

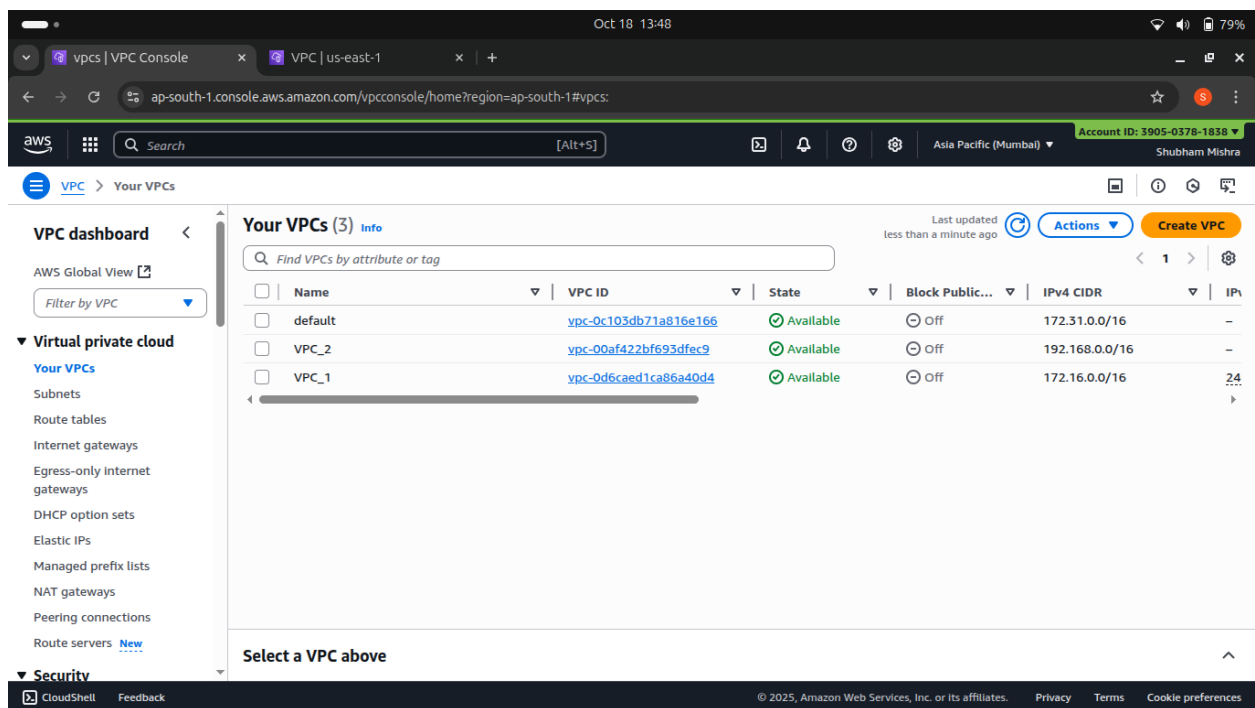
Inter-VPC Communication using AWS VPC Peering

Overview

Established secure private communication between two AWS VPCs using VPC Peering, enabling EC2 instances in both VPCs to communicate without using the public internet.

This project demonstrates network isolation, route management, and private connectivity between multi-tier environments.

Step 1) Created two vpc and i was default vpc



Step 2) Created two public and two private subnet for vpc1 and vpc2 also created private route table for both vpc1 and vpc2 and routed accordingly as shown below also created internet gateway in vpc1 and vpc2 and added route in public route table of both vpc1 and vpc2

The screenshot shows the AWS VPC console interface. On the left is a navigation sidebar with categories like 'Virtual private cloud', 'Security', and 'PrivateLink and Lattice'. The main content area displays 'Your VPCs (1/3)' in a table. The table has columns for Name, VPC ID, State, Block Public..., IPv4 CIDR, IPv6 CIDR, DHCP option set, and Mail. Three VPCs are listed: 'default', 'VPC_2', and 'VPC_1'. 'VPC_1' is selected. Below the table, the 'Resource map' for 'vpc-0d6caed1ca86a40d4 / VPC_1' is shown. It includes a diagram with boxes for 'VPC', 'Subnets (4)', 'Route tables (2)', and 'Network Connections (1)', connected by lines representing network topology.

Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR	DHCP option set	Mail
default	vpc-0c103db71a816e166	Available	Off	172.31.0.0/16	-	dopt-058b70b7b9994c...	rtb-
VPC_2	vpc-00af422bf693dfec9	Available	Off	192.168.0.0/16	-	dopt-058b70b7b9994c...	rtb-
VPC_1	vpc-0d6caed1ca86a40d4	Available	Off	172.16.0.0/16	2406:da1a:2d9:de00::/56	dopt-058b70b7b9994c...	rtb-

This screenshot shows the 'Details' tab for 'vpc-00af422bf693dfec9 / VPC_2'. The details are organized into several sections: 'VPC ID', 'State', 'Block Public Access', 'DNS hostnames', 'DNS resolution', 'Tenancy', 'DHCP option set', 'Main network ACL', 'Default VPC', 'IPv4 CIDR', 'Route 53 Resolver DNS Firewall rule groups', 'IPv6 CIDR (Network border group)', 'Network Address Usage metrics', 'Main route table', 'IPv6 pool', and 'Owner ID'. Each section contains specific configuration details for the selected VPC.

Section	Value
VPC ID	vpc-00af422bf693dfec9
State	Available
Block Public Access	Off
DNS hostnames	Disabled
DNS resolution	Enabled
Tenancy	default
DHCP option set	dopt-058b70b7b9994c329
Main network ACL	acl-0b34cc01d1645e2a7
Default VPC	No
IPv4 CIDR	192.168.0.0/16
Route 53 Resolver DNS Firewall rule groups	-
IPv6 CIDR (Network border group)	-
Network Address Usage metrics	Disabled
Main route table	rtb-0c63017f70965af7f
IPv6 pool	-
Owner ID	390503781838

Step 3) Now we create Natgateway for private subnet internet access

Oct 18 13:57

VPC | ap-south-1 x VPC | us-east-1 x +

ap-south-1.console.aws.amazon.com/vpconsole/home?region=ap-south-1#CreateNatGateway:

Search [Alt+S]

Asia Pacific (Mumbai) Account ID: 3905-0378-1838 Shubham Mishra

VPC > NAT gateways > Create NAT gateway

Elastic IP address 65.1.245.200 (eipalloc-0bdc52d5cddc176fe) allocated.

NAT gateway settings

Name - optional
Create a tag with a key of 'Name' and a value that you specify.
The name can be up to 256 characters long.
NAT-VPC1

Subnet
Select a subnet in which to create the NAT gateway.
subnet-044cfea4455da2e6e (Public_subnet_2_vpc1)

Connectivity type
Select a connectivity type for the NAT gateway.
☒ Public
☐ Private

Elastic IP allocation ID [Info](#)
Assign an Elastic IP address to the NAT gateway.
eipalloc-0bdc52d5cddc176fe [Allocate Elastic IP](#)

[Additional settings](#) [Info](#)

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	NAT-VPC1	Remove

[Add new tag](#)

You can add 49 more tags.

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

Oct 18 13:58

VPC | ap-south-1 x VPC | us-east-1 x +

ap-south-1.console.aws.amazon.com/vpconsole/home?region=ap-south-1#CreateNatGateway:

Search [Alt+S]

Asia Pacific (Mumbai) Account ID: 3905-0378-1838 Shubham Mishra

VPC > NAT gateways > Create NAT gateway

Elastic IP address 43.204.94.154 (eipalloc-012ab32a385f599b1) allocated.

NAT gateway settings

Name - optional
Create a tag with a key of 'Name' and a value that you specify.
The name can be up to 256 characters long.
NAT_VPC2

Subnet
Select a subnet in which to create the NAT gateway.
subnet-0d171ae472dc1379c (Public_subnet_2_vpc2)

Connectivity type
Select a connectivity type for the NAT gateway.
☒ Public
☐ Private

Elastic IP allocation ID [Info](#)
Assign an Elastic IP address to the NAT gateway.
eipalloc-012ab32a385f599b1 [Allocate Elastic IP](#)

[Additional settings](#) [Info](#)

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	NAT_VPC2	Remove

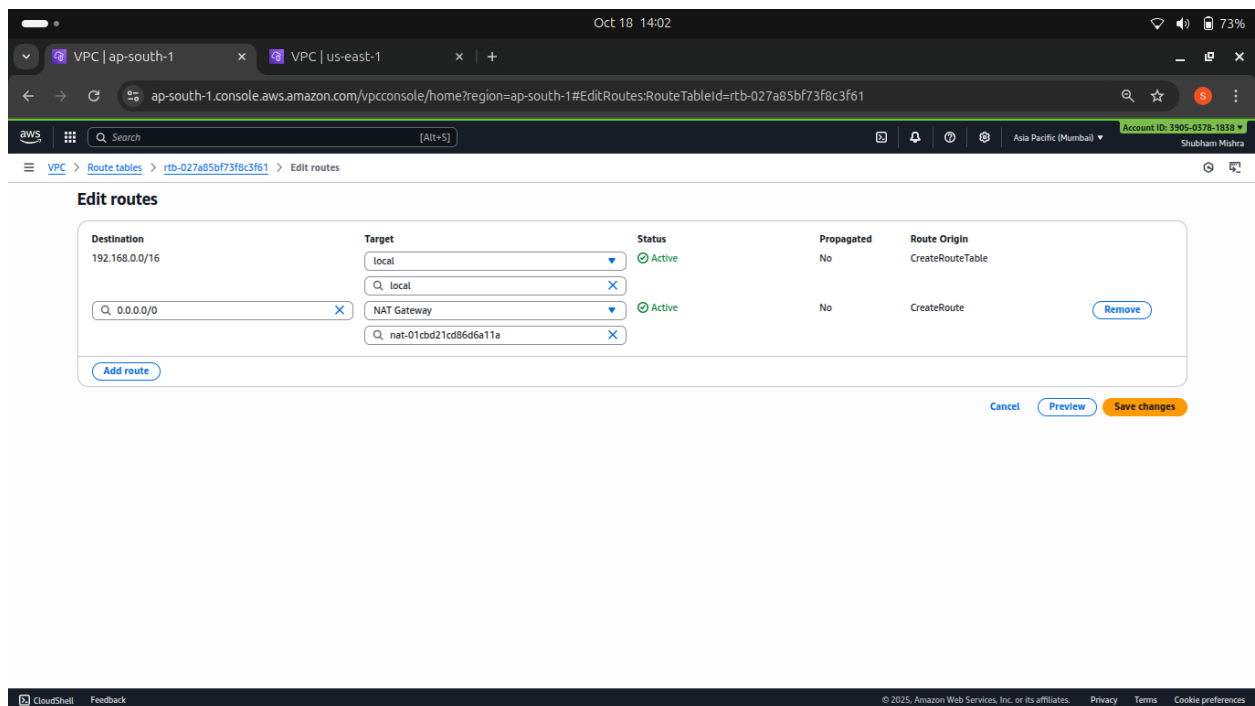
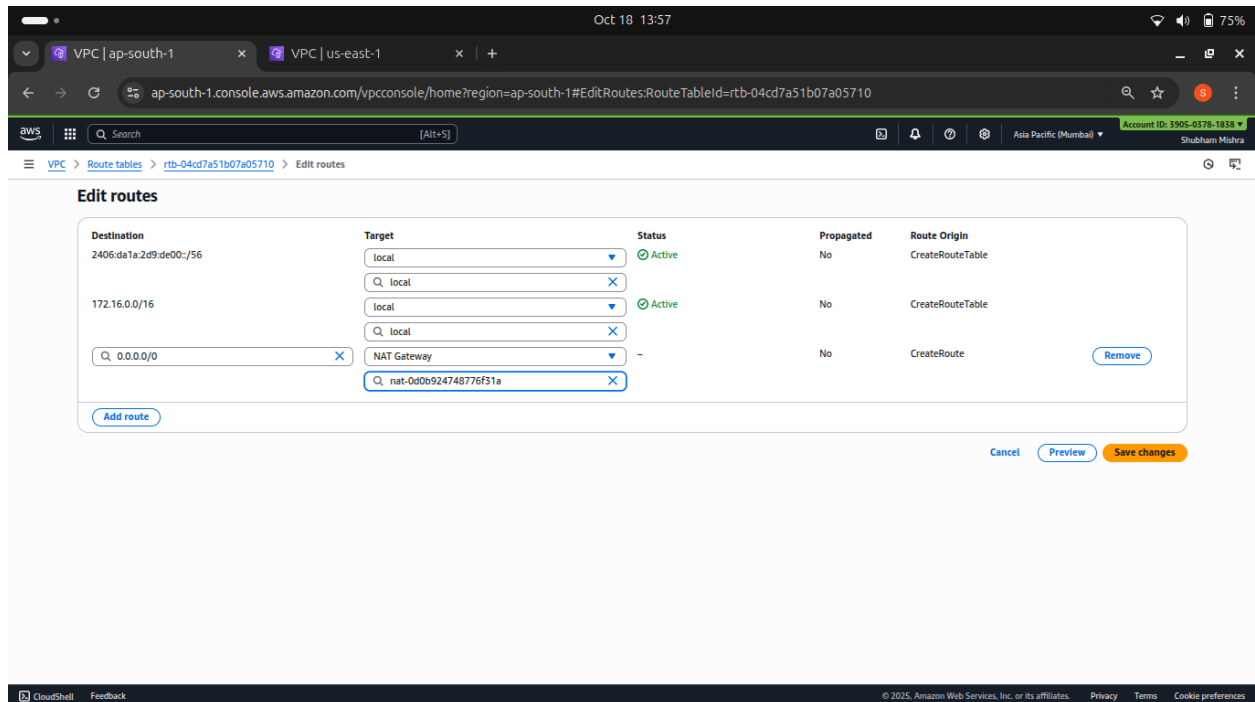
[Add new tag](#)

You can add 49 more tags.

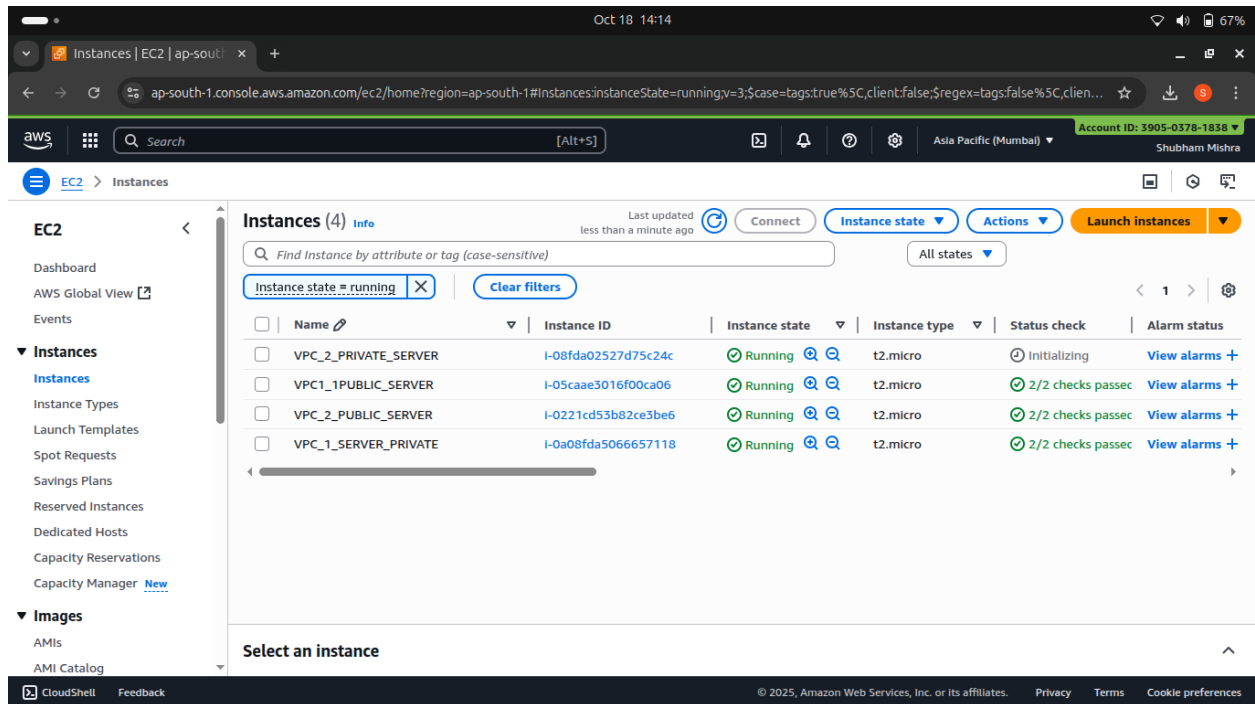
[Cancel](#) [Create NAT gateway](#)

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

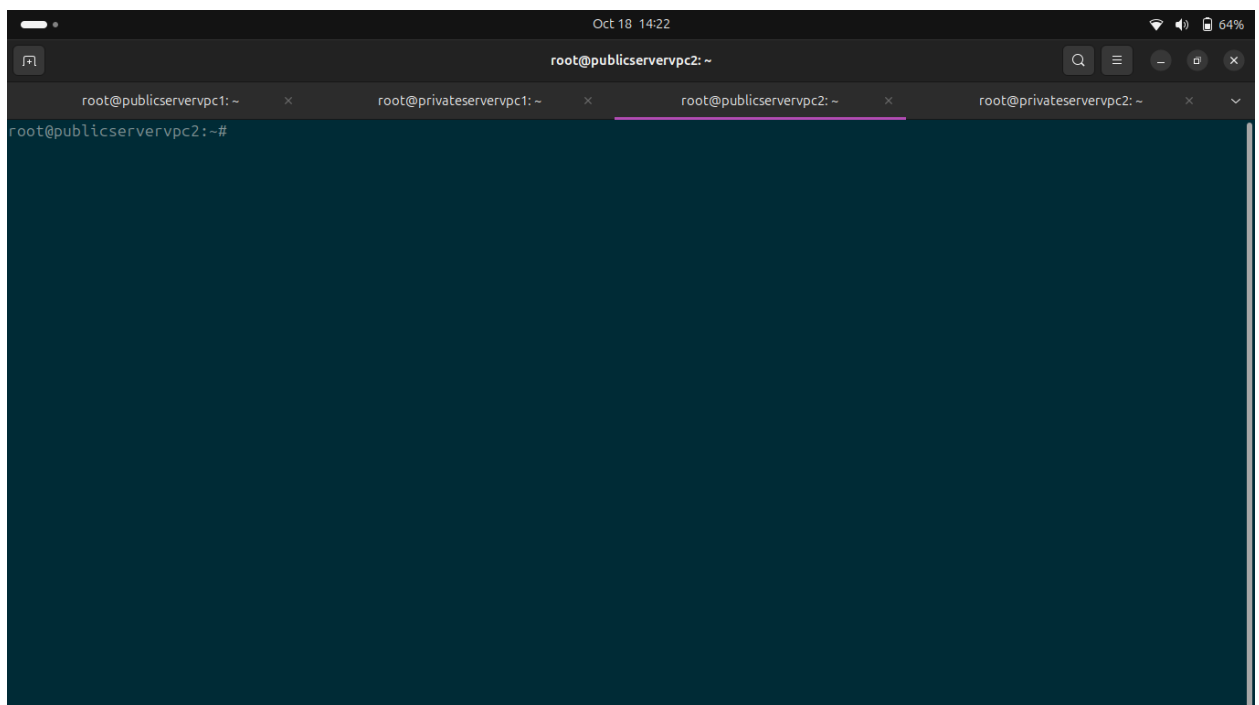
Step 4) We will update private route table to allow internet to go through natgateway



Step 5) Now We will launch 2 instances in both public and private subnet in both vpc1 and vpc2



Step 6) Now we will ssh into all 4 instances



Step 7) we will ping public and private ec2 instances in vpc1 and vpc2

```
Oct 18 14:27
root@publicservervpc1: ~
root@publicservervpc1:~# hostname -i
172.16.0.52 fe80::d1:3bff:fea9:af7d
root@publicservervpc1:~# ping 172.16.2.14
PING 172.16.2.14 (172.16.2.14) 56(84) bytes of data.
64 bytes from 172.16.2.14: icmp_seq=1 ttl=64 time=0.457 ms
64 bytes from 172.16.2.14: icmp_seq=2 ttl=64 time=0.499 ms
64 bytes from 172.16.2.14: icmp_seq=3 ttl=64 time=0.478 ms
64 bytes from 172.16.2.14: icmp_seq=4 ttl=64 time=0.491 ms
64 bytes from 172.16.2.14: icmp_seq=5 ttl=64 time=0.431 ms
64 bytes from 172.16.2.14: icmp_seq=6 ttl=64 time=0.459 ms
```

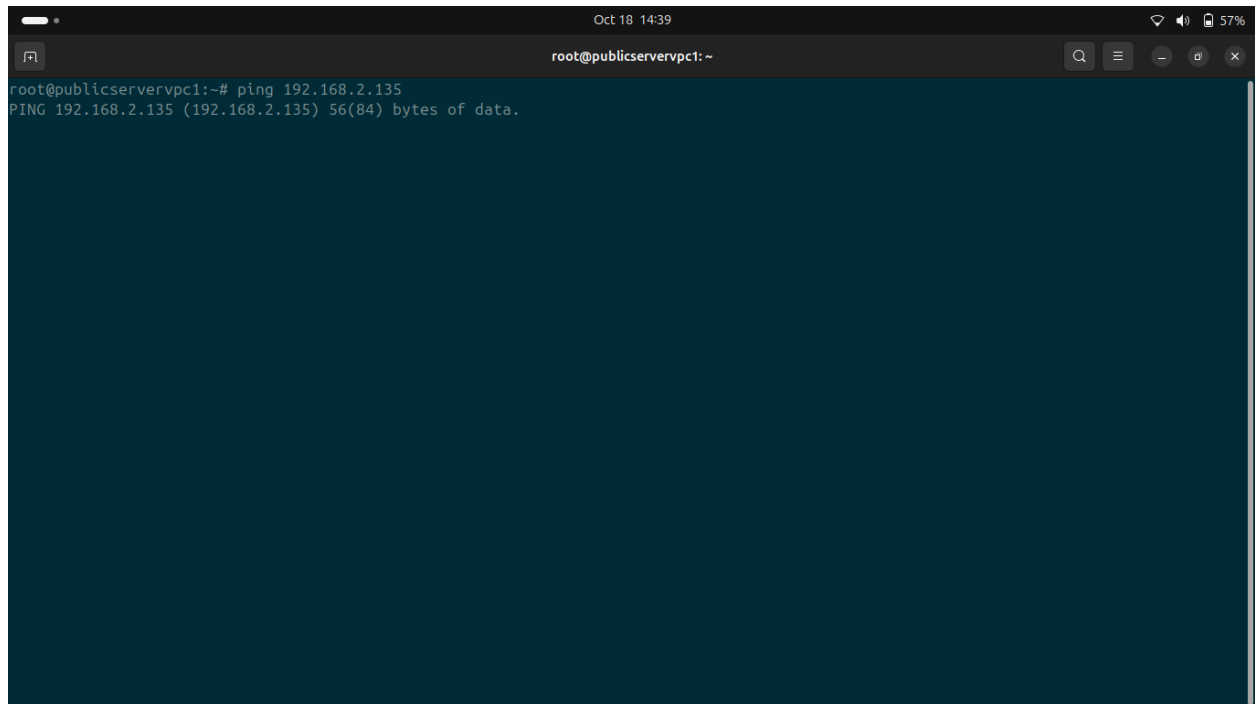
```
Oct 18 14:28
root@publicservervpc1: ~
root@publicservervpc1:~# hostname -i
172.16.0.52 fe80::d1:3bff:fea9:af7d
root@publicservervpc1:~# ping 172.16.2.14
PING 172.16.2.14 (172.16.2.14) 56(84) bytes of data.
64 bytes from 172.16.2.14: icmp_seq=1 ttl=64 time=0.457 ms
64 bytes from 172.16.2.14: icmp_seq=2 ttl=64 time=0.499 ms
64 bytes from 172.16.2.14: icmp_seq=3 ttl=64 time=0.478 ms
64 bytes from 172.16.2.14: icmp_seq=4 ttl=64 time=0.491 ms
64 bytes from 172.16.2.14: icmp_seq=5 ttl=64 time=0.431 ms
64 bytes from 172.16.2.14: icmp_seq=6 ttl=64 time=0.459 ms
64 bytes from 172.16.2.14: icmp_seq=7 ttl=64 time=0.552 ms
64 bytes from 172.16.2.14: icmp_seq=8 ttl=64 time=0.550 ms
64 bytes from 172.16.2.14: icmp_seq=9 ttl=64 time=0.432 ms
64 bytes from 172.16.2.14: icmp_seq=10 ttl=64 time=0.532 ms
64 bytes from 172.16.2.14: icmp_seq=11 ttl=64 time=0.444 ms
64 bytes from 172.16.2.14: icmp_seq=12 ttl=64 time=0.492 ms
```

a) Here both ec2 instances in same vpc1 can ping over private ip

```
Oct 18 14:29
root@publicservervpc2: ~
root@publicservervpc2: ~
root@publicservervpc2:~# hostname -i
192.168.0.137 fe80::c5:45ff:fe38:191f
root@publicservervpc2:~# ping 192.168.2.135
PING 192.168.2.135 (192.168.2.135) 56(84) bytes of data.
64 bytes from 192.168.2.135: icmp_seq=1 ttl=64 time=0.390 ms
64 bytes from 192.168.2.135: icmp_seq=2 ttl=64 time=0.417 ms
64 bytes from 192.168.2.135: icmp_seq=3 ttl=64 time=0.293 ms
64 bytes from 192.168.2.135: icmp_seq=4 ttl=64 time=0.884 ms
64 bytes from 192.168.2.135: icmp_seq=5 ttl=64 time=1.22 ms
```

```
Oct 18 14:30
root@privateservervpc2: ~
root@publicservervpc2: ~
root@privateservervpc2:~# hostname -i
192.168.2.135 fe80::86:2bff:fe1d:80ab
root@privateservervpc2:~# 192.168.0.137
192.168.0.137: command not found
root@privateservervpc2:~# ping 192.168.0.137
PING 192.168.0.137 (192.168.0.137) 56(84) bytes of data.
64 bytes from 192.168.0.137: icmp_seq=1 ttl=64 time=1.26 ms
64 bytes from 192.168.0.137: icmp_seq=2 ttl=64 time=0.357 ms
```

b) Here both instances in vpc2 can also ping over private ip

A terminal window with a dark background. The title bar at the top shows 'Oct 18 14:39' and '57%' battery. The terminal prompt is 'root@publicservervpc1: ~'. The command 'ping 192.168.2.135' has been executed, and the output is 'PING 192.168.2.135 (192.168.2.135) 56(84) bytes of data.'

```
root@publicservervpc1:~# ping 192.168.2.135
PING 192.168.2.135 (192.168.2.135) 56(84) bytes of data.
```

c) But now when trying to ping over private ip in different vpc no response

Step 8) Now we will establish vpc peering so that inter vpc communication can be established

The first screenshot shows the 'Create peering connection' page in the AWS Management Console. Under 'Peering connection settings', the 'Name' is 'PEERING-BETWEEN-VPC1-AND-VPC2'. The 'Requester VPC' is 'vpc-0d6caed1ca86a40d4 (VPC_1)'. Its CIDRs are 172.16.0.0/16 and 2406:da1a:2d9:de00::/56, both with an 'Associated' status. The 'Accepter VPC' is 'vpc-00af422bf693dfec9 (VPC_2)'. Its CIDRs are 193.129.0.0/16 and 2406:da1a:2d9:de00::/56, both with an 'Associated' status. The region is 'ap-south-1'.

The second screenshot shows the 'Peering connection details' page for 'pcx-0f741aa18bcb61b1 / PEERING-BETWEEN-VPC1-AND-VPC2'. The status is 'Pending acceptance'. A modal dialog 'Accept VPC peering connection request' is open, asking for confirmation. It lists the requester and accepter VPCs, their CIDRs, and the region (Mumbai, ap-south-1). The dialog has 'Cancel' and 'Accept request' buttons.

Step 9) Now we will update route table for other vpc cidr block to go through vpc peering

Oct 18 14:46

VPC | ap-south-1

ap-south-1.console.aws.amazon.com/vpconsole/home?region=ap-south-1#EditRoutes:RouteTableId=rtb-04cd7a51b07a05710

Account ID: 3905-0378-1838

Shubham Mishra

VPC > Route tables > rtb-04cd7a51b07a05710 > Edit routes

Edit routes

Destination	Target	Status	Propagated	Route Origin	
2406:da1a:2d9:de00::/56	local	Active	No	CreateRouteTable	
	Q local				
172.16.0.0/16	local	Active	No	CreateRouteTable	
	Q local				
Q 0.0.0.0/0	NAT Gateway	Active	No	CreateRoute	Remove
	Q nat-0d0b924748776f31a				
Q 192.168.0.0/16	Peering Connection	-	No	CreateRoute	Remove
	Q pcx-0f741aa18bcb61b1				

Add route

Cancel Preview Save changes

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

Oct 18 14:46

VPC | ap-south-1

ap-south-1.console.aws.amazon.com/vpconsole/home?region=ap-south-1#EditRoutes:RouteTableId=rtb-04f75c681bb9d60c8

Account ID: 3905-0378-1838

Shubham Mishra

VPC > Route tables > rtb-04f75c681bb9d60c8 > Edit routes

Edit routes

Destination	Target	Status	Propagated	Route Origin	
2406:da1a:2d9:de00::/56	local	Active	No	CreateRouteTable	
	Q local				
172.16.0.0/16	local	Active	No	CreateRouteTable	
	Q local				
Q 0.0.0.0/0	Internet Gateway	Active	No	CreateRoute	Remove
	Q igw-0b37a2e2127b457d5				
Q 192.168.0.0/16	Peering Connection	-	No	CreateRoute	Remove
	Q pcx-0f741aa18bcb61b1				

Add route

Cancel Preview Save changes

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

Oct 18 14:47

VPC | ap-south-1

ap-south-1.console.aws.amazon.com/vpconsole/home?region=ap-south-1#EditRoutes:RouteTableId=rtb-0c63017f70965af7f

Search [Alt+S]

Asia Pacific (Mumbai)

Account ID: 3905-0378-1838

Shubham Mishra

VPC > Route tables > rtb-0c63017f70965af7f > Edit routes

Edit routes

Destination	Target	Status	Propagated	Route Origin
192.168.0.0/16	local	Active	No	CreateRouteTable
0.0.0.0/0	Internet Gateway	Active	No	CreateRoute
172.16.0.0/16	Peering Connection	-	No	CreateRoute

Add route

Cancel Preview Save changes

CloudShell Feedback

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

Oct 18 14:48

VPC | ap-south-1

ap-south-1.console.aws.amazon.com/vpconsole/home?region=ap-south-1#EditRoutes:RouteTableId=rtb-027a85bf73f8c3f61

Search [Alt+S]

Asia Pacific (Mumbai)

Account ID: 3905-0378-1838

Shubham Mishra

VPC > Route tables > rtb-027a85bf73f8c3f61 > Edit routes

Edit routes

Destination	Target	Status	Propagated	Route Origin
192.168.0.0/16	local	Active	No	CreateRouteTable
0.0.0.0/0	NAT Gateway	Active	No	CreateRoute
172.16.0.0/16	Peering Connection	-	No	CreateRoute

Add route

Cancel Preview Save changes

CloudShell Feedback

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

Step 10) Now we will test again ping on private ip of vpc1 instance from vpc2 instance and viceversa

```
Oct 18 14:49
root@publicservervpc2: ~
root@publicservervpc2: ~
root@publicservervpc2:~# hostname -i
192.168.0.137 fe80::c5:45ff:fe38:191f
root@publicservervpc2:~# ping 172.16.0.52
PING 172.16.0.52 (172.16.0.52) 56(84) bytes of data.
64 bytes from 172.16.0.52: icmp_seq=1 ttl=64 time=0.807 ms
64 bytes from 172.16.0.52: icmp_seq=2 ttl=64 time=0.446 ms
```

```
Oct 18 14:50
root@privateservervpc2: ~
root@publicservervpc2: ~
root@privateservervpc2:~# hostname -i
192.168.2.135 fe80::86:2bff:fe1d:80ab
root@privateservervpc2:~# ping 172.16.0.52
PING 172.16.0.52 (172.16.0.52) 56(84) bytes of data.
64 bytes from 172.16.0.52: icmp_seq=1 ttl=64 time=0.753 ms
64 bytes from 172.16.0.52: icmp_seq=2 ttl=64 time=0.478 ms
64 bytes from 172.16.0.52: icmp_seq=3 ttl=64 time=0.822 ms
```

Similarly ping on private ip of vpc2 instance from vpc1 instance successful

Vpc peering successfully implemented

 Demo / Output

- **Successful ping and SSH between instances in different VPCs**
- **Private IP-based communication verified**
- **Internet not used for cross-VPC traffic**

 Impact

This project showcases my ability to design and implement secure, scalable, and cost-effective inter-VPC communication — a fundamental concept in Cloud Networking and DevOps architectures.