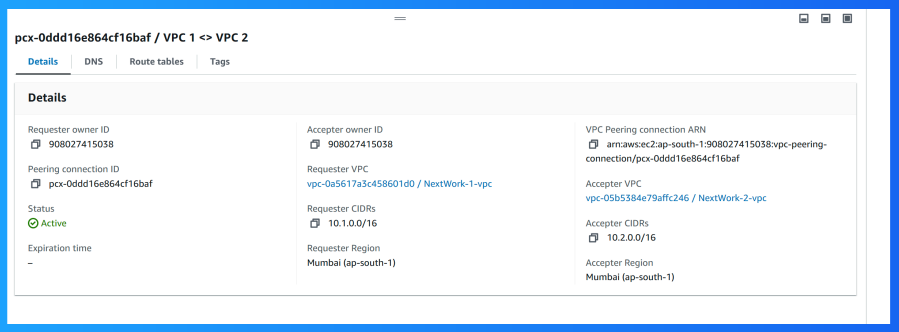
NextWork.org

VPC Peering

MA mallangiharinathreddy727@gmail.com



Introducing Today's Project!

What is Amazon VPC?

Amazon VPC is AWS foundational networking service that lets us create our own networks, control flow and security and organize our resources into public/private subnets.

How I used Amazon VPC in this project

I used Amazon VPC to set a multi VPC architecture (we set up two VPCs), create a peering connection between them AND update security group rules to run successful connectivity test.

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

I didn't expect to need a public IPV4 address for EC2 Instance connect also didn't expect that Elastic Ip's can assign static public IPV4 addresses to resources.

This project took me...

It takes 2 HOUR and 45 Minutes to complete this project.

In the first part of my project...

Step 1 - Set up my VPC

In this step we are using VPC resource map/launch wizard to create TWO VPCs and their components in just minutes.

Step 2 - Create a Peering Connection

I am setting up a VPC peering connection, which is a VPC component designed to directly connect two VPCs together.

Step 3 - Update Route Tables

Here we are updating our Route Tables to connect with VPC peering - If you want to share your resources, then you want make route for the resources to reach the destination. (another VPC)

Step 4 - Launch EC2 Instances

we are launching Ec2 instance in each of the VPCs (VPC1 AND VPC2) so that we can directly connect with our instances later and test our VPC peering connection.

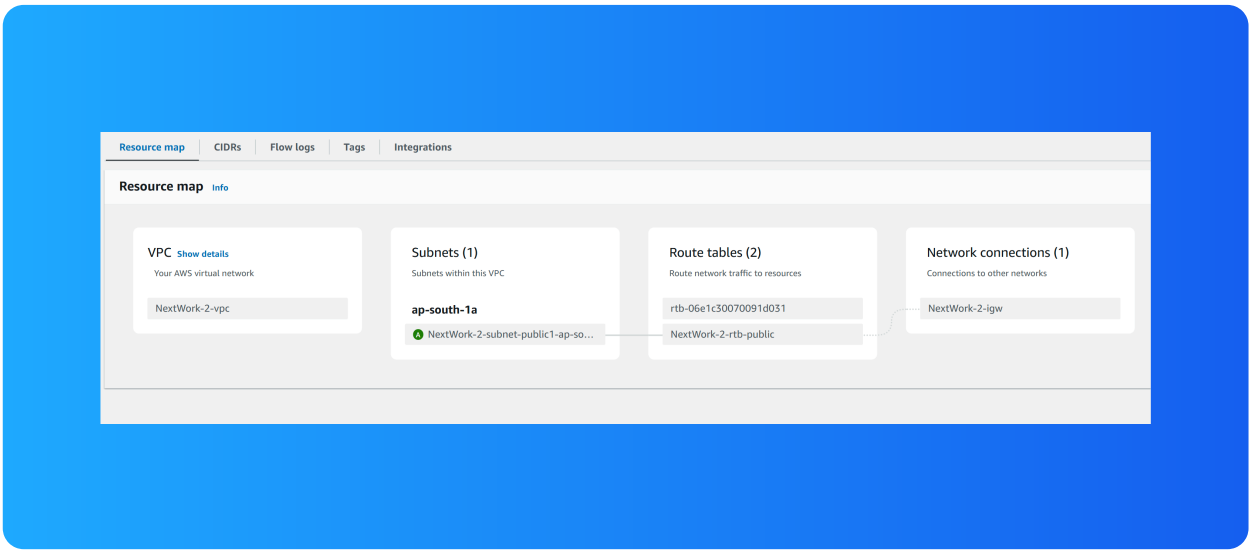
Multi-VPC Architecture

I started my project by launching two VPCs - they have unique CIDR blocks and they each have one public subnet.

The CIDR blocks for VPCs 1 and 2 are 10.1.0.0/16 and 10.2.0.0/16 respectively. They have to be unique because once you set up a VPC peering connection (between two VPCs), route tables need unique addresses for correct routing across VPCs.

I also launched 2 EC2 instances

I didn't set up key pairs for these EC2 instances because we are using EC2 instance connect to directly connect with our EC2 instance later in this project, which handles key pair creation and management for us.



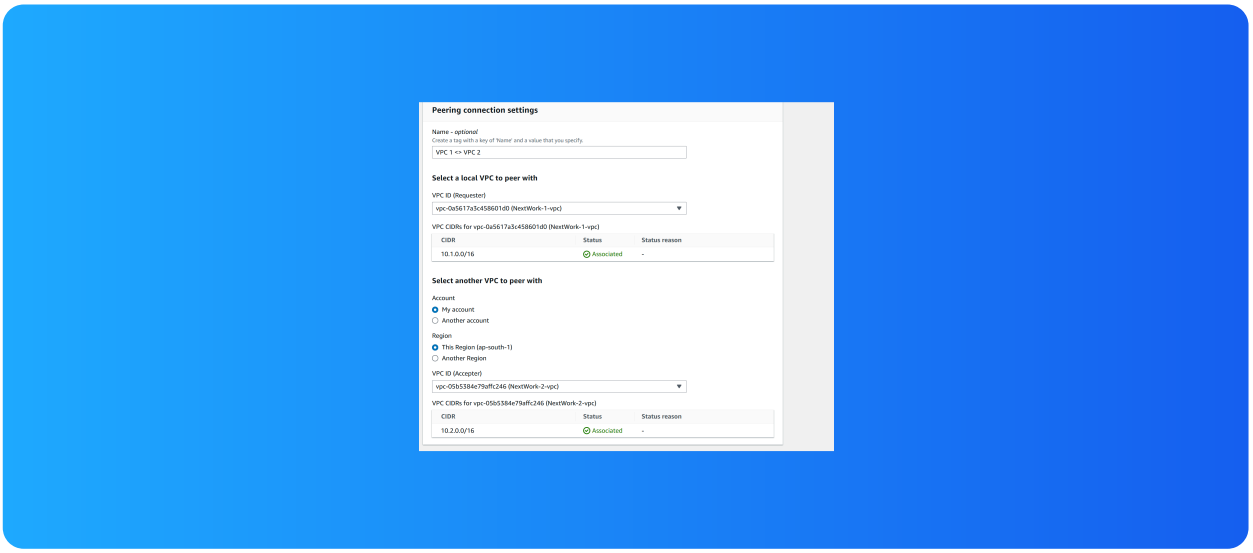
MA
mallangiharinathreddy72...
NextWork Student

VPC Peering

A VPC peering connection is the interaction between two VPCs with unique CIDR blocks.

VPCs would use peering connections to share the resources in private, without interacting with internet publicly.

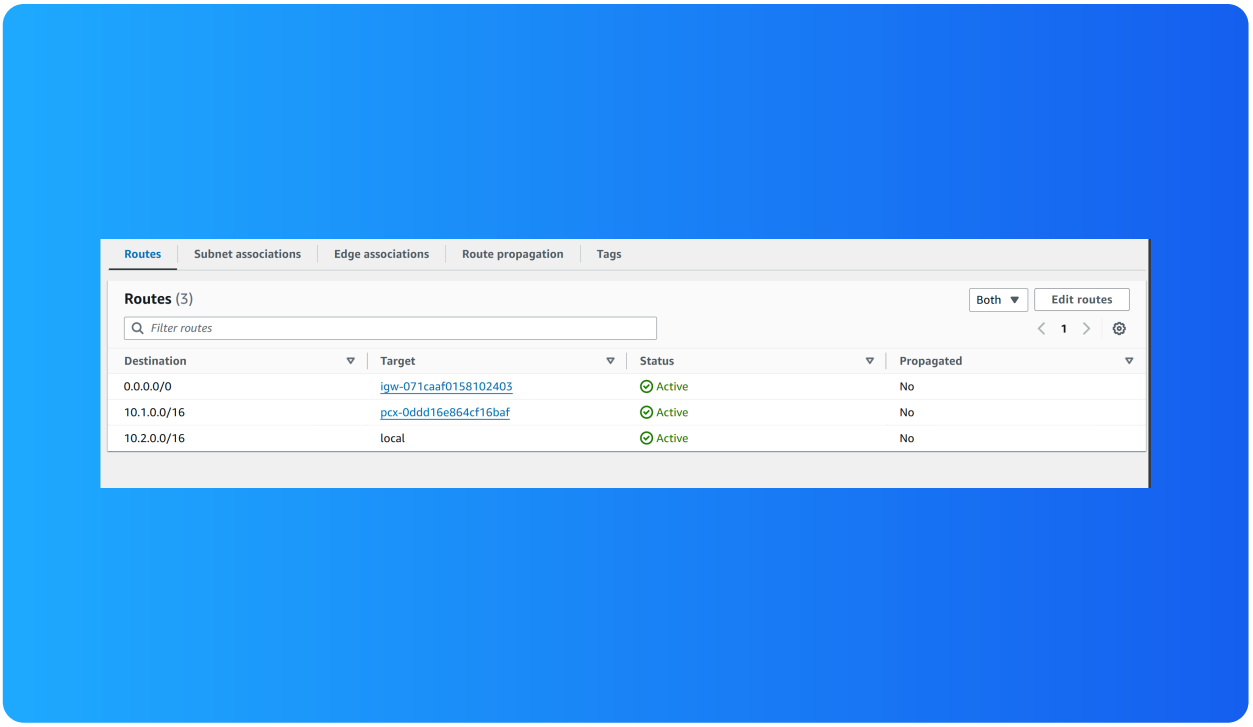
In VPC peering, the Accepter is the VPC that receives a peering connection request! The Accepter can either accept or decline the invitation. As the requester, they will be sending the other VPC an invitation to connect!



Updating route tables

Our VPCs' route tables need to be updated because the default route table doesn't have a route using the peering connection yet. This needs to be set up so that resources can be directed to the peering connection when trying to reach the other VPC.

My VPCs' new routes have a destination of the other VPCs CIDR block. The routes target was the peering connection we set up.



In the second part of my project...

Step 5 - Use EC2 Instance Connect

we are using EC2 instance connect to connect directly with our first EC2 instance. we need to do this because we need to use our EC2 instance for connectivity tests (to validate our VPC peering set up) later in this project.

Step 6 - Connect to EC2 Instance 1

We are reattempting our connection to Instance - NextWork VPC 1, and resolving another error preventing us from using EC2 Instance connect to directly connect with the Instance.

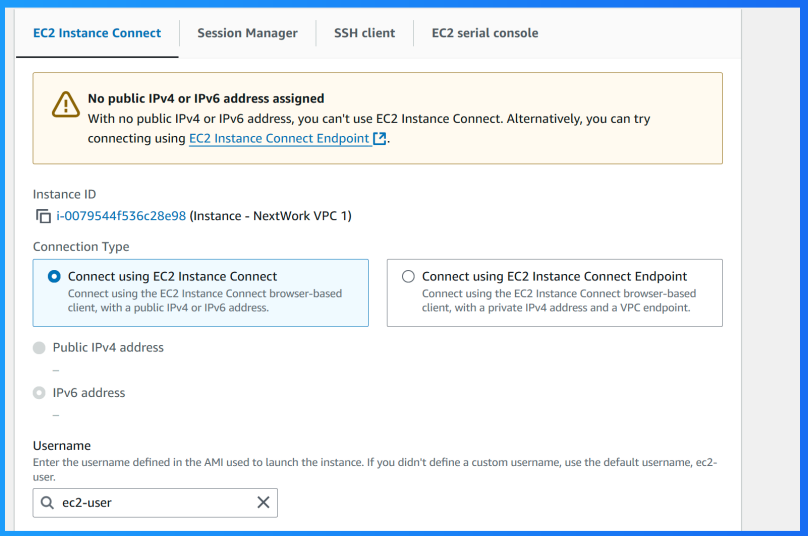
Step 7 - Test VPC Peering

we are going to use Instance - NextWork VPC 1 to attempt a direct connection with Instance - NextWork VPC 2 so that we can validate that our peering connection is set up properly.

Troubleshooting Instance Connect

Next, I used EC2 Instance Connect to directly connect with instance- NextWork VPC 1 just by using the AWS management console.

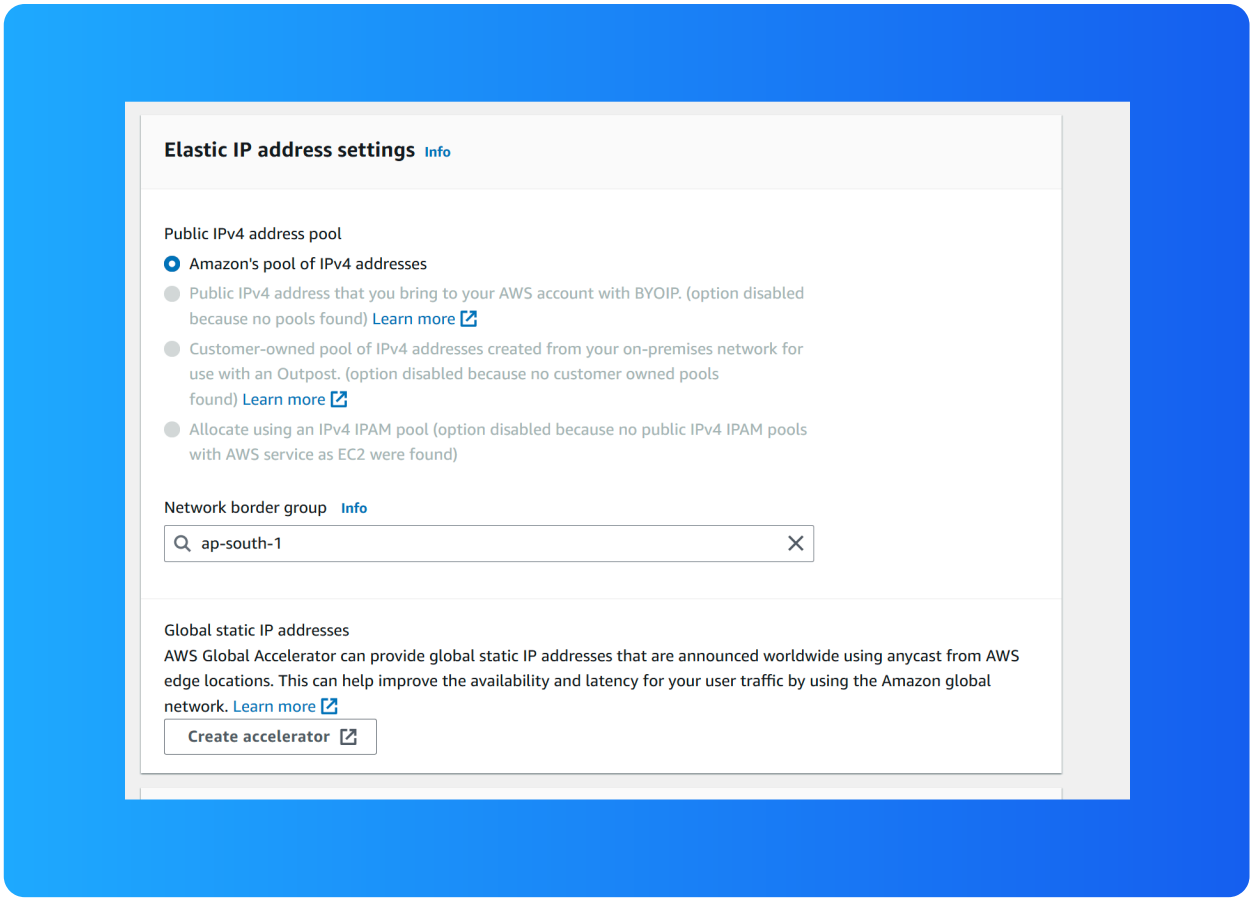
I was stopped from using EC2 Instance Connect as our instance did not have a public IPV4 address. In order for EC2 instance connect to work, the EC2 instance must have public IPV4 address and be in a public subnet.



Elastic IP addresses

To resolve this error, I set up Elastic IP addresses. Elastic IP addresses are static public IPV4 addresses that we can request for our AWS account, and then delicate to specific resources .

Associating an Elastic IP address resolved the error because it gives our EC2 instance a public IP address, fulfilling the requirements for Instance connect to work!

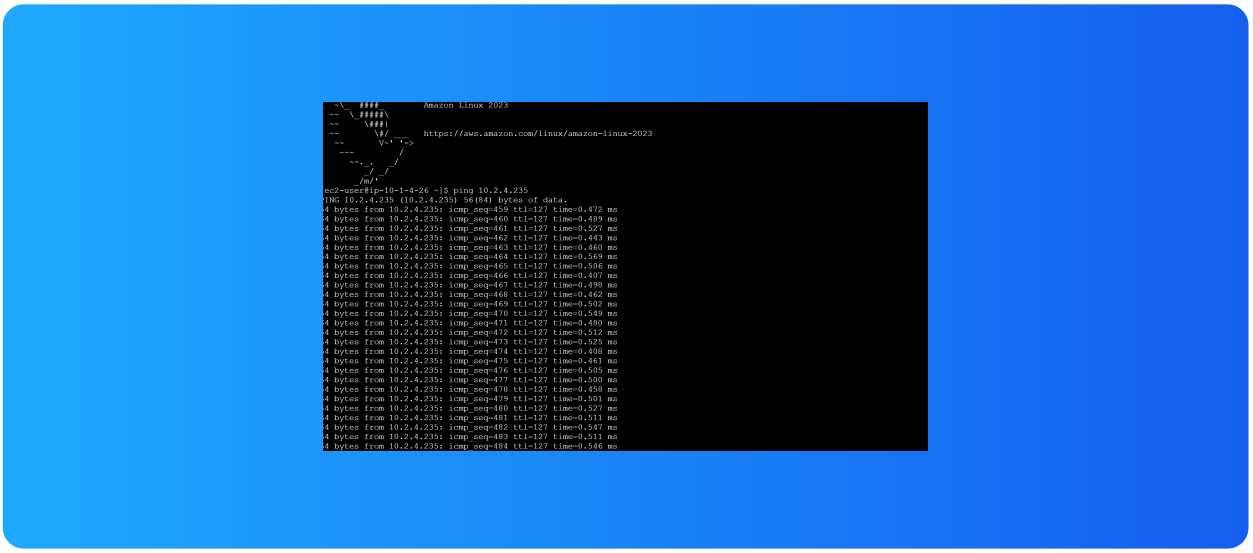


Troubleshooting ping issues

To test VPC peering, I ran the command -ping 10.2.4.235

A successful ping test would validate my VPC peering connection - this ping test would not get ANY replies from the other EC2 Instance if the connection didn't successfully connect our two VPCs.Getting ping replies= peering connection setup properly

I had to update my second EC2 instance's security group because it was not letting in ICMP traffic- which is the traffic type of a ping message. I added a new rule that allows ICMP traffic coming in from any resource in VPC2.



Everyone should be in a job they love.

Check out nextwork.org for more projects