1. Create one VPC,with 1 one public subnet and 1 private subnet.

**✅ 1. Create a VPC**

1. Go to **VPC Dashboard** in AWS Console: [VPC Console](https://console.aws.amazon.com/vpc/)
2. Click **Create VPC**
3. Choose **VPC only**
4. Provide the following details:
   * **Name tag**: MyVPC
   * **IPv4 CIDR block**: 10.0.0.0/16
   * **IPv6 CIDR block**: No IPv6 CIDR block
   * **Tenancy**: Default
5. Click **Create VPC**

A screenshot of a computer

AI-generated content may be incorrect.

**Create a Public Subnet**

1. In the **VPC Dashboard**, go to **Subnets** → Click **Create subnet**
2. Select:
   * **VPC**: Select MyVPC
   * **Subnet name**: PublicSubnet
   * **Availability Zone**: Choose one (e.g., us-east-1a)
   * **IPv4 CIDR block**: 10.0.1.0/24
3. Click **Create subnet**

**✅ 3. Create a Private Subnet**

1. Click **Create subnet** again
2. Select:
   * **VPC**: MyVPC
   * **Subnet name**: PrivateSubnet
   * **Availability Zone**: (e.g., us-east-1a)
   * **IPv4 CIDR block**: 10.0.2.0/24
3. Click **Create subnet**

**✅ 4. Create and Attach Internet Gateway (for Public Subnet)**

1. In the VPC Dashboard, click **Internet Gateways** → **Create internet gateway**
   * **Name tag**: MyIGW
2. Click **Create internet gateway**
3. Select the IGW → Click **Actions** → **Attach to VPC**
   * Select MyVPC → Click **Attach**
   * **✅ 5. Update Route Table for Public Subnet**
4. Go to **Route Tables** → Find the main route table associated with your VPC (or create a new one).
5. Rename it to PublicRouteTable (optional).
6. Click the route table → Go to **Routes** → Click **Edit routes**:
   * Add route:
     + Destination: 0.0.0.0/0
     + Target: Your **Internet Gateway (MyIGW)**
   * Save changes
7. Click **Subnet associations** → Edit subnet associations → Select PublicSubnet → Save

**✅ 6. (Optional) Create NAT Gateway for Private Subnet**

If you want the private subnet to access the internet (for updates, etc.):

* Create an **Elastic IP** from the **EC2 dashboard**
* Go to **NAT Gateways** → Click **Create NAT gateway**
  + Subnet: PublicSubnet
  + Elastic IP: Select the one you created
  + Name: MyNAT
* Once created, go to **Route Tables**, create a **PrivateRouteTable**
  + Add route: 0.0.0.0/0 → Target: **NAT Gateway**
  + Associate this table with PrivateSubnet

A screenshot of a computer

AI-generated content may be incorrect.

1. Enable VPC peering for cross region.

1. Two AWS VPCs in different regions.
2. Each VPC must have:
   * A unique, **non-overlapping CIDR** (e.g., 10.0.0.0/16 and 10.1.0.0/16).
   * At least one subnet and route table configured.
   * Optional: EC2 instances for testing.

**🧱 Step 1: Create VPCs in Two Regions**

1. Go to **VPC Dashboard** > **Your VPCs**.
2. Click **Create VPC**:
   * **Name tag**: VPC-East
   * **CIDR block**: 10.0.0.0/16
   * **Region**: us-east-1

Repeat for:

* **VPC-West** with CIDR 10.1.0.0/16 in region us-west-2.

**🔗 Step 2: Create the Peering Connection**

**In us-east-1 (requester side):**

1. Go to **VPC Dashboard** > **Peering Connections**.
2. Click **Create Peering Connection**.
3. Fill in:
   * **Name tag**: East-to-West-Peer
   * **VPC (requester)**: Select VPC-East
   * **Account**: My account (assuming both VPCs are in the same account)
   * **Region**: us-west-2
   * **VPC (accepter)**: Select VPC-West
4. Click **Create Peering Connection**.

**✅ Step 3: Accept the Peering Request**

**Switch to us-west-2:**

1. Go to **VPC Dashboard** > **Peering Connections**.
2. Find the **pending** connection.
3. Select it, then click **Actions > Accept Request**.

Now the peering connection is **active**.

**Step 4: Update Route Tables (Both Sides)**

To allow traffic to flow, you must **add routes** in both VPCs.

**In us-east-1 (VPC-East):**

1. Go to **Route Tables**.
2. Select the route table used by your subnet(s).
3. Click **Edit routes > Add route**:
   * **Destination**: 10.1.0.0/16
   * **Target**: The VPC Peering Connection ID

**In us-west-2 (VPC-West):**

Do the same, but:

* **Destination**: 10.0.0.0/16 (VPC-East)
* **Target**: The same peering connection

**Step 5: Update Security Groups**

1. Go to **EC2 > Security Groups** in each region.
2. Add **inbound rule**:
   * **Type**: All traffic
   * **Source**: Other VPC's CIDR block

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

 Launch EC2 in each VPC

 SSH into one and try to **ping the private IP** of the other

. [ec2-user@ip-10-0-0-218 ~]$ ping 10.1.1.137

.if it is pinging to the other region private server .

.Then our peering is successful across the region

A computer screen shot of a computer program

AI-generated content may be incorrect.

1. Enable VPC peering for cross account. (You can collaborate with your friend and do this tas

**Gather VPC Info**

Each account must know:

* VPC ID
* VPC CIDR block
* AWS Account ID

**Request VPC Peering from Account A**

**Login to Account A**.

Go to **VPC Dashboard > Peering Connections > Create Peering Connection**.

1. Fill the form:
   * **Peering connection name tag**: A-to-B
   * **Requester VPC**: Choose Account A’s VPC.
   * **Account**: Select **Another account**.
   * **Account ID**: Enter Account B’s AWS Account ID.
   * **Accepter VPC ID**: Enter Account B’s VPC ID.
2. Click **Create Peering Connection**.

A screenshot of a computer

AI-generated content may be incorrect.

**Update Route Tables in Both Accounts**

**In Account A:**

Go to **Route Tables**.

Select the route table associated with the VPC subnets.

1. Add a new route:
   * **Destination**: CIDR block of VPC in Account B.
   * **Target**: The VPC Peering Connection.

**In Account B:** Do the same, but:

* Destination: CIDR of Account A’s VPC.
* Target: The same VPC Peering Connection.

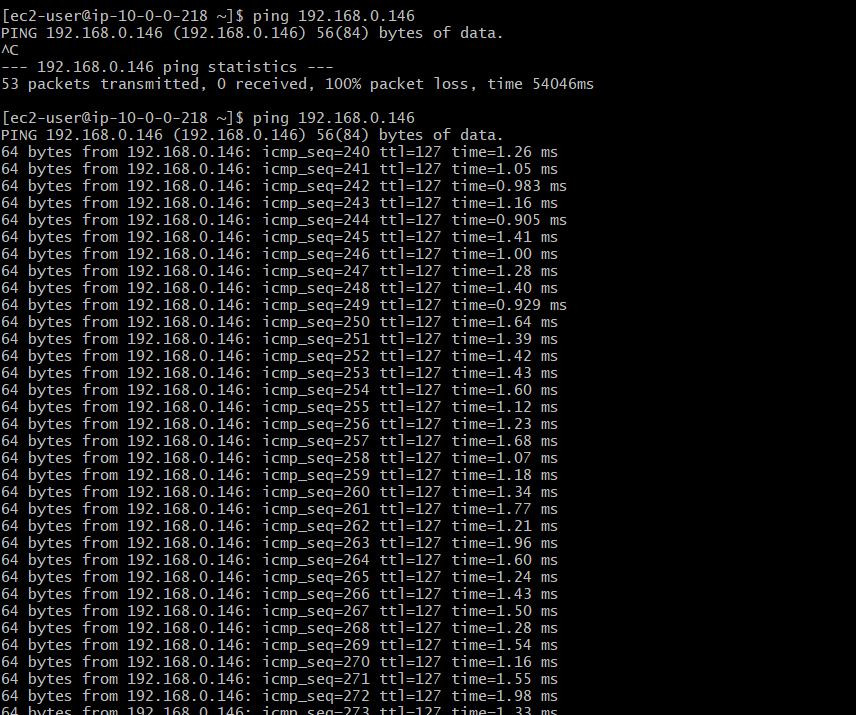
A screenshot of a computer

AI-generated content may be incorrect.

Try to ping or connect to resources across VPCs using private IPs to verify connectivity.

A computer screen shot of a black screen

AI-generated content may be incorrect.



1. Setup VPC Transist gateway.

**🔹 Create Two VPCs**

1. Go to **VPC Dashboard** > **Your VPCs**
2. Create VPC A:
   * CIDR block: 10.0.0.0/16
3. Create VPC B:
   * CIDR block: 10.1.0.0/16

Add subnets, route tables, and IGWs as needed.

**🔹 Create Transit Gateway (TGW)**

1. Go to **VPC Dashboard** > **Transit Gateways**
2. Click **Create Transit Gateway**
3. Fill in:
   * **Name**: My-TGW
   * **Amazon ASN**: leave default or set custom (e.g., 64512)
   * **DNS Support**: Enabled
   * **Default Route Table Association**: Enable
   * **Default Route Table Propagation**: Enable
4. Click **Create Transit Gateway**

✅ Wait until status is **Available**

**🔹 Attach VPCs to Transit Gateway**

1. Go to **Transit Gateway Attachments** > **Create Attachment**
2. Choose:
   * **Attachment Type**: VPC
   * **Transit Gateway**: Select the one you created
   * **VPC**: Select VPC A
   * **Subnets**: Select one subnet from each AZ you want to connect
3. Click **Create Attachment**

✅ Wait for attachments to become **Available**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. Repeat the same steps to attach **VPC B**

✅ Wait for attachments to become **Available**

A computer screen shot of a computer screen

AI-generated content may be incorrect.

A computer screen shot of a computer screen

AI-generated content may be incorrect.

**🔹 Update Route Tables in VPCs**

Go to **Route Tables** of each VPC and add the appropriate route:

**In VPC A:**

* Destination: 10.1.0.0/16
* Target: Transit Gateway

**In VPC B:**

* Destination: 10.0.0.0/16
* Target: Transit Gateway

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Launch EC2 Instances for Testing**

1. Launch one EC2 instance in each VPC
2. Use **private IPs** to ping or SSH between the instances.

A computer screen shot of a computer program

AI-generated content may be incorrect.

1. Setup VPC End Point.

=>create vpc atleast create one subnet in vpc and route table will be associated with subnet.

**1. Go to the VPC Dashboard**

* Sign in to AWS Management Console.
* Open the **VPC** service.

**2. Create Endpoint**

* In the left panel, click **Endpoints**.
* Click **"Create Endpoint"**.

**3. Configure Endpoint**

* **Service Category**: Select AWS services.
* **Service Name**:
  + Search for s3 and choose com.amazonaws.<region>.s3
  + Example: com.amazonaws.us-east-1.s3 for N. Virginia
* **VPC**: Choose your VPC.

**4. Configure Route Tables**

* Select the route table(s) associated with your subnet(s).
* This allows traffic to route to the endpoint.

**5. Policy**

* Choose **"Full access"** or **custom policy** depending on your security needs.

**6. Name tag (Optional)**

* Give your endpoint a name like MyS3Endpoint.

**7. Create Endpoint**

* Click **Create endpoint**.

A screenshot of a computer

AI-generated content may be incorrect.

Create a bucket in s3

A screenshot of a computer

AI-generated content may be incorrect.

Create access key.

A screenshot of a computer

AI-generated content may be incorrect.

**Test the VPC Endpoint**

1. Launch an EC2 instance **without a public IP** in that VPC.
2. Use the AWS CLI or curl to access an S3 bucket:

bash

CopyEdit

aws s3 ls s3://<your-bucket-name>

If this works from a private EC2, your VPC endpoint is functioning correctly.

A computer screen shot of a computer program

AI-generated content may be incorrect.