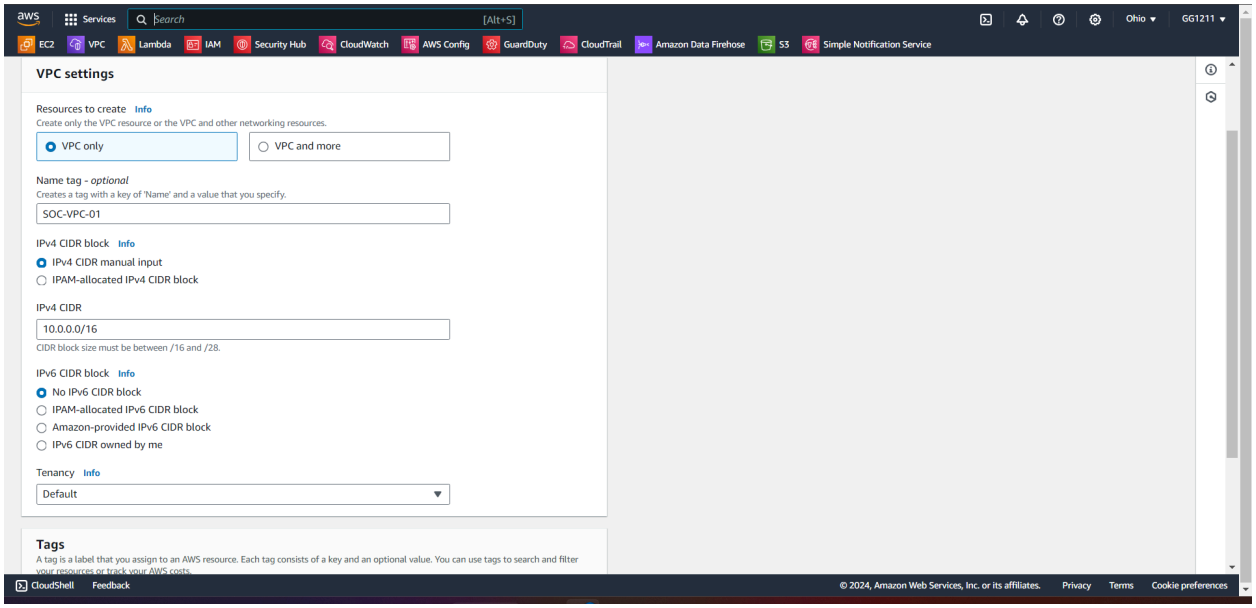
- Continuously monitor and refine correlation rules to reduce false positives and improve detection accuracy. Analyze false positives to understand the root cause and adjust rules accordingly.

Alert Prioritization and Triage

- Prioritize alerts based on a combination of factors like asset criticality (e.g., high-risk servers), threat level (e.g., potential data breach), and confidence score (e.g., high confidence based on multiple indicators).
- Alerts triggered by correlation rules can be routed to basic automation within the SIEM solution or integrated with a future SOAR platform for more complex workflows.
- Define high-fidelity alert scenarios (e.g., unauthorized login attempt from suspicious IP for a critical account) that trigger automated responses and notify security personnel for further investigation.

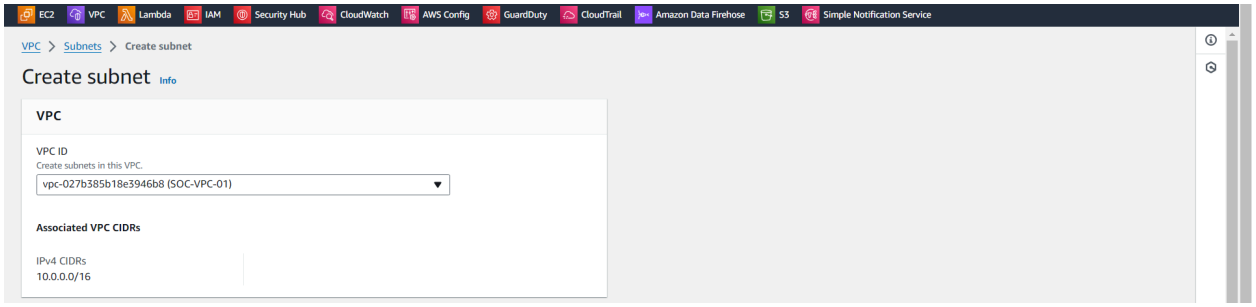
By incorporating these testing and validation steps, you can ensure your SOC platform effectively detects and responds to security threats.

Networking in AWS



Step 1: Create a VPC

1. Log in to the AWS Management Console.
2. Navigate to the VPC Dashboard:
   - Go to the "Services" dropdown and select "VPC".
3. Create a VPC:
   - Click on "Your VPCs" in the left-hand menu.
   - Click the "Create VPC" button.
   - Fill in the details:
     - Name tag: Provide a name for your VPC.
     - IPv4 CIDR block: Enter the IPv4 CIDR block for your VPC (e.g., `10.0.0.0/16`).
     - IPv6 CIDR block: Optional, can be left empty unless required.
     - Tenancy: Default.
   - Click "Create VPC".



Step 2: Create Subnets

1. Navigate to the Subnets section:
   - Click on "Subnets" in the left-hand menu.
2. Create Subnet 1:

- Click the "Create subnet" button.
- Select your VPC from the drop-down.
- Fill in the details for Subnet 1:
  - Name tag: Provide a name for the first subnet.
  - Availability Zone: Select an availability zone.
  - IPv4 CIDR block: Enter the CIDR block for Subnet 1 (e.g., `10.0.1.0/24`).
- Click "Create subnet".

3. Create Subnet 2:
- Click the "Create subnet" button again.
- Select your VPC from the drop-down.
- Fill in the details for Subnet 2:
  - Name tag: Provide a name for the second subnet.
  - Availability Zone: Select a different availability zone (optional).
  - IPv4 CIDR block: Enter the CIDR block for Subnet 2 (e.g., `10.0.2.0/24`).
- Click "Create subnet".

Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 2

Subnet name

Create a tag with a key of 'Name' and a value that you specify.

Computer-01

The name can be up to 256 characters long.

Availability Zone Info

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

US East (N. Virginia) / us-east-1a

IPv4 VPC CIDR block Info

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

10.0.0.0/16

IPv4 subnet CIDR block

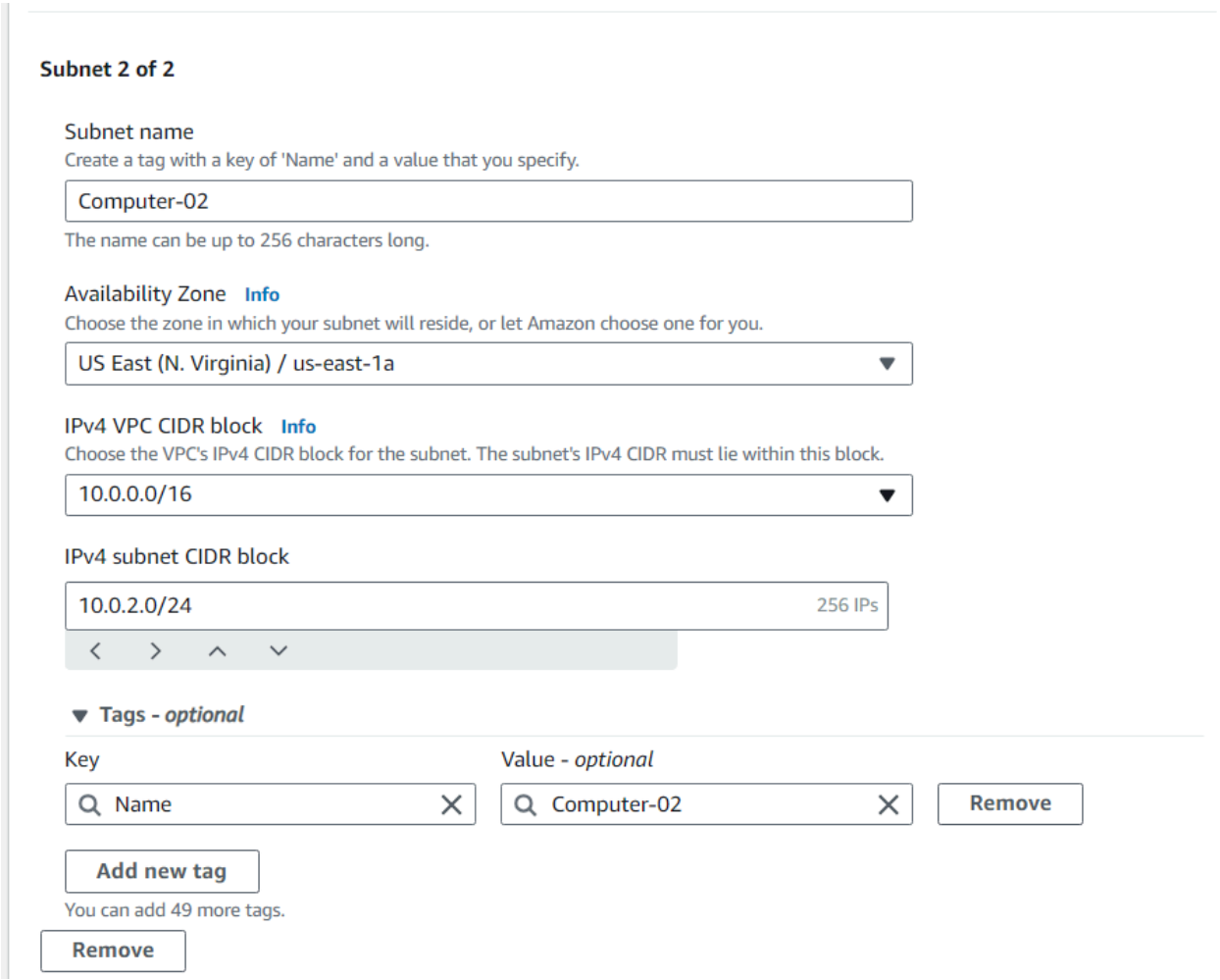
10.0.1.0/24 256 IPs

▼ Tags - *optional*

Key	Value - *optional*	
Name	Computer-01	Remove

Add new tag

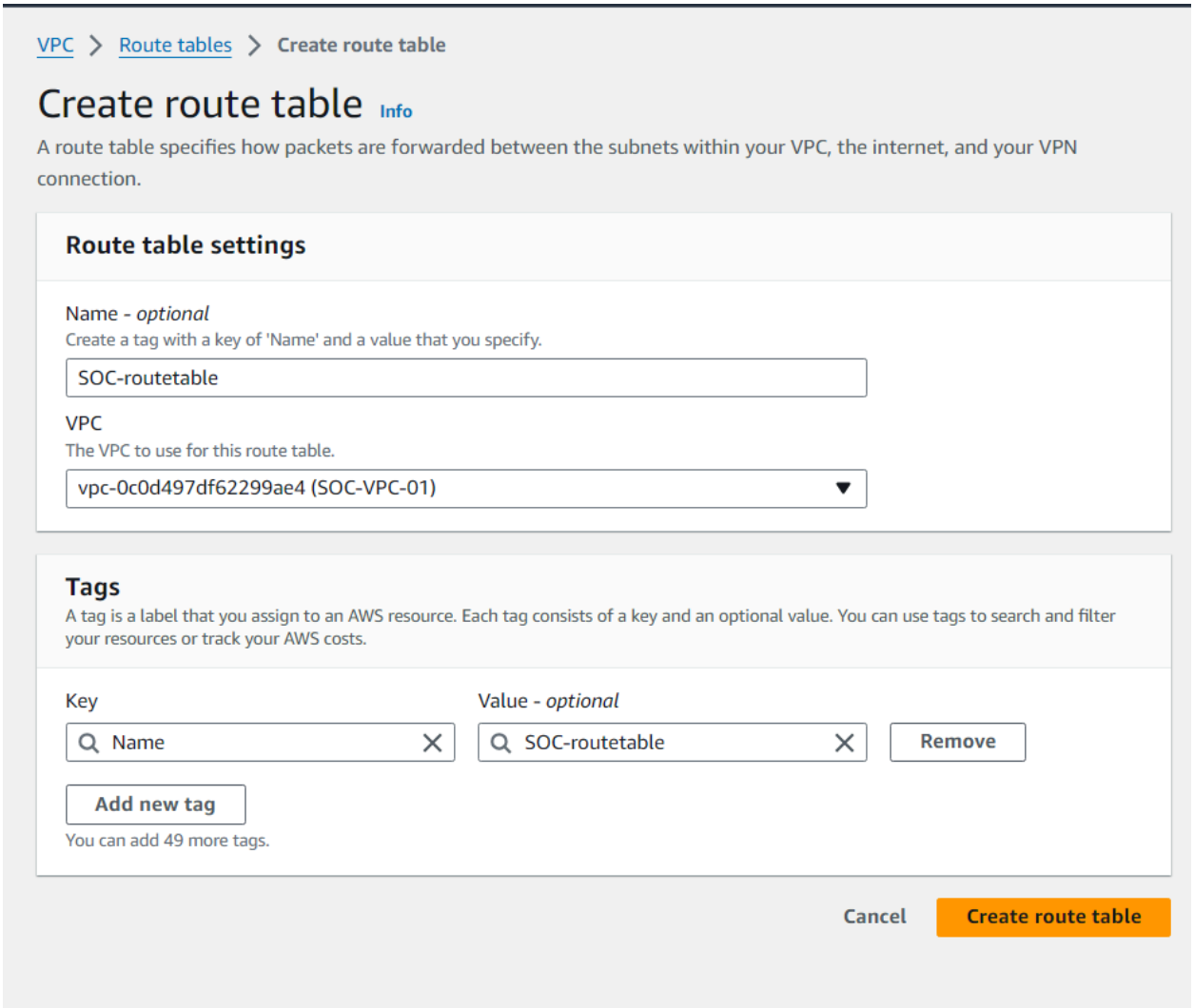
You can add 49 more tags.



Step 3: Create a Route Table

1. Navigate to the Route Tables section:
   - Click on "Route Tables" in the left-hand menu.
2. Create a Route Table:
   - Click the "Create route table" button.
   - Select your VPC from the drop-down.
   - Fill in the details:

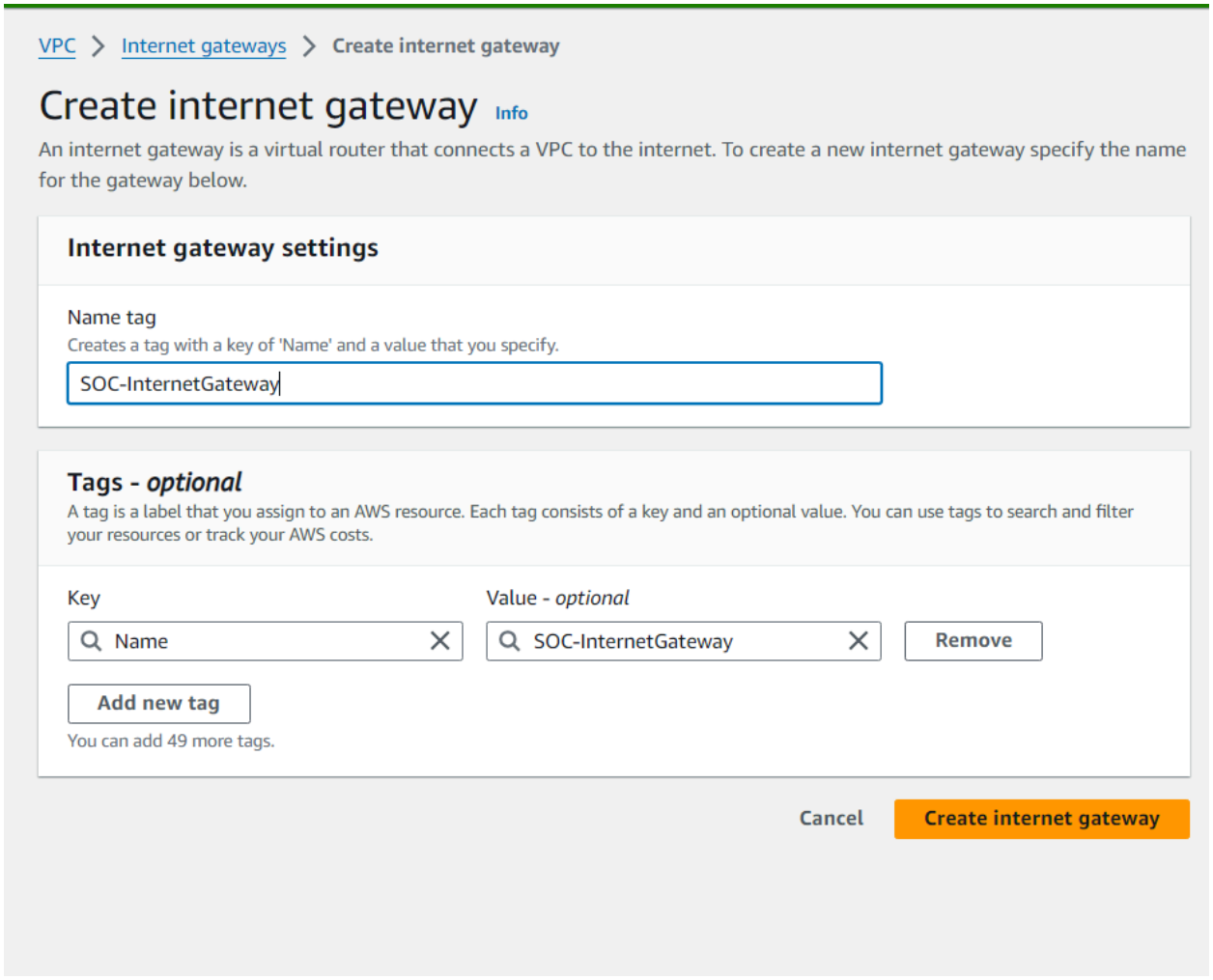
- Name tag: Provide a name for the route table.
- Click "Create route table".

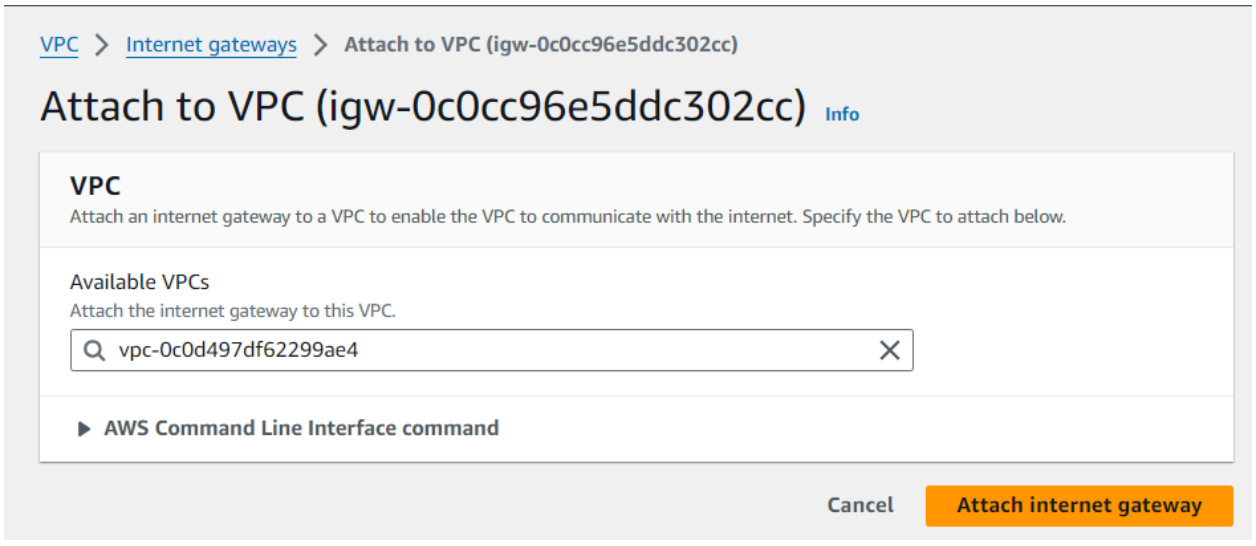


Step 4: Create the Internet Gateway (IGW)

1. Navigate to the Internet Gateways section:
   - Click on "Internet Gateways" in the left-hand menu.
2. Create an Internet Gateway:
   - Click the "Create internet gateway" button.
   - Fill in the details:

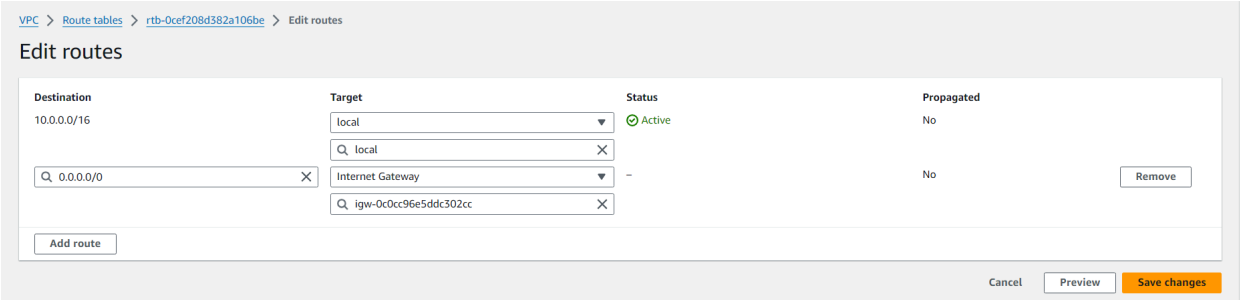
- Name tag: Provide a name for the IGW.
- Click "Create internet gateway".





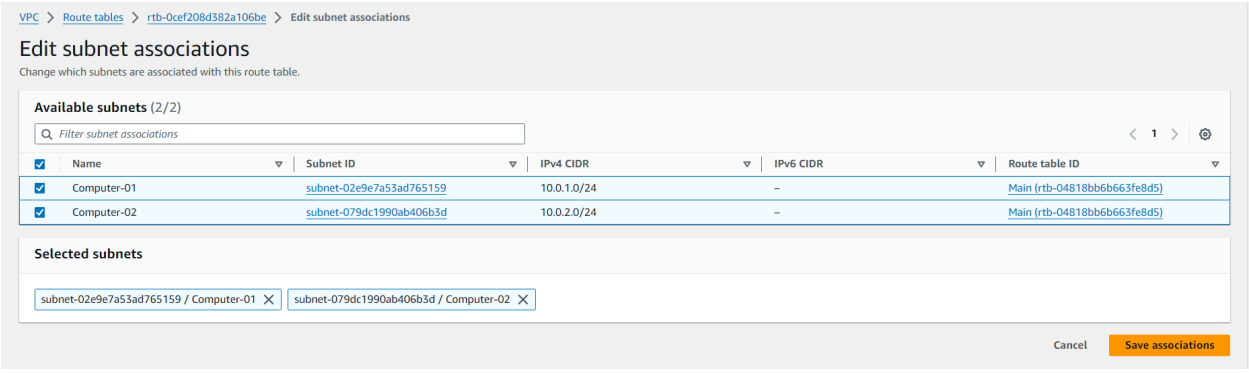
Step 5: Attach the Gateway to the VPC

1. Attach the IGW:
   - Select the IGW you just created.
   - Click the "Actions" button and choose "Attach to VPC".
   - Select your VPC from the drop-down.
   - Click "Attach internet gateway".



Step 6: Add the IGW Route to Your Route Table

1. Modify the Route Table:
   - Select the route table you created.
   - Click the "Routes" tab.
   - Click the "Edit routes" button.
   - Click "Add route".
   - Fill in the details:
     - Destination: `0.0.0.0/0` (this means all traffic).
     - Target: Select your Internet Gateway.
   - Click "Save routes".



Step 7: Associate Both Subnets to the Route Table

1. Associate Subnet 1:
   - Select the route table you created.
   - Click the "Subnet associations" tab.
   - Click the "Edit subnet associations" button.

- Select Subnet 1 from the list.
- Click "Save associations".

2. Associate Subnet 2:
   - Repeat the steps above for Subnet 2.

You have now created a VPC with two subnets, set up an internet gateway, created a route table, and associated your subnets with the route table. This setup allows your subnets to route traffic through the internet gateway, providing internet access to resources within those subnets.

This outlines the security data ingestion pipelines for the SOC platform, covering data sources, ingestion architecture, automation and scalability, error handling, log management and normalization, and access control.

1. Security Data Ingestion Pipelines (Matthew Escalera, Pedro Gomez)

1.1 Data Source Identification

The SOC platform will ingest security data from a variety of sources to provide a comprehensive view of the security posture. Here's a list of potential data sources:

- **Cloud Service Logs:**
  - VPC Flow Logs: Records network traffic flowing in and out of VPCs.
  - GuardDuty Findings: For collecting logs.
- **Network Security Appliance Logs:**
  - Firewalls: Logs security events like blocked traffic and access attempts.
  - Intrusion Prevention/Detection Systems (IPS/IDS): Alerts on suspicious network activity.
  - Web Application Firewalls (WAF): Logs attempts to exploit web application vulnerabilities.

1.2 Data Ingestion Architecture

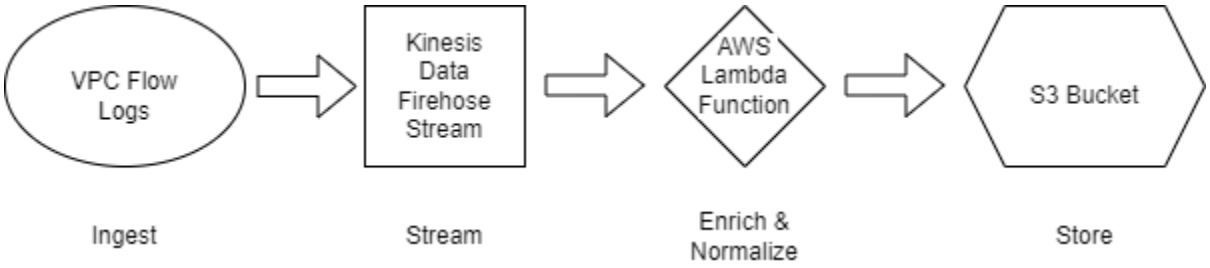
The data ingestion architecture will leverage various services to collect, transform, and deliver security data to the log management platform. Here's a breakdown of the key components:

- **Log Forwarding Mechanisms:**
  - Amazon EventBridge: A standard protocol for sending logs from devices to a central server, used for Aws config, Inspector and Guard Duty.

    - Cloud Trail made to native settings to send logs to CloudWatch, VPC flow logs has the same thing.
- **Data Ingestion Services:**
    - Amazon Kinesis Data Streams: Real-time streaming data for high-volume log ingestion.
    - AWS Lambda: Used for converting information coming from VPC flow logs and parsing and enriching it into a more readable log.
- **Data Transformation and Enrichment:**
    - AWS Lambda Functions: Serverless functions for real-time data processing and transformation tasks.

1.3 Data ingestion pipeline architecture (sample)

VPC FLOW LOGS → Kinesis Data Firehose Stream → AWS Lambda Function (data is being enriched normalized here) → S3 bucket (where the results are being stored.



1.3 Error Handling and Monitoring

```
mport logging

logger = logging.getLogger()

logger.setLevel(logging.INFO)

def lambda_handler(event, context):

    logger.info('Lambda execution started...')

    try:

        # Your Lambda function logic here

        logger.info('Function logic executed successfully.')

        return {

            'statusCode': 200,

            'body': 'Function executed successfully!'

        }
```

```
except Exception as e:
    # Log the error
    logger.error(f'Error: {str(e)}')
    # Return a custom error response
    return {
        'statusCode': 500,
        'body': 'Internal Server Error'
    }
finally:
    logger.info('Lambda execution completed.')
```

The code above can be of use when Lambda is acting unexpectedly. You can use detailed logging with the 'logger' module. This can be used with CloudWatch Logs to examine logs and identify the root cause of the unexpected behavior. When using detailed logging, this can provide a trail of events within the Lambda function.

2. Log Management & Normalization (Giovanni Garcia Flores, Mario Register)

2.1 Log Management Platform

AWS Cloud Watch

We decided to use AWS Cloud Watch as our log management platform.

We currently have these log groups for these services:

AWS Guard Duty

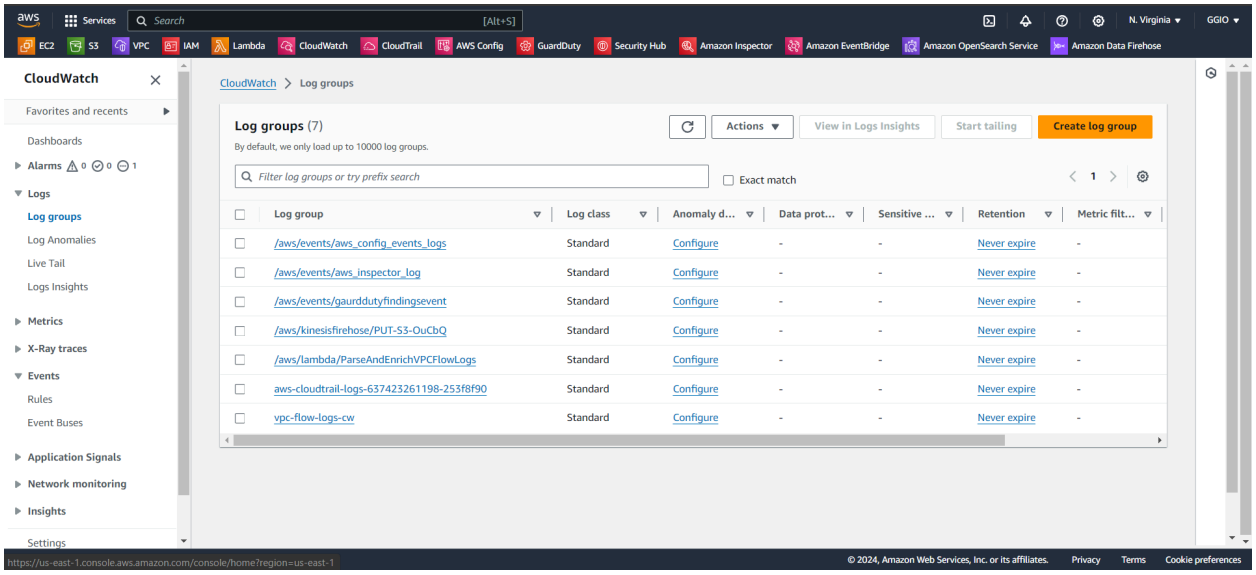
AWS Inspector

AWS Config

AWS VPC Flow Logs

AWS Cloudtrial

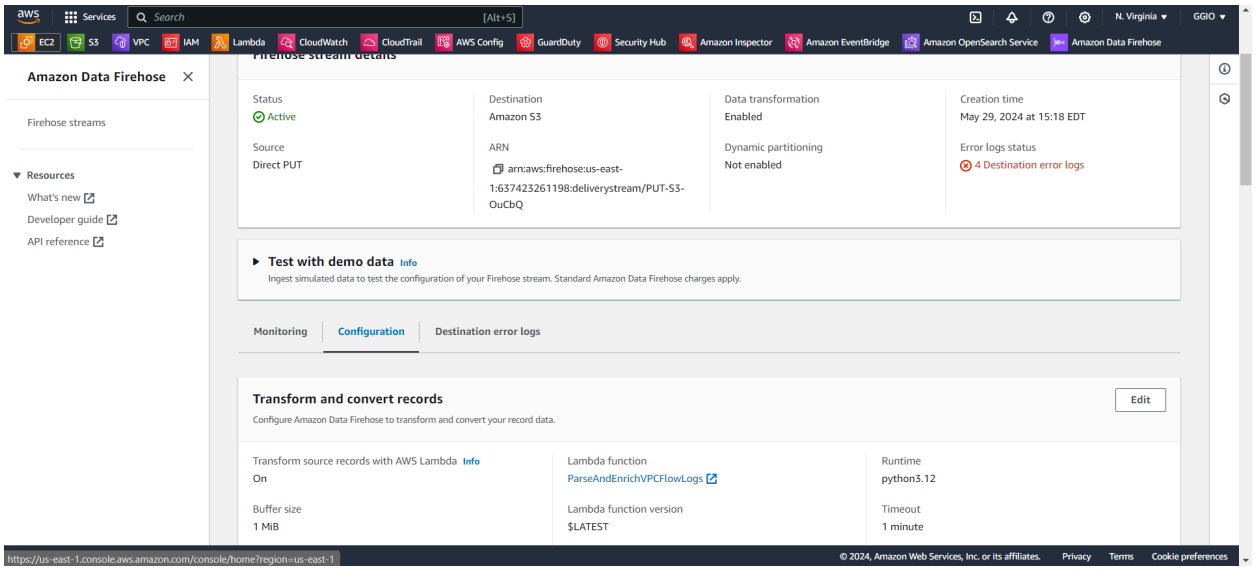
There are also logs being collected for both our Lambda and Kinesis Data Firehose services running our log data normalization and enrichment service.



We used a mix of Amazon Eventbridge rules and native AWS logging services within the application to send their logs to Cloudwatch.

2.2 Log Normalization and Enrichment

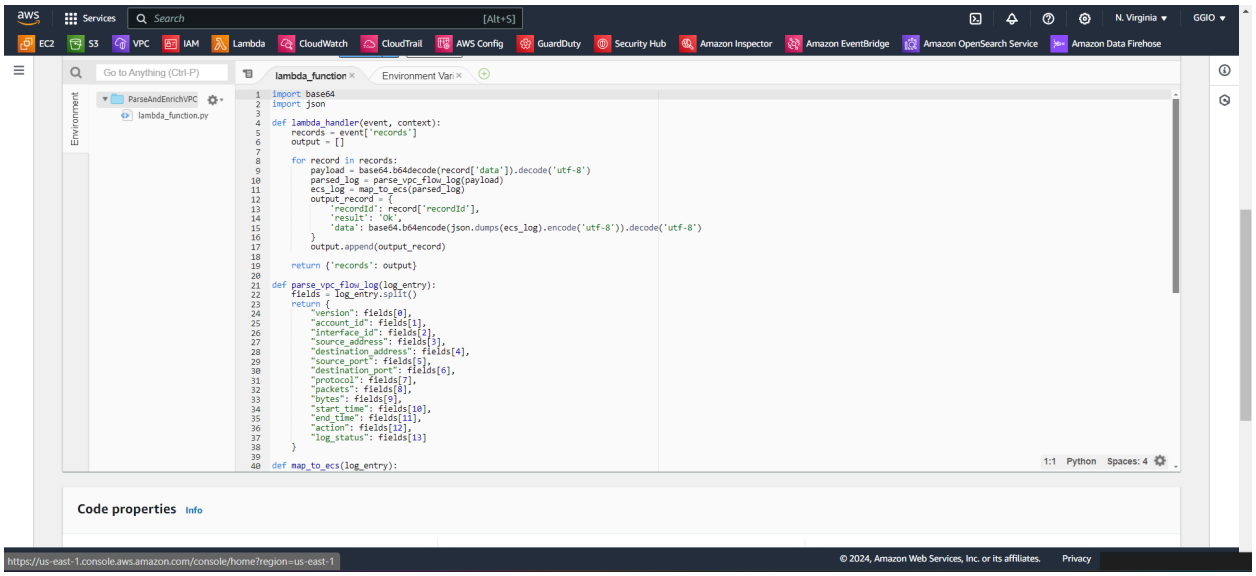
We created a Kinesis Data Firehose stream that would intake logs which would then be transformed by AWS Lambda to further normalize and enrich the data into a more readable and accessible type of data.



When creating the stream, an appropriate IAM role with the proper permissions to collect the logs and to be able to use the Lambda function is needed.

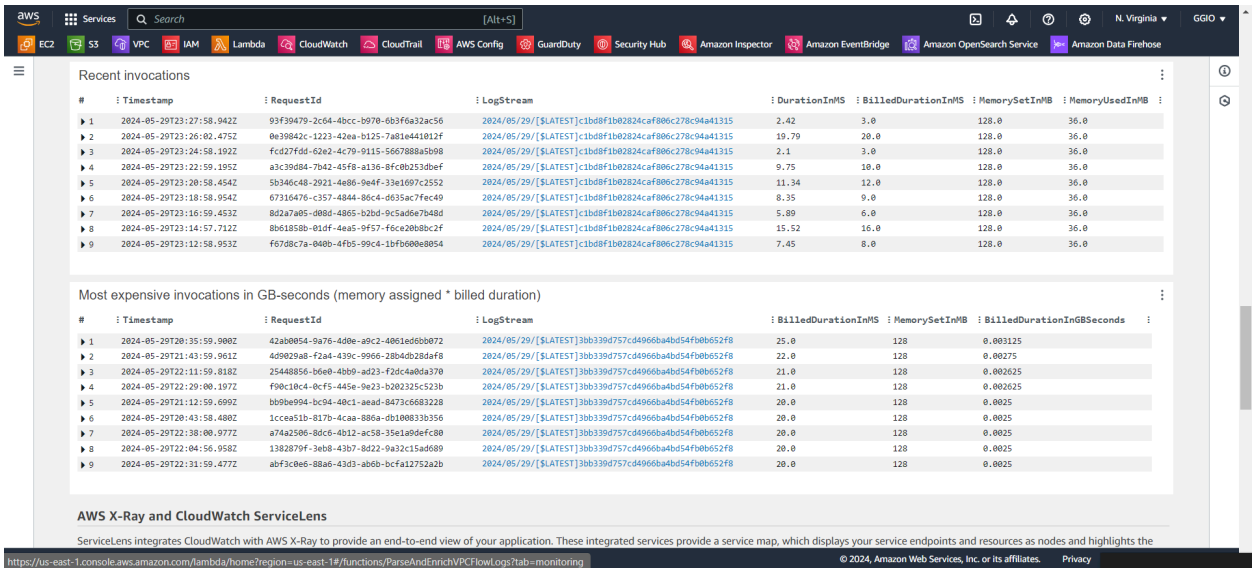
An IAM role also needs to be created for the Lambda function itself to be able to intake and change the logs but to also store them in an S3 bucket.

The Lambda function itself would also need to have a custom Python code to be able to do the data enrichment and normalization itself.

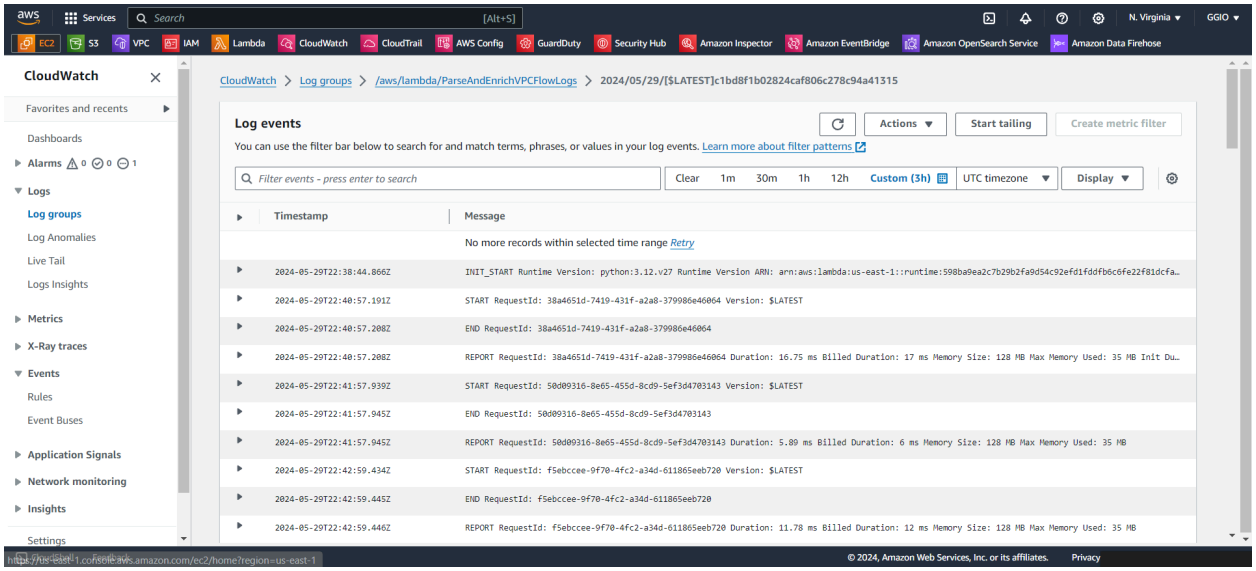


For demonstration, we used VPC Flow logs as an example. We created an EC2 to be able to simulate network traffic for new logs to be created and monitored.

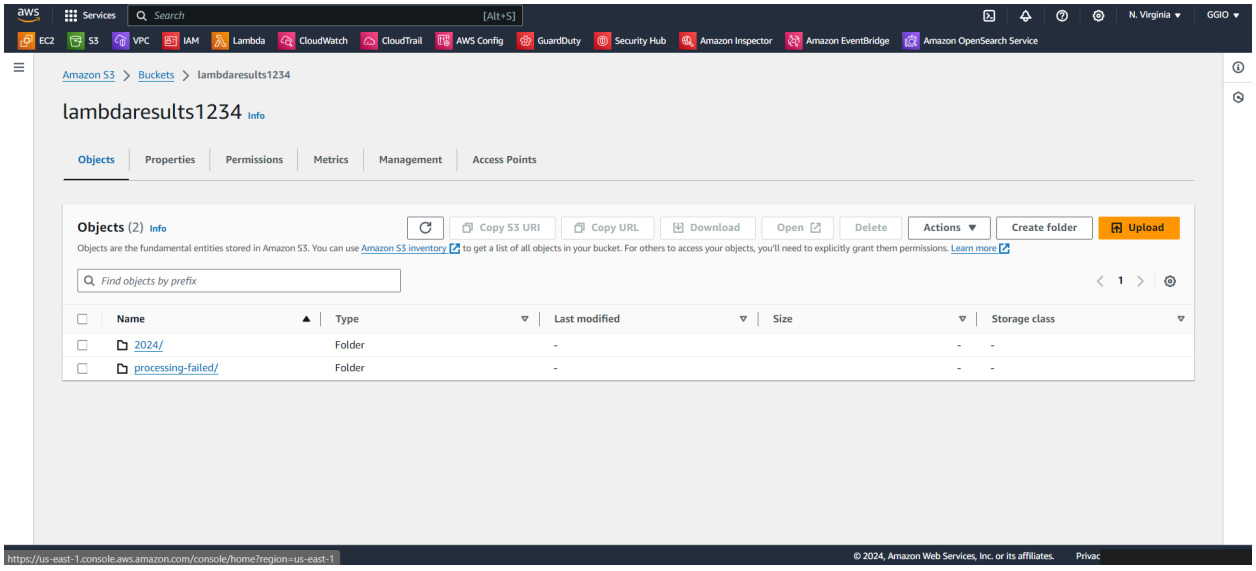
Checking the monitor tab will tell us if the code has been executed or not.



We can check on the logs and we can see there was a successful run,



We then go into the bucket where the data has been transformed and delivered to.



We then look out for the most recent file and download it, then open it. We personally used Visual Studio Code to view the file, here is the result.

Creating a GuardDuty Threat List and Integrating with Security Hub

This guide outlines the steps to create a GuardDuty threat list, test it with EC2 instances, and integrate the findings into Security Hub for improved security visibility:

Step 1: Create a GuardDuty Threat List

1. **Prepare the Threat List:**
    - Create a CSV file named `threatlist.csv` containing the IP addresses you want to monitor.
    - Example:

```
192.0.2.0/24
198.51.100.0/24
```

2. **Upload the Threat List to an S3 Bucket:**
    - Upload the `threatlist.csv` file to an S3 bucket in your AWS account.
3. **Enable GuardDuty and Create a Threat List:**
    - If not already enabled, navigate to the AWS Management Console and enable GuardDuty.
    - In the GuardDuty console, under "Lists," create a new Threat List.
    - Provide the S3 URI of the uploaded CSV file when creating the Threat List.

Step 2: Create and Configure EC2 Instances

1. **Launch Two EC2 Instances:**
    - Launch two EC2 instances in your AWS account, ensuring they are in the same region where GuardDuty is enabled.
2. **Simulate Malicious Activity:**
    - Connect to one of the EC2 instances.
    - Simulate traffic to one of the IP addresses in your threat list. You can use `curl` to make an HTTP request:
    - Bash

```
curl http://192.0.2.1
```

    -
    - (Replace `192.0.2.1` with the actual IP address from your list)

Step 3: Monitor GuardDuty Findings

1. **Check GuardDuty Findings:**
    - After simulating activity, navigate to the GuardDuty console.
    - Under "Findings," check for new findings related to the simulated activity. GuardDuty should detect connections to the IP addresses in your threat list.

Step 4: Integrate GuardDuty Findings into Security Hub

1. **Enable Security Hub:**

    - If not already enabled, navigate to the Security Hub console and enable Security Hub.
2. **Enable GuardDuty Integration:**

    - In the Security Hub console, go to the "Integrations" section.
    - Enable the GuardDuty integration if it's not already enabled.
3. **View Findings in Security Hub:**

    - After enabling the integration, navigate to the "Findings" section in Security Hub.
    - You should see GuardDuty findings listed along with other security findings.

Step 5: Automate Responses (Optional)

This step outlines an optional approach to automate responses to GuardDuty findings:

1. **Set Up Automated Responses with AWS Lambda:**

    - You can configure automated responses using AWS Lambda and CloudWatch Events.
2. **Create a CloudWatch Event Rule:**

    - Go to the CloudWatch console and create a new rule with the following event pattern to match GuardDuty findings:

- JSON

```
{
  "source": ["aws.guardduty"],
  "detail-type": ["GuardDuty Finding"]
}
```

-
-
- Set the target of the rule to the Lambda function that you want to invoke when a GuardDuty finding triggers the event.

3. **Create a Lambda Function:**

- Create a Lambda function to handle GuardDuty findings. The function can take actions like stopping an instance or sending notifications based on the finding details.

```
import boto3


def lambda_handler(event, context):
    ec2 = boto3.client('ec2')
    guardduty = boto3.client('guardduty')
    finding = event['detail']
    instance_id =
finding['resource']['instanceDetails']['instanceId']
```

```
        try:
            response = ec2.stop_instances(InstanceIds=[instance_id])
            print(f"Stopped instance: {instance_id}")
        except Exception as e:
            print(f"Error stopping instance: {str(e)}")
```

Important Notes:

- Ensure the Security Hub integration with GuardDuty is enabled for findings to appear.
- Consider testing the Lambda function independently before integrating it with GuardDuty findings to ensure proper functionality.
- Remember to follow AWS best practices for security and access control when setting up these services.

By following these steps, you can create a GuardDuty threat list, monitor suspicious activity, and integrate the findings with Security Hub for a centralized view of your security posture. Optionally, you can further enhance your security by automating responses using AWS Lambda and CloudWatch Events.

Technical Documentation (Focused on SIEM with Limited SOAR Integration)

SOAR Tool Selection and Integration (Future Consideration)

- **Chosen Tool (Placeholder):** While a full SOAR suite isn't implemented here, consider integrating a SOAR platform like Palo Alto Networks Cortex XSOAR or MacAfee MVISION SOAR in the future. These platforms offer comprehensive SOAR features and integrate with various AWS services.
- **Integration Process (Future Consideration):** Integration with a chosen SOAR platform would involve establishing secure communication channels between the SOAR tool and your AWS services. This might involve API integrations or dedicated connectors provided by the SOAR vendor. Security Hub could play a role here, as it aggregates security findings from AWS services and can be integrated with some SOAR platforms.
- **Challenges (Future Consideration):** Potential challenges include managing API keys and access controls for secure communication, ensuring data format compatibility between platforms, and potential limitations in the free tiers of some SOAR solutions.

Incident Response Playbooks (Limited with SIEM)

- **Overview:** While a full SOAR suite isn't implemented here, we can leverage basic automation within the SIEM solution for some incident response tasks. Playbooks might be limited in scope but can still be beneficial.
- **Incident Types:** Playbooks could address common security incidents like suspicious login attempts detected by CloudTrail, potential data exfiltration identified through VPC Flow Logs, or vulnerabilities discovered by Inspector.

- **Workflow and Automation:** Playbooks could trigger automated actions like user account lockout for suspicious logins, network traffic filtering for potential exfiltration attempts, or automated isolation of vulnerable instances.
- **Security Best Practices:** Playbooks should adhere to best practices like NIST Cybersecurity Framework or MITRE ATT&CK. Leverage these frameworks to define response procedures based on the identified incident type.

Playbook Testing and Validation

- **Testing and Validation:** Develop test scenarios simulating security incidents. These scenarios could involve injecting test data into CloudTrail logs or VPC Flow Logs to trigger playbooks and validate their functionality.
- **Security of Playbooks:** Implement access controls and permissions within the SIEM solution to restrict unauthorized modification of playbooks. Regularly review and update playbooks to ensure they remain effective and don't introduce vulnerabilities.

Threat Detection & Alert Correlation

SIEM Solution and Data Sources

- **SIEM Solution:** We'll leverage Amazon CloudWatch Logs as the central log management solution. CloudTrail, VPC Flow Logs, and Inspector findings can all be integrated with CloudWatch Logs for centralized analysis.
- **Data Sources:** Data sources include CloudTrail logs (API activity), VPC Flow Logs (network traffic), and Inspector findings (vulnerability scans).
- **Normalization and Enrichment:** CloudWatch Logs offers log parsing capabilities to normalize logs from different sources into a common format.

Additional enrichment might involve integrating threat intelligence feeds to add context to log data.

Correlation Rules and Threat Detection

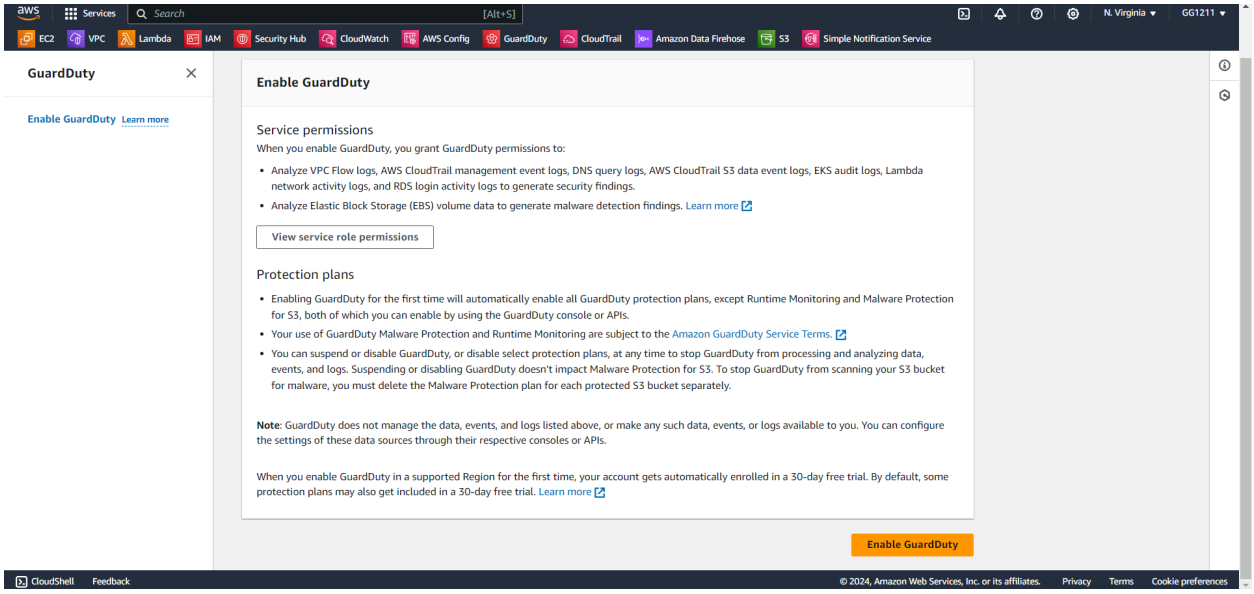
- **Correlation Rules:** Develop rules within CloudWatch Logs that analyze log data from various sources to identify potential threats. These rules might look for specific patterns in CloudTrail logs indicating suspicious activity or correlate network traffic logs with potential vulnerabilities identified by Inspector.
- **Techniques:** Techniques could include pattern matching (e.g., identifying known malicious commands in CloudTrail logs), statistical analysis (e.g., detecting unusual spikes in network traffic), or simple rule-based correlations.
- **Threat Intelligence and IoCs:** Integrate threat intelligence feeds from reputable providers to enrich correlation rules with the latest indicators of compromise (IoCs). This helps identify emerging threats and suspicious activity patterns.
- **Tuning and Refinement:** Continuously monitor and refine correlation rules to reduce false positives and improve detection accuracy. Analyze false positives to understand the root cause and adjust rules accordingly.

Alert Prioritization and Triage

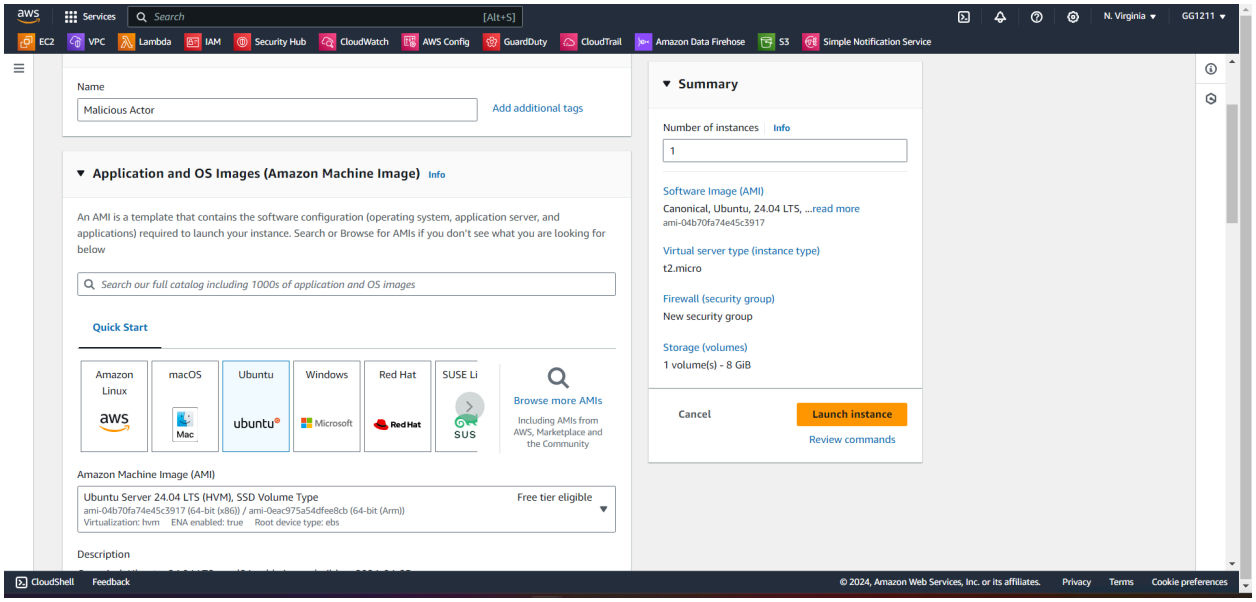
- **Prioritization:** Prioritize alerts based on a combination of factors like asset criticality (e.g., high-risk servers), threat level (e.g., potential data breach), and confidence score (e.g., high confidence based on multiple indicators).
- **Integration with Playbooks:** Alerts triggered by correlation rules can be routed to basic automation within the SIEM solution or integrated with a future SOAR platform for more complex workflows.
- **High-Fidelity Alerts (Example):** A high-fidelity alert might be triggered if CloudTrail logs show a successful login attempt from an unauthorized IP address

for a critical server account. This could initiate automated playbook actions to lock the compromised account and notify security personnel for further investigation.

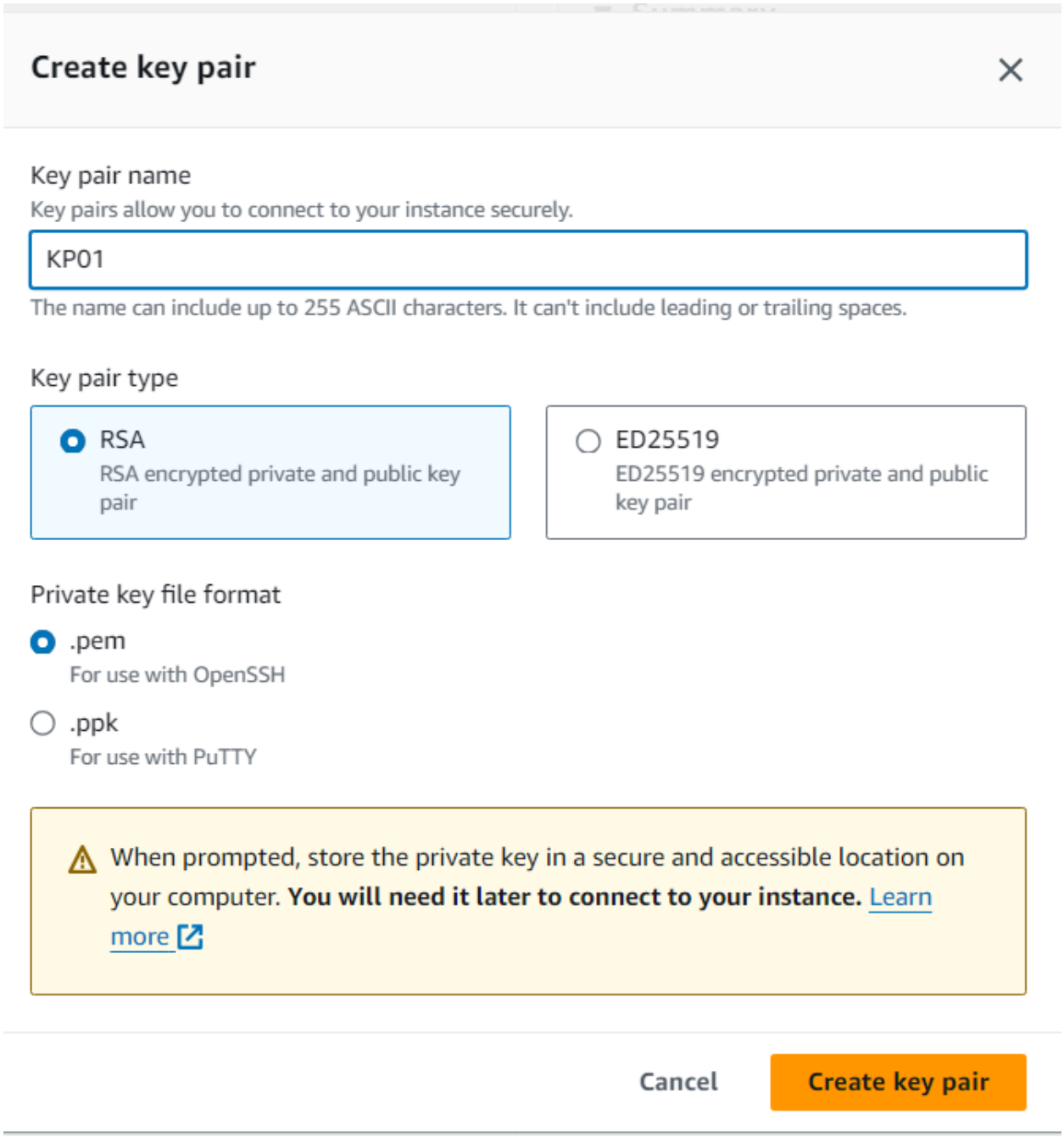
Enable Guard Duty:



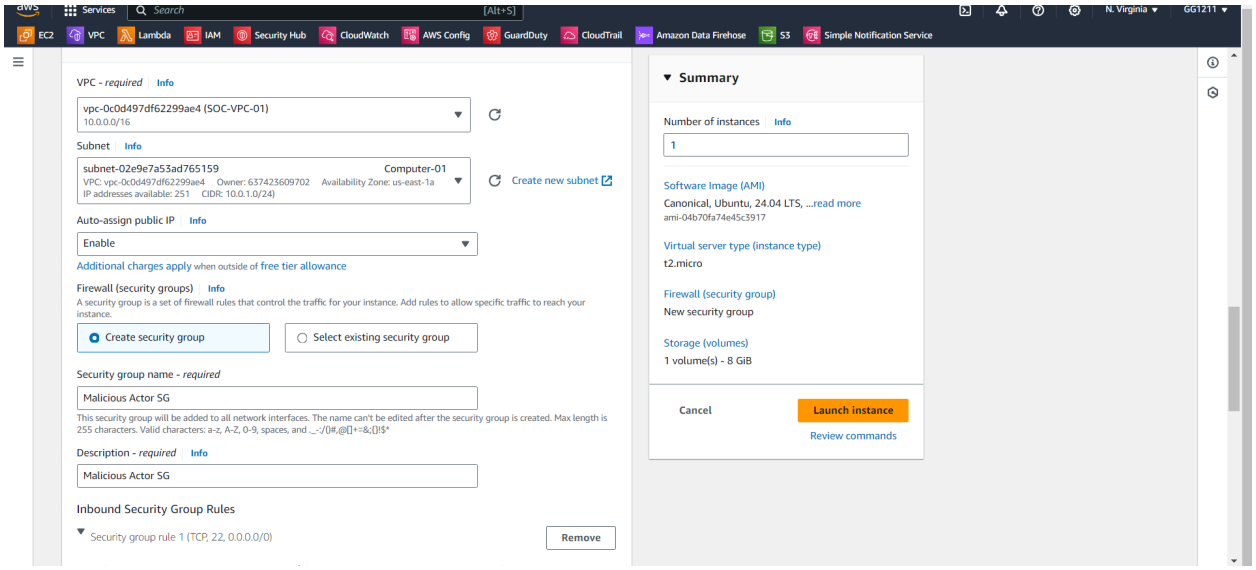
Go and create an EC2



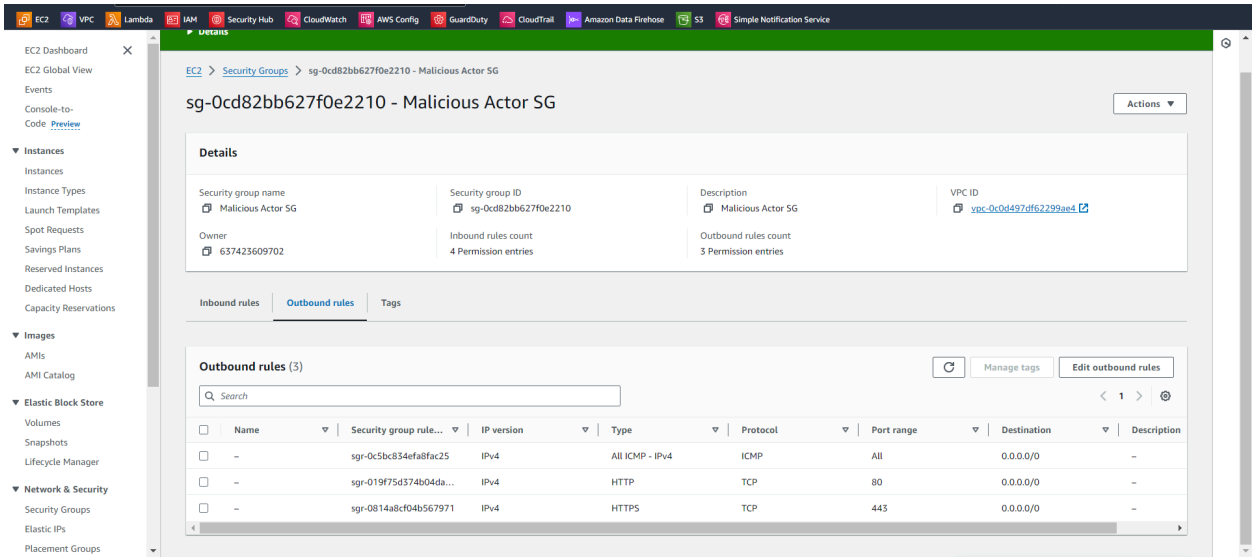
Create a new keypair, we will be using this for all our instances.

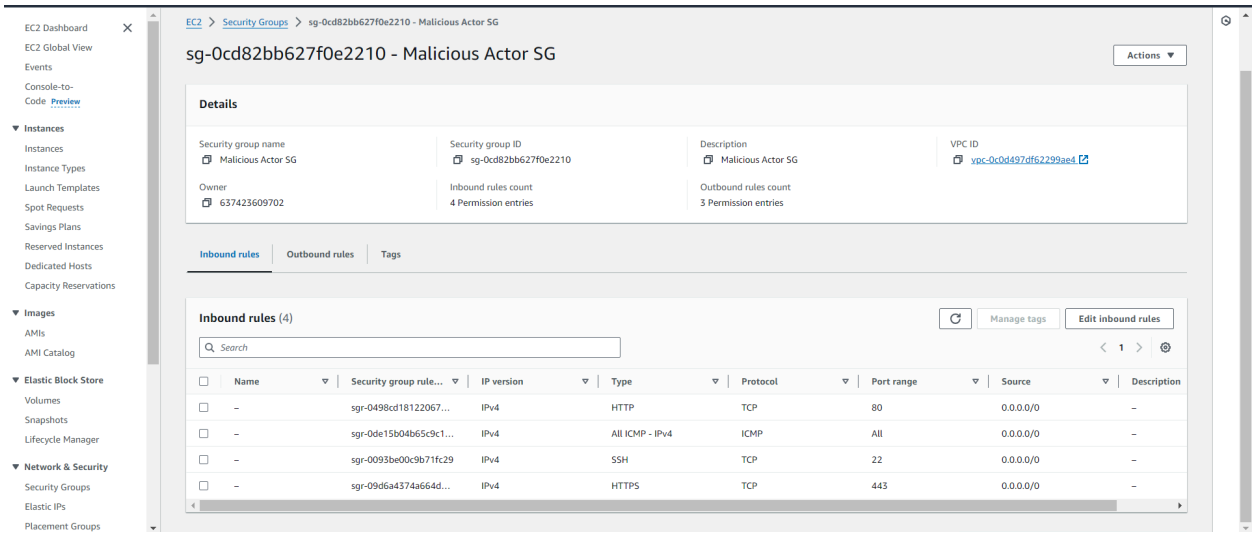


These are the next settings



These are the Security Group rules for this instance:

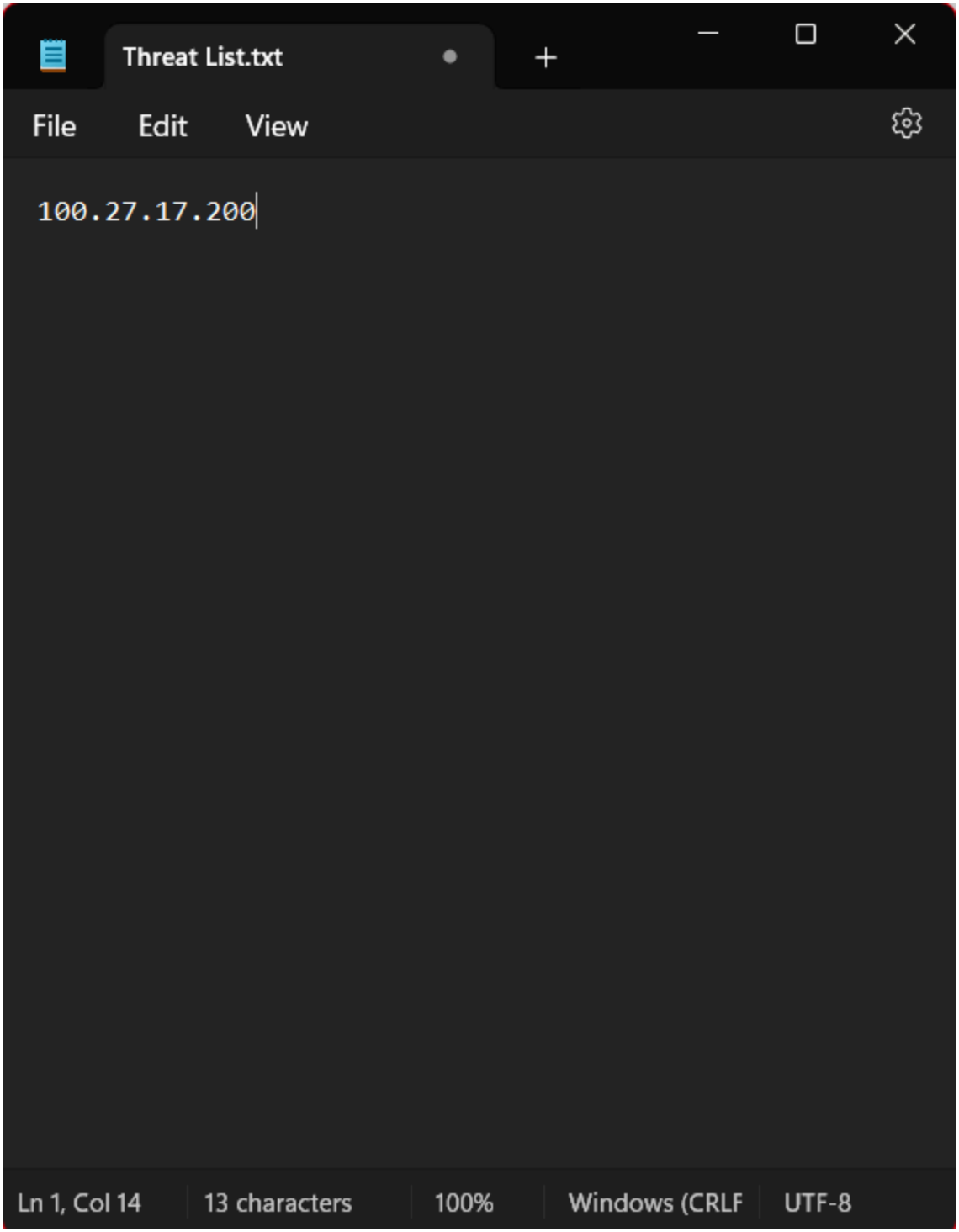




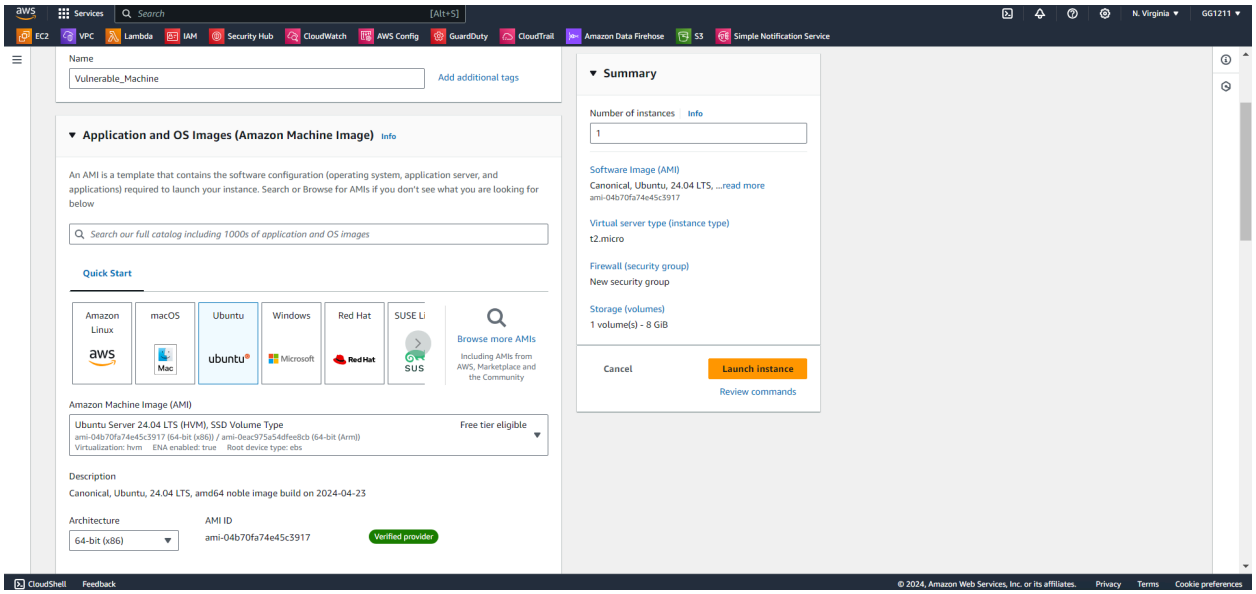
Keep a note of its public ip address

Public IPv4 address
100.27.17.200 | open address

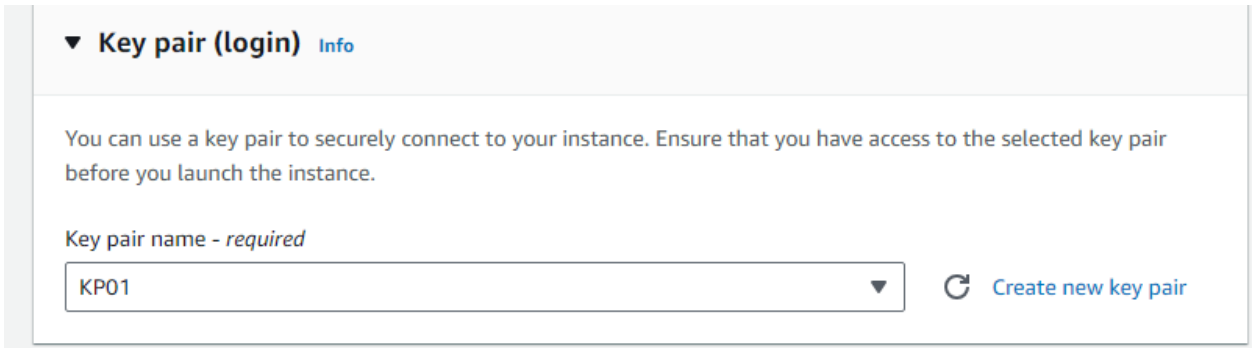
Put that address within a text editor and save it.



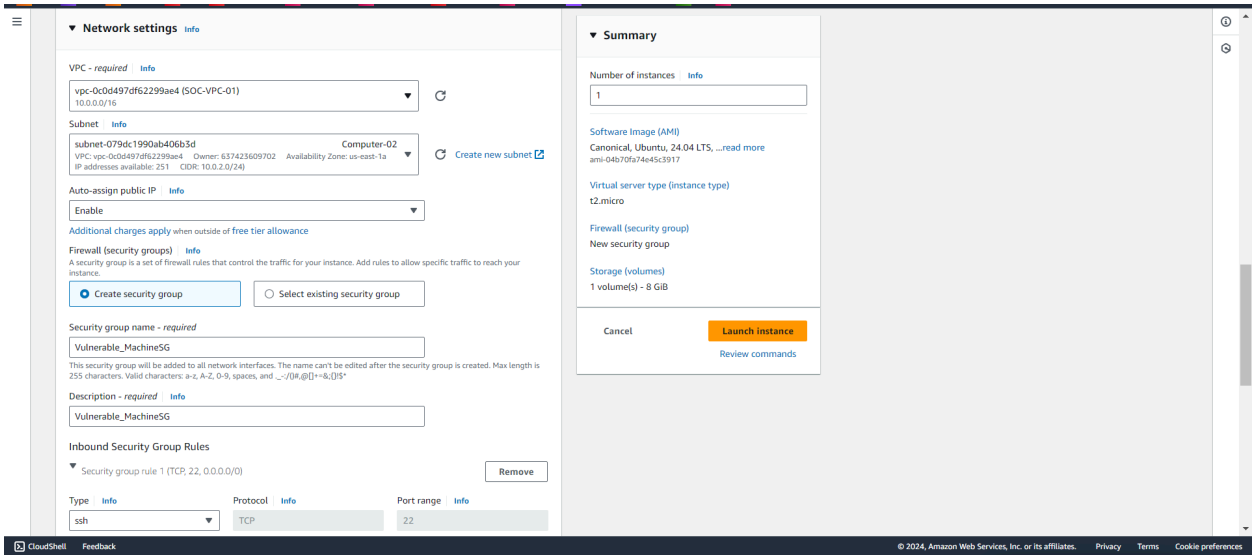
Now we will create an ec2 instance for our vulnerable machine. You can keep it to free tier like the last machine.



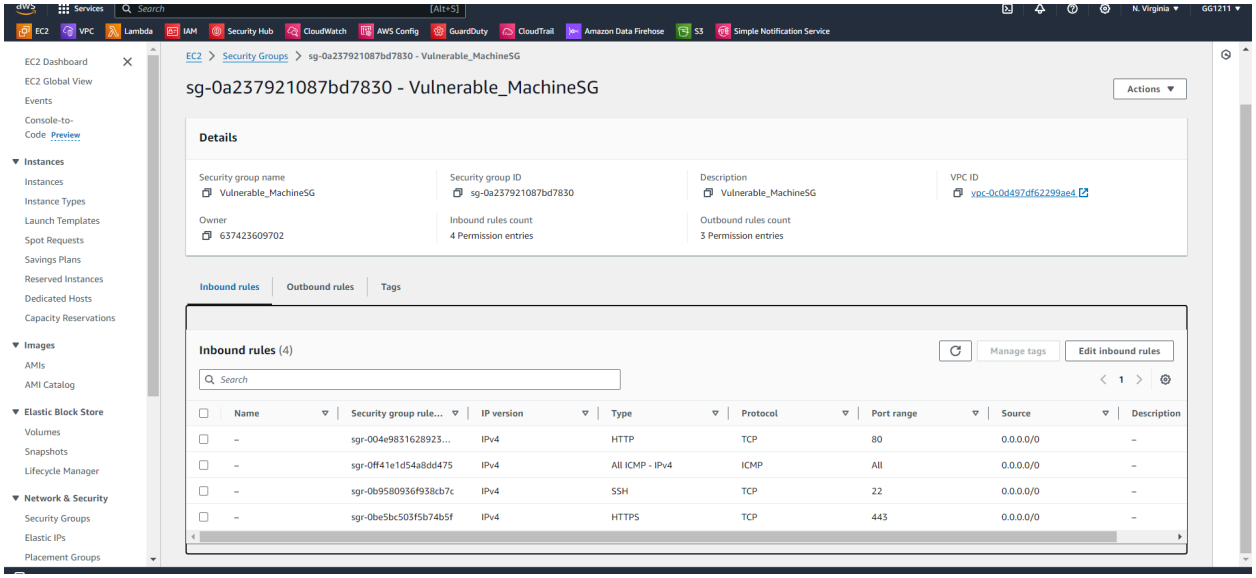
You can use the same keypair.

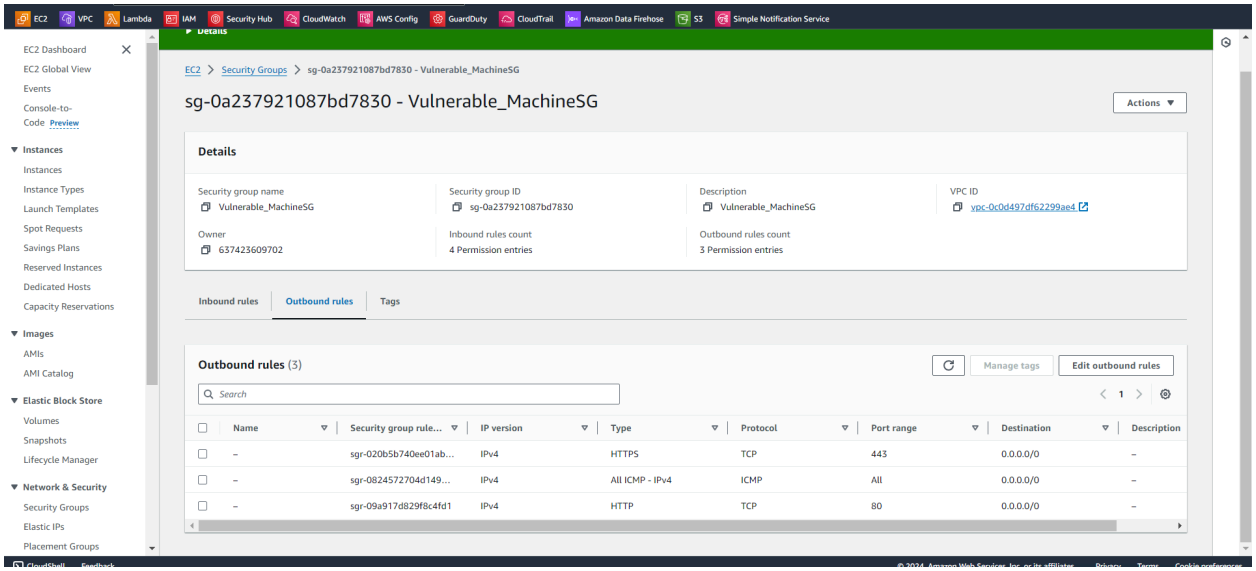


Here are its network settings

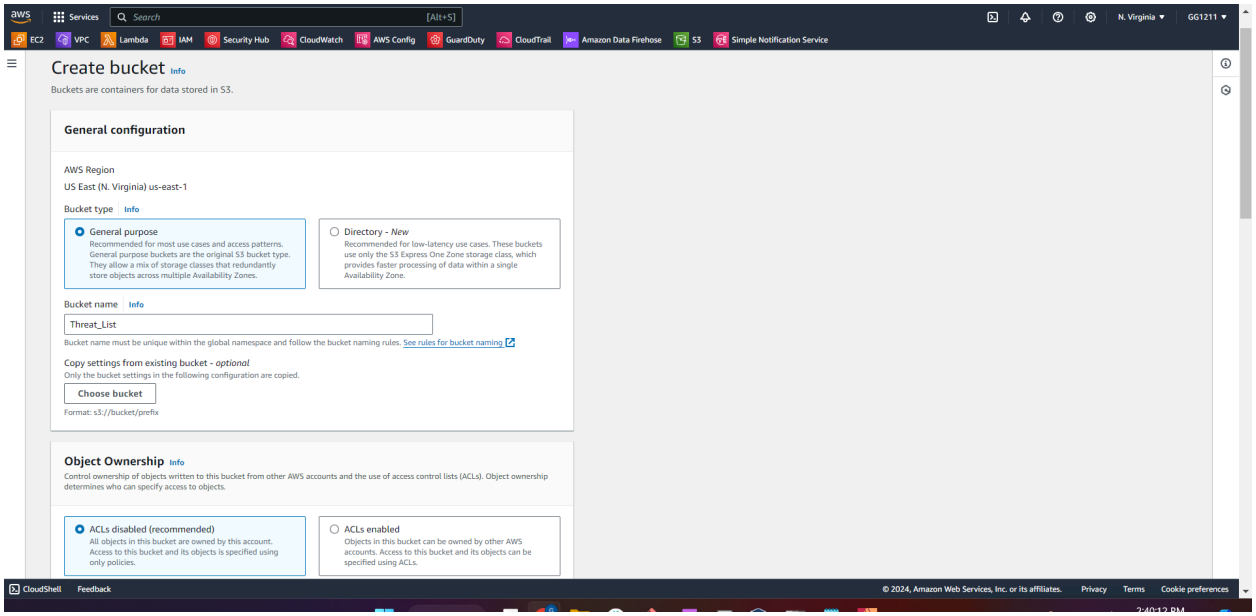


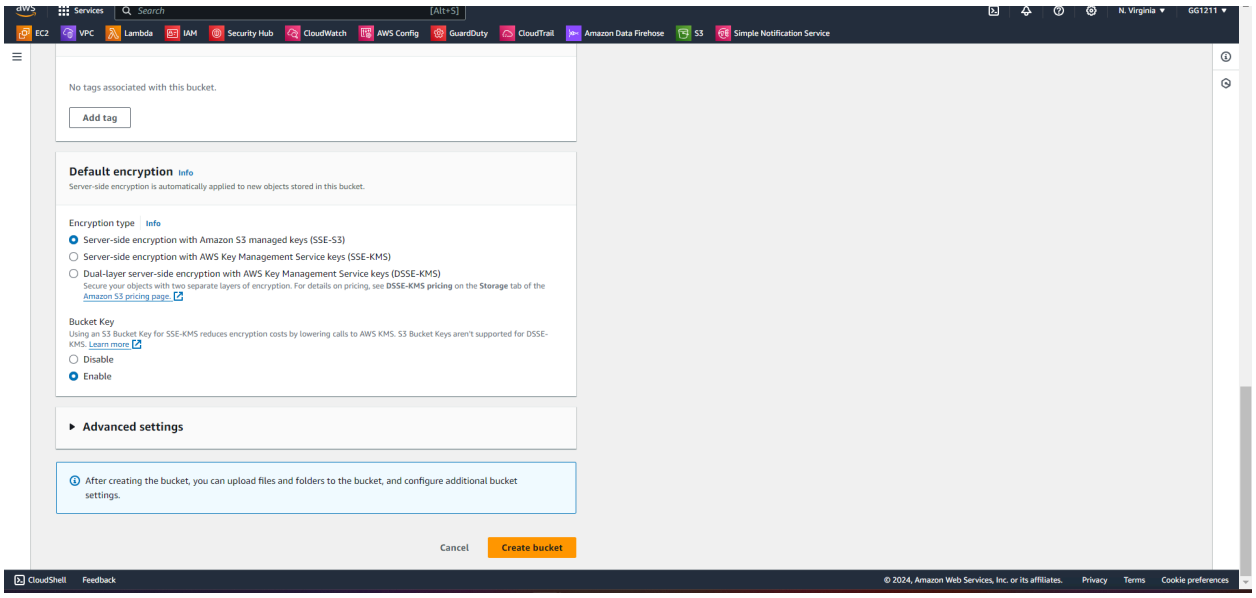
Its Security Group rules





Thats about it, now lets go to making an S3 bucket to store the sample threat list we saved on the notepad.





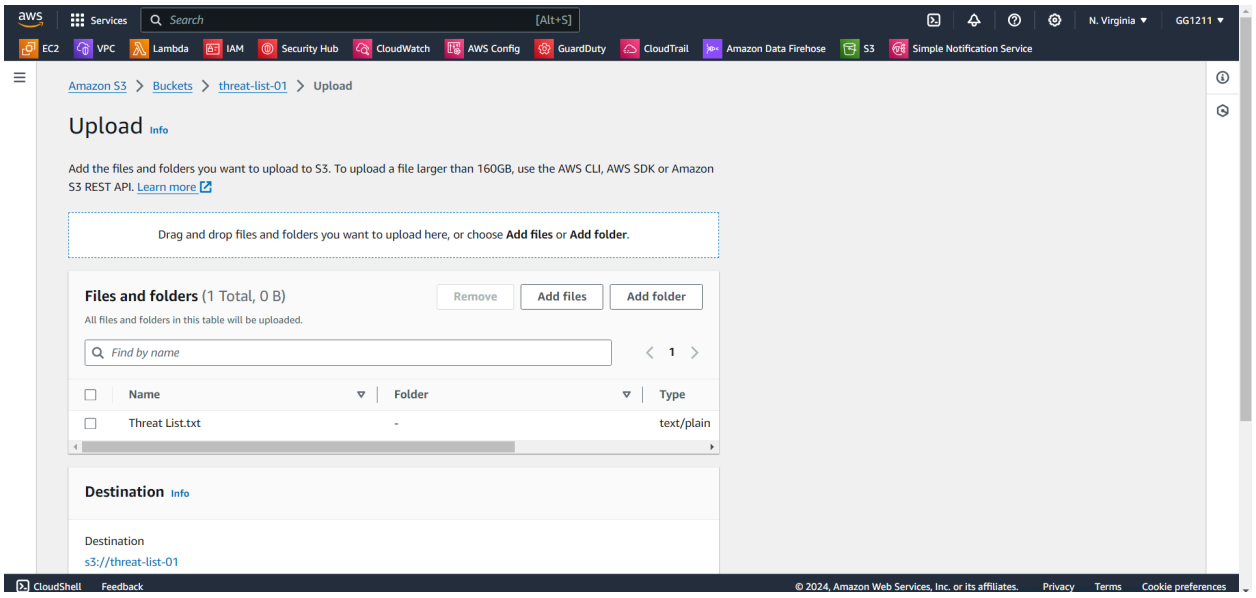
Insert a bucket policy to allow Guard Duty to grab the information from the list. Here is a sample code.

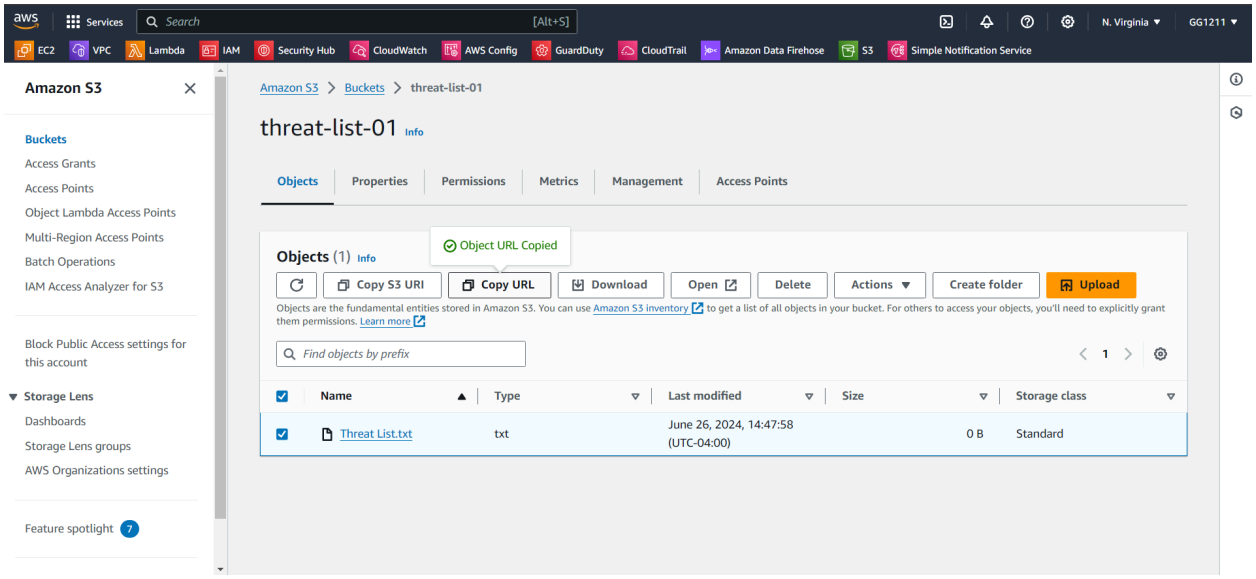
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "guardduty.amazonaws.com"
      },
      "Action": "s3:GetObject",
```

```
    "Resource": "arn:aws:s3:::your-bucket-name/threatlist.txt"
  }
 ]
}
```

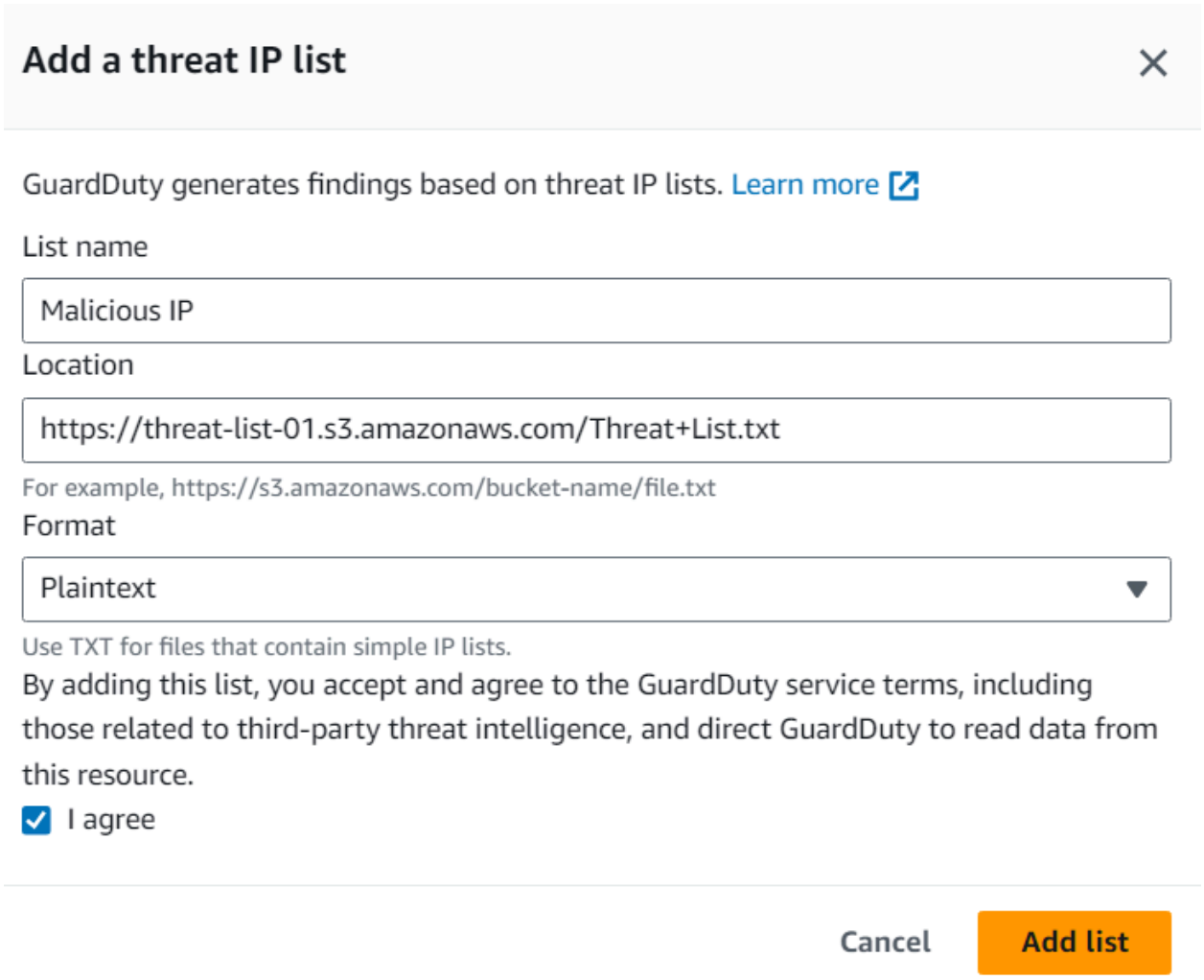
Input that into the policy permissions.

Finally upload the text file to the s3 bucket and save it.

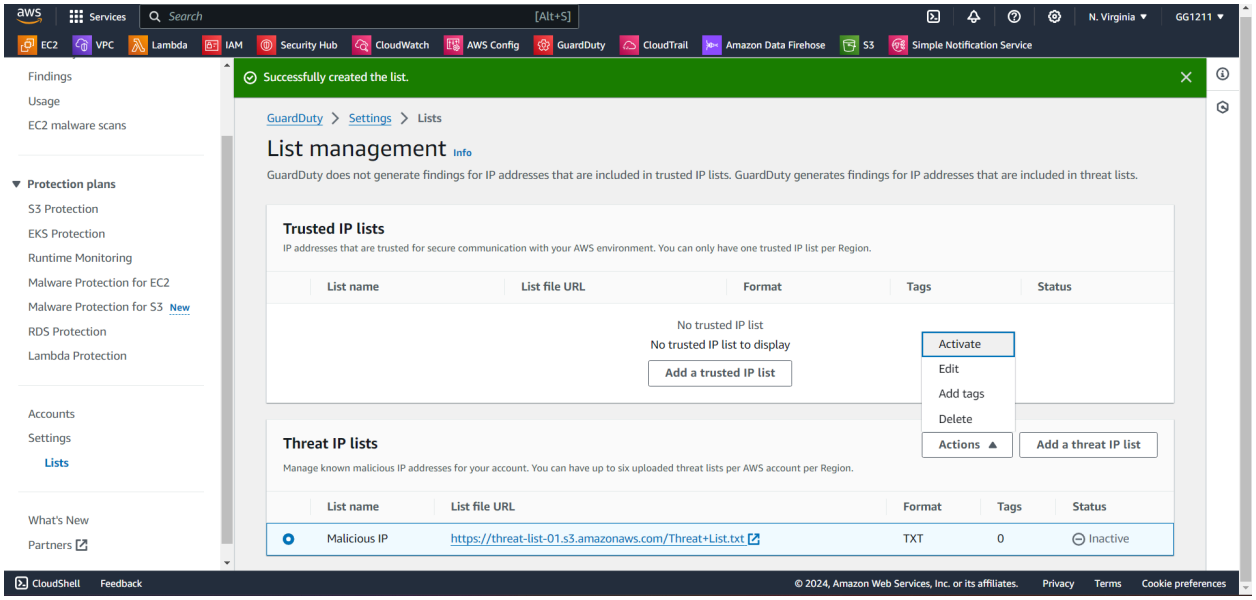




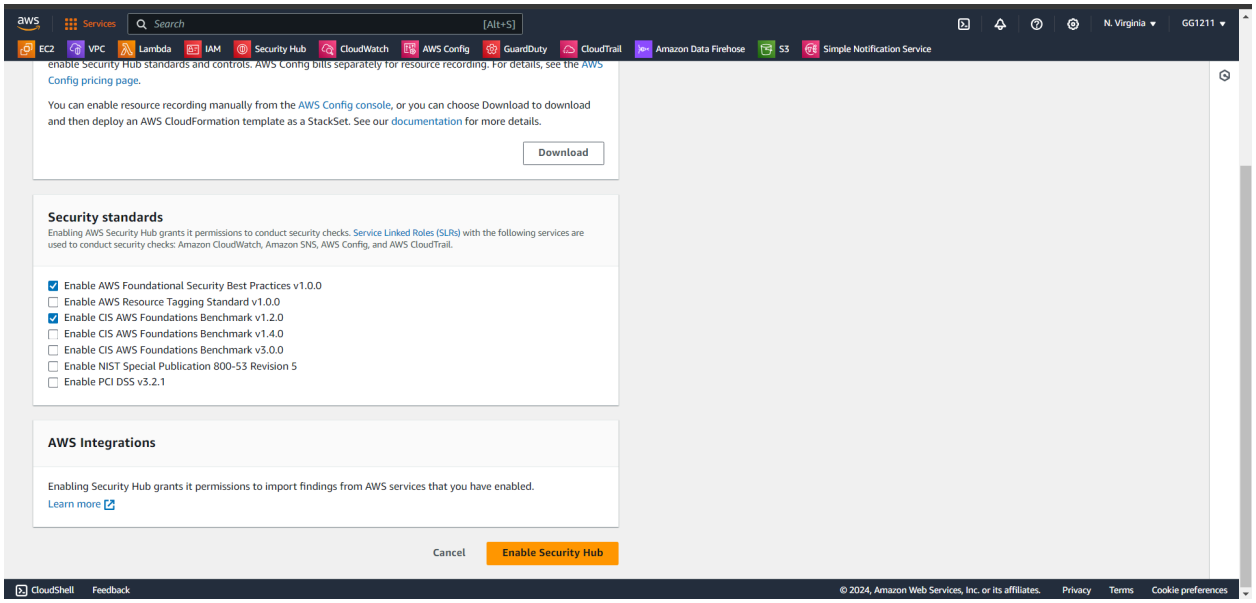
Copy the url.



Go to guard duty and then to lists to add the s3 bucket to it. Make sure you add it to the threat ip. You will need the url of the bucket in order to add it to the guard duty threat list.



And finally just activate it.

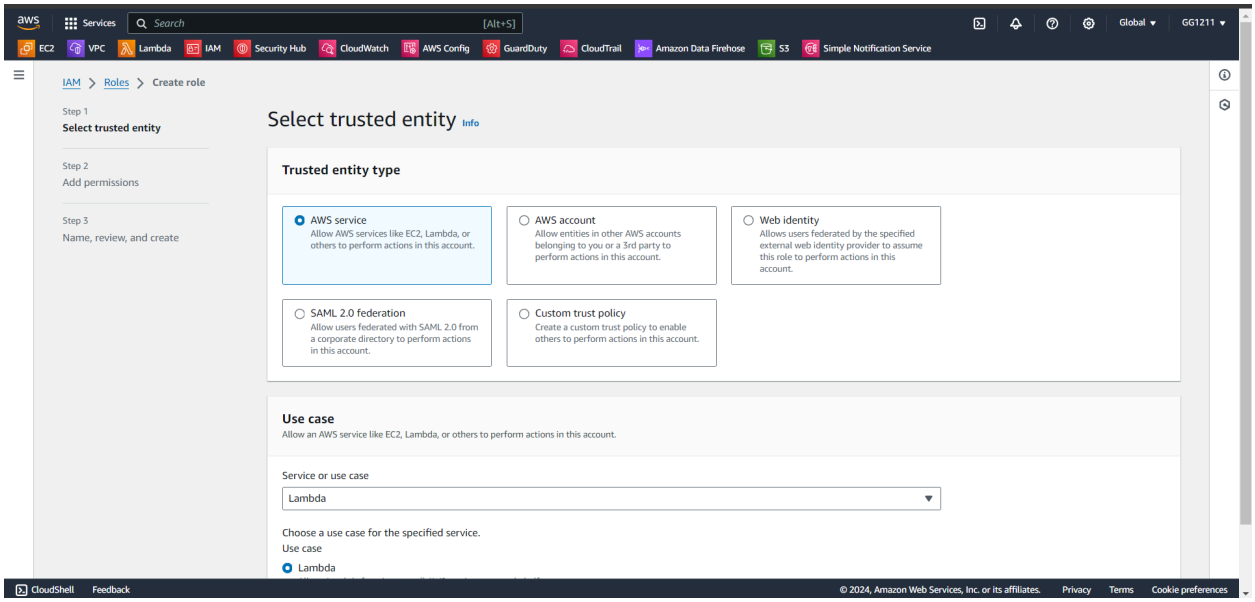


Optionally you can activate security hub now for extra insights into the sample attack we are about to make. NOTE you will need Config set up before hand.

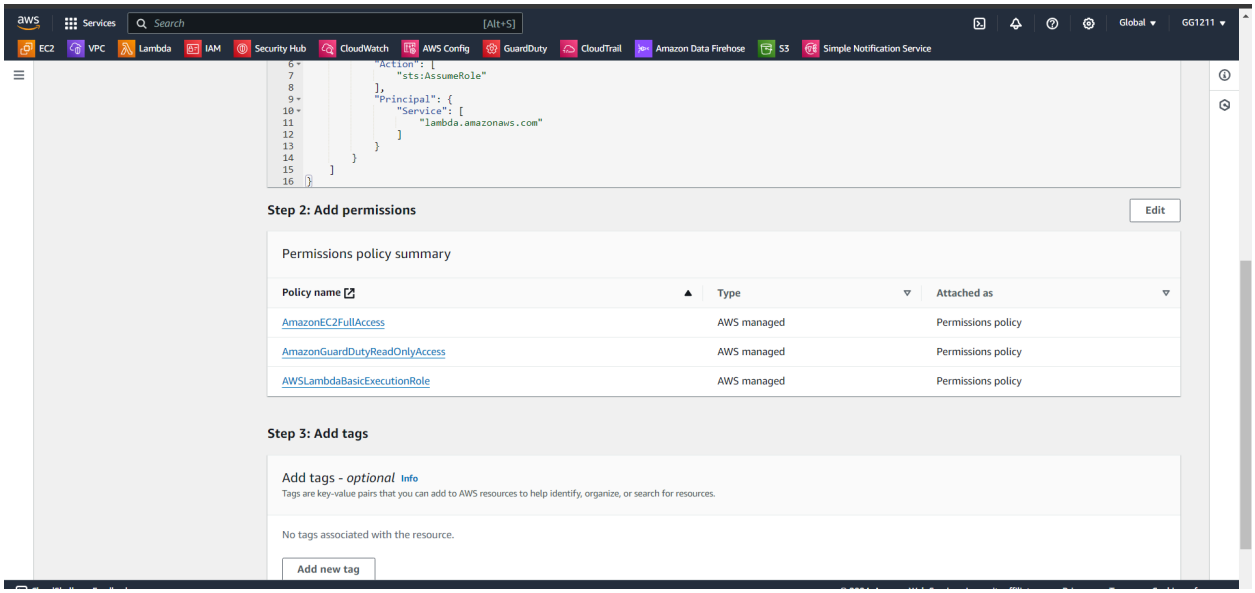
SOAR AUTOMATION SETUP

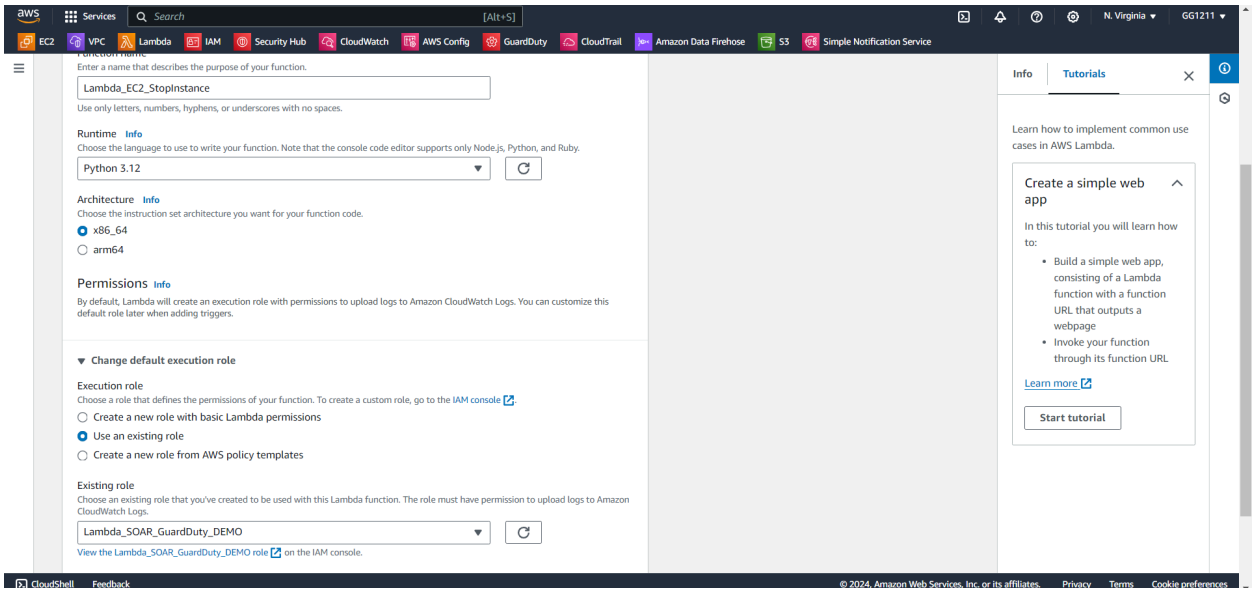
For the SOAR example, we will set up an event bridge rule and use lambda to create a remediation action.

Go to IAM and create a custom role for Lambda so it can have permission to change the ec2 instance state to listen to guard duty.



Make sure on the next step to add EC2 full access permission and guard duty read only access and lambda basic execution role

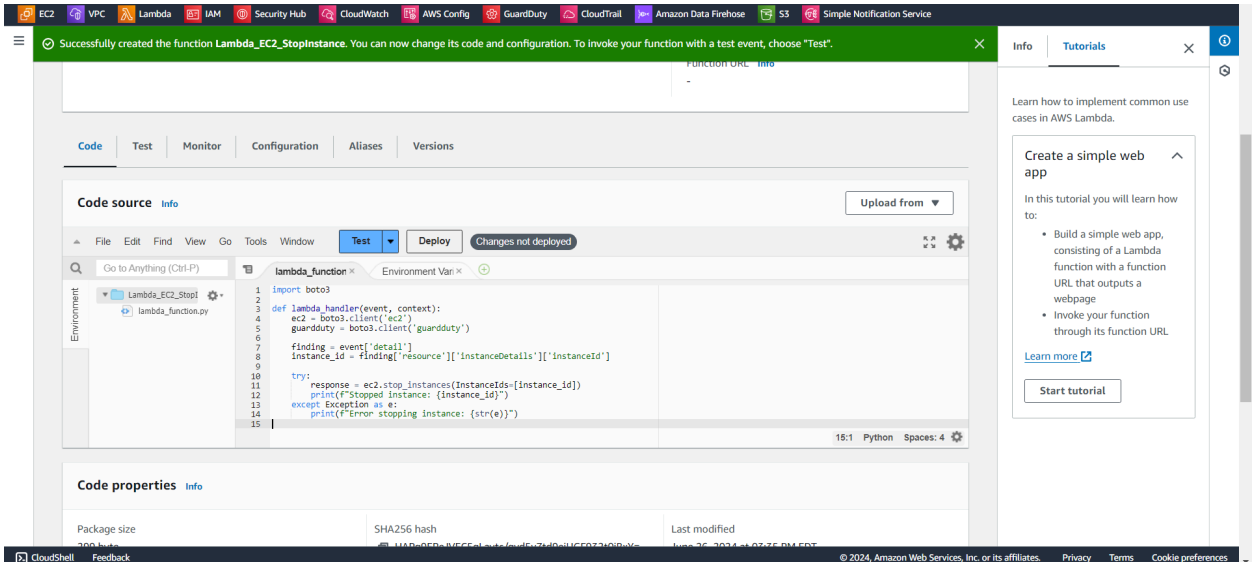




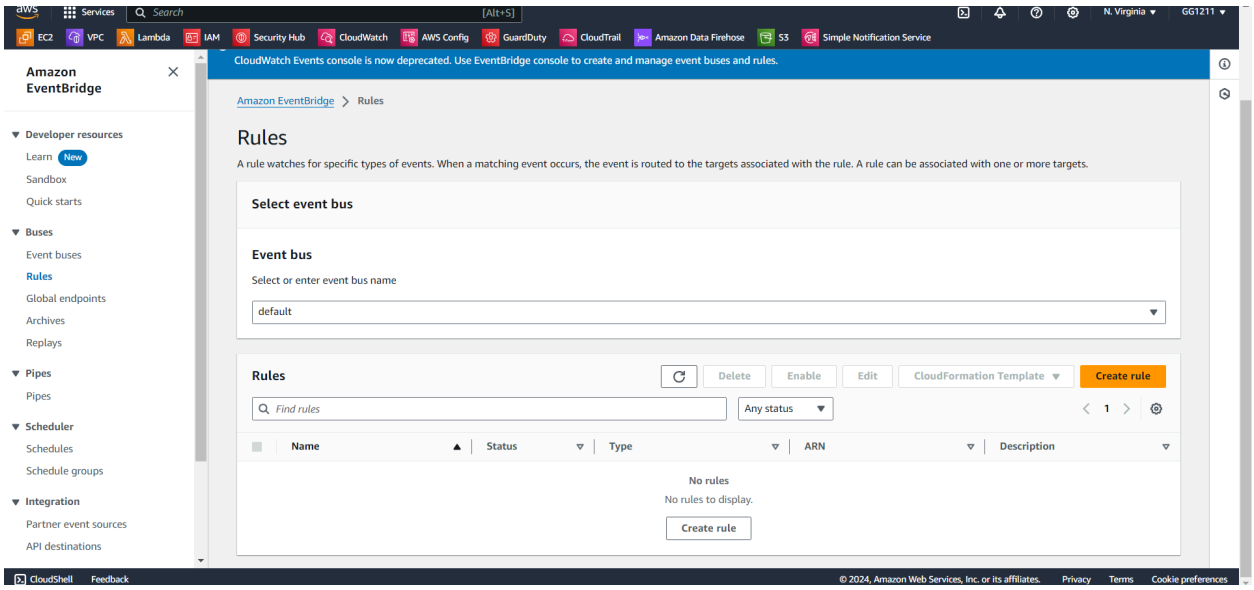
We will then create a new lambda function using these settings and make sure to select the IAM role we made and select it.

Once created input this code

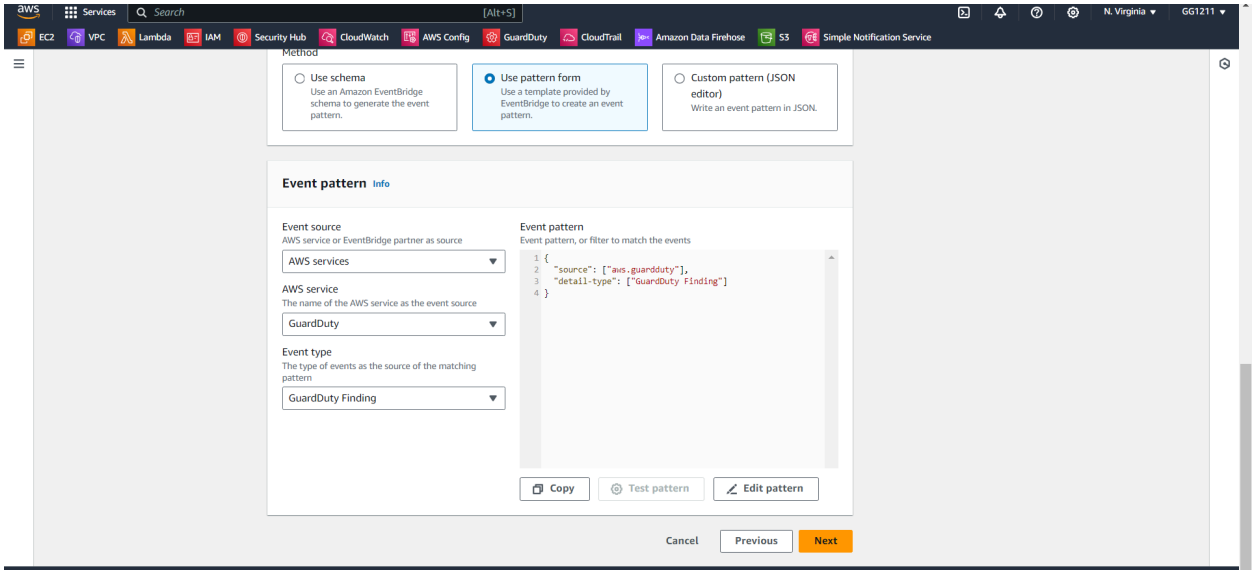
```
import boto3

def lambda_handler(event, context):
    ec2 = boto3.client('ec2')
    guardduty = boto3.client('guardduty')

    finding = event['detail']
    instance_id = finding['resource']['instanceDetails']['instanceId']

    try:
        response = ec2.stop_instances(InstanceIds=[instance_id])
        print(f"Stopped instance: {instance_id}")
    except Exception as e:
        print(f"Error stopping instance: {str(e)}")
```

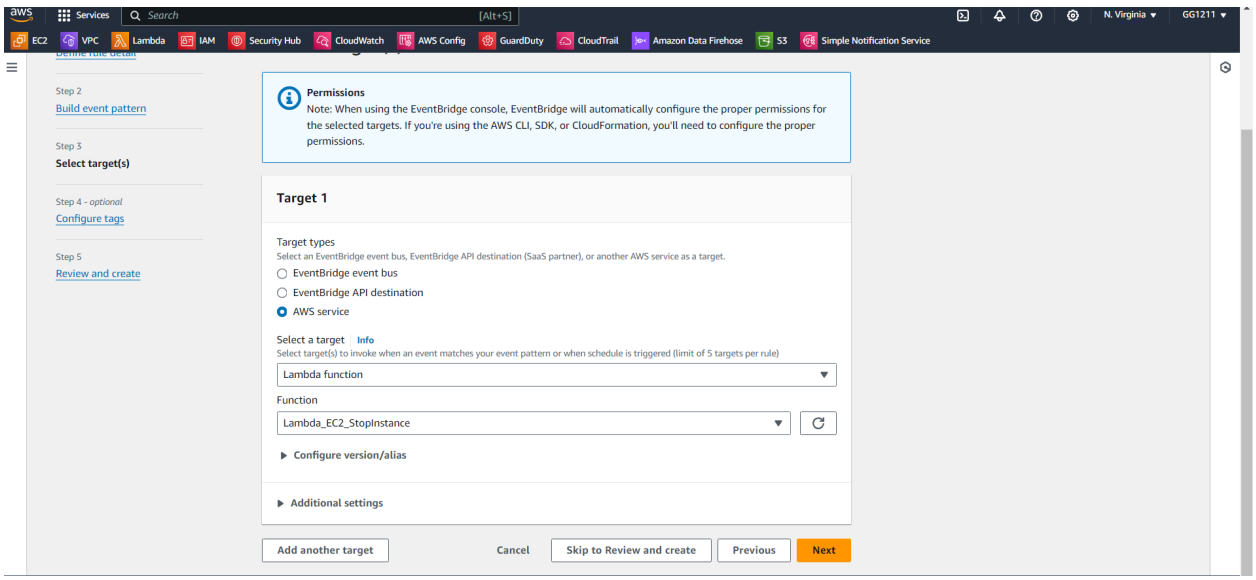


Click deploy and you should be good.



Next go to cloud watch and then rules to access eventbridge.

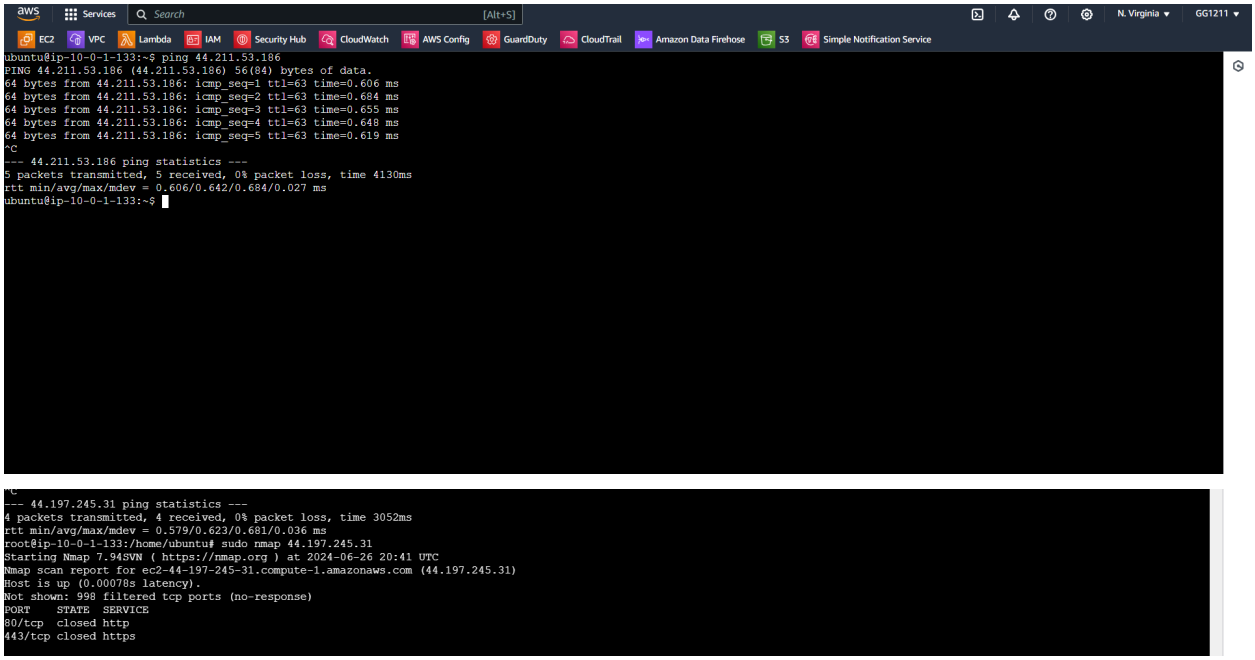




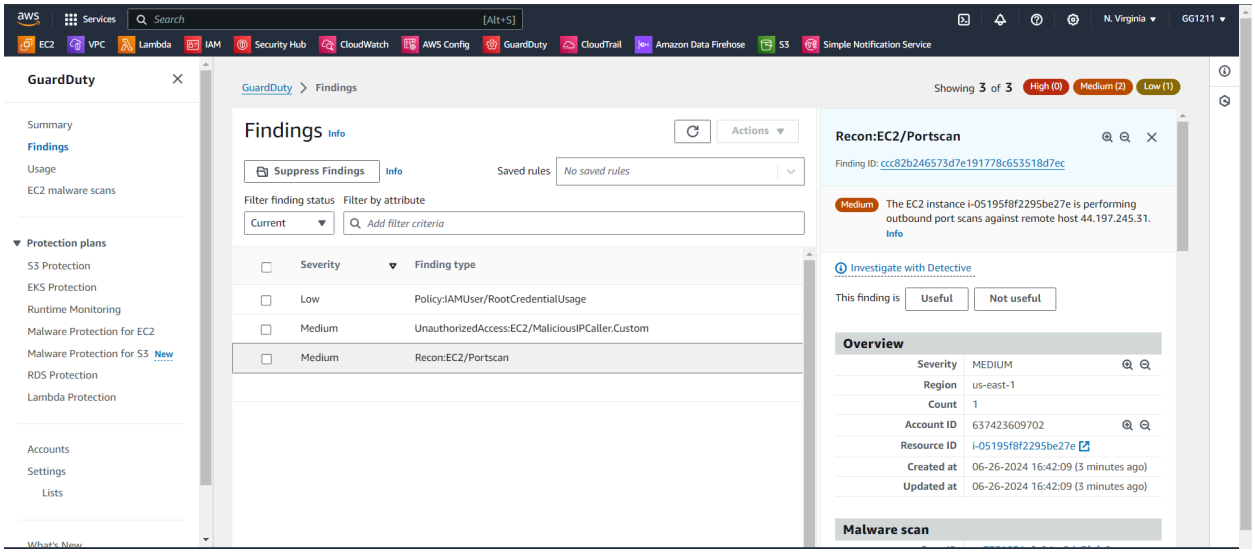
Direct the rule to the lambda function we created and finally create the rule.

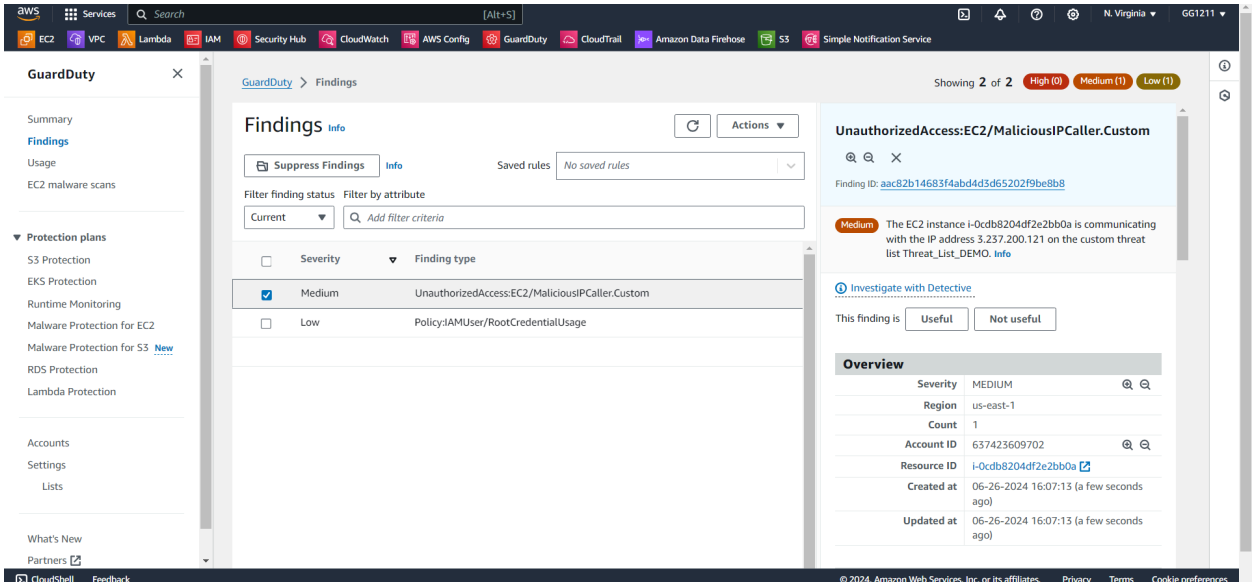
DEMO

SSH into both your vulnerable and malicious machine. ping or curl the vulnerable machine from the malicious actor machine and vice versa. It will take a minute for Guard Duty to capture the event and for the lambda function to start executing.

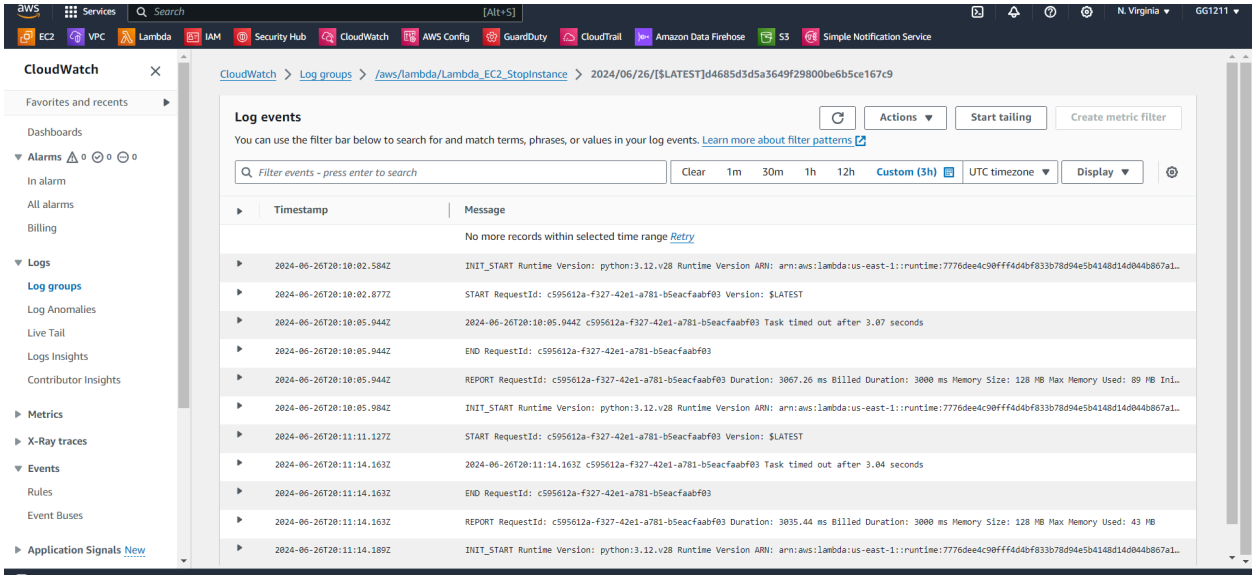


Portscan attack and ping command from malicious machine to vulnerable machine.

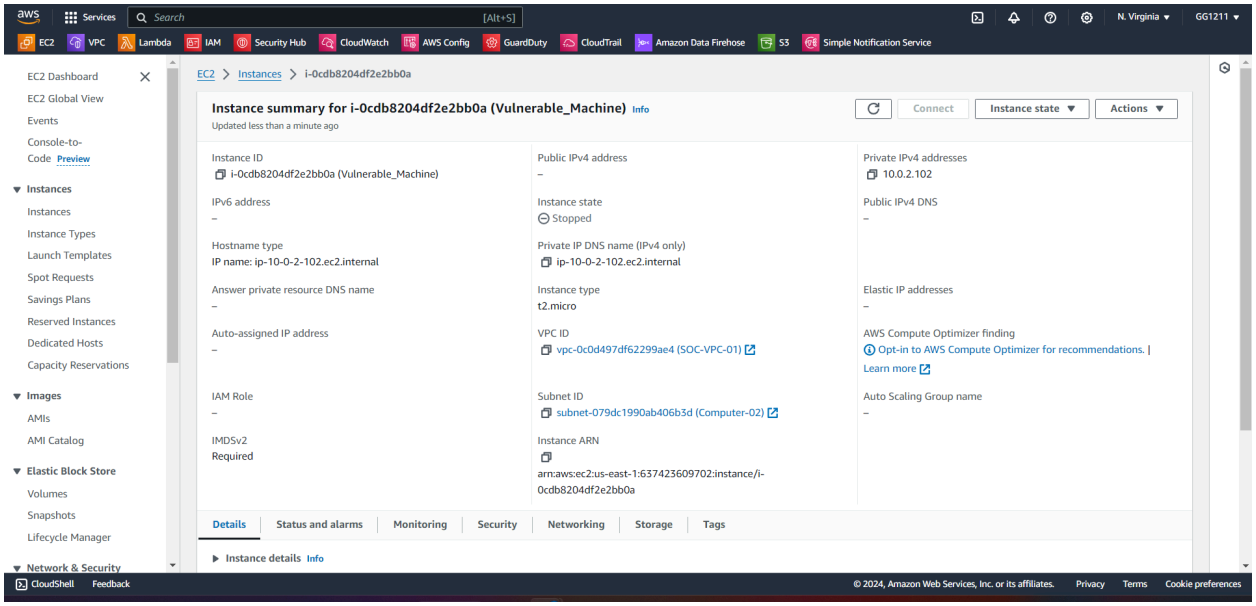




The threat was detected on guard duty.



The cloud watch log group for the lambda function shows a successful execution



Looking back at the vulnerable machine, the instance has stopped.

Monitoring Capabilities

1. **CloudWatch Logs:**
    - **Usage:** Capture and store log files from Amazon EC2 instances, AWS Lambda functions, and other services.
    - **Integration:** Easily integrate with other AWS services such as CloudTrail, VPC Flow Logs, and AWS WAF for comprehensive log analysis.
2. **CloudWatch Metrics:**
    - **Usage:** Monitor metrics related to AWS resources and applications.
    - **Integration:** Automatically collects metrics from AWS services like EC2, S3, DynamoDB, Lambda, etc.
3. **Dashboards and Alarms:**
    - **Dashboards:** Create customizable dashboards to visualize operational data and security metrics in real-time.
    - **Alarms:** Set up alarms based on thresholds for metric values, allowing proactive monitoring and alerting.

Implementing Security Monitoring with CloudWatch

1. **Enable CloudTrail Integration:**
    - Use CloudWatch Logs to collect and analyze AWS CloudTrail logs. This helps you track API calls and resource changes across your AWS environment.
2. **VPC Flow Logs:**

    - Store VPC Flow Logs in CloudWatch Logs for detailed analysis of network traffic patterns and potential security incidents.
3. **Custom Metrics and Logs:**
    - Create custom metrics and logs to monitor specific security-related events or application performance metrics.

Advantages of Using CloudWatch Exclusively

1. **Centralized Monitoring:**
    - CloudWatch provides a single interface to monitor logs and metrics across your AWS resources, simplifying management and troubleshooting.
2. **Integration with AWS Services:**
    - Seamless integration with various AWS services ensures comprehensive visibility into your cloud infrastructure.
3. **Scalability and Reliability:**
    - Built to handle large-scale environments with high reliability and scalability, ensuring that you can monitor and manage even complex architectures effectively.
4. **Cost Efficiency:**
    - CloudWatch offers flexible pricing based on the volume of logs ingested and metrics monitored. Utilizing CloudWatch exclusively can potentially simplify cost management compared to using multiple monitoring solutions.

Considerations

1. **Data Retention and Management:**
    - Plan for data retention policies in CloudWatch to ensure compliance and manage storage costs effectively.
2. **Complex Use Cases:**
    - For advanced analytics or specific use cases requiring machine learning insights or deeper analysis, consider additional AWS services or third-party tools that complement CloudWatch.

Conclusion

Using Amazon CloudWatch exclusively for your AWS Security Operations Center can be a cost-effective and efficient solution, especially if you prefer a unified platform for monitoring and analyzing logs and metrics across your AWS environment. Evaluate your specific requirements, scalability needs, and integration preferences to determine if CloudWatch meets all your monitoring and security operation needs or if additional tools or services are necessary.

References

- **Optimizing your SOC's threat coverage and data value:** https://m.youtube.com/watch?v=YD6QCEVDw20 (This video by Securonix discusses optimizing data ingestion for threat coverage and highlights the importance of accurate data intake for effective SOC operations.)

- **Live Demo: Get Faster Time-to-Value Across Your SOC Workflow with Hunters SOC Platform:** https://m.youtube.com/watch?v=YD6QCEVDw20 (This Hunters SOC Platform demo showcases how their platform connects to various data sources and emphasizes the importance of data readiness for analysis.)

- **Upgrade your SOC from log collection to true threat detection & response:** https://m.youtube.com/watch?v=YD6QCEVDw20 (This Microsoft and CyberProof discussion focuses on transitioning from basic log collection to true threat detection and response. While not directly addressing ingestion requirements, it highlights the importance of comprehensive data collection for effective SOC functionality.)

- **AWS re:Inforce 2022 - Using AWS security services to build your cloud security operations baseline** -
  AWS re:Inforce 2022 - Using AWS security services to build your cloud sec…
  (An AWS conference that explains GuardDuty and Security Hub in detail as it relates to building a cloud security operation).

- **SOC GUIDE** https://github.com/cyb3rxp/awesome-soc/blob/main/README.md

- AWS Security Best Practices and Beginner Tools

AWS Summit DC 2021: Security best practices, common mistakes, and solutions

- VPC Flow Log Tutorial

https://docs.aws.amazon.com/vpc/latest/tgw/flow-logs-cwl.html

- GaurdDuty Configuration Stufff

https://repost.aws/questions/QUBu_mmiH7QWKGq-MmZ_p2bQ/best-method-to-send-guardduty-logs-to-opensearch

- Guard duty events to cloud watch

https://docs.logz.io/docs/shipping/aws/guardduty/

- Explanation of why Sec Hub is not a siem solution by itself

Getting Hands on with Amazon GuardDuty - AWS Virtual Workshop

- Guard Duty automated response architecture

Amazon GuardDuty Deep Dive

- Exporting cloudwatch logs to s3

https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/S3ExportTasksConsole.html#ExportSingleAccount