VPC Peering

In this project, **VPC Peering** we're going to level up by setting up VPC Peering - we're going to play with TWO VPCs instead of one!

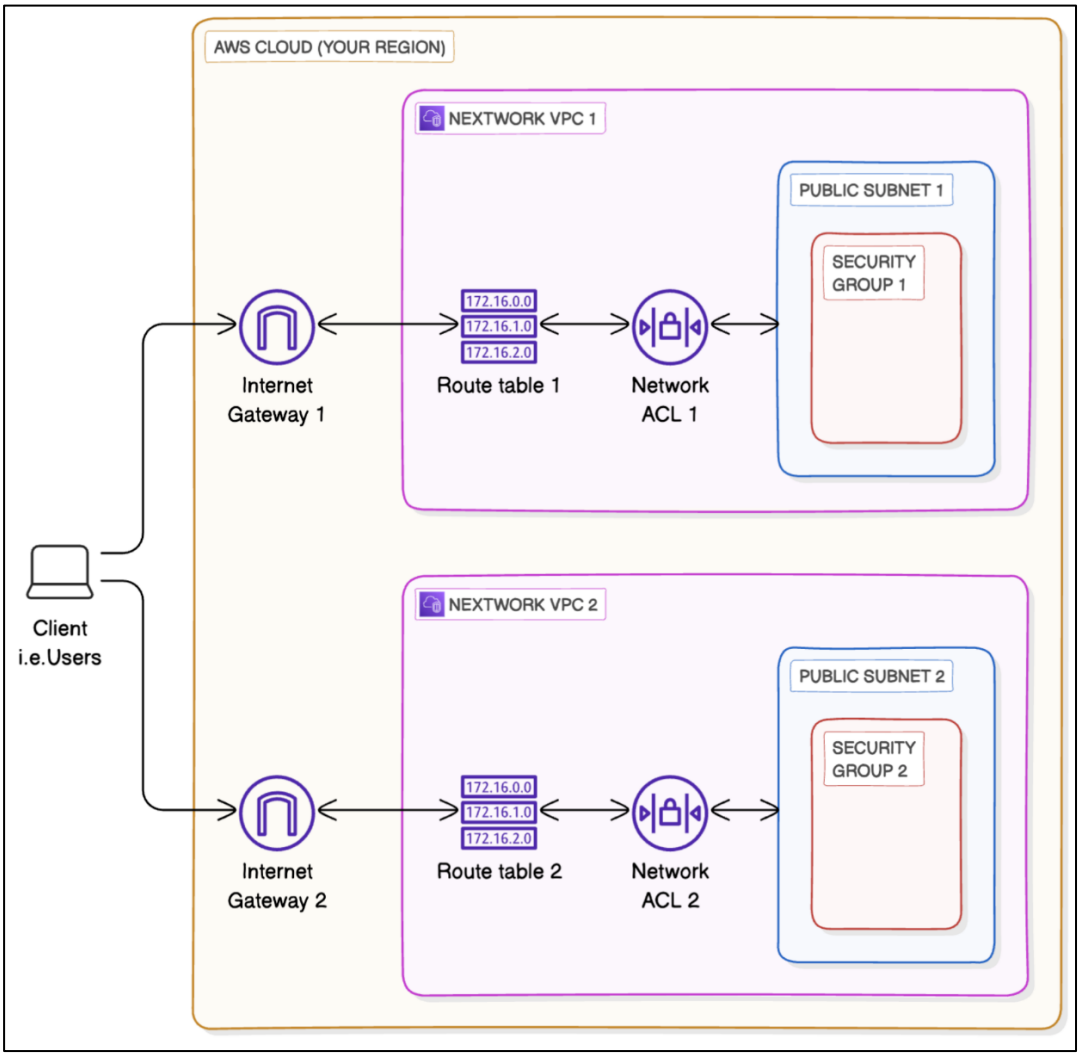
Aim:

1. Set up multiple VPCs.
2. Create a VPC peering connection - i.e. get two VPCs to talk to each other!
3. Test VPC peering with connectivity tests.

Set up your VPCs

In this step, you're going to:

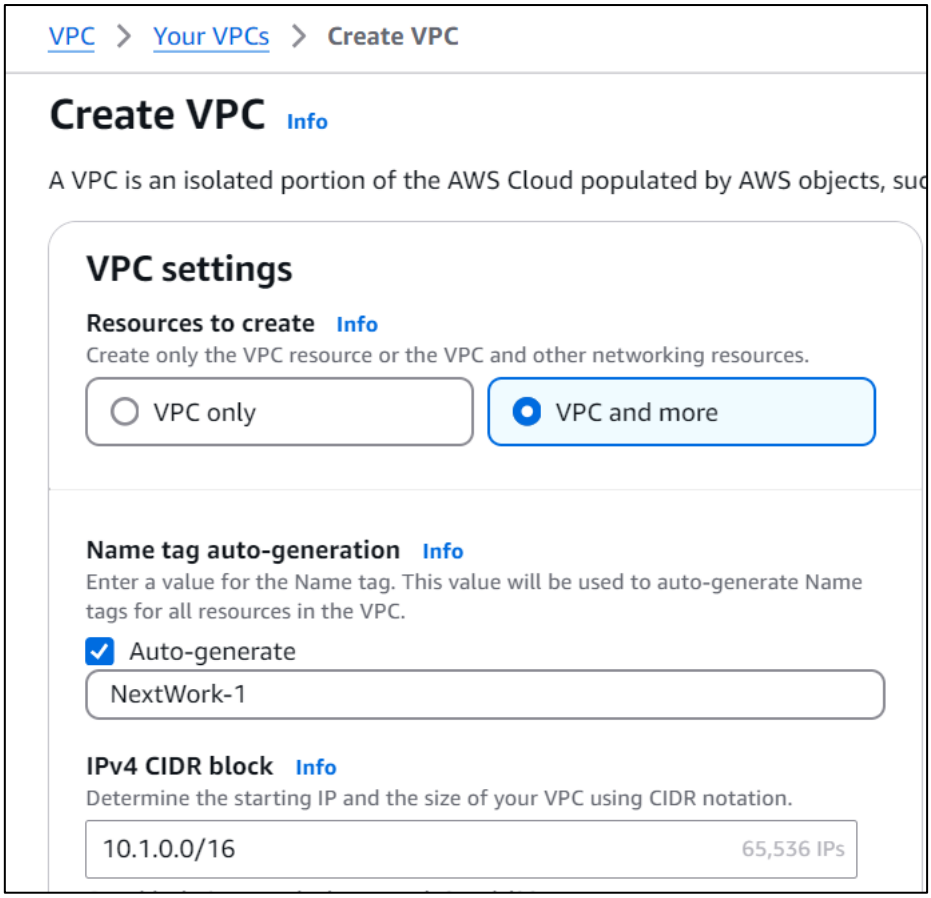
1. Create two VPCs from scratch!
2. Use the visual VPC resource map to create your VPCs

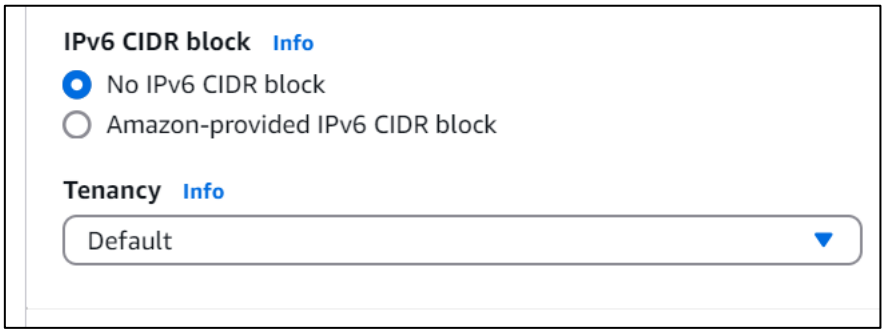


- Log in to your AWS Account.
- Head to your **VPC** console - search for VPC at the search bar at top of your page.
- From the left hand navigation bar, select **Your VPCs.**
- Select **Create VPC.**
- let's select **VPC and more.**

Create VPC 1

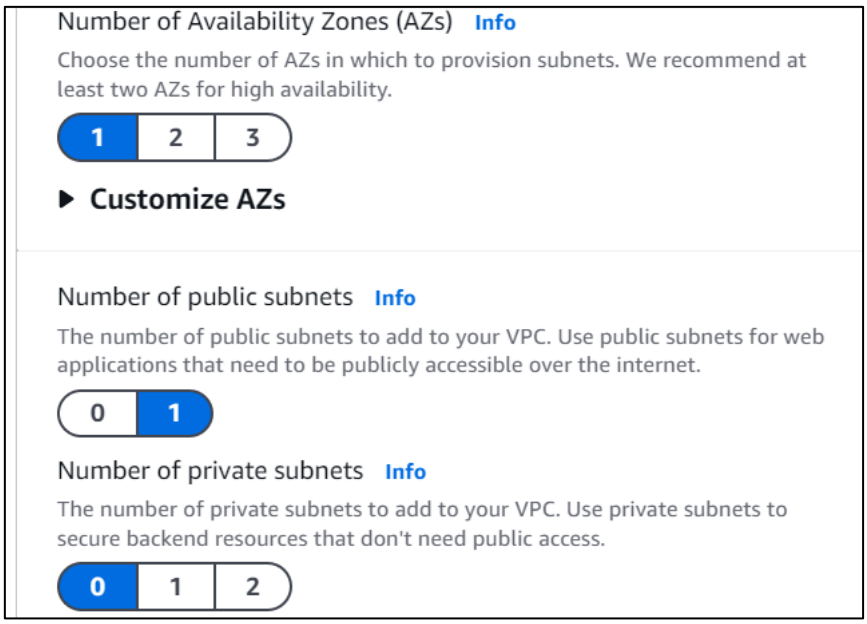
- Under **Name tag auto-generation**, enter NextWork-1
- The VPC's **IPv4 CIDR block** is already pre-filled to 10.0.0.0/16 - change that to 10.1.0.0/16
- For **IPv6 CIDR block**, we'll leave in the default option of **No IPv6 CIDR block.**
- For **Tenancy**, we'll keep the selection of **Default.**



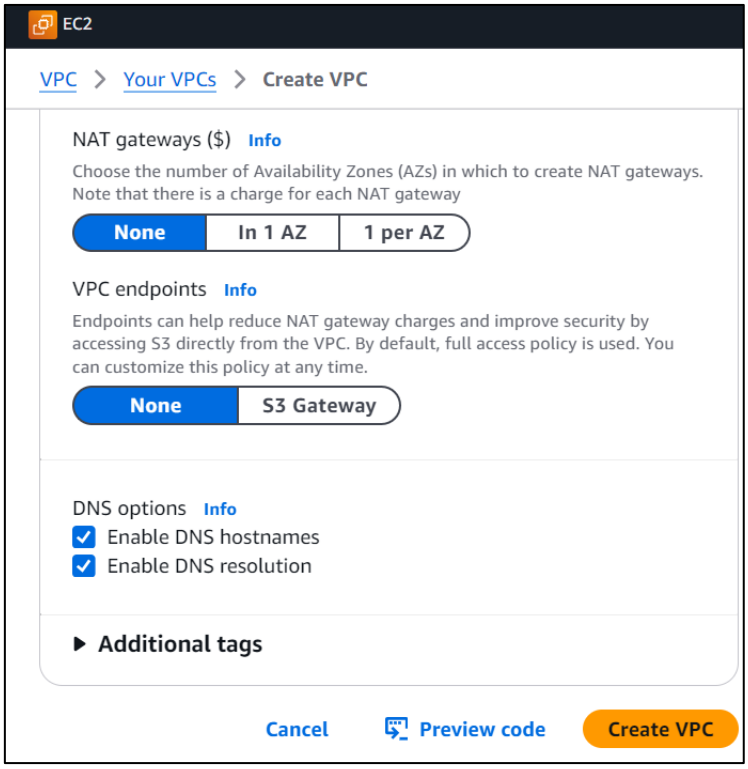


- For **Number of Availability Zones (AZs)**, we'll use just **1** Availability Zone.

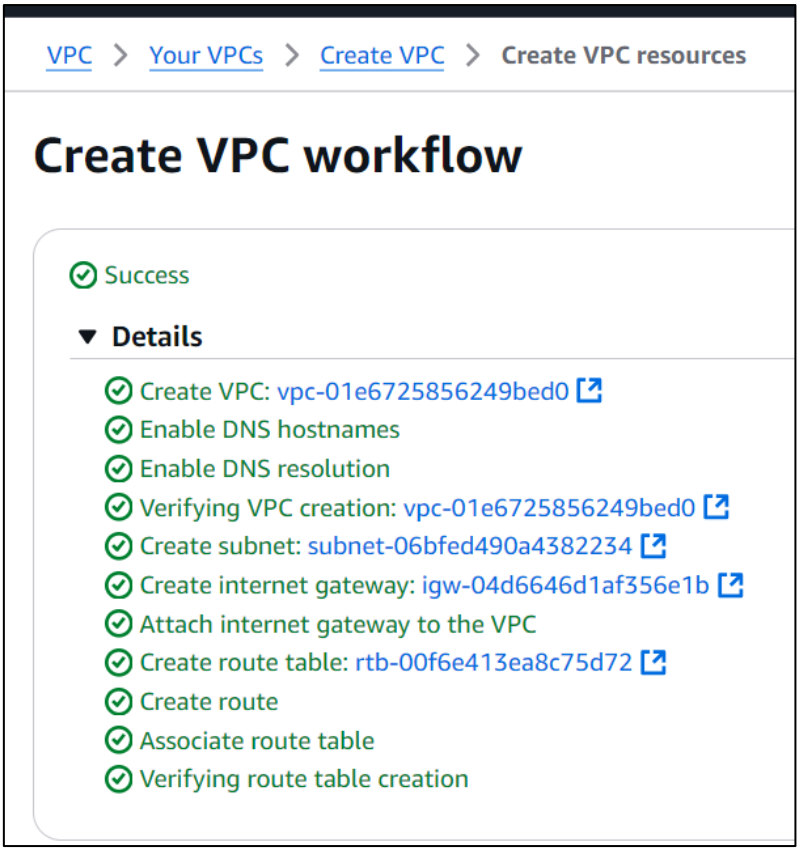
- Make sure the **Number of public subnets** chosen is **1**.
- For **Number of private subnets,** we'll keep thing simple today and go with **0** private subnets.



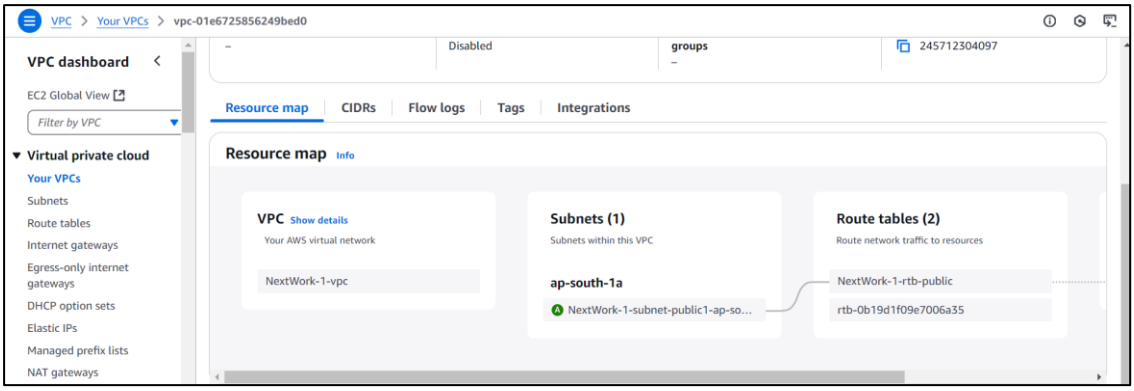
- Next, for the **NAT gateways ($)** option, make sure you've selected **None.** As the dollar sign suggests, NAT gateways cost money!
- Next, for the **VPC endpoints** option, select **None.**
- You can leave the **DNS options** checked.

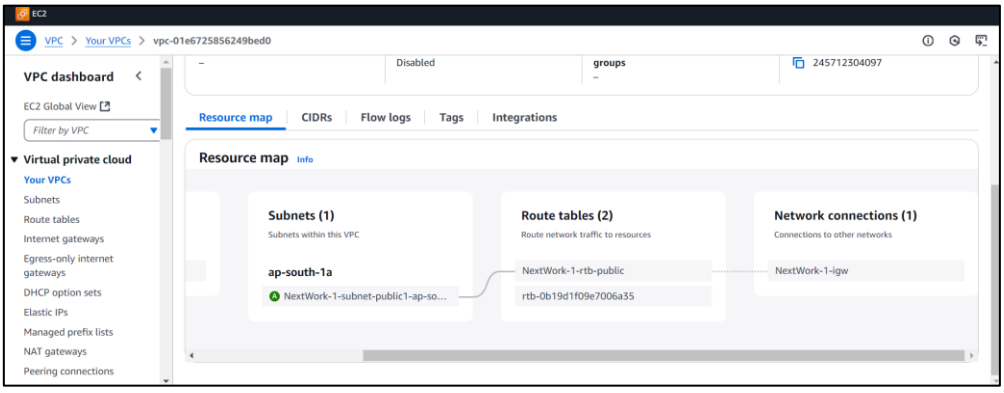


- Select **Create VPC.**



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

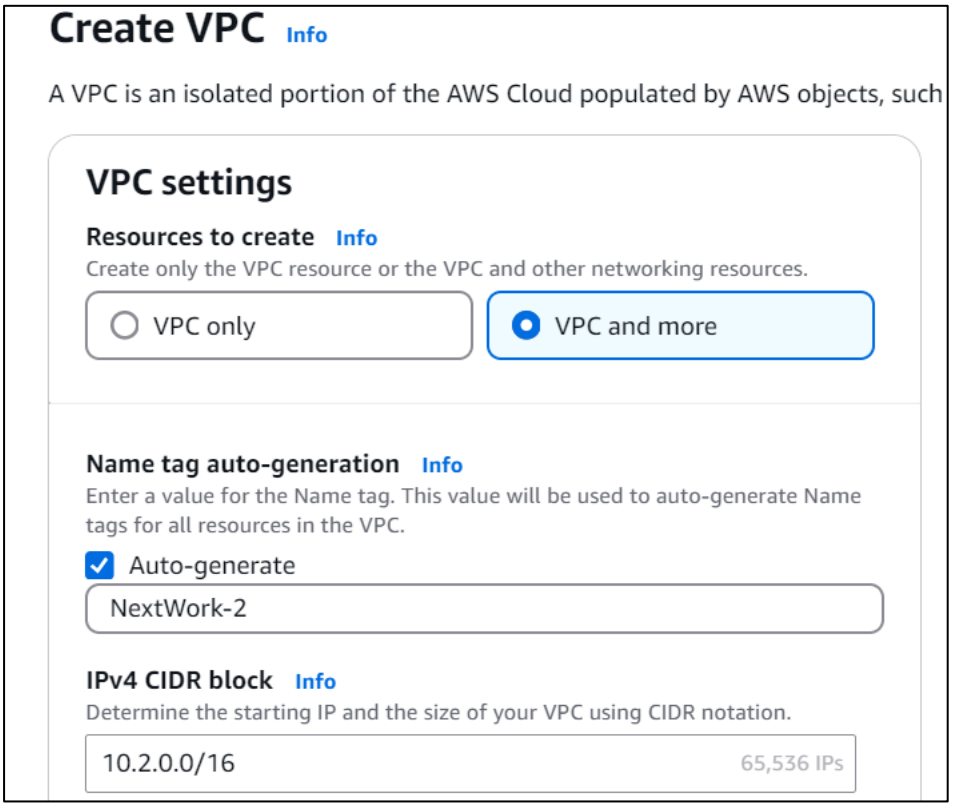




Set up VPC 2

We're going to set up another VPC with *slightly* different settings

- Select **Create VPC.**
- Select **VPC and more.**
- Under **Name tag auto-generation**, enter NextWork-2
- The VPC's **IPv4 CIDR block** should be unique! Make sure the CIDR block is NOT 10.1.0.0/16 - it should be 10.2.0.0/16



- 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 go with **0** for today's project. Let's keep it simple with just a single subnet!
- For the **NAT gateways ($)** option, select **None.**
- For the **VPC endpoints** option, select **None.**
- You can leave the **DNS options** checked.
- Select **Create VPC.**

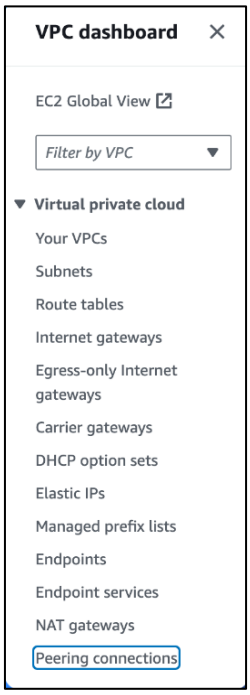
Create a Peering Connection

Now that you have two VPCs ready to go, let's bridge them together with a peering connection.

In this step, you're going to:

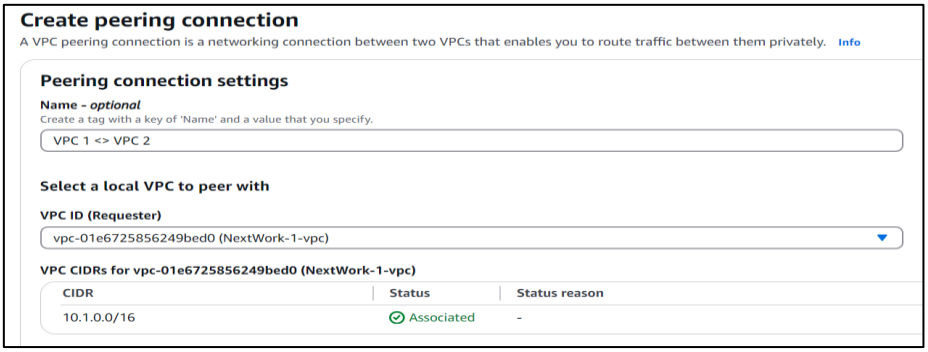
1. Set up a connection link between your VPCs.

- Still in the VPC console, click on **Peering connections** on the left hand navigation panel.

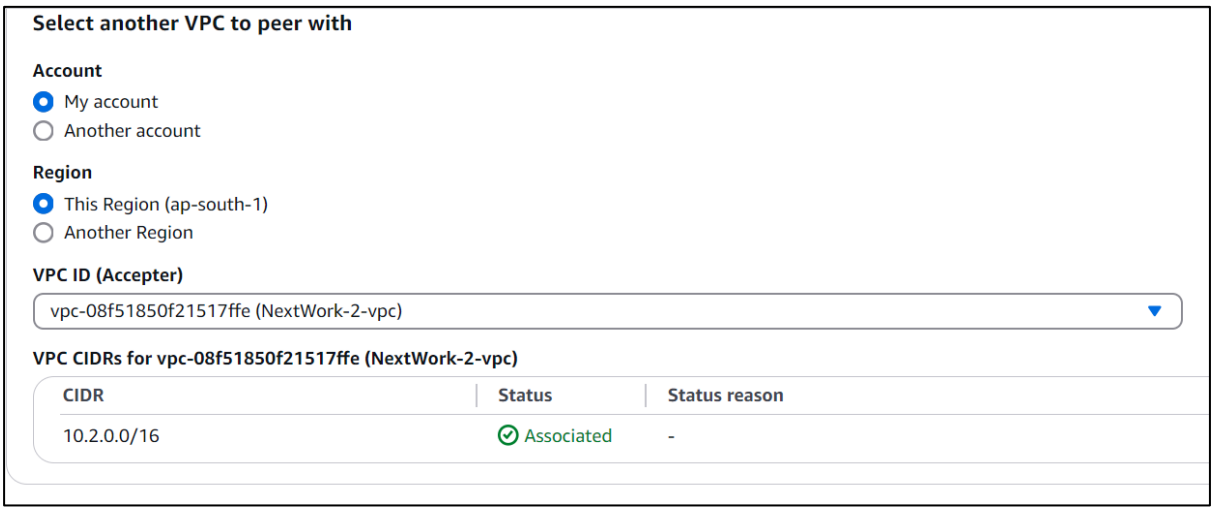


- Click on **Create peering connection** in the right hand corner.

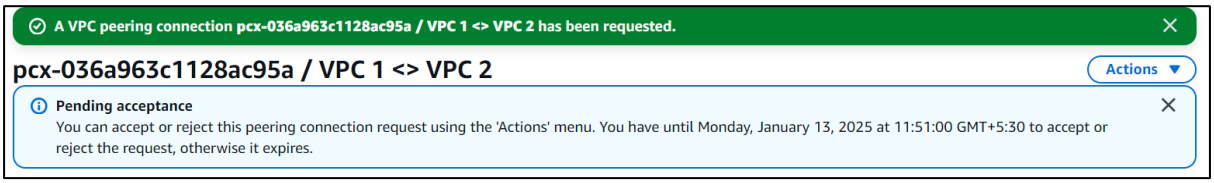
- Name your **Peering connection name** as VPC 1 <> VPC 2
- Select **NextWork-1-VPC** for your **VPC ID (Requester).**



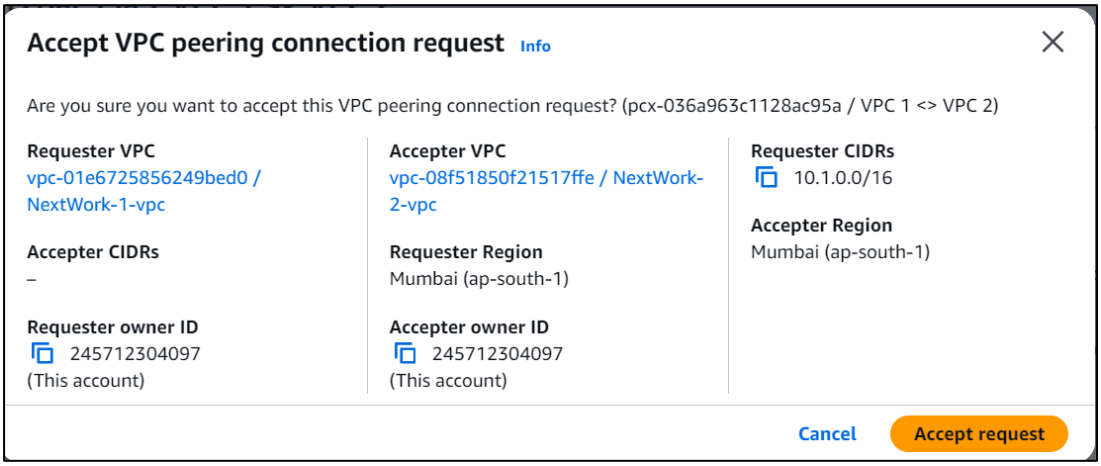
- Under **Select another VPC to peer with,** make sure **My Account** is selected.
- For **Region,** select **This Region.**
- For **VPC ID (Accepter)**, select **NextWork-2-VPC**

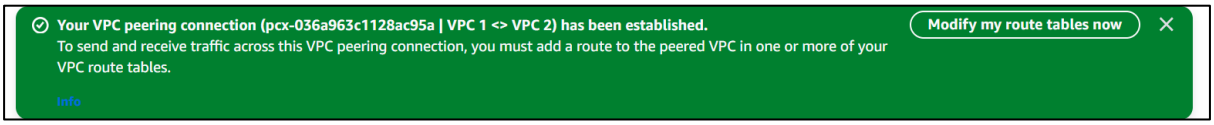


- Click on **Create peering connection.**
- Your newly created peering connection isn't finished yet! The green success bar says the peering connection **has been requested.**



- 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 Route Tables

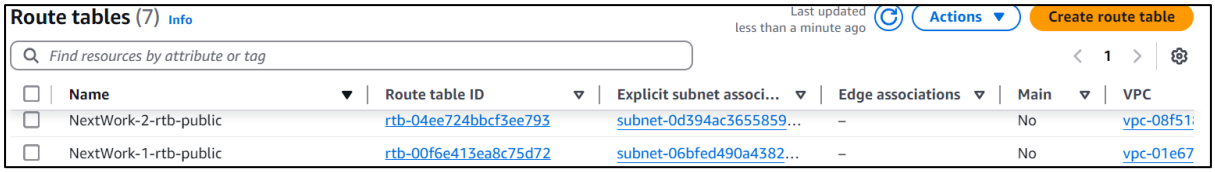
- With a peering connection all set up ✅, now it's time for traffic in your VPCs to learn how to use it.

In this step, you're going to:

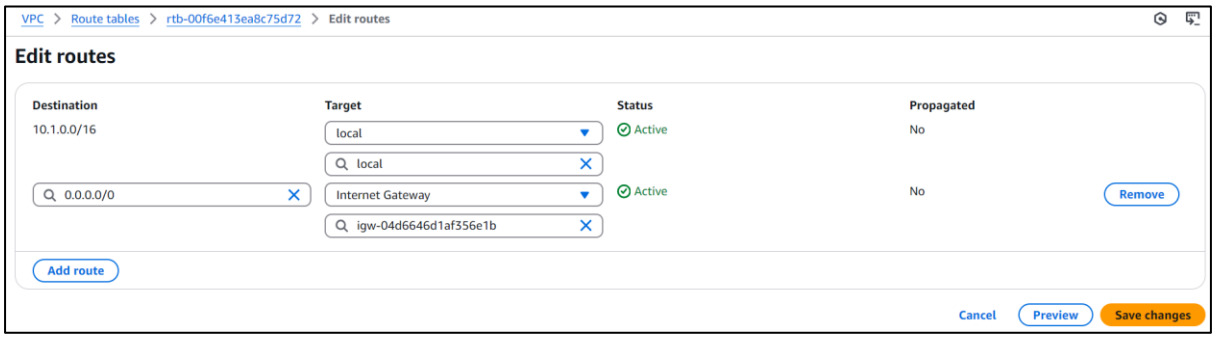
- Set up a way for traffic coming from VPC 1 to get to VPC 2.
- Set up a way for traffic coming from VPC 2 to get to VPC 1.

Update VPC 1's route table

- Select the checkbox next to VPC 1's route table i.e. called **NextWork-1-rtb-public.**

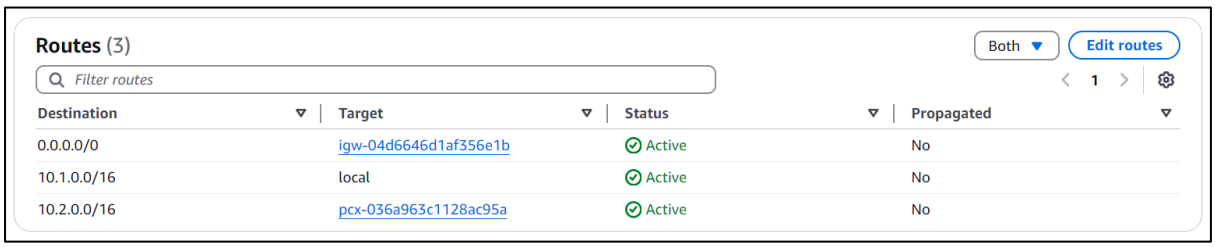


- Scroll down and click on the **Routes** tab.
- Click **Edit routes.**



- 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**.

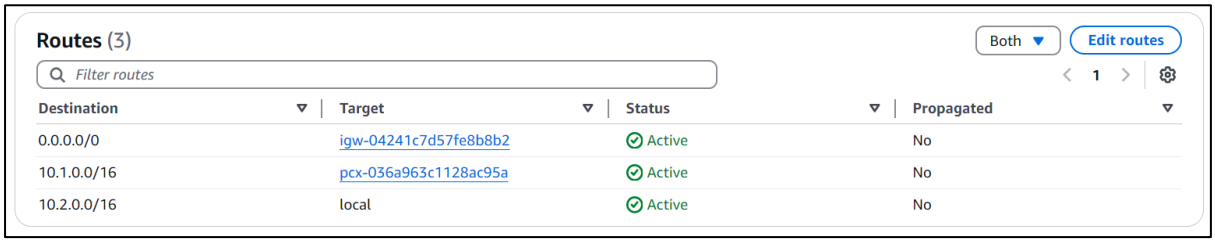
- Click **Save changes.**
- Routes appear in VPC 1's **Routes** tab now!



Update VPC 2's route table

set up the equivalent route in VPC 2's route table same as you have done above for VPC 1

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



Launch EC2 Instances

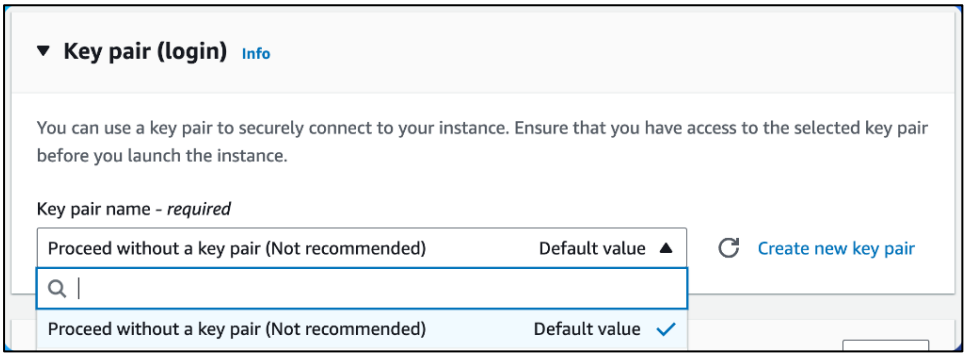
It's time to launch EC2 instances into your architecture!

In this step, you're going to:

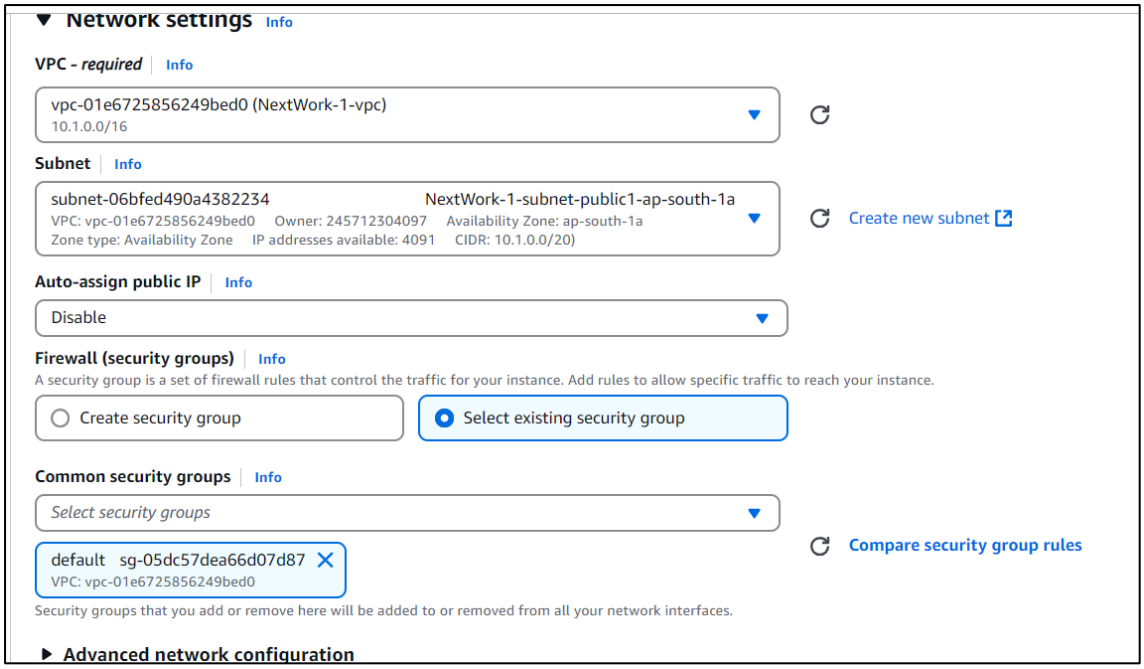
1. Launch an EC2 instance in each VPC, so we can use them to test your VPC peering connection later.

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).**



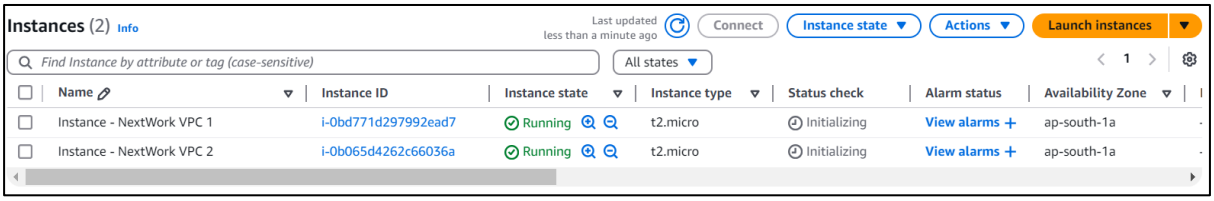
- At the **Network settings** panel, select **Edit** at the right hand corner.
- Under **VPC**, select **NextWork-vpc-1**.
- Under **Subnet,** select your VPC's public subnet.
- Keep the **Auto-assign public IP** setting to **Disable.**
- For the **Firewall (security groups)** setting, Amazon VPC already created a security group for your VPC - let's use that!
- Choose **Select existing security group.**
- Select the **default** security group for your VPC.
- Select **Launch instance.**



Launch an instance in VPC 2

set up an EC2 instance in VPC 2, use the same instructions above but make sure:

1. The **Name** is Instance - NextWork VPC 2
2. The **VPC** is **NextWork-vpc-2**.



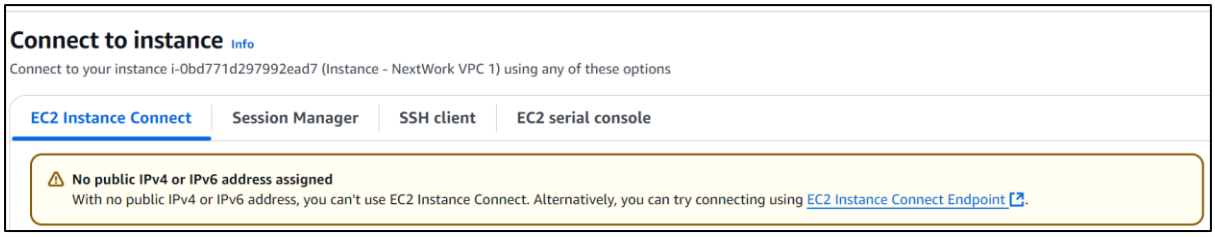
Connect to Instance 1

To test our VPC peering connection, we'll need to get one of our EC2 instances to try talk to the other.

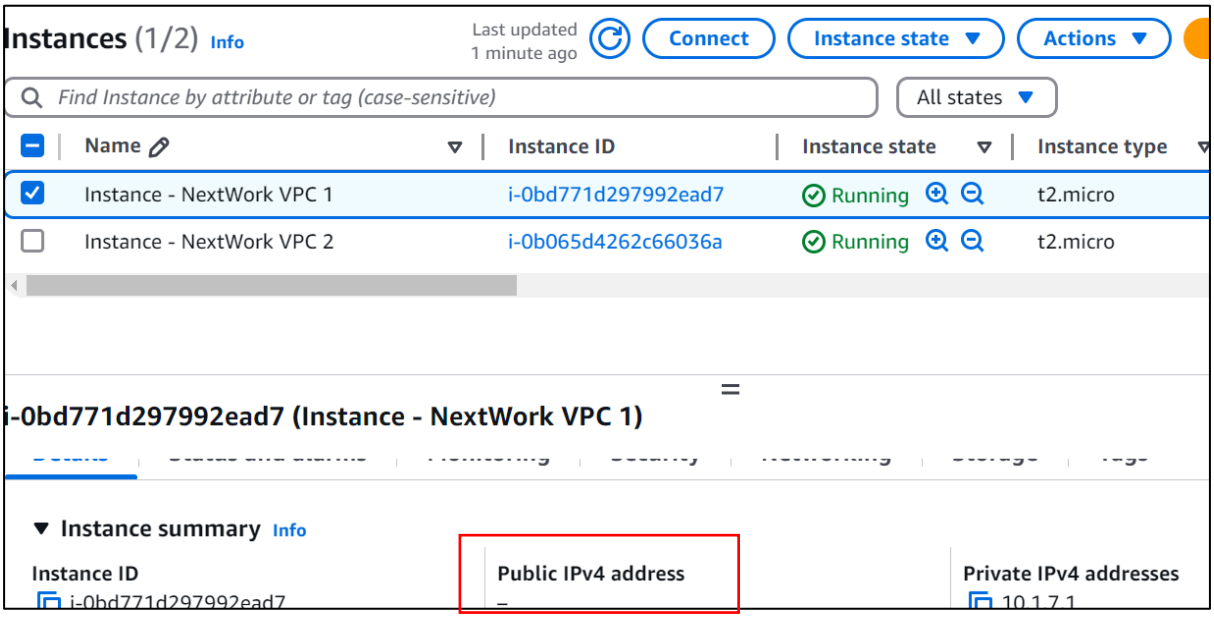
In this step, you're going to:

1. Use EC2 Instance Connect to connect to your first EC2 instance.
2. Fix a connection error!

- Still in your **EC2** console, select the checkbox next to **Instance - NextWork VPC 1.**
- Select **Connect**.

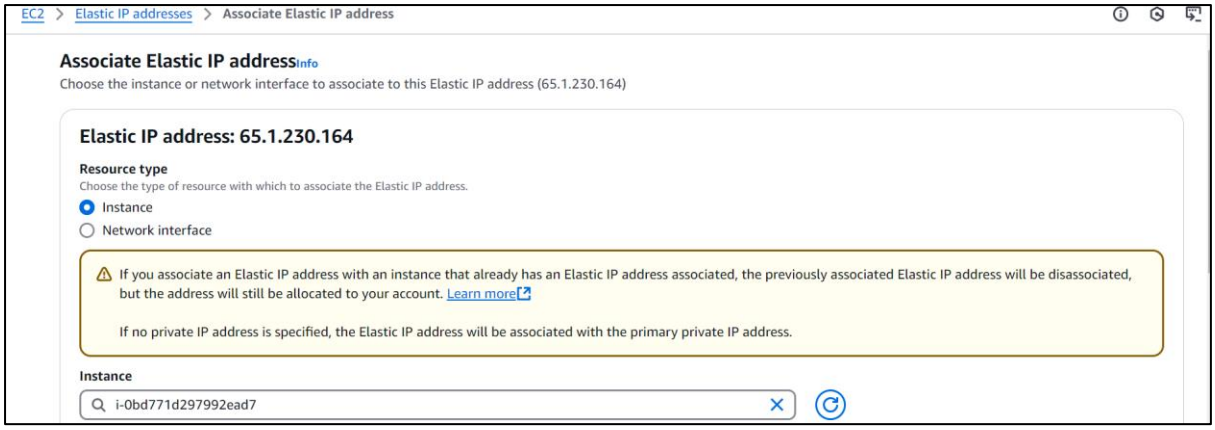


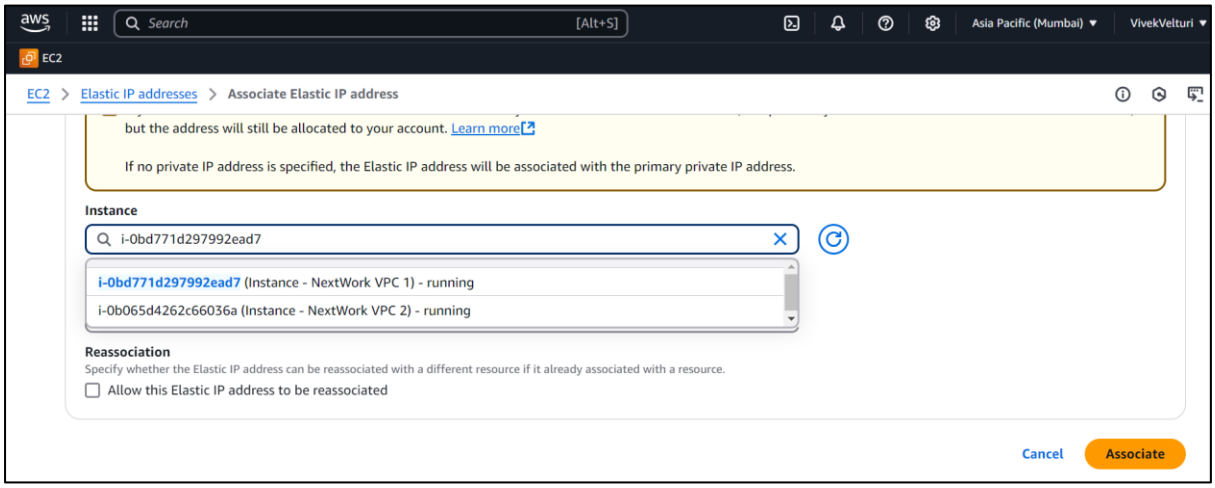
- Keeping **Disable** for the **Auto-assign IP address** option in our EC2 instance's network settings caused this error!
- Verify this by heading back to the **Instances** page in your EC2 console, and checking the Public IPv4 address field... it's empty



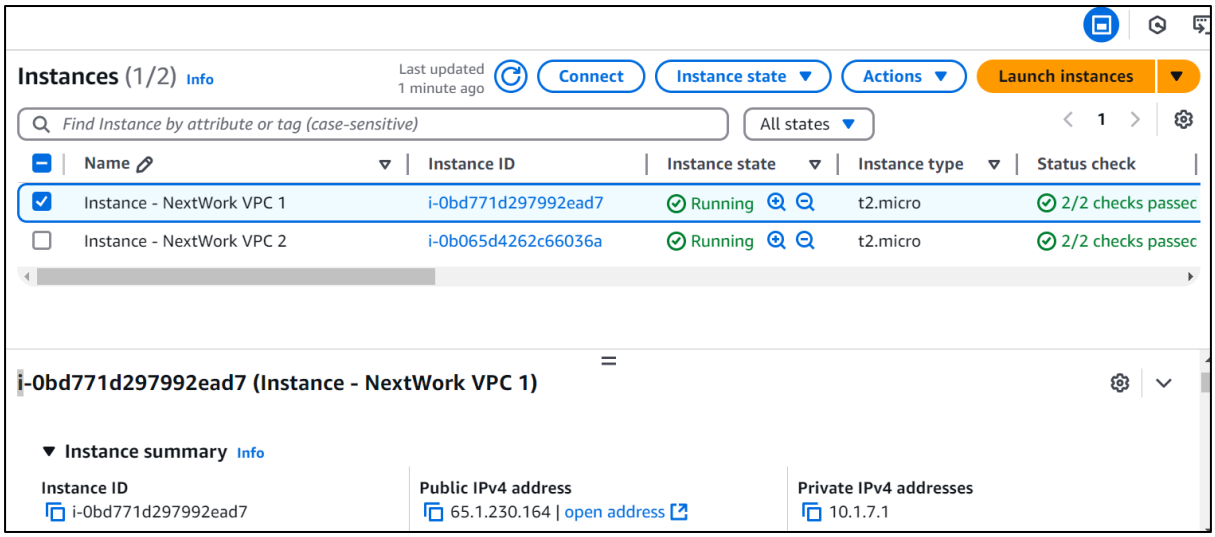
- On your EC2 console's left hand navigation panel, select **Elastic IPs.**

- Select **Allocate Elastic IP** addresses.
- Leave all default options.
- Select **Allocate.**
- Refresh your page, then select the new IP address you've set up.
- Select the **Actions** dropdown, then select **Associate Elastic IP address.**
- Under **Instance**, select **Instance - NextWork VPC 1.**





- Click **Associate**.
- Your EC2 Instance should have a public IP address now.
- To check this, select **Instances** from the left hand navigation panel.
- Select the checkbox next to **Instance - NextWork VPC 1.**
- You will see a **Public IPv4 address** for your EC2 instance now



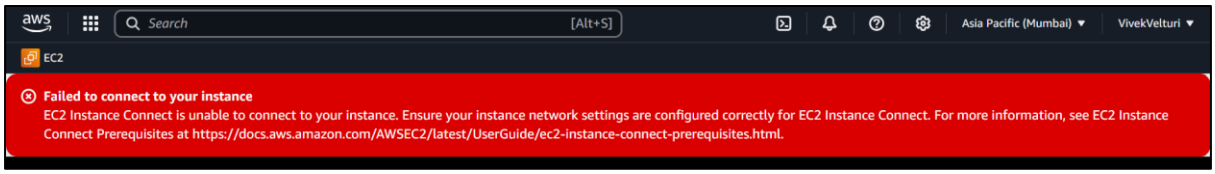
Connect to Instance 1 (round two!)

IP address should be all resolved now... let's try connecting to your EC2 instance again

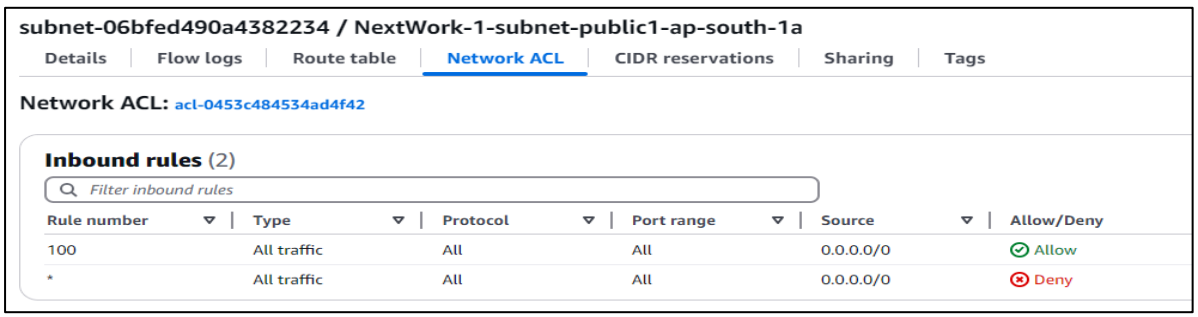
In this step, you're going to:

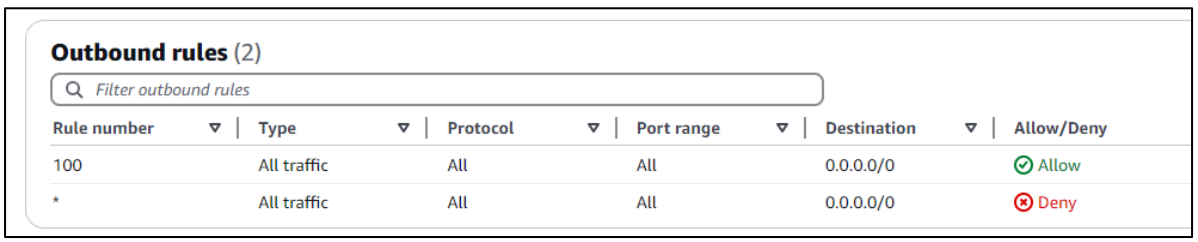
1. Use EC2 Instance Connect to connect to Instance 1 (one more time)!
2. Fix (another) error.

- Select **Connect.**
- In the EC2 Instance Connect set up page, select **Connect** again.

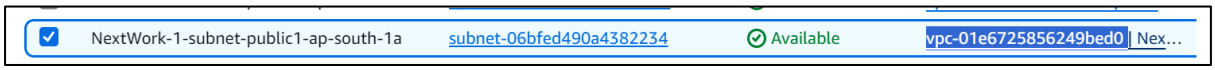


- We've failed to connect to our instance
- Head back to your **VPC console.**
- Select **Subnets** from the left hand navigation panel.
- Select the checkbox next to **NextWork-1-subnet-public1...**
- Investigate the **Route table** and **Network ACL** tabs

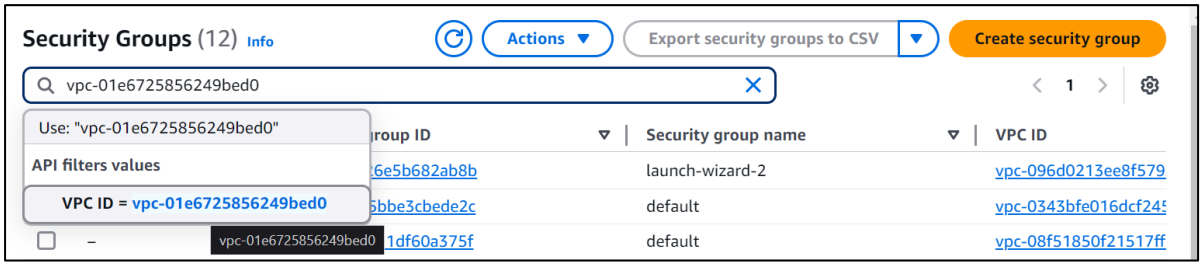




- leaves one more thing to investigate..., Copy the **VPC ID** of **NextWork-1-vpc.**



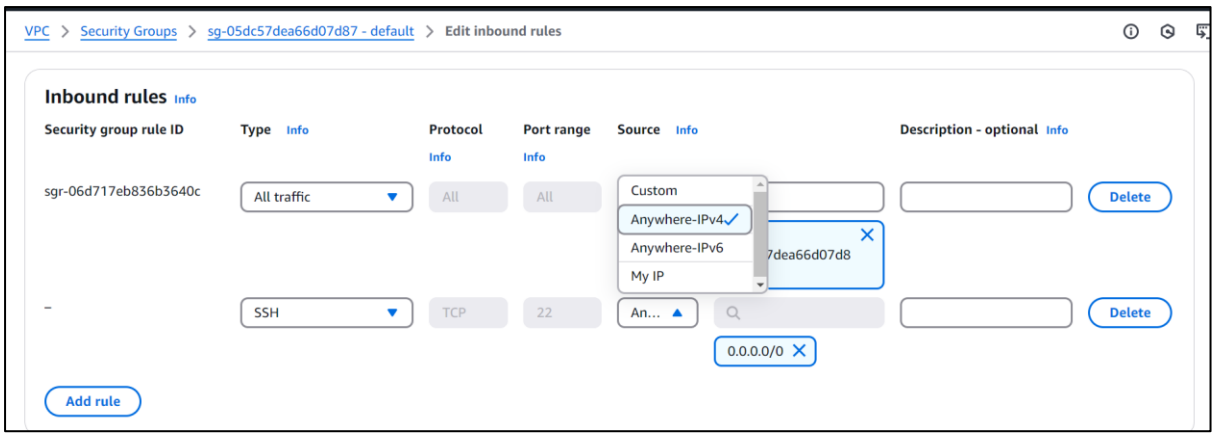
- Head into the **Security groups** page from the left hand navigation panel.
- There will be a bunch of nameless security groups!
- To find the VPC 1's default security group, let's use the handy search bar.
- Click into the search bar, and paste the ID you copied into the search bar. Make sure there aren't any empty spaces before your text.
- Select the filter!



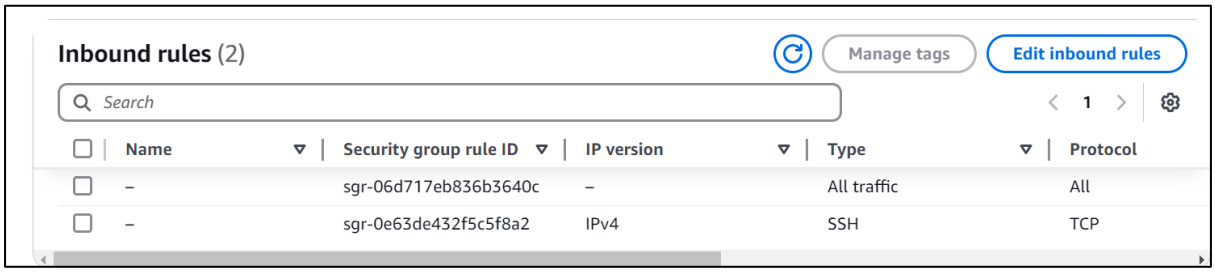
- Now that it narrowed it nicely for us. Select the checkbox next to VPC 1's **default** security group.
- Select the **Inbound rules** tab.

- The answer lies in the **Source** of your security group's inbound rules.
  We're trying to access Instance - NextWork VPC 1 using SSH through EC2 Instance Connect, which is trying to connect to your instance over the internet.
  Your default security group only allows inbound traffic from within the VPC, so traffic from the internet is being cut off!
  That's a key learning to take away: The default security group for a new VPC does not allow incoming traffic from outside of the VPC.
  You have to allow inbound SSH traffic on port 22 yourself!

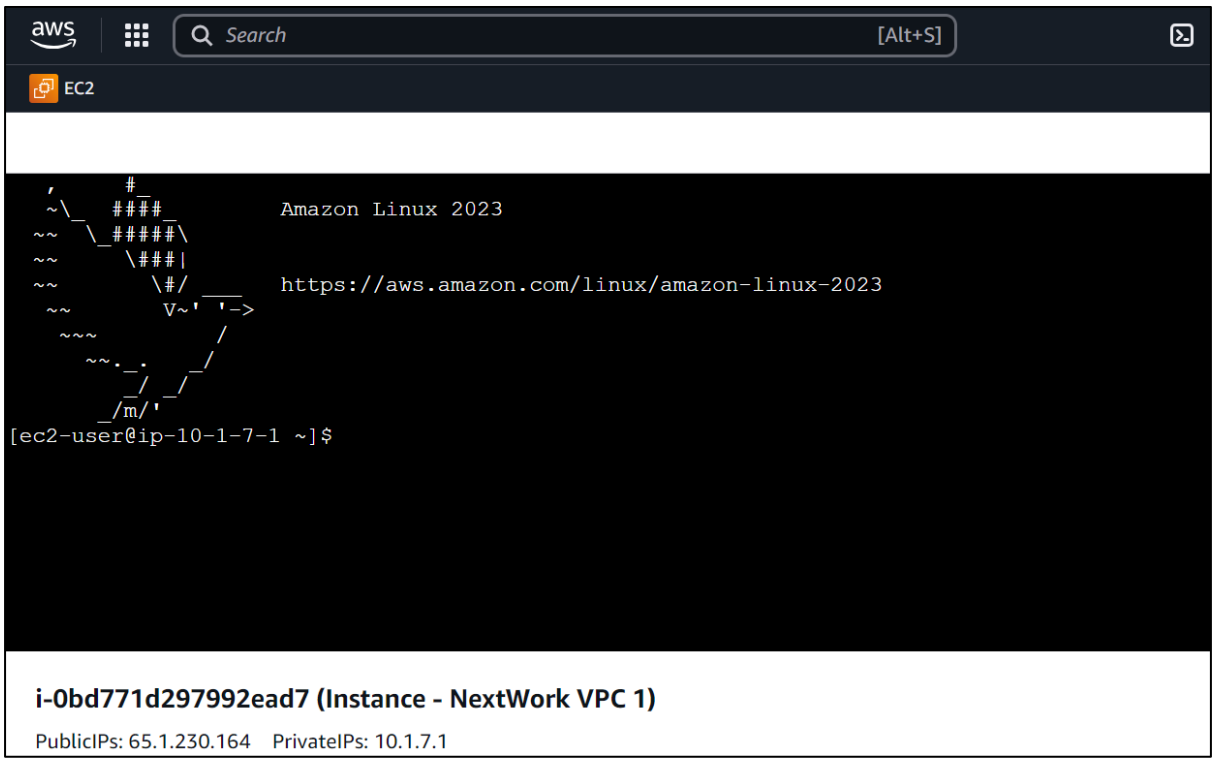
- In the **Inbound rules** tab, select **Edit inbound rules.**
- Select **Add rule.**
- For your new rule, configure the **Type** as **SSH.**
- Then, under **Source type**, select **Anywhere-IPv4.**



- Select **Save rules.**



- With that modified, refresh your EC2 console's **Instances** page.
- Select your **Instance-NextWork VPC 1** and select **Connect** again.
- Select **Connect** in the EC2 Instance Connect setup page.



- Success.

Test VPC Peering

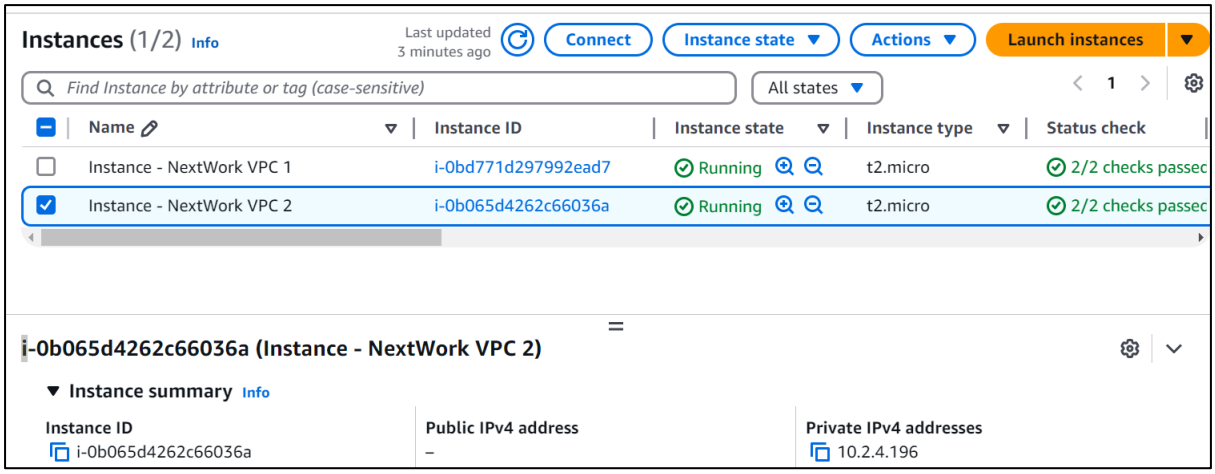
we've figured out how to connect with VPC 1's Instance!

Let's see if we can connect with VPC 2's instance from here.

In this step, you're going to:

1. Get Instance 1 to send test messages to Instance 2.
2. Solve connection errors until Instance 2 is able to send messages back.

- 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.
  - Eg: in my case: ping 10.2.4.196
- You should see a response similar to this

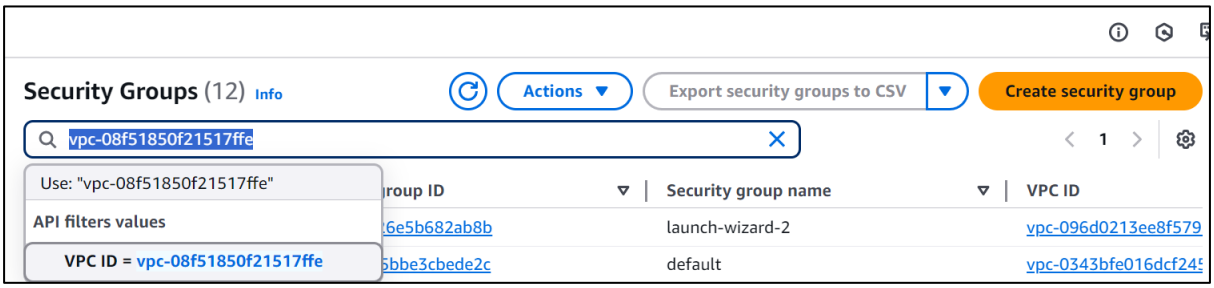
```
[ec2-user@ip-10-1-7-1 ~]$ ping 10.2.4.196
PING 10.2.4.196 (10.2.4.196) 56(84) bytes of data.
```

- This single line indicates that your Instance - NextWork VPC 1 has sent out a ping message... and that's about it.
  Usually, when you ping another computer successfully, you should see **several** replies back instantly. Each reply tells you how long it took for the message to go to the Instance - NextWork VPC 2 and come back.
  If you don't get any replies (that's our situation right now), or if the replies stop suddenly, it's usually a sign that there's a problem with the connection.
  One common reason for these issues is that the target server (Instance - NextWork VPC 2) or its network might be blocking the type of messages used in ping, which are known as **ICMP (Internet Control Message Protocol) traffic.**

Blocking ICMP traffic is often done to prevent network attacks, like attackers can overwhelming a server with ping messages so it can't respond to real users wanting to use your application. Fair enough that ICMP traffic is blocked by default!

- To resolve this connectivity error, let's investigate whether **Instance - NextWork VPC 2** is allowing inbound ICMP traffic.
- 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 2's subnet i.e. **NextWork-2-subnet-public1-...**

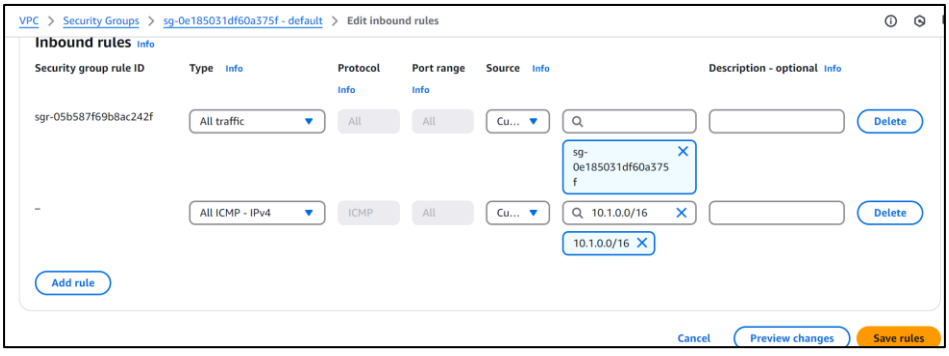
- Let's investigate the **Route tables** and **Network ACL** tabs for your public subnet
- The network ACL allows all types of inbound traffic from anywhere! So this looks perfectly fine.
- Before we finish, let's check the security groups!
- Copy the **VPC ID** of VPC 2.
- Select **Security groups** from the left hand navigation panel.
- Paste the **VPC ID** in the search bar, and select the suggested filter.



- Check your security group's **Inbound rules** tab - does this security group allow ICMP traffic from sources outside of VPC 2? (Nope!)

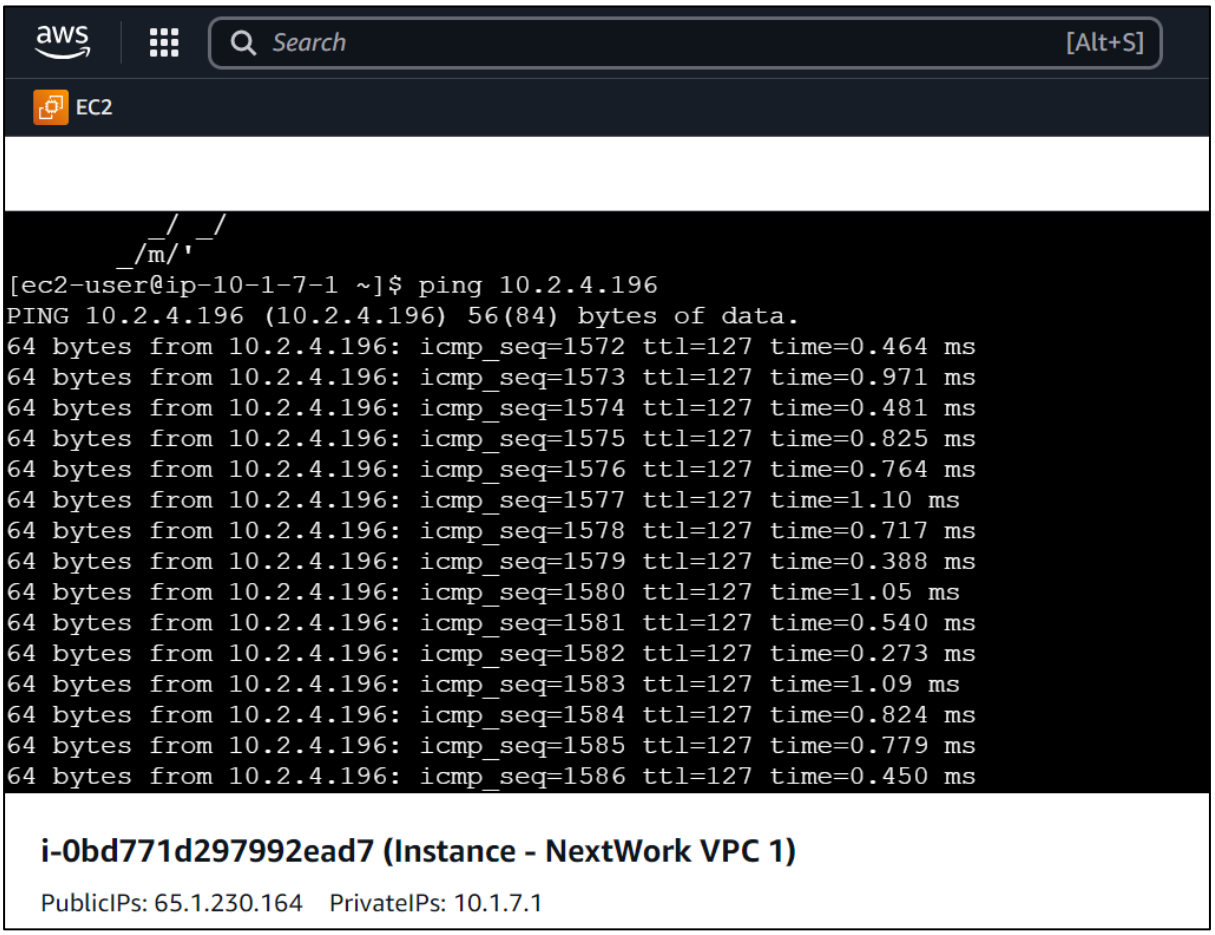
Let's fix this by letting inbound ICMP traffic from VPC 1.

- Select **Add new rule.**
- Change the **Type** to **All ICMP - IPv4**.
- Select **Edit inbound rule**
- Set the **Source** to traffic coming from VPC 1 - 10.1.0.0/16



- Select **Save rules.**

- Revisit the **EC2 Instance Connect** tab that's connected to Instance - NextWork VPC 1.
- Lots of new lines coming through in the terminal.

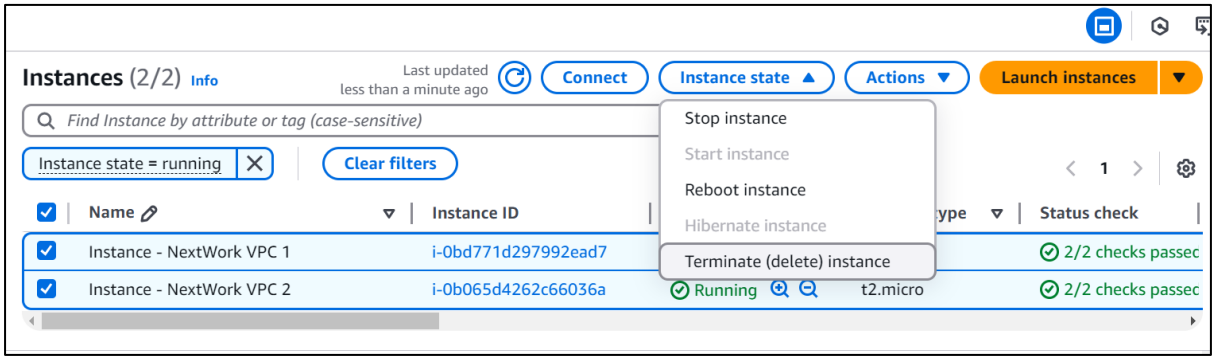


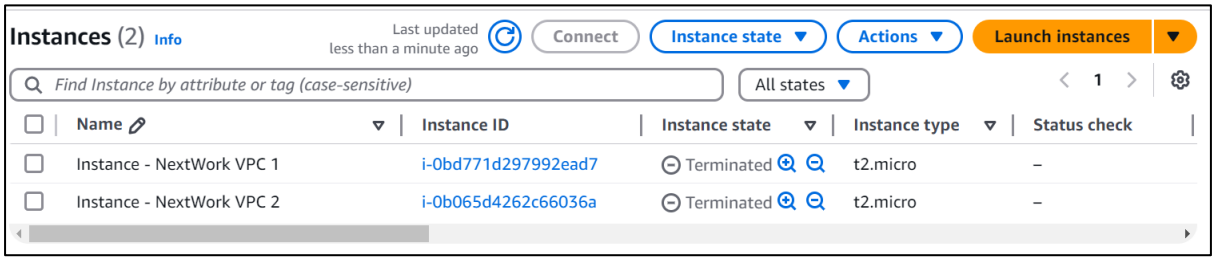
Congratulations! You've set up a peering architecture that connects VPC 1 to VPC 2 AND validated it with ping

Delete Your Resources

Delete your EC2 Instances

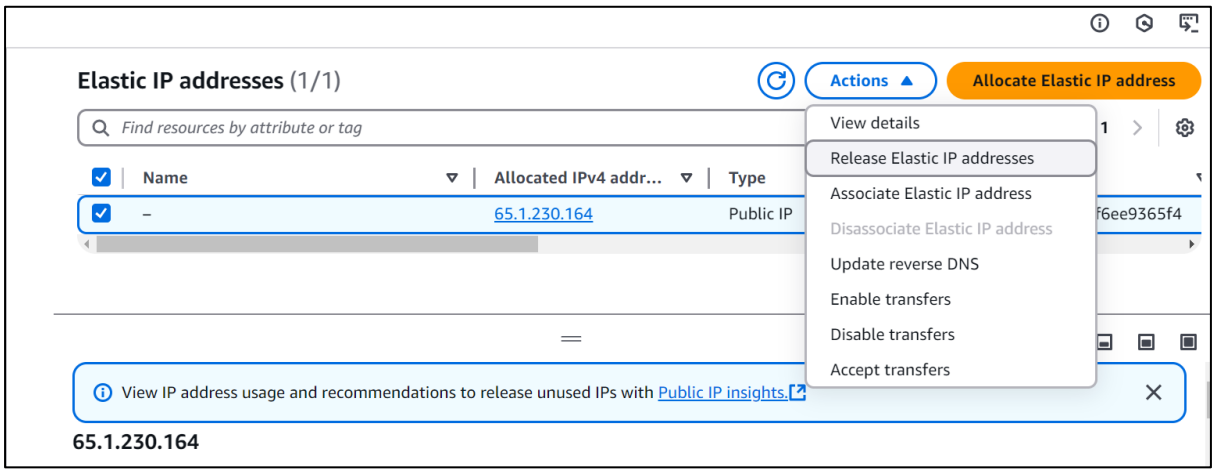
- Head back to the **Instances** page of your EC2 console.
- Select the checkboxes next to **Instance - NextWork VPC 1** and **Instance - NextWork VPC 2.**
- Select **Instance state,** then select **Terminate Instance.**
- Select **Terminate**.

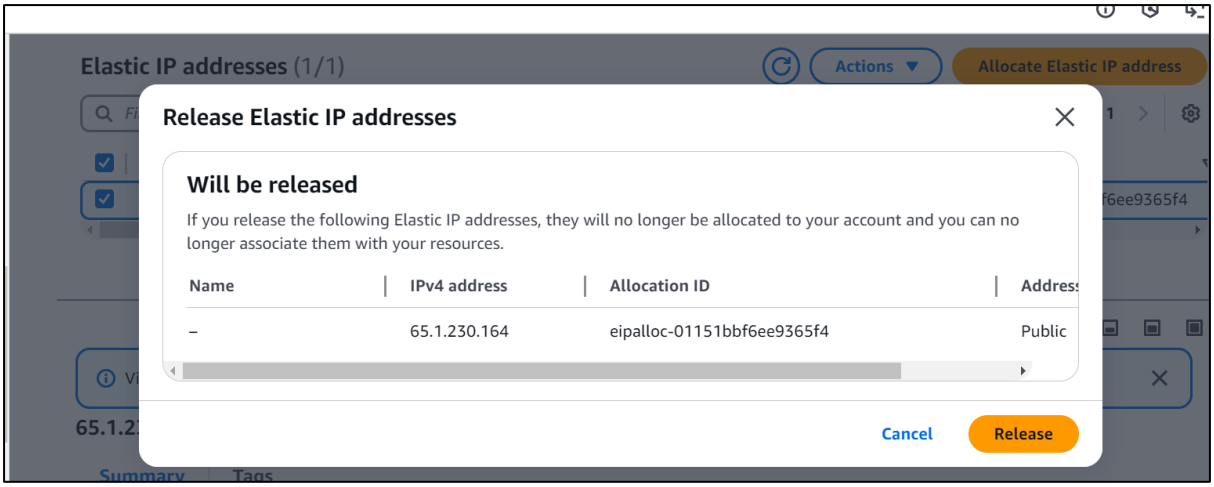




Delete your Elastic IP address

- Select **Elastic IPs** from the left hand navigation panel.
- Select the IP address you've created.
- Select Actions, then **Release Elastic IP addresses.**

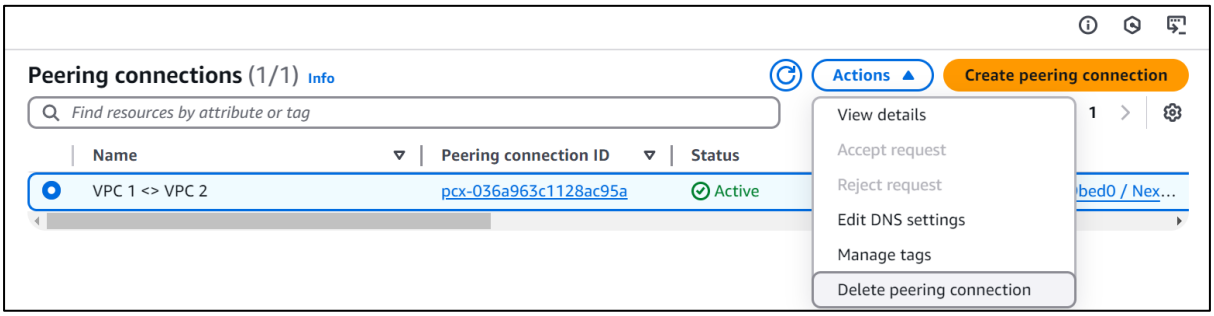




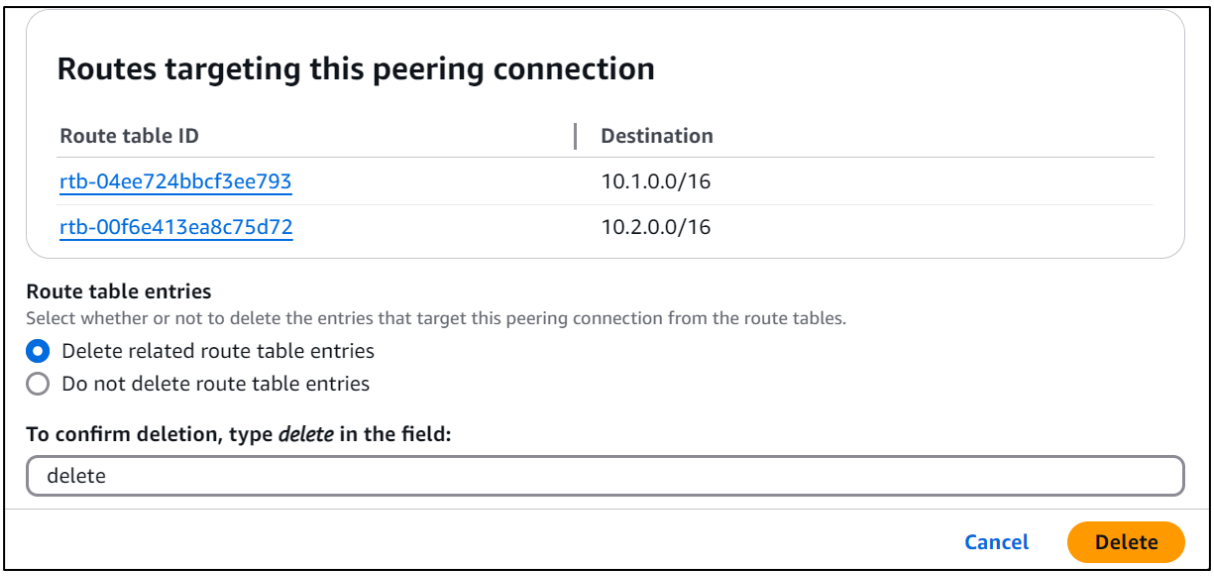
- Select **Release.**

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.**

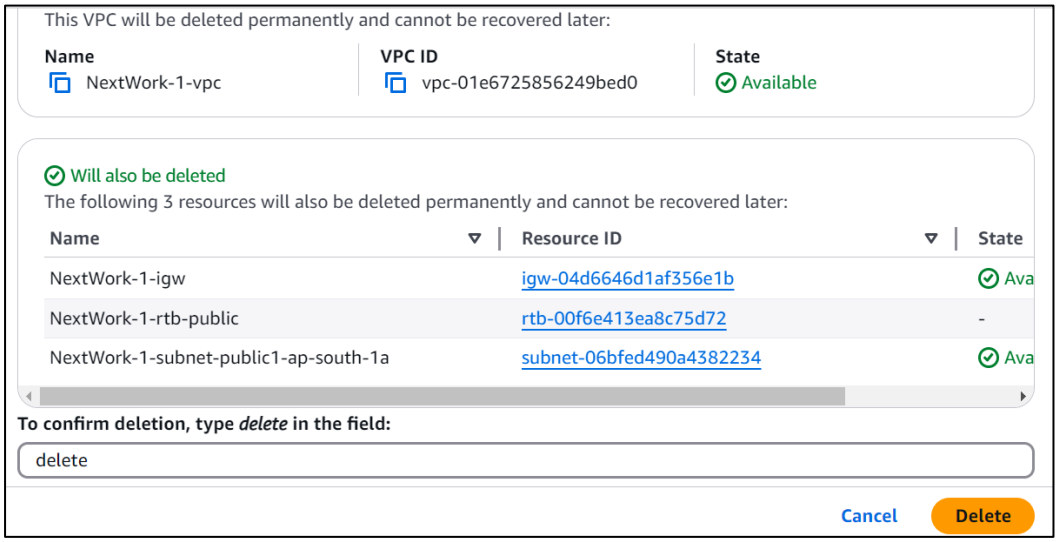


- Select the checkbox to **Delete related route table entries**.
- Type delete in the text box and click **Delete.**



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:

1. Subnets 2.Route tables 3. Internet gateways 4. Network ACLs 5.Security groups

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