Project Title:AWS CloudWatch Dashboards for Billing, Logs, Network Performance, and Security & Compliance Monitoring

Objective-:

To create a practical, low-cost AWS monitoring system using CloudWatch. The goal was to build four interactive dashboards to track billing, application logs, network activity, and security & compliance using CloudWatch Metrics, Logs Insights, CloudWatch Agent, GuardDuty, and AWS Config.

Services used:

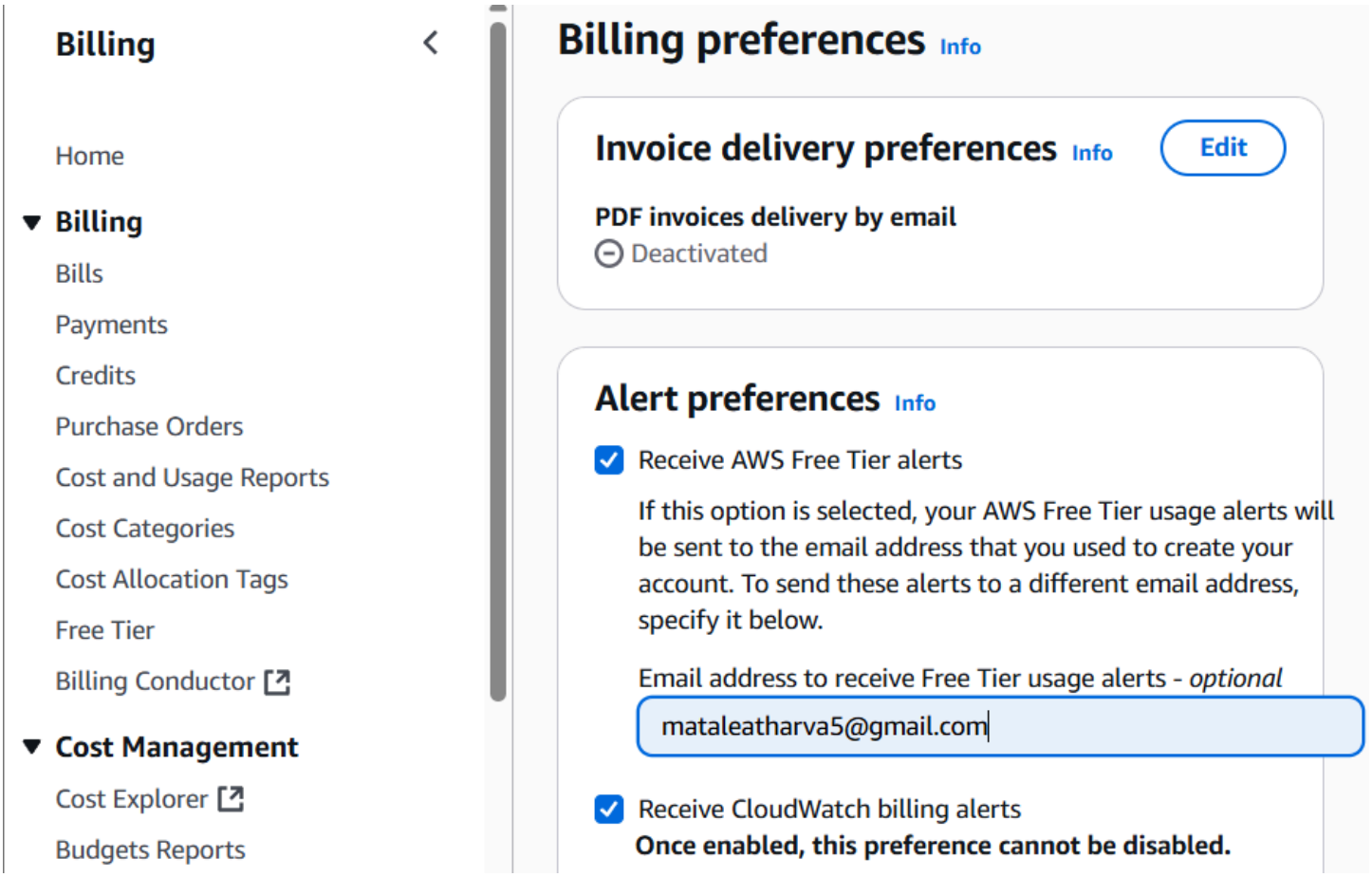
- Amazon CloudWatch (Dashboards, Metrics, Logs Insights)
- AWS Config (for compliance)
- AWS GuardDuty (for security threat detection)
- AWS CloudTrail (for API monitoring)
- IAM (for access control)
- EC2 (host to push logs)
- S3 (used for AWS Config)
- Application Load Balancer (for network monitoring)

Dashboard 1: Billing & Cost Monitoring:

Goal:
This dashboard helps you keep track of your AWS spending in real time.
It shows how much you're being charged overall and which services are costing the most, so you can manage your budget better.

Steps:
1)Enable Billing Alerts:

i)Navigate to *Billing → Preferences → Receive Billing Alerts*.
ii)This step ensures billing metrics appear in CloudWatch.

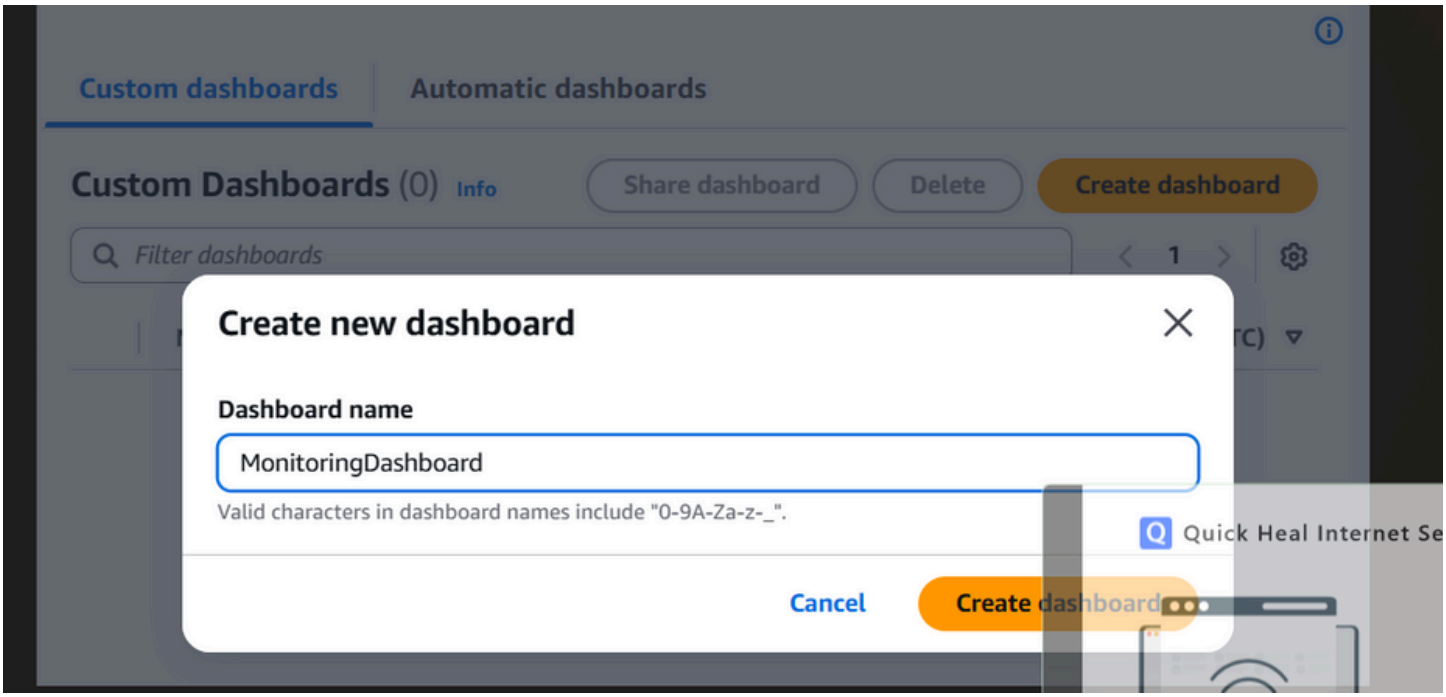


iii)we can see that the metrics are available in the cloud watch metrics.

2)Creating Dashboard:

Go to CloudWatch > Dashboards > Create Dashboard

Name it monitoringDashboard.



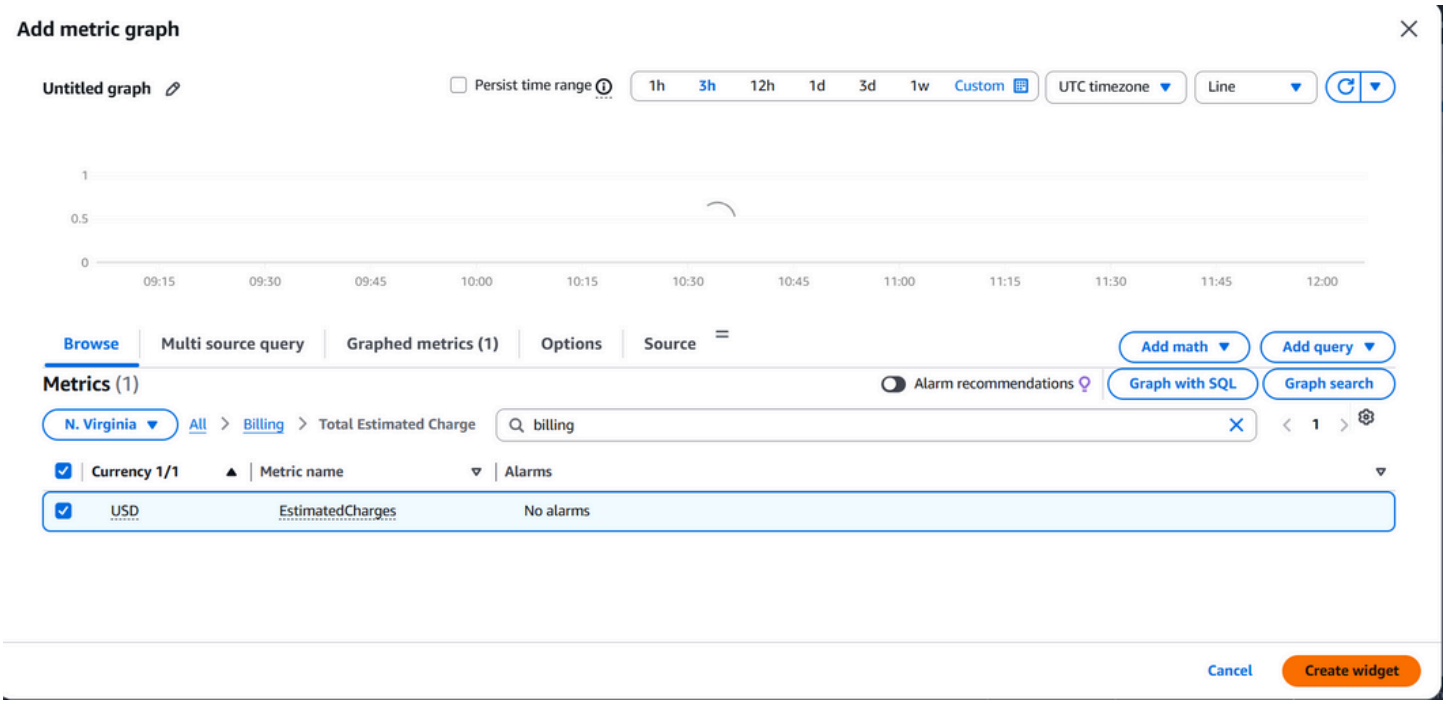
3.Add First Widget – Total Estimated Charges:

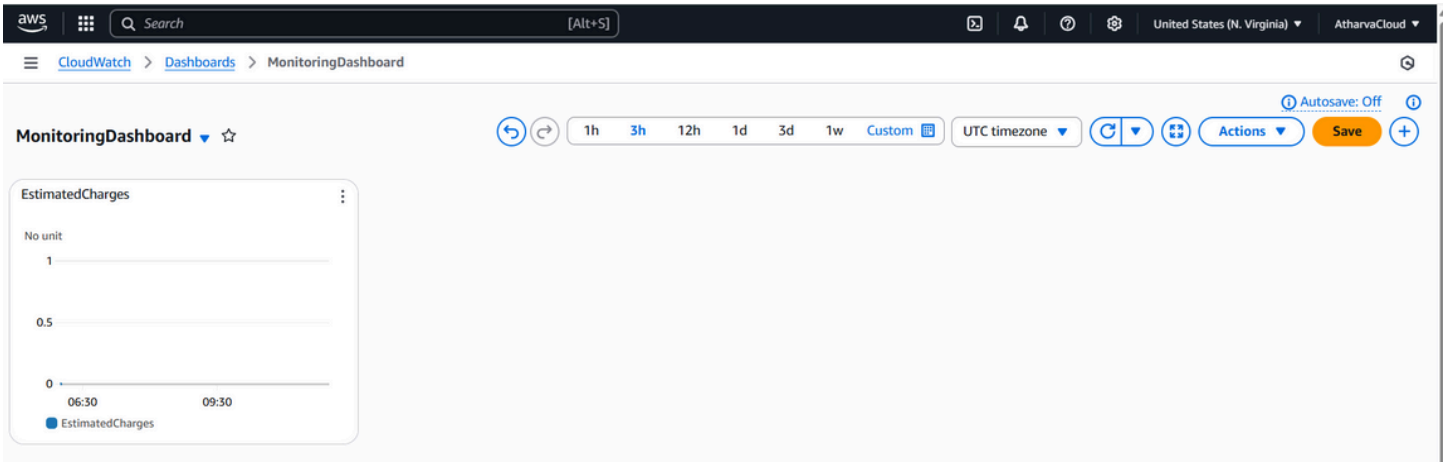
Namespace: AWS/Billing

Metric: EstimatedCharges

Visualization: Line Chart

purpose: The "Total Estimated Charges" widget helps you track the overall AWS spending in real time. It's crucial for cost visibility, especially if you're monitoring usage under a free-tier or have a specific monthly budget.



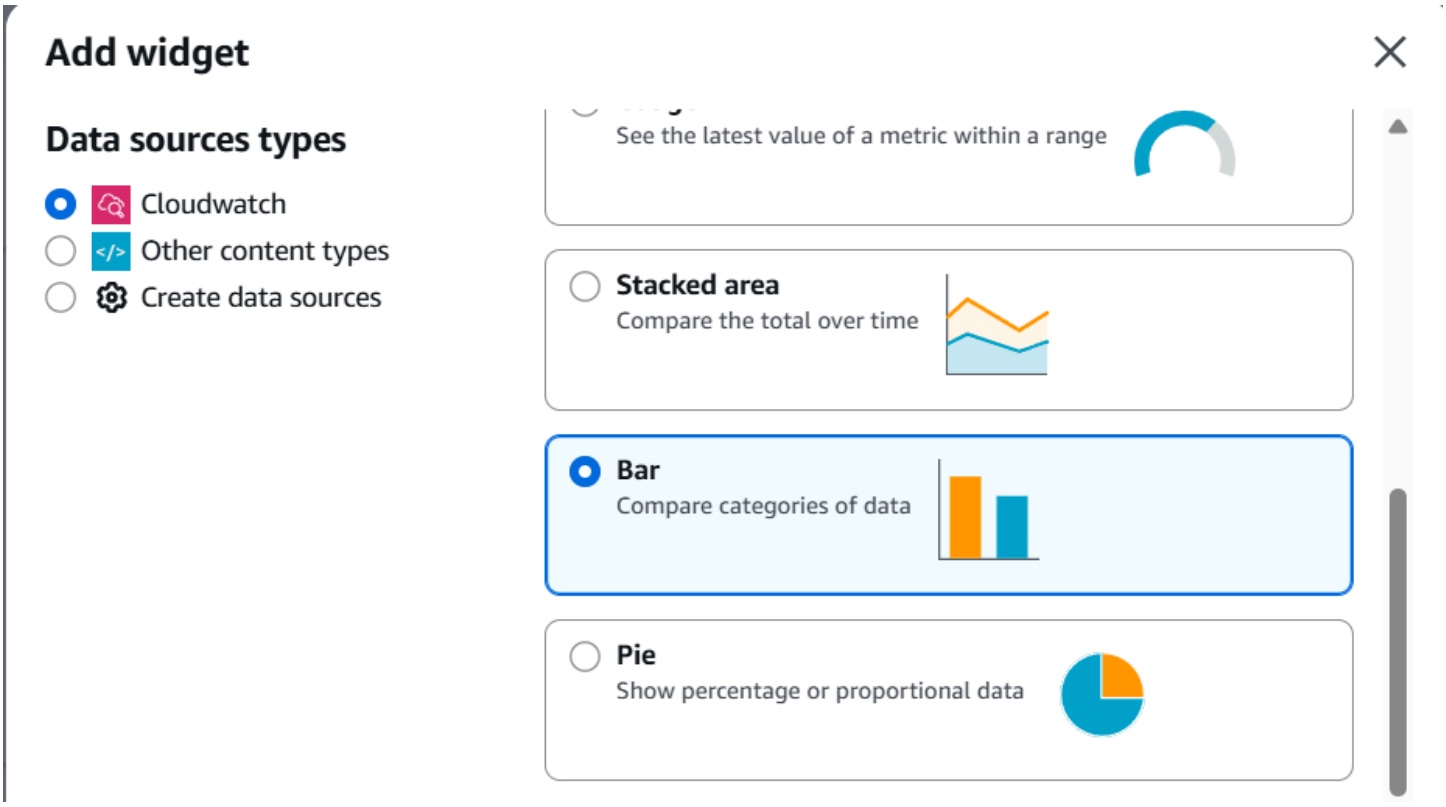


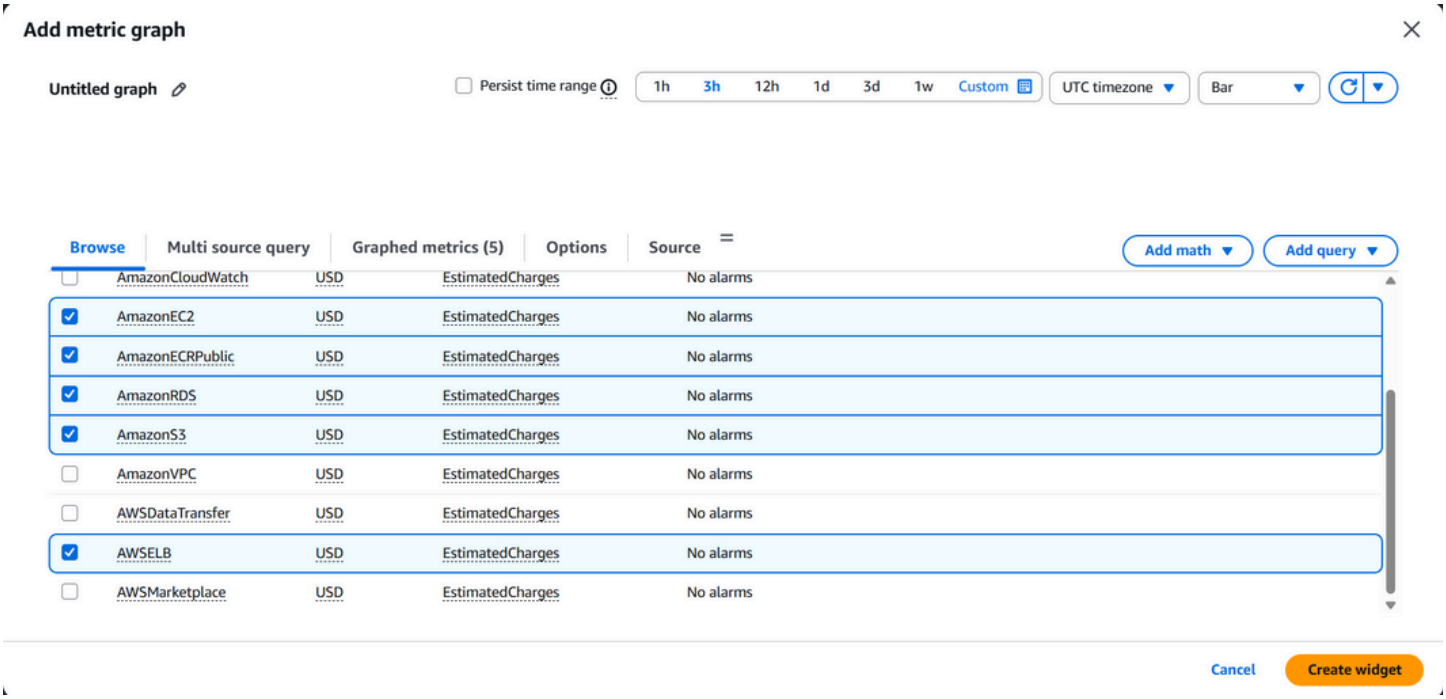
4)Add Second Widget – Charges by Service (Bar Chart)

Grouped by service

Helps identify top-cost contributors

pupose: This widget displays AWS charges grouped by individual services (e.g., EC2, S3, RDS), helping you quickly identify which services are contributing the most to your total bill.
It’s useful for pinpointing high-cost services and optimizing your AWS spending accordingly.

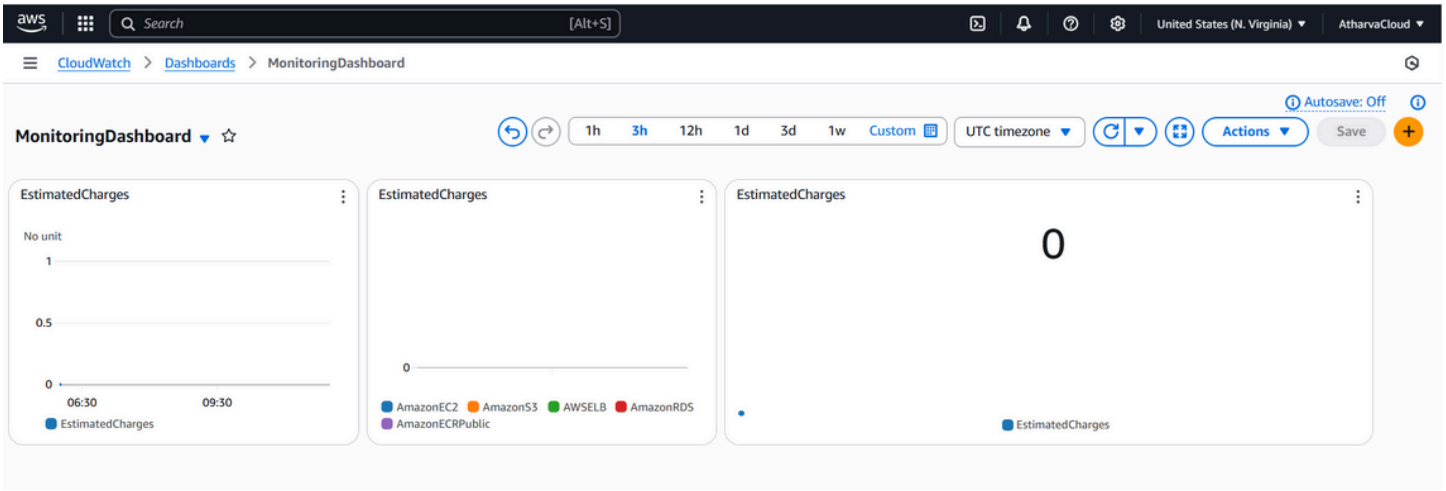




5)Add Third Widget – Number View for Charges

Clean numeric display for quick overview

Purpose: This widget provides a clear, real-time numeric display of your total estimated AWS charges.
It offers a quick-glance overview without needing to analyze charts or graphs.

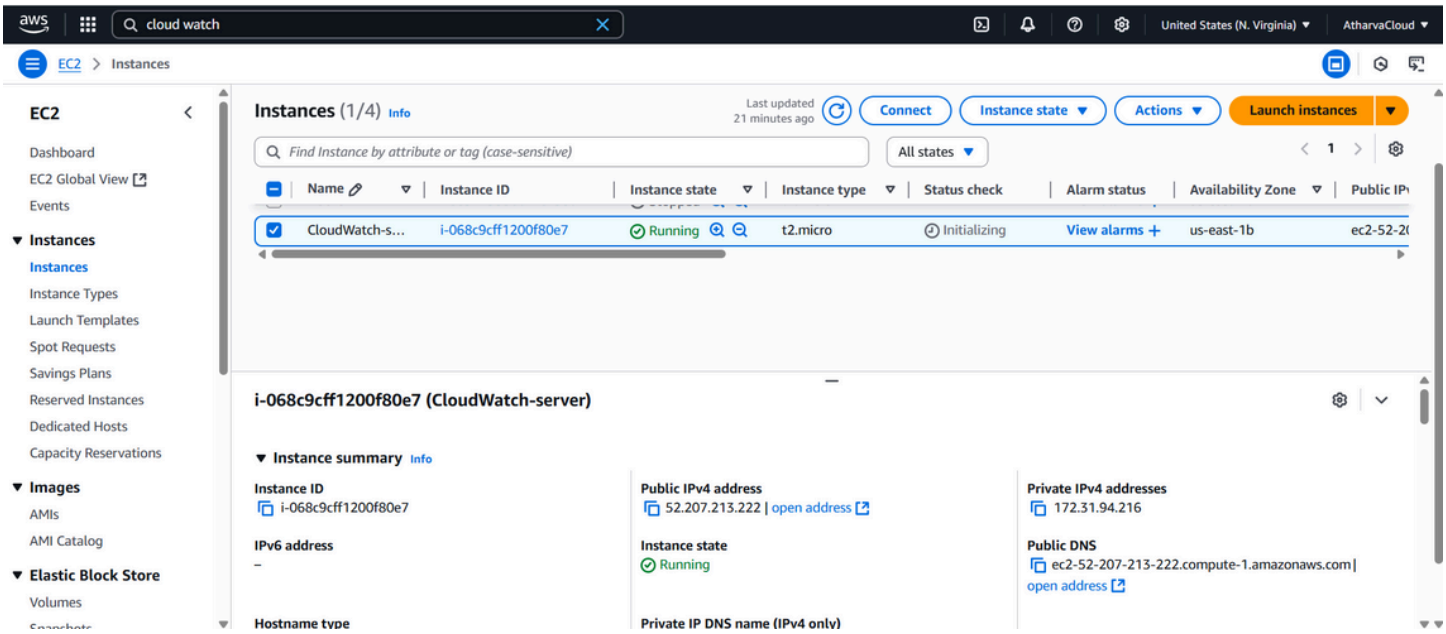


Dashboard 2: Application & System Logs Monitoring:

purpose:
This dashboard helps you view system logs like syslog and auth.log from your EC2 instance in real time.
It lets you easily monitor system activity and detect issues like failed login attempts using CloudWatch Logs and Insights.

Steps:

1)Launch Ubuntu EC2 instance:



2)Install CloudWatch Agent:
Run:
“sudo apt update
sudo apt install -y unzip
wget https://s3.amazonaws.com/amazoncloudwatch-agent/ubuntu/amd64/latest/amazon-cloudwatch-agent.deb
sudo dpkg -i amazon-cloudwatch-agent.deb”

Why?

i)Installing the CloudWatch Agent is necessary to collect and send system-level metrics and log files (like syslog and auth.log) from your EC2 instance to Amazon CloudWatch.

ii)Without the agent, CloudWatch cannot access logs stored locally on your EC2 instance — so it's essential for log monitoring and insights dashboards.

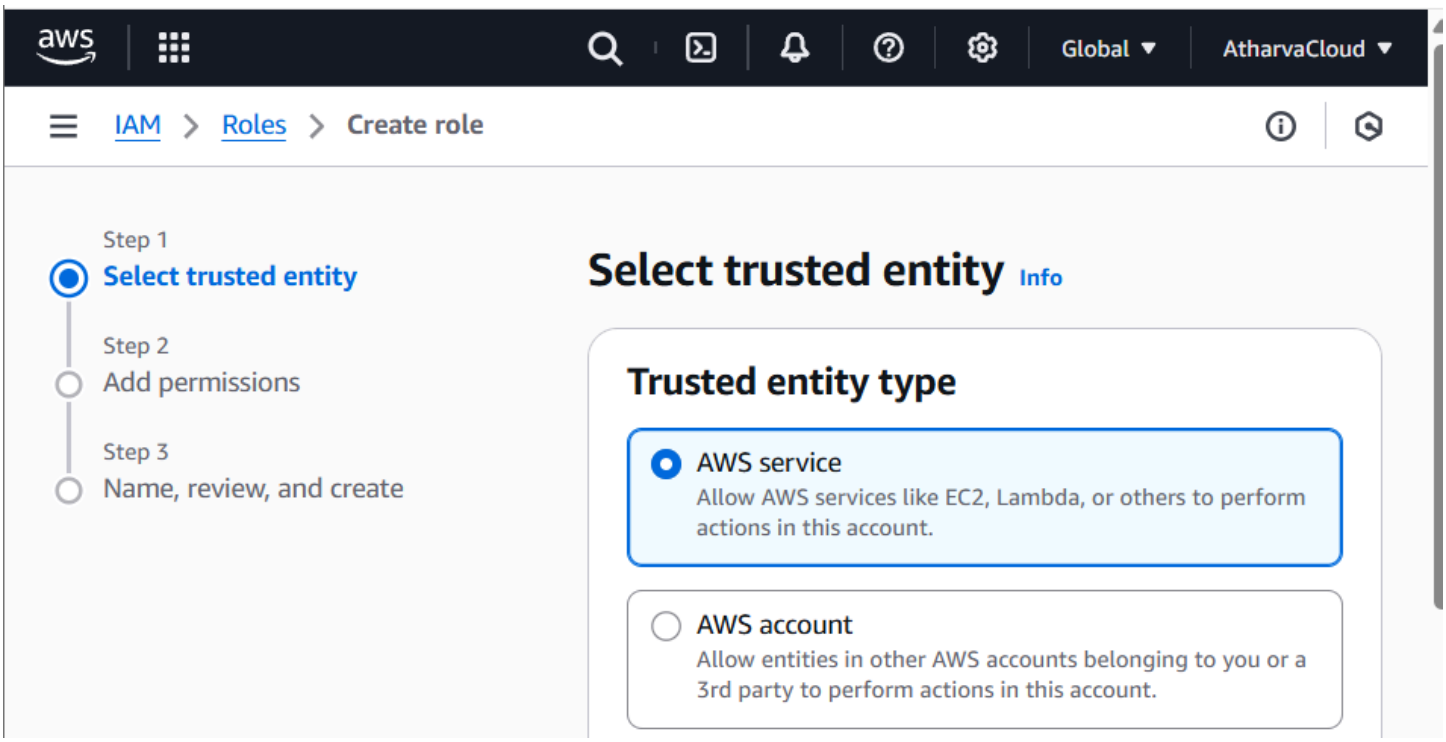
```
ubuntu@ip-172-31-94-216:~$ sudo dpkg -i -E ./amazon-cloudwatch-agent.deb
Selecting previously unselected package amazon-cloudwatch-agent.
(Reading database ... 70681 files and directories currently installed.)
Preparing to unpack ./amazon-cloudwatch-agent.deb ...
create group cwagent, result: 0
create user cwagent, result: 0
Unpacking amazon-cloudwatch-agent (1.300057.0b1161-1) ...
Setting up amazon-cloudwatch-agent (1.300057.0b1161-1) ...
```

```
ubuntu@ip-172-31-94-216:~$ sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-
cloudwatch-agent-config-wizard
=================================================================
= Welcome to the Amazon CloudWatch Agent Configuration Manager =
=                                                               =
= CloudWatch Agent allows you to collect metrics and logs from =
= your host and send them to CloudWatch. Additional CloudWatch =
= charges may apply.                                            =
=================================================================
On which OS are you planning to use the agent?
1. linux
2. windows
3. darwin
default choice: [1]:

Trying to fetch the default region based on ec2 metadata...
I! imds retry client will retry 1 timesAre you using EC2 or On-Premises host
s?
1. EC2
2. On-Premises
default choice: [1]:

Which user are you planning to run the agent?
1. cwagent
2. root
3. others
default choice: [1]:

Do you want to turn on StatsD daemon?
1. yes
2. no
default choice: [1]:
```

```
lled or the Agent will fail to start
. yes
. no
efault choice: [1]:

o you want to monitor any host metrics? e.g. CPU, memory, etc.
. yes
. no
efault choice: [1]:

o you have any existing CloudWatch Log Agent (http://docs.aws.amazon.com/Am
zonCloudWatch/latest/logs/AgentReference.html) configuration file to import
for migration?
. yes
. no
efault choice: [2]:

o you want to monitor any log files?
. yes
```

3)Create IAM Role for EC2 with CloudWatchAgentServerPolicy:

i)The CloudWatch Agent needs permission to push log data (e.g., syslog, auth.log) and metrics from the EC2 to CloudWatch:

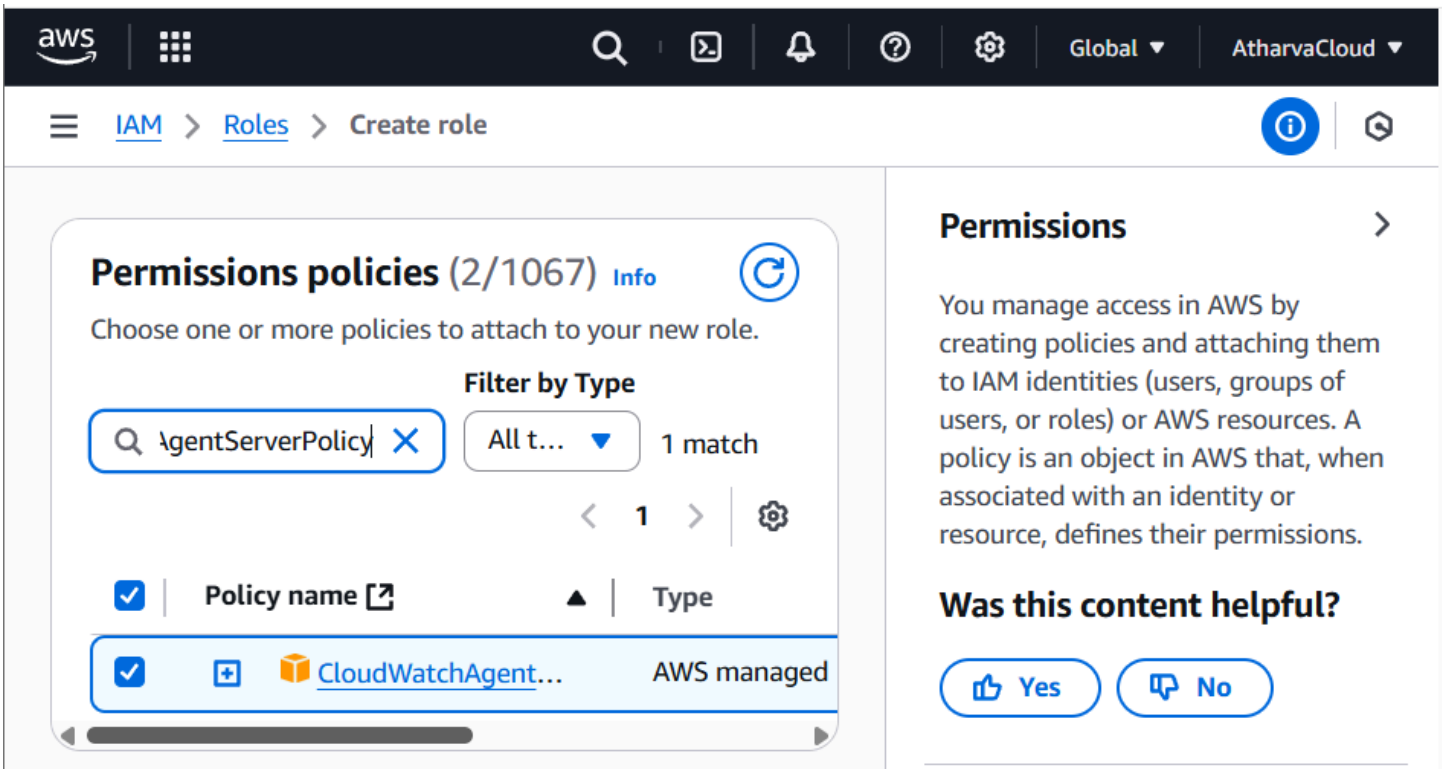


ii) IAM Role for EC2 Instance (Logs Monitoring Dashboard)

Role Type: EC2 instance role

Policy Attached: CloudWatchAgentServerPolicy

Purpose: Allows the CloudWatch Agent on EC2 to push logs (e.g., syslog, auth.log) to CloudWatch Logs.

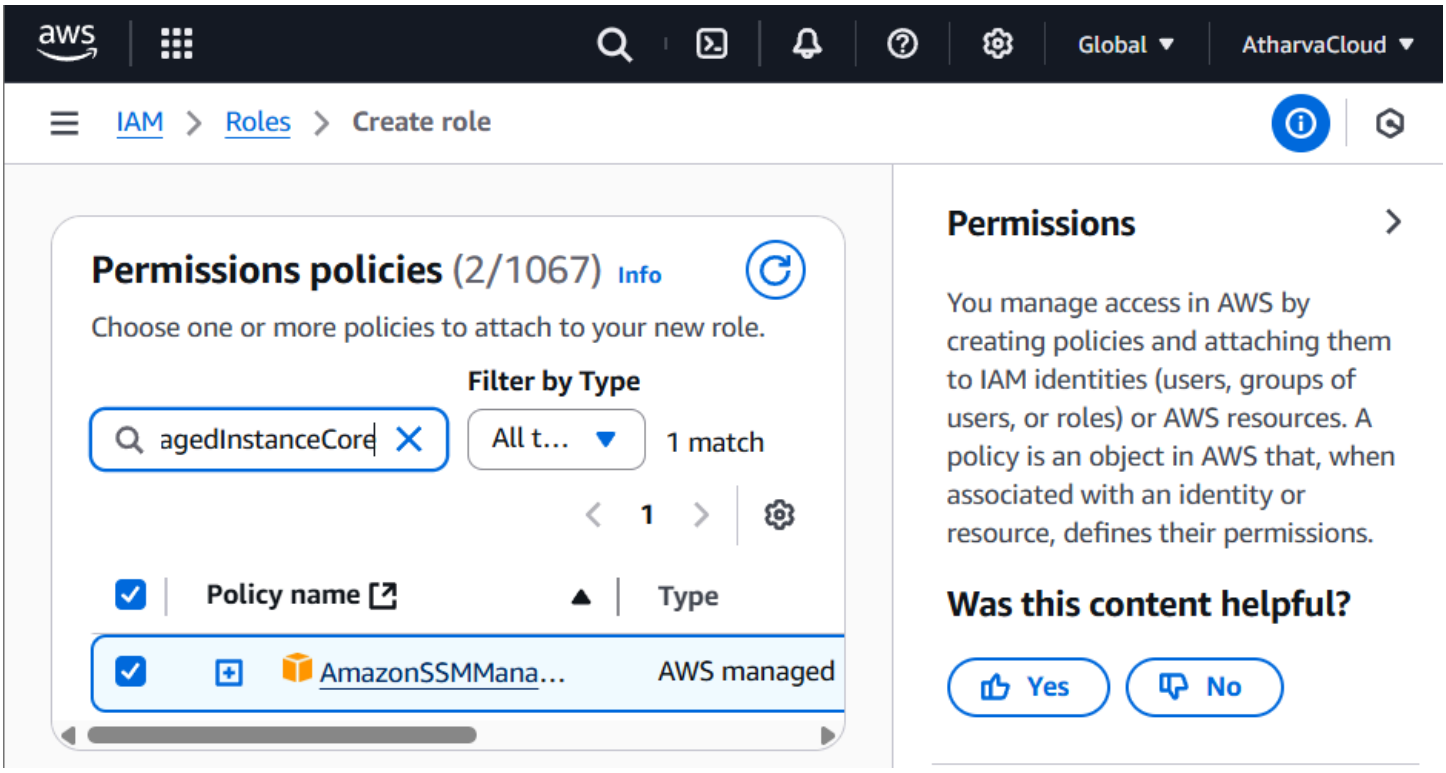


iii)IAM Role for EC2 Management via SSM (Used Alongside CloudWatch Agent)

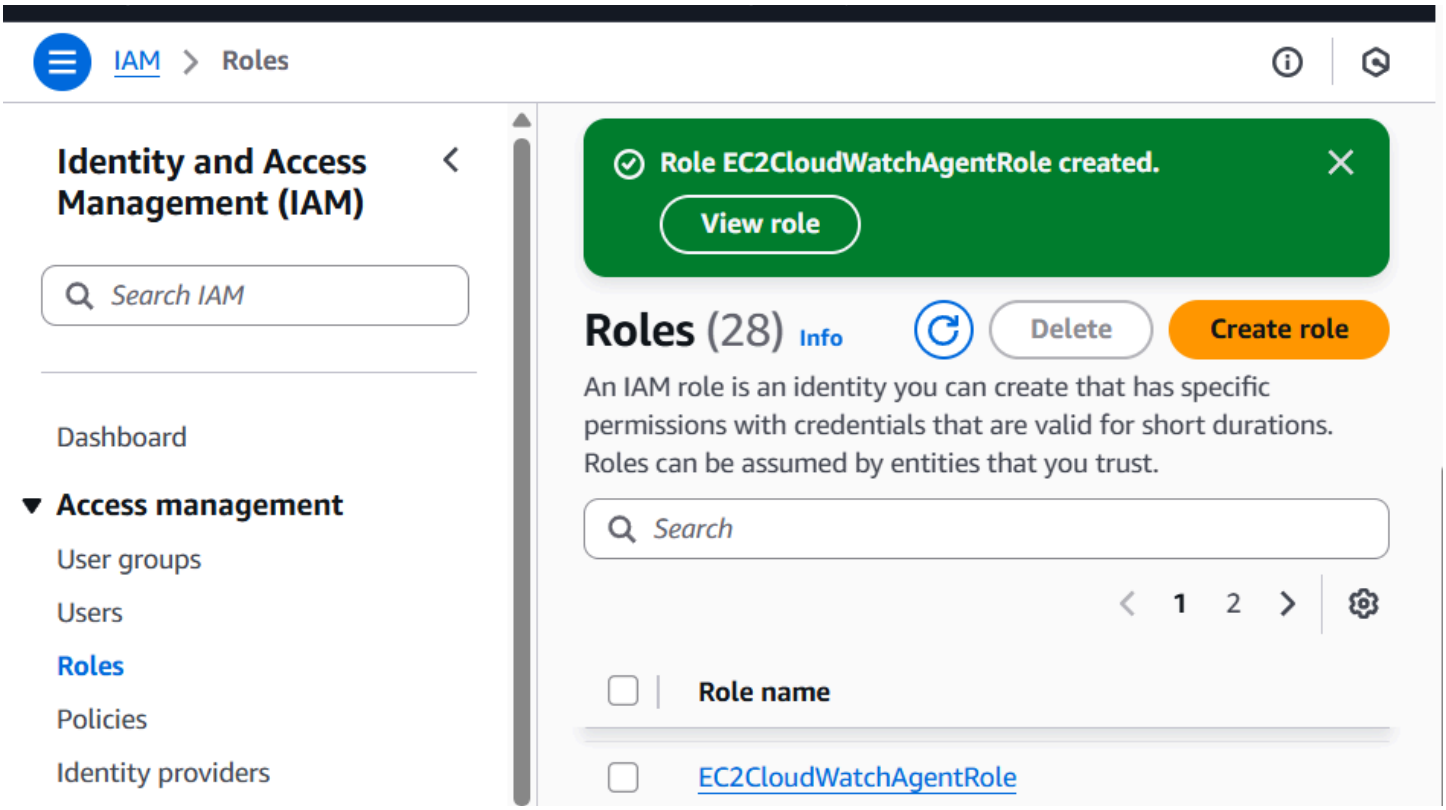
Role Type: EC2 instance role

Policy Attached: AmazonSSMManagedInstanceCore

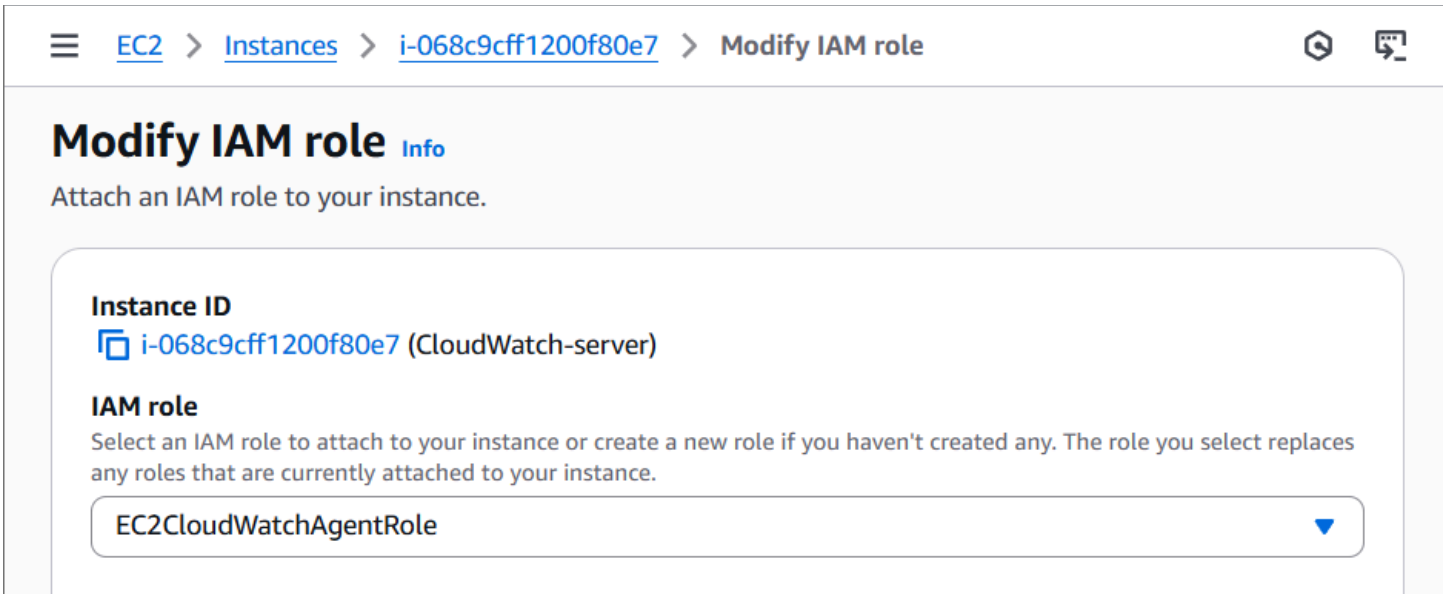
Purpose: Enables AWS Systems Manager (SSM) to manage and interact with the EC2 instance (e.g., for Session Manager, patching, and sending/receiving parameters).



iv)role is created



v)Attached this role to EC2



4)Generate Config File using wizard:

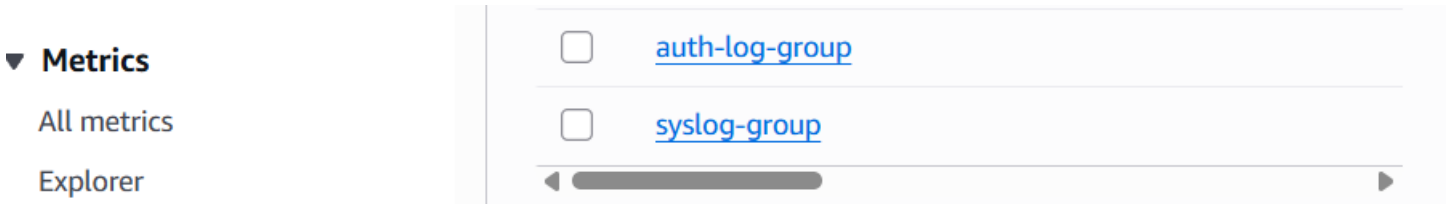
The config wizard helps you create a custom CloudWatch Agent configuration file to define what logs and metrics to collect.
In this case, it's used to specify the log file paths (/var/log/syslog, /var/log/auth.log) that should be sent to CloudWatch for monitoring.

```
{
        "agent": {
                "metrics_collection_interval": 60,
                "run_as_user": "root"
        },
        "logs": {
                "logs_collected": {
                        "files": {
                                "collect_list": [
                                        {
                                                "file_path": "/var/log/auth.log",
                                                "log_group_class": "STANDARD",
                                                "log_group_name": "auth-log-group",
                                                "log_stream_name": "{instance_id}",
                                                "retention_in_days": 30
                                        },
                                        {
                                                "file_path": "/var/log/syslog",
                                                "log_group_class": "STANDARD",
                                                "log_group_name": "syslog-group",
                                                "log_stream_name": "{instance_id}",
                                                "retention_in_days": 30
                                        }
                                ]
                        }
                }
        },
```

5)Start the Agent using below command:

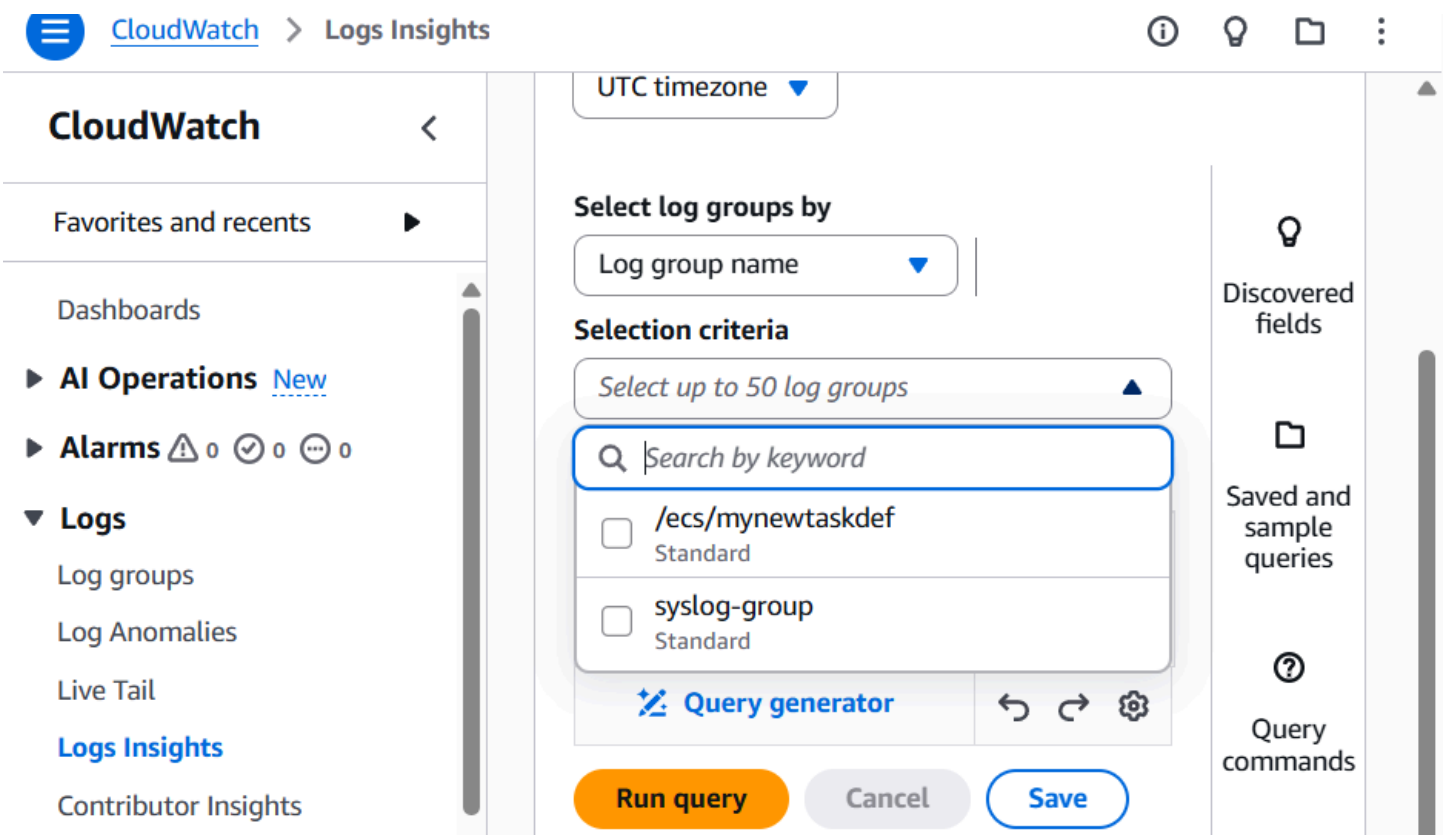
sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl \
 -a fetch-config -m ec2 \
-c file:/opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.d/file_config.json \
-s

iii)log groups are created :

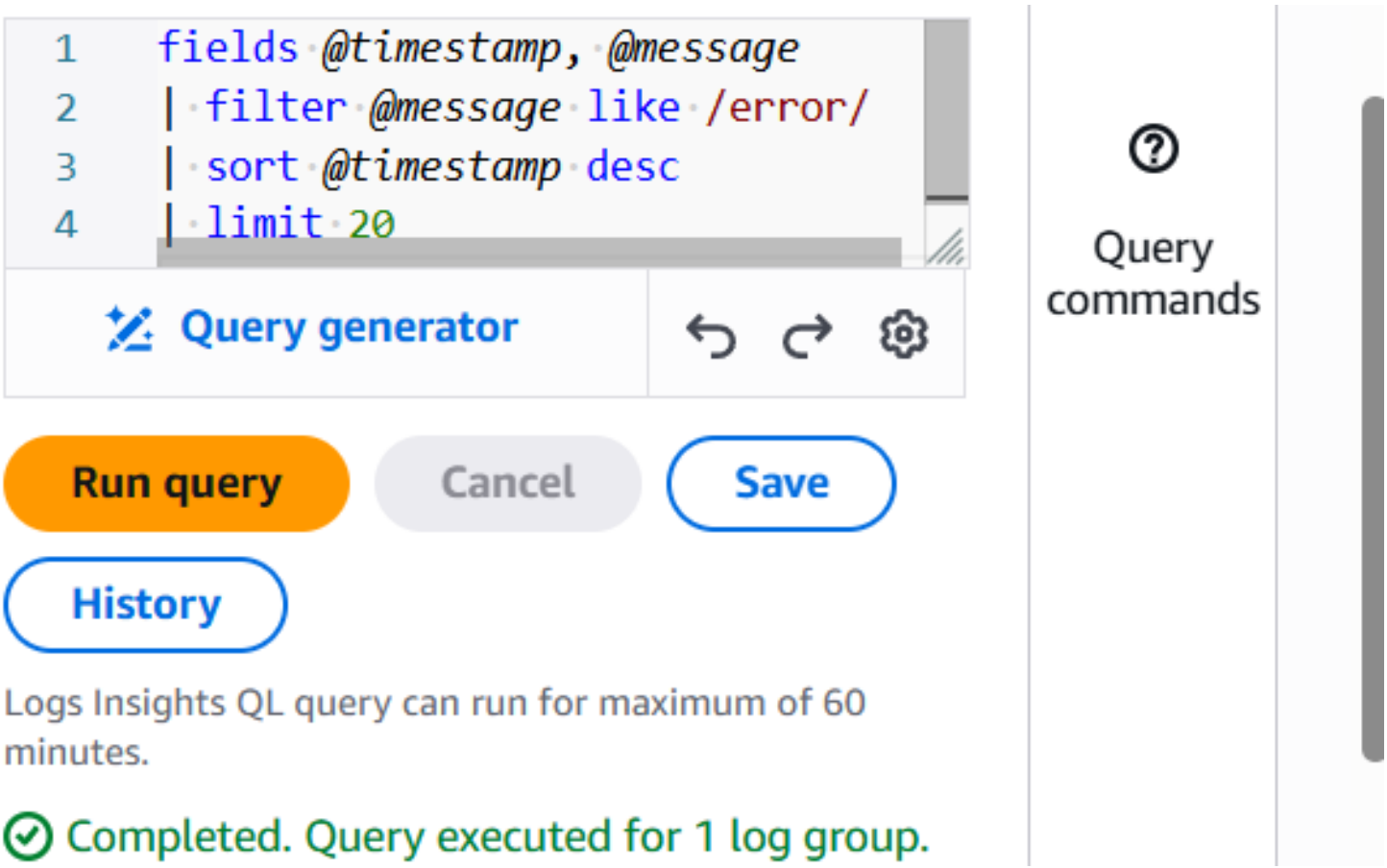


6)Use CloudWatch Logs Insights Queries:

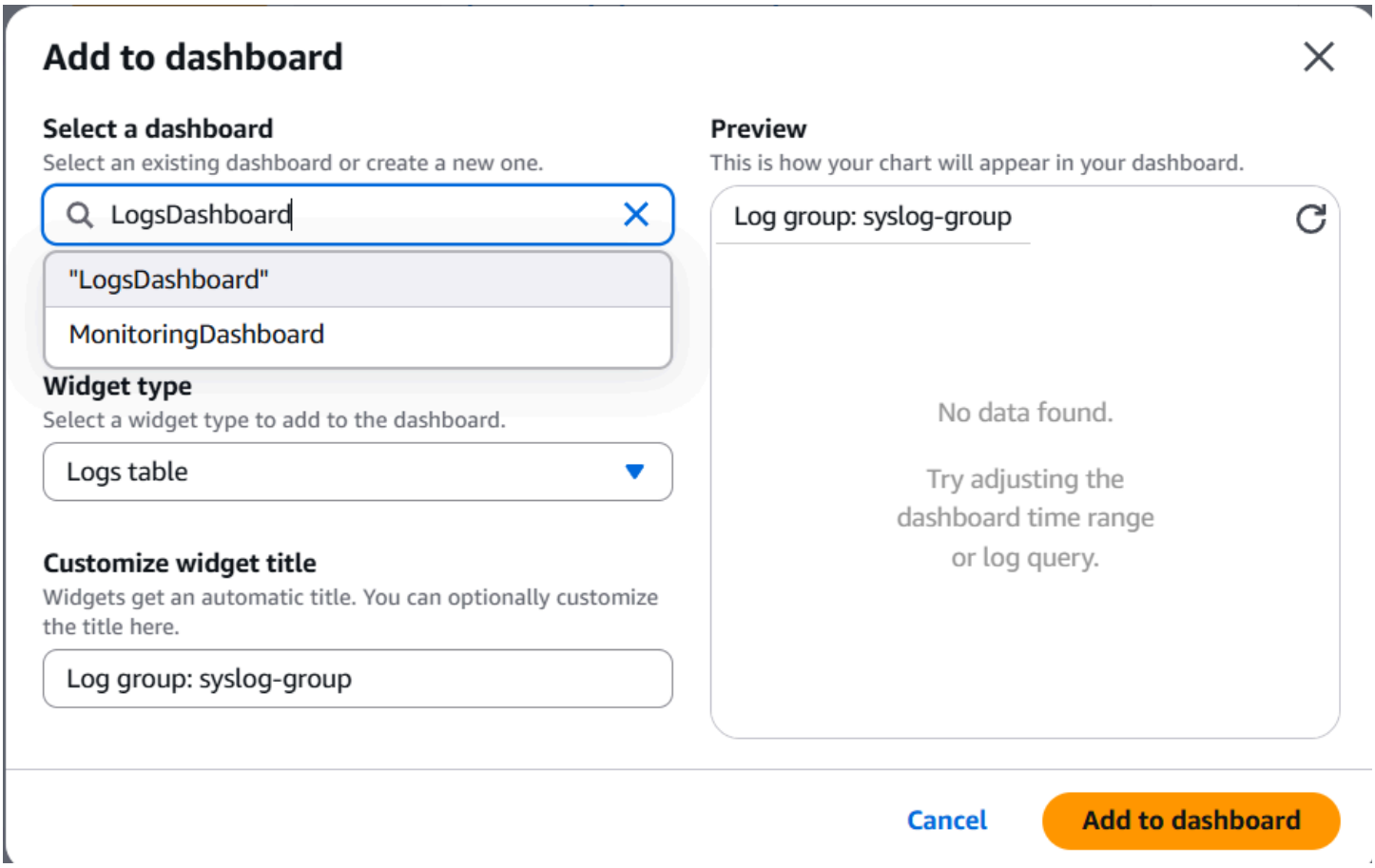
i)Query1(syslog):



ii) This CloudWatch Logs Insights query filters log messages containing the word "error", sorts them by timestamp in descending order (latest first), and shows the top 20 results.
It displays two fields: the log time (@timestamp) and the message content (@message) for quick error analysis.

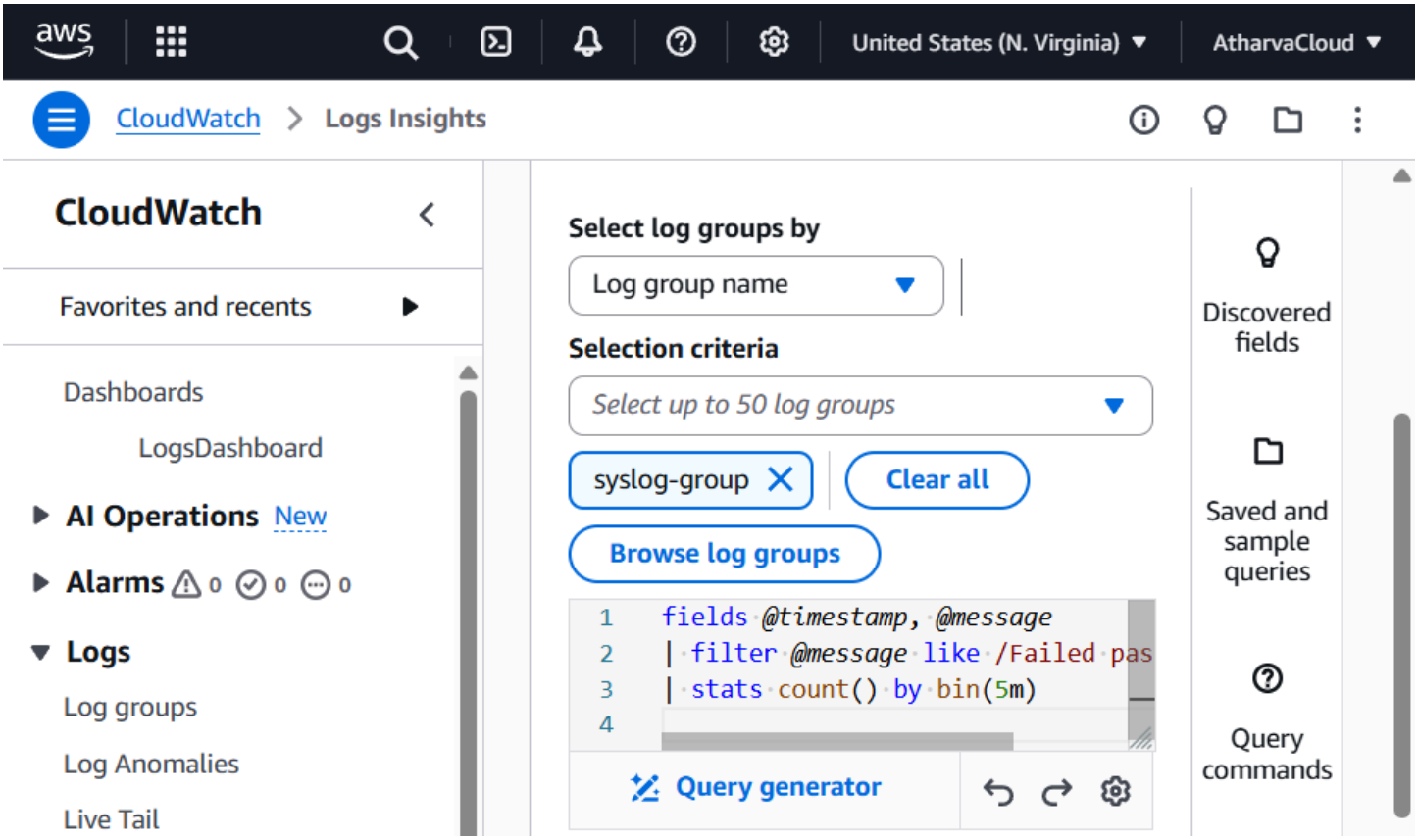


iii)Query2(auth log):

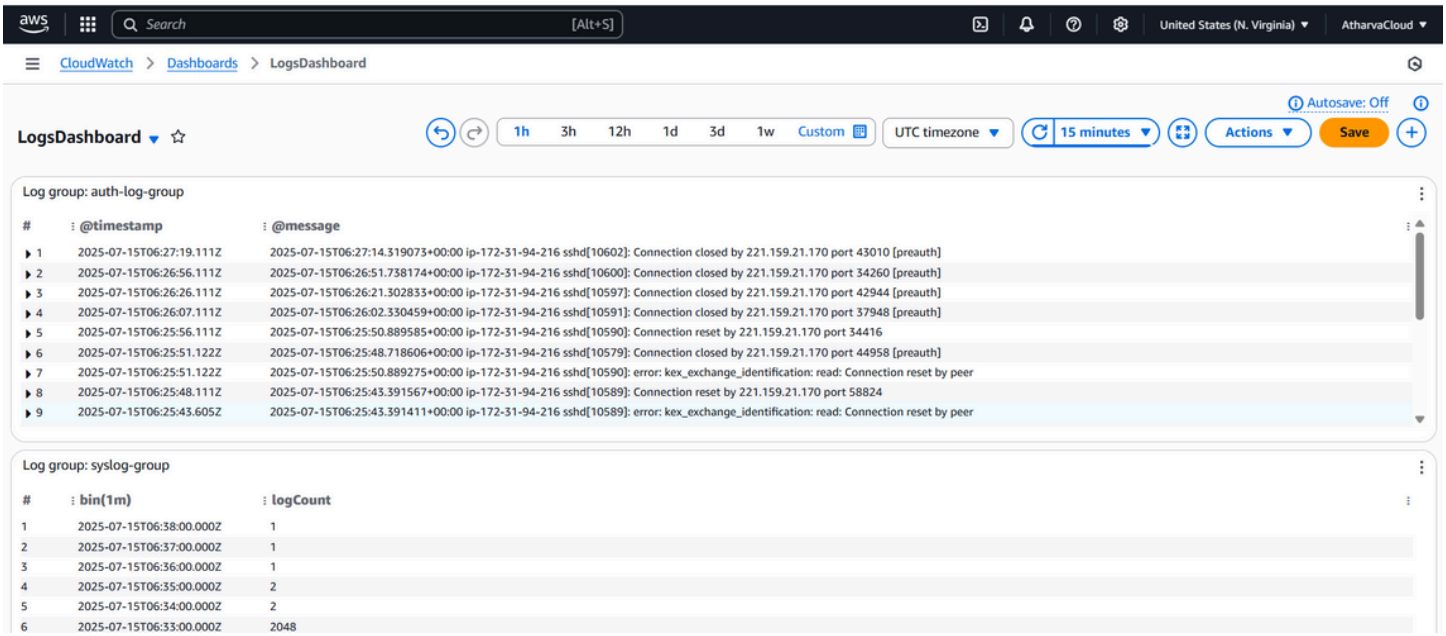


iv)This below query looks for log messages in the syslog-group that contain the phrase "Failed password" (usually failed login attempts).

It then counts how many such events happened every 5 minutes, helping you spot unusual login activity over time.



v)we can see the two widgets:

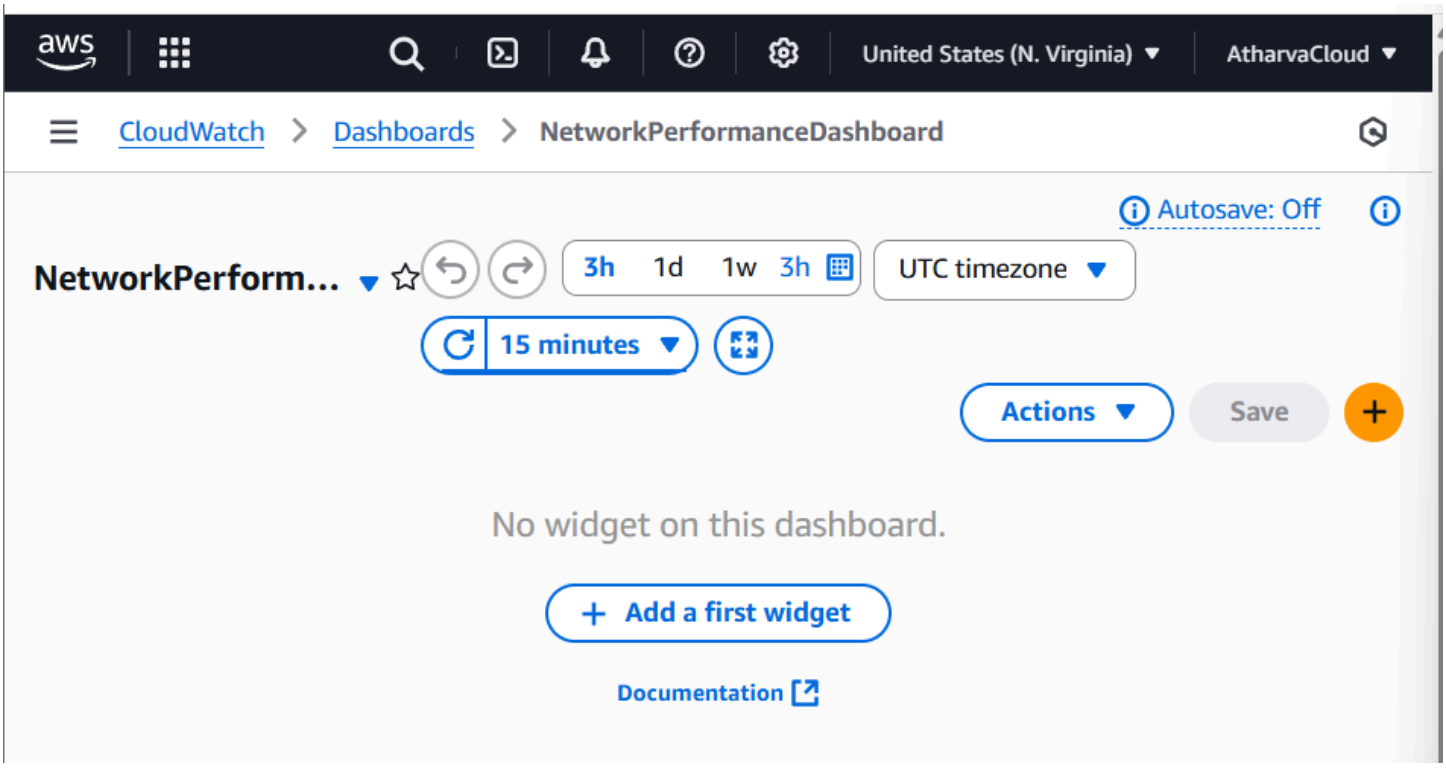


Dashboard 3: Network Performance Monitoring:

Purpose:
This dashboard helps you monitor network traffic and performance of your EC2 and ALB resources in real time.
It shows how much data is flowing, how fast targets are responding, and highlights any 4xx/5xx errors for quick troubleshooting.

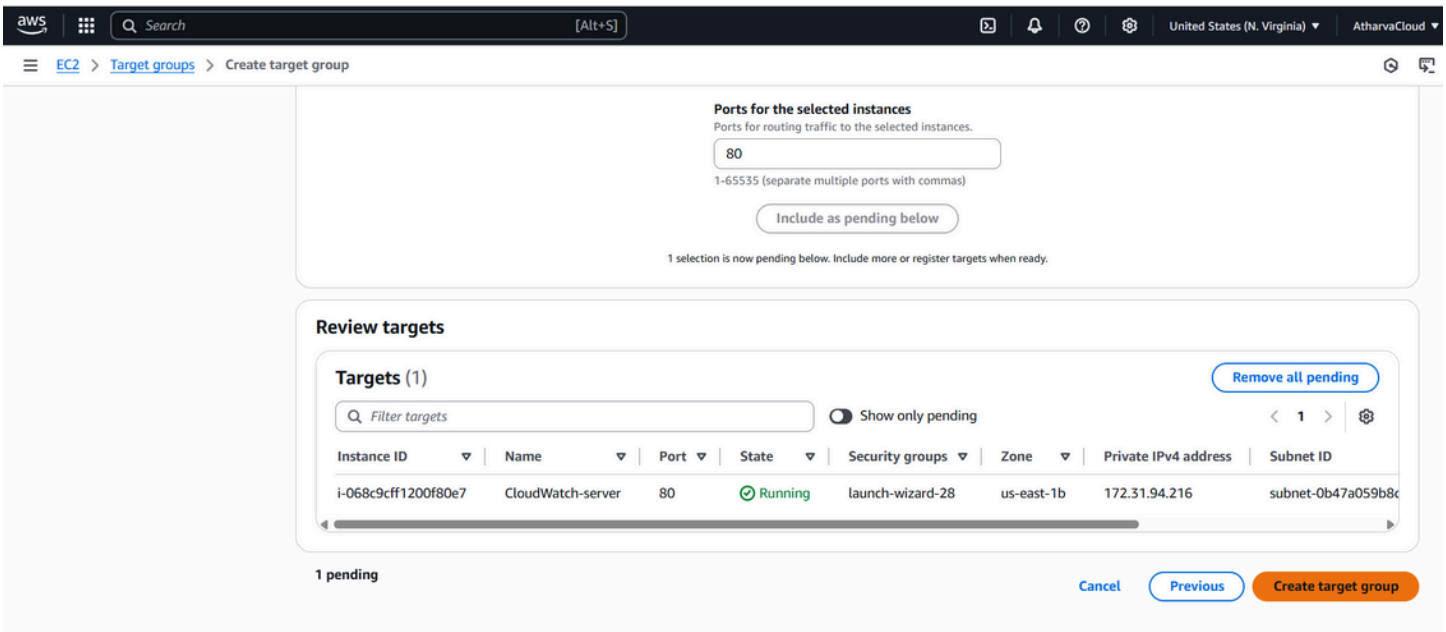
Steps:

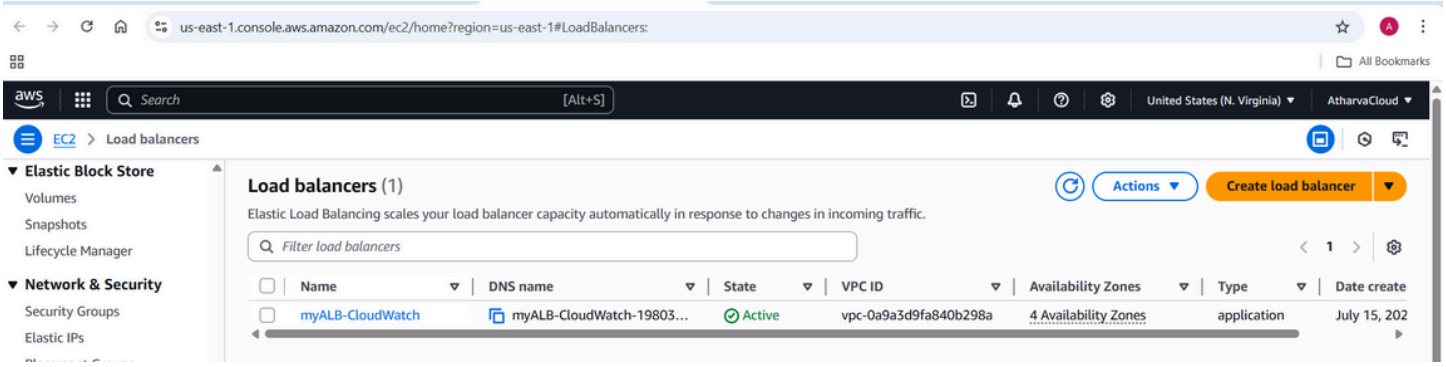
1)Create Dashboard → Add EC2 Network Metrics



2)Create Application Load Balancer (ALB)

- Create Target Group (add EC2 instance)
- Create ALB (enable port 80)
- Install NGINX on EC2 for real HTTP response





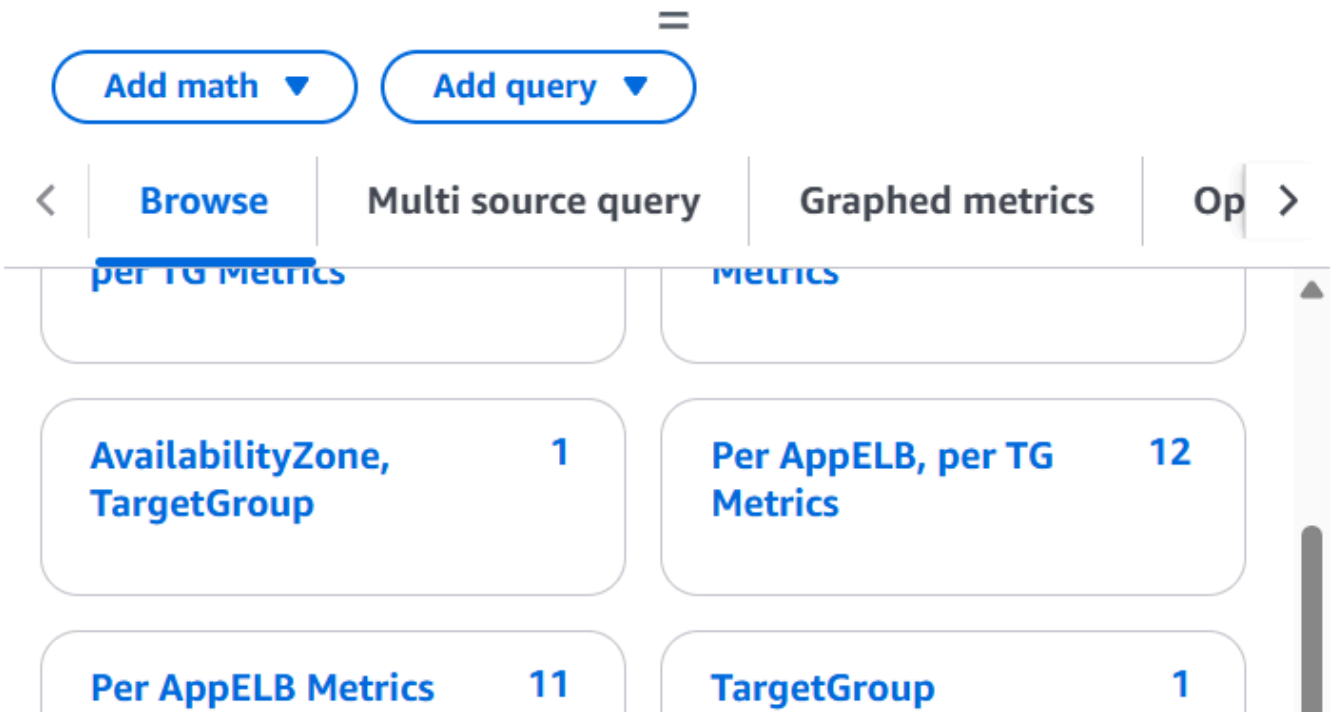
3)Why Metrics Weren't Initially Visible

ALB metrics appear only after valid HTTP traffic is processed.

Before NGINX, targets were unhealthy → no metrics

4)Add ALB Widgets

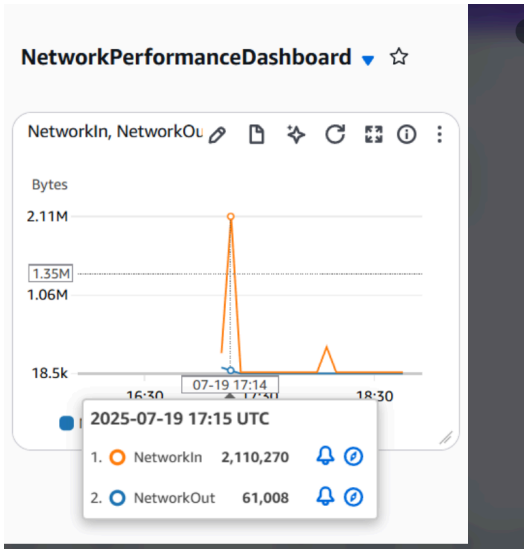
- RequestCount (Line Chart)
- TargetResponseTime (Line Chart)
- HTTPCode_Target_4XX_Count (Gauge)
- HTTPCode_Target_5XX_Count (Gauge)



i)Added 1st widget for NetworkIn and NetworkOut:

These metrics show the amount of incoming (NetworkIn) and outgoing (NetworkOut) data from your EC2 instance.

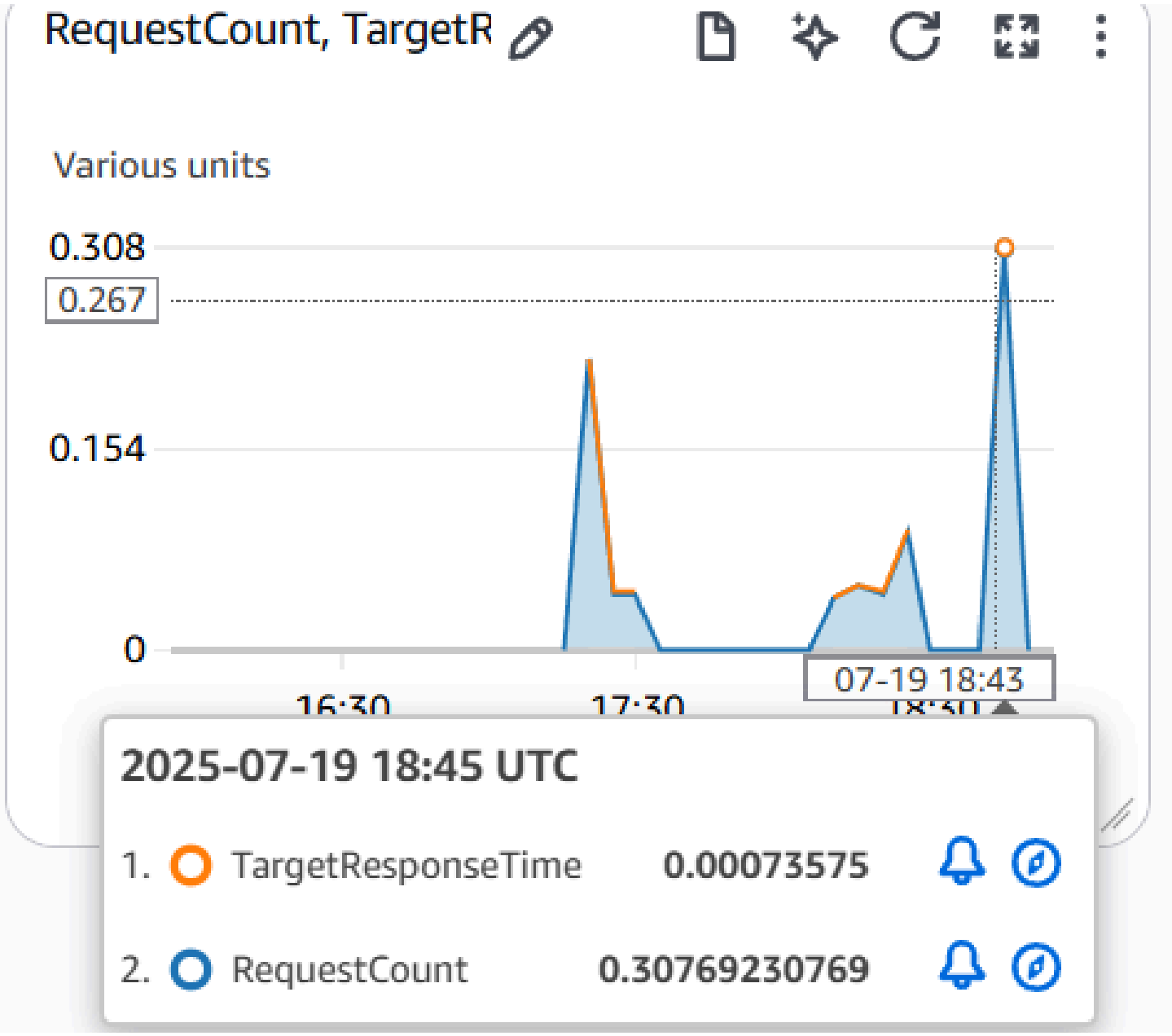
They help you monitor real-time traffic flow, detect abnormal spikes or drops, and ensure your instance is handling network load properly.

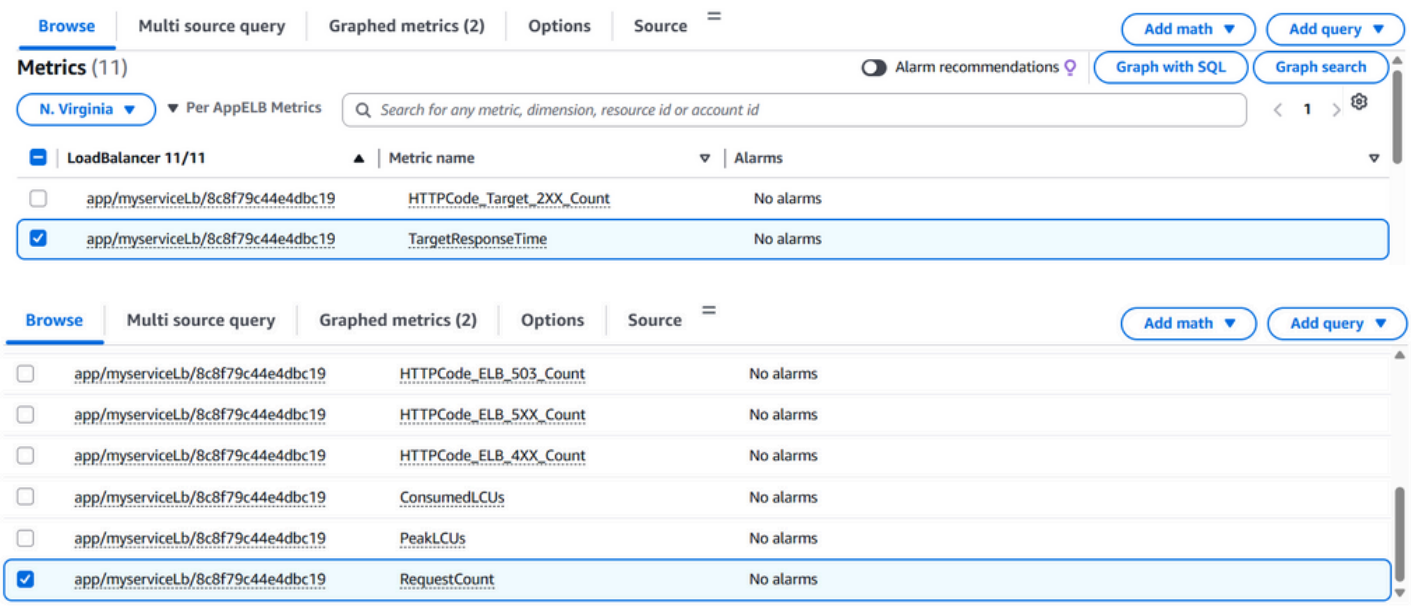


iiAdded 2nd widgets for RequestCount And TargetResponseTime:

RequestCount shows how many requests are hitting your Load Balancer, helping you track traffic volume.

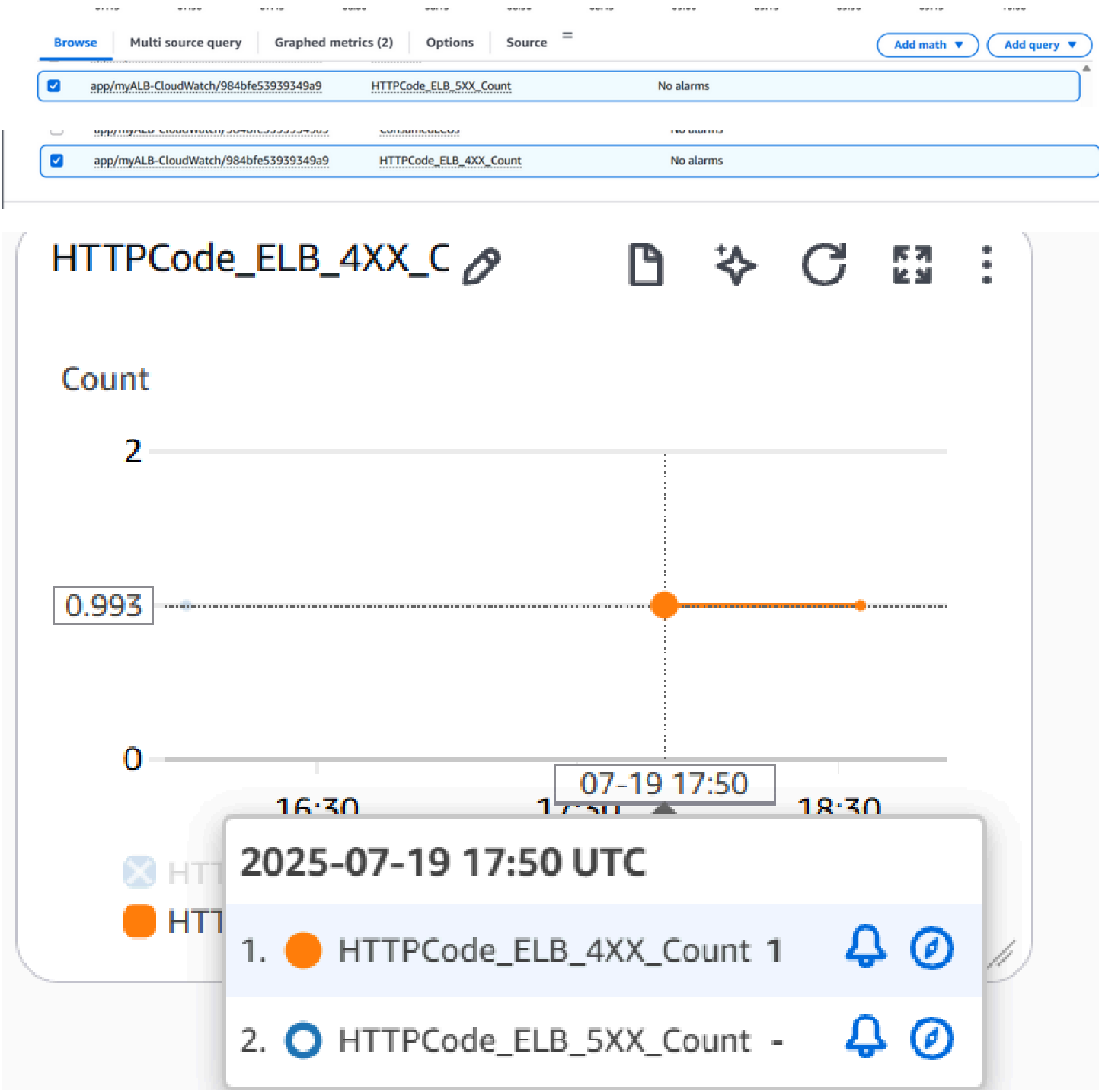
TargetResponseTime shows how long it takes for your backend (EC2) to respond, helping you monitor performance and detect slowdowns.

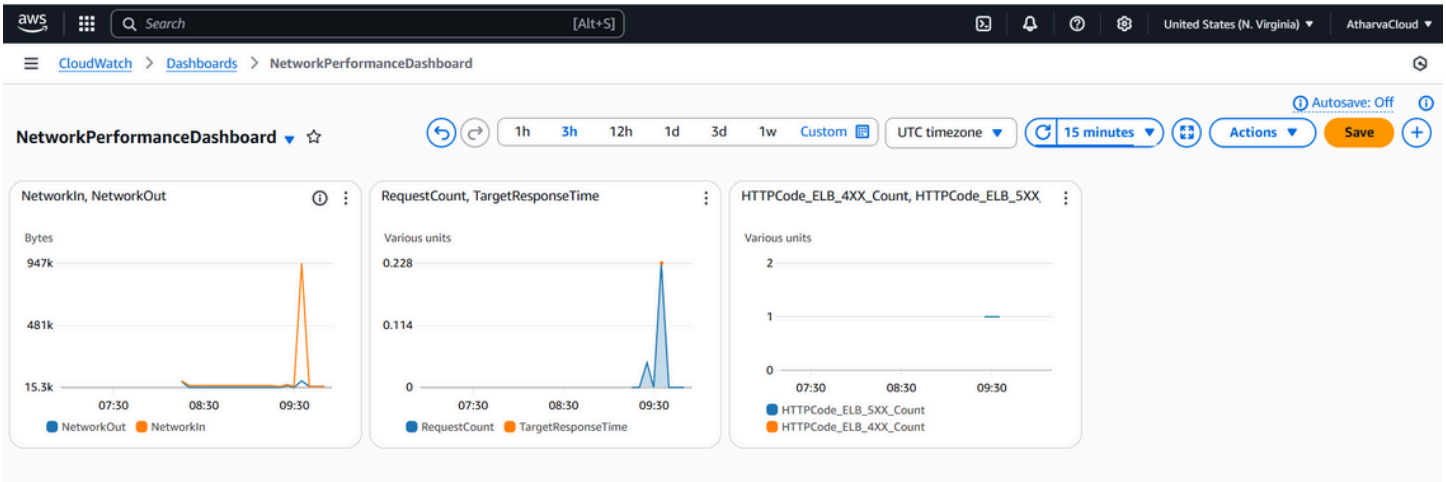




iii)Added 3rd Widgets for HTTPCode_Target_4XX_Count And HTTPCode_Target_5XX_Count:

These widgets show how many client-side errors (4XX) and server-side errors (5XX) are happening on your Application Load Balancer.
They help you quickly spot issues like bad requests from users or problems with your backend server (e.g., NGINX or app failures).





Dashboard 4: Security & Compliance Monitoring:

Purpose:
This dashboard helps you detect security threats and check compliance issues using logs from GuardDuty, AWS Config, IAM, and CloudTrail.
It shows alerts like unauthorized access, misconfigurations, failed logins, and API misuse — all visualized using custom log groups and queries, even in a free-tier setup.

Steps:

1)Enable GaurdDuty:
Service → Get Started → Enable

GuardDuty is a security service that automatically detects threats in your AWS account. Enabling it helps you monitor for suspicious activities like unauthorized access, malware, or unusual API calls — keeping your environment safe.

Security, Identity, & Compliance

Amazon GuardDuty

Intelligent threat protection for accounts and workloads

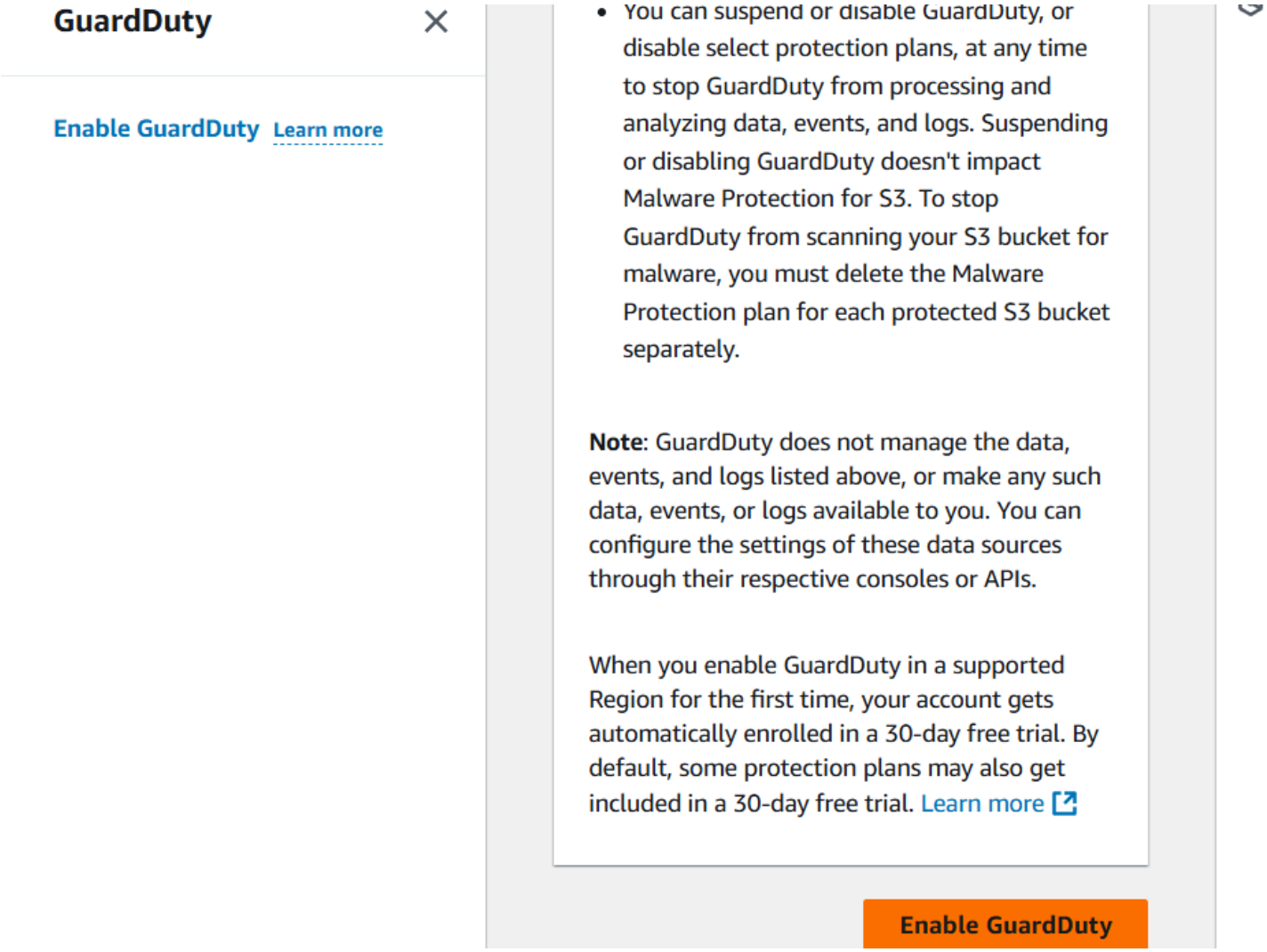
Single-step threat detection

Designed to reduce security risk by using continuous intelligent threat detection capabilities for your AWS accounts, containers, workloads, and data.

Try threat detection with GuardDuty

- (●) Amazon GuardDuty - all features
  Experience threat detection capabilities in your AWS environment.
- ( ) GuardDuty Malware Protection for S3 only
  Detect malicious files that are newly uploaded to your Amazon S3 buckets. You don't need to enable Amazon GuardDuty.

Get started



2)Enable AWS Config:
i)Create S3 bucket
Add rules (e.g., cloudtrail-enabled, restricted-ssh, etc.)

Enabling AWS Config allows you to track changes and evaluate the security and compliance of your AWS resources over time.

The S3 bucket stores detailed configuration snapshots and rule evaluations.

The rules (like cloudtrail-enabled, restricted-ssh) help automatically detect non-compliant or insecure setups, so you can fix issues early.

AWS Config

Record and evaluate configurations of your AWS resources

AWS Config provides a detailed view of the resources associated with your AWS account, including how they are configured, how they are related to one another, and how the configurations and their relationships have changed over time.

Set up AWS Config

A summarized view of AWS and non-AWS resources and the compliance status of the rules and the resources in each AWS Region.

Get started | 1-click setup

Delivery channel

Amazon S3 bucket

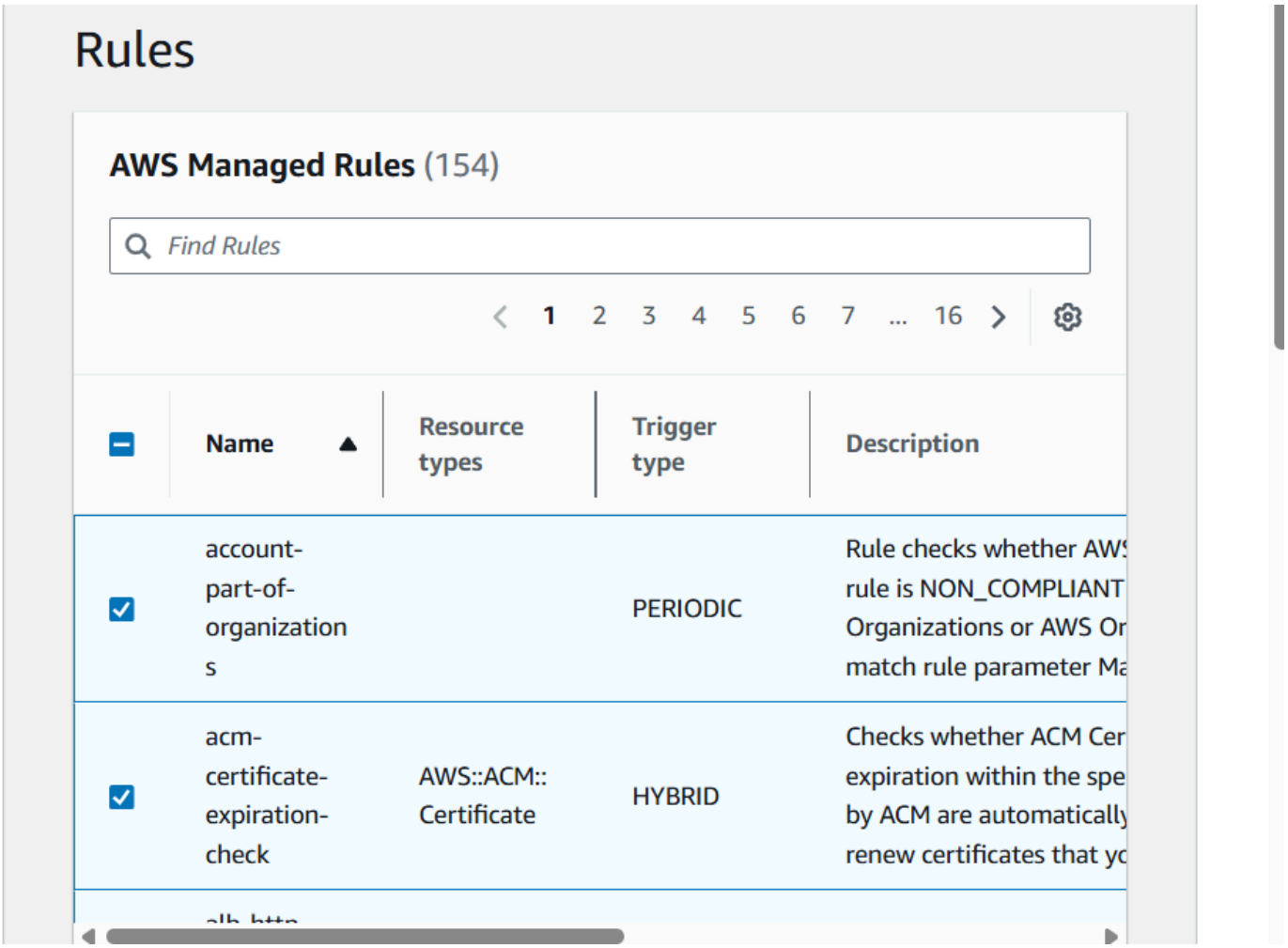
- (●) Create a bucket
- ( ) Choose a bucket from your account
- ( ) Choose a bucket from another account

Ensure appropriate permissions are available in this S3 bucket's policy. Learn more .

S3 Bucket name (required)

config-bucket-2004atharva

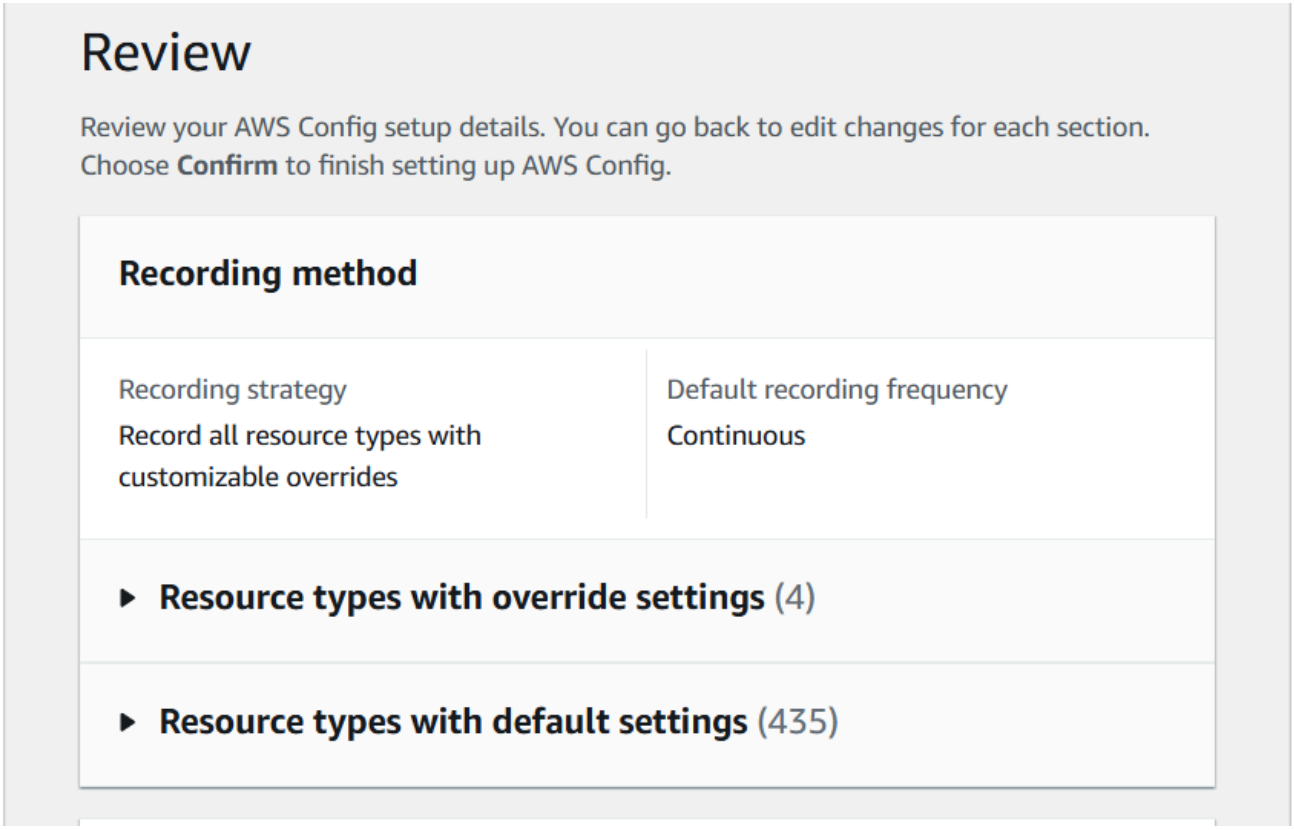
Prefix (optional)



ii)Enabled recording in Aws config:

Enabling recording in AWS Config means it starts tracking and saving the configuration details of your AWS resources (like EC2, S3, IAM, etc.).

It records every time a resource is created, changed, or deleted, so you can see who changed what and when, which is useful for audits, troubleshooting, and compliance checks.



3)Enable CloudTrail for API Monitoring:

i)Enabling CloudTrail lets you record all API calls and actions made in your AWS account (like starting EC2, changing IAM roles, etc.).

It helps you monitor user activity, detect suspicious behavior, and maintain security and compliance by keeping a full history of events.

CloudTrail > Dashboard > Create trail

Step 1
Choose trail attributes

Step 2
Choose log events

Step 3
Review and create

Choose trail attributes

General details

A trail created in the console is a multi-region trail. Learn more

Trail name

Enter a display name for your trail.

management-events

3-128 characters. Only letters, numbers, periods, underscores, and dashes are allowed.

Enable for all accounts in my organization

To review accounts in your organization, open AWS Organizations. See all accounts

Storage location | Info

Create new S3 bucket
Create a bucket to store logs for the trail.

CloudTrail > Dashboard > Create trail

Trail log bucket and folder

Enter a new S3 bucket name and folder (prefix) to store your logs. Bucket names must be globally unique.

config-bucket-2004atharva

Logs will be stored in config-bucket-2004atharva/AWSLogs/440744244333

Log file SSE-KMS encryption | Info

Enabled

Customer managed AWS KMS key

New

Existing

AWS KMS alias

Enter KMS alias

KMS key and S3 bucket must be in the same region.

CloudTrail > Dashboard > Create trail

Create a new SNS topic

- (●) New
- ( ) Existing

SNS topic

aws-cloudtrail-logs-440744244333-9628471c

CloudWatch Logs - *optional*

Configure CloudWatch Logs to monitor your trail logs and notify you when specific activity occurs. Standard CloudWatch and CloudWatch Logs charges apply. Learn more

CloudWatch Logs Info

- [x] Enabled

Log group Info

- (●) New
- ( ) Existing

Log group name

CloudTrail > Dashboard > Create trail

forward slashes, and periods are allowed.

IAM Role Info

AWS CloudTrail assumes this role to send CloudTrail events to your CloudWatch Logs log group.

- (●) New
- ( ) Existing

Role name

CloudTrailRoleForCloudWatchLogs_{trail-name}

▶ **Policy document**

Tags - *optional* Info

You can add one or more tags to help you manage and organize your resources, including trails.

Key

project

Value - *optional*

SecurityDashboard

CloudTrail > Dashboard > Create trail

Step 1
Choose trail attributes

Step 2
Choose log events

Step 3
Review and create

Choose log events

Events Info

Record API activity for individual resources, or for all current and future resources in AWS account. Additional charges apply

Event type

Choose the type of events that you want to log.

- [x] Management events
  Capture management operations performed on your AWS resources.
- [x] Data events
  Log the resource operations performed on or within a resource.
- [ ] Insights events
  Identify unusual activity, errors, or user behavior in your account.
- [ ] Network activity events
  Network activity events provide information about resource operations performed on a resource within a virtual private cloud endpoint.

CloudTrail > Dashboard > Create trail

Management events Info

Management events show information about management operations performed on resources in your AWS account.

No additional charges apply to log management events on this trail because this is your first copy of management events.

API activity

Choose the activities you want to log.

- [x] Read
- [x] Write
- [ ] Exclude AWS KMS events
- [ ] Exclude Amazon RDS Data API events

CloudTrail > Dashboard > Create trail

over the data events captured by your trail.

Switch to basic event selectors

Data event: S3

Remove

Resource type

Choose the resource type for which you want to log data events.

S3

Log selector template

Log all events

Selector name - *optional*

s3-DataEvents

1,000 character limit

JSON view

CloudTrail > Dashboard > Create trail

Choose trail attributes

Step 2
Choose log events

Step 3
Review and create

Review and create

Step 1: Choose trail attributes

Edit

General details

Trail name
SecurityAuditTrail

Multi-region trail
Yes

Apply trail to my organization
Not enabled

Trail log location
config-bucket-2004atharva/AWSLogs/440744244333

Log file SSE-KMS encryption
Not enabled

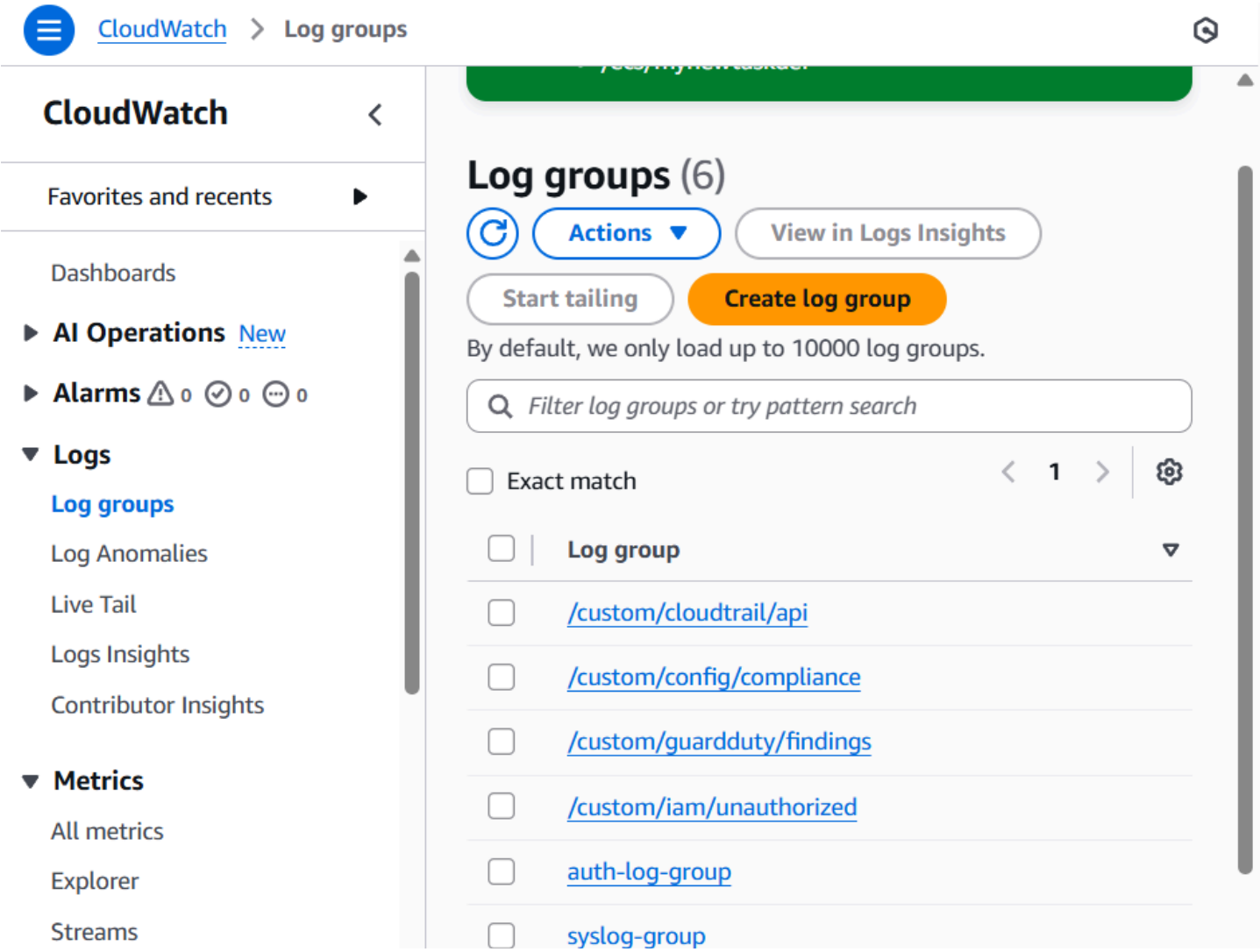
Log file validation
Enabled

SNS notification delivery
aws-cloudtrail-logs-440744244333-9628471c

ii)Manually Push Logs(simulated):
Example:
echo '{ "timestamp": "...", "type": "Trojan", "resource": "i-abc123" }' >> /var/log/custom-logs/guardduty.log
Repeat for config, IAM, and cloudtrail logs:

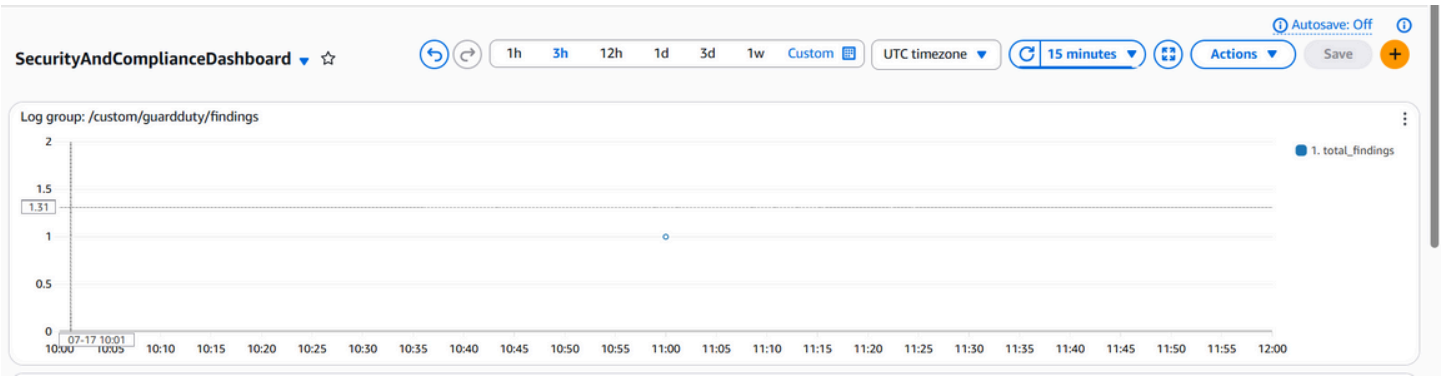
iii)Also Added the log groups path of these 4 new created log groups in the same config file:

iii)Final Log Agent Config Contains 6 Log Groups:



iv)fields @timestamp, type, resource, severity, status
| sort @timestamp desc
| limit 10

This query shows the latest 10 GuardDuty findings with details like time, type of threat, affected resource, severity, and status.
It helps you quickly identify and review recent security alerts in your AWS environment.

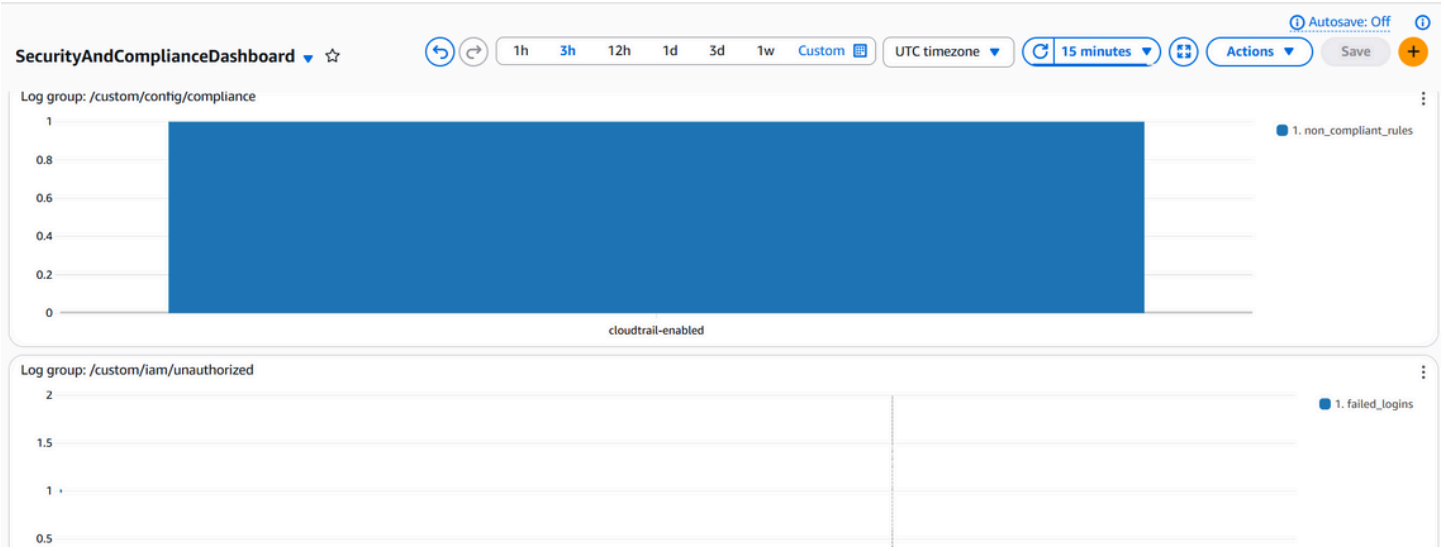


v)fields @timestamp, rule, compliance
| stats count(*) by compliance

This query displays how many AWS Config rule evaluations are compliant or non-compliant.
It shows the @timestamp, the rule name, and whether it passed or failed, then counts how many times each compliance status occurred.
Useful for quickly checking the overall compliance status of your environment.

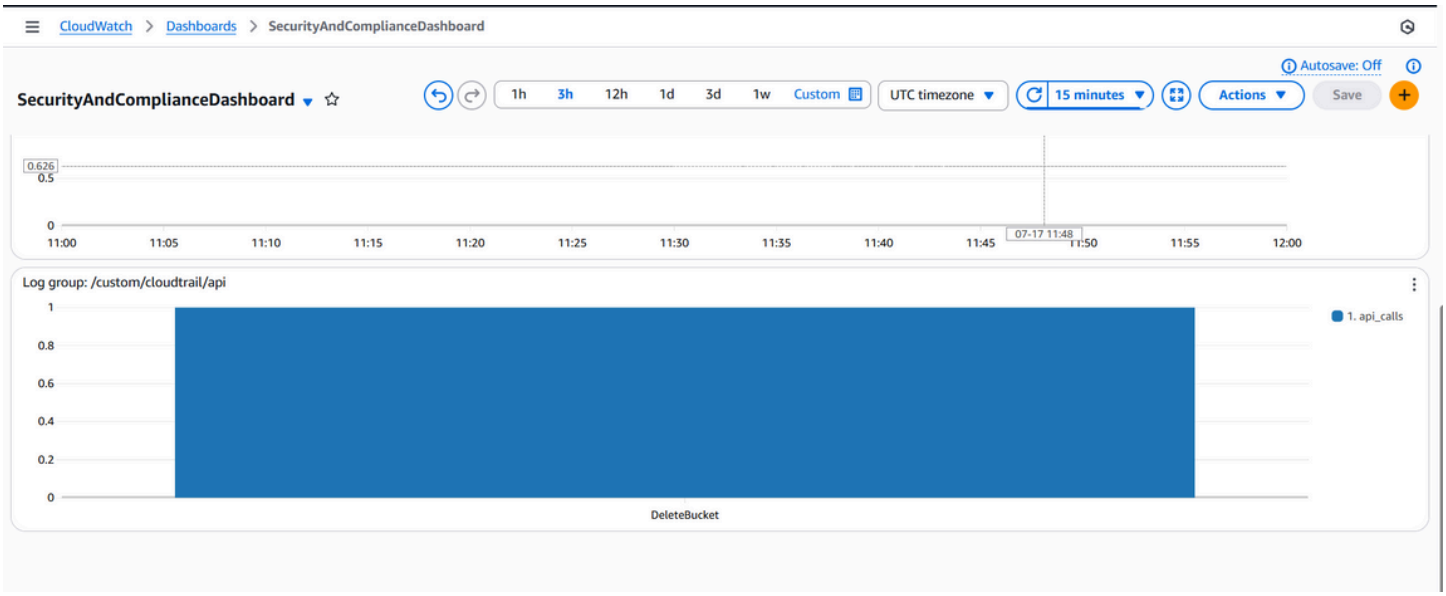
vi)fields @timestamp, user, action, status
| filter status="Failed"
| stats count(*) by user

This query filters IAM logs to show only failed actions (e.g., failed login attempts).
It displays the user, action, and status, then counts how many failures happened per user.
Helpful for identifying which users are experiencing or causing failed access attempts.



vii)fields @timestamp, api, user, result
| stats count(*) by api, result

This query analyzes CloudTrail logs to count how many times each API call was made and whether it succeeded or failed.
It groups the results by API name and result (like "Success" or "AccessDenied").
Useful for tracking frequently used APIs and spotting failed or suspicious operations.



Summary:

This project shows how to create four AWS CloudWatch dashboards to keep an eye on billing, logs, network traffic, and security. I used an EC2 instance with CloudWatch Agent to send system logs. I also created log groups and used simple queries to check failed logins, system messages, and API actions. For network monitoring, I set up a load balancer with NGINX to generate real traffic. I gave the EC2 instance the right IAM roles so it could send logs and be managed using SSM. The whole setup is low-cost and uses only free AWS tools — no Lambda or extra services were needed.