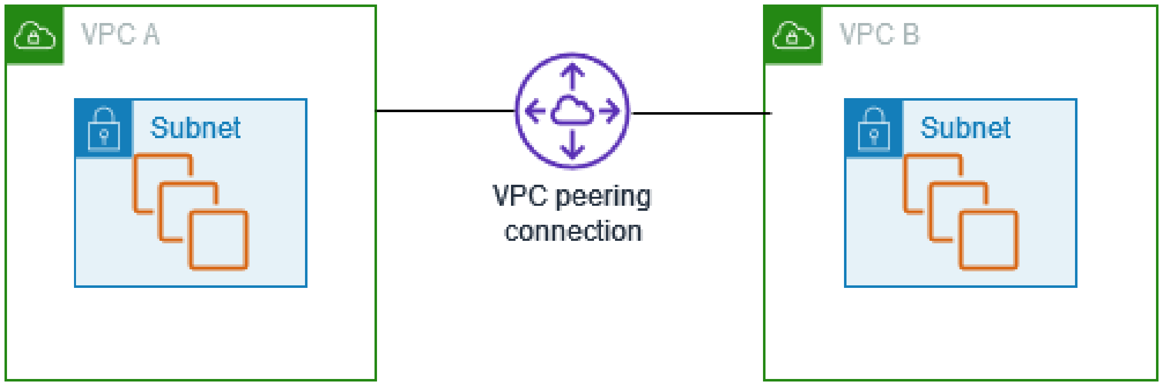
Assignment -1:-

Q. Could you please explain the process of creating peering in two different regions of the VPC?

VPC Peering:

A *virtual private cloud* (VPC) is a virtual network dedicated to your AWS account. It is logically isolated from other virtual networks in the AWS Cloud. You can launch AWS resources, such as Amazon EC2 instances, into your VPC.

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses. Instances in either VPC can communicate with each other as if they are within the same network. You can create a VPC peering connection between your own VPCs, or with a VPC in another AWS account. The VPCs can be in different Regions (also known as an inter-Region VPC peering connection).

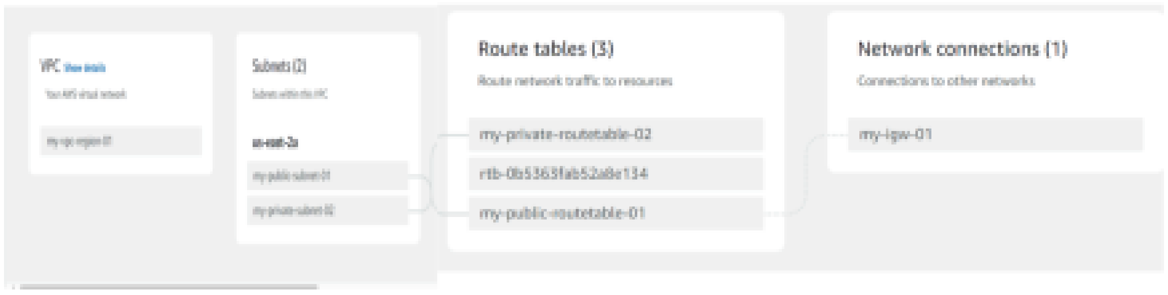


AWS uses the existing infrastructure of a VPC to create a VPC peering connection; it is neither a gateway nor a VPN connection, and does not rely on a separate piece of physical hardware. There is no single point of failure for communication or a bandwidth bottleneck.

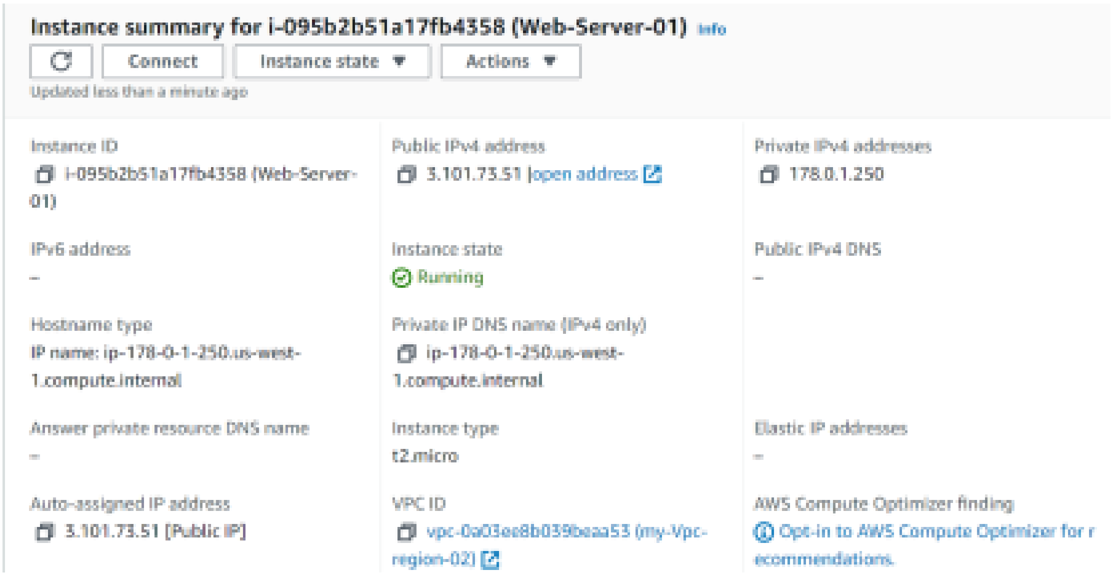
A VPC peering connection helps you to facilitate the transfer of data. For example, if you have more than one AWS account, you can peer the VPCs across those accounts to create a file sharing network. You can also use a VPC peering connection to allow other VPCs to access resources you have in one of your VPCs.

Practical:

1. Create a Vpc (Virtual Private Cloud) in a N.California region with name My-vpc-region-01 in aws console log along with the following connections attached i.e., Subnets, routing tables, Internet gateway.



2. Create an Ec2 instance with name Web-Server-01 by attaching the Vpc network connection and then launch the instance.



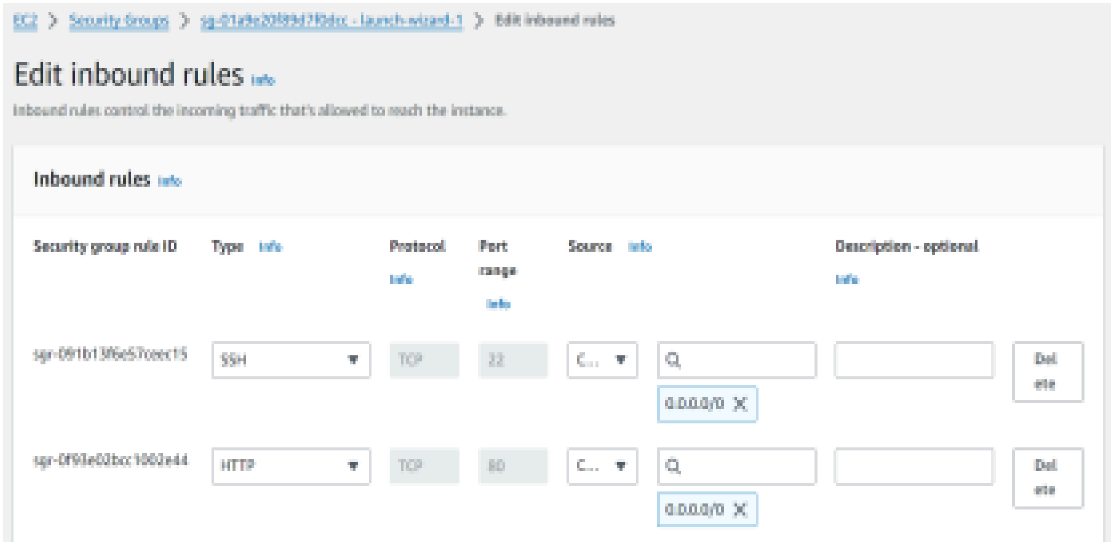
Instance summary for i-095b2b51a17fb4358 (Web-Server-01) Info

Connect | Instance state ▼ | Actions ▼

Updated less than a minute ago

Instance ID i-095b2b51a17fb4358 (Web-Server-01)	Public IPv4 address 3.101.73.51 | open address	Private IPv4 addresses 178.0.1.250
IPv6 address –	Instance state Running	Public IPv4 DNS –
Hostname type IP name: ip-178-0-1-250.us-west-1.compute.internal	Private IP DNS name (IPv4 only) ip-178-0-1-250.us-west-1.compute.internal	
Answer private resource DNS name –	Instance type t2.micro	Elastic IP addresses –
Auto-assigned IP address 3.101.73.51 [Public IP]	VPC ID vpc-0a03ee8b039beaa53 (my-Vpc-region-02)	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations.

3. Then goto the security groups in an instance and edit inbound rules add the add Http and source and save changes.



EC2 > Security Groups > sg-01a9e20f89d7f0dcc - launch-wizard-1 > Edit inbound rules

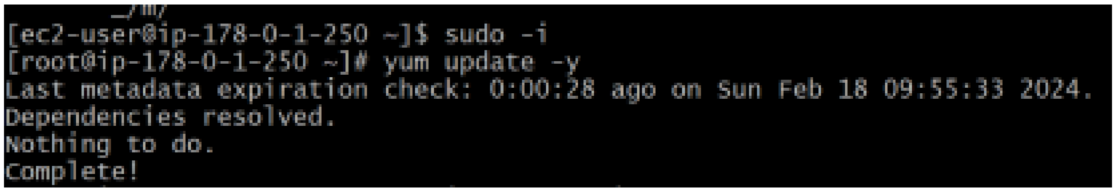
Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

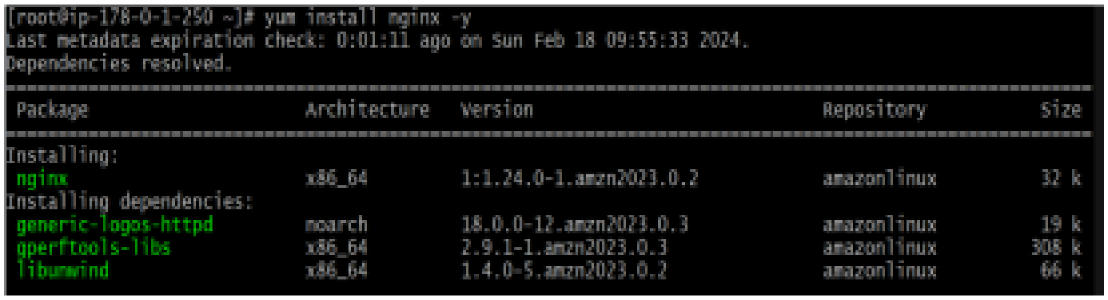
Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-091b13f6e57ceec15	SSH ▼	TCP	22	C... ▼	0.0.0.0/0 ✕		Delete
sgr-0f93e02bcc1002e44	HTTP ▼	TCP	80	C... ▼	0.0.0.0/0 ✕		Delete

4. After launching the ec2 instance, connect to the CLI and do the following commands to connect to the root user and to update all the application packages.



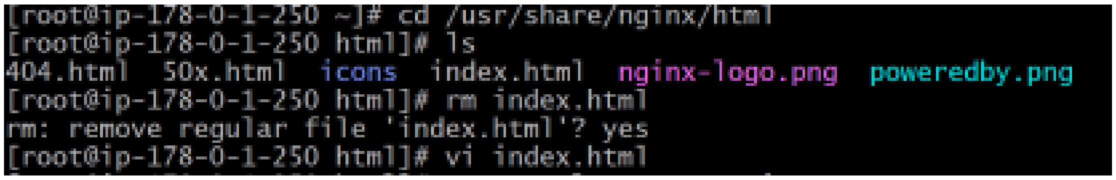
```
[ec2-user@ip-178-0-1-250 ~]$ sudo -i
[root@ip-178-0-1-250 ~]# yum update -y
Last metadata expiration check: 0:00:28 ago on Sun Feb 18 09:55:33 2024.
Dependencies resolved.
Nothing to do.
Complete!
```

5. Then install nginx which a proxy server it is used for checking if the peering connection is established in b/w the two Vpc or not.



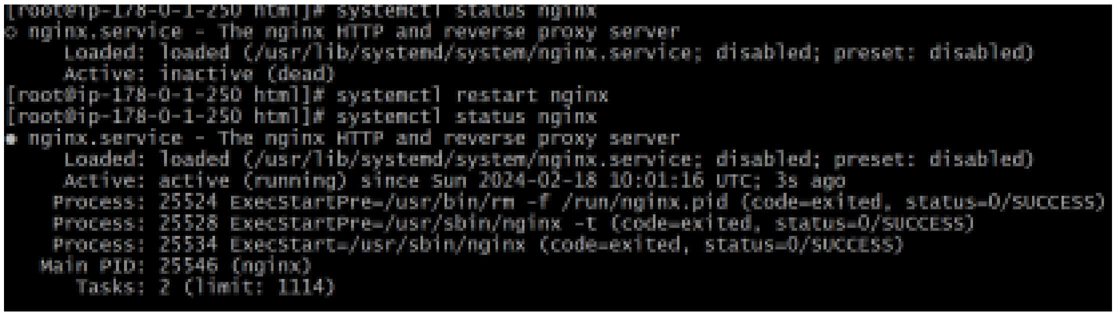
```
[root@ip-178-0-1-250 ~]# yum install nginx -y
Last metadata expiration check: 0:01:11 ago on Sun Feb 18 09:55:33 2024.
Dependencies resolved.
================================================================================================
 Package               Architecture   Version                        Repository          Size
================================================================================================
Installing:
 nginx                 x86_64         1:1.24.0-1.amzn2023.0.2        amazonlinux         32 k
Installing dependencies:
 generic-logos-httpd   noarch         18.0.0-12.amzn2023.0.3         amazonlinux         19 k
 gperftools-libs       x86_64         2.9.1-1.amzn2023.0.3           amazonlinux        308 k
 libunwind             x86_64         1.4.0-5.amzn2023.0.2           amazonlinux         66 k
```

6. After installation nginx use cd command to change directory to default address of nginx and use ls to list all the files and directory and then remove the index.html file.
7. Create a new file with a name index.html by adding some of the data in the file.



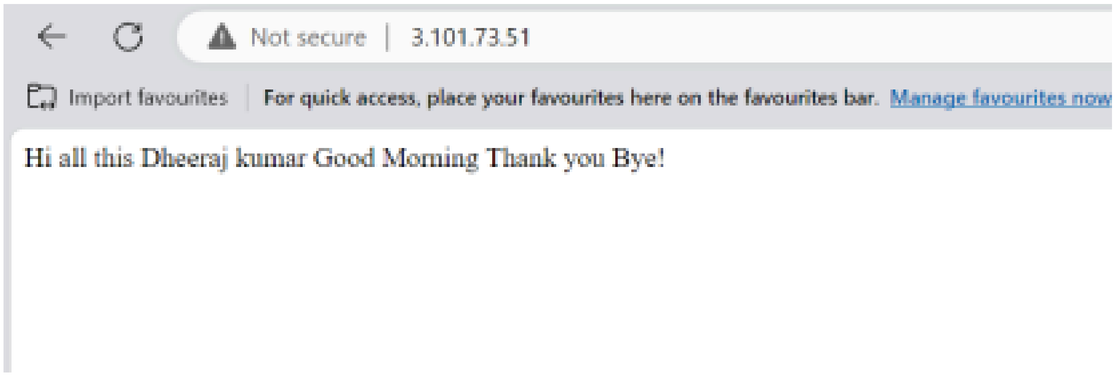
```
[root@ip-178-0-1-250 ~]# cd /usr/share/nginx/html
[root@ip-178-0-1-250 html]# ls
404.html  50x.html  icons  index.html  nginx-logo.png  poweredby.png
[root@ip-178-0-1-250 html]# rm index.html
rm: remove regular file 'index.html'? yes
[root@ip-178-0-1-250 html]# vi index.html
```

8. After creating the file use some basic commands to check if the server is active or dead.



```
[root@ip-178-0-1-250 html]# systemctl status nginx
○ nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)
     Active: inactive (dead)
[root@ip-178-0-1-250 html]# systemctl restart nginx
[root@ip-178-0-1-250 html]# systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)
     Active: active (running) since Sun 2024-02-18 10:01:16 UTC; 3s ago
    Process: 25524 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
    Process: 25528 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
    Process: 25534 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 25546 (nginx)
      Tasks: 2 (limit: 1114)
```

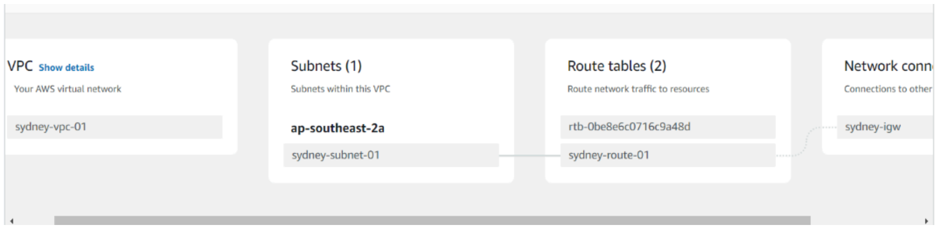
9. If the server is active then copy the public address and paste it and add default port number of nginx which is 80 by using colon : after the public ip



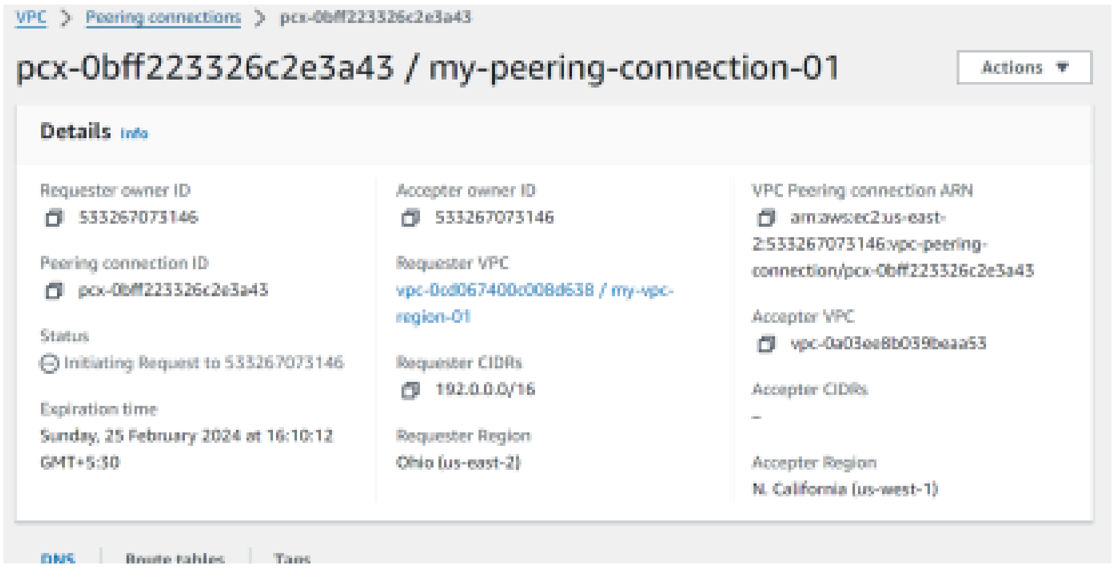
Not secure | 3.101.73.51
Import favourites
For quick access, place your favourites here on the favourites bar. Manage favourites now

Hi all this Dheeraj kumar Good Morning Thank you Bye!

10. Now create a new Vpc My-vpc-region-02 in another region Ohio with the following connections attached i.e., Subnets, routing tables, Internet gateway.



11. Goto the peering connection and create a new peering connection(my-peering-connection-01) for the two Vpc's in the different regions by passing Vpc Id.



pcx-0bff223326c2e3a43 / my-peering-connection-01

Details Info

Requester owner ID
533267073146

Peering connection ID
pcx-0bff223326c2e3a43

Status
Initiating Request to 533267073146

Expiration time
Sunday, 25 February 2024 at 16:10:12 GMT+5:30

Accepter owner ID
533267073146

Requester VPC
vpc-0cd067400c008d638 / my-vpc-region-01

Requester CIDRs
192.0.0.0/16

Requester Region
Ohio (us-east-2)

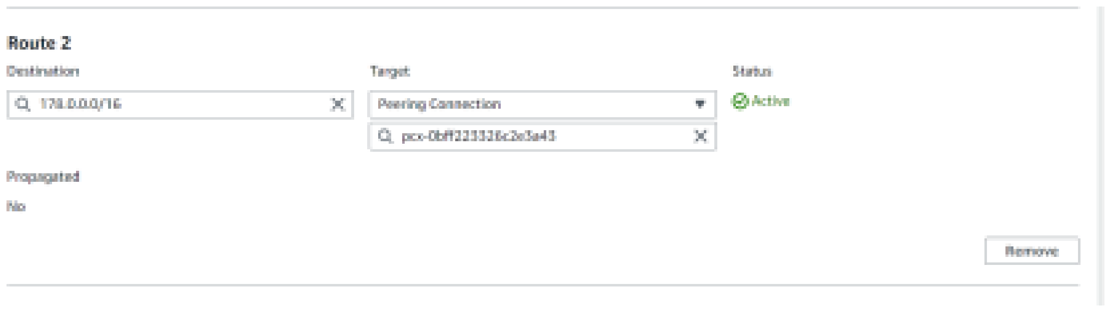
VPC Peering connection ARN
arn:aws:ec2:us-east-2:533267073146:vpc-peering-connection/pcx-0bff223326c2e3a43

Accepter VPC
vpc-0a03ee8b039beaa53

Accepter CIDRs
–

Accepter Region
N. California (us-west-1)

12. After creating the peering connection accept the peering connection for connection establishment and then goto the routetable give the destination address to allow the traffic to flow in both Vpcs.



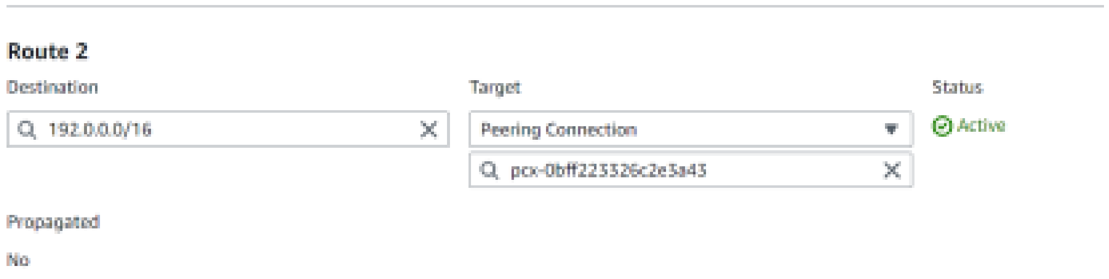
Route 2

Destination
178.0.0.0/16

Target
Peering Connection
pcx-0bff223326c2e3a43

Status
Active

Propagated
No


Route 2

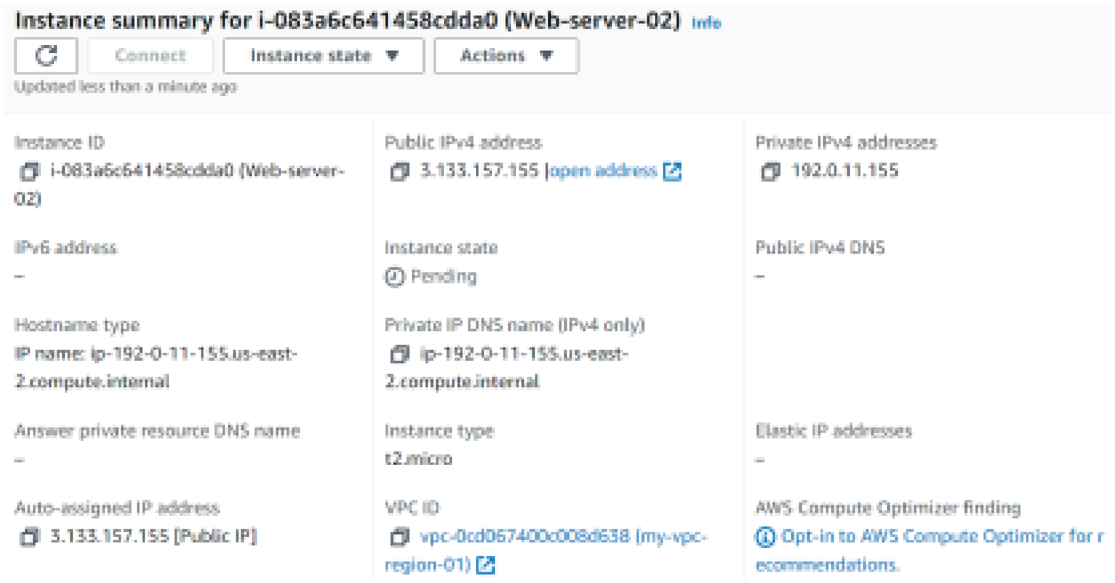
Destination: 192.0.0.0/16

Target: Peering Connection
pcx-0bff223326c2e3a43

Status: Active

Propagated: No

13. After connecting the two VPCs in two different regions by using peering goto the instance in the new region and create a new instance with the following VPC settings.


Instance summary for i-083a6c641458cdda0 (Web-server-02)

Updated less than a minute ago

Instance ID: i-083a6c641458cdda0 (Web-server-02)

Public IPv4 address: 3.133.157.155 |open address

Private IPv4 addresses: 192.0.11.155

IPv6 address: –

Instance state: Pending

Public IPv4 DNS: –

Hostname type: IP name: ip-192-0-11-155.us-east-2.compute.internal

Private IP DNS name (IPv4 only): ip-192-0-11-155.us-east-2.compute.internal

Answer private resource DNS name: –

Instance type: t2.micro

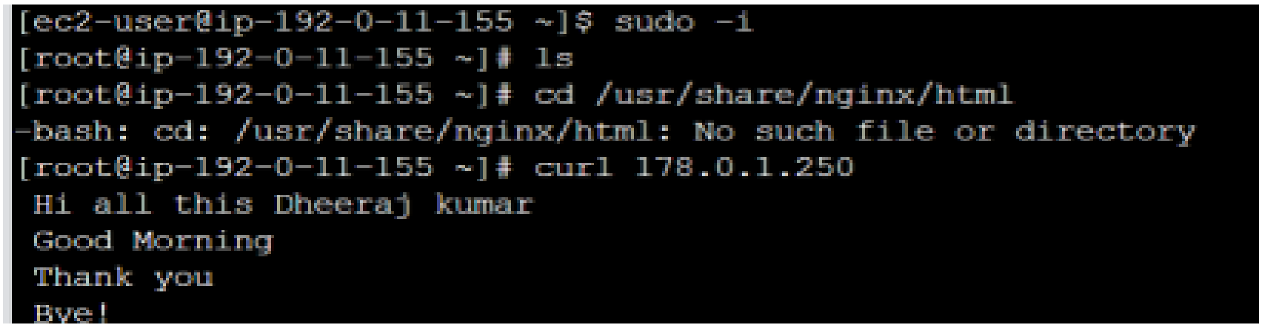
Elastic IP addresses: –

Auto-assigned IP address: 3.133.157.155 [Public IP]

VPC ID: vpc-0cd067400c008d638 (my-vpc-region-01)

AWS Compute Optimizer finding: Opt-in to AWS Compute Optimizer for recommendations.

14. Now connect to the CLI and connect to the root user for checking the peering connection establishment in the new instance which is on a different region.


```
[ec2-user@ip-192-0-11-155 ~]$ sudo -i
[root@ip-192-0-11-155 ~]# ls
[root@ip-192-0-11-155 ~]# cd /usr/share/nginx/html
-bash: cd: /usr/share/nginx/html: No such file or directory
[root@ip-192-0-11-155 ~]# curl 178.0.1.250
 Hi all this Dheeraj kumar
 Good Morning
 Thank you
 Bye!
```