



[NextWork.org](https://NextWork.org)

# VPC Monitoring with Flow Logs



Alex Ten



Alex Ten  
NextWork Student

[NextWork.org](https://www.nextwork.org)

# Introducing Today's Project!

## What is Amazon VPC?

Amazon VPC is a foundational AWS service that lets us control the underline network for our resources, so that we can control traffic flow, monitor for security ,and organize our resources.

## How I used Amazon VPC in this project

In todays project we achieved two big milestones. First, we learned to troubleshoot VPC peering connectivity issues. Secondly, we learned how to monitor network traffic using VPC flow logs.

## One thing I didn't expect in this project was...

I didnt expect to see how queries really shows you whats going on under the hood. So much data can be analyzed for troubleshooting purposes.

## This project took me...

This project took me under 1 hr to do while learning how Peering, log insights and Flow Logs work.



Alex Ten  
NextWork Student

[NextWork.org](https://www.nextwork.org)

## In the first part of my project...

### Step 1 - Set up VPCs

In this step I have created the 2 VPC's from scratch in minutes. Networking monitoring can still be done with just a single VPC.

### Step 2 - Launch EC2 instances

In this step I will be creating and launching two instances, one in each VPC. Our EC2 instances will generate traffic that our VPC flow logs will monitor.

### Step 3 - Set up Logs

In this step we are setting up VPC flow logs to start monitoring network traffic. We are also setting up a storage space for our flow logs.

## Step 4 - Set IAM permissions for Logs

In this step we provide VPC Flow Logs with the permission to create logs and upload them into our log group in CloudWatch.



Alex Ten  
NextWork Student

[NextWork.org](https://NextWork.org)

## Multi-VPC Architecture

I started my project by launching 2 VPCs and created 2 public subnets. One public subnets per VPC with no private subnets.

Each subnet must have a unique CIDR block to ensure that it can be properly identified and communicate within the VPC. Overlapping CIDR blocks between subnets should be avoided to prevent routing conflicts and potential network issues."

### I also launched EC2 instances in each subnet

My EC2 instances' security groups allow all SSH traffic and ICMP type traffic. EC2 Instance Connect will need to access both traffic types for connectivity tests.

The screenshot shows the 'Resource map' section of the NextWork interface. It displays four main components: 'Subnets (1)', 'Route tables (2)', and 'Network connections (1)'. Each component has a detailed view below it. 'Subnets (1)' shows 'us-east-1a' with 'VPC for peering 2-subnet-public1...'. 'Route tables (2)' shows 'rtb-0907082950c19d551'. 'Network connections (1)' shows 'VPC for peering 2-lgw'.



Alex Ten  
NextWork Student

[NextWork.org](https://www.nextwork.org)

# Logs

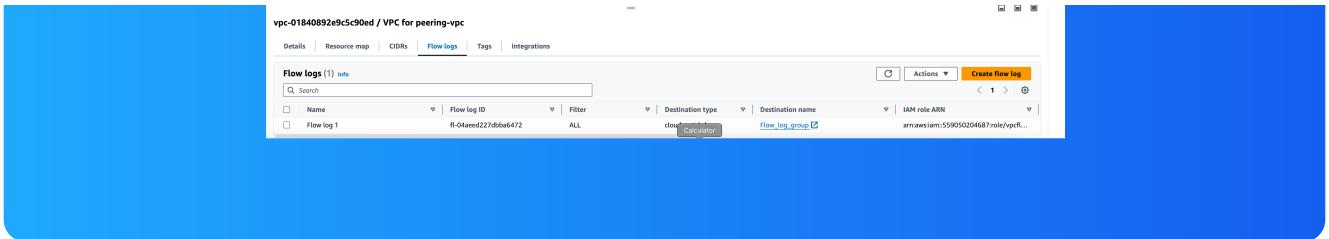
Logs are data entries for our computer system. They're detailed activity related to the traffic/resource/AWS service that the logs tracks.

Log groups are like folders that contain the actual log activity within the project/application/source and are often in a log group together.

I also set up a flow log for VPC 1

The screenshot shows the 'Your VPCs (1/3) Info' section of the AWS CloudWatch Metrics interface. It lists three VPCs: 'VPC for peering-vpc', 'VPC for peering 2...', and 'VPC for peering 3...'. Each VPC entry includes its Name, VPC ID, State, IPv4 CIDR, IPv6 CIDR, DHCP option set, Main route table, Main network ACL, and Tenuity. A message at the top indicates a successful flow log creation.

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP option set	Main route table	Main network ACL	Tenuity
VPC for peering-vpc	vpc-01840892e5cb...	Available	10.1.0.0/16	-	dopt-069f554c9dc7bf8e6	rtb-0a0f02353632c55	ad-06955847fa45fb0c	Default
VPC for peering 2...	vpc-02f26ed919d55...	Available	10.2.0.0/16	-	dopt-069f554c9dc7bf8e6	rtb-0907082950c19d551	ad-0655573218647a1846	Default
VPC for peering 3...	vpc-0921427cd8d627...	Available	172.1.0.0/16	-	dopt-069f554c9dc7bf8e6	rtb-0d4edf3f82a2c53ed	ad-0b7efadef5e5e44b6	Default



**Alex Ten**  
NextWork Student

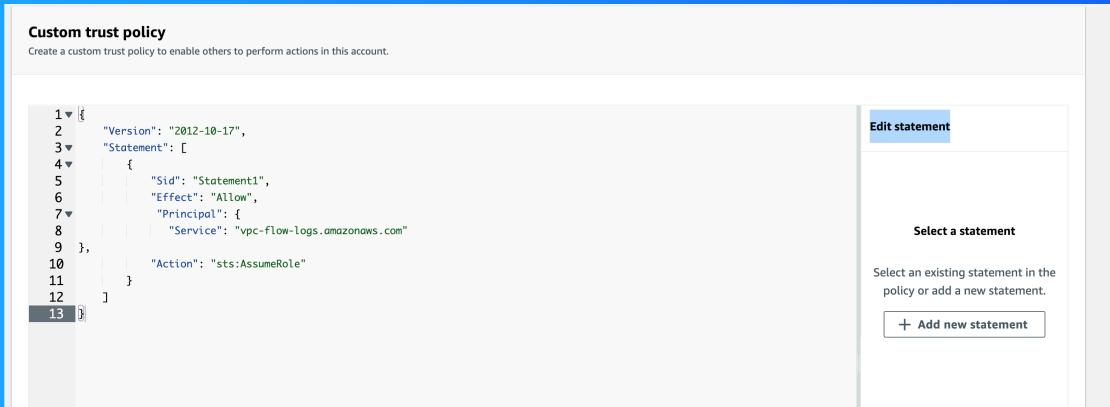
[NextWork.org](https://www.nextwork.org)

# IAM Policy and Roles

I created an IAM policy so that we can define a rule that allows policy holders e.g our VPC Flow Logs service the ability to create log streams and upload them into CloudWatch.

I also created an IAM role because services like VPC Flow Logs have to be associated with a role instead of JSON. Creating an IAM role will be necessary to give our VPC Flow Logs the access it needs to record and upload logs.

A custom trust policy is a specific type of policy used for designing who / what is allowed to access the IAM role.



The screenshot shows the 'Custom trust policy' configuration page in the AWS IAM console. The policy document is displayed as JSON code:

```
1▼ [{  
2    "Version": "2012-10-17",  
3    "Statement": [  
4        {  
5            "Sid": "Statement1",  
6            "Effect": "Allow",  
7            "Principal": {  
8                "Service": "vpc-flow-logs.amazonaws.com"  
9            },  
10           "Action": "sts:AssumeRole"  
11        }  
12    ]  
13}]
```

On the right side, there are three buttons: 'Edit statement' (highlighted in blue), 'Select a statement', and '+ Add new statement'.



Alex Ten  
NextWork Student

[NextWork.org](https://www.nextwork.org)

# In the second part of my project...

## Step 5 - Ping testing and troubleshooting

In this step we are generating network traffic. This becomes important we are communicating about cloudworks/cloud networking/or cloud engineering.

## Step 6 - Set up a peering connection

In this step we are setting up a peering connection so that our VPC's can communicate with each other.

## Step 7 - Update VPC route tables

## Step 8 - Analyze flow logs

In this step we are tracking the network data that's been collected on our VPC's and then analyze that data to extracts insights.

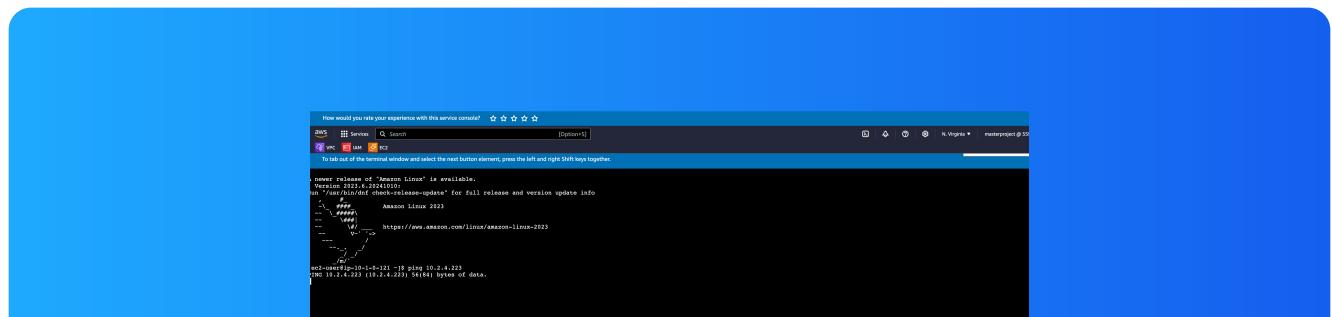


Alex Ten  
NextWork Student

[NextWork.org](https://NextWork.org)

## Connectivity troubleshooting

My first ping test between my EC2 instances had no replies, which means icmp Traffic could be blocked. It could be blocked by the security groups/network acls, or maybe our traffic is being routed to a wrong path.





I could receive ping replies if I ran the ping test using the other instance's public IP address, which means second instance is actually allowing ICMP traffic.



Alex Ten  
NextWork Student

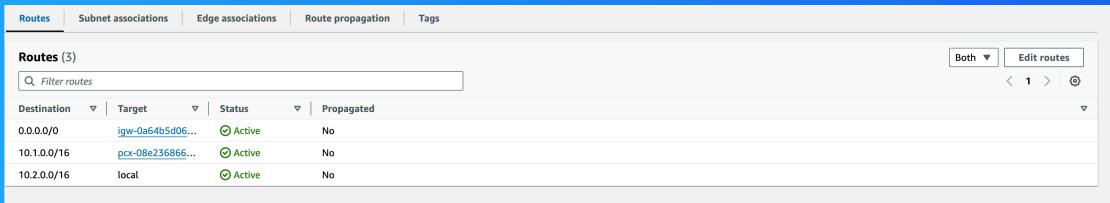
[NextWork.org](https://www.nextwork.org)

## Connectivity troubleshooting

Looking at VPC 1's route table, I identified that the ping test with Instance 2's private address failed because we do not have a route in our VPC's route tables that directs traffic from VPC to another.

**To solve this, I set up a peering connection between my VPCs**

I also updated both VPCs' route tables, so that traffic of one of the VPC's heading to other VPC's private IPV4 address can get directed to go through the peering connection instead of the public internet.



A screenshot of the AWS Route Tables interface. The top navigation bar includes tabs for Routes, Subnet associations, Edge associations, Route propagation, and Tags. The 'Routes' tab is selected. Below the tabs is a search bar labeled 'Filter routes'. To the right of the search bar are buttons for 'Both' (selected), 'Edit routes', and navigation arrows. The main content area displays a table of routes:

Destination	Target	Status	Propagated
0.0.0.0/0	igw-0a64b5d06...	Active	No
10.1.0.0/16	pxx-08e236866...	Active	No
10.2.0.0/16	local	Active	No



Alex Ten  
NextWork Student

[NextWork.org](https://www.nextwork.org)

# Connectivity troubleshooting

We received ping replies from instance 2's private IP address. This means setting up the peering connection and the route table solved the connectivity error of our VPC's traffic not being able to navigate VPC to another.

```
64 bytes from 10.2.4.223: icmp_seq=12 ttl=127 time=0.661 ms
64 bytes from 10.2.4.223: icmp_seq=13 ttl=127 time=0.968 ms
64 bytes from 10.2.4.223: icmp_seq=14 ttl=127 time=1.05 ms
64 bytes from 10.2.4.223: icmp_seq=15 ttl=127 time=1.01 ms
64 bytes from 10.2.4.223: icmp_seq=16 ttl=127 time=1.38 ms
64 bytes from 10.2.4.223: icmp_seq=17 ttl=127 time=1.21 ms
64 bytes from 10.2.4.223: icmp_seq=18 ttl=127 time=2.35 ms
64 bytes from 10.2.4.223: icmp_seq=19 ttl=127 time=1.30 ms
64 bytes from 10.2.4.223: icmp_seq=20 ttl=127 time=1.11 ms
64 bytes from 10.2.4.223: icmp_seq=21 ttl=127 time=0.997 ms
64 bytes from 10.2.4.223: icmp_seq=22 ttl=127 time=0.552 ms
64 bytes from 10.2.4.223: icmp_seq=23 ttl=127 time=1.12 ms
64 bytes from 10.2.4.223: icmp_seq=24 ttl=127 time=0.618 ms
64 bytes from 10.2.4.223: icmp_seq=25 ttl=127 time=0.523 ms
64 bytes from 10.2.4.223: icmp_seq=26 ttl=127 time=0.595 ms
64 bytes from 10.2.4.223: icmp_seq=27 ttl=127 time=1.38 ms
64 bytes from 10.2.4.223: icmp_seq=28 ttl=127 time=0.956 ms
64 bytes from 10.2.4.223: icmp_seq=29 ttl=127 time=0.981 ms
64 bytes from 10.2.4.223: icmp_seq=30 ttl=127 time=0.694 ms
64 bytes from 10.2.4.223: icmp_seq=31 ttl=127 time=0.692 ms
64 bytes from 10.2.4.223: icmp_seq=32 ttl=127 time=0.692 ms
64 bytes from 10.2.4.223: icmp_seq=33 ttl=127 time=0.415 ms
64 bytes from 10.2.4.223: icmp_seq=34 ttl=127 time=0.373 ms
64 bytes from 10.2.4.223: icmp_seq=35 ttl=127 time=0.610 ms
64 bytes from 10.2.4.223: icmp_seq=36 ttl=127 time=0.729 ms
64 bytes from 10.2.4.223: icmp_seq=37 ttl=127 time=1.14 ms
64 bytes from 10.2.4.223: icmp_seq=38 ttl=127 time=1.68 ms
64 bytes from 10.2.4.223: icmp_seq=39 ttl=127 time=0.533 ms
64 bytes from 10.2.4.223: icmp_seq=40 ttl=127 time=1.46 ms
64 bytes from 10.2.4.223: icmp_seq=41 ttl=127 time=0.48 ms
64 bytes from 10.2.4.223: icmp_seq=42 ttl=127 time=1.48 ms
64 bytes from 10.2.4.223: icmp_seq=43 ttl=127 time=0.706 ms
64 bytes from 10.2.4.223: icmp_seq=44 ttl=127 time=1.19 ms
64 bytes from 10.2.4.223: icmp_seq=45 ttl=127 time=2.03 ms
64 bytes from 10.2.4.223: icmp_seq=46 ttl=127 time=0.47 ms
64 bytes from 10.2.4.223: icmp_seq=47 ttl=127 time=0.441 ms
64 bytes from 10.2.4.223: icmp_seq=48 ttl=127 time=0.666 ms
64 bytes from 10.2.4.223: icmp_seq=49 ttl=127 time=0.877 ms
64 bytes from 10.2.4.223: icmp_seq=50 ttl=127 time=0.607 ms
64 bytes from 10.2.4.223: icmp_seq=51 ttl=127 time=0.640 ms
64 bytes from 10.2.4.223: icmp_seq=52 ttl=127 time=1.45 ms
64 bytes from 10.2.4.223: icmp_seq=53 ttl=127 time=0.671 ms
```

Screenshot



Alex Ten  
NextWork Student

[NextWork.org](https://www.nextwork.org)

# Analyzing flow logs

Flow logs tell us about the source and destination of the network traffic and the amount of data being transferred, whether the traffic was accepted or rejected.

For example, the flow log I've captured tells us that the traffic went from that the log group associated with a network interface (ENI) with the ID "eni-0694526016d645852, also a timestamp for each event and if it was accepted or rejected.

Timestamp	Message
2024-10-19T04:54:15.000Z	2 55980240487 eni-0694526016d645852 147.185.132.168 10.1.0.321 98497 32 6 1 44 1729316055 3729513686 ACCEPT OK
2024-10-19T04:54:15.000Z	2 55980240487 eni-0694526016d645852 199.45.154.176 10.1.0.321 9454 83 1 44 1729316055 3729513686 REJECT OK
2024-10-19T04:54:15.000Z	2 55980240487 eni-0694526016d645852 19.1.0.221 197.18.1.114 388 22 9847 6 1 44 1729316055 3729513686 ACCEPT OK
2024-10-19T04:54:15.000Z	2 55980240487 eni-0694526016d645852 19.1.0.221 197.18.1.114 388 22 9847 6 1 44 1729316055 3729513686 ACCEPT OK
2024-10-19T04:54:15.000Z	2 55980240487 eni-0694526016d645852 35.283.210.106 10.1.0.321 934 43396 6 1 44 1729316055 3729513686 REJECT OK
2024-10-19T04:54:15.000Z	2 55980240487 eni-0694526016d645852 54.157.218.248 10.1.0.321 9374 17 1 44 1729316055 3729513686 ACCEPT OK
2024-10-19T04:54:15.000Z	2 55980240487 eni-0694526016d645852 79.109.42.193 10.1.0.321 9454 8399 6 1 44 1729316055 3729513686 REJECT OK
2024-10-19T04:54:48.000Z	2 55980240487 eni-0694526016d645852 206.168.34.10 10.1.0.321 98497 1897 6 1 44 1729316055 3729513686 REJECT OK
2024-10-19T04:54:48.000Z	2 55980240487 eni-0694526016d645852 13.88.2.56 10.1.0.321 9319 431 6 1 44 1729316055 3729513686 REJECT OK
2024-10-19T04:54:48.000Z	2 55980240487 eni-0694526016d645852 147.185.132.41 10.1.0.321 9328 9396 6 1 44 1729316055 3729513686 REJECT OK
2024-10-19T04:54:48.000Z	2 55980240487 eni-0694526016d645852 49.213.289.59 10.1.0.321 9355 8899 6 1 44 1729316055 3729513686 REJECT OK
2024-10-19T04:54:48.000Z	2 55980240487 eni-0694526016d645852 87.247.158.145 10.1.0.321 93797 17269 6 1 44 1729316055 3729513686 REJECT OK
2024-10-19T04:54:48.000Z	2 55980240487 eni-0694526016d645852 95.214.27.29 10.1.0.321 45728 3886 6 1 44 1729316055 3729513686 REJECT OK
2024-10-19T04:54:48.000Z	2 55980240487 eni-0694526016d645852 87.247.158.174 10.1.0.321 93751 1489 6 1 44 1729316055 3729513686 REJECT OK
2024-10-19T04:54:48.000Z	2 55980240487 eni-0694526016d645852 147.185.155.126 10.1.0.321 48484 6867 6 1 44 1729316055 3729513686 REJECT OK
2024-10-19T04:54:48.000Z	2 55980240487 eni-0694526016d645852 45.64.89.2 10.1.0.321 63482 951 6 1 44 1729316055 3729513686 REJECT OK



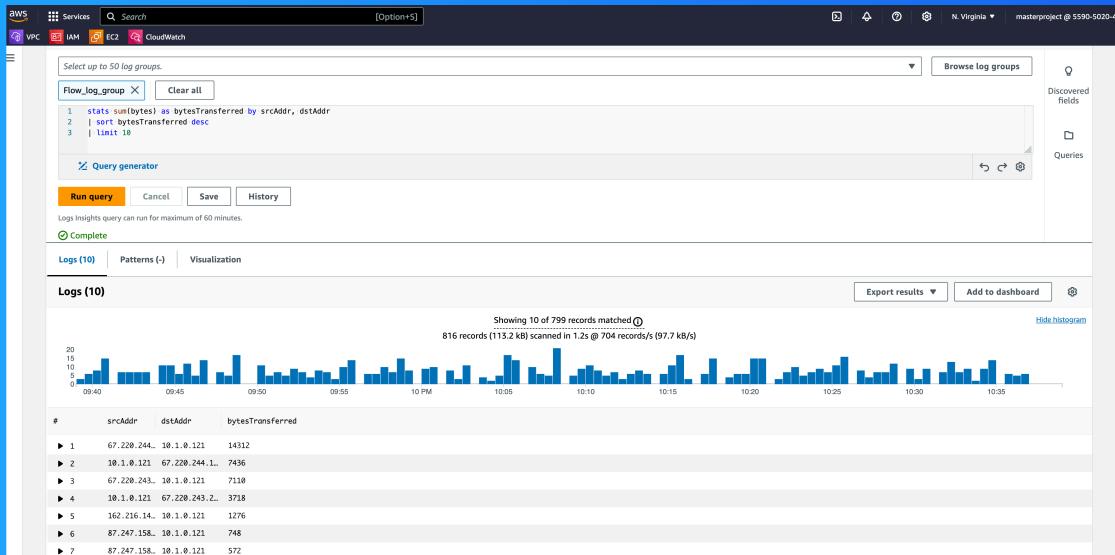
Alex Ten  
NextWork Student

[NextWork.org](http://NextWork.org)

# Logs Insights

Logs Insights is a special tool within Amazon CloudWatch that helps us with analysing logs and creating visual graphs and charts through queries.

I ran the query Top 10 byte transfers by source and destination IP addresses, This query analyses the flow logs collected on EC2 Instance 1, and return the top 10 pairs of IP addresses based on the amount of data transferred between them.



NextWork.org

# Everyone should be in a job they love.

Check out [nextwork.org](https://nextwork.org) for  
more projects

