

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**.

The screenshot shows the 'Create VPC' wizard in the AWS VPC service. The first step, 'VPC settings', is selected. The 'Resources to create' dropdown is set to 'VPC only'. The 'Name tag' field contains 'VPC-1'. The 'IPv4 CIDR block' dropdown is set to '10.0.0.0/16'. The 'Tenancy' dropdown is set to 'Default'. In the 'VPC encryption controls (Beta) - optional' section, the 'None' option is selected. A note about 'Additional charges apply' is visible. At the bottom, there are 'Cancel', 'Preview code', and a large orange 'Create VPC' button.

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

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

VPC settings

Resources to create: [info](#) Create resources required to support my VPC and other networking resources.

VPC only VPC and more

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

VPC-2

IPv4 CIDR Block [info](#)

- IPv4 CIDR block
- IPv4+CIDR block
- IAM-allocated IPv4 CIDR block
- Shared IPv4+CIDR block
- IPv4 CIDR derived from route table

IPv4 CIDR: 172.16.0.0/16

Subnets must be between /16 and /24.

IPv4 CIDR Block [info](#)

- No IPv4 CIDR block
- IAM-allocated IPv4 CIDR block
- Shared IPv4+CIDR block
- IPv4 CIDR derived from route table

Traffic [info](#)

- Default
- VPC encryption control (new) [info](#) This mode provides visibility into encrypted traffic. It's also mode prevents unencrypted traffic. Additional charges apply.

VPC encryption control (new) [info](#) This mode provides visibility into encrypted traffic. It's also mode prevents unencrypted traffic. Additional charges apply.

- None
- Monitor mode See which resources in your VPC are encrypted or decrypted. This mode is best suited for monitoring of unencrypted resources.
- Enforce mode Requires all resources, except exclusions, in your VPC to be encrypted. This mode is best suited for 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: Name Value: optional

Add tag Remove tag

You can add 40 more tags.

[Cancel](#) [Preview code](#) [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

VPC [info](#) Choose the VPC in which the subnet will reside.

VPC ID: [info](#) vpc-0ac1e8aef956a4d8a (VPC-1)

Associated VPC CIDRs

IPv4 CIDR: 10.0.0.0/17

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-1

The value can be up to 256 characters long.

Availability Zone [info](#) Choose the one in which your subnet will reside, or let Amazon choose one for you.

No preferred AZ

IPv4 CIDR block [info](#) Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must be within this block.

10.0.0.0/17

IPv4 subnet CIDR block [info](#) 10.0.0.0/17 32,768 IP

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: Name Value: optional

Add new tag Remove tag

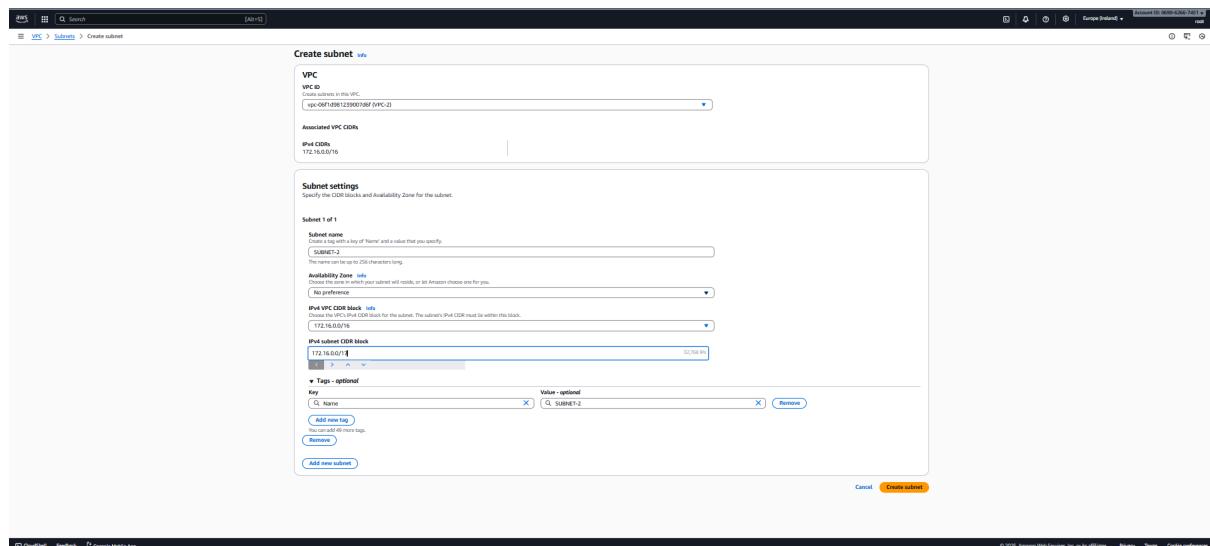
You can add 40 more tags.

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

Snapshot: Subnet creation

Ireland Subnet-2

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



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.

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC	Owner ID
-	rtb-0045d99d95b41d49c	-	-	Yes	vpc-0acdebad...	069062667451

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

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

The screenshot shows the AWS VPC dashboard with the 'Route tables' section selected. A search bar at the top has 'vpc-0acdebad950a449a' entered. Below the search is a table titled 'Route tables (1/1)'. It contains one row for 'VPC-1 ROUTE TABLE' with Route table ID 'rtb-0045d99d95b41d49c'. The 'Name' column is checked. To the right of the table are 'Actions' and 'Create route table' buttons. Below the table is a detailed view for 'rtb-0045d99d95b41d49c / VPC-1 ROUTE TABLE'. The 'Details' tab is selected, showing the route table ID, Main status (Yes), and explicit subnet associations (none). The 'Edge associations' tab shows none.

Snapshot: Renaming the Root Table

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

The screenshot shows the same AWS VPC dashboard and route table details as the previous snapshot, but with a green success message at the top: 'Updated routes for rtb-0045d99d95b41d49c / VPC-1 ROUTE TABLE successfully'. Below this message is the 'Routes' tab of the detailed view. The 'Explicit subnet associations' section shows a table with one entry: 'SUBNET-1' associated with 'subnet-06d2b9895ccdd67f8' and '10.0.0.0/17'. The 'Subnets without explicit associations' section shows a table with one entry: 'SUBNET-1' associated with 'subnet-06d2b9895ccdd67f8' and '10.0.0.0/17'. The 'Edit subnet associations' button is visible above the association table.

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

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (1/1)			
Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
<input checked="" type="checkbox"/> SUBNET-1	subnet-06d2b9895ccdd67f8	10.0.0.0/17	-

Selected subnets

subnet-06d2b9895ccdd67f8 / SUBNET-1 X
--

Buttons: Cancel, Save associations

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**.

Edit routes

Destination	Target	Status	Propagated	Route Origin
172.16.0.0/16	local	Active	No	CreateRouteTable
0.0.0.0/0	Internet Gateway	-	No	CreateRoute
	igw-038126783ca4017fa	-		Remove

Buttons: Add route, Cancel, Preview, Save changes

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.

The screenshot shows the 'Edit subnet associations' page in the AWS VPC console. At the top, there's a breadcrumb navigation: AWS > VPC > Route tables > rtb-00c3591d7658e7e4c > Edit subnet associations. The main section is titled 'Edit subnet associations' with the sub-instruction 'Change which subnets are associated with this route table.' Below this is a table titled 'Available subnets (1/1)'. It has columns for Name, Subnet ID, IPv4 CIDR, IPv6 CIDR, and Route table ID. One row is selected: 'SUBNET-2' with Subnet ID 'subnet-04f2e9904fb7767d2', IPv4 CIDR '172.16.0.0/17', and Route table ID 'Main (rtb-00c3591d7658e7e4c / ROUT...'. Below the table is a section titled 'Selected subnets' containing 'subnet-04f2e9904fb7767d2 / SUBNET-2'. At the bottom right are 'Cancel' and 'Save associations' buttons.

Snapshot: Edit Subnet Associations

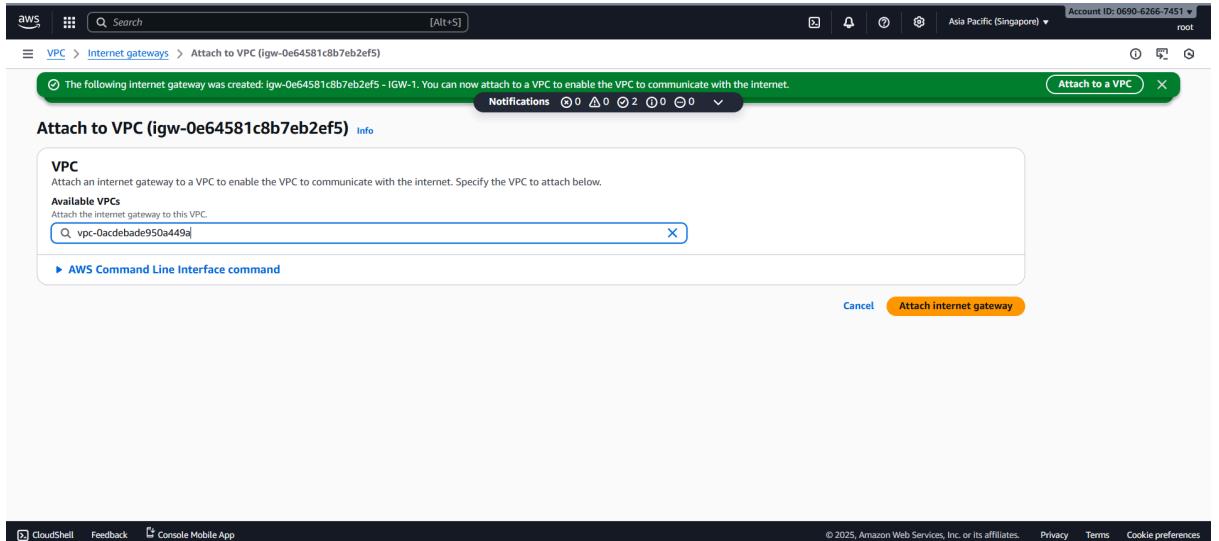
Step 5: Create Internet Gateway (IGW)

Region A IGW

- Create IGW → Attach to VPC-1

The screenshot shows the 'Create internet gateway' page in the AWS VPC console. The top navigation bar includes 'aws', a search bar, and account information 'Account ID: 0690-6266-7451'. The breadcrumb path is VPC > Internet gateways > Create internet gateway. The main form is titled 'Create internet gateway' with an info link. It contains a note: 'An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway specify the name for the gateway below.' Below this is a 'Internet gateway settings' section with a 'Name tag' field containing 'IGW-1'. There's also a 'Tags - optional' section where a single tag 'IGW-1' is listed with key 'Name' and value 'IGW-1'. At the bottom right are 'Cancel' and 'Create internet gateway' buttons.

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](#).

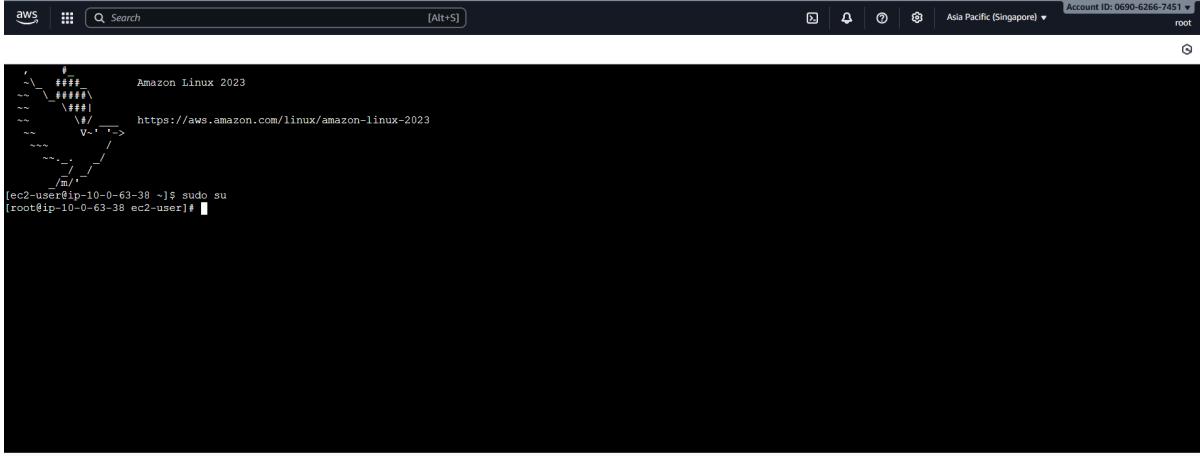
6. Configure Firewall (Security Group)

- 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)



```

Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

[ec2-user@ip-10-0-63-38 ~]$ sudo su
[root@ip-10-0-63-38 ec2-user]#

```

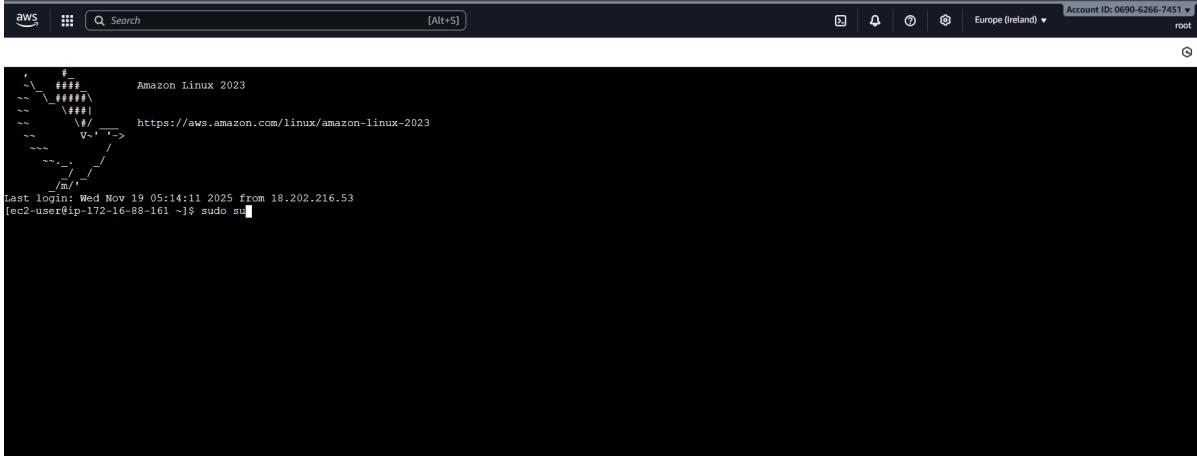
i-0c1ce60cdfbb82210 (TESTSERVER-1)
PublicIPs: 47.129.4.12 PrivateIPs: 10.0.63.38

[CloudShell](#) [Feedback](#) [Console Mobile App](#)

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

Snapshot: Testserver-1

TESTSERVER-2 (Ireland)



```

Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

Last login: Wed Nov 19 05:14:11 2025 from 18.202.216.53
[ec2-user@ip-172-16-88-161 ~]$ sudo su

```

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: 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

The screenshot shows the 'Create peering connection' dialog box. The 'Name' field is filled with 'PEERING-1'. Under 'Select a local VPC to peer with', the 'VPC ID (Requester)' dropdown shows 'vpc-0cdebd0d950da49a (VPC-1)'. Under 'Select another VPC to peer with', the 'Region' dropdown shows 'Asia Pacific (Singapore)' and the 'VPC ID (Accepter)' dropdown shows 'vpc-0ff7d981235007bf'. A single tag 'PEERING-1' is added under 'Tags'. The 'Create peering connection' button is at the bottom right.

Snapshot: Initiating Peering Request

Step 2 — Accept Peering in Ireland

- Switch region → Ireland
- Accept peering request

The screenshot shows the AWS VPC Peering connections page. A single peering connection named "pcx-0f3138835466b893b" is listed in the table. The status is "Pending acceptance". The requester VPC is "vpc-0acdebadef950a449a" and the accepter VPC is "vpc-06f1d981". On the right side, there is a context menu with options: "Accept request" (highlighted in orange), "Reject request", "Edit DNS settings", "Manage tags", and "Delete peering connection".

Snapshot: Accepting Peering Request

The screenshot shows the same AWS VPC Peering connections page. A modal dialog titled "Accept VPC peering connection request" is open over the table. It contains fields for Requester VPC (vpc-0acdebadef950a449a), Acceptor VPC (vpc-06f1d981239007d6f / VPC-2), Requester CIDRs (10.0.0.0/16), Acceptor CIDRs (10.0.0.0/16), Requester Region (Singapore (ap-southeast-1)), Acceptor Region (Ireland (eu-west-1)), Requester owner ID (069062667451 (This account)), and Acceptor owner ID (069062667451 (This account)). At the bottom of the dialog are "Cancel" and "Accept request" buttons.

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

The screenshot shows the AWS VPC Route Tables interface. The top navigation bar includes the AWS logo, search bar, and account information (Account ID: 0690-6266-7451, Region: Asia Pacific (Singapore)). The current page is 'Route tables > rtb-0045d99d95b41d49c > Edit routes'. The main section is titled 'Edit routes' and contains a table with one existing route and one new route being added. The table columns are: Destination, Target, Status, Propagated, and Route Origin. The existing route is for destination 10.0.0.0/16, target local, status Active, propagated No, and origin CreateRouteTable. The new route being added has a destination of 172.16.0.0/16, a target of Peering Connection (with a dropdown value of pnx-0f3138835466b893b), and a status of Active. The 'Add route' button is at the bottom left, and 'Save changes' is at the bottom right. The footer includes links for CloudShell, Feedback, and Console Mobile App, along with copyright information (© 2025, Amazon Web Services, Inc. or its affiliates) and links for Privacy, Terms, and Cookie preferences.

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
```

The screenshot shows the AWS CloudShell interface. The terminal window displays the output of a ping command from EC2-A (IP 172.16.88.161) to EC2-B (IP 10.0.63.38). The output shows multiple ICMP echo requests being sent with TTL=127, and the responses from EC2-B arriving with TTL=116. The session includes two stopped tasks, one for each ping command.

```
aws | Search [Alt+S] Account ID: 0690-6266-7451 | root | Asia Pacific (Singapore) | Last login: Wed Nov 19 04:53:11 2025 from 3.0.5.36 [ec2-user@ip-10-0-63-38 ~]$ sudo su [root@ip-10-0-63-38 ~]# ping 172.16.88.161 PING 172.16.88.161 (172.16.88.161) 56(84) bytes of data. 64 bytes from 172.16.88.161: icmp_seq=1 ttl=127 time=174 ms 64 bytes from 172.16.88.161: icmp_seq=2 ttl=127 time=174 ms 64 bytes from 172.16.88.161: icmp_seq=3 ttl=127 time=174 ms 64 bytes from 172.16.88.161: icmp_seq=4 ttl=127 time=174 ms 64 bytes from 172.16.88.161: icmp_seq=5 ttl=127 time=174 ms 64 bytes from 172.16.88.161: icmp_seq=6 ttl=127 time=174 ms 64 bytes from 172.16.88.161: icmp_seq=7 ttl=127 time=174 ms 64 bytes from 172.16.88.161: icmp_seq=8 ttl=127 time=174 ms 64 bytes from 172.16.88.161: icmp_seq=9 ttl=127 time=174 ms 64 bytes from 172.16.88.161: icmp_seq=10 ttl=127 time=174 ms 64 bytes from 172.16.88.161: icmp_seq=11 ttl=127 time=174 ms 64 bytes from 172.16.88.161: icmp_seq=12 ttl=127 time=174 ms [1]+ Stopped ping 172.16.88.161 [root@ip-10-0-63-38 ec2-user]# ping 34.247.48.166 PING 34.247.48.166 (34.247.48.166) 56(84) bytes of data. 64 bytes from 34.247.48.166: icmp_seq=1 ttl=116 time=172 ms 64 bytes from 34.247.48.166: icmp_seq=2 ttl=116 time=172 ms 64 bytes from 34.247.48.166: icmp_seq=3 ttl=116 time=172 ms [2]+ Stopped ping 34.247.48.166 [root@ip-10-0-63-38 ec2-user]# i-0c1ce60cdffbb82210 (TESTSERVER-1) PublicIPs: 47.129.4.12 PrivateIPs: 10.0.63.38
```

Snapshot: Testing Private Connectivity

The screenshot shows the AWS CloudShell interface. The terminal window displays the output of a ping command from EC2-B (IP 10.0.63.38) to EC2-A (IP 172.16.88.161). The output shows multiple ICMP echo requests being sent with TTL=127, and the responses from EC2-A arriving with TTL=127. The session includes two stopped tasks, one for each ping command.

```
aws | Search [Alt+S] Account ID: 0690-6266-7451 | root | Europe (Ireland) | Last login: Wed Nov 19 05:14:11 2025 from 18.202.216.53 [ec2-user@ip-172-16-88-161 ~]$ sudo su [root@ip-172-16-88-161 ~]# ping 10.0.63.38 PING 10.0.63.38 (10.0.63.38) 56(84) bytes of data. 64 bytes from 10.0.63.38: icmp_seq=1 ttl=127 time=184 ms 64 bytes from 10.0.63.38: icmp_seq=2 ttl=127 time=184 ms 64 bytes from 10.0.63.38: icmp_seq=3 ttl=127 time=185 ms 64 bytes from 10.0.63.38: icmp_seq=4 ttl=127 time=183 ms 64 bytes from 10.0.63.38: icmp_seq=5 ttl=127 time=184 ms 64 bytes from 10.0.63.38: icmp_seq=6 ttl=127 time=184 ms 64 bytes from 10.0.63.38: icmp_seq=7 ttl=127 time=184 ms 64 bytes from 10.0.63.38: icmp_seq=8 ttl=127 time=184 ms 64 bytes from 10.0.63.38: icmp_seq=9 ttl=127 time=180 ms 64 bytes from 10.0.63.38: icmp_seq=10 ttl=127 time=184 ms 64 bytes from 10.0.63.38: icmp_seq=11 ttl=127 time=184 ms 64 bytes from 10.0.63.38: icmp_seq=12 ttl=127 time=184 ms 64 bytes from 10.0.63.38: icmp_seq=13 ttl=127 time=184 ms 64 bytes from 10.0.63.38: icmp_seq=14 ttl=127 time=184 ms 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 [1]+ Stopped ping 172.16.88.161 [root@ip-172-16-88-161 ~]# i-06439e63d32d0acf (TESTSERVER-2) PublicIPs: 34.247.48.166 PrivateIPs: 172.16.88.161
```

Snapshot: Testing Private Connectivity

```

AWS CloudShell | Search [Alt+S] | Account ID: 0690-6266-7451 | Europe (Ireland) | root
[1]+ Stopped ping 10.0.63.38
[root@ip-172-16-88-161 ~]# 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
[1]+ Stopped ping 10.0.63.38
[root@ip-172-16-88-161 ~]#
[2]+ Stopped ping 47.129.4.12
[root@ip-172-16-88-161 ~]#

```

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