VPC Monitoring with Flow Logs & CloudWatch Logs

By Haroon Zaman | December 2025

Introduction

In this project, I explored how to monitor network traffic inside AWS using **VPC Flow Logs** and **Amazon CloudWatch Logs**. Network monitoring is essential for understanding how resources communicate within a VPC, how data moves in and out, and whether any security rules are blocking traffic.

Before enabling monitoring, I first set up the full networking environment:

- Created **two separate VPCs**, each with its own CIDR block
- Launched **one EC2 instance in each VPC**
- Created a **VPC Peering connection** so both VPCs could communicate privately
- Verified **successful connectivity** between both EC2 instances

Once the two VPCs were communicating, I added monitoring tools to track what kind of traffic was flowing between them.

Network monitoring helps engineers:

- Check the source and destination of traffic inside the VPC
- Analyse how much data is being transferred
- Identify traffic that is being blocked by security groups or NACLs
- Ensure resources are communicating securely
- Optimize network performance in advanced setups

In this project, I completed the following tasks:

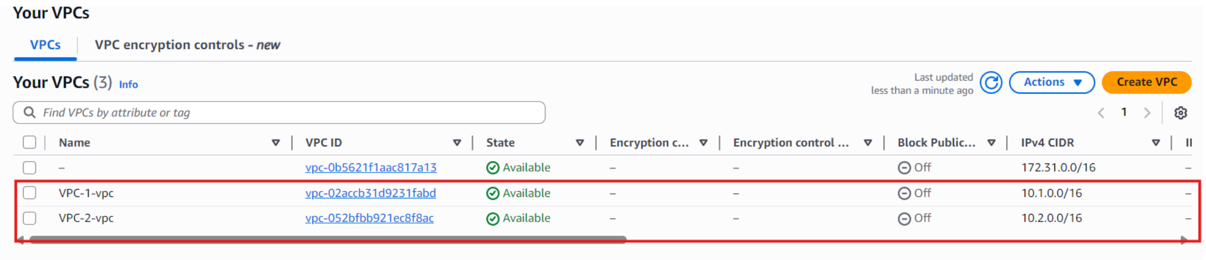
1. Set up **two VPCs** and validated their peering connection
2. Enabled **VPC Flow Logs** to collect detailed network traffic information
3. Sent logs to **Amazon CloudWatch Logs**
4. Created the necessary **IAM role and permissions policy**
5. Analysed flow log data using **CloudWatch Log Insights**

This setup allowed me to fully monitor the communication between my VPCs and understand how network traffic behaves at a low level inside AWS.

Creating Two VPCs for Monitoring

VPC 1 Setup

- I created the first VPC and named it **VPC_1**.
- I assigned it the IPv4 CIDR block **10.1.0.0/16**, giving it a large private IP range.
- I created **one public subnet** inside this VPC.



Your VPCs

VPCs
VPC encryption controls - *new*

Your VPCs (3) Info

Last updated less than a minute ago
Actions ▼
Create VPC
Find VPCs by attribute or tag
1

□	Name	VPC ID	State	Encryption c...	Encryption control ...	Block Public...	IPv4 CIDR
□	–	vpc-0b5621f1aac817a13	✓ Available	–	–	Off	172.31.0.0/16
□	VPC-1-vpc	vpc-02accb31d9231fabd	✓ Available	–	–	Off	10.1.0.0/16
□	VPC-2-vpc	vpc-052bfbb921ec8f8ac	✓ Available	–	–	Off	10.2.0.0/16

- I selected **No NAT Gateway**, since I wasn’t using private subnets that require outbound internet access.
- I also selected **No S3 Gateway Endpoint**, because this project focuses on VPC-to-VPC connectivity and monitoring, not private S3 access.

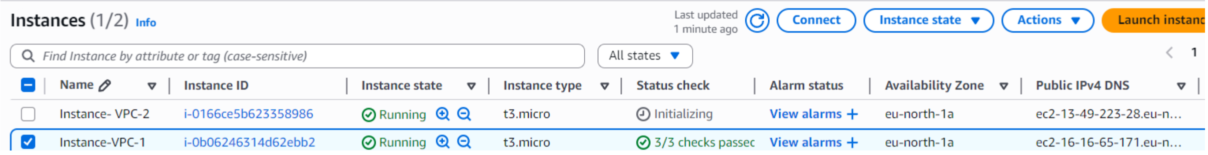
VPC 2 Setup

- Next, I created the second VPC named **VPC_2**.
- I assigned it a different IPv4 CIDR block: **10.2.0.0/16**.
  (Each VPC must have a unique CIDR range for peering and communication.)
- Just like the first VPC, I added **one public subnet**.
- Again, I chose **No NAT Gateway**, since the subnet didn't require private outbound traffic.
- I selected **No S3 Gateway Endpoint** for the same reasons as VPC_1.

Creating EC2 Instances in Both VPCs

Instance in VPC_1

- I created the first instance and named it **Instance_VPC_1**.
- I attached it to **VPC_1** and selected the **public subnet** of that VPC.
- I **enabled Auto-assign Public IPv4**, so this instance could be reached from the internet for testing and monitoring.
- In the **Security Group settings**, I created a new SG specifically for this instance.
  - I allowed **SSH (port 22)** so I could connect to it.
  - I allowed **ICMP – IPv4** so I could use ping for connectivity tests.
- After setting the rules, I launched the instance.



Instances (1/2) Info

Find Instance by attribute or tag (case-sensitive)

All states

☐	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
☐	Instance- VPC-2	i-0166ce5b623358986	Running	t3.micro	Initializing	View alarms +	eu-north-1a	ec2-13-49-223-28.eu-n...
☑	Instance-VPC-1	i-0b06246314d62ebb2	Running	t3.micro	3/3 checks passec	View alarms +	eu-north-1a	ec2-16-16-65-171.eu-n...

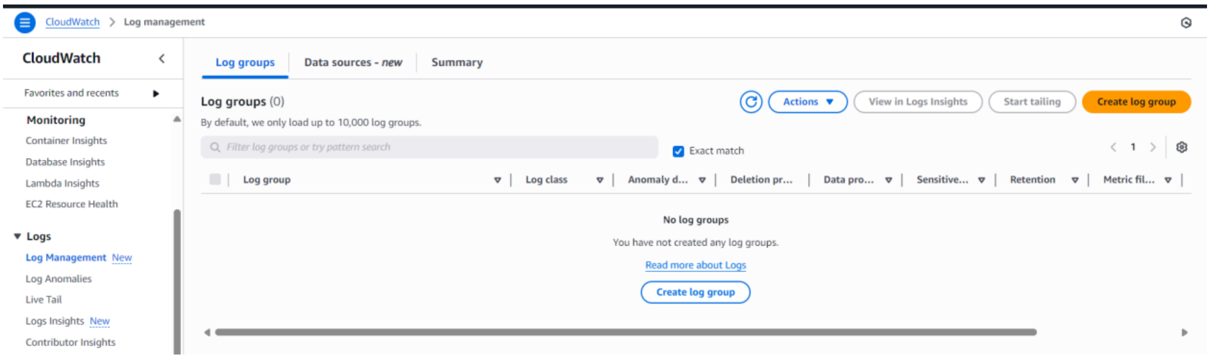
Instance in VPC_2

- I created the second instance and placed it inside **VPC_2**.
- I selected the **public subnet** of VPC_2 as well.
- This instance also uses private communication for peering tests, so the setup matched the first instance (except no need for a public IP if not connecting externally).
- I launched **Instance_VPC_2** with similar settings to keep the testing environment consistent.

Setting Up CloudWatch for VPC Monitoring

What is Amazon CloudWatch?

- **Amazon CloudWatch** is a powerful monitoring service that observes everything happening in your AWS environment in real time.
- AWS services send their **metrics and logs** to CloudWatch, where you can view them as clear dashboards, graphs, and insights.
- CloudWatch helps track performance, detect unusual traffic, and understand the overall health of your AWS environment.
- It can even **automate actions**, such as blocking illegitimate traffic or alerting you when something suspicious happens.



Log groups (0)

By default, we only load up to 10,000 log groups.

No log groups

You have not created any log groups.

Creating a Log Group in CloudWatch

- I searched for **CloudWatch** in the AWS Console and opened the service.
- On the left panel, under **Logs**, I selected **Log groups**.
- I created a new log group, which is required to store all VPC Flow Log data.
- This log group will act as the **destination** where flow logs from both VPCs will be sent for monitoring and analysis.

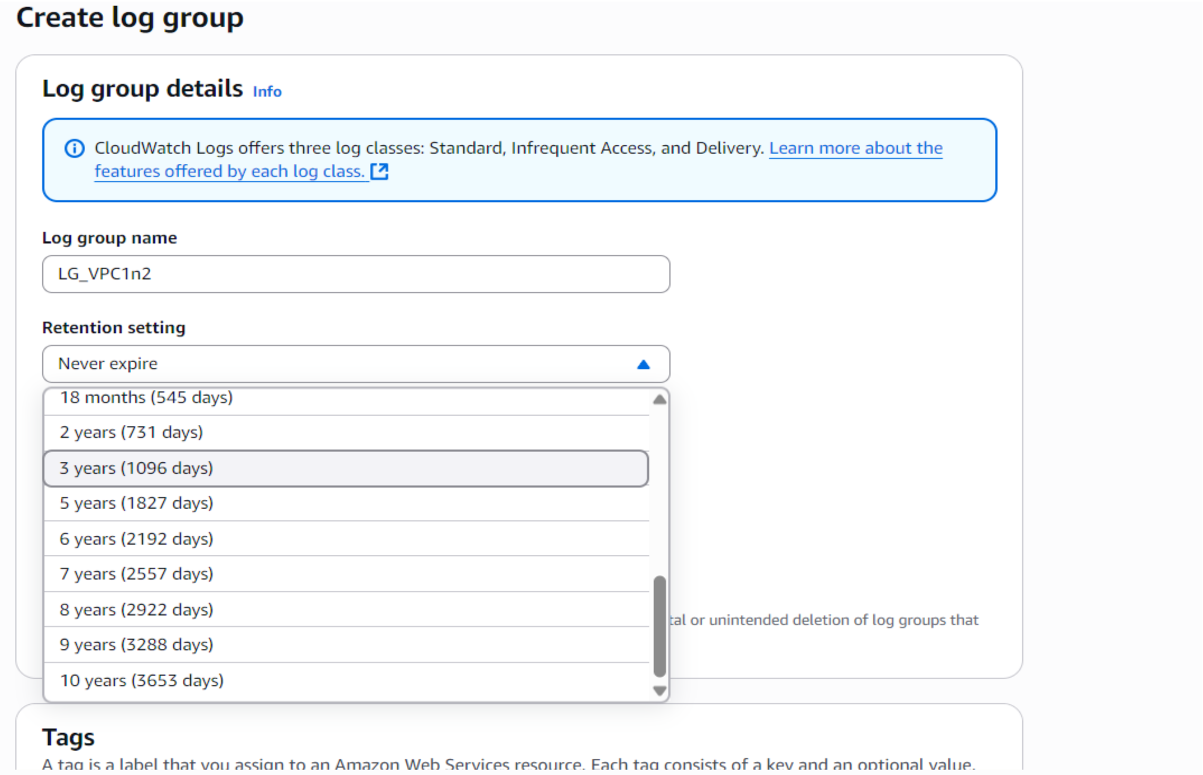
Creating the Log Group in CloudWatch

What is a Log Group?

- A **Log Group** is a container inside CloudWatch where log data is stored and organized.
- All logs from a resource (like VPC Flow Logs, Lambda logs, EC2 logs, etc.) are sent into a log group.
- Think of it as a folder that stores and manages all related log streams for easier monitoring and analysis.

Cofiguring the Log Group

- I named my log group **LG_VPC1n2**, since it will store flow logs from both VPCs.
- The retention period defines **how long the logs will be stored** before CloudWatch automatically deletes them.
- Options range from 1 day up to 10 years.
- I selected **Never Expire**, which keeps all logs permanently unless deleted manually.
- This is useful for long-term troubleshooting or security analysis.
- CloudWatch offers two log classes:
  Standard – Designed for logs that will be accessed or analyzed regularly.
  Infrequent Access – Lower storage cost but higher cost per retrieval; suited for long-term archiving.



Create log group

Log group details Info

CloudWatch Logs offers three log classes: Standard, Infrequent Access, and Delivery. Learn more about the features offered by each log class.

Log group name

LG_VPC1n2

Retention setting

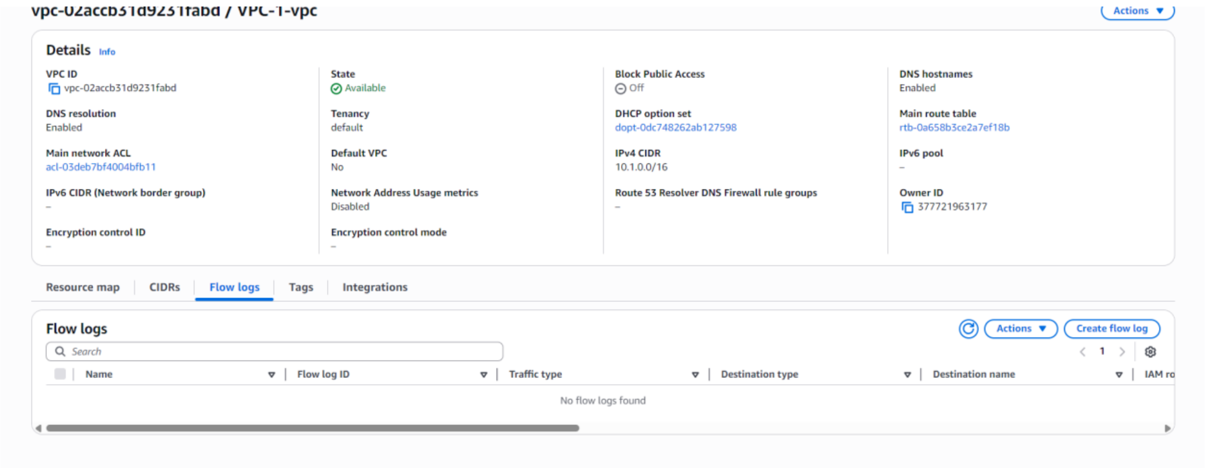
Never expire

Tags

A tag is a label that you assign to an Amazon Web Services resource. Each tag consists of a key and an optional value.

- I chose **Standard**, since I will be actively analyzing the VPC flow logs.
- A **KMS Key ARN** is an encryption key used to encrypt logs at rest.

- Since encrypting flow logs wasn't required for this project, I left this setting empty.
- It remains optional unless strict compliance or encryption policies are needed.
- CloudWatch offers an option to protect the log group from accidental deletion.
- I left this **unchecked**, as I may modify or recreate the log group during practice.
- After reviewing all settings, I clicked **Create Log Group**, and **LG_VPC1n2** was successfully created.



vpc-02accb31d9231fabd / VPC-1-vpc

Details

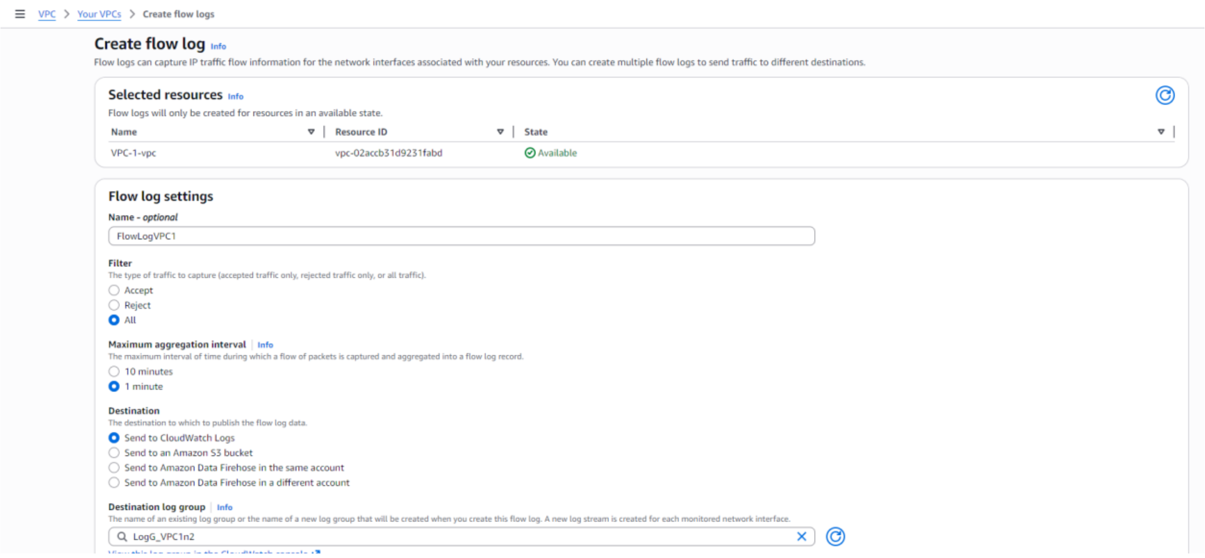
VPC ID vpc-02accb31d9231fabd	State Available	Block Public Access Off	DNS hostnames Enabled
DNS resolution Enabled	Tenancy default	DHCP option set dopt-0dc748262ab127598	Main route table rtb-0a658b3ce2a7ef18b
Main network ACL acl-03deb7bf4004bfb11	Default VPC No	IPv4 CIDR 10.1.0.0/16	IPv6 pool –
IPv6 CIDR (Network border group) –	Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups –	Owner ID 377721963177
Encryption control ID –	Encryption control mode –		

Flow logs

Name	Flow log ID	Traffic type	Destination type	Destination name	IAM ro
No flow logs found					

Creating VPC Flow Logs for VPC_1

- After creating the Log Group, I went back to the **VPC Console** and selected **VPC_1**.
- On the left-side menu, I opened **Flow Logs** and clicked **Create Flow Log**.



Create flow log

Flow logs can capture IP traffic flow information for the network interfaces associated with your resources. You can create multiple flow logs to send traffic to different destinations.

Selected resources

Flow logs will only be created for resources in an available state.

Name	Resource ID	State
VPC-1-vpc	vpc-02accb31d9231fabd	Available

Flow log settings

Name - *optional*: FlowLogVPC1

Filter
The type of traffic to capture (accepted traffic only, rejected traffic only, or all traffic).
○ Accept
○ Reject
● All

Maximum aggregation interval
The maximum interval of time during which a flow of packets is captured and aggregated into a flow log record.
○ 10 minutes
● 1 minute

Destination
The destination to which to publish the flow log data.
● Send to CloudWatch Logs
○ Send to an Amazon S3 bucket
○ Send to Amazon Data Firehose in the same account
○ Send to Amazon Data Firehose in a different account

Destination log group
The name of an existing log group or the name of a new log group that will be created when you create this flow log. A new log stream is created for each monitored network interface.
LogG_VPC1n2

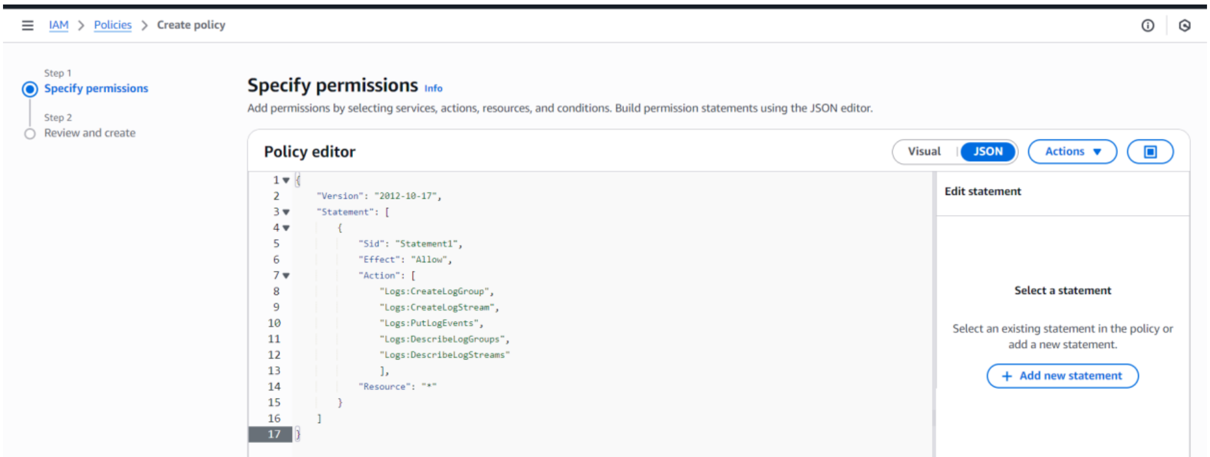
Configuring the Flow Log for VPC_1

- I named the flow log **FlowLogVPC1** to clearly identify which VPC it belongs to.
- For the **Filter**, I selected **All**.
- This captures **accepted**, **rejected**, and **all network traffic** going in and out of the VPC.

- Choosing “All” provides complete visibility, which is ideal for learning and troubleshooting.
- I selected the **1-minute** interval.
- **What is Maximum Aggregation Interval?**
  - It defines how often AWS delivers flow log records to the destination (CloudWatch).
  - Shorter intervals (like 1 min) give **faster visibility** into traffic.
  - Longer intervals (10 min) reduce cost but delay log delivery.
- For monitoring and testing, **1 minute** is the best choice.
- For the destination, I selected **CloudWatch Logs**.
- Then I chose the log group I created earlier: **LG_VPC1n2**.
- This ensures all captured VPC traffic is stored and viewable inside CloudWatch.

Service Role Requirement

- Flow Logs require an **IAM Service Role** with permission to deliver logs into CloudWatch.
- **Why is a service role needed?**
  - Because VPC Flow Logs must **write log data** to CloudWatch Logs.
  - The VPC service itself doesn’t have permission by default.
  - The service role grants AWS permission to **push logs** into your CloudWatch log group on your behalf.
- Since I didn’t have a service role yet, I had to **create one before completing the flow log setup**.



Specify permissions

Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.

Policy editor

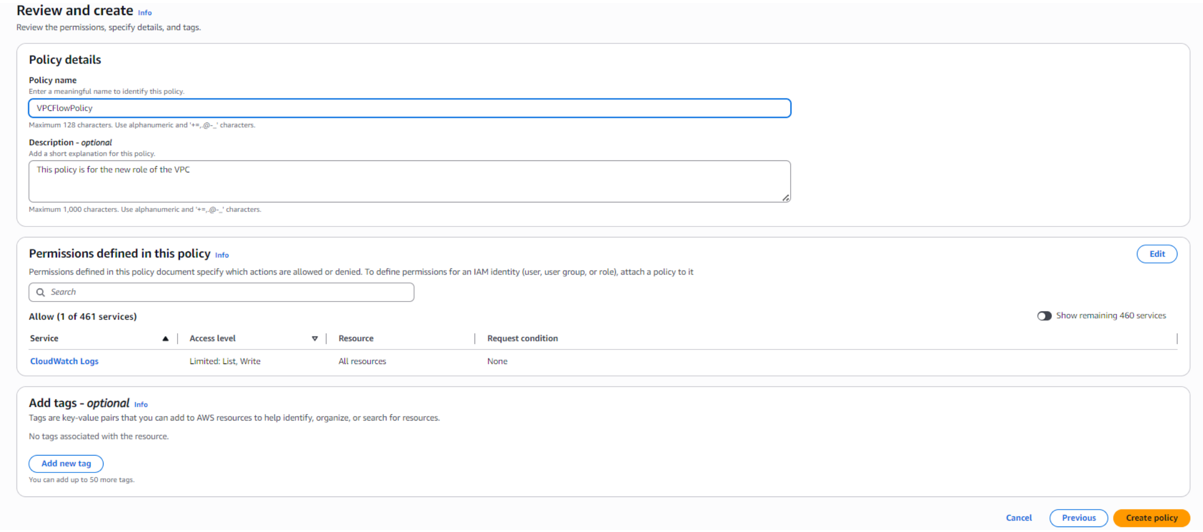
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Action": [
                "Logs:CreateLogGroup",
                "Logs:CreateLogStream",
                "Logs:PutLogEvents",
                "Logs:DescribeLogGroups",
                "Logs:DescribeLogStreams"
                ],
            "Resource": "*"
        }
    ]
}
```

Select an existing statement in the policy or add a new statement.

Creating the IAM Policy for Flow Logs

- I needed a **service role**, so I searched for **IAM** in the console.
- I first tried to create the role, but AWS required a **policy** to exist before the role could be created.
  So I went to **IAM → Policies → Create policy**.
- I switched to the **JSON editor** because I wanted to create a **custom policy**.
  - A **custom policy** lets me define EXACTLY what permissions are allowed.
  - It works like a VIP list — only the services we choose can use this permission.

- In the JSON editor, I added the actions required for VPC Flow Logs to write logs to CloudWatch:
    - `logs:CreateLogGroup` – allows creating new log groups
    - `logs:CreateLogStream` – allows creating streams inside log groups
    - `logs:PutLogEvents` – allows sending log data into CloudWatch
    - `logs:DescribeLogGroups` – lets the role view log groups
    - `logs:DescribeLogStreams` – lets it view log streams
- For **Resource**, I used `"*"`, meaning the policy applies to all log groups and streams.



Review and create Info

Review the permissions, specify details, and tags.

Policy details

Policy name

Enter a meaningful name to identify this policy.

VPCFlowPolicy

Maximum 128 characters. Use alphanumeric and '+=,.@-_' characters.

Description - *optional*

Add a short explanation for this policy.

This policy is for the new role of the VPC

Maximum 1,000 characters. Use alphanumeric and '+=,.@-_' characters.

Permissions defined in this policy Info

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

Allow (1 of 461 services)

Show remaining 460 services

Service	Access level	Resource	Request condition
CloudWatch Logs	Limited: List, Write	All resources	None

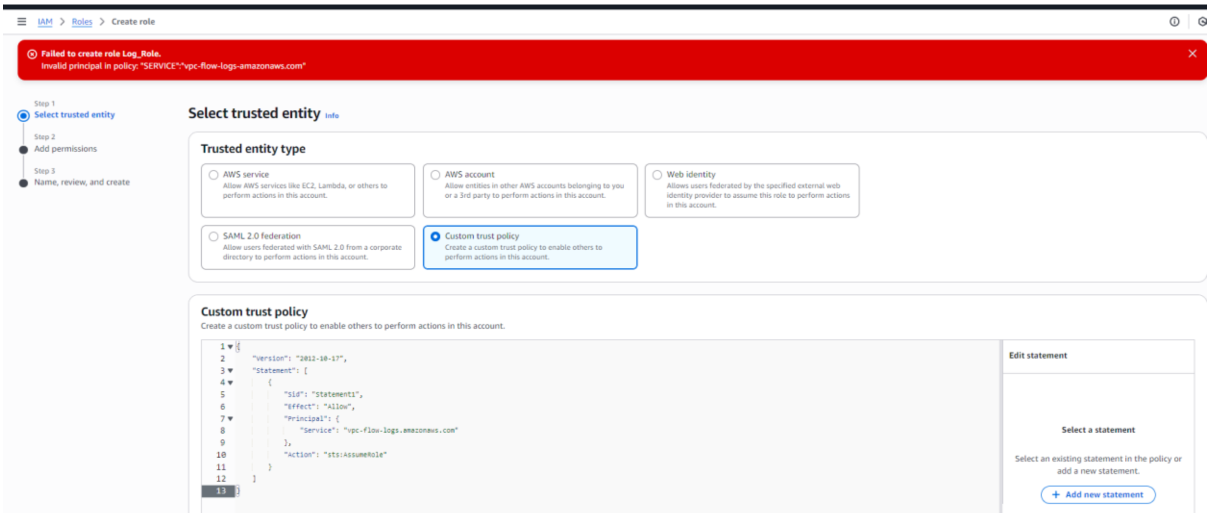
Add tags - *optional* Info

Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

You can add up to 50 more tags.

- After clicking **Next**, I reviewed the policy details to make sure everything was correct.
- I gave the policy the name **VPCFlowPolicy** so it clearly represents its purpose.
- Then I clicked **Create policy**, and the custom policy was successfully created.



IAM > Roles > Create role

Failed to create role Log_Role.
Invalid principal in policy: "SERVICE":"vpc-flow-logs-amazonaws.com"

Step 1 Select trusted entity

Step 2 Add permissions

Step 3 Name, review, and create

Select trusted entity Info

Trusted entity type

○ AWS service — Allow AWS services like EC2, Lambda, or others to perform actions in this account.

○ AWS account — Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

○ Web identity — Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.

○ SAML 2.0 federation — Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.

● Custom trust policy — Create a custom trust policy to enable others to perform actions in this account.

Custom trust policy

Create a custom trust policy to enable others to perform actions in this account.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Principal": {
                "Service": "vpc-flow-logs.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Edit statement

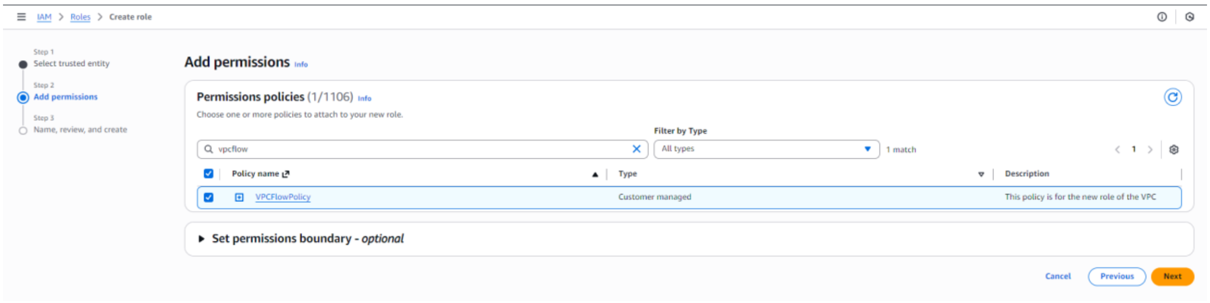
Select a statement

Select an existing statement in the policy or add a new statement.

Creating the IAM Role for Flow Logs

- After creating the policy, I went to **IAM** → **Roles** and clicked **Create role**.

- For the trusted entity type, I selected **Custom trust policy**.
  This option lets me specify exactly **which AWS service** is allowed to assume (use) this role.
- In the trust policy, I had to add the **Principal**, which identifies the AWS service that will use this role.
  I added:
  `"Service": "vpc-flow-logs.amazonaws.com"`
- **What is this service?**
  - This is the AWS service responsible for generating VPC Flow Logs.
  - By adding it as the principal, I am telling AWS that **only VPC Flow Logs** are allowed to assume this role and send logs to CloudWatch.
  - No other service or user can use this role.
- The red error I saw at first was simply a typo because earlier I wrote:
  `vpc-flow-logs"-"amazonaws.com`
  After correcting it to the proper format, the trust policy validated successfully.



IAM > Roles > Create role

Step 1
Select trusted entity
Step 2
Add permissions
Step 3
Name, review, and create

Add permissions Info

Permissions policies (1/1106) Info
Choose one or more policies to attach to your new role.

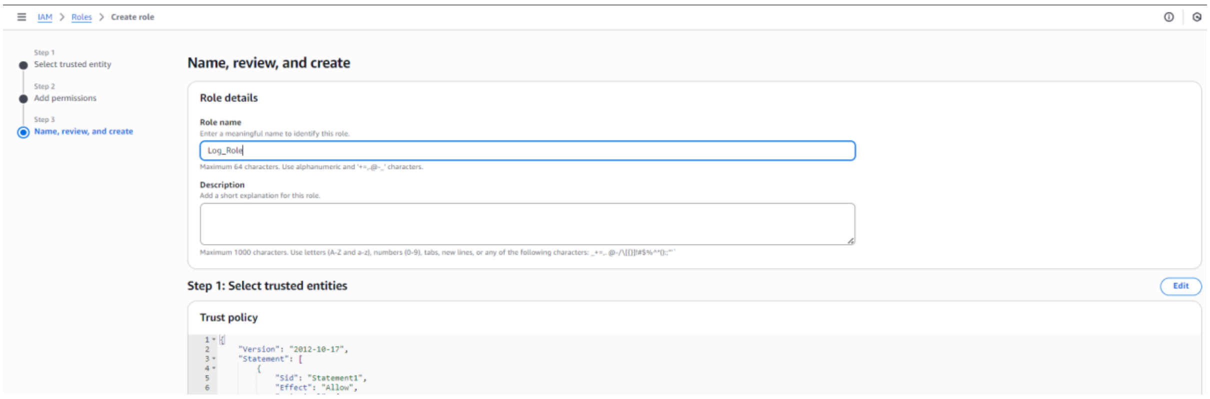
Filter by Type

vpcflow | All types | 1 match

☑	Policy name	Type	Description
☑	VPCFlowPolicy	Customer managed	This policy is for the new role of the VPC

▸ Set permissions boundary - *optional*

- After fixing the trust policy, I clicked **Next** to continue the role setup.
- On the **Add permissions** page, I selected the custom policy I created earlier: **VPCFlowPolicy**.
- This attaches the CloudWatch Logs permissions to the role so VPC Flow Logs can create log groups and push log data.
- After selecting the policy, I clicked **Next** to proceed.



IAM > Roles > Create role

Step 1
Select trusted entity
Step 2
Add permissions
Step 3
Name, review, and create

Name, review, and create

Role details

Role name
Enter a meaningful name to identify this role.
Log_Role
Maximum 64 characters. Use alphanumeric and '+=,.@-_' characters.

Description
Add a short explanation for this role.

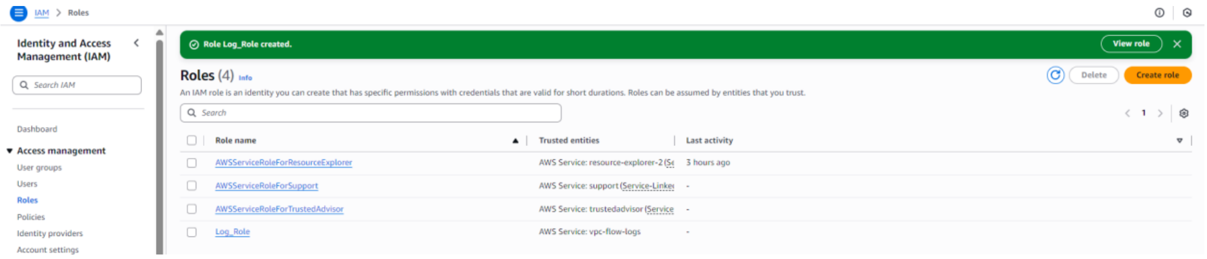
Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: _+=,.@-/\[{}]!#$%^*();:"'`

Step 1: Select trusted entities

Trust policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
```

- On the final review page, I gave the role the name **Log_Role** so it’s clear that this role is used for VPC Flow Logs.
- After confirming everything looked correct, I clicked **Create role**.



IAM > Roles

Identity and Access Management (IAM)

Search IAM

Dashboard

Access management
- User groups
- Users
- Roles
- Policies
- Identity providers
- Account settings

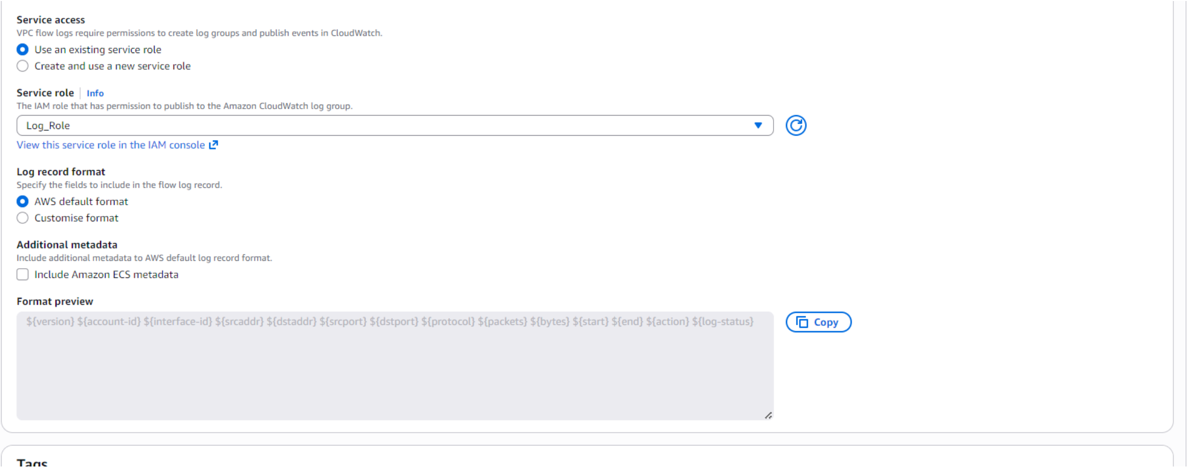
Role Log_Role created.

Roles (4) Info

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

Role name	Trusted entities	Last activity
AWSServiceRoleForResourceExplorer	AWS Service: resource-explorer-2 (Se	3 hours ago
AWSServiceRoleForSupport	AWS Service: support (Service-Linke	-
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service	-
Log_Role	AWS Service: vpc-flow-logs	-

- The role was successfully created and is now ready to be used by the VPC Flow Logs service.



Service access
VPC flow logs require permissions to create log groups and publish events in CloudWatch.
● Use an existing service role
○ Create and use a new service role

Service role | Info
The IAM role that has permission to publish to the Amazon CloudWatch log group.
Log_Role
View this service role in the IAM console

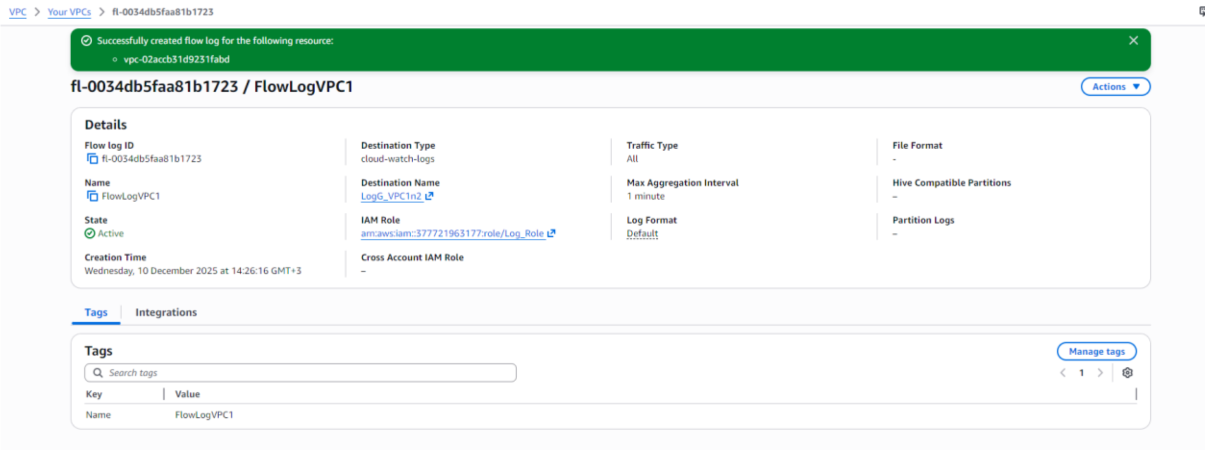
Log record format
Specify the fields to include in the flow log record.
● AWS default format
○ Customise format

Additional metadata
Include additional metadata to AWS default log record format.
□ Include Amazon ECS metadata

Format preview

```
${version} ${account-id} ${interface-id} ${srcaddr} ${dstaddr} ${srcport} ${dstport} ${protocol} ${packets} ${bytes} ${start} ${end} ${action} ${log-status}
```

- After creating the IAM role, I went back to the **Flow Log setup** page for **VPC_1**.
- In the **Service Role** field, I selected the new role I created: **Log_Role**.
  This gives VPC Flow Logs permission to write traffic logs into CloudWatch.
- There was also an option to choose a **log format**.
  AWS offers custom formats where you can include additional fields like packet source, destination port, flow direction, etc.
- For this project, I kept the **default log format**, which already includes the essential details needed for monitoring.



VPC > Your VPCs > fl-0034db5faa81b1723

Successfully created flow log for the following resource:
- vpc-02accb31d9231fabd

fl-0034db5faa81b1723 / FlowLogVPC1

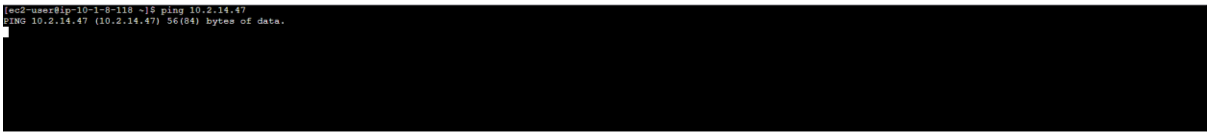
Details

Flow log ID: fl-0034db5faa81b1723
Name: FlowLogVPC1
State: Active
Creation Time: Wednesday, 10 December 2025 at 14:26:16 GMT+3
Destination Type: cloud-watch-logs
Destination Name: LogG_VPC1n2
IAM Role: arn:aws:iam::377721963177:role/Log_Role
Cross Account IAM Role: –
Traffic Type: All
Max Aggregation Interval: 1 minute
Log Format: Default
File Format: -
Hive Compatible Partitions: –
Partition Logs: –

Tags

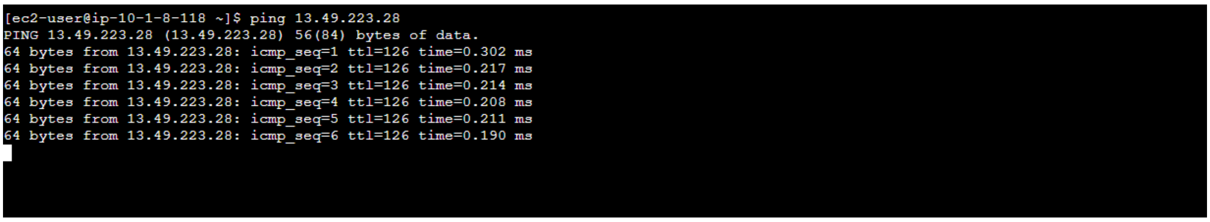
Key	Value
Name	FlowLogVPC1

- After reviewing all settings, I clicked **Create Flow Log**, and the flow log for **VPC_1** was successfully created.



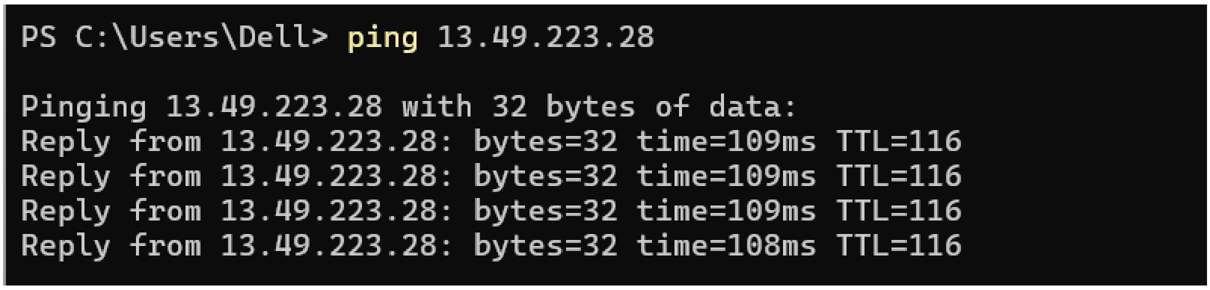
```
[ec2-user@ip-10-1-8-118 ~]$ ping 10.2.14.47
PING 10.2.14.47 (10.2.14.47) 56(84) bytes of data.
```

- Then I connected to **Instance_VPC_1**.
- I started pinging the **private IP** of **Instance_VPC_2**.
- Only **one ping was sent**, and that was it — no more replies came back.



```
[ec2-user@ip-10-1-8-118 ~]$ ping 13.49.223.28
PING 13.49.223.28 (13.49.223.28) 56(84) bytes of data.
64 bytes from 13.49.223.28: icmp_seq=1 ttl=126 time=0.302 ms
64 bytes from 13.49.223.28: icmp_seq=2 ttl=126 time=0.217 ms
64 bytes from 13.49.223.28: icmp_seq=3 ttl=126 time=0.214 ms
64 bytes from 13.49.223.28: icmp_seq=4 ttl=126 time=0.208 ms
64 bytes from 13.49.223.28: icmp_seq=5 ttl=126 time=0.211 ms
64 bytes from 13.49.223.28: icmp_seq=6 ttl=126 time=0.190 ms
```

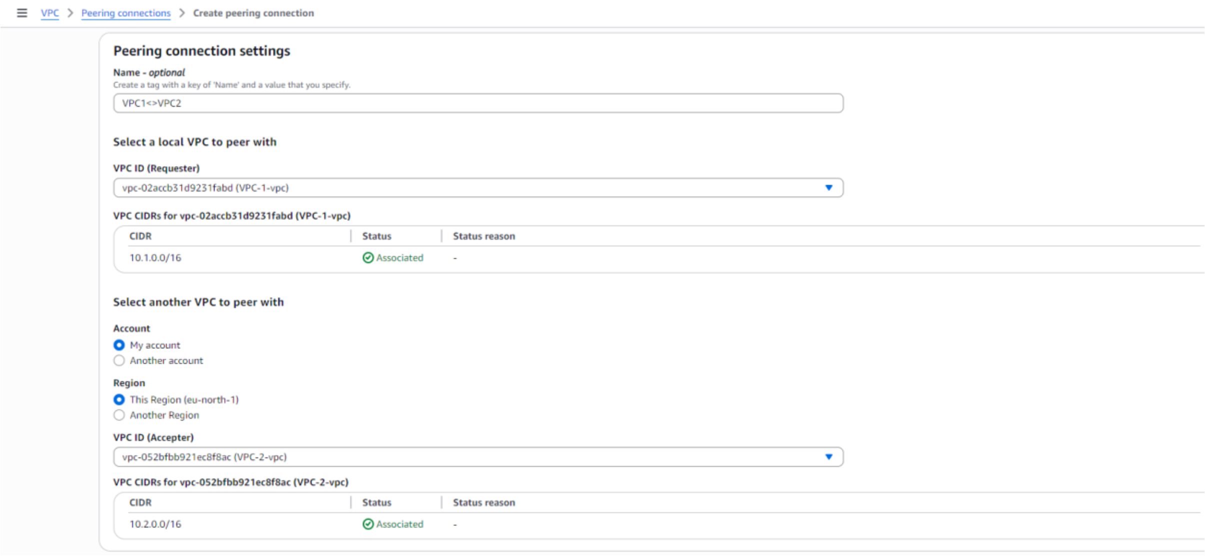
- Although the **public IP** of that instance was reachable from **Instance_VPC_1**.



```
PS C:\Users\Dell> ping 13.49.223.28

Pinging 13.49.223.28 with 32 bytes of data:
Reply from 13.49.223.28: bytes=32 time=109ms TTL=116
Reply from 13.49.223.28: bytes=32 time=109ms TTL=116
Reply from 13.49.223.28: bytes=32 time=109ms TTL=116
Reply from 13.49.223.28: bytes=32 time=108ms TTL=116
```

- I also tried from my own PC, and the **public IP** of that instance was responding.
- It did not respond to the **private IP** from **Instance_VPC_1** because there was **no peering connection** created between the VPCs.



Peering connection settings

Name - *optional*
Create a tag with a key of 'Name' and a value that you specify.
VPC1<>VPC2

Select a local VPC to peer with

VPC ID (Requester)
vpc-02accb31d9231fabd (VPC-1-vpc)

VPC CIDRs for vpc-02accb31d9231fabd (VPC-1-vpc)

CIDR	Status	Status reason
10.1.0.0/16	✓ Associated	-

Select another VPC to peer with

Account
● My account
○ Another account

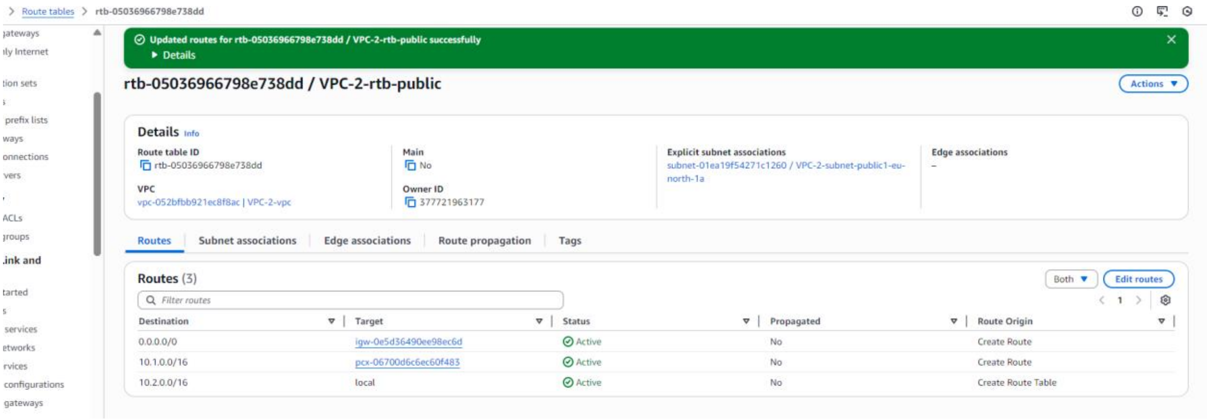
Region
● This Region (eu-north-1)
○ Another Region

VPC ID (Accepter)
vpc-052bfbb921ec8f8ac (VPC-2-vpc)

VPC CIDRs for vpc-052bfbb921ec8f8ac (VPC-2-vpc)

CIDR	Status	Status reason
10.2.0.0/16	✓ Associated	-

- So I went to **Peering Connections** and created a new one named **VPC1 <> VPC2**.
- I chose **VPC_1** as the requestor and **VPC_2** as the accepter, since both are in my account and in the same region.



Updated routes for rtb-05036966798e738dd / VPC-2-rtb-public successfully

rtb-05036966798e738dd / VPC-2-rtb-public

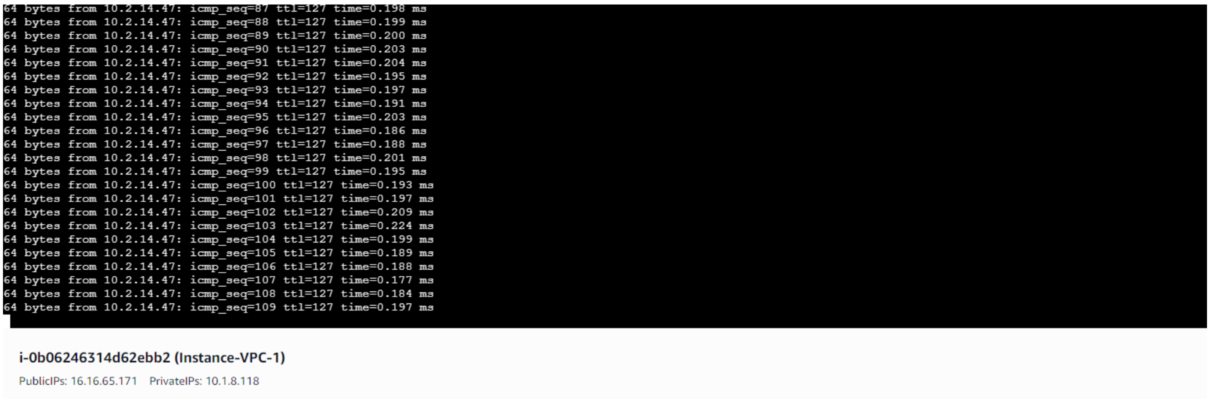
Details

Route table ID: rtb-05036966798e738dd
Main: No
Explicit subnet associations: subnet-01ea19f54271c1260 / VPC-2-subnet-public1-eu-north-1a
Edge associations: –
VPC: vpc-052bfbb921ec8f8ac | VPC-2-vpc
Owner ID: 377721963177

Routes (3)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-0e5d36490ee98ec6d	Active	No	Create Route
10.1.0.0/16	pcx-06700d6c6ec60f483	Active	No	Create Route
10.2.0.0/16	local	Active	No	Create Route Table

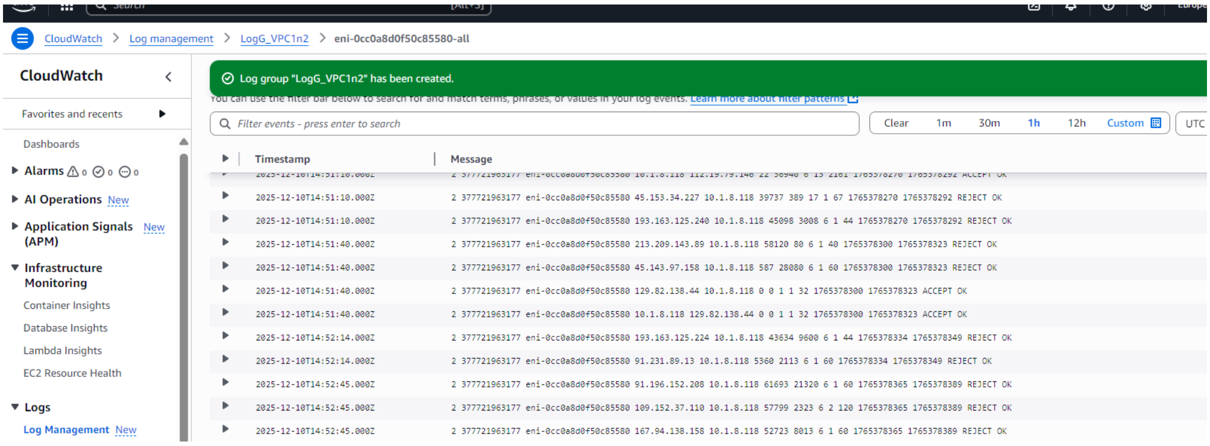
- Then I went on and added the **routes**.
- In the **VPC_1 route table**, I added the destination **10.2.0.0/16**.
- In the **VPC_2 route table**, I added the destination **10.1.0.0/16**.



```
64 bytes from 10.2.14.47: icmp_seq=87 ttl=127 time=0.198 ms
64 bytes from 10.2.14.47: icmp_seq=88 ttl=127 time=0.199 ms
64 bytes from 10.2.14.47: icmp_seq=89 ttl=127 time=0.200 ms
64 bytes from 10.2.14.47: icmp_seq=90 ttl=127 time=0.203 ms
64 bytes from 10.2.14.47: icmp_seq=91 ttl=127 time=0.204 ms
64 bytes from 10.2.14.47: icmp_seq=92 ttl=127 time=0.195 ms
64 bytes from 10.2.14.47: icmp_seq=93 ttl=127 time=0.197 ms
64 bytes from 10.2.14.47: icmp_seq=94 ttl=127 time=0.191 ms
64 bytes from 10.2.14.47: icmp_seq=95 ttl=127 time=0.203 ms
64 bytes from 10.2.14.47: icmp_seq=96 ttl=127 time=0.186 ms
64 bytes from 10.2.14.47: icmp_seq=97 ttl=127 time=0.188 ms
64 bytes from 10.2.14.47: icmp_seq=98 ttl=127 time=0.201 ms
64 bytes from 10.2.14.47: icmp_seq=99 ttl=127 time=0.195 ms
64 bytes from 10.2.14.47: icmp_seq=100 ttl=127 time=0.193 ms
64 bytes from 10.2.14.47: icmp_seq=101 ttl=127 time=0.197 ms
64 bytes from 10.2.14.47: icmp_seq=102 ttl=127 time=0.209 ms
64 bytes from 10.2.14.47: icmp_seq=103 ttl=127 time=0.224 ms
64 bytes from 10.2.14.47: icmp_seq=104 ttl=127 time=0.199 ms
64 bytes from 10.2.14.47: icmp_seq=105 ttl=127 time=0.189 ms
64 bytes from 10.2.14.47: icmp_seq=106 ttl=127 time=0.188 ms
64 bytes from 10.2.14.47: icmp_seq=107 ttl=127 time=0.177 ms
64 bytes from 10.2.14.47: icmp_seq=108 ttl=127 time=0.184 ms
64 bytes from 10.2.14.47: icmp_seq=109 ttl=127 time=0.197 ms
```

i-0b06246314d62ebb2 (Instance-VPC-1)
PublicIPs: 16.16.65.171 PrivateIPs: 10.1.8.118

- After that, the traffic started.



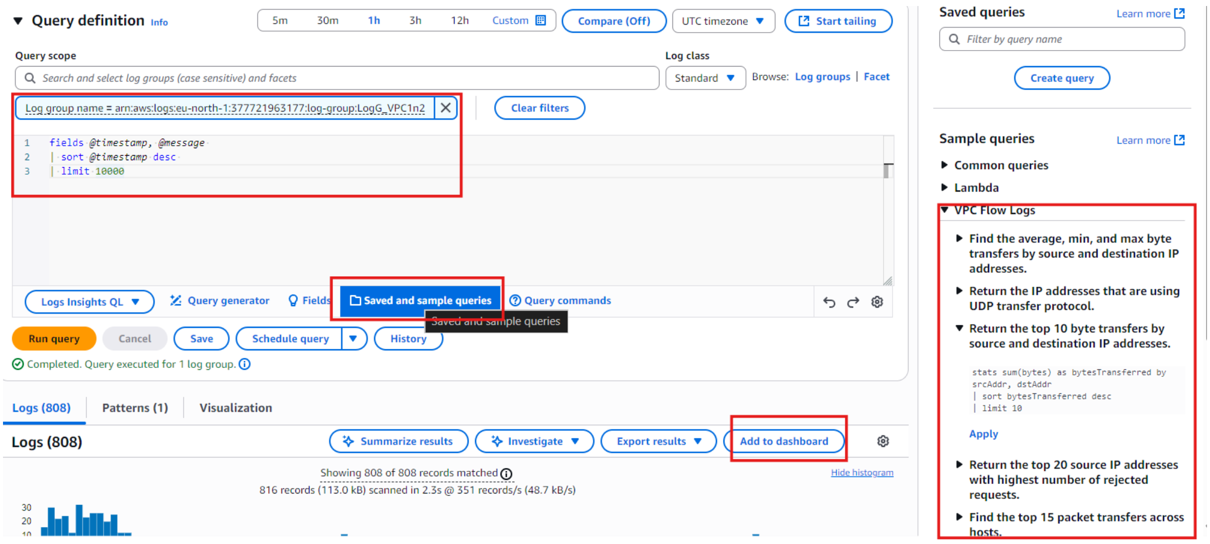
Log group "LogG_VPC1n2" has been created.

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

Filter events - press enter to search

```
Timestamp                  Message
2025-12-10T14:51:10.000Z   2 377721963177 eni-0cc0a8d0f50c85580 10.1.8.118 112.19.79.146 22 36940 6 13 2161 1765378270 1765378292 ACCEPT OK
2025-12-10T14:51:10.000Z   2 377721963177 eni-0cc0a8d0f50c85580 45.153.34.227 10.1.8.118 39737 389 17 1 67 1765378270 1765378292 REJECT OK
2025-12-10T14:51:10.000Z   2 377721963177 eni-0cc0a8d0f50c85580 193.163.125.240 10.1.8.118 45098 3008 6 1 44 1765378270 1765378292 REJECT OK
2025-12-10T14:51:40.000Z   2 377721963177 eni-0cc0a8d0f50c85580 213.209.143.89 10.1.8.118 58120 80 6 1 40 1765378300 1765378323 REJECT OK
2025-12-10T14:51:40.000Z   2 377721963177 eni-0cc0a8d0f50c85580 45.143.97.158 10.1.8.118 587 28080 6 1 60 1765378300 1765378323 REJECT OK
2025-12-10T14:51:40.000Z   2 377721963177 eni-0cc0a8d0f50c85580 129.82.138.44 10.1.8.118 0 0 1 1 32 1765378300 1765378323 ACCEPT OK
2025-12-10T14:51:40.000Z   2 377721963177 eni-0cc0a8d0f50c85580 10.1.8.118 129.82.138.44 0 0 1 1 32 1765378300 1765378323 ACCEPT OK
2025-12-10T14:52:14.000Z   2 377721963177 eni-0cc0a8d0f50c85580 193.163.125.224 10.1.8.118 43634 9600 6 1 44 1765378334 1765378349 REJECT OK
2025-12-10T14:52:14.000Z   2 377721963177 eni-0cc0a8d0f50c85580 91.231.89.13 10.1.8.118 5360 2113 6 1 60 1765378334 1765378349 REJECT OK
2025-12-10T14:52:45.000Z   2 377721963177 eni-0cc0a8d0f50c85580 91.196.152.208 10.1.8.118 61693 21320 6 1 60 1765378365 1765378389 REJECT OK
2025-12-10T14:52:45.000Z   2 377721963177 eni-0cc0a8d0f50c85580 109.152.37.110 10.1.8.118 57799 2323 6 2 120 1765378365 1765378389 REJECT OK
2025-12-10T14:52:45.000Z   2 377721963177 eni-0cc0a8d0f50c85580 167.94.138.158 10.1.8.118 52723 8013 6 1 60 1765378365 1765378389 REJECT OK
```

- After that, I went to **CloudWatch** and selected **Log management**, where the log group is located.
- When I selected the log group, the **logs were shown** to me.



Log group name = arn:aws:logs:eu-north-1:377721963177:log-group:LogG_VPC1n2

```
fields @timestamp, @message
| sort @timestamp desc
| limit 10000
```

Completed. Query executed for 1 log group.

Logs (808)

Showing 808 of 808 records matched

816 records (113.0 kB) scanned in 2.3s @ 351 records/s (48.7 kB/s)

Sample queries

- Common queries
- Lambda
- VPC Flow Logs
  - Find the average, min, and max byte transfers by source and destination IP addresses.
  - Return the IP addresses that are using UDP transfer protocol.
  - Return the top 10 byte transfers by source and destination IP addresses.

```
stats sum(bytes) as bytesTransferred by
srcAddr, dstAddr
| sort bytesTransferred desc
| limit 10
```

  - Return the top 20 source IP addresses with highest number of rejected requests.
  - Find the top 15 packet transfers across hosts.

- Then I selected **Log Insights** from the side panel.
- In Log Insights, there are many options for how we want the logs to appear and how they can be filtered.
- First, I had to select the **query scope**, and when I selected my log group, it gave me a **ready-made query**.
- We also have the option to **write our own query**, or choose from **saved** and **sample queries**.
- On the right side, there are sample query templates that we can use.
- At the end, we can also **add the query result to a CloudWatch dashboard**.

Conclusion

In this project, I learned how to monitor network traffic inside AWS by setting up **VPC Flow Logs** and analyzing the data through **CloudWatch Logs and Log Insights**. I created two VPCs, deployed instances inside them, and established a peering connection to allow private communication. After enabling flow logs, I was able to capture and observe all network activity between the VPCs.

I also created a custom IAM policy and role to allow VPC Flow Logs to deliver data to CloudWatch. With Log Insights, I explored different ways to filter, view, and analyze traffic patterns. This project helped me understand how logs are generated, how they are stored, and how they can be used to troubleshoot or monitor VPC connectivity.

Overall, this hands-on experience strengthened my understanding of AWS monitoring tools and gave me practical exposure to analyzing network traffic inside a cloud environment.