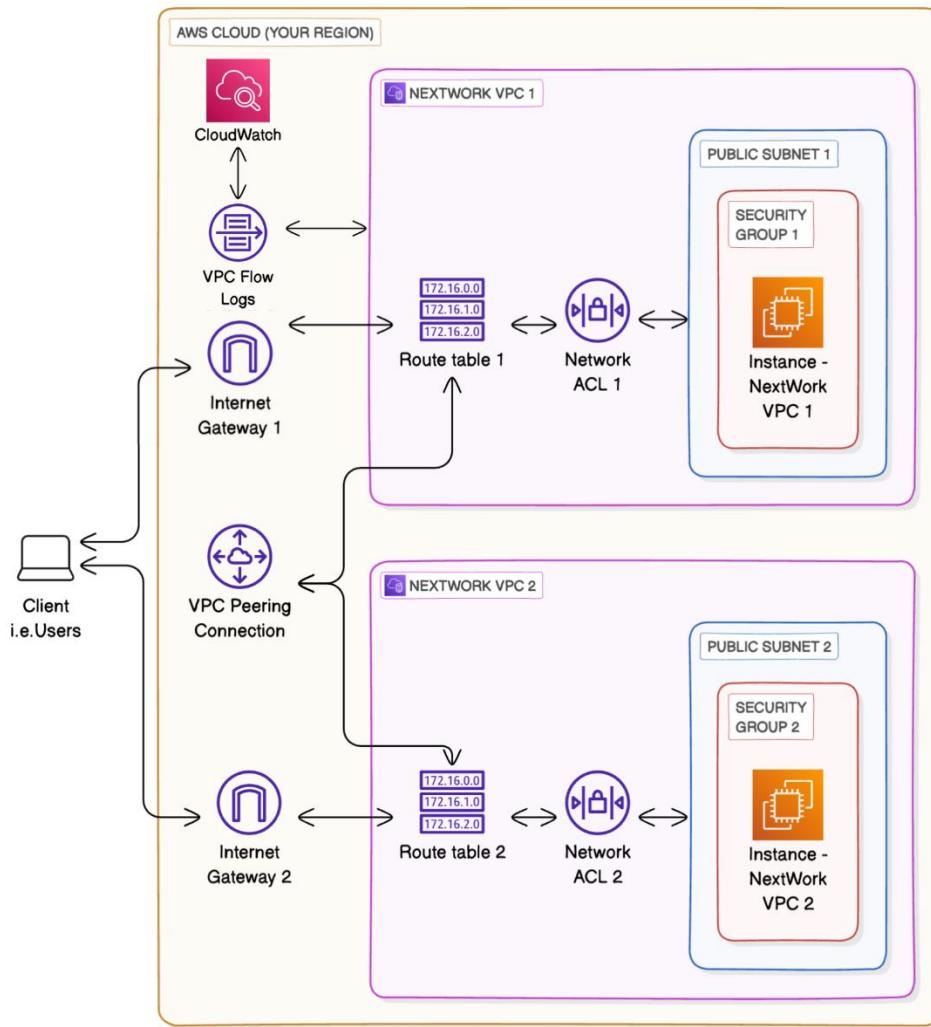


## VPC Monitoring with Flow Logs

Get ready to:

1. Set up two VPCs and test their peering connection.
2. Set up VPC Flow Logs that collect network traffic data.
3. Analyse network traffic using CloudWatch's Log Insights.



### Create a Peering Connection and Configure Route Tables

Head to the VPC console, click on **Peering connections** on the left-hand navigation panel.

Click on **Create peering connection** in the right-hand corner.

- Name your **Peering connection name** as VPC 1 <> VPC 2
- Select **NextWork-1-VPC** for your **VPC ID (Requester)**.
- Under **Select another VPC to peer with**, make sure **My Account** is selected.
- For **Region**, select **This Region**.
- For **VPC ID (Acceptor)**, select **NextWork-2-VPC**
- Click on **Create peering connection**.

## Create peering connection

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them privately. [Info](#)

### Peering connection settings

#### Name - optional

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

VPC 1 <> VPC 2

#### Select a local VPC to peer with

##### VPC ID (Requester)

vpc-0e970c2613daa05e7 (NextWork-1-vpc)

##### VPC CIDRs for vpc-0e970c2613daa05e7 (NextWork-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 (us-east-1)
- Another Region

##### VPC ID (Acceptor)

- Your newly created peering connection isn't finished yet! The green success bar says the peering connection **has been requested**.
- On the next screen, select **Actions** and then select **Accept request...**
- Click on **Accept request** again on the pop-up panel.

(i) A VPC peering connection pcx-0657b7033d0a74d61 / VPC 1 <> VPC 2 has been requested.

**pcx-0657b7033d0a74d61 / VPC 1 <> VPC 2**

(i) Pending acceptance  
You can accept or reject this peering connection request using the 'Actions' menu. You have until Monday, February 3, 2025 at 13:11:35 GMT+1 to accept or reject the request, expires.

Details <a href="#">Info</a>		Accept VPC peering connection request <a href="#">Info</a>	
Requester owner ID	329599662468	Requester VPC	vpc-0e970c2613daa05e7 / NextWork-1-vpc
Peering connection ID	pcv-0657b7033d0a74d61	Acceptor VPC	vpc-098733df15802b758 / NextWork-2-vpc
Status	(i) Pending Acceptance by 329599662468	Requester CIDRs	10.1.0.0/16
Expiration time	Monday, February 3, 2025 at 13:11:35	Acceptor CIDRs	-
Requester owner ID	329599662468 (This account)	Requester Region	N. Virginia (us-east-1)
Peering connection ID	pcv-0657b7033d0a74d61	Acceptor owner ID	329599662468 (This account)
Status	(i) Pending Acceptance by 329599662468	Requester Region	N. Virginia (us-east-1)
Expiration time	Monday, February 3, 2025 at 13:11:35	Acceptor Region	N. Virginia (us-east-1)
DNS	Route tables	Cancel	<b>Accept request</b>

- Click on **Modify my route tables now** on the top right corner.

(i) Your VPC peering connection (pcx-0657b7033d0a74d61 | VPC 1 <> VPC 2) has been established.  
To send and receive traffic across this VPC peering connection, you must add a route to the peered VPC in one or more of your VPC route tables. [Info](#)

**Modify my route tables now** X

- A peering connection is all set now

## Update VPC 1's route table

- Select the checkbox next to VPC 1's route table i.e. called **NextWork-1-rtb-public**.
- Scroll down and click on the **Routes** tab.
- Click **Edit routes**.
- Let's add a new route.
- Add a new route to **VPC 2** by entering the CIDR block 10.2.0.0/16 as our **Destination**.
- Under Target, select **Peering Connection**.
- Select **VPC 1 < VPC 2**.

Edit routes

Destination	Target	Status	Propagated	
10.1.0.0/16	local	Active	No	
0.0.0.0/0	Internet Gateway	Active	No	<button>Remove</button>
10.1.0.0/16	Peering Connection	-	No	<button>Remove</button>

- Click **Save changes**.
- Confirm that the new route appears in VPC 1's **Routes** tab!

## Update VPC 2's route table

- The route table you're updating is **NextWork-2-rtb-public**.
- The **Destination** is the CIDR block 10.1.0.0/16

Routes (3)				
Destination	Target	Status	Propagated	
0.0.0.0/0	igw-0f6a7f78bbd1da167	Active	No	
10.1.0.0/16	local	Active	No	
10.2.0.0/16	pcx-0657b7033d0a74d61	Active	No	

- You save your changes
- Revisit the **EC2 Instance Connect** tab that's connected to NextWork Public Server.

```
[ec2-user@ip-10-1-5-112 ~]$ ping 10.2.15.210
PING 10.2.15.210 (10.2.15.210) 56(84) bytes of data.
64 bytes from 10.2.15.210: icmp_seq=412 ttl=127 time=0.522 ms
64 bytes from 10.2.15.210: icmp_seq=413 ttl=127 time=0.494 ms
64 bytes from 10.2.15.210: icmp_seq=414 ttl=127 time=0.503 ms
64 bytes from 10.2.15.210: icmp_seq=415 ttl=127 time=0.513 ms
64 bytes from 10.2.15.210: icmp_seq=416 ttl=127 time=0.519 ms
64 bytes from 10.2.15.210: icmp_seq=417 ttl=127 time=0.469 ms
64 bytes from 10.2.15.210: icmp_seq=418 ttl=127 time=0.526 ms
64 bytes from 10.2.15.210: icmp_seq=419 ttl=127 time=0.493 ms
64 bytes from 10.2.15.210: icmp_seq=420 ttl=127 time=0.514 ms
64 bytes from 10.2.15.210: icmp_seq=421 ttl=127 time=0.513 ms
```

- You've successfully resolved the connectivity issue by setting up a peering architecture between VPC 1 and VPC 2!

- As an extension (this is optional!), validate this by trying to run the same **ping command** from your **local terminal**.
- Do you get ping replies here too?

```
PING 10.2.15.210 (10.2.15.210): 56 data bytes
Request timeout for icmp_seq 0
Request timeout for icmp_seq 1
Request timeout for icmp_seq 2
```

- You can quit your local terminal's session now! Validating DONE 😊
- Another optional extension! Back in your EC2 Instance Connect tab, run the same ping command but add -c 5 to the end of the command.
- Your final result should look like ping 10.2.15.210 -c 5

```
[ec2-user@ip-10-1-5-112 ~]$ ping 10.2.15.210 -c 5
PING 10.2.15.210 (10.2.15.210) 56(84) bytes of data.
64 bytes from 10.2.15.210: icmp_seq=1 ttl=127 time=0.539 ms
64 bytes from 10.2.15.210: icmp_seq=2 ttl=127 time=0.693 ms
64 bytes from 10.2.15.210: icmp_seq=3 ttl=127 time=0.500 ms
64 bytes from 10.2.15.210: icmp_seq=4 ttl=127 time=0.600 ms
64 bytes from 10.2.15.210: icmp_seq=5 ttl=127 time=0.537 ms

--- 10.2.15.210 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4141ms
rtt min/avg/max/mdev = 0.500/0.573/0.693/0.067 ms
```

## Analyze Flow Logs

In this step, you're going to:

1. Review the flow logs recorded about VPC 1's public subnet.
2. Analyse the flow logs to get some tasty insights
3. Head to your **CloudWatch** console.
4. Select **Log groups** from the left-hand navigation panel.
5. Click into **NextWorkVPCFlowLogsGroup**.
6. Click into your log stream to see flow logs from EC2 Instance 1!

The screenshot shows the AWS CloudWatch Log Events interface. At the top, there are buttons for 'Log events' (highlighted), 'Actions', 'Start tailing', and 'Create metric filter'. Below this is a filter bar with a search input ('Filter events - press enter to see'), time range ('1m 1h Clear'), UTC timezone dropdown, and a 'Display' dropdown. A link 'more about filter patterns' is also present. The main area displays a table with columns 'Timestamp' and 'Message'. The message column contains log entries for EC2 instance 1, such as: '2024-08-01T23:45:42.000Z 2 471112976395 eni-08a0e21a6bb867b64 18.237.140.165.', '2024-08-01T23:45:42.000Z 2 471112976395 eni-08a0e21a6bb867b64 10.1.5.112 18...', and several other entries. A note at the bottom says 'There are older events to load. [Load more](#)'.

Timestamp	Message
2024-08-01T23:45:42.000Z	2 471112976395 eni-08a0e21a6bb867b64 18.237.140.165..
2024-08-01T23:45:42.000Z	2 471112976395 eni-08a0e21a6bb867b64 10.1.5.112 18...
2024-08-01T23:45:42.000Z	2 471112976395 eni-08a0e21a6bb867b64 205.210.31.15 ..
2024-08-01T23:45:42.000Z	2 471112976395 eni-08a0e21a6bb867b64 172.234.94.172..
2024-08-01T23:45:42.000Z	2 471112976395 eni-08a0e21a6bb867b64 35.203.211.18 ..

7. In the left-hand navigation panel, click on **Logs Insights**.
8. Select **NextWorkVPCFlowLogsGroup** from the **Select log group(s)** dropdown.
9. Select the **Queries** folder on the right-hand side.
  
  
  
10. Under **Flow Logs**, select **Top 10 byte transfers by source and destination IP addresses**.
11. Click **Apply**, and then **Run query**.

The screenshot shows the AWS CloudWatch Logs Insights interface. At the top, there's a search bar with "Select up to 50 log groups." and a dropdown arrow, followed by a "Browse log groups" button. Below it, a box contains the query:

```
1 fields @timestamp, @message, @logStream, @log
2 | sort @timestamp desc
3 | limit 10000
```

On the right, a sidebar titled "Discoverd fields" shows a "Queries" section with a blue outline. Below the query editor are buttons for "Run query" (orange), "Cancel", "Save", and "History". A note at the bottom says "Logs Insights query can run for maximum of 60 minutes".

Underneath, the results are displayed. It starts with a histogram titled "Showing 10 of 1,105 records matched" with a link to "Hide histogram". The histogram shows data between 11:15 and 11:45 on Friday, 02. The x-axis is time and the y-axis is byte count (0-50). Below the histogram is a summary: "1,105 records (153.2 kB) scanned in 2.3s @ 487 records/s (67.6 kB/s)".

At the top of the results table, there are buttons for "Logs (10)", "Export results", "Add to dashboard", and a refresh icon. The table has columns: #, srcAddr, dstAddr, and bytesTransferred. The data is as follows:

#	srcAddr	dstAddr	bytesTransferred
► 1	52.92.209.138	10.1.5.112	27779598
► 2	3.5.76.41	10.1.5.112	344895
► 3	10.1.5.112	10.2.15.210	48720
► 4	10.1.5.112	52.92.209.138	33338
► 5	52.218.236.137	10.1.5.112	32993
► 6	10.1.5.112	18.237.140.165	32733
► 7	18.237.140.165	10.1.5.112	32480
► 8	10.1.5.112	54.185.174.167	26880
► 9	52.94.182.119	10.1.5.112	13920