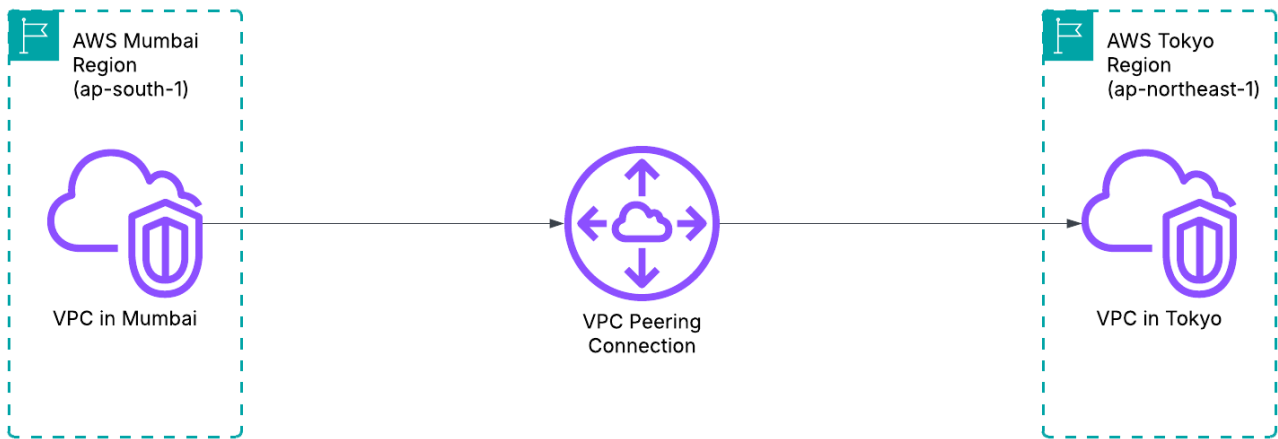


VPC-Peering



This project demonstrates the setup of a **VPC Peering connection** between two AWS regions: **Mumbai (ap-south-1)** and **Tokyo (ap-northeast-1)**. The Mumbai VPC has a **CIDR range of 10.0.0.0/16**, while the Tokyo VPC uses **192.0.0.0/16**.

The process includes:

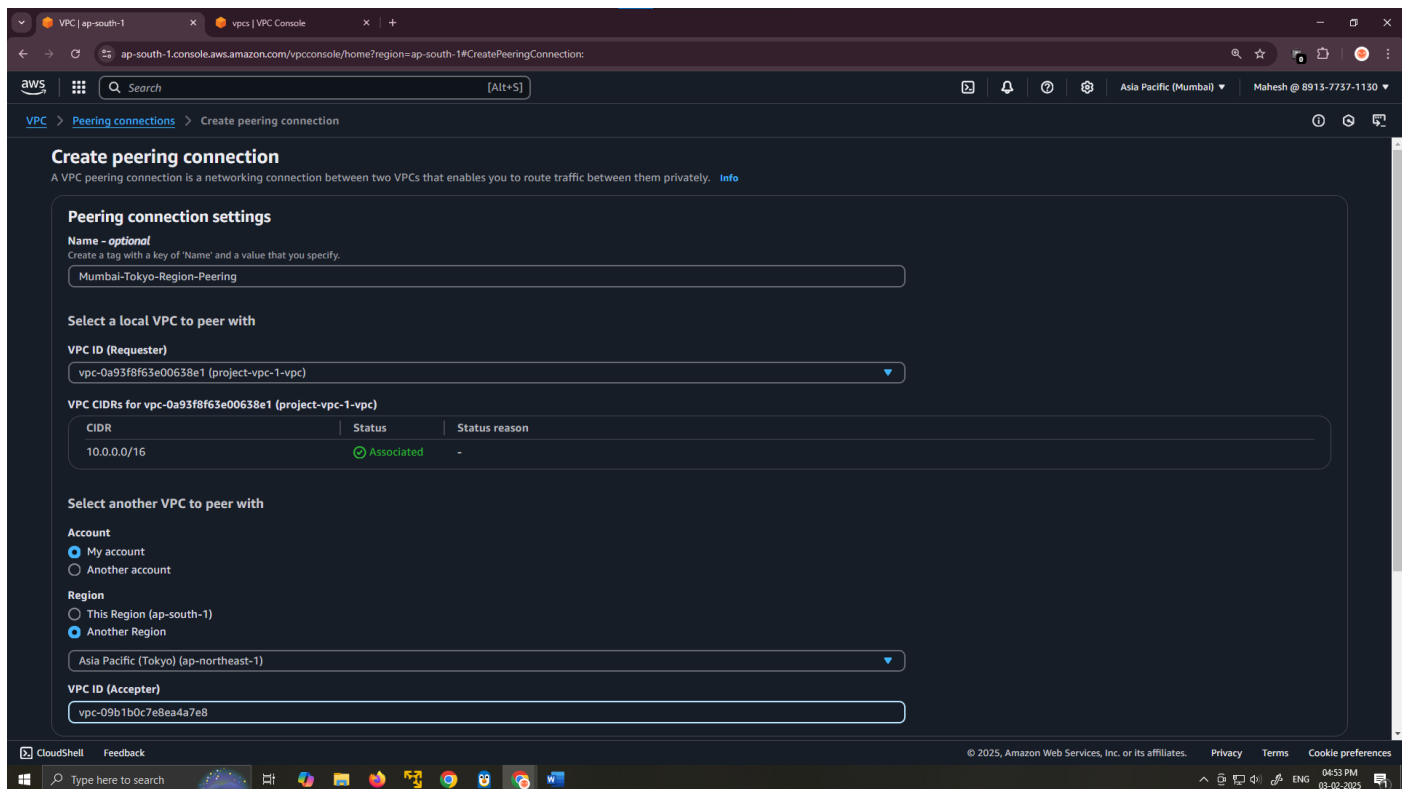
1. **Creating and accepting the VPC Peering Connection** between the two VPCs.
2. **Updating route tables** in both VPCs to allow cross-region communication.
3. **Configuring security groups** to permit ICMP (ping) and other necessary traffic.
4. **Testing connectivity** between instances in both regions using the ping command.

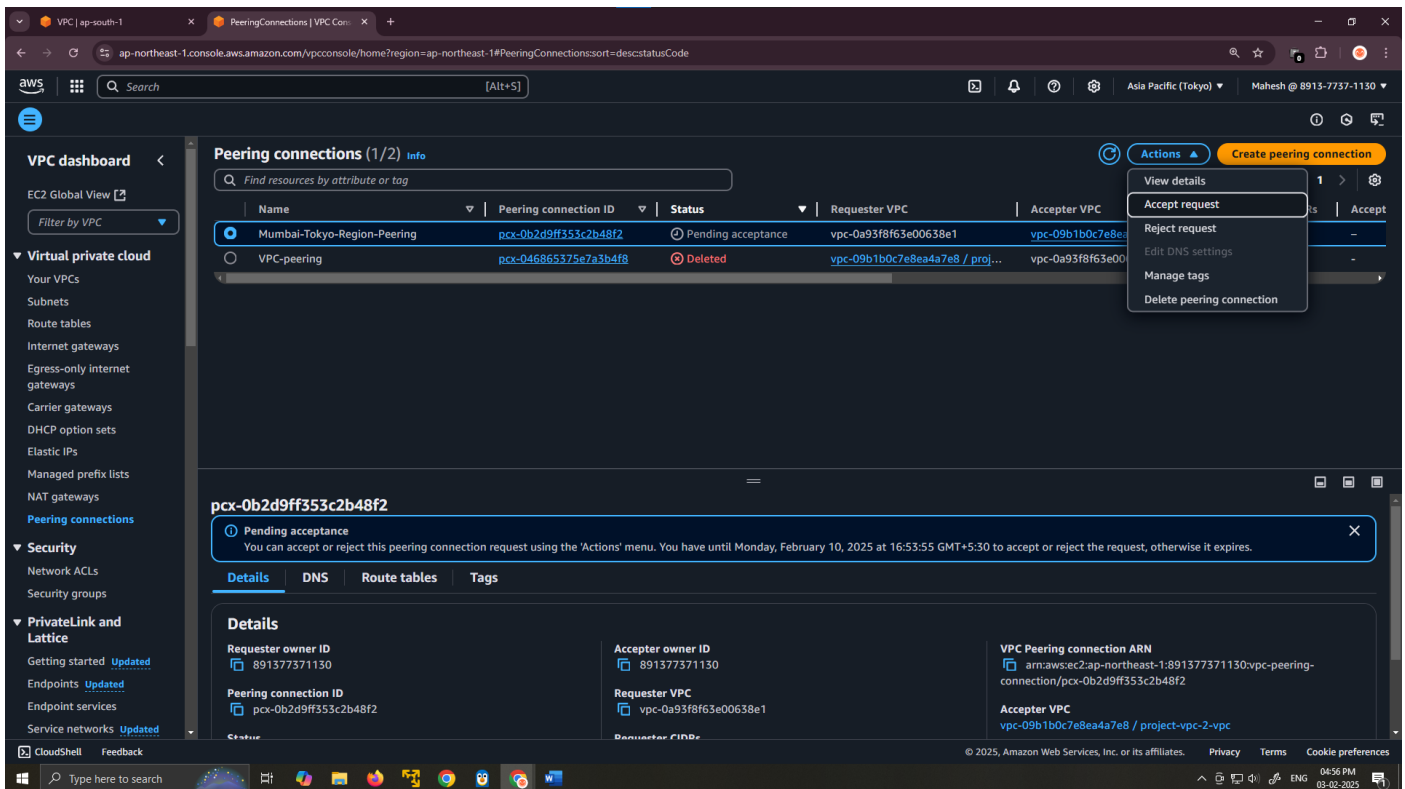
This setup enables secure, low-latency, and private communication between VPCs without using the public internet. 🛡️

Step 1: Create a VPC Peering Connection

1. **Login to AWS Console** and navigate to **VPC Dashboard**.
2. In the **Mumbai region (ap-south-1)**:
 - Go to **Peering Connections** → Click **Create Peering Connection**.
 - **Name tag:** Mumbai-Tokyo-Peering
 - **VPC Requester:** Select Mumbai VPC (10.0.0.0/16).

- **VPC Acceptor:** Choose **Another account** or **Another region**.
 - **Region:** Select Tokyo (ap-northeast-1).
 - **Acceptor VPC ID:** Select Tokyo VPC (192.0.0.0/16).
 - Click **Create Peering Connection**.
3. Now, go to the **Tokyo region** (ap-northeast-1):
- Navigate to **VPC Peering Connections**.
 - Select the **Peering Request** received from Mumbai VPC.
 - Click **Accept Request**.
4. **Verify Peering Connection:**
- After accepting, the **status** of the peering connection should change to Active.





Step 2: Update Route Tables

After establishing the VPC peering connection, you need to update the **route tables** in both VPCs to allow traffic to pass.

In Mumbai Region (ap-south-1)

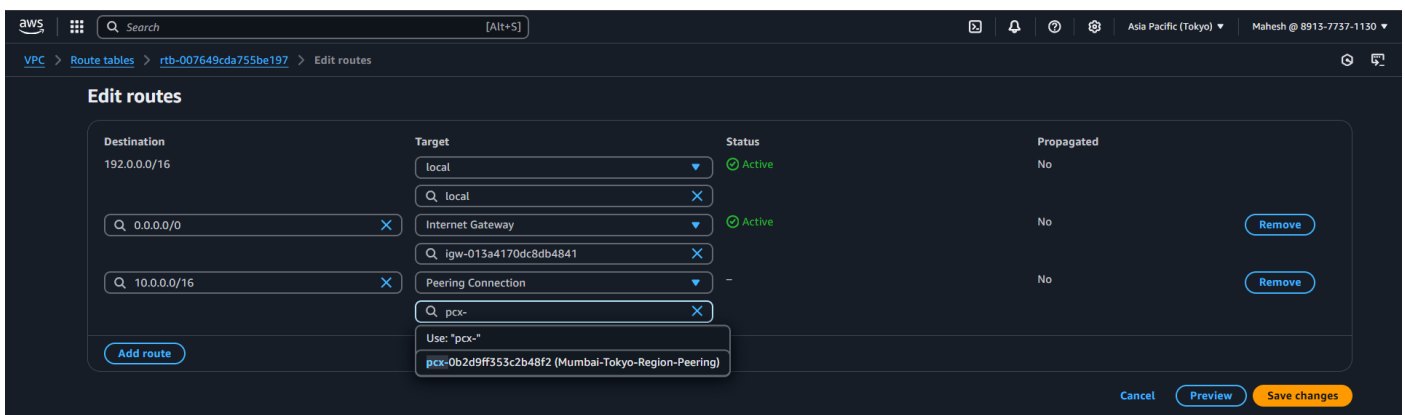
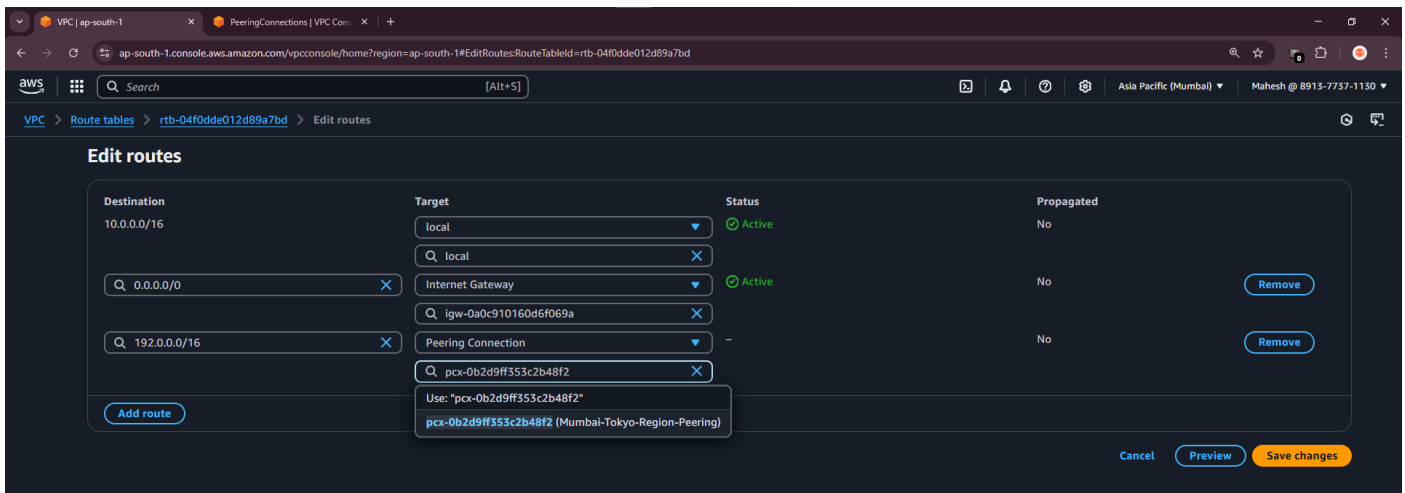
1. Navigate to **VPC → Route Tables**.
2. Select the **Route Table** associated with your **Mumbai VPC (10.0.0.0/16)**.
3. Click on **Routes → Edit Routes**.
4. Add a new route:
 - **Destination:** 192.0.0.0/16 (Tokyo VPC CIDR)
 - **Target:** Select the **Peering Connection (pcx-xxxxxxxxxx)**.
5. Click **Save changes**.

In Tokyo Region (ap-northeast-1)

1. Navigate to **VPC → Route Tables**.
2. Select the **Route Table** associated with your **Tokyo VPC (192.0.0.0/16)**.
3. Click on **Routes → Edit Routes**.
4. Add a new route:
 - **Destination:** 10.0.0.0/16 (Mumbai VPC CIDR)

- **Target:** Select the **Peering Connection (pcx-xxxxxxxxxx)**.

5. Click **Save changes**.



Step 3: Update Security Groups

By default, security groups restrict all inbound and outbound traffic. You need to allow **ICMP (ping)** and **other necessary traffic**.

In Mumbai Region (ap-south-1)

1. Navigate to **EC2 → Security Groups**.
2. Select the **Security Group** associated with Mumbai VPC instances.
3. Go to the **Inbound rules** and click **Edit inbound rules**.
4. Add a new rule:
 - **Type:** All ICMP - IPv4
 - **Protocol:** ICMP
 - **Port Range:** All
 - **Source:** 192.0.0.0/16 (Tokyo VPC CIDR)
5. Click **Save rules**.

In Tokyo Region (ap-northeast-1)

1. Navigate to **EC2** → **Security Groups**.
2. Select the **Security Group** associated with Tokyo VPC instances.
3. Go to the **Inbound rules** and click **Edit inbound rules**.
4. Add a new rule:
 - **Type:** All ICMP - IPv4
 - **Protocol:** ICMP
 - **Port Range:** All
 - **Source:** 10.0.0.0/16 (Mumbai VPC CIDR)
5. Click **Save rules**.

The screenshot shows the AWS Management Console interface for editing inbound rules of a security group in the Mumbai region. The breadcrumb trail is EC2 > Security Groups > sg-02813dbd1acb39ed6 - launch-wizard-1 > Edit inbound rules. The page title is 'Edit inbound rules' with an 'Info' link. A subtitle states: 'Inbound rules control the incoming traffic that's allowed to reach the instance.'

The 'Inbound rules' section contains a table with the following columns: Security group rule ID, Type, Protocol, Port range, Source, and Description - optional. There are two existing rules:

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-08a6531346d4d7a57	All ICMP - IPv4	ICMP	All	Custom	192.0.0.0/16
sgr-0457748d465b24a22	SSH	TCP	22	Custom	0.0.0.0/0

Below the table is an 'Add rule' button. At the bottom of the console, there is a yellow warning banner: 'Rules with source of 0.0.0.0/0 or ::0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.'

At the bottom right, there are three buttons: 'Cancel', 'Preview changes', and 'Save rules'.

The screenshot shows the AWS Management Console interface for editing inbound rules of a security group in the Tokyo region. The breadcrumb trail is EC2 > Security Groups > sg-08216a448fa210b86 - launch-wizard-6 > Edit inbound rules. The page title is 'Edit inbound rules' with an 'Info' link. A subtitle states: 'Inbound rules control the incoming traffic that's allowed to reach the instance.'

The 'Inbound rules' section contains a table with the following columns: Security group rule ID, Type, Protocol, Port range, Source, and Description - optional. There are two existing rules:

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-0dadcb320a86dc233	SSH	TCP	22	Custom	0.0.0.0/0
sgr-0908b6601746320ef	All ICMP - IPv4	ICMP	All	Custom	10.0.0.0/16

Below the table is an 'Add rule' button. At the bottom of the console, there is a yellow warning banner: 'Rules with source of 0.0.0.0/0 or ::0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.'

At the bottom right, there are three buttons: 'Cancel', 'Preview changes', and 'Save rules'.

Test the Peering Connection.

1. SSH into an instance
2. ping Private IP of instance

The screenshot shows the AWS CloudShell interface. The terminal output displays a successful ping command from the instance's private IP (10.0.11.90) to the target private IP (192.0.8.73). The ping results show 28 successful requests with a consistent time of 139 ms.

```
ubuntu@ip-10-0-11-90:~$ ping 192.0.8.73
PING 192.0.8.73 (192.0.8.73) 56(84) bytes of data:
64 bytes from 192.0.8.73: icmp_seq=1 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=2 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=3 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=4 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=5 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=6 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=7 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=8 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=9 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=10 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=11 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=12 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=13 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=14 ttl=64 time=138 ms
64 bytes from 192.0.8.73: icmp_seq=15 ttl=64 time=140 ms
64 bytes from 192.0.8.73: icmp_seq=16 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=17 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=18 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=19 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=20 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=21 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=22 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=23 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=24 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=25 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=26 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=27 ttl=64 time=139 ms
64 bytes from 192.0.8.73: icmp_seq=28 ttl=64 time=139 ms
```

Below the terminal output, a notification box for instance **i-0910a53b70a6f1369 (1st-vpc)** is visible, showing its Public IP (13.232.99.205) and Private IP (10.0.11.90).

The screenshot displays the AWS Management Console for two EC2 instances. The top instance, **i-084a985742401f026 (2nd-vpc)**, is in the 'Running' state. It has a Public IPv4 address of 18.183.236.32 and a Private IPv4 address of 192.0.8.73. The bottom instance, **i-0910a53b70a6f1369 (1st-vpc)**, is also in the 'Running' state. It has a Public IPv4 address of 13.232.99.205 and a Private IPv4 address of 10.0.11.90. Both instances are located in the Asia Pacific (Mumbai) region.

