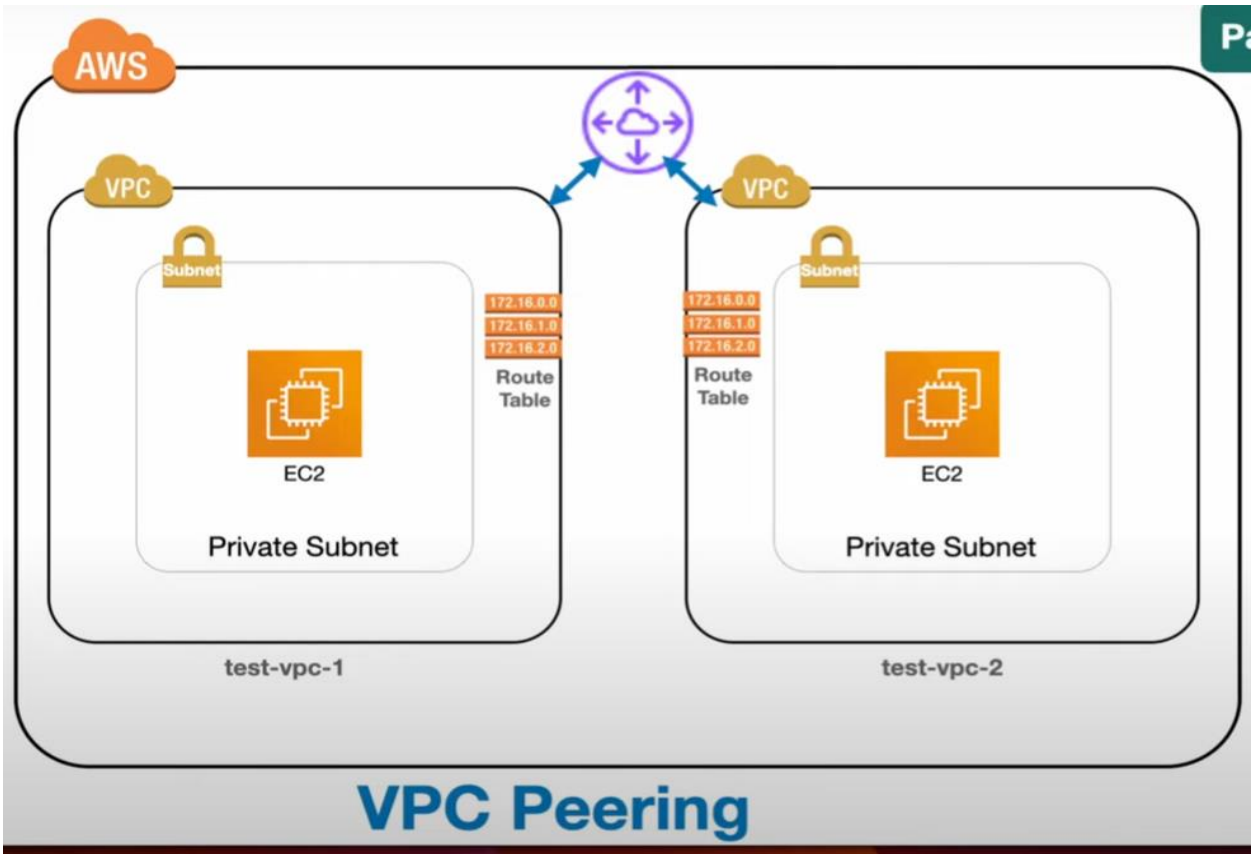
Vpc Peering



__What is VPC?

A VPC (Virtual Private Cloud) in AWS is a logically isolated private network where you canlaunch AWS resources, such as EC2 instances, databases, and containers. It allows you to defineyour own IP range, subnets, routing, and security settings to control traffic and access.

🌐 What is VPC Peering?

VPC Peering is a networking connection between two **Amazon Virtual Private Clouds (VPCs)** that allows them to communicate with each other as if they are within the same network. It provides a **secure and private** connection without using the public internet.

☑ Key Features of VPC Peering

- **Private Communication:** Data transfers between VPCs happen over **private IPs**.
- **No Bandwidth Bottleneck:** There is no single point of failure as it uses AWS backbone network.
- **Low Latency:** Ensures low latency and high-performance network connectivity.

- **Cross-Region & Cross-Account Support:** You can peer VPCs in the same or different AWS regions and accounts.
- **No Overlapping CIDR:** VPCs must have **unique, non-overlapping CIDR blocks**.

🛠️ Use Cases of VPC Peering

- Connecting applications across different VPCs.
- Accessing shared resources like databases or internal services.
- Multi-tier applications where different components reside in separate VPCs.
- Cross-account connectivity in large organizations.

🔎 Example Scenario

- **VPC A (10.0.0.0/16)** in **Account 1**
- **VPC B (192.168.0.0/16)** in **Account 2**
- Create a **VPC Peering Connection** → Accept → Add routes:
  - **VPC A → VPC B →** `192.168.0.0/16`
  - **VPC B → VPC A →** `10.0.0.0/16`
- Update **Security Groups** to allow traffic.

📦 Limitations of VPC Peering

- No **transitive peering** (Traffic from VPC A to VPC C through VPC B is not allowed).
- Cannot peer VPCs with **overlapping CIDR blocks**.
- **Manual routing** is required using route tables.
- No support for **Edge-to-Edge Routing**.

How to Create the VPC Peering?

First crate Two VPC…

Open AWS→search VPC on search bar→Open VPC→Click CreateVPC(MyVPC-1, MyVPC-2)→Give IPv4 CIDR Block → Define the IP range (e.g.,12.0.0.0/16, 13.0.0.0/16)→Create VPC

Now crate Route Tables:

Navigate to VPC→click on Route tables→Create Route table→Name:e.g(my-route-1, my-route-2)→select my-VPC-1 for rt-1 and My-VPC-2 for rt-2→Create Route table.

Now Create Two Public Subnets:

Navigate to VPC Dashboard→ Click on "Subnets"→ Subnet Name: Give a meaningful name (e.g., Demo-vpc-1-Sub -1, Demo-vpc-2-sub-2)→ Availability Zone: Choose an AZ (e.g., us-east-1a)→IPv4 CIDR Block: Define the subnet range (e.g., 12.0.1.0/24, 13.0.1.0/24)→Click "Create Subnet"

Then Attach to route table→Go to the route table click the subnet associations-edit subnet associations→choose subnet-vpc-1→ save associations.

Now Create Two Internet Gateway:

Navigate to VPC Dashboard→ Click on "Internet Gateways"→ Click "Create Internet Gateway"→ Enter a Name (e.g., My-Internet-Gateway-1, My-Internet-Gateway-2)→Click "Create Internet Gateway"→After creation, click "Attach to VPC"→Select your VPC (e.g., My-VPC-1, My-VPC-2)→Click "Attach"

Now Give the IGW's to route table:

Go to Route table→ edit route-1, route-2→ edit route→add IGW(0.0.0.0/0)→add (IGW-1, IGW-2) to route table→save changes.

Launch Two EC2 Instances:

1. Go to **EC2 Dashboard → Instances → Launch Instances**.
2. Provide:
   - **AMI**: Select Amazon Linux, Ubuntu, or your preferred OS.
   - **Instance Type**: Example - `t2.micro` (Free Tier Eligible).
3. **Configure Network**:
   - **Network**: Select your Custom for (EC2-1→VPC-1, EC2-2→VPC-2).
   - **Subnet**: Select your Custom Subnet(same here also).
   - **Auto-assign Public IP**: Enable if you want public access.
4. **Add Storage**: Choose the size and type.
5. **Configure Security Group**: Select the Security Group you created.
6. **Add Key Pair**: Create a new key pair or select an existing one for SSH access.
7. **Launch Instance**.

Note: Install Apache In both servers..

Create VPC Peering Connection

1. **Login to AWS Management Console** → Navigate to **VPC**.
2. Select **Peering Connections** → **Create Peering Connection**.
3. Provide:
    - **Peering Connection Name**: Example - `MyVPCPeering`
    - **VPC (Requester)**: Select the VPC-1 you own.
    - **VPC (Accepter)**: Choose the target VPC-2 (within the same or another AWS account).
4. If the target VPC is in another AWS account, enter the **Account ID** of the owner.
5. **Create Peering Connection** → You'll see the connection in **Pending Acceptance** status

Accept VPC Peering Connection

- If the accepter VPC is within your account:
    1. Go to **VPC** → **Peering Connections**.
    2. Select the peering connection → **Actions** → **Accept Request**.
- If it's in another account:
    1. The owner of the target VPC must log in and follow the same steps to accept.

Update Route Tables

After the peering is accepted, add routes to enable communication:

1. Go to **VPC** → **Route Tables**.
2. Select the route table associated with your VPC.
3. **Edit Routes** → **Add Route**:
    - **Destination**: CIDR block of the peer VPC (e.g., `10.1.0.0/16`).
    - **Target**: Select the VPC Peering Connection.
4. Do the same for the target VPC's route table.

Now put the IP range of our Destination:

Go to route table→open route-1→edit route→enter the IP range of the destination(VPC-2)→save changes.

Same here→ give the destination for VPC-1 IP range form route table-2→save changes.

Connect the both instances and try to connect with using curl IP addresses.