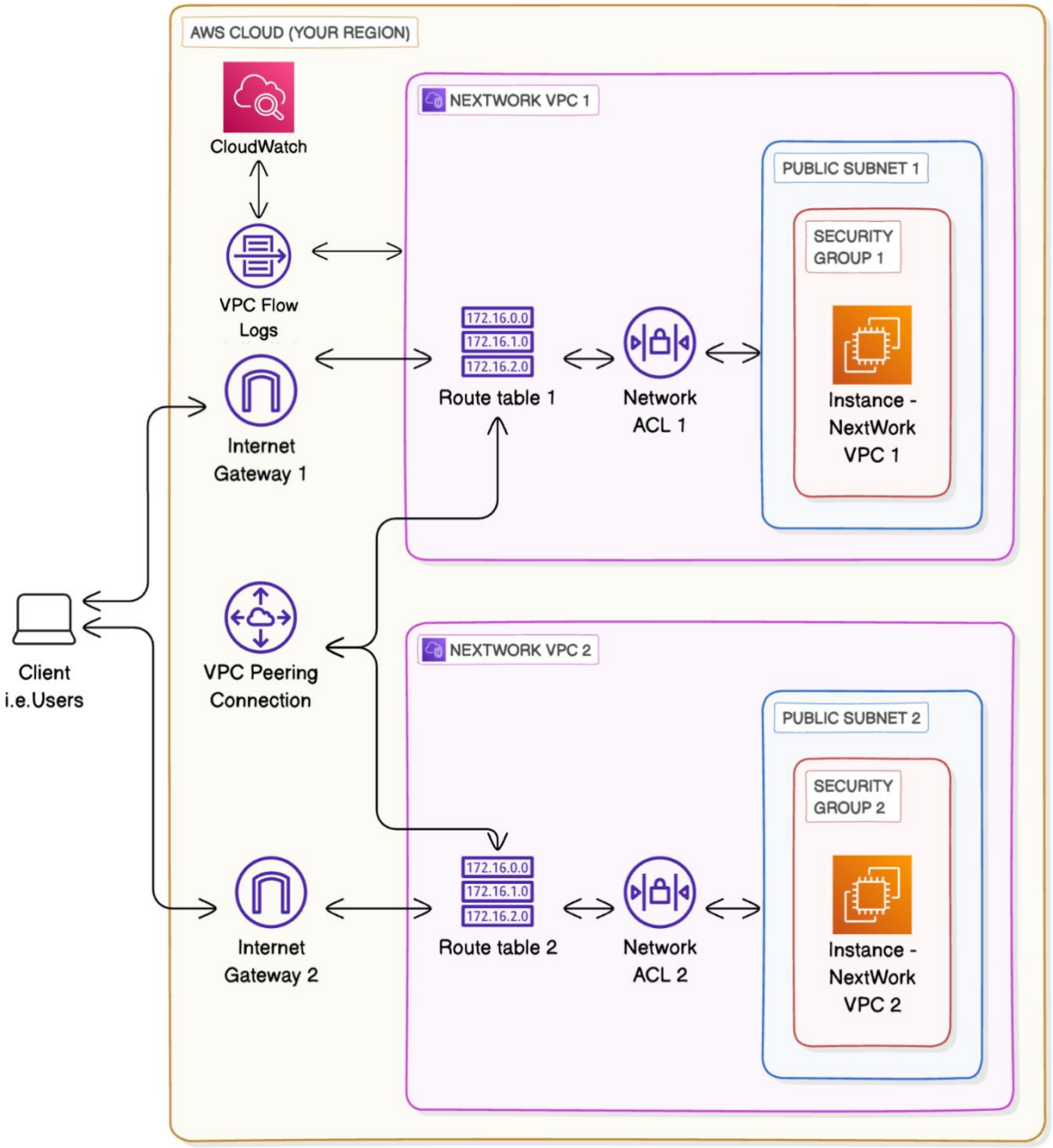
VPC Monitoring with Flow Logs

In this project, **VPC Monitoring with Flow Logs**, we're adding **monitoring** to our VPC.

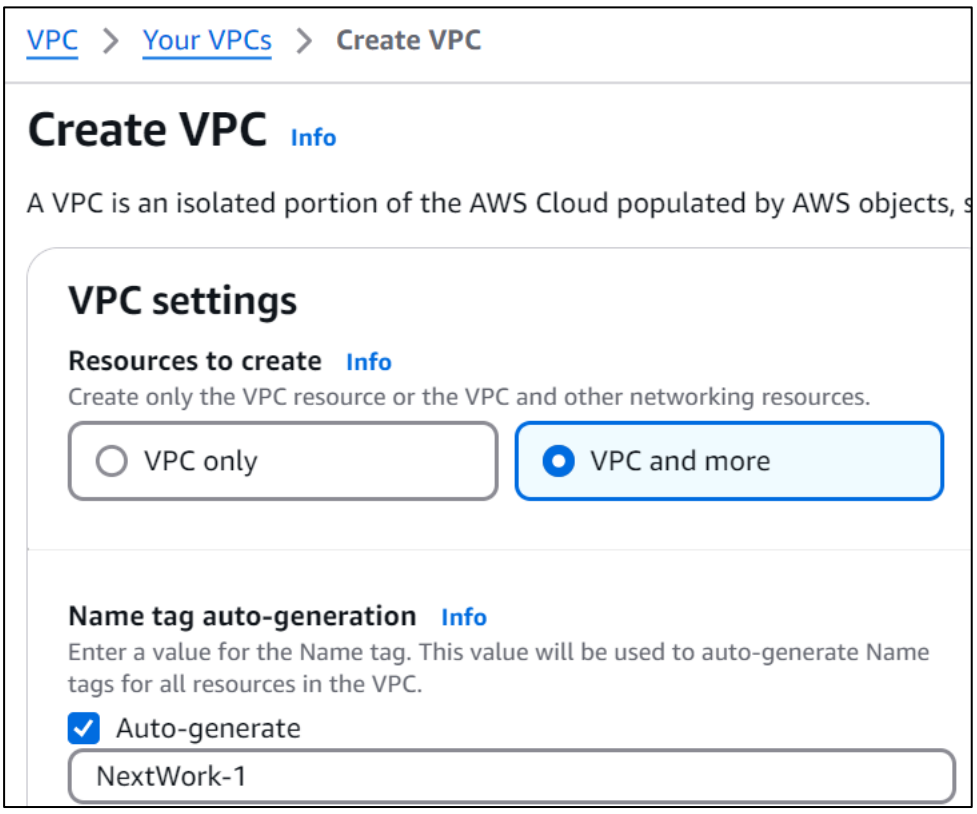


Set up your VPCs

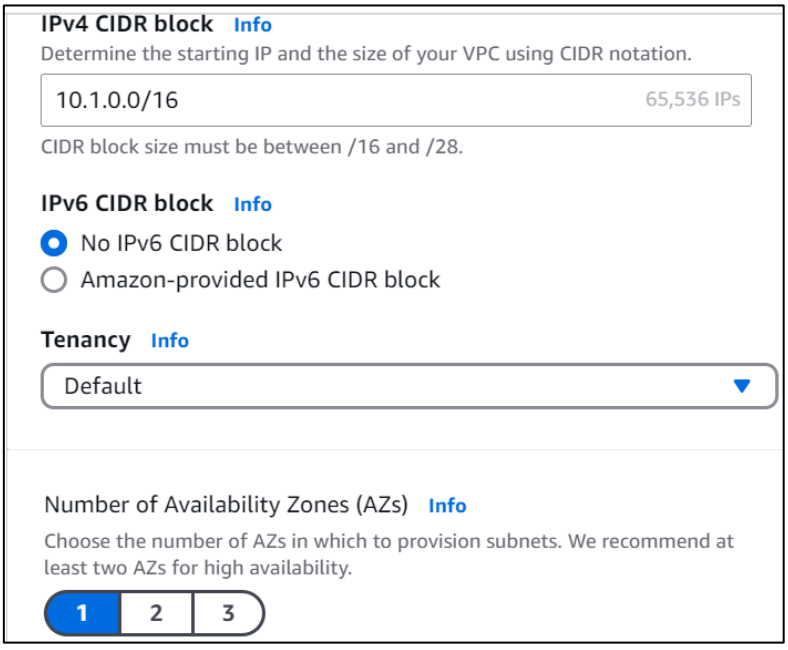
- Log in to your AWS Account.
- Head to your **VPC** console - search for VPC at the search bar at top of your page.
- From the left hand navigation bar, select **Your VPCs.**
- Select **Create VPC.**
- Select **VPC and more.**

Create VPC 1

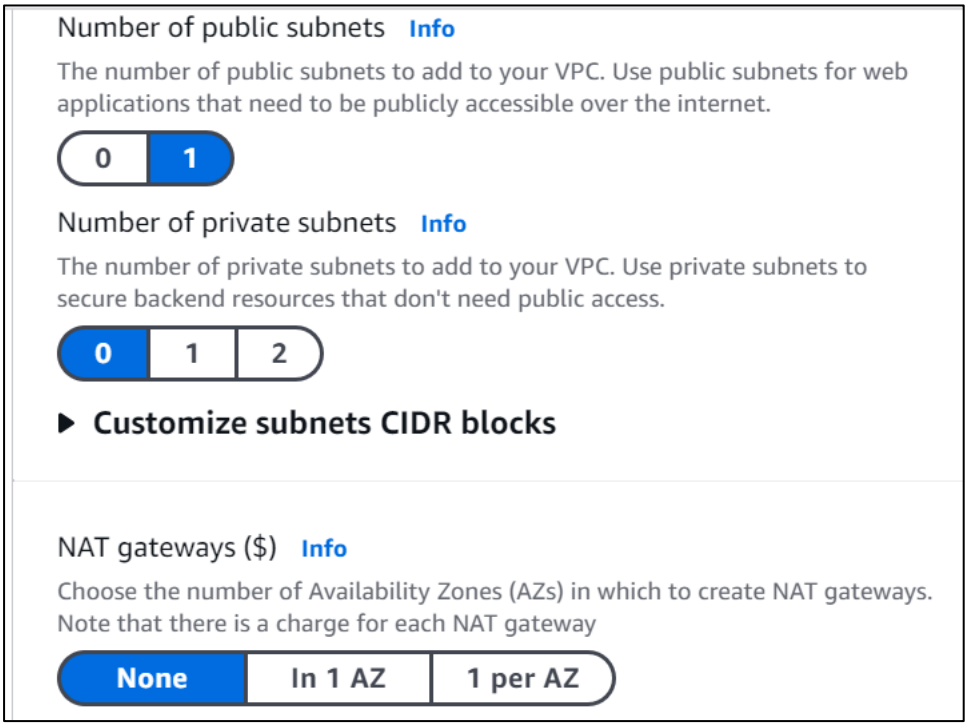
- Under **Name tag auto-generation**, enter NextWork-1



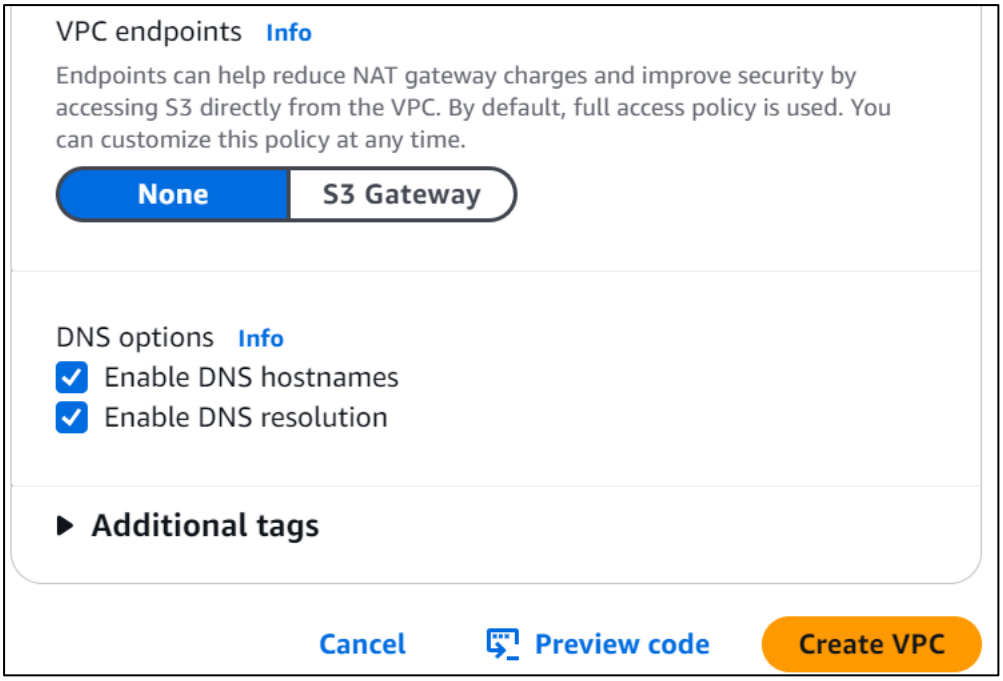
- The VPC's **IPv4 CIDR block** is already pre-filled to 10.0.0.0/16 - change that to 10.1.0.0/16
- For **IPv6 CIDR block**, we'll leave in the default option of **No IPv6 CIDR block.**
- For **Tenancy**, we'll keep the selection of **Default.**
- For **Number of Availability Zones (AZs)**, we'll use just **1** Availability Zone.



- Make sure the **Number of public subnets** chosen is **1**.
- For **Number of private subnets,** we'll keep thing simple today and go with **0** private subnets.
- Next, for the **NAT gateways ($)** option, make sure you've selected **None.** As the dollar sign suggests, NAT gateways cost money!

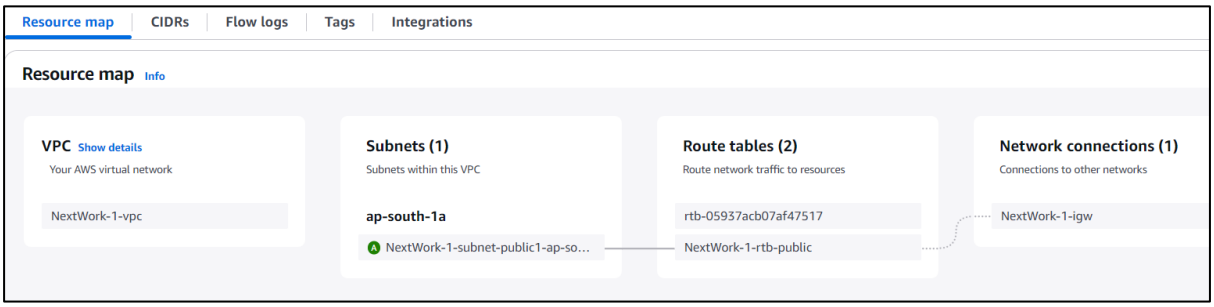


- Next, for the **VPC endpoints** option, select **None.**
- You can leave the **DNS options** checked.



- Select **Create VPC.**

- Select **View VPC**.
- Select the **Resource map** tab - nice, all of these resources have been set up for you in a flash!

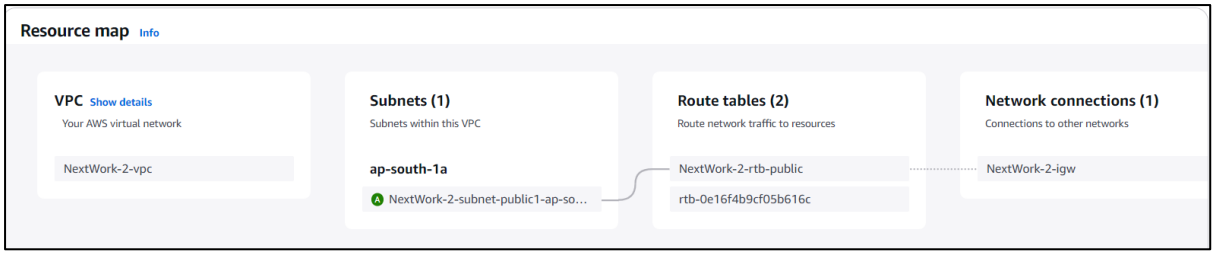


Set up VPC 2

Challenge yourself - can you set up your second VPC without any guidance?

Your second VPC has the exact same settings, except:

- Under **Name tag auto-generation**, enter NextWork-2
- The VPC's **IPv4 CIDR block** should be unique! Make sure the CIDR block is 10.2.0.0/16 - NOT 10.1.0.0/16.
- Follow the rest of the steps as similar as above used to setup VPC1.



Launch EC2 Instances

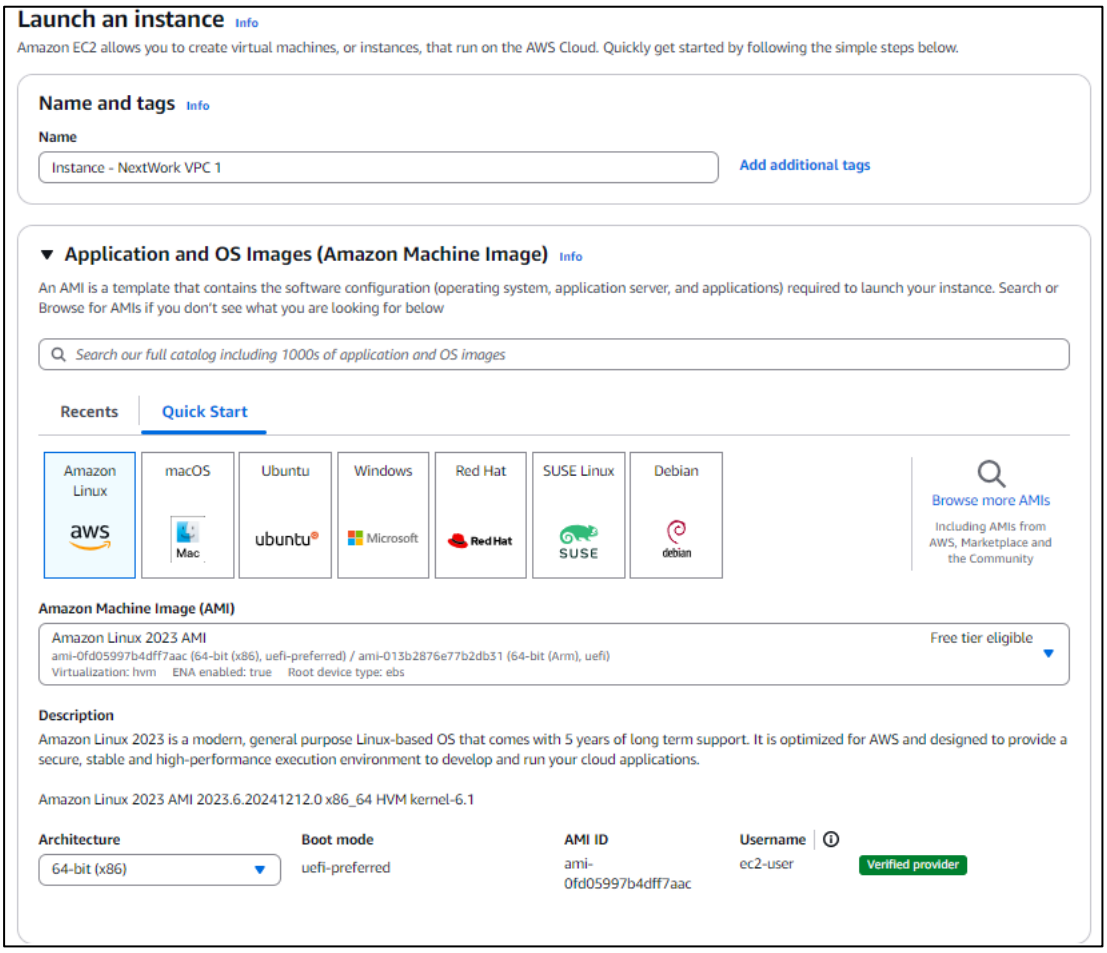
We need to create these EC2 instances so that they can send data to each other later in the project, which gives us network activity to monitor.

In this step, you're going to:

1. Launch an EC2 instance in each VPC, so we can use them to test your VPC peering connection later.

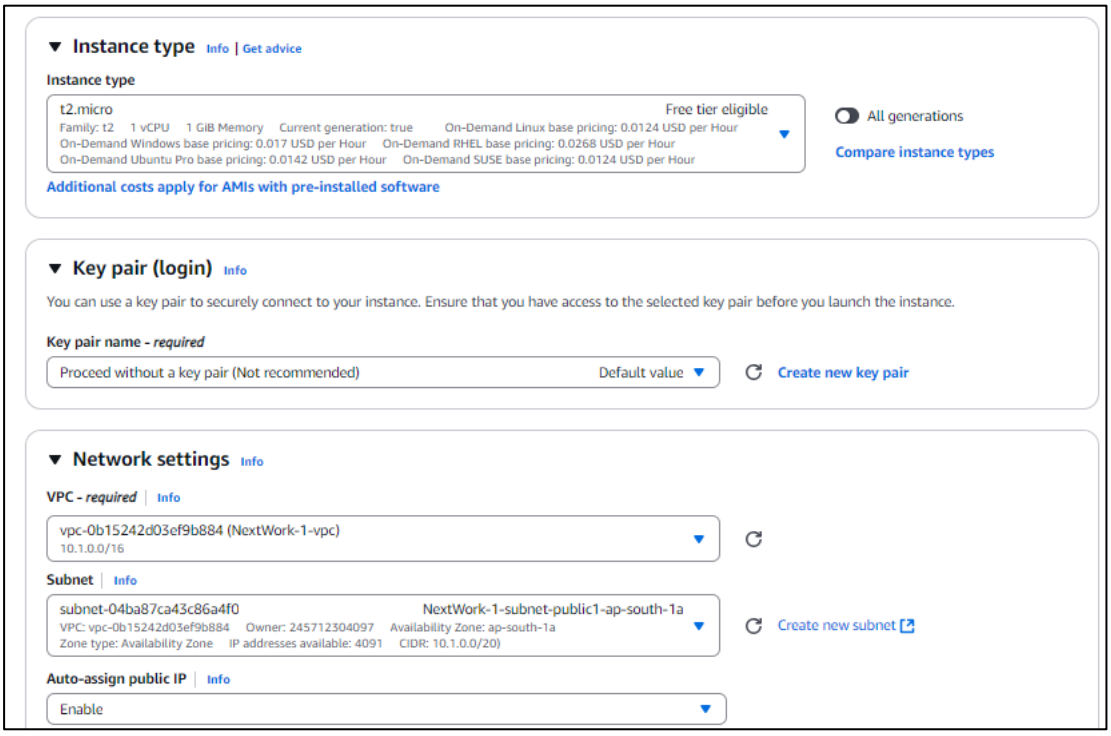
Launch an instance in VPC 1

- Head to the **EC2 console** - search for EC2 in the search bar at the top of screen.
- Select **Instances** at the left hand navigation bar.
- Select **Launch instances**.
- Since your first EC2 instance will be launched in your first VPC, let's name it Instance - NextWork VPC 1
- For the **Amazon Machine Image,** select **Amazon Linux 2023 AMI.**



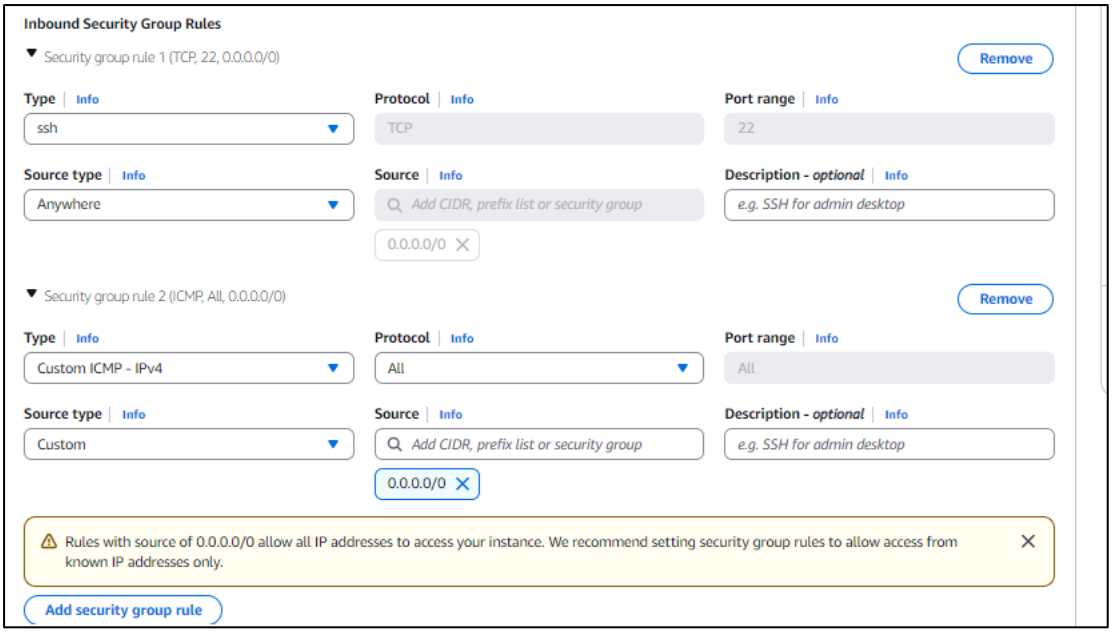
- For the **Instance type,** select **t2.micro.**
- For the **Key pair (login)** panel, select **Proceed without a key pair (not recommended).**
- At the **Network settings** panel, select **Edit** at the right hand corner.

- Under **VPC**, select **NextWork-vpc-1**.
- Under **Subnet,** select your VPC's public subnet.
- Keep the **Auto-assign public IP** setting
- Select **Enable.**



- For the **Firewall (security groups)** setting, choose **Create security group.**
- Name your security group NextWork-1-SG

- Choose **Add security group rule.**
- For the new rule's **Type**, select **All ICMP - IPv4.**
- For the new rule's **Source**, select 0.0.0.0/0
- Select **Launch instance.**

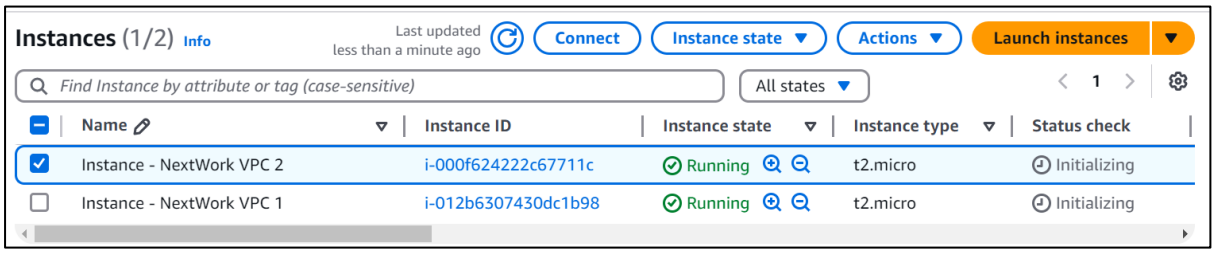


Launch an instance in VPC 2

set up an EC2 instance in VPC 2 smilarly as you did for VPC1

Follow the same instructions as above but make sure:

- The **Name** is Instance - NextWork VPC 2
- The **VPC** is **NextWork-vpc-2**.
- Make sure you select **Enable** for **Auto-assign public IP** here too
- Name your security group NextWork-2-SG
- Allow ICMP traffic from ALL IP addresses.



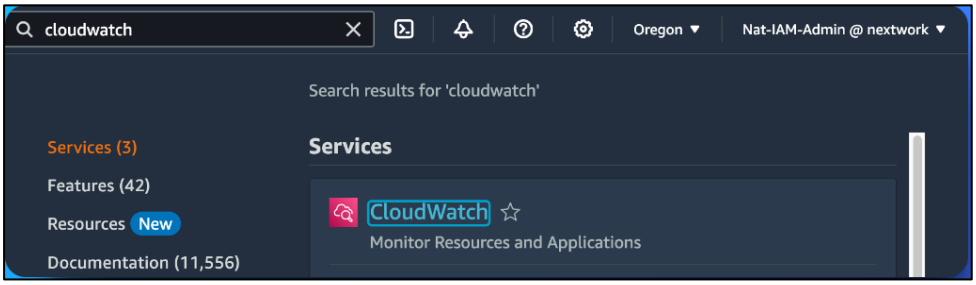
Set Up Flow Logs

- EC2 instances LAUNCHED
- Next up, we have to start monitoring VPC traffic.
- We're using a tool called VPC flow logs to set it up.

In this step, you're going to:

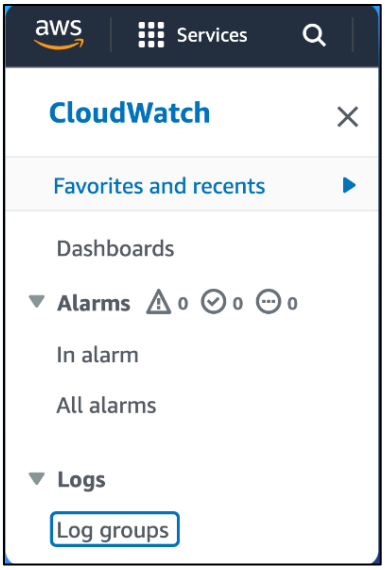
1. Set up a way to track all inbound and outbound network traffic.
2. Set up a space that stores all of these records.

- Navigate to the **CloudWatch console** - search for CloudWatch in your Management Console's search bar.

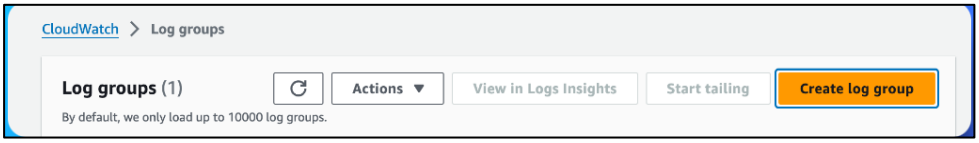


- Check the **Region** you're on - is this the same Region as the one you've used to launch your VPCs?

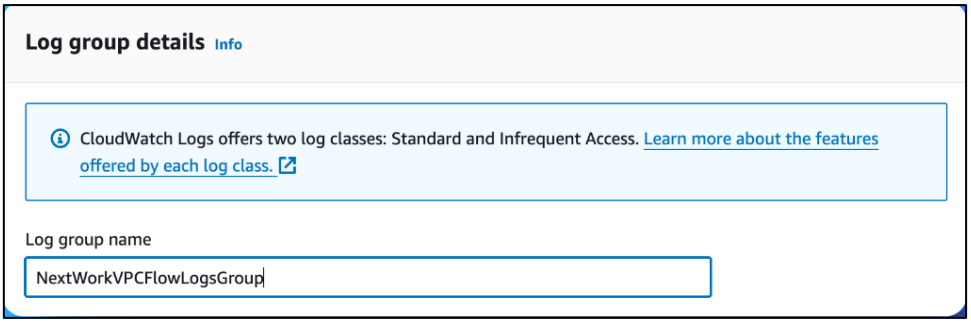
- Double check your Region by looking at the top right hand corner of your console. Your Region is right next to your account name!
- At the left-hand navigation panel, click **Log groups** under **Logs.**



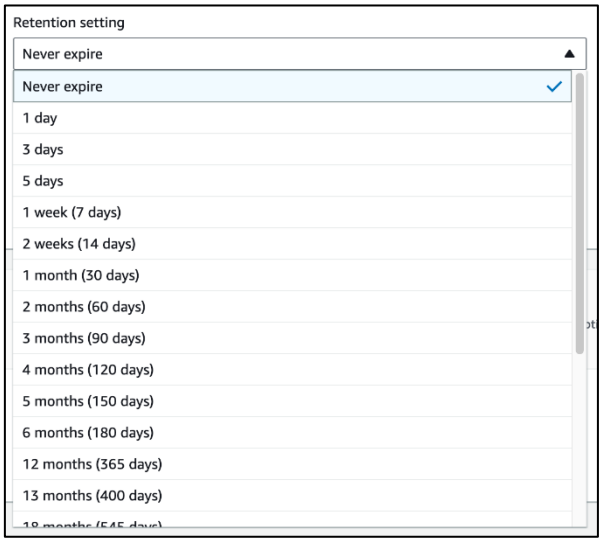
- Click **Create log group** at the top right.



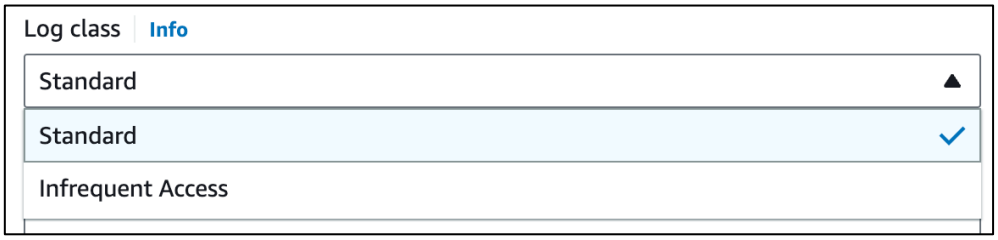
- Enter **NextWorkVPCFlowLogsGroup** as the **Log group name.**

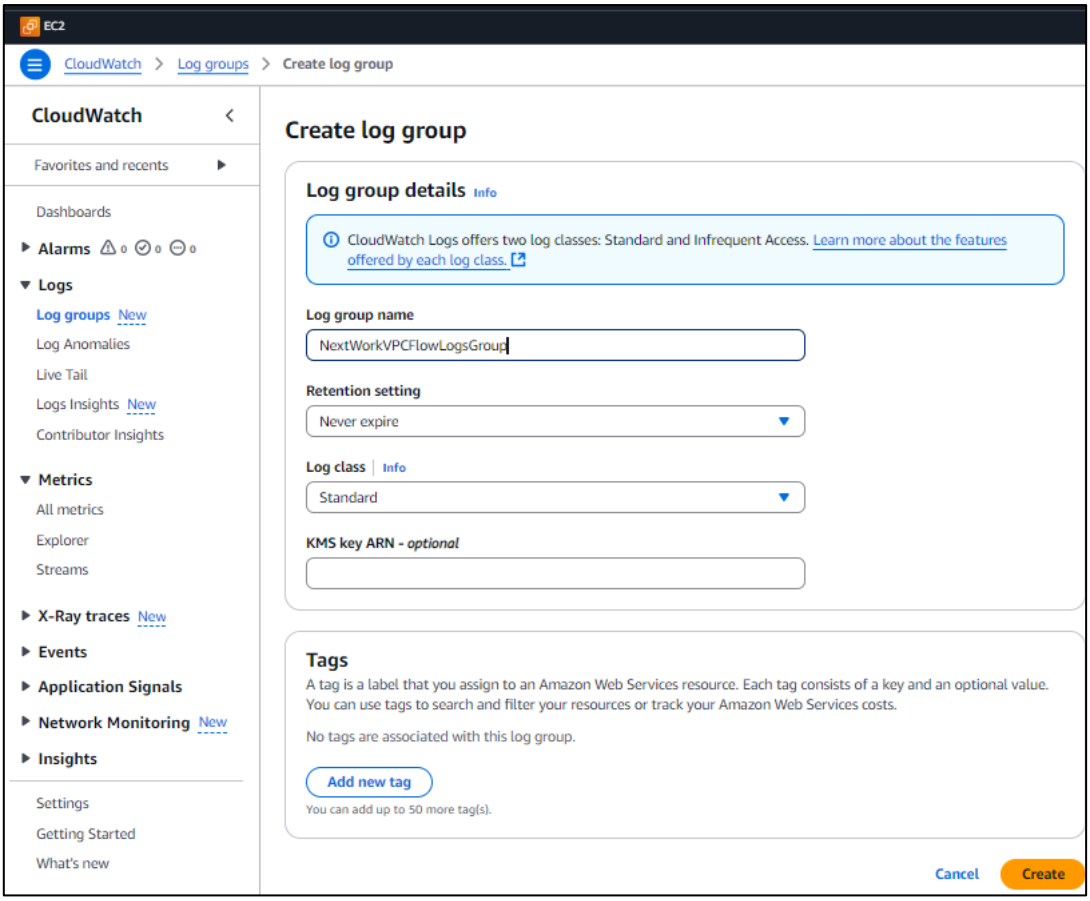


- **Retention setting** is **Never expire** by default, which means your logs won't be deleted over time. They'll stick around as long as you need them, unless you decide to clear them out yourself.



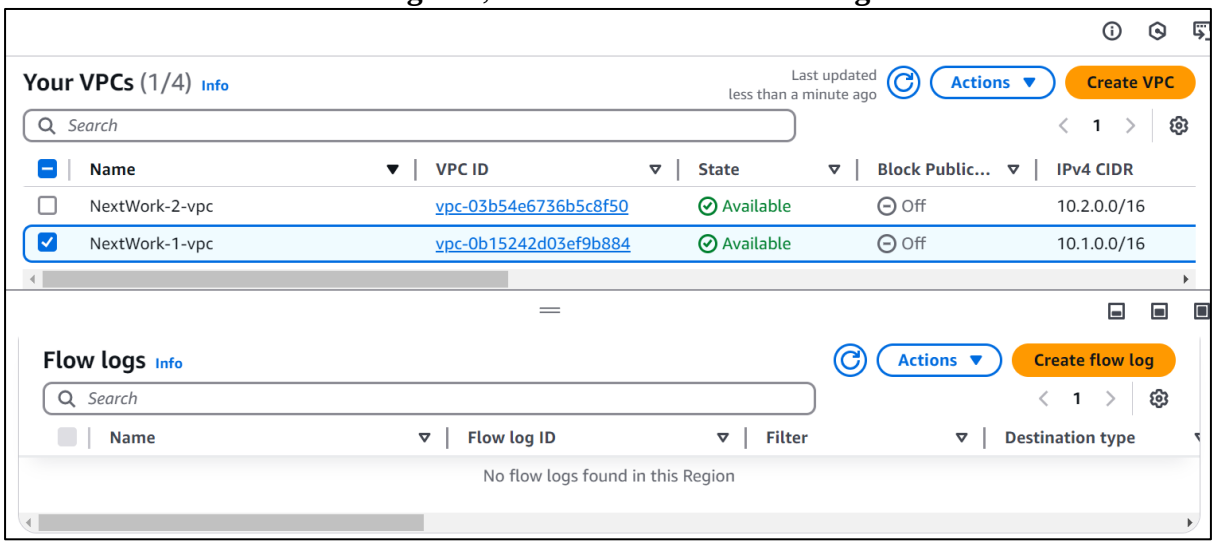
- **Log class** is **Standard** by default, which means the logs that get created will get accessed or analyzed regularly.
  If we chose **Infrequent Access** instead, your logs will be stored for long-term archiving - you are charged less for storage, but higher for each time you need to access the, for analysis. This setting isn't quite important since our usage will fall under the Free Tier!





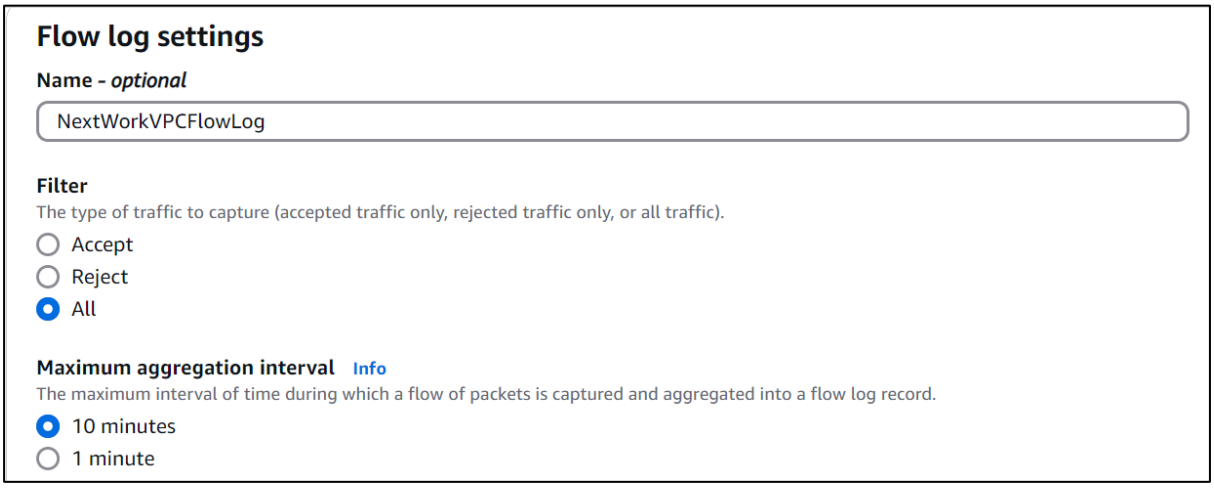
- Click **Create.**

- Head back to your **VPC** console.
- Select the **Your VPCs** page.
- Select **NextWork-1-vpc.**
- Scroll down to the **Flow Logs** tab, and click on **Create flow log.**

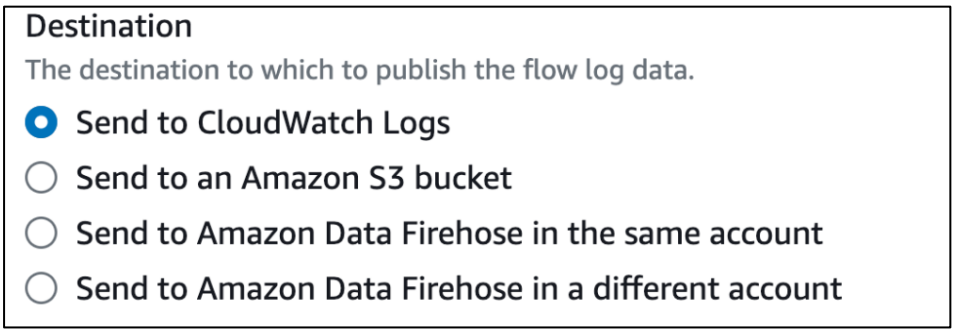


- Welcome to the **Flow log settings** page! Nice, you've just unlocked a new section of VPC set up.
- Enter NextWorkVPCFlowLog in the **Name** field.

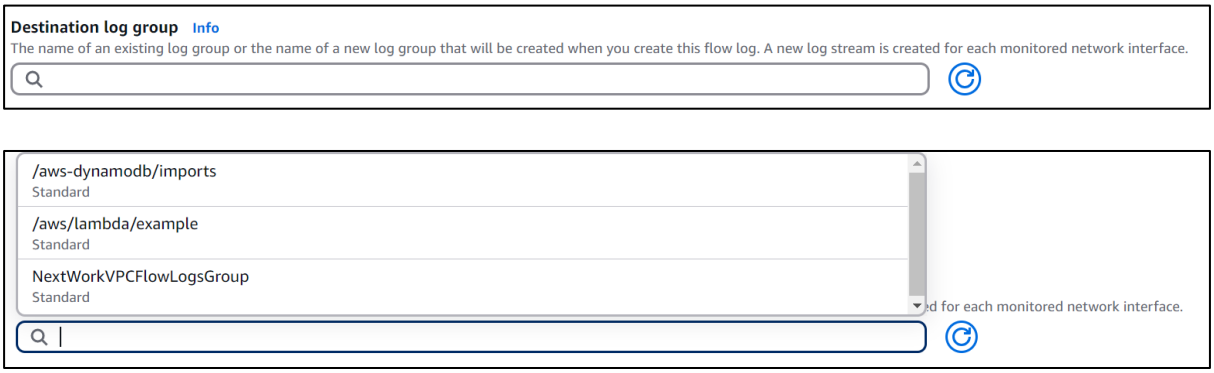
- Set **Filter** to **All.**
- Set the **Maximum aggregation interval** to **1 minute**



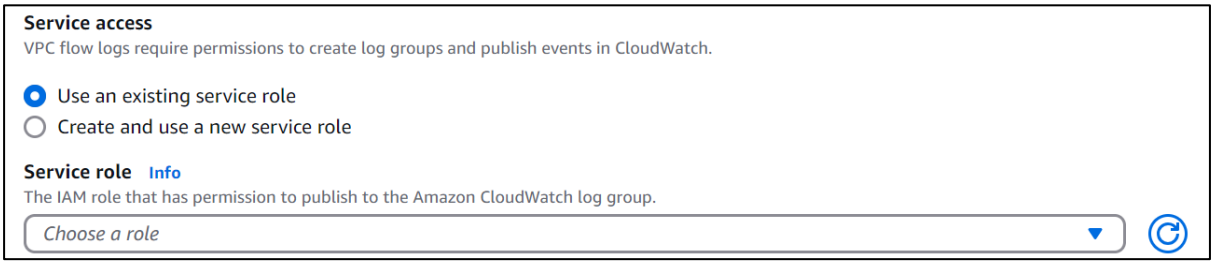
- Leave **Destination** as **Send to CloudWatch Logs.**

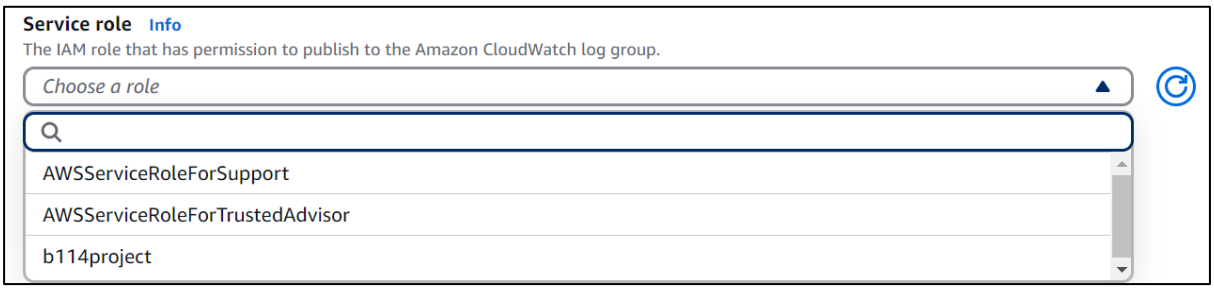


- Set **Destination log group** as **NextWorkVPCFlowLogsGroup.**



- Under the **Service role**, you might notice that there isn't an IAM role that's designed for Flow Logs!





So, we have to setup an IAM Role for our Flow Logs.

Let's set up an IAM role for your Flow Logs!

Set Up A Flow Log IAM Policy and Role

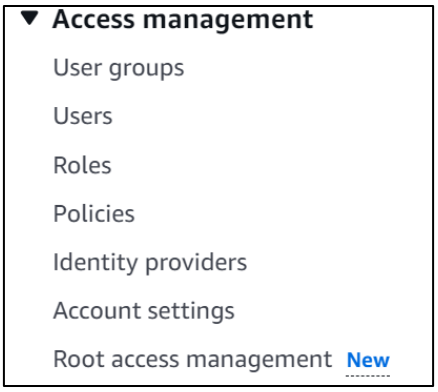
VPC Flow Logs doesn't have the permission to write logs and send them to CloudWatch... yet.

Let's give Flow Logs the permission to do both, using the power of IAM roles and policies!

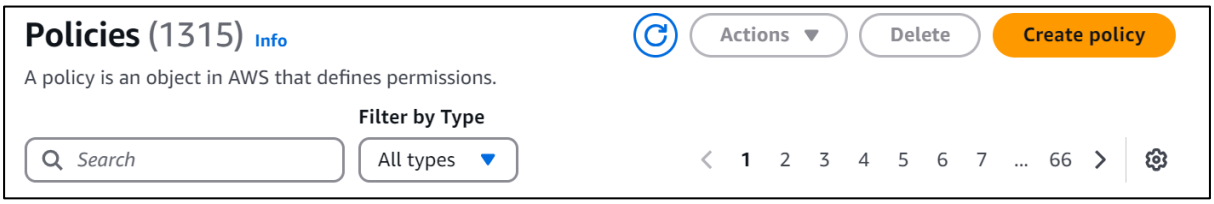
In this step, you're going to:

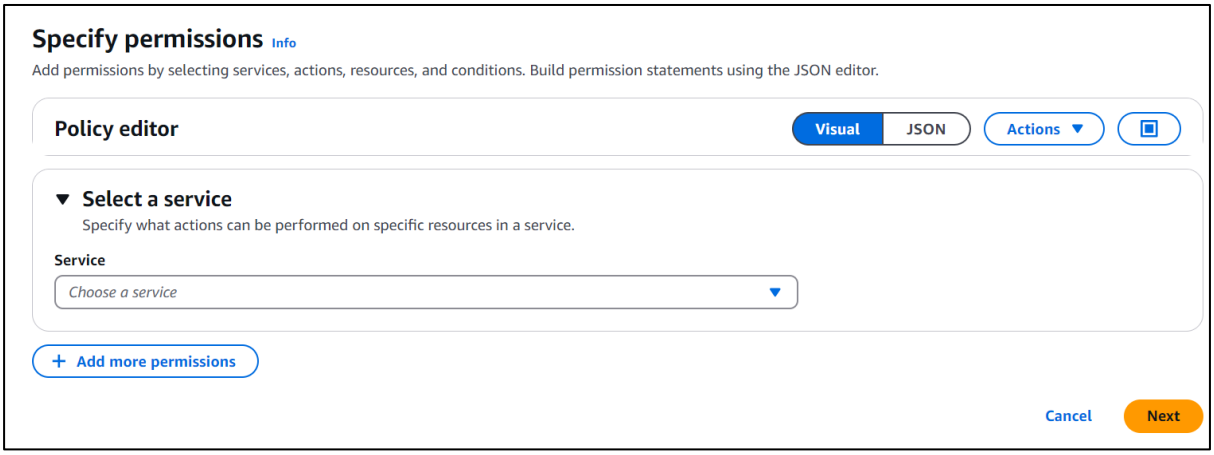
1. Give VPC Flow Logs the permission to write logs and send them to CloudWatch.
2. Finish setting up your subnet's flow log.

- Navigate to IAM Dashboard in a new tab.
- In the navigation pane, choose **Policies**.



- Choose **Create policy**.



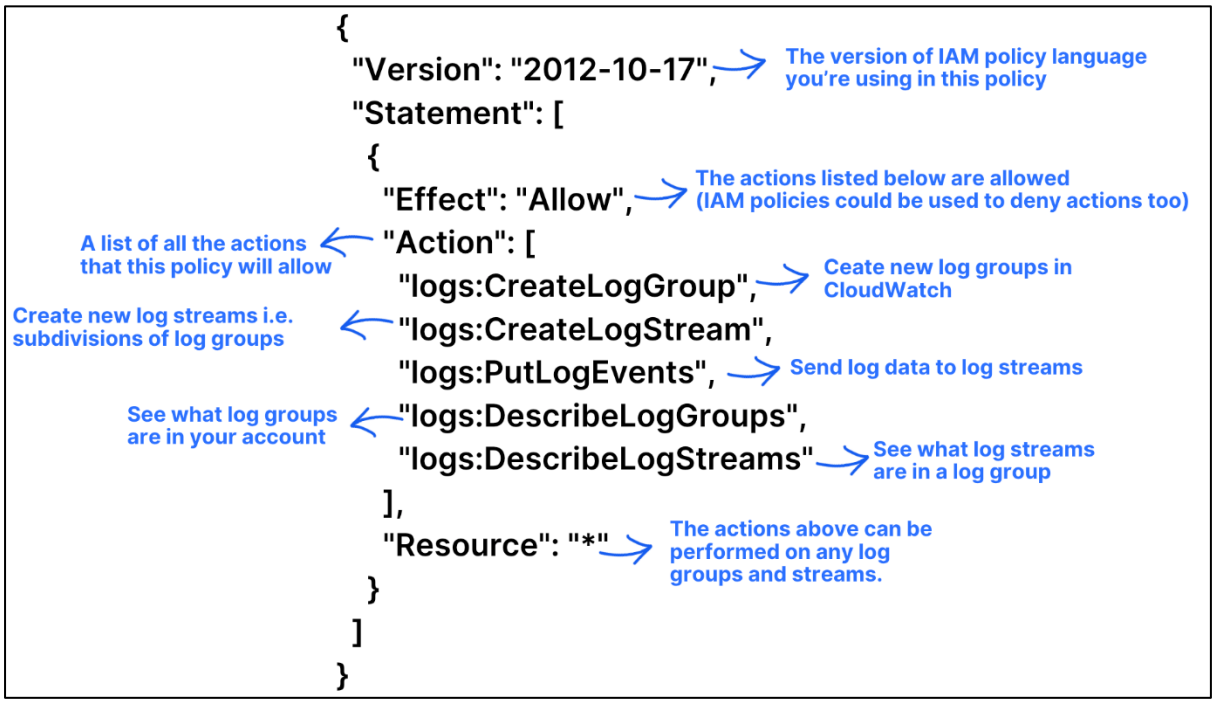


- Choose **JSON**.
- Delete everything in the **Policy editor.**
- Add this JSON policy to the empty Policy editor:

```
{
 "Version": "2012-10-17",
 "Statement": [
   {
    "Effect": "Allow",
    "Action": [
     "logs:CreateLogGroup",
     "logs:CreateLogStream",
     "logs:PutLogEvents",
     "logs:DescribeLogGroups",
     "logs:DescribeLogStreams"
    ],

    "Resource": "*"

   }

 ]

}
```

Why are we creating this policy? What does this policy say?
VPC Flow Logs by default don't have the permission to record logs and store them in your CloudWatch log group. This policy makes sure that your VPC can now send log data to your log group!

We can also break this statement down line by line:



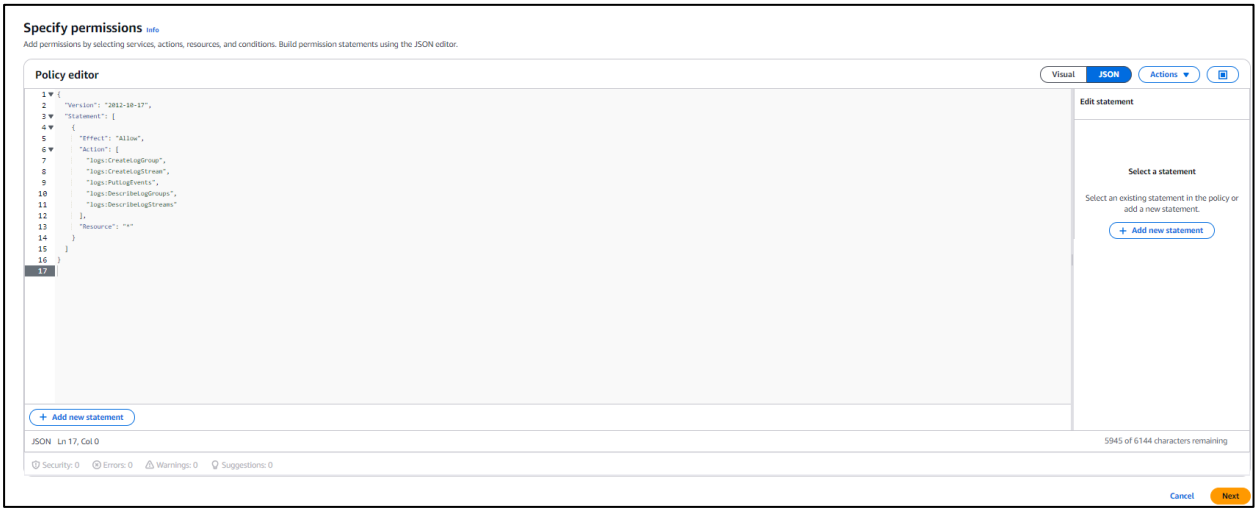
Version: "2012-10-17"
The version of IAM policy language that you're using. This is the latest version of IAM policy language, older policies use 2008-10-17!
Statement:
This section contains the set of permissions that make up this policy.
Effect: "Allow"
This line states that the actions you're about to list below are allowed. This line is quite powerful - if you change "Allow" to "Deny", this policy would immediately block Flow Logs from creating and sending logs!

Action:
This section lists specific actions that are allowed under this policy:

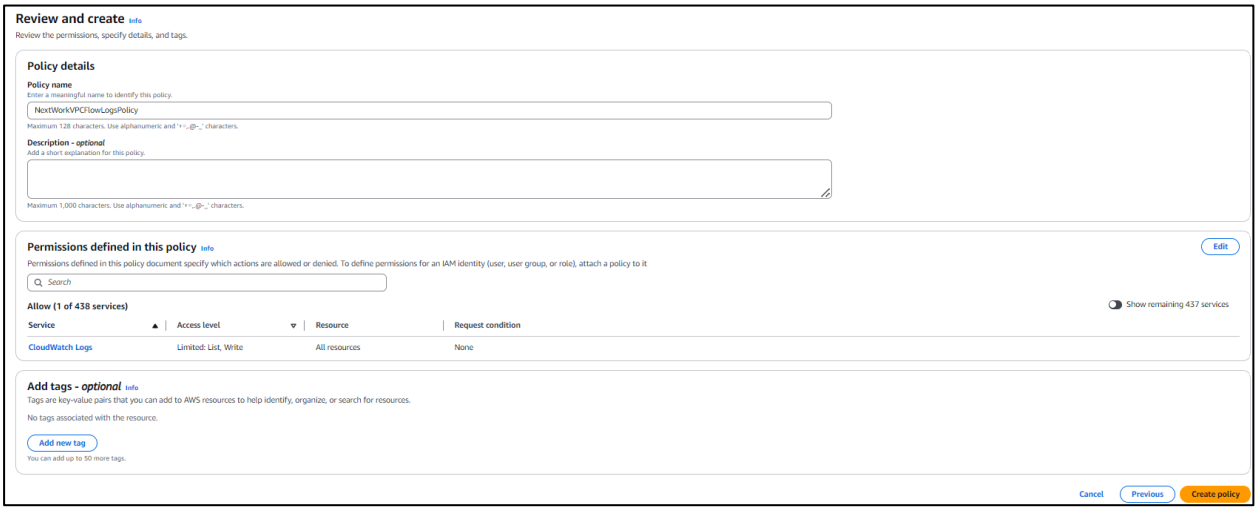
- **logs:CreateLogGroup**: Allows the IAM role to create new log groups in CloudWatch.
- **logs:CreateLogStream**: Allows the IAM role to create log streams within those groups.
- **logs:PutLogEvents**: Allows the IAM role to send log data to the streams.
- **logs:DescribeLogGroups** and **logs:DescribeLogStreams**: These actions allow the role to see information about the log groups and streams.

Resource: "*"
This specifies that the permissions apply to all log groups and streams.

- Choose **Next**.



- For your policy's name, let's call it NextWorkVPCFlowLogsPolicy



- Choose **Create policy**.

IAM policy done!

Can we head back to your Flow Logs set up now?

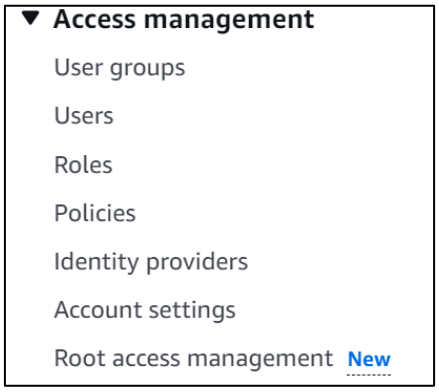
Not quite...

Don't forget that your Flow Logs set up is asking for an IAM **role**.

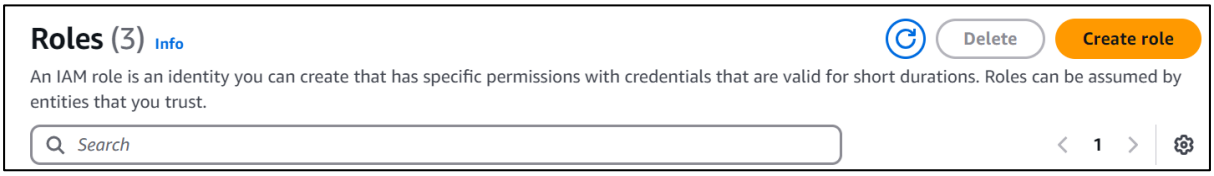
What's the difference between an IAM policy and role? Why did we need to create the policy just now?
IAM policies and roles go hand in hand, so it's important to know their differences and how they work together!

1. IAM policies are like rules that determine what someone/something can or cannot do in your AWS account.
2. When you bundle IAM policies together, you create an IAM role. You then assign this role to AWS services or users, giving them the permissions included in the attached policies.

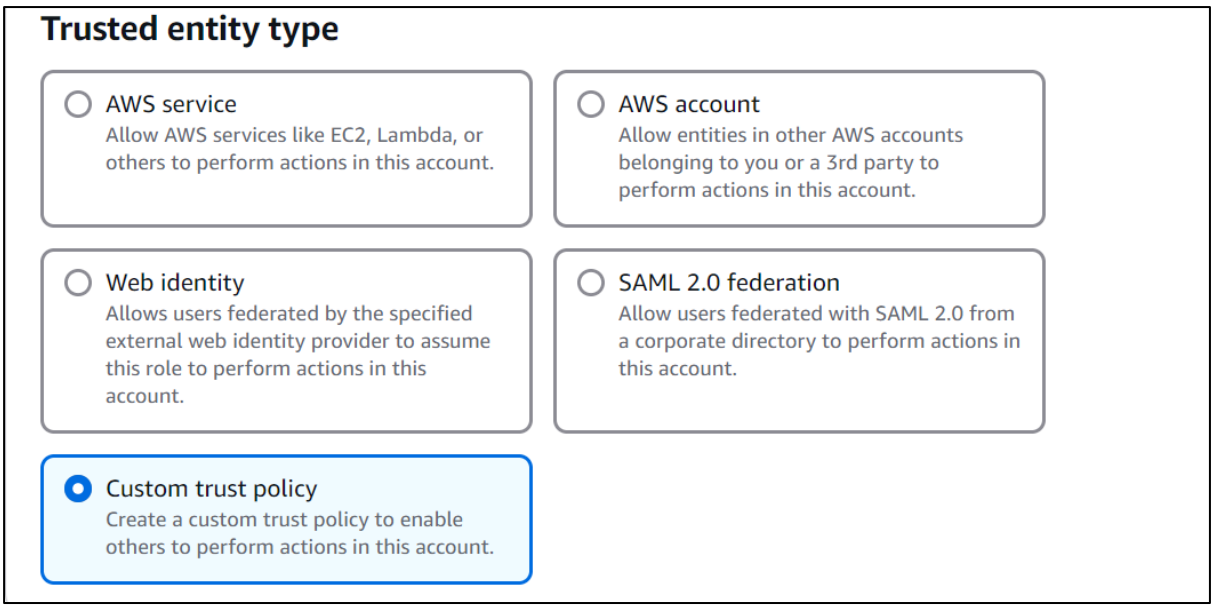
- In the left hand navigation pane of IAM, choose **Roles**.



- Choose **Create role**.



- For **Trusted entity type**, choose **Custom trust policy**.



What is a custom trust policy

A custom trust policy is specific type of policy! They're different from IAM policies. While IAM policies help you define the actions a user/service can or cannot do, custom trust policies are used to very narrowly define who can use a role.

Here's another way to think about it: using a custom trust policy is like using a special VIP list - only the services you pinpoint in your policy would be allowed to use your role.

Why did we pick Custom trust policy here?

Don't forget why we're creating an IAM role - to give VPC Flow Logs the permission to write and send logs to CloudWatch. We only want Flow Logs to have this access, not just any service.

By choosing a custom trust policy, we're making sure that only VPC Flow Logs can use this role.

- A custom trust policy panel will be found as you scroll down.

- Replace "Principal": {}, in the above policy statement with the following:

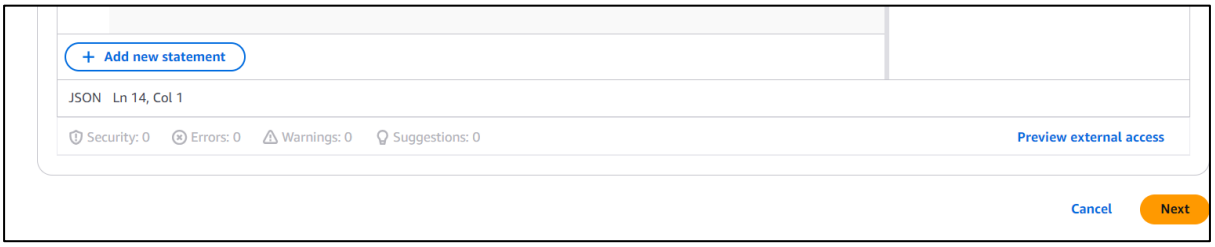
```
"Principal": {
  "Service": "vpc-flow-logs.amazonaws.com"
},
```

What does this statement mean?
The statement "Service": " vpc-flow-logs.amazonaws.com " in a trust policy specifically points to VPC Flow Logs as the only service that can use this role!

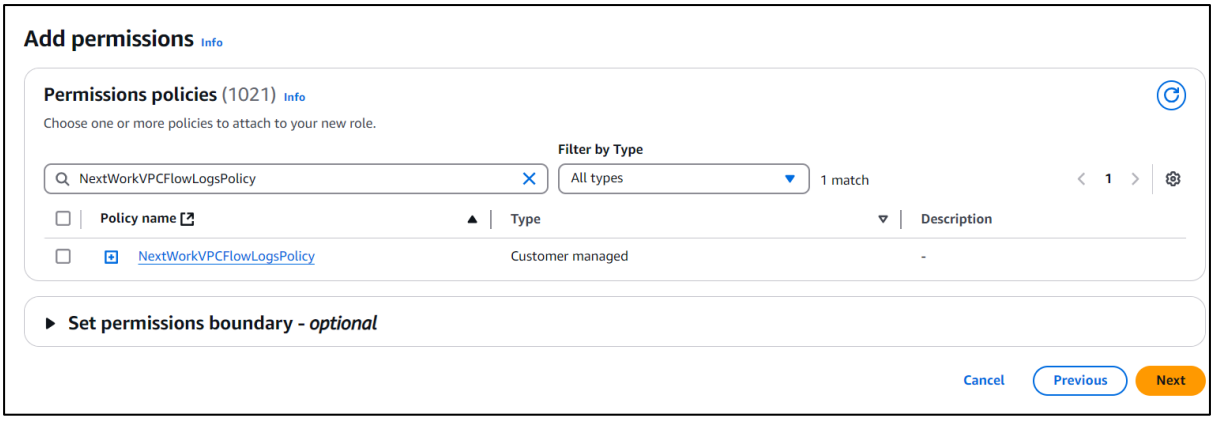
Even if you try to give this role to other AWS services, they can't use it because the permissions are locked down to just VPC Flow Logs. This is so good for security, in case this role gets accidentally assigned to another service/user.

Hot tip: "Principal" defines the entity that is given the permissions in this policy. In this case, the entity is a service (VPC Flow Logs), but other entity types are IAM Users and IAM roles!

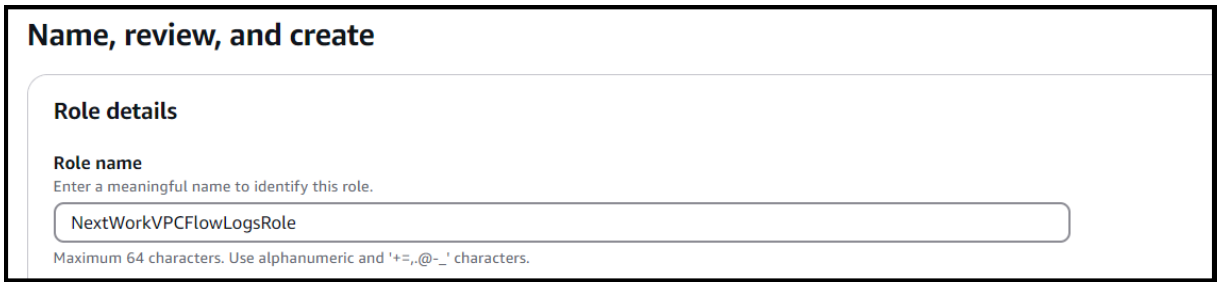
- Scroll Down and Choose **Next**.



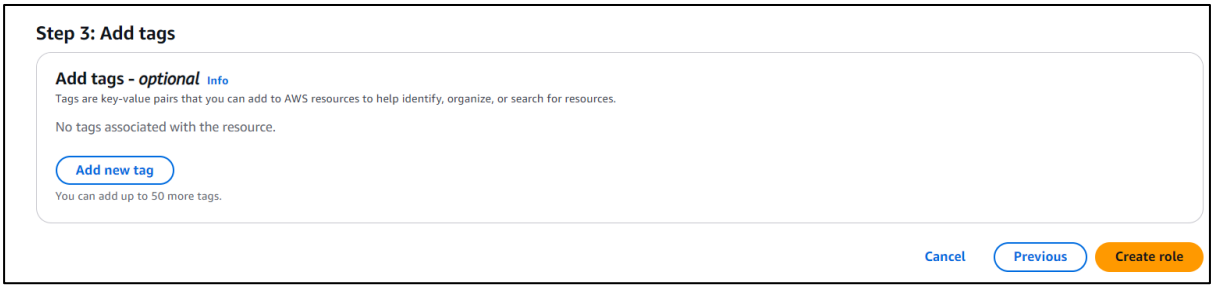
- On the **Add permissions** page, search for the policy you've created - NextWorkVPCFlowLogsPolicy

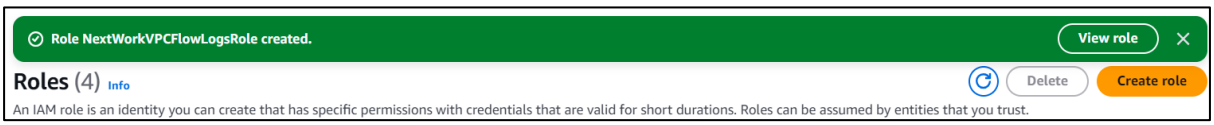


- Select your policy.
- Choose **Next**.
- Enter a name for your role - NextWorkVPCFlowLogsRole.



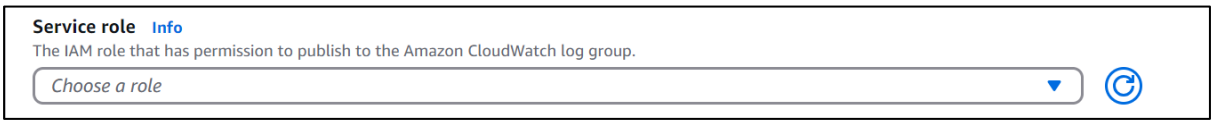
- Scroll Down and Choose **Create role**.





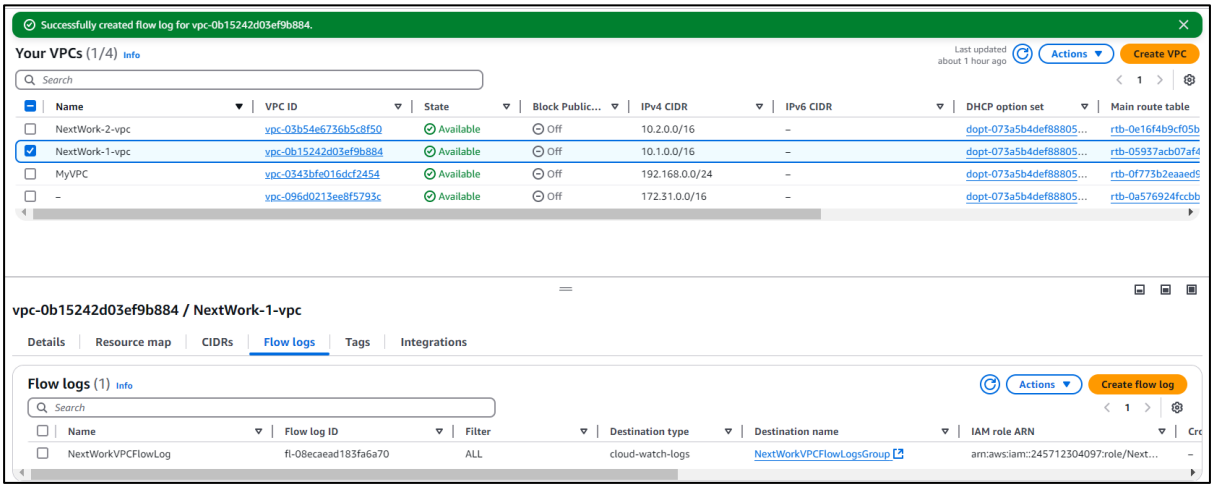
IAM role set up DONE!

- Head back to your VP console's **Create flow log** page.
- Scroll to **Service Role** and hit the refresh button on the right side.



- Now when u click on *Choose a role*, the New IAM Role (**NextWorkVPCFlowLogsRole**) you created will be visible

- Select your IAM role - **NextWorkVPCFlowLogsRole**.
- Scroll Down and Click on **Create flow log.**



The flow log is all set up! This means network traffic going into and out of your VPC is **now getting tracked**

Test VPC Peering

Now that the flow log set up is all done!

Let's generate some network traffic and see whether our flow logs can pick up on them.

We're going to generate network traffic by trying to get our instance in VPC 1 to send a message to our instance in VPC 2.

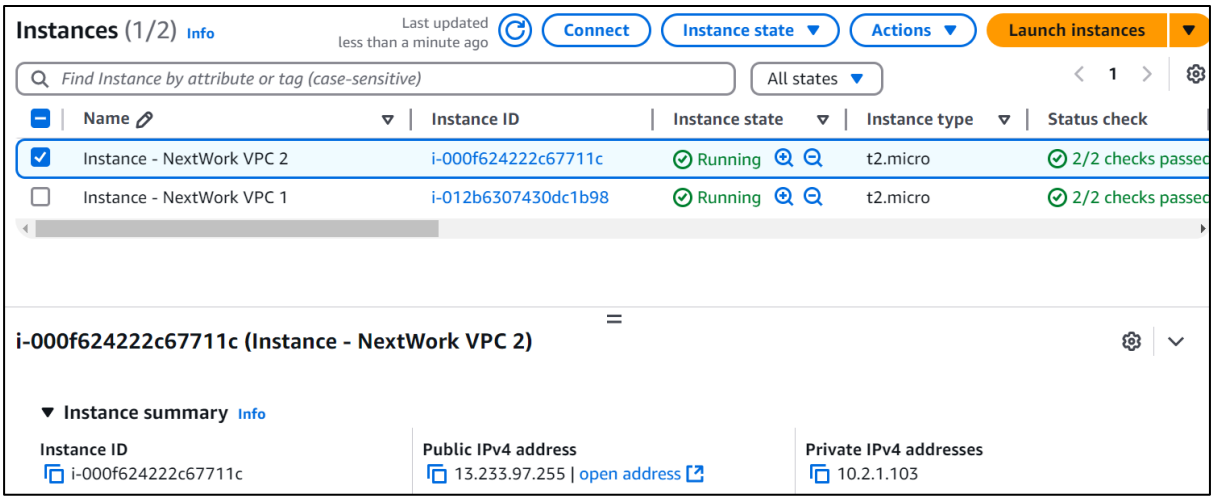
Since we're trying to get our instances to talk to each other, this means we're also testing our VPC peering setup at the same time!

In this step, you're going to:

1. Get Instance 1 to send test messages to Instance 2.

- Head to your **EC2** console and the **Instances** page.
- Select the checkbox next to **Instance - NextWork VPC 1.**
- Select **Connect**.
- In the EC2 Instance Connect set up page, select **Connect** again.

- Success.
- Leave open the **EC2 Instance Connect** tab, but head back to your **EC2** console in a new tab.
- Select **Instance - NextWork VPC 2.**
- Copy Instance - NextWork VPC 2's **Private IPv4 address.**



- Switch back to the **EC2 Instance Connect** tab.
- Run ping [the Private IPv4 address you just copied] in the terminal.
  - Your final result should look similar to something like ping 10.2.1.103 (as per my Private IPv4 address)
- You should see a response similar to this:

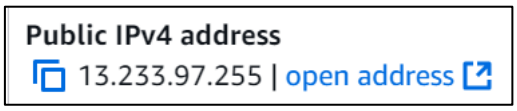
```
[ec2-user@ip-10-1-8-247 ~]$ ping 10.2.1.103
PING 10.2.1.103 (10.2.1.103) 56(84) bytes of data.
```

- We are not getting any replies back right now.

- Let's do some investigating... press **Ctrl + C** on your keyboard to end this ping test.

```
^C
--- 10.2.1.103 ping statistics ---
140 packets transmitted, 0 received, 100% packet loss, time 144548ms
```

- See if you can test the connection from VPC 1 to VPC 2's **public** IP address.
- Head back to your EC2 console, and copy the **Public IPv4 address** of **Instance - NextWork VPC 2.**



- Head back to your EC2 Instance Connect tab and run a ping test with this public IPv4 address.
- Your final result should look similar to something like ping [public IPv4 address], in my case: ping 13.233.97.255

```
[ec2-user@ip-10-1-8-247 ~]$ ping 13.233.97.255
PING 13.233.97.255 (13.233.97.255) 56(84) bytes of data.
64 bytes from 13.233.97.255: icmp_seq=1 ttl=126 time=0.589 ms
64 bytes from 13.233.97.255: icmp_seq=2 ttl=126 time=0.459 ms
64 bytes from 13.233.97.255: icmp_seq=3 ttl=126 time=0.621 ms
64 bytes from 13.233.97.255: icmp_seq=4 ttl=126 time=0.615 ms
64 bytes from 13.233.97.255: icmp_seq=5 ttl=126 time=0.567 ms
64 bytes from 13.233.97.255: icmp_seq=6 ttl=126 time=0.636 ms
64 bytes from 13.233.97.255: icmp_seq=7 ttl=126 time=0.569 ms
64 bytes from 13.233.97.255: icmp_seq=8 ttl=126 time=0.561 ms
64 bytes from 13.233.97.255: icmp_seq=9 ttl=126 time=0.450 ms
64 bytes from 13.233.97.255: icmp_seq=10 ttl=126 time=0.518 ms
64 bytes from 13.233.97.255: icmp_seq=11 ttl=126 time=0.531 ms
```

- press **Ctrl + C** on your keyboard again to end this ping test.
- Ping your EC2 Instance 2's **private** address again.

```
[ec2-user@ip-10-1-8-247 ~]$ ping 10.2.1.103
PING 10.2.1.103 (10.2.1.103) 56(84) bytes of data.
```

- Still now replies.

  We receive ping replies when we use Instance 2's **public** IP address, whicih confirms that VPC 2's security groups and NACL's *are* letting in ICMP traffic.

  But using Instance 2's **private** IP address doesn't give us any ping replies.

- Leave open the **EC2 Instance Connect** tab, but head back to your **VPC** console in a new tab.
- In the VPC console, select the **Subnets** page.

- Select VPC 1's subnet i.e. **NextWork-1-subnet-public1-...**
- Let's investigate the **Route tables** and **Network ACL** tabs for your subnet.
- The network ACL allows all types of inbound traffic from anywhere! So this looks perfectly fine.
- But let's take a closer look at the route tables...

What's the issue with this route table?
where is the direct route to VPC 2 from VPC 1? Can you find it in this route table?

Nope! The missing ingredient in our architecture is the **VPC peering connection** that directly connects VPCs 1 and 2.

But there's a route to 0.0.0.0/0 in the route table! Doesn't that get traffic anywhere, including Instance 2?
The answer lies in the purpose of a peering connection - why do we set one up?

The purpose of a peering connection is to create a **direct** link between two resources so they can communicate with their private IP addresses.

You'd be correct to say that Instance 1 and Instance 2 are currently connected through the route with a destination of 0.0.0.0/0, but that traffic is through the internet gateway i.e. traffic will travel through and be exposed to the public internet.

To make sure communication between Instances 1 and 2 is direct, we need to set up a new route that directs traffic to our peering connection (instead of the public internet).

Let's fix this by setting up our VPC peering connection.

Create a Peering Connection and Configure Route Tables

we have a missing link that's causing this connectivity error... did you catch it ahead of time that our network doesn't have a peering connection?

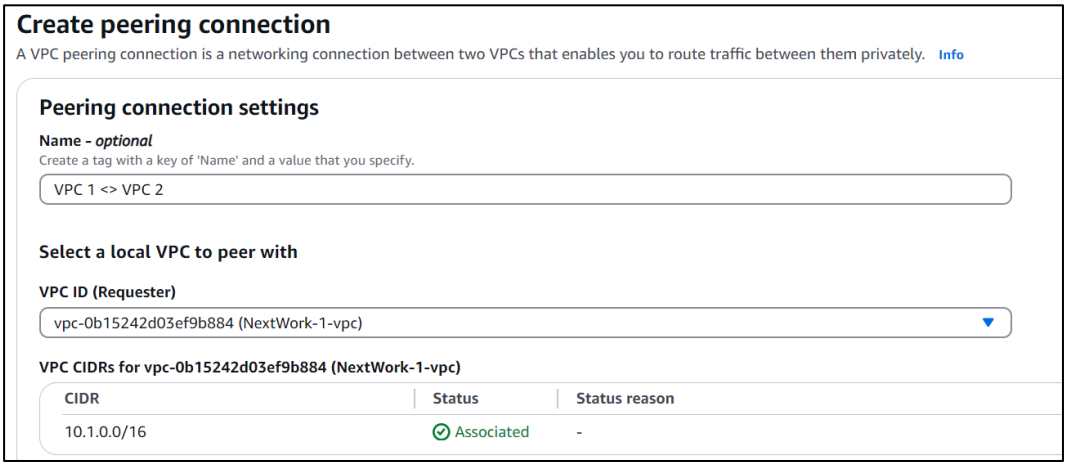
Let's add that peering connection in this step to bridge our VPCs together!

In this step, you're going to:

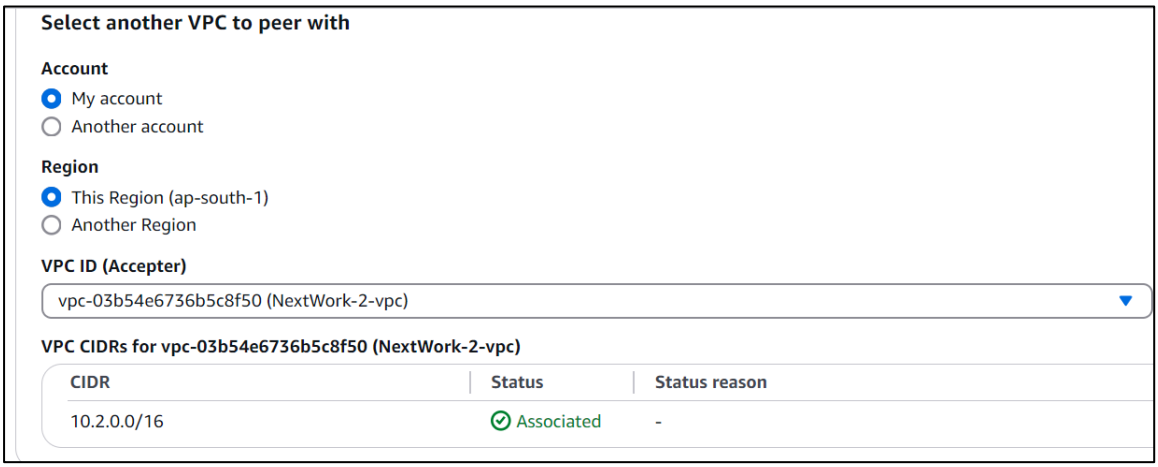
1. Set up a connection link between your VPCs.

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

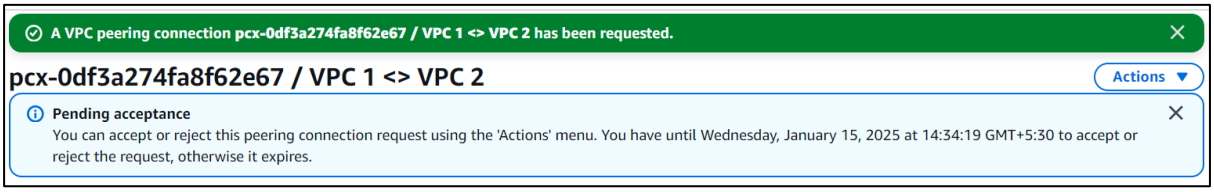
- Click on **Create peering connection** in the right hand corner.
- Name your **Peering connection name** as VPC 1 <> VPC 2
- Select **NextWork-1-VPC** for your **VPC ID (Requester).**



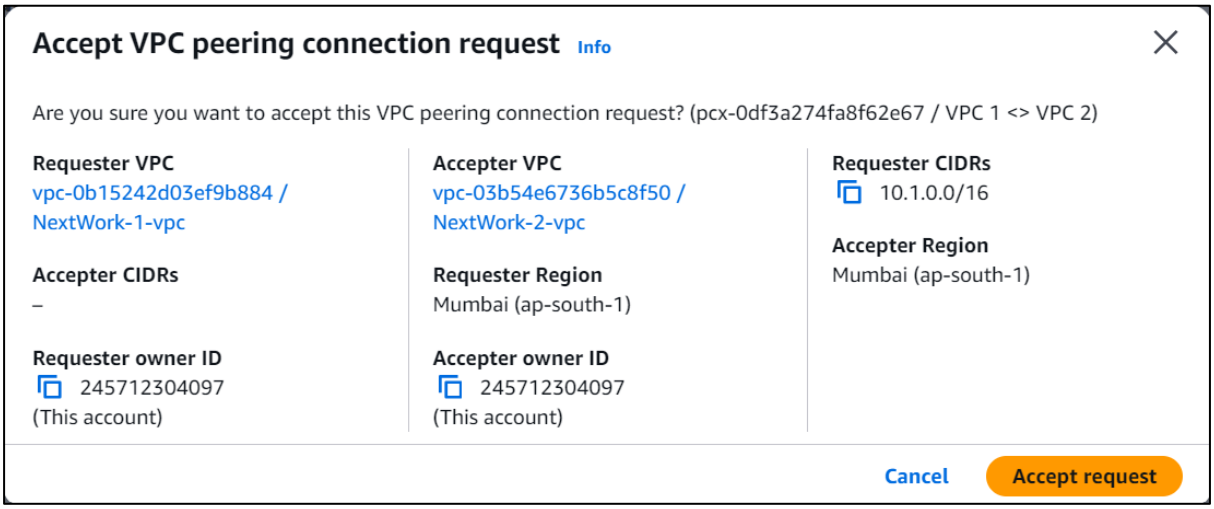
- Under **Select another VPC to peer with,** make sure **My Account** is selected.
- For **Region,** select **This Region.**
- For **VPC ID (Accepter)**, select **NextWork-2-VPC**



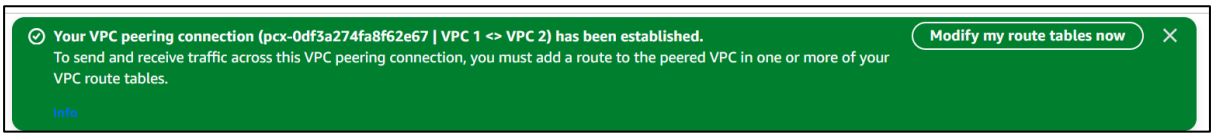
- Click on **Create peering connection.**
- Your newly created peering connection isn't finished yet! The green success bar says the peering connection **has been requested.**



- On the next screen, select **Actions** and then select **Accept request**

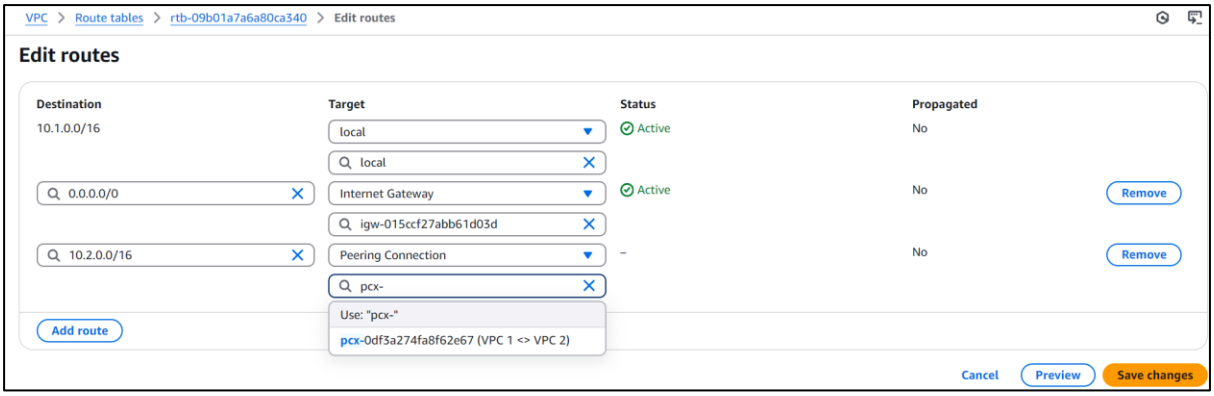


- Click on **Accept request** again on the pop up panel.
- Click on **Modify my route tables now** on the top right corner.

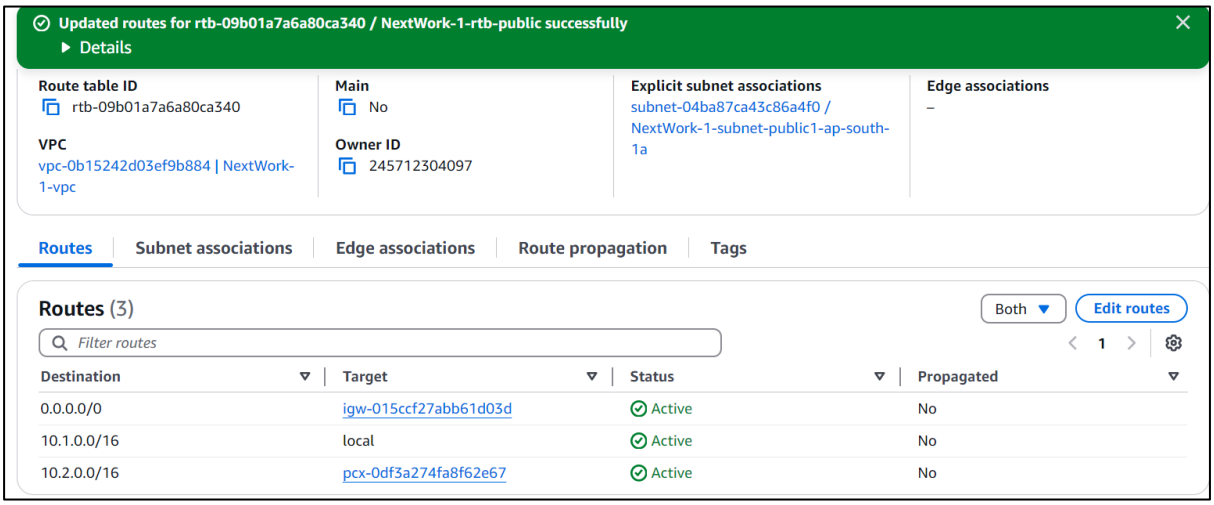


Update VPC 1's route table

- 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**.



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

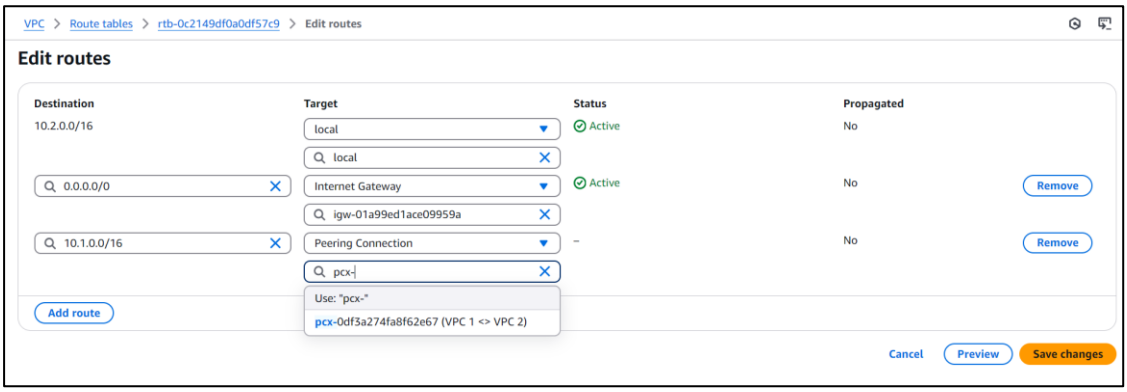


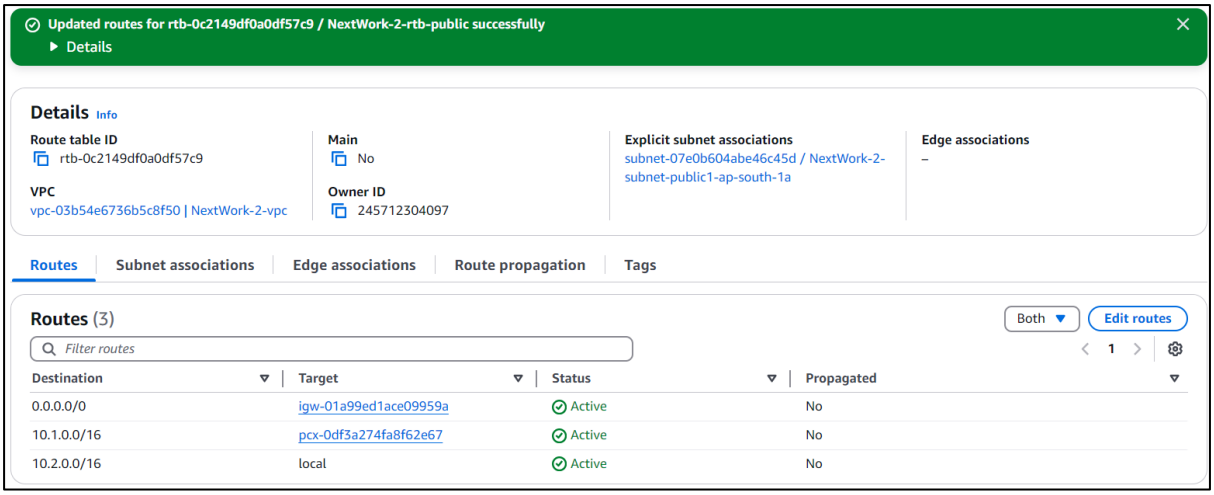
Update VPC 2's route table

set up the equivalent route in VPC 2's route table

If you get stuck, use the same instructions above but make sure:

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





- Revisit the **EC2 Instance Connect** tab that's connected to NextWork Public Server.
- Lots of new lines coming through in the terminal.

```
PING 10.2.1.103 (10.2.1.103) 56(84) bytes of data.
64 bytes from 10.2.1.103: icmp_seq=1284 ttl=127 time=0.477 ms
64 bytes from 10.2.1.103: icmp_seq=1285 ttl=127 time=0.465 ms
64 bytes from 10.2.1.103: icmp_seq=1286 ttl=127 time=0.422 ms
64 bytes from 10.2.1.103: icmp_seq=1287 ttl=127 time=0.511 ms
64 bytes from 10.2.1.103: icmp_seq=1288 ttl=127 time=0.502 ms
64 bytes from 10.2.1.103: icmp_seq=1289 ttl=127 time=0.468 ms
64 bytes from 10.2.1.103: icmp_seq=1290 ttl=127 time=0.456 ms
64 bytes from 10.2.1.103: icmp_seq=1291 ttl=127 time=0.534 ms
64 bytes from 10.2.1.103: icmp_seq=1292 ttl=127 time=0.453 ms
64 bytes from 10.2.1.103: icmp_seq=1293 ttl=127 time=0.549 ms
64 bytes from 10.2.1.103: icmp_seq=1294 ttl=127 time=0.584 ms
64 bytes from 10.2.1.103: icmp_seq=1295 ttl=127 time=0.522 ms
64 bytes from 10.2.1.103: icmp_seq=1296 ttl=127 time=0.509 ms
64 bytes from 10.2.1.103: icmp_seq=1297 ttl=127 time=0.470 ms
64 bytes from 10.2.1.103: icmp_seq=1298 ttl=127 time=0.416 ms
64 bytes from 10.2.1.103: icmp_seq=1299 ttl=127 time=0.449 ms
```

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

- 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 10.2.1.103 -c 5 (in my case)

```
[ec2-user@ip-10-1-8-247 ~]$ ping 10.2.1.103 -c 5
PING 10.2.1.103 (10.2.1.103) 56(84) bytes of data.
64 bytes from 10.2.1.103: icmp_seq=1 ttl=127 time=0.466 ms
64 bytes from 10.2.1.103: icmp_seq=2 ttl=127 time=0.531 ms
64 bytes from 10.2.1.103: icmp_seq=3 ttl=127 time=0.467 ms
64 bytes from 10.2.1.103: icmp_seq=4 ttl=127 time=1.06 ms
64 bytes from 10.2.1.103: icmp_seq=5 ttl=127 time=0.527 ms

--- 10.2.1.103 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4102ms
rtt min/avg/max/mdev = 0.466/0.609/1.055/0.224 ms
```

- The ping test automatically finishes after 5 packets have been sent.

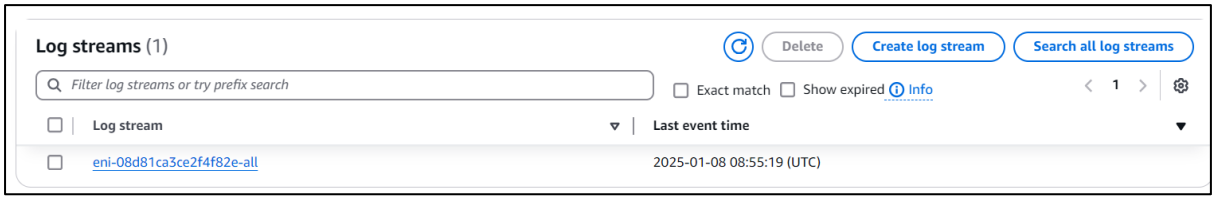
Analyze Flow Logs

To wrap things up, let's check out what VPC Flow Logs has recorded about your network's activity!

In this step, you're going to:

1. Review the flow logs recorded aboout VPC 1's public subnet.
2. Analyse the flow logs to get some tasty insights

- Head to your **CloudWatch** console.
- Select **Log groups** from the left hand navigation panel.
- Click into **NextWorkVPCFlowLogsGroup.**
- Click into your log stream to see flow logs from EC2 Instance 1!

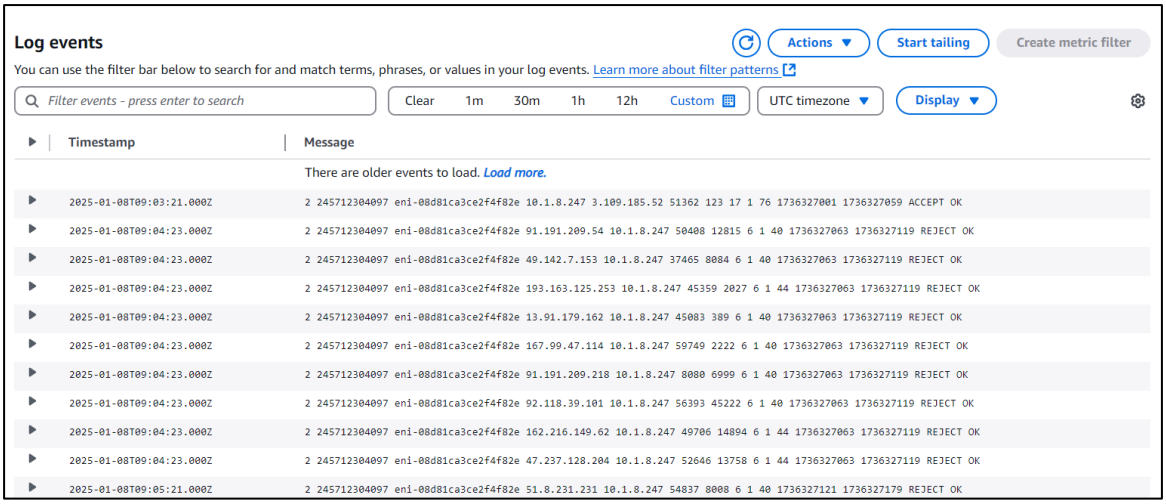


Why is my log stream *named* eni-xxx?
Log streams in CloudWatch are often named after the network interface ID (eni-xxx) when they're associated with VPC flow logs. This helps you organise which streams are tracking traffic to which resources in your VPC.

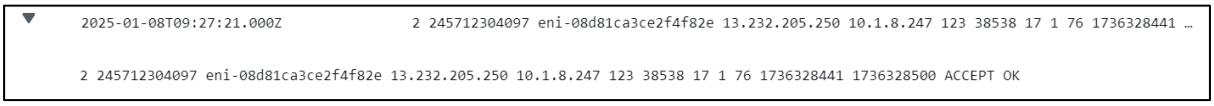
Tip: eni = Elastic Network Interface (ENI)! If you ever wonder how an EC2 instance can have a public/private IP address, security group rules and the option connect to other services in your AWS environment, the ENI is the answer.

Think of ENI as a cloud networking component that attaches to an EC2 instance and gives the EC2 instance networking capabilities. Every EC2 instance **must** have an ENI to exist and be in a VPC.

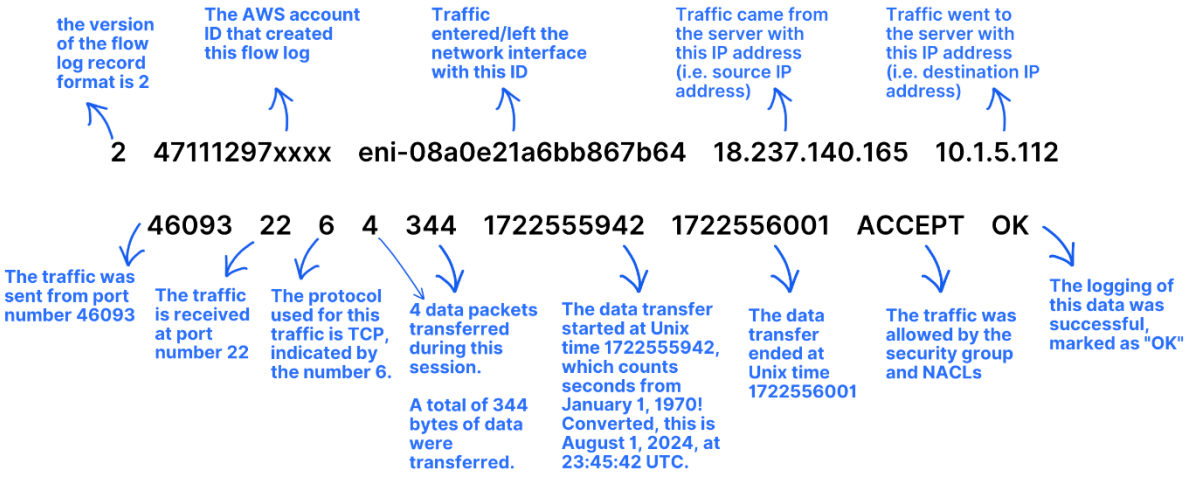
- check out these **log events!**



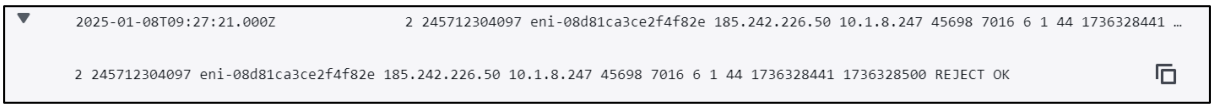
- Scroll to the very top and try expanding a log at the top welcome to this flow log.
- Expand one of your flow logs



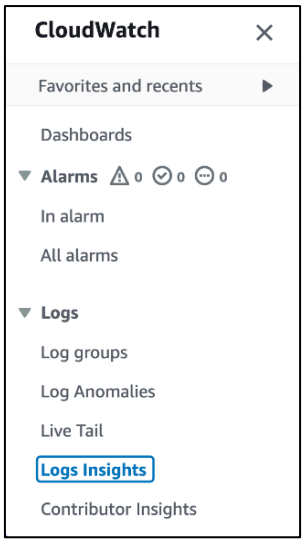
What is this flow log saying?
Let's break it down!



- Now scroll to the very bottom and select one of the newer logs, are there any differences?
- For example, you might find one that says **REJECT OK** instead of **ACCEPT OK** at the end. These would represent the ping messages that failed to reach Instance 2!

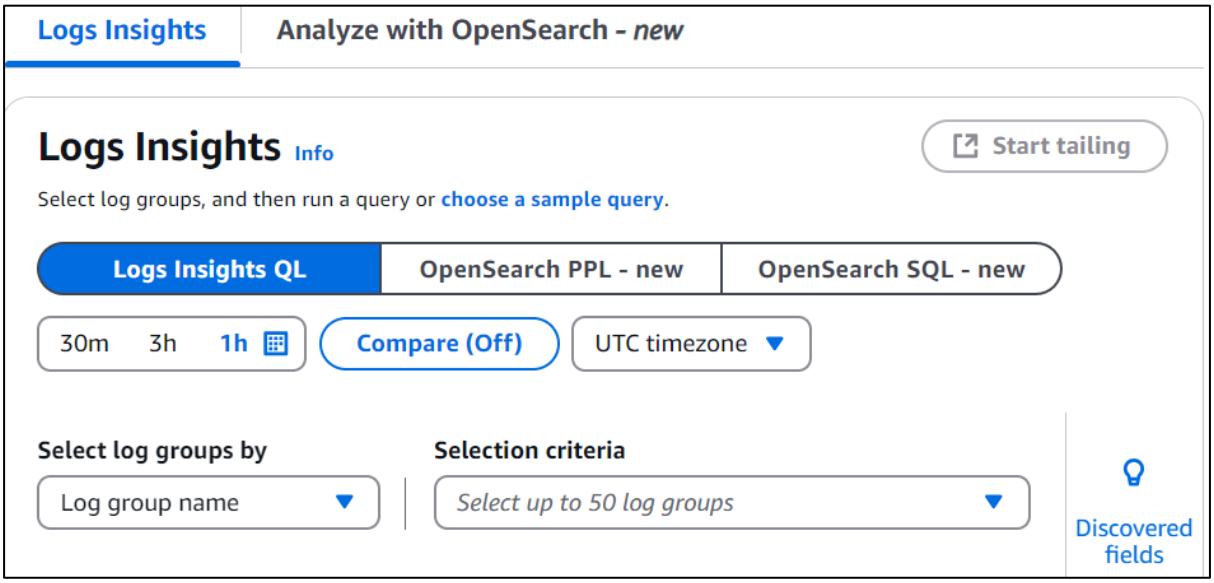


- In the left hand navigation panel, click on **Logs Insights.**

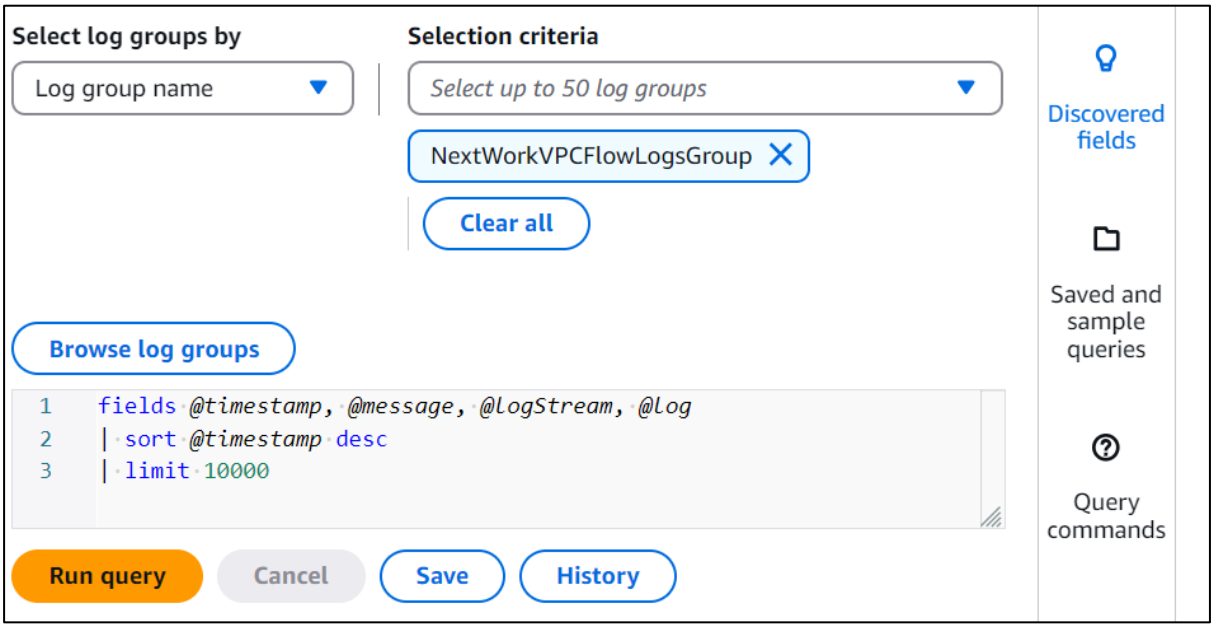


What are Logs Insights?

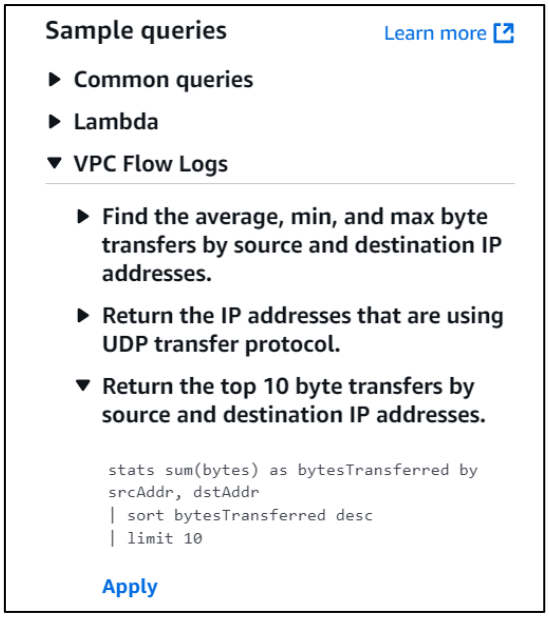
Logs Insights is a CloudWatch feature that analyzes your logs. In Log Insights, you use queries to filter, process and combine data to help you troubleshoot problems or better understand your network traffic!



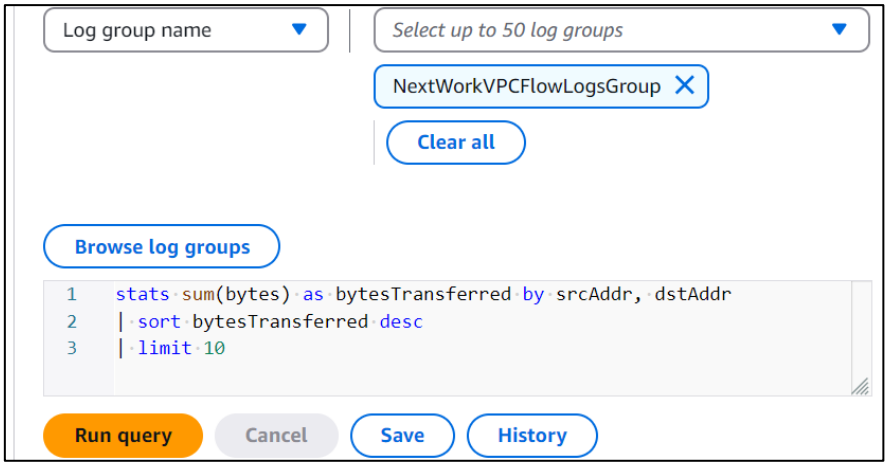
- Select **NextWorkVPCFlowLogsGroup** from the **Select log group(s)** dropdown.



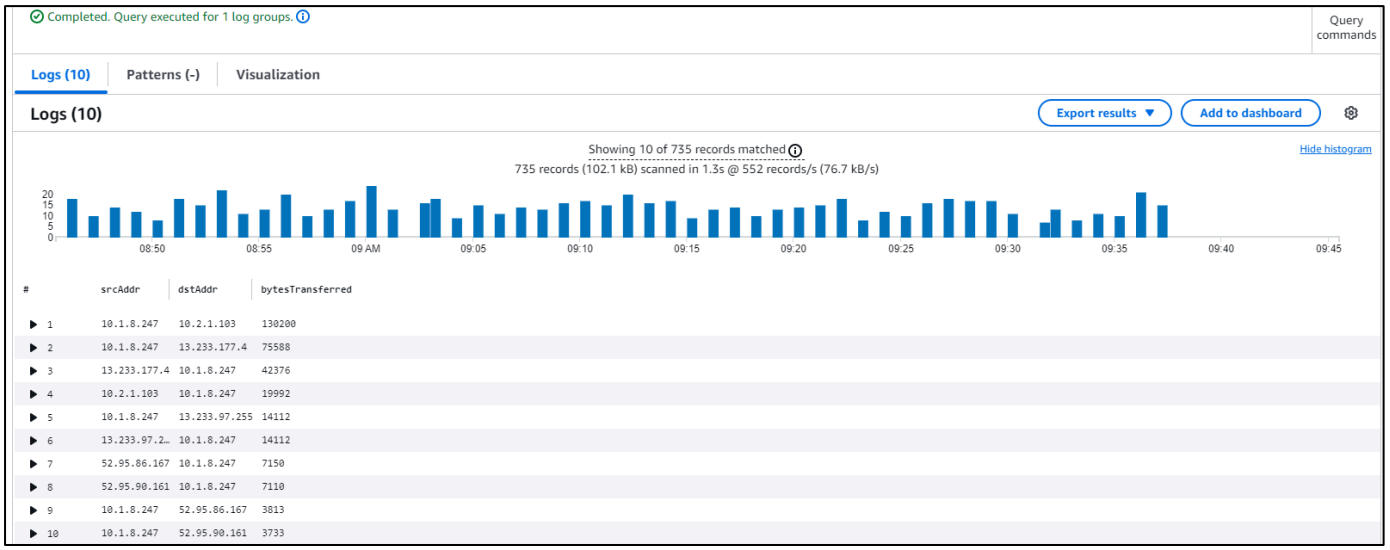
- Select the **Saved and sample queries** folder on the right hand side.
- Under **Flow Logs**, select **Return the top 10 byte transfers by source and destination IP addresses.**



- Click **Apply,** and then **Run query.**



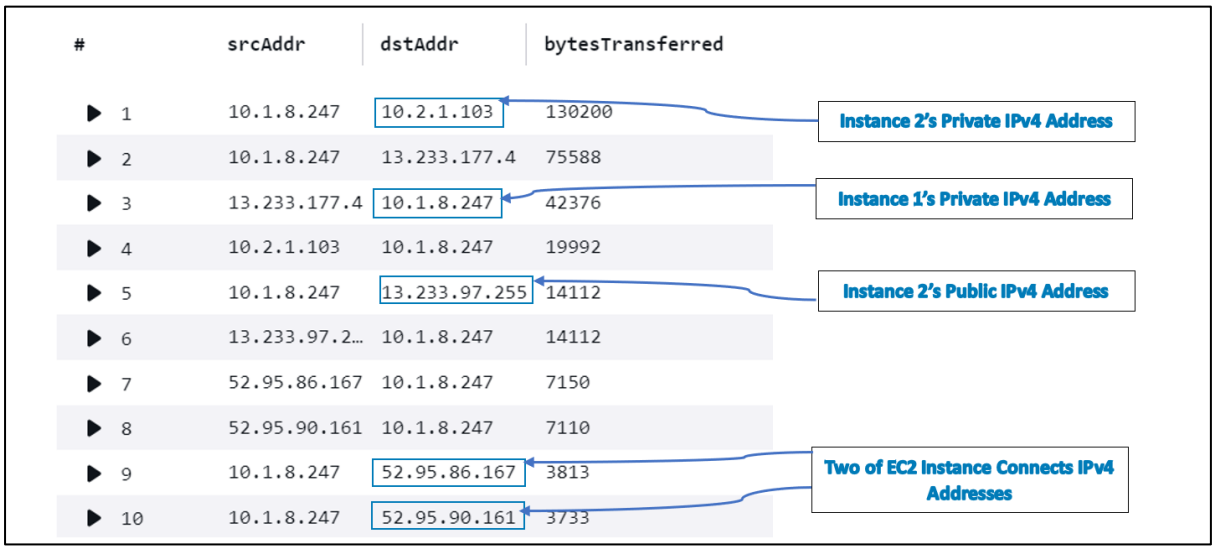
- Review the query results



What are these results telling me?

Wow! Out of all the logs that Flow Logs has captured, here are the ten pairs of source and destination IP addresses that transferred the most data between them.
So, as you might imagine, this is a great query for investigating any heavy traffic flows or unusual data transfers!
p.s. the bar chart at the top is just a little visualization to show you how many logs were captured at specific times of the day. The table below are the actual results of your query.

You can map the logs with IP addresses you've come across during this project?



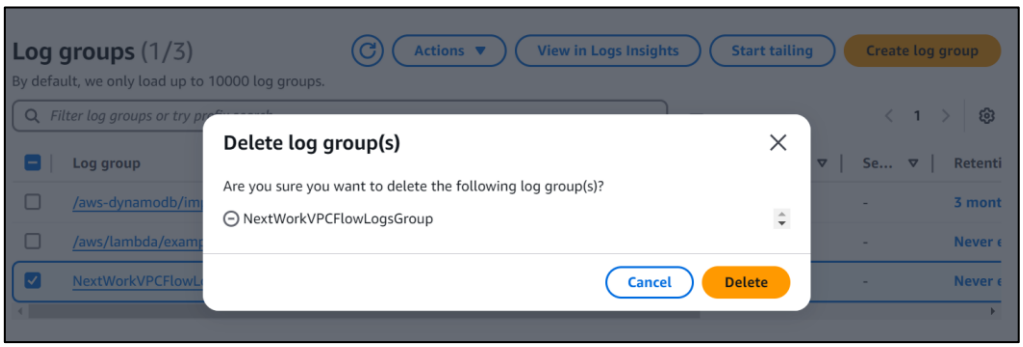
#	srcAddr	dstAddr	bytesTransferred
1	10.1.8.247	10.2.1.103	130200
2	10.1.8.247	13.233.177.4	75588
3	13.233.177.4	10.1.8.247	42376
4	10.2.1.103	10.1.8.247	19992
5	10.1.8.247	13.233.97.255	14112
6	13.233.97.2…	10.1.8.247	14112
7	52.95.86.167	10.1.8.247	7150
8	52.95.90.161	10.1.8.247	7110
9	10.1.8.247	52.95.86.167	3813
10	10.1.8.247	52.95.90.161	3733

You've just completed today's project and **learnt how to monitor traffic within your VPC.**

Delete Your Resources

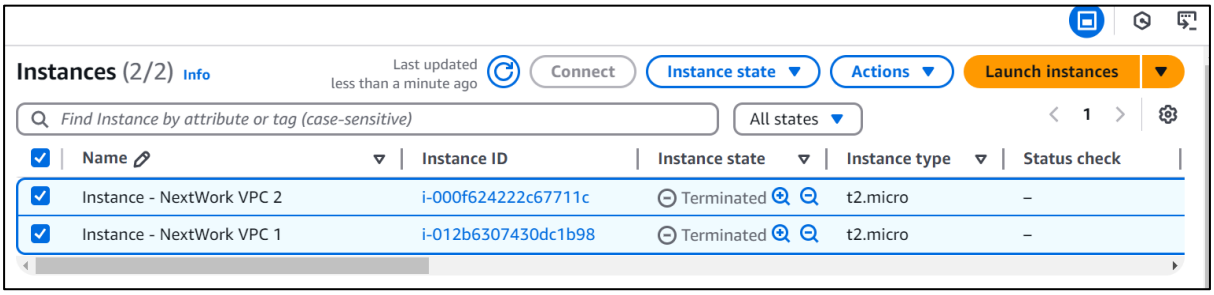
Delete your CloudWatch Log Group

- In your CloudWatch console, select **Log groups** from the left hand navigation panel.
- Select your **NextWorkVPCFlowLogsGroup.**
- Click the **Actions** button, and select **Delete log group(s)**.
- Confirm by pressing the **Delete** button.



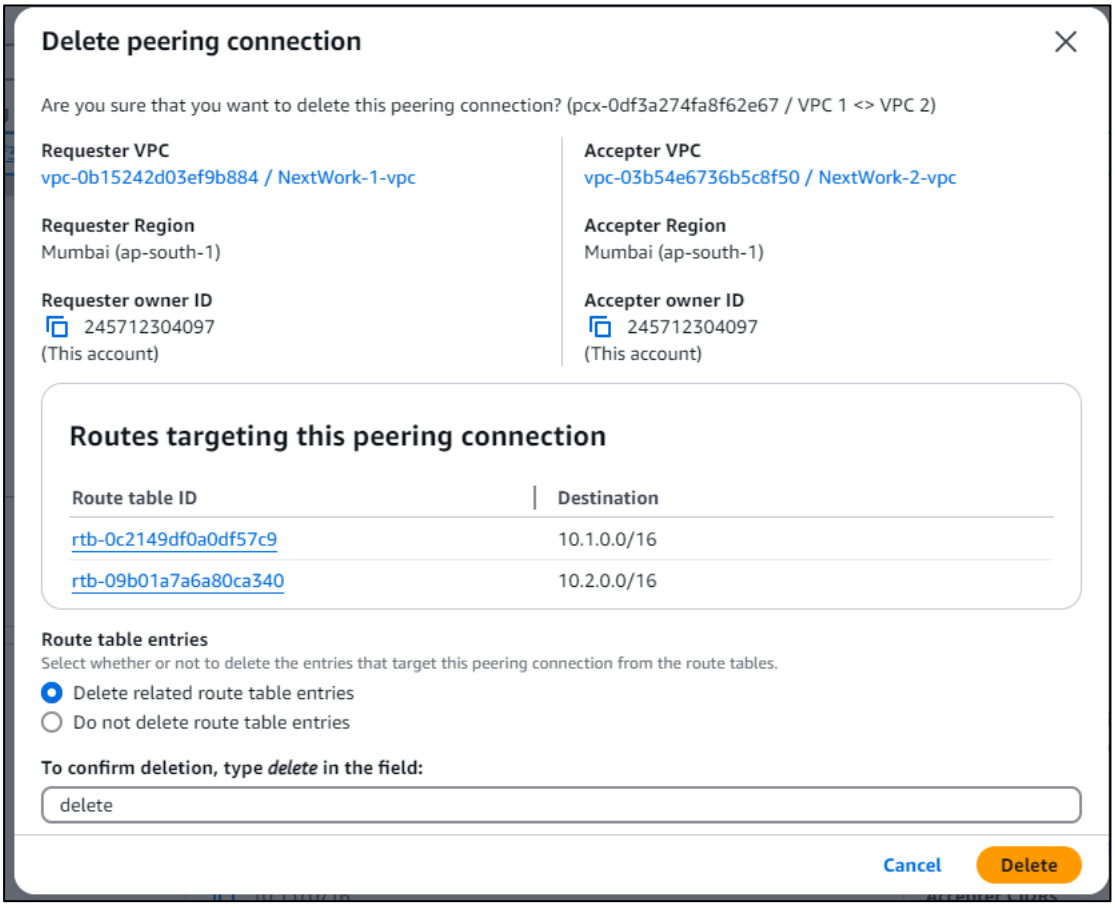
Delete your EC2 Instances

- Head back to the **Instances** page of your EC2 console.
- Select the checkboxes next to **Instance - NextWork VPC 1** and **Instance - NextWork VPC 2.**
- Select **Instance state,** then select **Terminate Instance.**
- Select **Terminate**.



Delete VPC Peering Connections

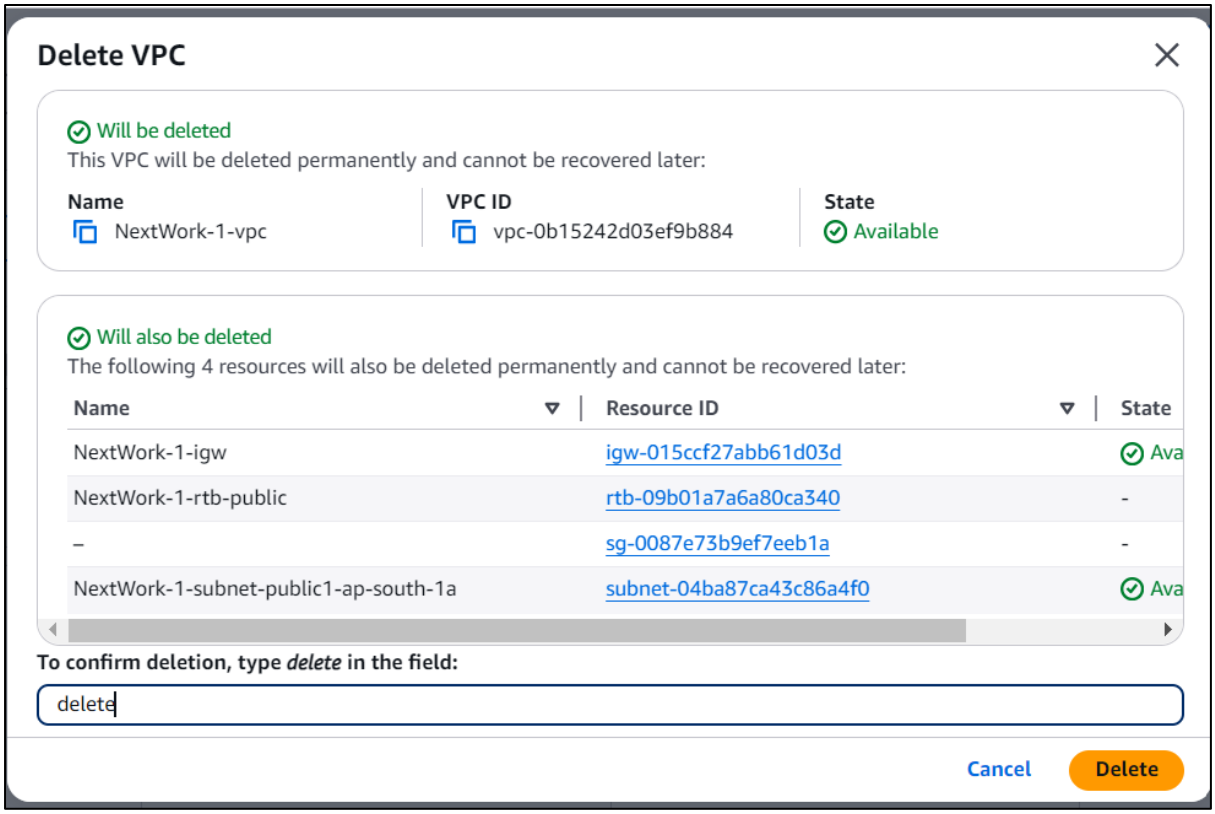
- Head back to your **VPC** console.
- Select **Peering connections** from your left hand navigation panel.
- Select the VPC 1 <> VPC 2 peering connection.
- Select **Actions**, then **Delete peering connection.**
- Select the checkbox to **Delete related route table entries**.
- Type delete in the text box and click **Delete.**



Delete your VPCs

- Select **Your VPCs** from your left hand navigation panel.
- Select **NextWork-1-vpc,** then **Actions**, and **Delete VPC.**
- Type delete in the text box and click **Delete.**

Note: if you get stopped from deleting your VPC because **network interfaces** are still attached to your VPC - delete all the attached network interfaces first!



- Select **NextWork-2-vpc,** then **Actions**, and **Delete VPC.**
- Type delete in the text box and click **Delete.**

Other network components should be automatically deleted with your VPC, but it's always a good idea to check anyway.

Don't forget to **refresh** each page before checking if these resources are still in your account:

- Subnets
- Route tables
- Internet gateways
- Network ACLs
- Security groups

Delete your CloudWatch IAM Role and Policy

- Head to your **IAM** console.
- Select **Policies** from the left hand navigation panel.

- Search for FlowLogs, and select **NextWorkVPCFlowLogsPolicy.**
- Select **Delete**, then enter NextWorkVPCFlowLogsPolicy to confirm your deletion.
- Select **Roles** from the left hand navigation panel.
- Search for FlowLogs, and select **NextWorkVPCFlowLogsRole.**
- Select **Delete**, then enter NextWorkVPCFlowLogsRole to confirm your deletion.