

# Cross-Region VPC Peering

This document explains the entire setup of **Cross-Region VPC Peering** between:

- **Region A:** Asia Pacific (Singapore) ap-southeast-1
  - **Region B:** Europe (Ireland) eu-west-1
- 

## 1. Project Overview

This project demonstrates how to create two vpc's in different regions/accounts and set up secure communication via peering test with ec2 instances.

---

## 2. Architecture Summary

### Region A: Asia Pacific (Singapore) ap-southeast-1

- VPC-A CIDR: 10.0.0.0/16
- Subnet-A: 10.0.0.0/17
- EC2-A Private IP: 10.0.63.38

### Region B: Europe (Ireland) eu-west-1

- VPC-B CIDR: 172.16.0.0/16
- Subnet-B: 172.16.0.0/17
- EC2-B Private IP: 172.16.88.161

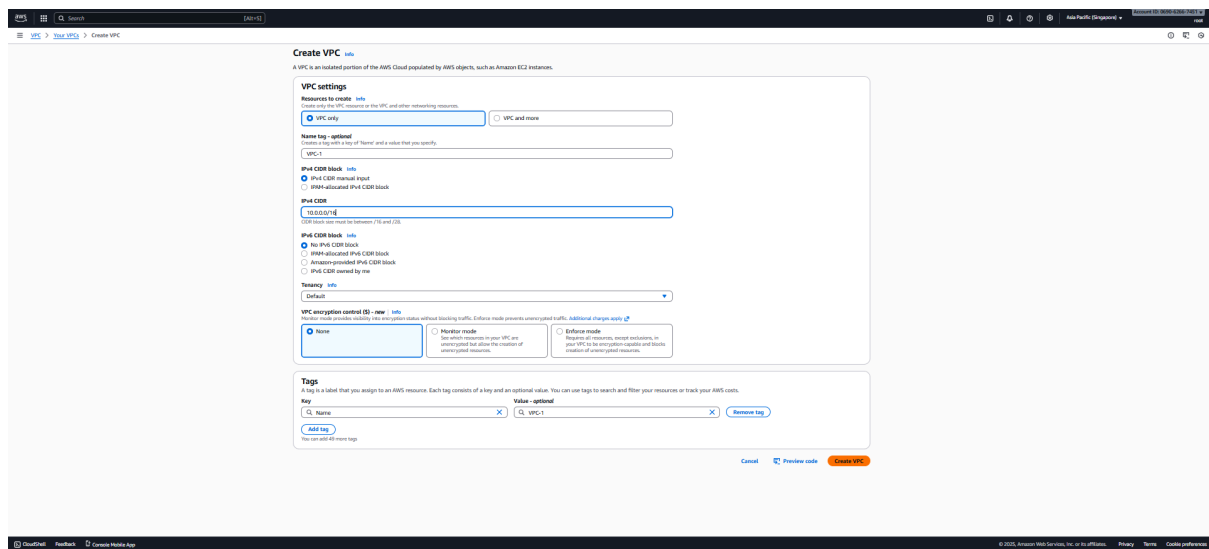
### Connection

- VPC Peering initiated from Singapore → accepted in Ireland
- Route tables updated on both sides
- Security groups configured to allow ICMP/SSH

# 3. Step-by-Step Implementation

## Step 1: Create VPC in Singapore (Region A)

1. Go to **AWS Console** → **VPC Dashboard** > **Your VPCs** > **Create VPC**
2. Enter:
  - Name: **VPC-1**
  - CIDR: **10.0.0.0/16**
3. Click **Create VPC**.



Snapshot: Singapore VPC creation

## Step 2: Create VPC in Ireland (Region B)

1. Switch region to **Europe (Ireland)**
2. Create a new VPC:
  - Name: **VPC-2**
  - CIDR: **172.16.0.0/16**
3. Click **Create VPC**.

**Create VPC** info

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.

**VPC settings** info

**Resources to create** info

Create only the VPC resource or the VPC and other networking resources.

☒ VPC only ☐ VPC and more

**Name tag - optional** info

Create a tag with a key of 'Name' and a value that you specify.

VPC-2

**IPv4 CIDR block** info

☒ IPv4 CIDR network input ☐ IPAM-allocated IPv4 CIDR block

10.0.0.0/16

CIDR block size must be between /16 and /28.

**IPv6 CIDR block** info

☒ No IPv6 CIDR block ☐ IPAM-allocated IPv6 CIDR block ☐ Amazon-provided IPv6 CIDR block ☐ IPv6 CIDR owned by me

**Tenancy** info

Default

**VPC encryption control (EC2)** new info

Amazon uses encrypted network flow encryption states without blocking traffic. Enforce mode prevents unencrypted traffic. Additional charges apply. [Additional charges apply](#)

☒ None ☐ Monitor mode ☐ Enforce mode

Monitor mode: See which resources in your VPC are unencrypted but allow the creation of unencrypted resources.

Enforce mode: Resources in your VPC are unencrypted but your VPC is in encryption-enabled and blocks creation of unencrypted resources.

**Tags** info

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

**Key** **Value - optional**

Name VPC-2 [Remove tag](#)

[Add new tag](#)

You can add at most 50 tags.

[Cancel](#) [Preview costs](#) [Create VPC](#)

**Snapshot: Ireland VPC creation**

## Step 3: Create Subnets

### Singapore Subnet-1

AWS Console → VPC → Subnets → Create Subnet

- CIDR: 10.0.0.0/17
- AZ: (any available zone )(optional)

**Create subnet** info

**VPC**

VPC-ID

Create subnets in this VPC.

vpc-0a4b4d4d4d4d4d4d (VPC-1)

**Associated VPC CIDRs**

IPv4 CIDRs

10.0.0.0/16

**Subnet settings** info

Specify the CIDR blocks and Availability Zone for the subnet.

**Subnet 1 of 1**

**Subnet name** info

Create a tag with a key of 'Name' and a value that you specify.

SUBNET-1

This tag can be up to 256 characters long.

**Availability Zone** info

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

No preference

**IPv4 VPC CIDR block** info

Choose the IPv4 VPC CIDR block for the subnet. The subnet's IPv4 CIDR must be within this block.

10.0.0.0/16

**IPv4 subnet CIDR block**

10.0.0.0/17

**Tags - optional**

**Key** **Value - optional**

Name SUBNET-1 [Remove](#)

[Add new tag](#)

You can add at most 50 tags.

[Remove](#)

[Add new subnet](#)

[Cancel](#) [Create subnet](#)

**Snapshot: Subnet creation**

## Ireland Subnet-2

- CIDR: 172.16.0.0/17
- AZ: (any available zone )(optional)

**Create subnet** [info](#)

**VPC**  
vpc-0  
vpc-0af1f9b123900108f (VPC-02)

**Associated VPC CIDRs**  
IPv4 CIDRs  
172.16.0.0/16

**Subnet settings**  
Specify the CIDR blocks and Availability Zone for the subnet.

**Subnet 1 of 1**

**Subnet name**  
Create a tag with a key of 'Name' and a value that you specify.  
Subnet-2  
The name can be up to 256 characters long.

**Availability Zone** [info](#)  
Choose the zone in which your subnet will reside, or let Amazon choose one for you.  
No preference

**IPv4 VPC CIDR block** [info](#)  
Choose the IPv4 VPC CIDR block for this subnet. The subnet's IPv4 CIDR must fit within this block.  
172.16.0.0/16

**IPv4 subnet CIDR block**  
172.16.0.0/17

**Tags - optional**  
Key: Name Value: Subnet-2  
[Add new tag](#)  
You can add up to 50 tags.  
[Remove](#)  
[Add new subnet](#)

[Cancel](#) [Create subnet](#)

Snapshot: Subnet creation

## Step 4: Create Route Tables

### Region A Route Table

VPC Dashboard → Route Tables

1. Filtered the route tables by **VPC-1**, so only VPC-1–related route tables were visible.

**Route tables (1)** [info](#)

Find route tables by attribute or tag  
vpc-0acdebade950a449a [Clear filters](#)

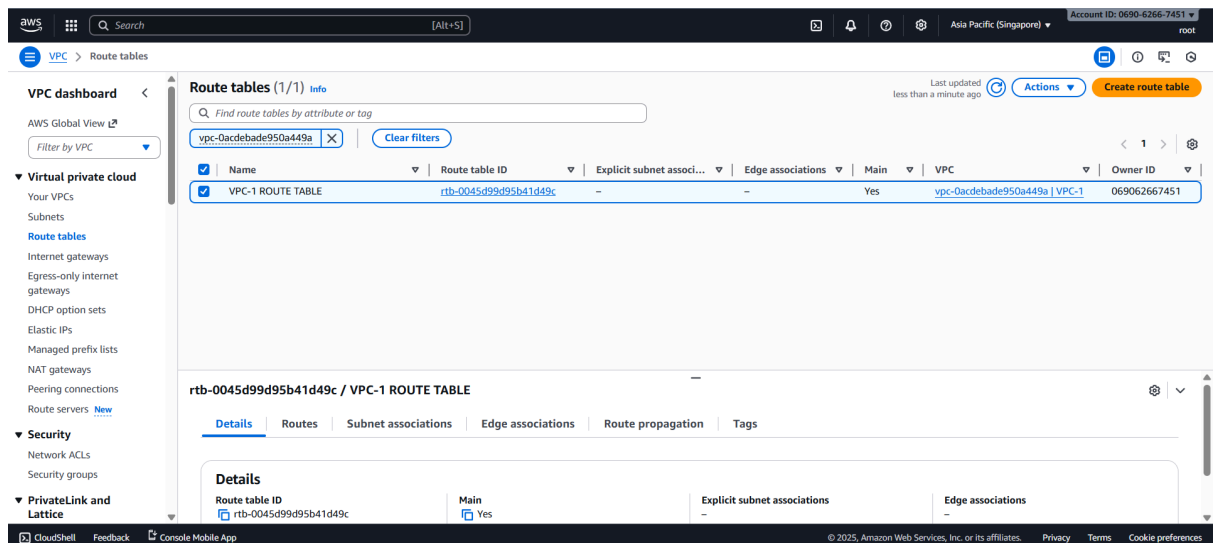
Last updated less than a minute ago [Actions](#) [Create route table](#)

<input type="checkbox"/>	Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC	Owner ID
<input type="checkbox"/>	-	rtb-0045d99d95b41d49c	-	-	Yes	vpc-0acdebade950a449a   VPC-1	069062667451

Select a route table

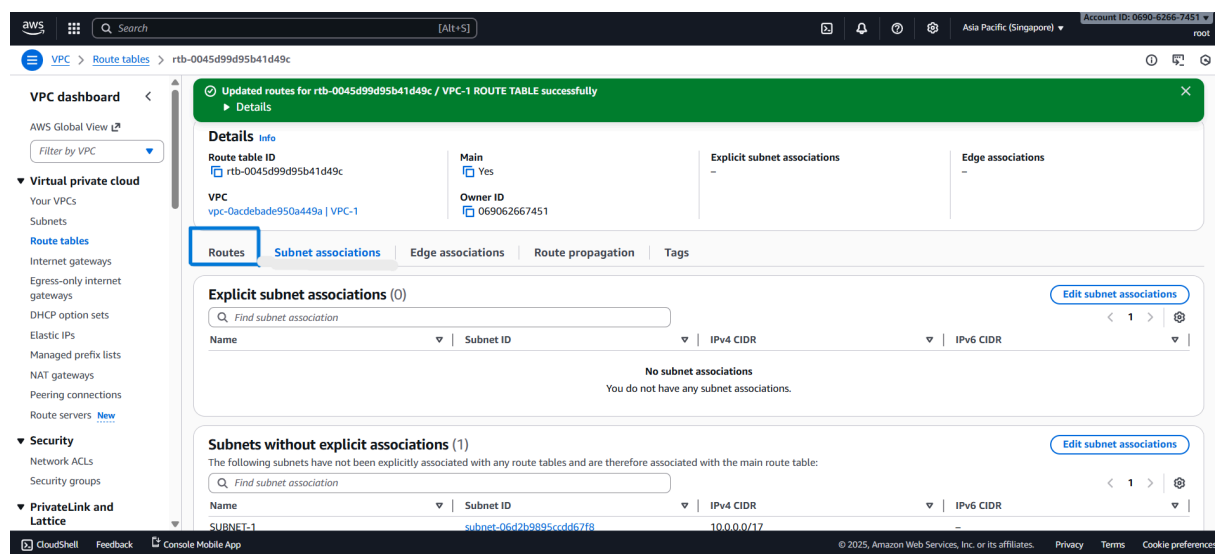
Snapshot: Filtering the route tables by **VPC-1**

2. Selected the default route table and **renamed it to:**  
**VPC-1 ROUTE TABLE**



Snapshot: Renaming the Root Table

3. Opened the route table → **Routes** tab → **Edit Routes**.

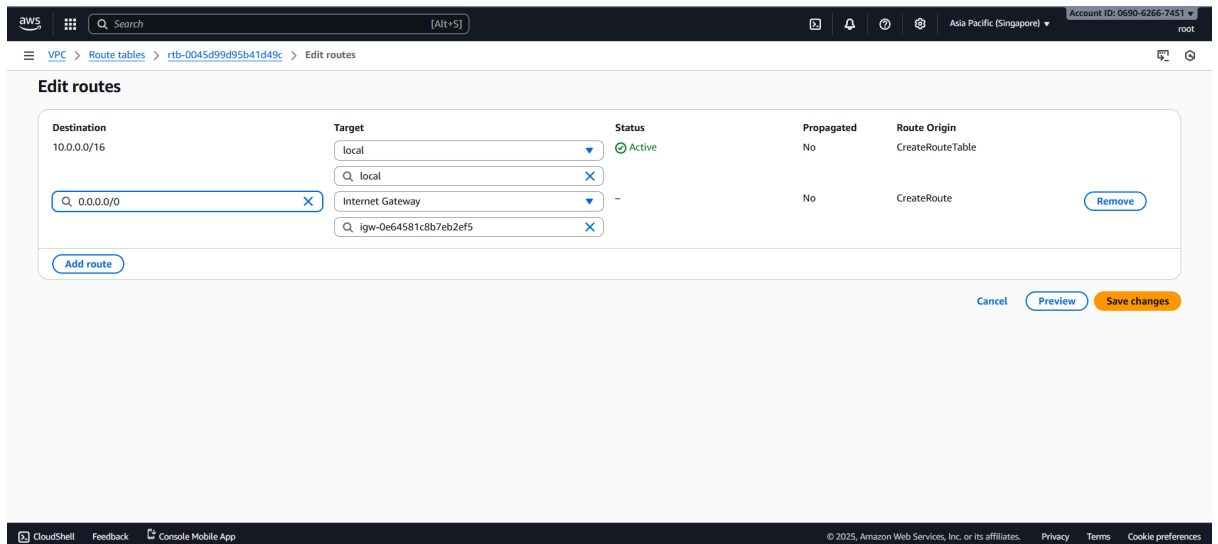


Snapshot: Root tab

4. Added a new route for communication with Region B:

- **Destination:** 0.0.0.0/0
- **Target:** Internet Gateway

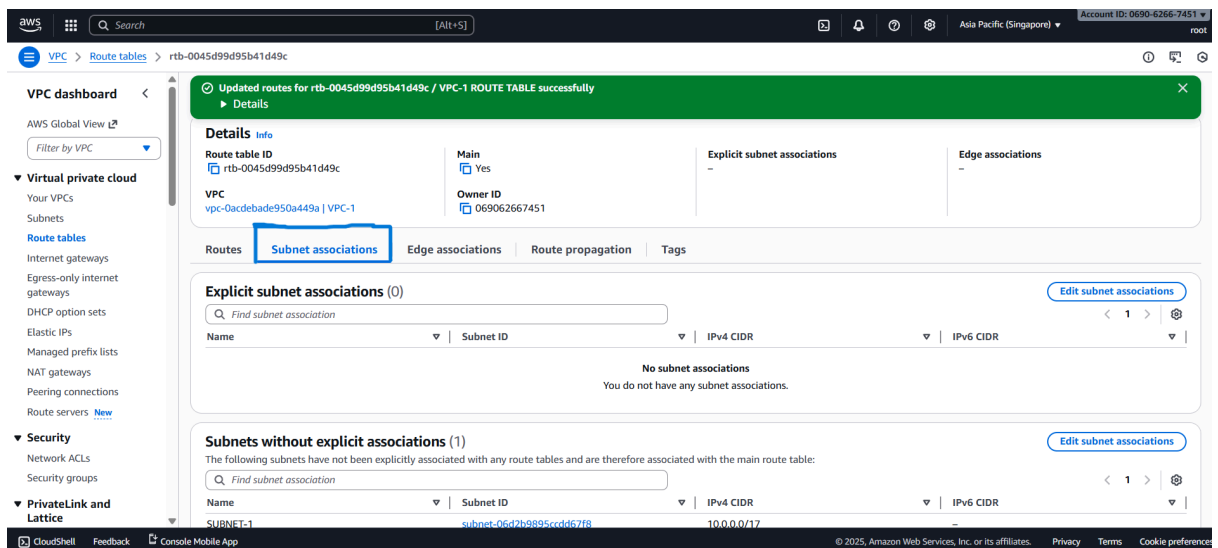
Saved changes.



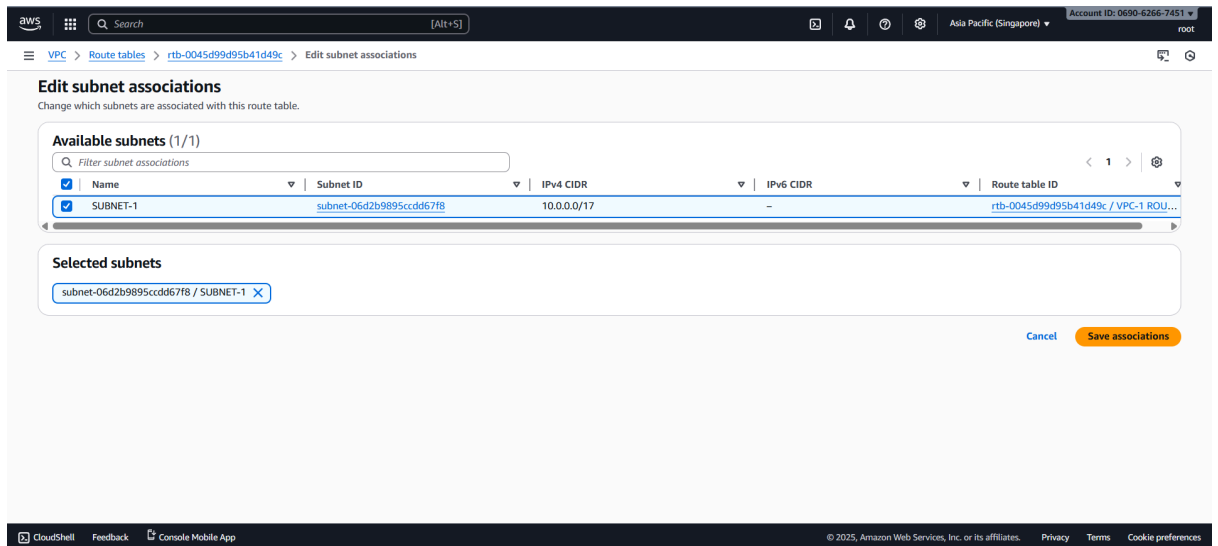
Snapshot :Edit Routes

5. Under **Subnet Associations** tab:

- Edit subnet associations
- Selected **SUBNET-1 (10.0.0.0/17)**
- Saved.



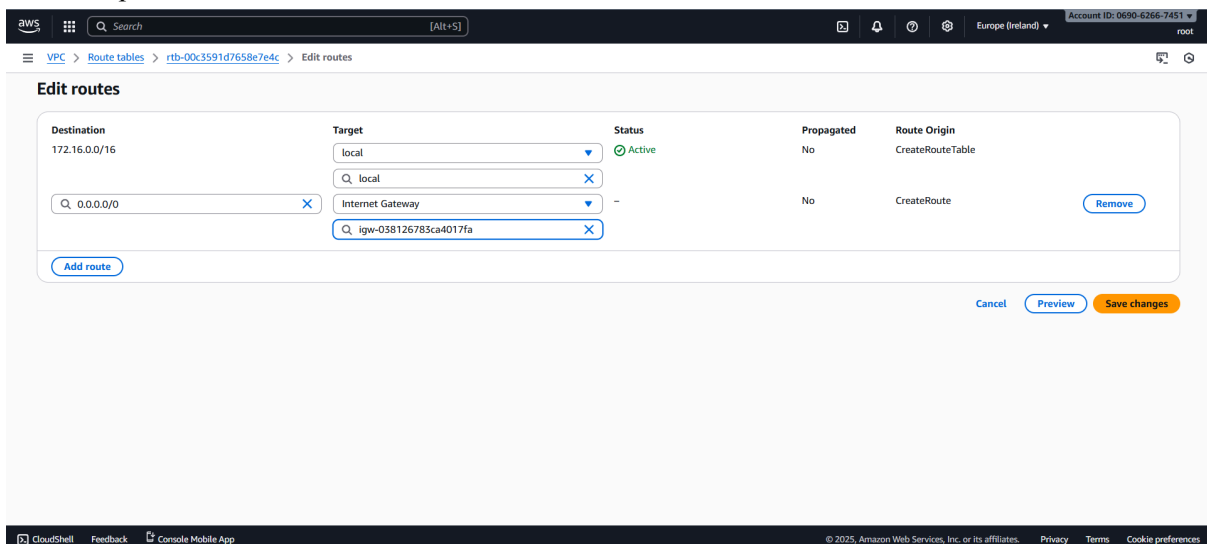
Snapshot: Subnet Associations Tab



### Snapshot: Edit Subnet Associations

## Region B Route Table

1. Filtered all route tables by **VPC-2** so only route tables belonging to this VPC were shown.
2. Selected the main route table and renamed it to:  
**VPC-2 ROUTE TABLE**
3. Opened the route table → **Routes** tab → clicked **Edit Routes**.

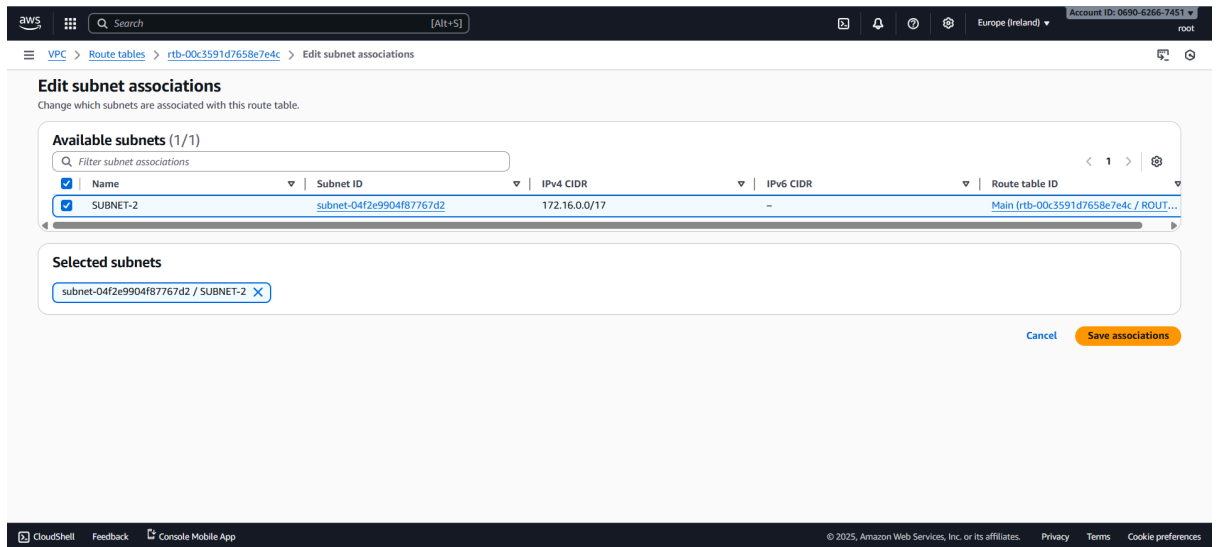


### Snapshot: Snapshot :Edit Routes

4. Added a new route for communication with Region A:
  - **Destination:** 0.0.0.0/0
  - **Target:** Internet Gateway
  - Saved the route.

5. Opened the **Subnet Associations** tab:

- Clicked **Edit subnet associations**
- Selected **Subnet-B (172.16.0.0/17)**
- Saved.

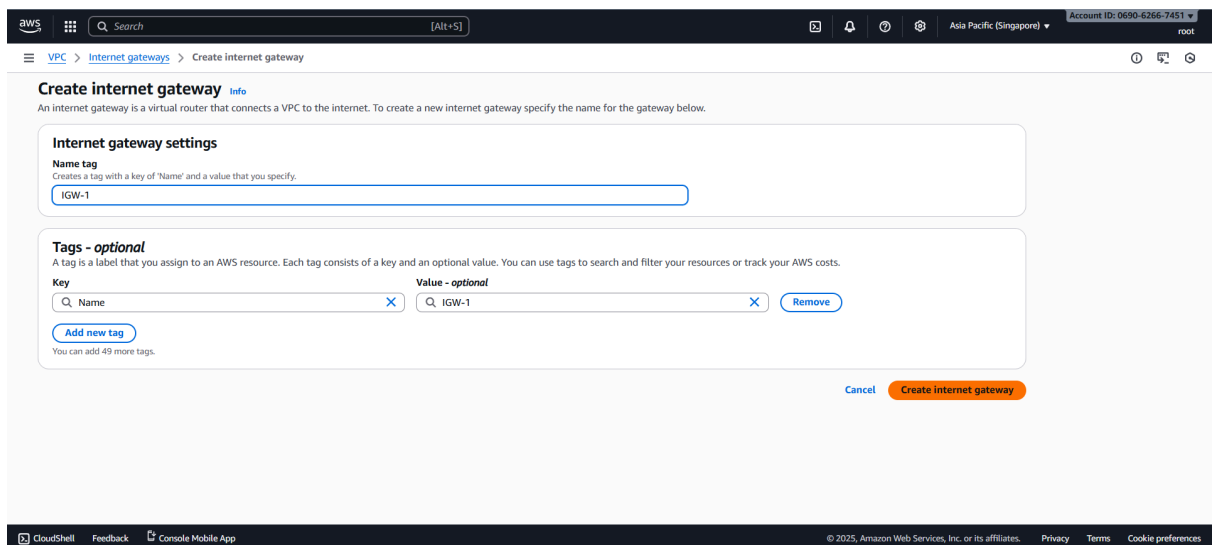


Snapshot: Edit Subnet Associations

## Step 5: Create Internet Gateway (IGW)

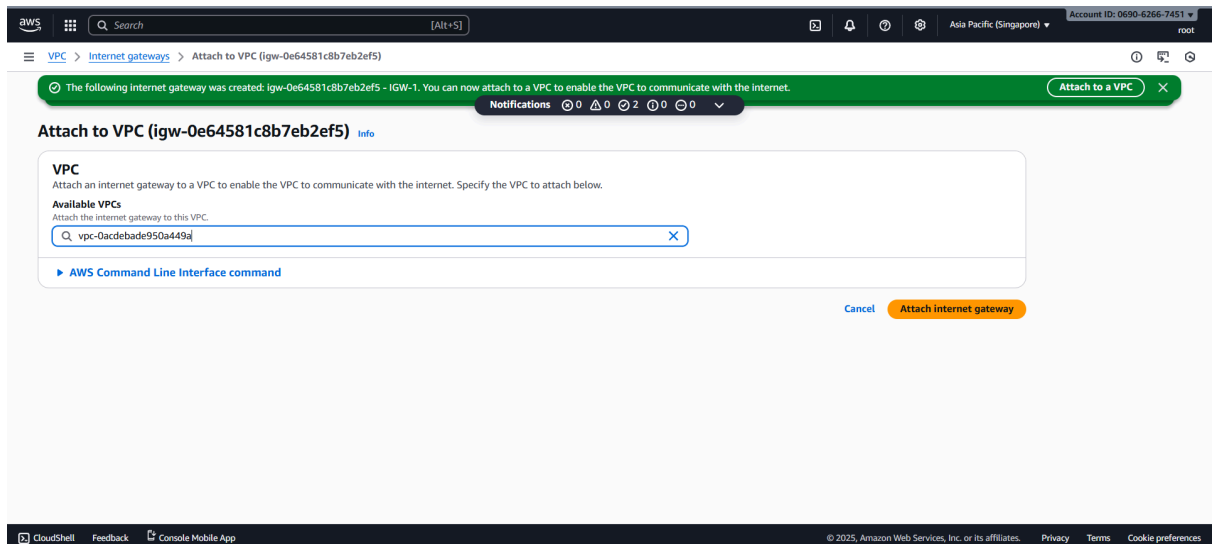
### Region A IGW

- Create IGW → Attach to VPC-1



Snapshot: Create Internet Gateway





**Snapshot: Attach to VPC**

## Region B IGW

- Create IGW → Attach to VPC-B  
(same as above procedure)

---

## 4. Create EC2 Instances

AWS Console → Services → EC2 → Instances → Launch Instances

1. Name the Instance

2. Choose Application & OS Image (AMI)

- Select **Amazon Linux**.
- AMI: **Amazon Linux 2023 kernel-6.1 AMI**

3. Select Instance Type

- Recommended: **t3.micro** or **t2.micro** (Free-tier eligible).

4. Configure Key Pair

- **Create a new key pair** for SSH access.

5. Configure Network Settings

- Select the appropriate VPC → choose **VPC-1**.
- **Enable** Auto-assign Public IP.

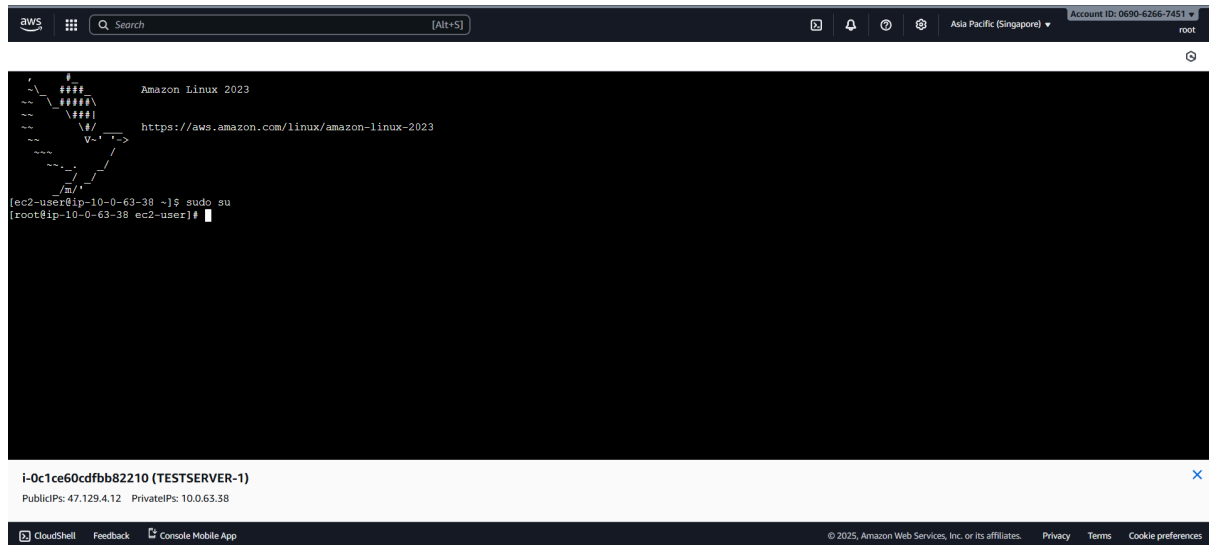
6. Configure Firewall (Security Group)

- Create a new security group.
- Provide a security group name.
- Set Type: **All traffic** .

## 7. Launch Instance

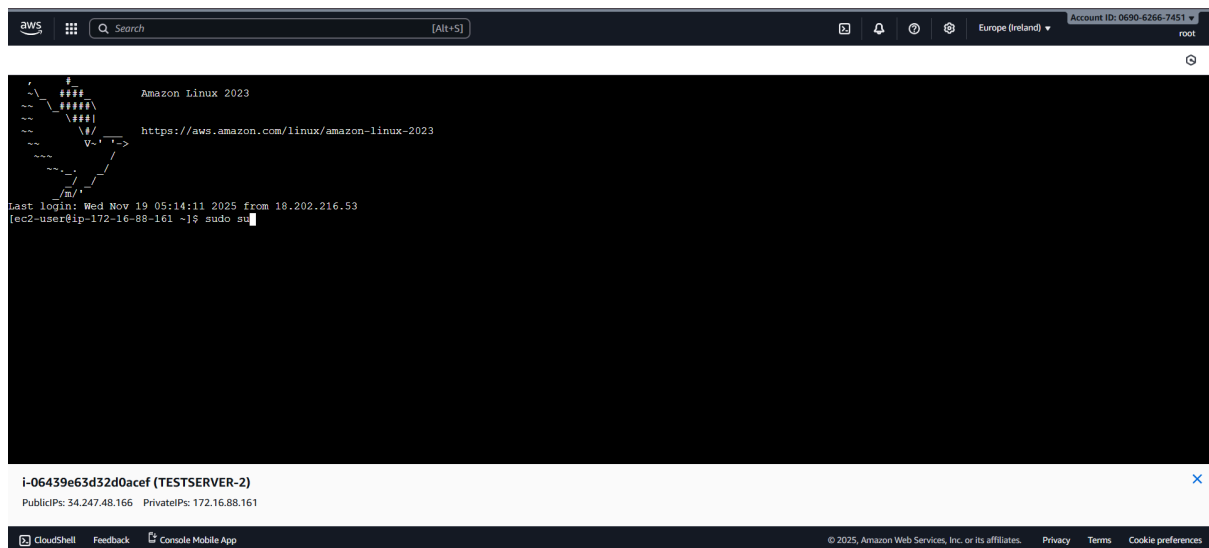
- Review all settings and click Launch Instance.

## TESTSERVER-1 (Singapore)



### Snapshot: Testserver-1

## TESTSERVER-2 (Ireland)



### Snapshot: Testserver-2

# 5. Create VPC Peering Connection

AWS Console → VPC → Peering Connections → Create Peering Connection

## Steps to Create the Peering Connection

### Step 1: Open the Peering Creation Page

1. Go to **VPC Dashboard**
2. In the left menu, click **Peering Connections**
3. Click **Create Peering Connection**

### Step 1 — Initiate Peering Request from Singapore

**Create peering connection**

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them privately. [Info](#)

**Peering connection settings**

**Name - optional**  
Create a new entry or type of 'Name' and a value that you specify.  
PENDING-1

**Select a local VPC to peer with**

**VPC ID (Requester)**  
vpc-0a4b4a6d50a4494a (VPC-1)

**VPC CIDR for vpc-0a4b4a6d50a4494a (VPC-1)**

CIDR	Status	Status reason
10.0.0.0/16	Associated	

**Select another VPC to peer with**

**Account**  
☒ My account  
☐ Another account

**Region**  
☐ This Region (ap-south-east-1)  
☒ Another Region  
Europe (Ireland) (eu-west-1)

**VPC ID (Requester)**  
vpc-0af1498123000786f

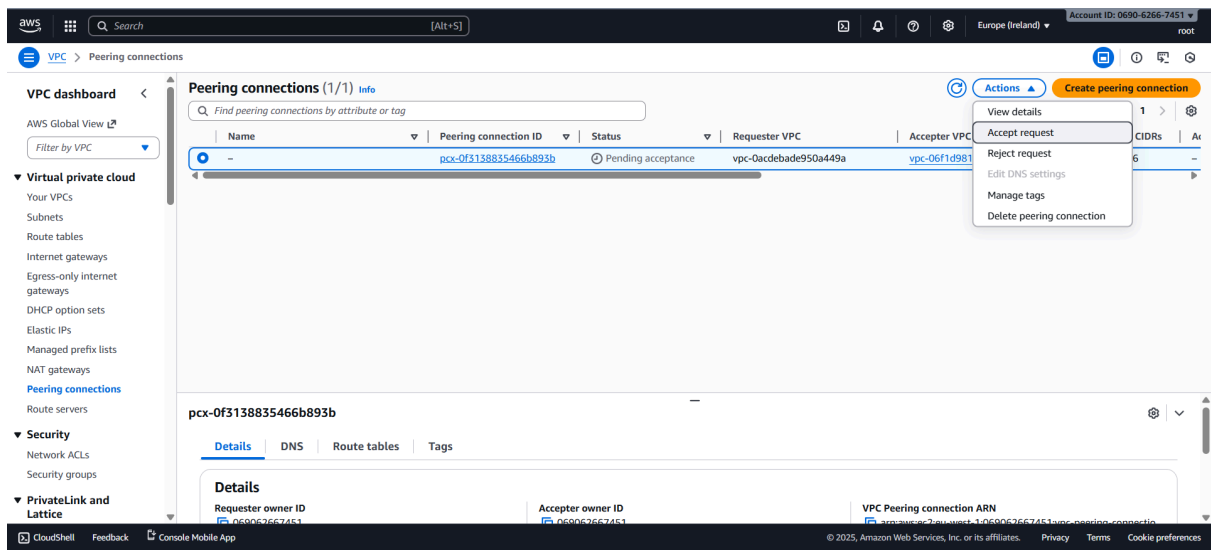
**Tags**  
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.  
**Key** **Value - optional**  
Name PENDING-1 [Remove](#)  
[Add new tag](#)  
You can add 40 more tags.

[Cancel](#) [Create peering connection](#)

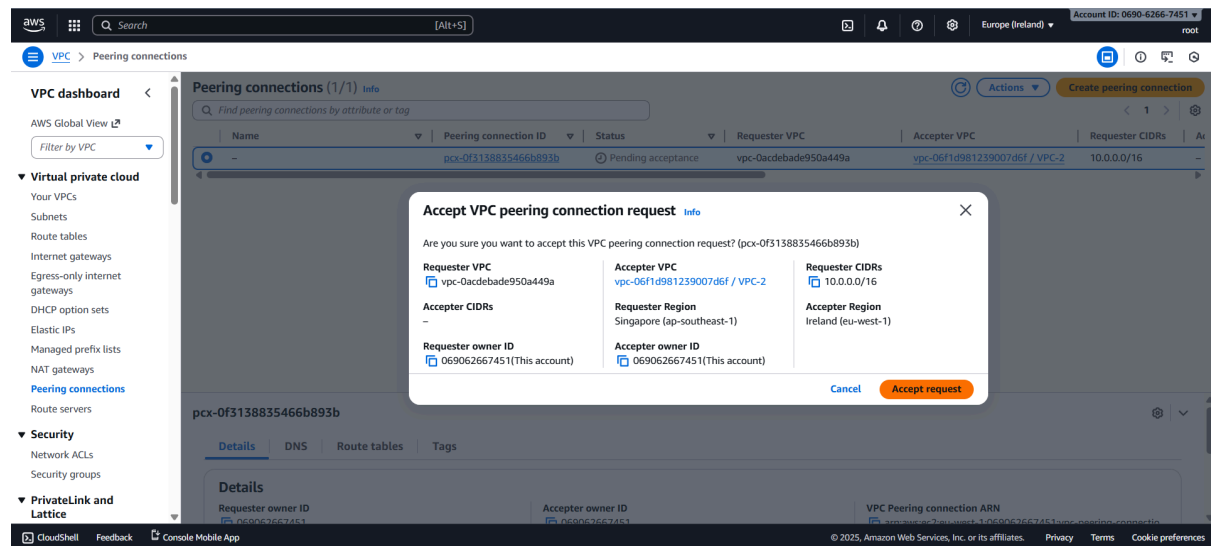
**Snapshot:** Initiating Peering Request

### Step 2 — Accept Peering in Ireland

- Switch region → Ireland
- Accept peering request



Snapshot: Accepting Peering Request



Snapshot: Accepting Peering Request

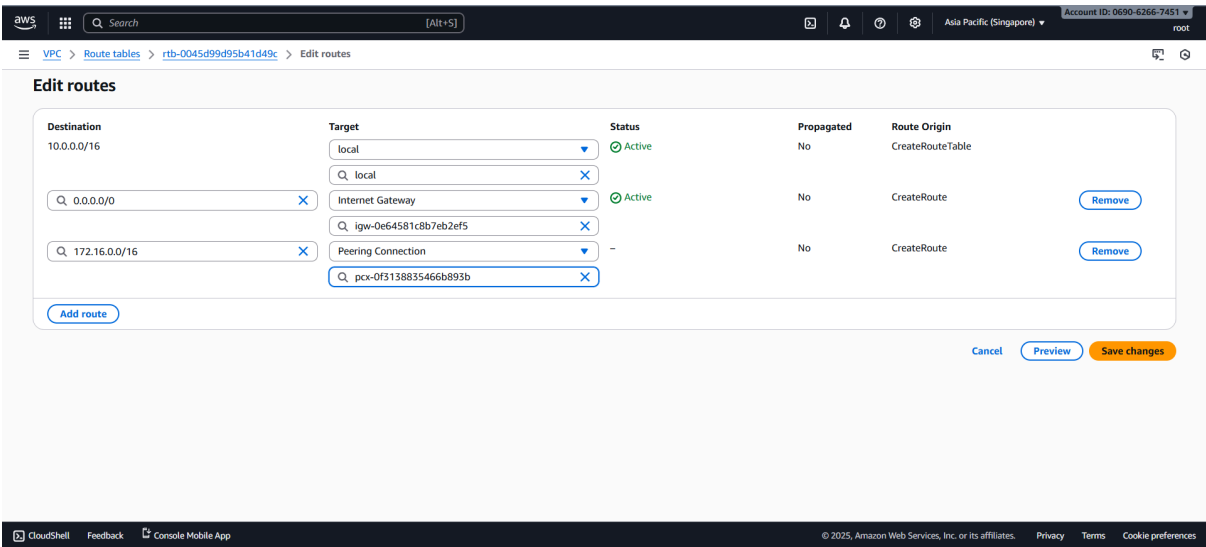
# 6. Update Route Tables

After the peering is active, update the routes.

## Singapore Route Table (RT-A)

Add route:

- Destination: 172.16.0.0/16
- Target: Peering Connection



Snapshot: Updating Route Tables

## Ireland Route Table (RT-B)

Add route:

- Destination: 10.0.0.0/16
- Target: Peering IConnection

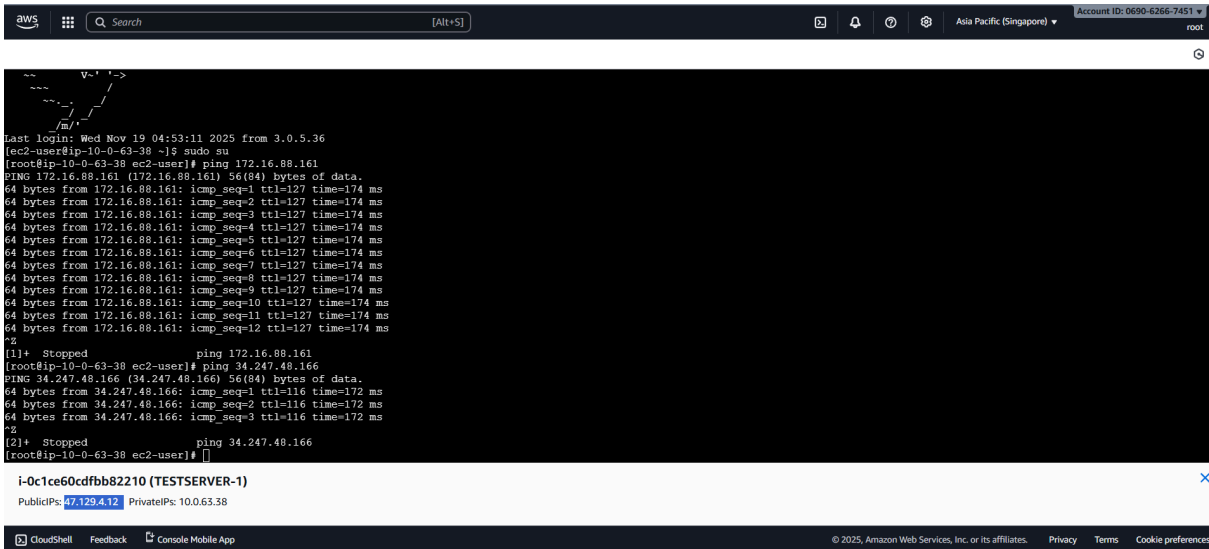
# 7. Test Private Connectivity

SSH into EC2-A and run:

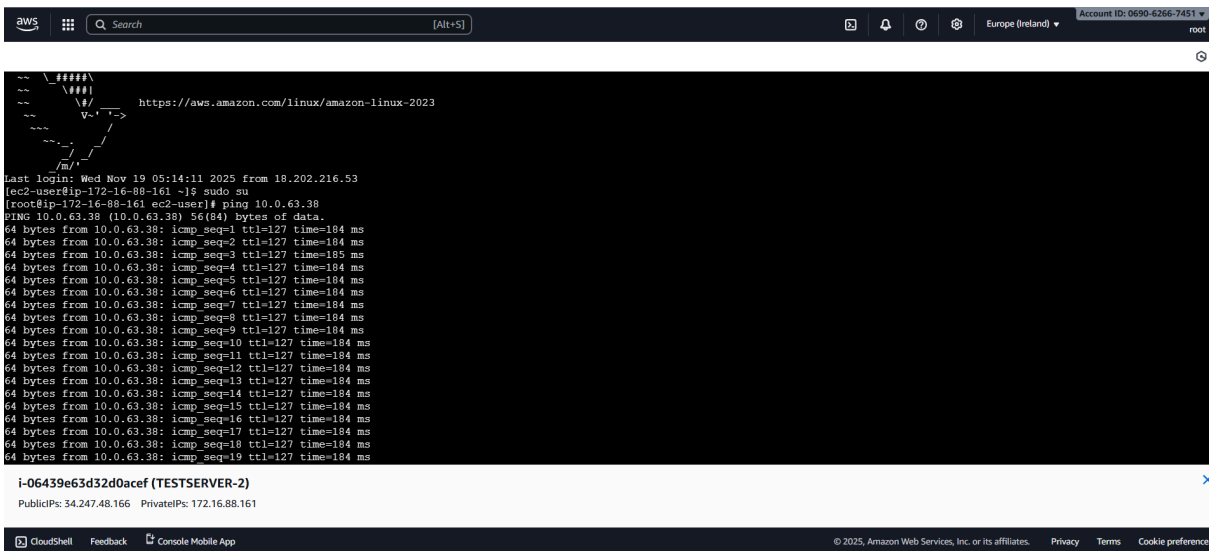
ping 172.16.88.161

SSH into EC2-B and run:

ping 10.0.63.38



## Snapshot: Testing Private Connectivity



## Snapshot: Testing Private Connectivity

```
aws
[Alt+S]
Europe (Ireland)
Account ID: 0690-6266-7451
root

64 bytes from 10.0.63.38: icmp_seq=15 ttl=127 time=184 ms
64 bytes from 10.0.63.38: icmp_seq=16 ttl=127 time=184 ms
64 bytes from 10.0.63.38: icmp_seq=17 ttl=127 time=184 ms
64 bytes from 10.0.63.38: icmp_seq=18 ttl=127 time=184 ms
64 bytes from 10.0.63.38: icmp_seq=19 ttl=127 time=184 ms
^Z
[1]+  Stopped                  ping 10.0.63.38
[root@ip-172-16-88-161 ec2-user]# ping 47.129.4.12
PING 47.129.4.12 (47.129.4.12) 56(84) bytes of data.
64 bytes from 47.129.4.12: icmp_seq=1 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=2 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=3 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=4 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=5 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=6 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=7 ttl=116 time=173 ms
64 bytes from 47.129.4.12: icmp_seq=8 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=9 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=10 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=11 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=12 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=13 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=14 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=15 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=16 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=17 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=18 ttl=116 time=172 ms
64 bytes from 47.129.4.12: icmp_seq=19 ttl=116 time=172 ms
^Z
[2]+  Stopped                  ping 47.129.4.12
[root@ip-172-16-88-161 ec2-user]#
```

i-06439e63d32d0acef (TESTSERVER-2)

PublicIPs: 34.247.48.166 PrivateIPs: 172.16.88.161

CloudShell Feedback Console Mobile App

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

## Snapshot: Testing Private Connectivity

# 8. Final Outcome

By completing this setup:

- Both EC2 instances communicate privately across regions