

# How to Set Up VPC Peering in AWS: A Step-by-Step Guide



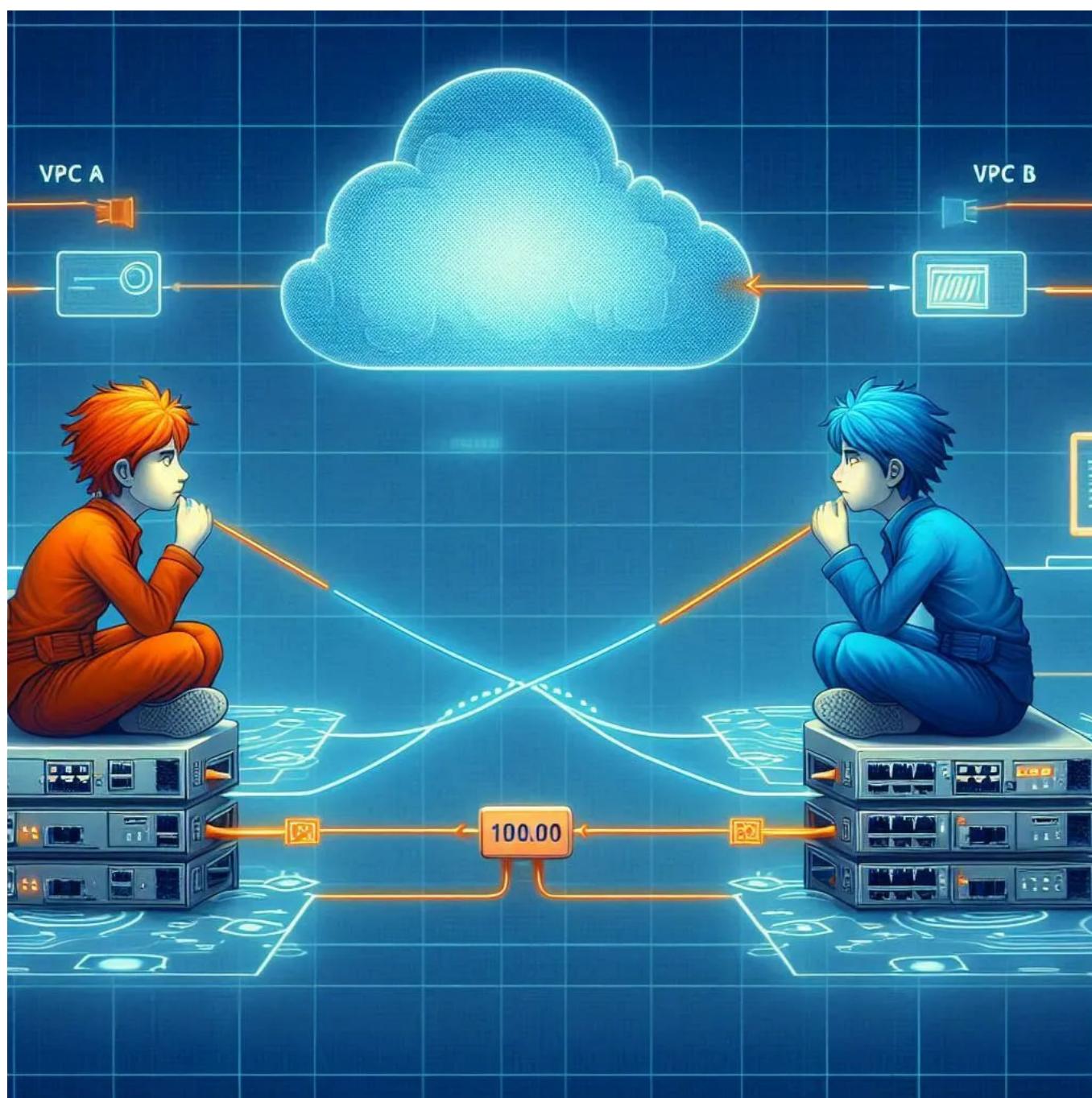
Deepak Bajaj

Follow

4 min read · Oct 18, 2024



In modern cloud architectures, network isolation is key to maintaining security and performance. Amazon Web Services (AWS) provides a solution called Virtual Private Cloud (VPC), allowing you to create logically isolated networks in the cloud. One common scenario is the need for communication between different VPCs. VPC Peering is AWS's answer for securely connecting two VPCs together, even across different AWS regions or accounts.



In this blog, I will walk through how to set up VPC peering, explain its use cases, and offer best practices for successful implementation.

## What is VPC Peering?

VPC Peering is a networking connection between two VPCs that allows them to communicate as if they were part of the same network. This connection is

bidirectional and allows for a private and secure exchange of traffic without the need for internet gateways, VPNs, or additional bandwidth costs.

### **Common Use Cases:**

- **Multi-tier Application Deployments:** Different parts of your app can reside in separate VPCs for better security and management.
- **Cross-Account Resource Access:** When you have resources in different AWS accounts but need to share or communicate between them.
- **Region Isolation with Cross-Region Communication:** When resources need to stay in different regions for redundancy or compliance, but they must communicate securely.

### **Prerequisites**

Before starting the setup, ensure you have the following:

- Two VPCs that don't have overlapping CIDR blocks. VPC Peering does not support communication between VPCs with overlapping IP ranges.
- Appropriate permissions in IAM to create VPC Peering connections.
- VPC route tables configured to allow traffic between the VPCs.

## **Step-by-Step Guide to Set Up VPC Peering**

### **Step 1: Identify the VPCs to Peer**

You'll need to know the details (VPC IDs, region, etc.) of the two VPCs you want to peer. Let's assume:

- VPC A has CIDR 10.0.0.0/16
- VPC B has CIDR 10.1.0.0/16

Make sure these CIDR blocks don't overlap. Overlapping CIDR ranges will prevent successful peering.

### **Step 2: Create a VPC Peering Connection**

1. Navigate to VPC Dashboard.
  2. Log in to your AWS console and go to the VPC Dashboard.
  3. Create a Peering Connection
- On the left sidebar, select Peering Connections, and then click Create Peering Connection.
  - Fill in the required details:
    - Peering connection name: Give it a descriptive name.
    - VPC Requester: Select the VPC A (the initiator of the request).
    - VPC Acceptor: Choose the VPC B (the receiver).
    - If the VPC is in another account or region, input the Account ID and

region details of the acceptor VPC.

- Submit the Peering Request

Once filled in, click Create Peering Connection. The connection request is now pending and must be accepted by the other VPC.

Get Deepak Bajaj's stories in your inbox

Join Medium for free to get updates from this writer.

Enter your email

Subscribe

### **Step 3: Accept the VPC Peering Connection**

1. Go to Peering Connections.

2. On the VPC dashboard of the acceptor account, go to Peering Connections.

3. Accept the Request

Select the peering connection you created and click Actions, then select Accept Request. This will establish the peering connection between VPC A and VPC B.

### **Step 4: Update Route Tables**

For the two VPCs to communicate, you must update their route tables to allow traffic to flow through the peering connection.

#### **1. Navigate to Route Tables**

In the VPC console, select **Route Tables** from the left-hand menu.

#### **2. Update the Route for VPC A**

- Select the route table for VPC A.
- Click **Routes > Edit Routes > Add Route**.
- For **Destination**, enter the CIDR block of VPC B (`10.1.0.0/16`).
- For **Target**, choose the Peering Connection ID created earlier.

#### **1. Update the Route for VPC B**

Repeat the same steps for VPC B:

- Destination: Enter the CIDR block of VPC A (`10.0.0.0/16`).
- Target: Use the Peering Connection ID.

### **Step 5: Update Security Groups**

Both VPCs have their own security groups, which act as firewalls for controlling traffic. To ensure proper communication, you need to update security group rules to allow traffic from the peered VPC.

#### **1. Navigate to Security Groups**

In the VPC console, select **Security Groups** from the left-hand menu.

#### **2. Modify Rules for VPC A**

- Select the security group associated with your instances in VPC A.
- Add inbound rules that allow traffic from the CIDR block of VPC B.
- Repeat for outbound rules if necessary.
- **Modify Rules for VPC B**  
Similarly, modify the security groups for VPC B to allow traffic from VPC A.

## Best Practices for VPC Peering

- 1. Avoid Overlapping CIDR Blocks:** Always make sure your VPC CIDR blocks do not overlap to avoid routing issues.
- 2. Use Tags for Easy Management:** Name and tag your VPC peering connections for easier tracking.
- 3. Minimise the Number of Peering Connections:** VPC Peering is point-to-point, meaning if you have many VPCs that need to communicate, a mesh of peering connections may become complex. Consider using AWS Transit Gateway if you expect a large number of inter-VPC connections.

## Troubleshooting

- **Route Table Issues:** If the instances in the peered VPCs cannot communicate, check if the correct routes have been added to the route tables in both VPCs.
- **Security Group Restrictions:** Ensure that security groups on both sides allow inbound and outbound traffic from the peered VPC CIDR range.
- **Network ACLs:** If you are using network ACLs, make sure they are not blocking traffic between the two VPCs.

## Conclusion

Setting up VPC Peering is a powerful tool for securely connecting different VPCs within AWS or across regions. It enables the sharing of resources like databases or micro services between isolated networks while maintaining high security and control over traffic flow.

AWS

Vpc Peering

Cloud Computing

Amazon Web Services

Vpc



Written by Deepak Bajaj

0 followers · 1 following

Follow

Member Research Staff (Scientist) @ CRL

# No responses yet



Write a response

What are your thoughts?

## More from Deepak Bajaj



Deepak Bajaj

### Testcontainers—Testing a Web Application with MySQL

WHY Testcontainers

Oct 22, 2024



Deepak Bajaj

### Docker—More of Development tool than DevOps

We all are aware that Docker is a popular platform that allows developers to automate...

Oct 18, 2024



[See all from Deepak Bajaj](#)

## Recommended from Medium



AWS Identity and Access Management (IAM)

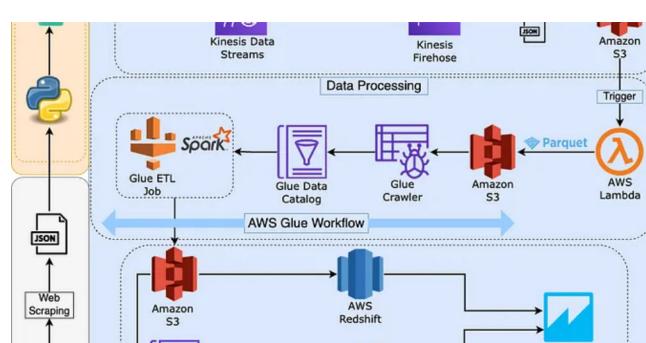
FromCodeToCloud

### The Most Misunderstood AWS Concept: IAM Policies Explained i...

IAM is backbone of Cloud Security

Dec 11

6



Dogukan Ulu

### AWS Cloud Engineering Project—Part 1—AWS Glue, Lambda,...

Repository

Dec 6

20





## 10 Most Common Multi-Select Questions for AWS Cloud Practitioner (CLF-C02) –

In AWS in Plain English by TechyTrooper

## 10 Most Common Multi-Select Questions for AWS Cloud...

I've helped quite a few people prepare for the AWS Cloud Practitioner (CLF-C02) exam, an...

⭐ Dec 12 ⌗ 72



In Beyond Localhost by The Speedcraft Lab

## I Failed 47 System Design Interviews—Then One Netflix...

A single shift in how I framed scalability turned rejection into three competing offers...

⭐ Nov 5 ⌗ 2.1K ⌛ 56



In Towards AWS by Muhammad Yawar Malik

## My SRE Starter Pack: Tools and Practices I Wish I Knew Sooner

Why did nobody warn me that CloudWatch dashboards would become my second...

⭐ Jun 27 ⌗ 13



In Data Engineer Things by Yunus Gurguz

## Building an End-to-End Data Pipeline on AWS with Lambda, S3...

If you don't have any Medium membership, please click this link to read the article.

⭐ Dec 7 ⌗ 85



See more recommendations