



[nextwork.org](https://nextwork.org)

# VPC Peering



Mohamed Galole

### Accept VPC peering connection request [Info](#)

Are you sure you want to accept this VPC peering connection request? (pcx-07e0324bfc79ba97 / VPC 1 <> VPC 2)

<b>Requester VPC</b> vpc-0a9f8dc29c35fea78 / NextWork-1-vpc	<b>Accepter VPC</b> vpc-00acf89c8d01de49e / NextWork-2-vpc	<b>Requester CIDRs</b> 10.1.0.0/16
<b>Accepter CIDRs</b> -	<b>Requester Region</b> N. Virginia (us-east-1)	<b>Accepter Region</b> N. Virginia (us-east-1)
<b>Requester owner ID</b> 289474598410(This account)	<b>Accepter owner ID</b> 289474598410(This account)	

[Cancel](#) [Accept request](#)



# Introducing Today's Project!

## What is Amazon VPC?

Amazon VPC (Virtual Private Cloud) is a service that lets you create a private, isolated network within the AWS cloud, where you can launch and manage AWS resources like EC2 instances.

## How I used Amazon VPC in this project

In today's project, I used Amazon VPC to create a secure, isolated network environment for my AWS resources. Specifically, I used it to: launched EC2 instances in a controlled subnet, managed internet access, controlled traffic, tested connectivity.

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

One thing I didn't expect in this project was how many small network settings must work together for everything to connect properly.

## This project took me...

This project took me one hour



## In the first part of my project...

### Step 1 - Set up my VPC

In this step, I will Create two VPCs from scratch Use the visual VPC resource map to create your VPCs supa fast because it lets me see all the components subnets, route tables, gateways at a glance, reducing errors and speeding up the process.

### Step 2 - Create a Peering Connection

In this step, I will Set up a connection link between my VPCs because because it allows resources in VPC-1 and VPC-2 to communicate with each other securely.

### Step 3 - Update Route Tables

In this step, I will update the route tables in both VPCs to allow traffic to flow between them because, even though the VPC peering connection exists, each VPC still needs explicit routes to know how to reach the other network.

### Step 4 - Launch EC2 Instances

In this step, I will launch an EC2 instance in each VPC because I need a way to test and verify that my VPC peering connection is working correctly.



## Multi-VPC Architecture

I started my project by launching a new VPC with the default settings for IPv6, tenancy, and DNS options. I created 1 public subnet in 1 Availability Zone, and I didn't create any private subnets, NAT gateways, VPC endpoints to keep things simple.

The CIDR blocks for VPCs 1 and 2 are 10.1.0.0/16 and 10.2.0.0/16. They have to be unique because each VPC is a separate network, and overlapping CIDR blocks would cause IP address conflicts.

I also launched 2 EC2 instances

I didn't set up key pairs for these EC2 instances as setting up a key pair was a key step for learning how SSH and directly accessing an EC2 instance works. AWS actually manages a key pair for us.





## VPC Peering

A VPC peering connection is a networking link between two VPCs that allows them to communicate with each other privately using their internal IP addresses.

VPCs would use peering connections to communicate and share resources privately without using the public internet.

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

☰ VPC > Peering connections > Create peering connection

Select another VPC to peer with

**Account**

☒ My account  
☐ Another account

**Region**

☒ This Region (us-east-1)  
☐ Another Region

**VPC ID (Acceptor)**

vpc-00acf89c8d01de49e (NextWork-2-vpc)

**VPC CIDRs for vpc-00acf89c8d01de49e (NextWork-2-vpc)**

CIDR	Status	Status reason
10.2.0.0/16	Associated	-



## Updating route tables

After accepting a peering connection, my VPCs' route tables need to be updated because the VPCs don't automatically know how to reach each other's networks.

My VPCs' new routes have a destination of 10.1.0.0/16 and 10.2.0.0/16 The routes' target was pcx-07e0324befc79ba97

The screenshot shows the AWS Management Console interface for a route table. A green notification banner at the top states: "Updated routes for rtb-061a2ce73978707fa / NextWork-2-rtb-public successfully". Below this, the "Routes" tab is selected, showing a list of three routes. The routes are: 0.0.0.0/0 (target: igw-0a87a684848bf6..., status: Active), 10.1.0.0/16 (target: pcx-07e0324befc79b..., status: Active), and 10.2.0.0/16 (target: local, status: Active). The "Propagated" column shows "No" for all routes. The "Route Origin" column shows "Create Route" for the first two and "Create Route Table" for the third. A search bar and pagination controls are also visible.

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	<a href="#">igw-0a87a684848bf6...</a>	Active	No	Create Route
10.1.0.0/16	<a href="#">pcx-07e0324befc79b...</a>	Active	No	Create Route
10.2.0.0/16	local	Active	No	Create Route Table



## In the second part of my project...

### Step 5 - Use EC2 Instance Connect

In this step, I will use EC2 Instance Connect to access my first EC2 instance and troubleshoot any connection errors because it's important to verify that my instance is reachable and properly configured within the VPC.

### Step 6 - Connect to EC2 Instance 1

In this step, I will Use EC2 Instance Connect to connect to Instance 1

### Step 7 - Test VPC Peering

In this step, I get Instance 1 to send test messages to Instance 2 and solve connection errors until Instance 2 is able to send messages back.





# Troubleshooting Instance Connect

Next, I used EC2 Instance Connect to securely access my first EC2 instance directly from the AWS Management Console without needing an SSH client or key file.

I was stopped from using EC2 Instance Connect as I got an error "Error establishing SSH connection to your instance. Try again later"





## Elastic IP addresses

To resolve this error, I set up Elastic IP addresses. Elastic IP addresses are public IPv4 addresses provided by AWS that you can associate with your EC2 instances.

Associating an Elastic IP address resolved the error because they provided a fixed public IP that I could use to access my EC2 instance from the internet.

✓ Elastic IP address associated successfully.  
Elastic IP address 3.229.254.218 has been associated with instance i-0a7b3d84bc5dba368

3.229.254.218

Actions Associate Elastic IP address

**Summary**

Allocated IPv4 address  
3.229.254.218

Association ID  
eipassoc-055dcad6a2ed81f17

Network interface ID  
eni-0a08448a266be8b4f

Address pool  
Amazon

Type  
Public IP

Scope  
VPC

Network interface owner account ID  
289474598410

Network border group  
us-east-1

Allocation ID  
eipalloc-036927d6087d1c376

Associated instance ID  
i-0a7b3d84bc5dba368

Public DNS  
ec2-3-229-254-218.compute-1.amazonaws.com

Service managed  
-

Reverse DNS record  
-

Private IP address  
10.1.11.70

NAT Gateway ID  
-

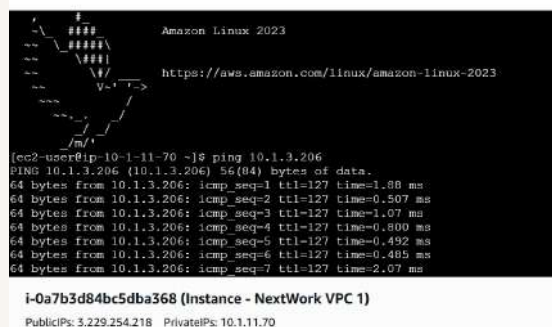


## Troubleshooting ping issues

To test VPC peering, I ran the command `ping 10.1.3.206`. This command sends ICMP packets from my EC2 instance in one VPC to the EC2 instance in the other VPC to check if they can communicate through the peering connection.

A successful ping test would validate my VPC peering connection because it proves that network traffic can flow between the two VPCs using their private IP addresses.

I had to update my second EC2 instance's security group because by default, security groups block all inbound traffic except what's explicitly allowed. I added a new rule that Allowed traffic from VPC-1's CIDR block (10.1.0.0/16).



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

[ec2-user@ip-10-1-11-70 ~]$ ping 10.1.3.206
PING 10.1.3.206 (10.1.3.206) 56(84) bytes of data:
64 bytes from 10.1.3.206: icmp_seq=1 ttl=127 time=1.88 ms
64 bytes from 10.1.3.206: icmp_seq=2 ttl=127 time=0.507 ms
64 bytes from 10.1.3.206: icmp_seq=3 ttl=127 time=1.07 ms
64 bytes from 10.1.3.206: icmp_seq=4 ttl=127 time=0.800 ms
64 bytes from 10.1.3.206: icmp_seq=5 ttl=127 time=0.492 ms
64 bytes from 10.1.3.206: icmp_seq=6 ttl=127 time=0.489 ms
64 bytes from 10.1.3.206: icmp_seq=7 ttl=127 time=2.07 ms

i-0a7b3d84bc5dba368 (Instance - NextWork VPC 1)
PublicIPs: 3.229.254.218 PrivateIPs: 10.1.11.70
```



[nextwork.org](https://nextwork.org)

# The place to learn & showcase your skills

Check out [nextwork.org](https://nextwork.org) for more projects

