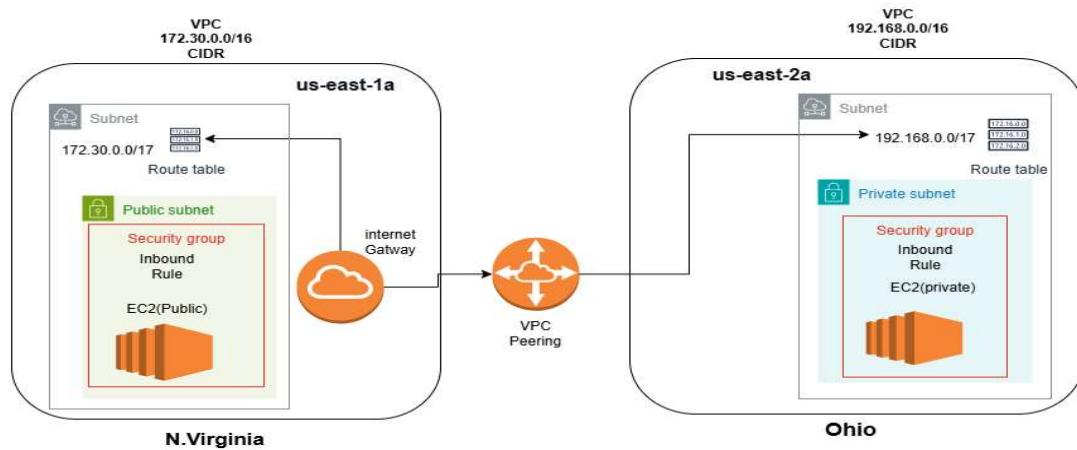


## VPC Peering :-

Connecting two VPCs with each other explaining concept of public server (ec2) internet giving to private server(ec2).



Note : VPC peering need to create in that VPC where the Internet gateway created.

First we will create a VPC in N.virginia

The screenshot shows the AWS VPC dashboard with the following details:

- Header:** AWS logo, Search bar, United States (N. Virginia), Ganraj.
- Navigation:** EC2, IAM, VPC, Billing and Cost Management, Aurora and RDS, S3.
- Left sidebar:** VPC dashboard, EC2 Global View, Virtual private cloud (Your VPCs, Subnets, Route tables, Internet gateways, Egress-only internet gateways, Carrier gateways, DHCP option sets, Elastic IPs, Managed prefix lists, NAT gateways).
- Main Content:**
  - A green banner at the top says "You successfully deleted vpc-0898aba00d301c744 / VPC\_NV".
  - Your VPCs (1)**: A table showing one VPC:

Name	VPC ID	State	Block Public Access	IPv4 CIDR	IPv6 CIDR
vpc-0dbbb05820b87165c	vpc-0dbbb05820b87165c	Available	Off	172.31.0.0/16	-
  - vpc-0898aba00d301c744 / VPC\_NV**: A detailed view of the VPC:
    - Details tab: VPC ID (vpc-0898aba00d301c744), State (Available), Block Public Access (Off), DNS hostnames (Disabled).
    - Resource map, CIDs, Flow logs, Tags, Integrations tabs.

**Create VPC** [Info](#)

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

VPC only ✓

VPC and more

**Name tag - optional**  
Creates a tag with a key of 'Name' and a value that you specify.

VPC\_NV

**IPv4 CIDR block** [Info](#)  
 IPv4 CIDR manual input  
 IPAM-allocated IPv4 CIDR block

**IPv4 CIDR**  
172.30.0.0/16

IPv4 block size must be between 16 and 30

**VPC dashboard** <

**Your VPCs (2)** [Info](#)

Last updated less than a minute ago ⟳ [Actions](#) Create VPC

Find VPCs by attribute or tag					
<input type="checkbox"/> Name <span style="float: right;">▼</span>	VPC ID <span style="float: right;">▼</span>	State <span style="float: right;">▼</span>	Block Public... <span style="float: right;">▼</span>	IPv4 CIDR <span style="float: right;">▼</span>	IPv6 CIDR <span style="float: right;">▼</span>
<input type="checkbox"/> -	vpc-0dbbb05820b87165c	<span style="color: green;">Available</span>	<input type="checkbox"/> Off	172.31.0.0/16	-
<input type="checkbox"/> VPC_NV	vpc-090b4aadbfcfd4bafe	<span style="color: green;">Available</span>	<input type="checkbox"/> Off	172.30.0.0/16	-

< 1 > ⟳

**Virtual private cloud**

- Your VPCs**
- Subnets
- Route tables

Creating Subnet in same VPC.

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

**VPC**  
VPC ID  
Create subnets in this VPC.  
vpc-090b4aadbfcfd4bafe (VPC\_NV)

**Associated VPC CIDRs**

**IPv4 CIDRs**  
172.30.0.0/16

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

Subnet 1 of 1

aws | Search [Alt+S] | United States (N. Virginia) | Ganraj

EC2 IAM VPC Billing and Cost Manage... Aurora and RDS S3

VPC > Subnets > Create subnet

**Subnet 1 of 1**

**Subnet name**  
Create a tag with a key of 'Name' and a value that you specify.  
**Subnet\_NV**  
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.  
**United States (N. Virginia) / us-east-1a**

**IPv4 VPC CIDR block** Info  
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.  
**172.30.0.0/16**

**IPv4 subnet CIDR block**  
**172.30.0.0/17** 32,768 IPs

## Creating Internet Gateway with Same VPC.

aws | Search [Alt+S] | United States (N. Virginia) | Ganraj

EC2 IAM VPC Billing and Cost Manage... Aurora and RDS S3

VPC > Internet gateways

**VPC dashboard**

**Virtual private cloud**

- Your VPCs
- Subnets
- Route tables
- Internet gateways**
- Egress-only internet gateways
- Carrier gateways
- DHCP option sets
- Elastic IPs
- Managed prefix lists

**Internet gateways (2) Info**

Name	Internet gateway ID	State	VPC ID
-	igw-014b7c1da638c3747	Attached	vpc-0dbbb05820b87165c
-	igw-07bc94b08053c58d9	Detached	-

**Create internet gateway**

aws | Search [Alt+S] | United States (N. Virginia) | Ganraj

EC2 IAM VPC Billing and Cost Manage... Aurora and RDS S3

VPC > Internet gateways > Create internet gateway

**Internet gateway successfully deleted - igw-0de89d3a3071ec824**

**Internet gateway settings**

**Name tag**  
Creates a tag with a key of 'Name' and a value that you specify.  
**NV\_IGW**

**Tags - optional**  
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**

Q Name	X	Q NV_IGW	X	Remove
--------	---	----------	---	--------

**Add new tag**  
You can add 49 more tags.

**Create internet gateway**

Need to attach to VPC that we have created one.

VPC dashboard < Details Info

Internet gateway ID: igw-0ba601fa5b34dcff | State: Detached | VPC ID: - | Owner: 35135295

Tags:

Key	Value
Name	NV_IGW

Actions ▾

- Attach to VPC
- Detach from VPC
- Manage tags
- Delete

Attach to VPC (igw-0ba601fa5b34dcff) Info

VPC

Attach an internet gateway to a VPC to enable the VPC to communicate with the internet. Specify the VPC to attach below.

Available VPCs

Attach the internet gateway to this VPC.

vpc-0add352ddb43cfb96

Use: "vpc-0add352ddb43cfb96"

Cancel **Attach internet gateway**

Need to connect route table to internet gateway is very important.

VPC dashboard < Details Info

vpc-0add352ddb43cfb96

DNS resolution: Enabled | Main network ACL: adl-0760be325026fe833 | IPv6 CIDR (Network border group): -

Resource map | CDRs | Flow logs | Tags | Integrations

Resource map Info

VPC Show details Your AWS virtual network

Subnets (1) Subnets within this VPC

us-east-1a Subnet\_NV

Route tables (1) Route network traffic to resources

rtb-012cd9974c7b95e57

Network connections (1) Connections to other networks

NV\_IGW

**VPC dashboard**

**vpc-0add352ddb43cfb96**

<b>DNS resolution</b> Enabled	<b>Tenancy</b> default	<b>Off</b>	<b>Disabled</b>
<b>Main network ACL</b> acl-0760be325026fe833	<b>Default VPC</b> No	<b>DHCP option set</b> dept-058c9890e15e8aa56	<b>Main route table</b> rtb-012cd9974c7b95e57
<b>IPv6 CIDR (Network border group)</b> -	<b>Network Address Usage metrics</b> Disabled	<b>IPv4 CIDR</b> 172.30.0.0/16	<b>IPv6 pool</b> -
		<b>Route 53 Resolver DNS Firewall rule groups</b> -	<b>Owner ID</b> 351352951218

**Resource map** | CIDs | Flow logs | Tags | Integrations

**Resource map** Info

**VPC** Show details  
Your AWS virtual network  
VPC\_NV

**Subnets (1)**  
Subnets within this VPC  
us-east-1a  
Subnet\_NV

**Route tables (1)**  
Route tables within this VPC  
rtb-012cd9974c7b95e57  
1 subnet association  
1 route including local

**Network connections (1)**  
Connections to other networks  
NV\_IGW

**VPC dashboard**

**rtb-012cd9974c7b95e57**

**Details** Info

<b>Route table ID</b> rtb-012cd9974c7b95e57	<b>Main</b> Yes	<b>Explicit subnet associations</b> -	<b>Edge associations</b> -
<b>VPC</b> vpc-0add352ddb43cfb96   VPC_NV	<b>Owner ID</b> 351352951218		

**Routes** | Subnet associations | Edge associations | Route propagation | Tags

**Routes (1)**

Destination	Target	Status	Propagated
172.30.0.0/16	local	Active	No

**VPC** > **Route tables** > **rtb-012cd9974c7b95e57** > Edit routes

**Edit routes**

<b>Destination</b> 172.30.0.0/16	<b>Target</b> local	<b>Status</b> Active	<b>Propagated</b> No
	<input type="text"/> local	<input type="button" value="X"/>	

**Add route**

**Cancel** **Preview** **Save changes**

Screenshot of the AWS VPC Edit routes interface. A route is being added to a route table with a destination of 172.30.0.0/16, target set to Internet Gateway, and status set to Active. The 'Save changes' button is highlighted with a red box.

Screenshot of the AWS VPC Resource map interface. It shows the connection between a VPC (VPC\_NV), Subnets (us-east-1a), Route tables (rtb-012cd9974c7b95e57), and Network connections (NV\_IGW).

See we have made connection between internet gateway route tables and subnets.

Now we will create VPC Ohio Region .

Screenshot of the AWS VPC Create VPC interface. A new VPC is being created with the name 'VPC\_Ohio'. The IPv4 CIDR block is set to 192.168.0.0/16.

aws Search [Alt+S] United States (Ohio) Ganraj

EC2 IAM VPC Billing and Cost Manage... Aurora and RDS S3

VPC > Subnets > Create subnet

Create a new subnet with the settings you specify:

**Subnet\_ohio**

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.  
**United States (Ohio) / us-east-2a**

**IPv4 VPC CIDR block Info**  
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.  
**192.168.0.0/16**

**IPv4 subnet CIDR block**  
**192.168.0.0/17** 32,768 IPs

**Tags - optional**

**Key** Name **Value - optional** Subnet\_ohio **Remove**

Add new tag

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

aws Search [Alt+S] United States (Ohio) Ganraj

EC2 IAM VPC Billing and Cost Manage... Aurora and RDS S3

VPC > Subnets > Create subnet

**VPC ID**  
Create subnets in this VPC.  
**vpc-07abf75b72739b67e (VPC\_Ohio)**

**Associated VPC CIDRs**

**IPv4 CIDRs**  
192.168.0.0/16

**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\_ohio**

The name can be up to 256 characters long.

aws Search [Alt+S] United States (Ohio) Ganraj

EC2 IAM VPC Billing and Cost Manage... Aurora and RDS S3

VPC > Subnets > Create subnet

Create a new subnet with the settings you specify:

**Subnet\_ohio**

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.  
**United States (Ohio) / us-east-2a**

**IPv4 VPC CIDR block Info**  
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.  
**192.168.0.0/16**

**IPv4 subnet CIDR block**  
**192.168.0.0/17** 32,768 IPs

**Tags - optional**

**Key** Name **Value - optional** Subnet\_ohio **Remove**

Add new tag

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

We have created subnet in **ohio vpc** but need to check that it not auto assign **IP(public)** because we creating a private network or the private subnet which dosen't have the public access IP

The screenshot shows the AWS VPC Subnets page. On the left, there's a sidebar with 'Virtual private cloud' options like Your VPCs, Subnets, Route tables, etc. The main area shows a subnet named 'subnet-02c211cb51340b7e6 / Subnet\_ohio'. The 'Details' section includes fields for Subnet ID, Subnet ARN, State (Available), and Block Public Access (Off). The 'Auto-assign public IPv4 address' field is explicitly highlighted with a red box.

Now we will create **VPC Peering** :-

VPC peering created only where the internet gateway create VPC (N.virginia)

Note : We also connect VPC Peering CrossAccount also

The screenshot shows the AWS VPC Peering connections page. The sidebar has 'Virtual private cloud' options. The main area shows a table for 'Peering connections' with columns for Name, Peering connection ID, Status, Requester VPC, and Acceptor VPC. A large orange 'Create peering connection' button is highlighted with a red box at the top right of the table area.

Screenshot of the AWS VPC Peering Connections creation page. The 'Name - optional' field contains 'NV\_Peering'. The 'Select a local VPC to peer with' dropdown shows 'vpc-0add352ddb43cfb96 (VPC\_NV)'. The 'VPC CIDRs for vpc-0add352ddb43cfb96 (VPC\_NV)' table lists one entry: CIDR 172.30.0.0/16, Status Associated.

First go to Ohio Region to take VPC ID from VPC\_ohio and paste it over VPC ID Acceptor

And create peering.

Screenshot of the AWS VPC dashboard for the Ohio Region. The VPC 'vpc-07abf75b72739b67e / VPC\_Ohio' is selected. A red box highlights the 'Copied' status next to the VPC ID 'vpc-07abf75b72739b67e' in the list.

Screenshot of the AWS VPC Peering Connections creation page. The 'Region' section shows 'United States (Ohio) (us-east-2)' selected. The 'VPC ID (Acceptor)' dropdown shows 'vpc-07abf75b72739b67e'. The 'Tags' section contains a single tag 'NV\_Peering'. The 'Create peering connection' button is at the bottom right.

AWS Search [Alt+S] United States (N. Virginia) Ganraj

EC2 IAM VPC Billing and Cost Manage... Aurora and RDS S3

VPC Peering connections ppx-035c75809b23269f8

**VPC dashboard**

EC2 Global View Filter by VPC

**Virtual private cloud**

- Your VPCs
- Subnets
- Route tables
- Internet gateways
- Egress-only internet gateways
- Carrier gateways
- DHCP option sets
- Elastic IPs
- Managed prefix lists
- NAT gateways
- Peering connections**
- Route servers New

**pcc-035c75809b23269f8 / NV\_Peering**

**Details Info**

Requester owner ID	351352951218	Acceptor owner ID	351352951218
Peering connection ID	pcc-035c75809b23269f8	Requester VPC	vpc-0add352ddb43cfb96 / VPC_NV
Status	Initiating Request to 351352951218	Requester CIDRs	172.30.0.0/16
Expiration time	Friday, July 11, 2025 at 19:39:27 GMT+5:30	Requester Region	N. Virginia (us-east-1)
		VPC Peering connection ARN	arn:aws:ec2:us-east-1:351352951218:vpc-peering-connection/pcc-035c75809b23269f8
		Acceptor VPC	vpc-07abf75b72739b67e
		Acceptor CIDRs	-
		Acceptor Region	Ohio (us-east-2)

**DNS** Route tables Tags

**DNS settings** Edit DNS settings

VPC Peering is created and N.Virginia VPC requested to Ohio VPC for peering Connection We need to accept that request from ohio region VPC Peering.

AWS Search [Alt+S] United States (Ohio) Ganraj

EC2 IAM VPC Billing and Cost Manage... Aurora and RDS S3

VPC Peering connections

**Virtual private cloud**

- Your VPCs
- Subnets
- Route tables
- Internet gateways
- Egress-only internet gateways
- DHCP option sets
- Elastic IPs
- Managed prefix lists
- NAT gateways
- Peering connections**
- Route servers New

**Peering connections (1/1) Info**

Name	Peering connection ID	Status	Requester VPC	Acceptor VPC
-	pcc-035c75809b23269f8	Pending acceptance	vpc-0add352ddb43cfb96	vpc-07abf75b72739b67e

**pcc-035c75809b23269f8**

**Pending acceptance**

You can accept or reject this peering connection request using the 'Actions' menu. You have until Friday, July 11, 2025 at 19:39:27 GMT+5:30 to accept or reject the request, otherwise it expires.

**Details DNS Route tables Tags**

AWS Search [Alt+S] United States (Ohio) Ganraj

EC2 IAM VPC Billing and Cost Manage... Aurora and RDS S3

VPC Peering connections

**Virtual private cloud**

- Your VPCs
- Subnets
- Route tables
- Internet gateways
- Egress-only internet gateways
- DHCP option sets
- Elastic IPs
- Managed prefix lists
- NAT gateways
- Peering connections**
- Route servers New

**Peering connections (1/1) Info**

Name	Peering connection ID	Status	Requester VPC	Acceptor VPC
-	pcc-035c75809b23269f8	Pending acceptance	vpc-0add352ddb43cfb96	vpc-07abf75b72739b67e

**pcc-035c75809b23269f8**

**Pending acceptance**

You can accept or reject this peering connection request using the 'Actions' menu. You have until Friday, July 11, 2025 at 19:39:27 GMT+5:30 to accept or reject the request, otherwise it expires.

**Actions** Create peering connection

- View details
- Accept request
- Reject request
- Edit DNS settings
- Manage tags
- Delete peering connection

Your VPC peering connection (pxc-035c75809b23269f8) has been established.

To send and receive traffic across this VPC peering connection, you must add a route to the peered VPC in one or more of your VPC route tables.

**Peering connections (1) Info**

Requester CIDRs	Acceptor CIDRs	Requester owner ID	Acceptor owner ID	Requester Region	Acceptor Region
172.30.0.0/16	192.168.0.0/16	351352951218	351352951218	N. Virginia (us-east-1)	Ohio (us-east-2)

Your VPC peering connection (pxc-035c75809b23269f8) has been established.

To send and receive traffic across this VPC peering connection, you must add a route to the peered VPC in one or more of your VPC route tables.

**Peering connections (1) Info**

Name	Peering connection ID	Status	Requester VPC	Acceptor VPC
-	pxc-035c75809b23269f8	Active	vpc-0add352ddb43cfb96	vpc-07abf

**Note : we only create one VPC Peering always because its process of requester and acceptor connecting.**

Now we have route tables need to connect each other

First we will go to Ohio private subnet route table.

**Resource map**

VPC	Subnets	Route tables	Network connections
VPC Show details	Subnets (1)	Route tables (1)	Network connections (0)
Subnet_VPC_Ohio	us-east-2a	rtb-08fe9687e9931917c	

**Route tables (1) Info**

Route table	Description	Associations
rtb-08fe9687e9931917c	Route network traffic to resources	1 subnet association 1 route including local

The screenshot shows the AWS VPC Route Tables page. The route table ID is rtb-08fe8687e9931917c. It has one route with destination 192.168.0.0/16 and target local, status Active, and propagated No.

If need secure connection so putting 0.0.0.0/0 you can use direct CIDR of N.Virginia VPC  
172.30.0.0/16

The 'Edit routes' dialog shows a new route being added. Destination: 192.168.0.0/16, Target: local, Status: Active, Propagated: No. A new route is being added with Destination: 172.30.0.0/16, Target: Peering Connection, Status: - (Pending), Propagated: No. The 'Save changes' button is highlighted.

Here we have added route table peering with N.virginia

The screenshot shows the AWS VPC Route Tables page again. The route table now has two routes: one to 172.30.0.0/16 via Peering Connection and one to 192.168.0.0/16 via local.

Now we will do same in N.virginia like cross connection.

The screenshot shows the AWS VPC Route Tables page. The route table ID is rtb-012cd9974c7b95e57. It has one main entry (0.0.0.0/0) pointing to the Internet Gateway (igw-0ba601fa5b34dcff). There are no explicit subnet associations or edge associations.

Destination	Target	Status	Propagated
0.0.0.0/0	igw-0ba601fa5b34dcff	Active	No
172.30.0.0/16	local	Active	No

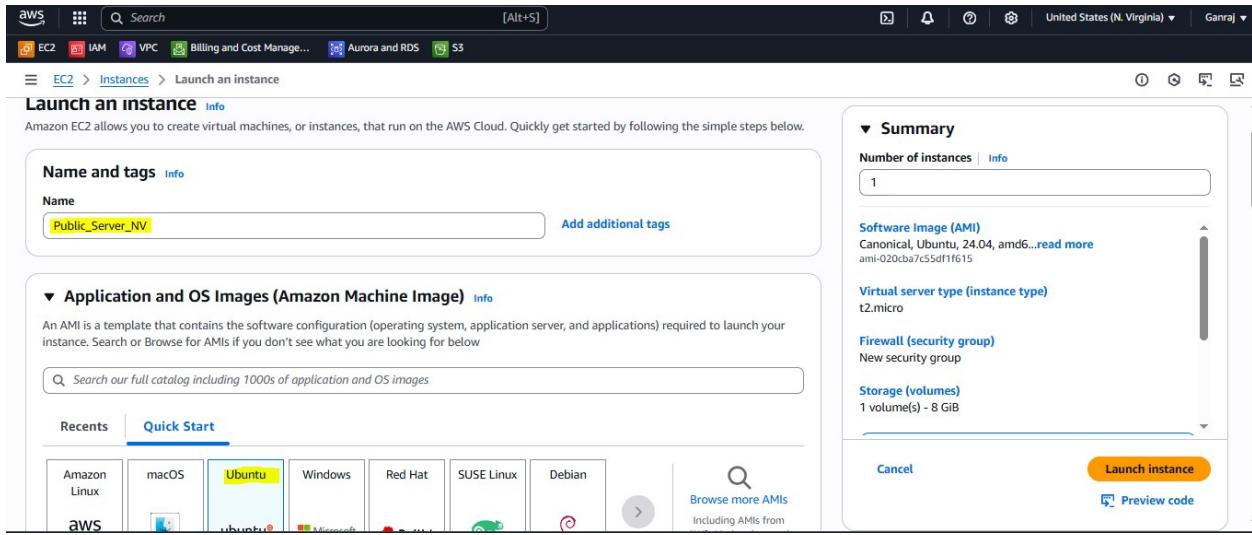
The screenshot shows the 'Edit routes' dialog box for route table rtb-012cd9974c7b95e57. A new route is being added for destination 172.30.0.0/16, targeting a Peering Connection (pcx-035c75809b23269f8). The 'Save changes' button is highlighted with a red box.

Destination	Target	Status	Propagated
172.30.0.0/16	local	Active	No
0.0.0.0/0	Internet Gateway	Active	No
192.168.0.0/16	Peering Connection	-	No
	pcx-035c75809b23269f8		

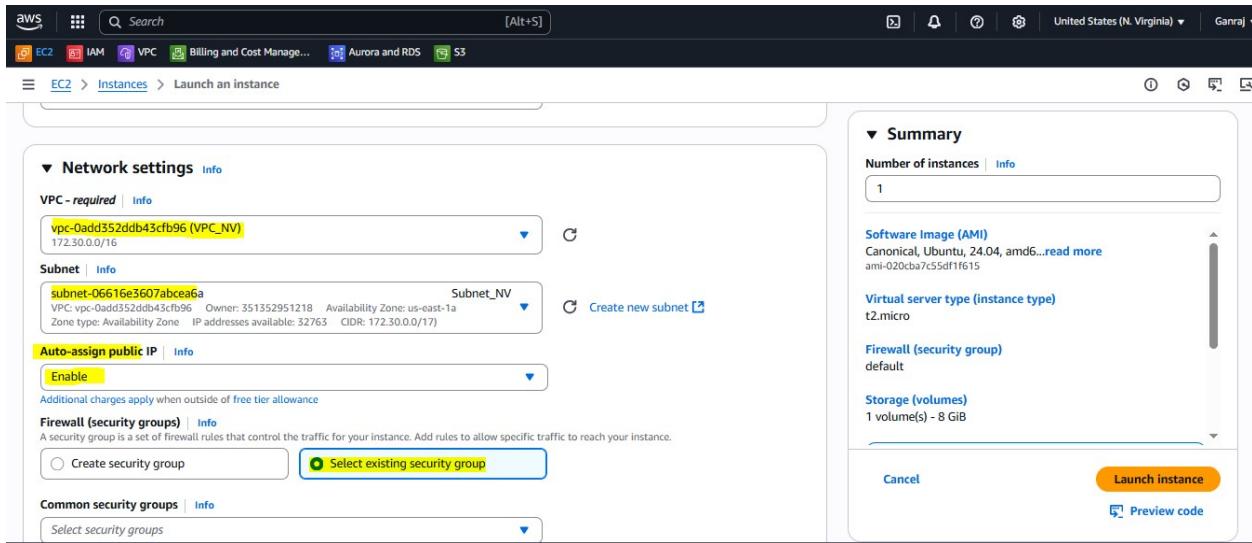
The screenshot shows the AWS VPC Route Tables page after saving the changes. The route table now has three entries: 0.0.0.0/0 pointing to the Internet Gateway, 172.30.0.0/16 pointing to local, and 192.168.0.0/16 pointing to the Peering Connection (pcx-035c75809b23269f8).

Destination	Target	Status	Propagated
0.0.0.0/0	igw-0ba601fa5b34dcff	Active	No
172.30.0.0/16	local	Active	No
192.168.0.0/16	pcx-035c75809b23269f8	Active	No

## CONNECTION MADE BETWEEN TWO VPC USING PEERING CONNECTION NOW WE HAVE CREATE EC2'S FOR EACH VPC ONE WILL BE PUBLIC ANOTHER.



Here we have to select created VPC (**VPC\_NV**) while creating Ec2 Instance



And then we Launch the instance and don't forget to changes in security group because here we are selected default and in the that security group not enabled the ports like SSH And HTTP or all TCP which provides connection Internet and access.

We are making changes in inbound Rule beware while changing it

The screenshot shows the AWS EC2 Security Groups console. The current view is "Edit inbound rules" for a security group named "sg-0da1d5a552eb1ea02 - default". There are four rules listed:

- Rule 1: Type "All traffic", Protocol "All", Port range "All", Source "Custom" (set to "sg-0da1d5a552eb1ea02").
- Rule 2: Type "All TCP", Protocol "TCP", Port range "0 - 65535", Source "Any...".
- Rule 3: Type "SSH", Protocol "TCP", Port range "22", Source "Any...".
- Rule 4: Type "All traffic", Protocol "All", Port range "All", Source "Any...".

A warning message at the bottom states: "⚠ 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." Below the message are "Cancel", "Preview changes", and "Save rules" buttons.

We need to create one EC2 in Ohio which is private server.

Here also we have select vpc that we have created. Need to changes in security groups also.

And see here we are not enabled the auto sign IP so this server will **remain Private**

The screenshot shows the AWS EC2 Instances console. The instance summary for "i-0a6b6782596eb5151 (Private\_Server\_Ohio)" is displayed. Key details include:

- Instance ID:** i-0a6b6782596eb5151
- Public IPv4 address:** [REDACTED]
- Private IPv4 address:** 192.168.100.9
- Instance state:** Running
- Private IP DNS name (IPv4 only):** ip-192-168-100-9.us-east-2.compute.internal
- Instance type:** t2.micro
- VPC ID:** vpc-07abf75b72739b67e (VPC\_Ohio)
- Auto Scaling Group name:** [REDACTED]

The left sidebar shows navigation links for EC2, Dashboard, EC2 Global View, Events, Instances (with sub-links for Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations), and Images (AMIs).

here you can see this server has no Public IP.

This will only get internet from public server

**IMPORTANT-NOTE:** Check security groups of both EC2 . AWS sometime not properly so always

Check that SSH, All traffic , ALL TCP and **if need ICMP**.

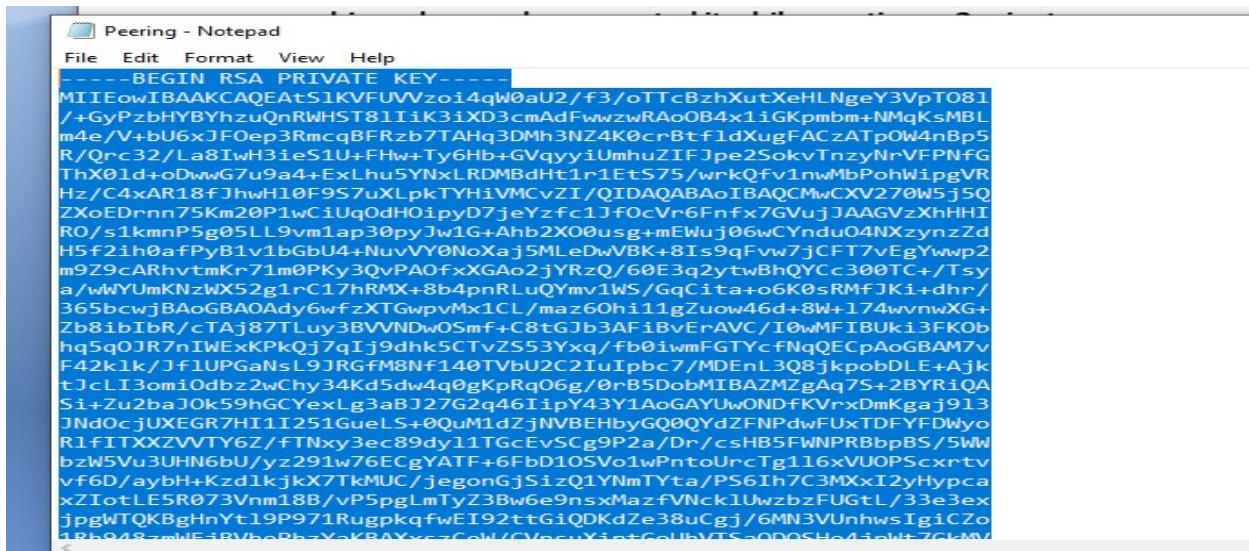
As you can see in the screenshot connection between Public Ec2 to Private Ec2 made.

```
root@ip-172-30-110-129:~# ping 192.168.100.9
PING 192.168.100.9 (192.168.100.9) 56(84) bytes of data.
64 bytes from 192.168.100.9: icmp_seq=1 ttl=64 time=11.5 ms
64 bytes from 192.168.100.9: icmp_seq=2 ttl=64 time=11.5 ms
64 bytes from 192.168.100.9: icmp_seq=3 ttl=64 time=11.5 ms
64 bytes from 192.168.100.9: icmp_seq=4 ttl=64 time=11.5 ms
```

Now we will connect Private Sever (ec2) using Public Server but remember we need ssh key for that process and that ssh is store in the server.

So in that public ec2 server we first need to add ssh key which we have download on our local machine when we have created it while creating ec2 private server

First we open that key using notepad and copy it creating a pem file using vim editor and paste that ket content it.



Using **vim peering.pem** command opening the text editor and paste the content

```

-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAetSlKVfUVVzoia4gW0aU2/f3/oTTcBzhXutXeHLNgEY3VpT081
/+GyPzbHYBYhzuznRWHST8lIiK3iXD3cmAdFwzvRa0B4x1iGKpmnbNMQkMsB5
m4e/V+bU6x/JF0ep3RmcgBFxz7TAHQ3Mh3N24K0crBtfldxugFACz2AtpW4nBp5
R/Qrc32/LaIW31eS1U+FHw+TyHh+GvqyyUmhuZ1BUppe25okvInzyNr-VPNN6
ThX0ldt0wwwG7u9e4+Exihu5NxLRUMBdHtlr1EtS75/vkOfvlnwMbPohWpigVR
Hz/C4xAR18fJhwH10F9S7uLpkTYHiVMCv2I/QIDNQABaoIBAQCMwCXV270W5j5Q
ZXoEDrn75km20PlwC1UgOdHoipyDjeyzfcIJFoc-vrEfmx/GWujJAAGVzXhHH
RO/s1kmnP5g05L9vn1ap30pyJw1G+Ab2X0Usug+mEWu106wCYnduo4Nxzynzzd
H5f2izh0afPyB1vlGbU4+NuvYYONoXa5M1iebwBx+8ls9gFw7jCFT7vegXwp2
m929cARhvtmKz71m0PKy3QvPAOfxXGao21yRzQ/60E3z2ytwhqYCc300TC-/Tsy
a/wWYUmKnzWX52glrC17hRMX+8b4pnRluQMy1WS/GqCiata+6k0srMFJKi+dhr/
365bcwjBaoGBAoAdyGwfzXTGwpvMx1CL/mazG0h11g2uow4d+8w+174vwvwXG+
Zb81b1bR-cTAj87Tlu3BVVNDwOSmf+C8tGJh3Af1BvErAVC/I0wMFIBUi3FKob
hq5gQJR/nIWEzxPkQj7q1j9dnh5CrzSS3x/q/b01wmFGYcfnqQCPaGhBAM7v
F42K1k/jf1UPGaNsL9JRGFM8Nf140TVUZC2iuIpbct/MDENi3Q8jkpodULt+Ajk
tjciLi3miOdbz2wChy34Kd5dw4q0gRpQo6g/0xB5dohMTBAZM2gq75+ZYRiQA
S1+zu2baJok59hGCYexLg3aBu2762g461ipY43Y1AoGAYUwONDEKVrxNmRgaJ913
JNd0cJUXEGR7H11251GueLS+0QuMldzjNVBEHbyGO00Yd2fNPdxFUXtDEYFDWyo
R1ftTXZvVvTY6z/TNxv3ec89dyll17GcFvsCg9P2a/Dx/cshB5FWNPRBbpBS/5WW
bzW5vu3UHNGbU/yz291w76BcgyATF+6Pbd10SVo1wPntoUrcfg116xVUOBScxrtv
vf6D/ayBh+rzdlkjxK7TkMDC/jegonGjSizQ1YNmTYta/PS6iH7C3MxxI2yHypca
xz1otLE5R073Vm188/vp5pqlmryz3Bw6e9nsxMazfVNckl0wzbzrUgtL/33e3ex
JpgWTQkBghNy19P971RugpkqfwEI92ttG1QDK2e38uCgj/6MN3VUhnws1g1C2o

```

1,29

i-053e039deeb325ebc (Public\_Server\_NV)  
Public IPs: 44.214.142.248 Private IPs: 172.30.110.129

And save it.

Now we will give permission to key run and access

**chmod 400 peering.pem**

Now will ssh command to access the private server in public server

example :- **ssh -i private-key.pem ec2-user@<Private-EC2-Private-IP>**

**ssh -i peering.pem ubuntu@192.168.100.9**

Part	Meaning
<b>ssh</b>	Secure Shell command—used to log in remotely to a Linux server.
<b>-i peering.pem</b>	<b>-i</b> means “identity file.” This tells SSH to use your <b>private key file named peering.pem</b> for authentication.
<b>Ubuntu@192.168.100.9</b>	This is the <b>login user and target host</b> : - Ubuntu: The <b>username</b> you are trying to log in as. - 192.168.100.9: The <b>private IP address</b> of your EC2 instance inside the Ohio VPC.

OS	Username
Amazon Linux 2	ec2-user
Ubuntu	ubuntu
RHEL	ec2-user
CentOS	centos

Always remember while using names of username its case sensitive so type carefully

```

aws [Alt+S] United States (N. Virginia) Ganraj
EC2 IAM VPC Billing and Cost Manage... Aurora and RDS S3

root@ip-172-30-110-129:~# ssh -i peering.pem ubuntu@192.168.100.9
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-1029-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of Fri Jul  4 15:45:12 UTC 2025

System load: 0.0      Processes:          103
Usage of /: 25.6% of 6.71GB   Users logged in: 0
Memory usage: 20%      IPv4 address for enX0: 192.168.100.9
Swap usage: 0%

Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

i-053e039deeb325ebc (Public_Server_NV)
PublicIPs: 44.214.142.248 PrivateIPs: 172.30.110.129

```

Using that command we have connected to private server(192.168.100.9) as you see in screenshot

```

aws [Alt+S] United States (N. Virginia) Ganraj
EC2 IAM VPC Billing and Cost Manage... Aurora and RDS S3

root@ip-192-168-100-9:~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eno0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc fq_codel state UP group default qlen 1000
    link/ether 02:ff:03:ff:e9:el brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.9/17 metric 100 brd 192.168.127.255 scope global dynamic eno0
        valid_lft 3466sec preferred_lft 3466sec
    inet6 fe80::ff3fe:fe3ff:fe9e1/64 scope link
        valid_lft forever preferred_lft forever
root@ip-192-168-100-9:~#

```