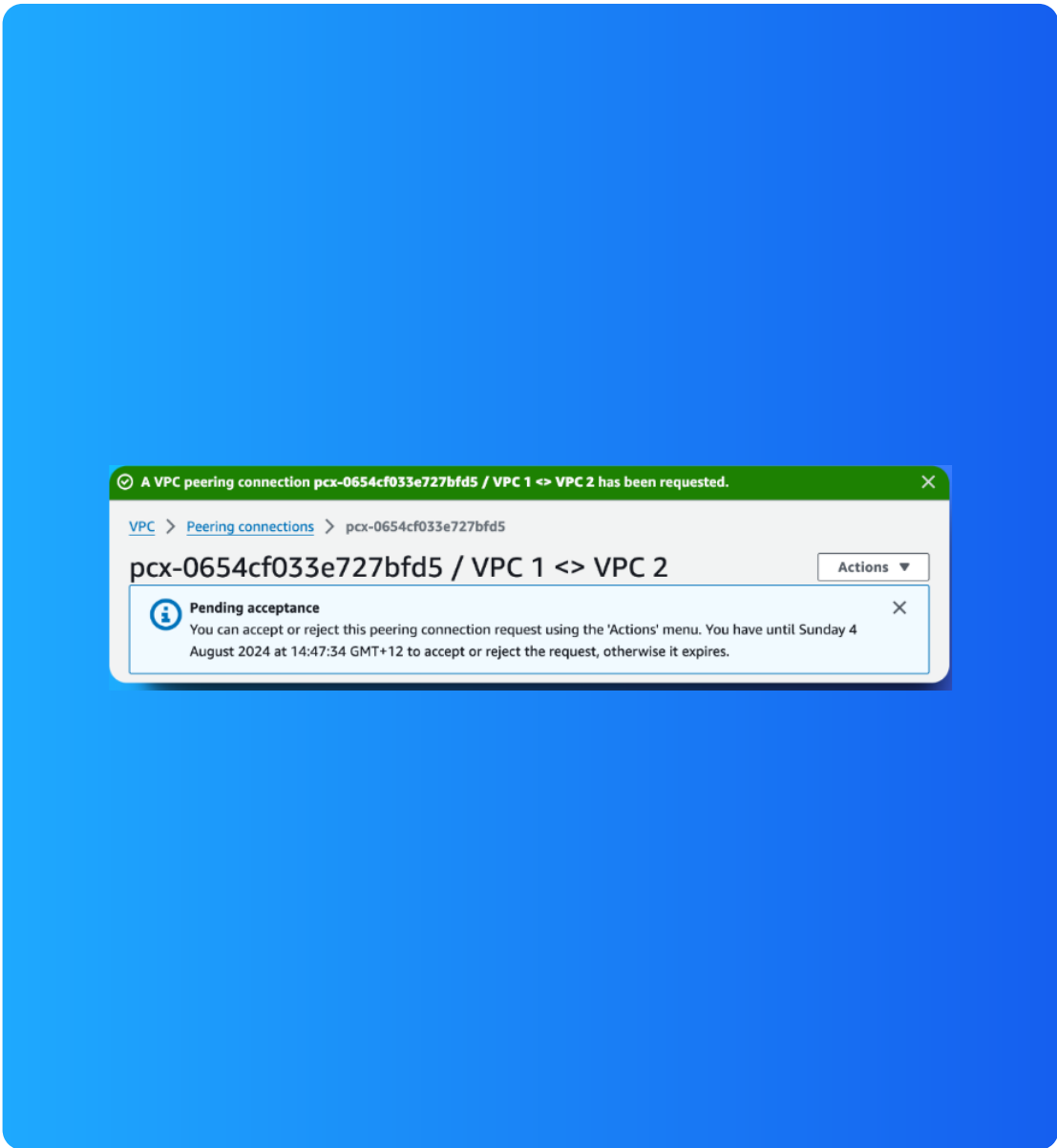




VPC Peering

 mmoorej24@gmail.com





mmoorej24@gmail.com
NextWork Student

NextWork.org

Introducing Today's Project!

What is Amazon VPC?

Amazon Virtual Private Cloud is a virtual network that resembles a traditional network that you'd operate in a data center. It gives you full control over your virtual networking environment, including resource placement, connectivity, and security.

How I used Amazon VPC in this project

I used Amazon VPC for peering two VPC so that the resources can connect to each other.

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

I didn't expect this project to take as long as it did. This was a complicated project but gave me an understanding of how much control you can have on the VPC and its' resources.

This project took me...

This project took me about 2 hours and 45 mins.



In the first part of my project...

Step 1 - Set up my VPC

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

Step 2 - Create a Peering Connection

We are setting up a VPC Peering Connection, which is a VPC component designed to directly connect two VPCs together.

Step 3 - Update Route Tables

In this step we are going to route traffic from VPC 1 to VPC 2 and VPC 2 to VPC 1. This way they will be able to communicate with each other.

Step 4 - Launch EC2 Instances

We are launching an EC2 instance in each of the VPCs, so that we can directly connect with our instances later and test our VPC peering connection.



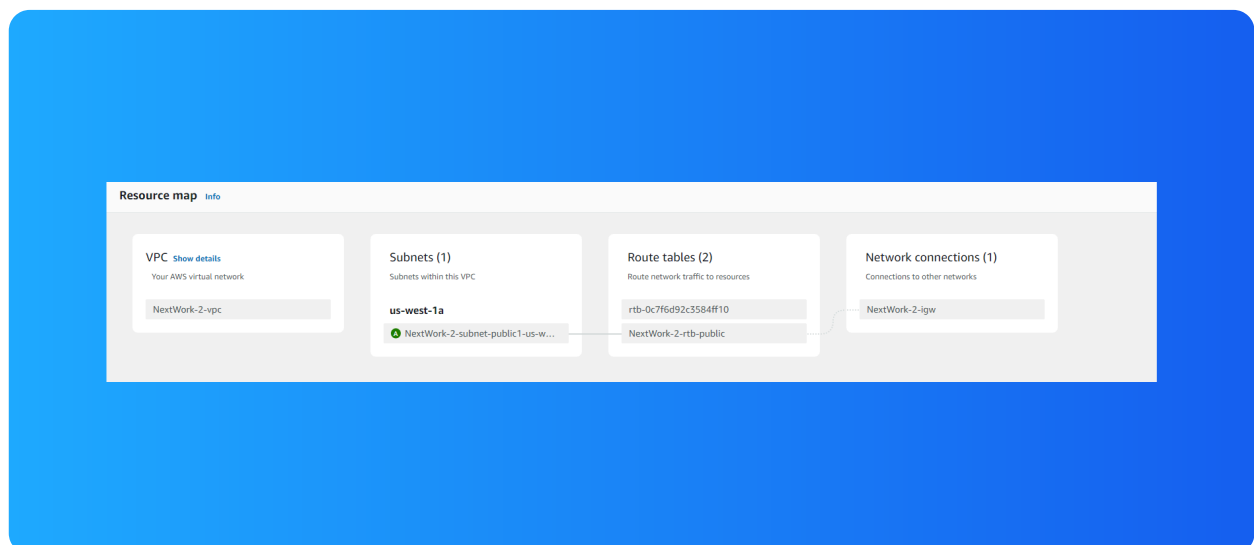
Multi-VPC Architecture

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.



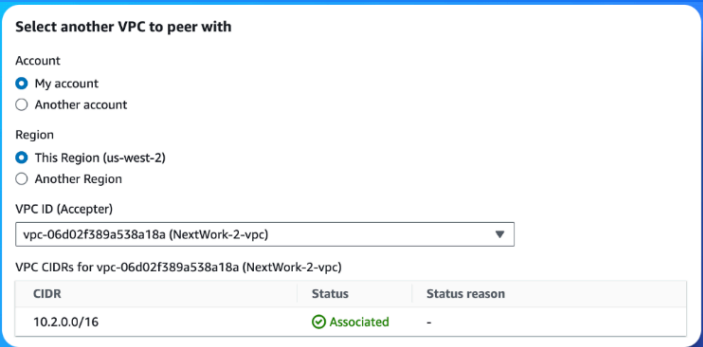


VPC Peering

A VPC peering connection is a direct connection between two VPCs! A peering connection lets VPCs and their resources route traffic between them using their private IP addresses.

VPCs would use peering connections to let VPCs and their resources route traffic between them using their private IP addresses.

The difference between a Requester and an Acceptor in a peering connection is the Requester is the VPC that initiates a peering connection, and the the Acceptor is the VPC that receives a peering connection request!



The screenshot shows the AWS console interface for selecting a VPC to peer with. It includes radio buttons for account and region selection, a dropdown for VPC ID, and a table of VPC CIDRs with their status.

CIDR	Status	Status reason
10.2.0.0/16	Associated	-



Updating route tables

My VPCs' route tables need to be updated because the default table doesn't have a route using the peering connection yet - this needs to be setup 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 VPC's CIDR block. The routes' target was the peering connection we set up.

Destination	Target	Status	Propagated
0.0.0.0/0	igw-04191cb09ea01a58a	Active	No
10.1.0.0/16	pcx-08326efe3581e3491	Active	No
10.2.0.0/16	local	Active	No



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 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

I am reattempting our connection to Instance Connect - VPC 1, and resolving another error preventing us from using EC2 Instance Connect to directly connect to the instance.

Step 7 - Test VPC Peering

We are going to use the Instance - NextWork VPC 1 to attempt a direct connection with the 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 didn't have a public IPv4 address. In order for EC2 instance to work, the EC2 instance must have a public IPv4 address and be in a public subnet.

The screenshot shows the AWS Management Console interface for EC2 Instance Connect. The navigation bar includes 'EC2 Instance Connect', 'Session Manager', 'SSH client', and 'EC2 serial console'. Two error messages are displayed in yellow boxes:

- EC2 Instance Connect service IP addresses are not authorized**
Port 22 (SSH) is authorized in [your security group](#). However, to use EC2 Instance Connect, it is recommended to also authorize port 22 for the EC2 Instance Connect service IP addresses in your Region: 13.52.6.112/29. [Learn more](#).
- No public IPv4 address assigned**
With no public IPv4 address, you can't use EC2 Instance Connect. Alternatively, you can try connecting using [EC2 Instance Connect Endpoint](#).



Elastic IP addresses

To resolve this error, I set up Elastic IP addresses. Elastic IP addresses are public static IPv4 addresses that we can request for our AWS Account, and then delegate 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.

Allocate Elastic IP address Info

Elastic IP address settings Info

Public IPv4 address pool

- Amazon's pool of IPv4 addresses
- Public IPv4 address that you bring to your AWS account with BYOIP. (option disabled because no pools found) [Learn more](#)
- Customer-owned pool of IPv4 addresses created from your on-premises network for use with an Outpost. (option disabled because no customer owned pools found) [Learn more](#)

Global static IP addresses

AWS Global Accelerator can provide global static IP addresses that are announced worldwide using anycast from AWS edge locations. This can help improve the availability and latency for your user traffic by using the Amazon global network. [Learn more](#)

[Create accelerator](#)

Tags - optional

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

[Add new tag](#)

You can add up to 50 more tag



Troubleshooting ping issues

To test VPC peering, I ran the command ping and the private IPv4 address of the other EC2 instance in VPC 2.

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 peering connection did not successfully connect our two VPCs. Getting ping replies = peering connection.

I had to update my second EC2 instance's security group because it wasn't letting in ICMP traffic which is the traffic type of the ping message. I added a new rule that allows ICMP traffic coming in from any resource in VPC 2.

```
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

last login: Mon Aug 5 22:07:37 2024 from 13.52.6.115
[ec2-user@ip-10-1-2-29 ~]$ ping 10.2.9.71
PING 10.2.9.71 (10.2.9.71) 56(84) bytes of data:
64 bytes from 10.2.9.71: icmp_seq=232 ttl=127 time=0.709 ms
64 bytes from 10.2.9.71: icmp_seq=233 ttl=127 time=0.511 ms
64 bytes from 10.2.9.71: icmp_seq=234 ttl=127 time=0.496 ms
64 bytes from 10.2.9.71: icmp_seq=235 ttl=127 time=0.480 ms
64 bytes from 10.2.9.71: icmp_seq=236 ttl=127 time=0.453 ms
64 bytes from 10.2.9.71: icmp_seq=237 ttl=127 time=0.564 ms
64 bytes from 10.2.9.71: icmp_seq=238 ttl=127 time=0.457 ms
64 bytes from 10.2.9.71: icmp_seq=239 ttl=127 time=0.472 ms
64 bytes from 10.2.9.71: icmp_seq=240 ttl=127 time=0.467 ms
64 bytes from 10.2.9.71: icmp_seq=241 ttl=127 time=0.405 ms
64 bytes from 10.2.9.71: icmp_seq=242 ttl=127 time=0.463 ms
64 bytes from 10.2.9.71: icmp_seq=243 ttl=127 time=0.534 ms
64 bytes from 10.2.9.71: icmp_seq=244 ttl=127 time=0.507 ms
64 bytes from 10.2.9.71: icmp_seq=245 ttl=127 time=0.504 ms
^C
--- 10.2.9.71 ping statistics ---
245 packets transmitted, 14 received, 94.2857% packet loss, time 253782ms
rtt min/avg/max/mdev = 0.405/0.501/0.709/0.068 ms
[ec2-user@ip-10-1-2-29 ~]$
```



NextWork.org

Everyone should be in a job they love.

Check out nextwork.org for
more projects

