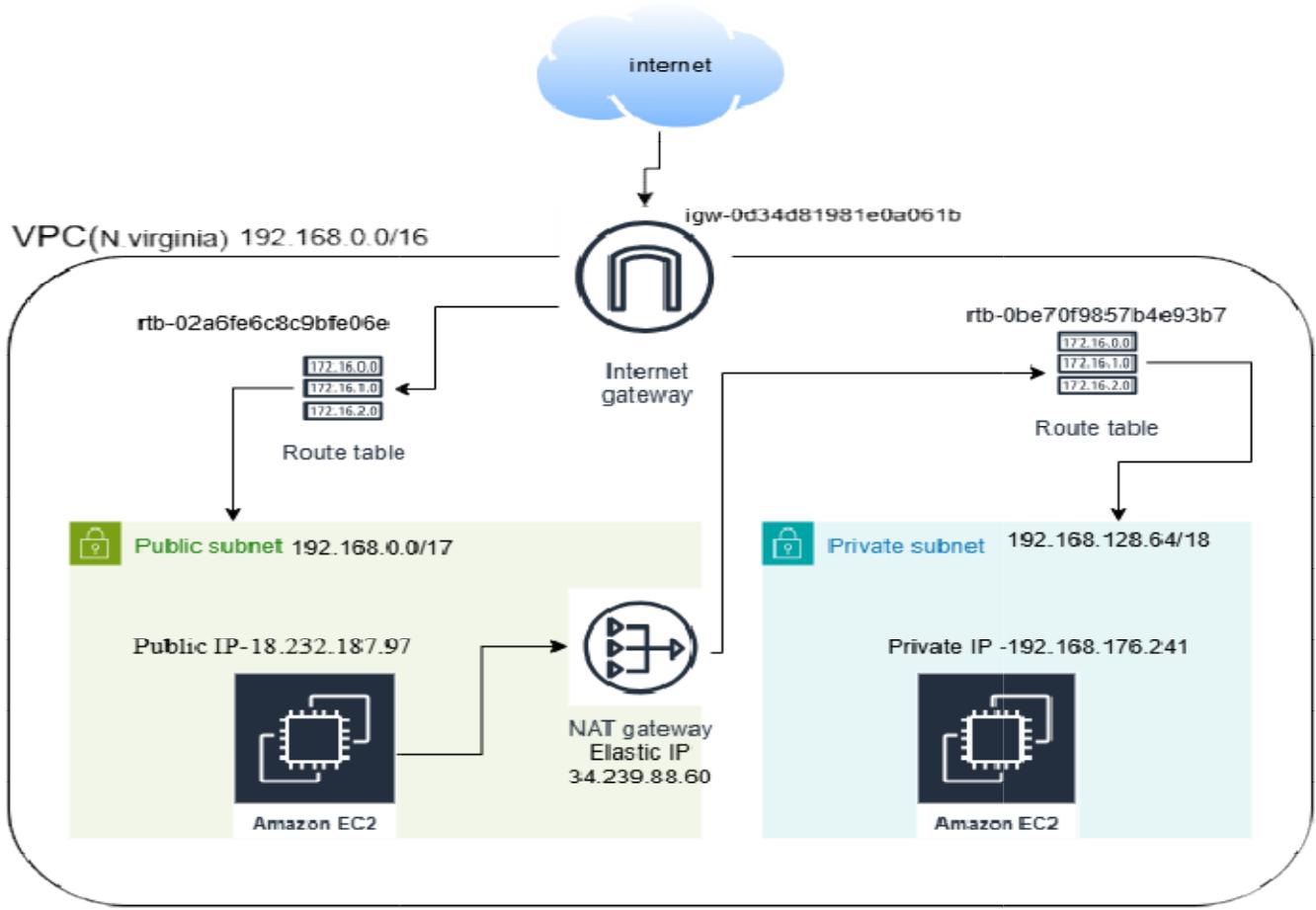


Connecting Public Server's(EC2) **Internet** to Private Server(EC2) Using NAT Gateway (VPC) security process endpoint will be public server



Issue : - For security reasons we have to not show our private server to everyone for that we will provide internet through public server.

First we will create a VPC (CIDR-192.168.0.0/16)

The screenshot shows the 'Create VPC' configuration page. Under 'VPC settings', the 'Resources to create' dropdown is set to 'VPC only'. The 'Name tag - optional' field contains 'NAT-VPN'. The 'IPv4 CIDR block' is set to '192.168.0.0/16'. The 'IPv6 CIDR block' section is set to 'No IPv6 CIDR block'.

Now in that VPC we need create Subnets (**Assigning IPs for subnet always check IPs Range in CIDRs /16**)

The screenshot shows the 'Create subnet' page in the AWS VPC console. At the top, it says 'VPC ID' and lists 'vpc-039630cd94add2198 (NAT-VPC)'. Under 'Associated VPC CIDRs', it shows 'IPV4 CIDRs' with '192.168.0.0/16'. In the 'Subnet settings' section, 'Subnet 1 of 1' is selected. The 'Subnet name' field contains 'Public_NAT'. The 'Availability Zone' dropdown is set to 'No preference'. The 'IPv4 VPC CIDR block' dropdown is set to '192.168.0.0/16'. The 'IPv4 subnet CIDR block' dropdown is set to '192.168.0.0/17'.

Our Public subnet will be - 192.168.0.0/17

This screenshot shows the same 'Create subnet' page with more detailed configurations. The 'Subnet name' is now 'Public_NAT'. The 'Availability Zone' dropdown is set to 'No preference'. The 'IPv4 VPC CIDR block' dropdown is set to '192.168.0.0/16'. The 'IPv4 subnet CIDR block' dropdown is set to '192.168.0.0/17'. A red arrow points to the '192.168.0.0/17' entry.

Private Subnet - 192.168.128.64/18

This screenshot shows the 'Create subnet' page for a private subnet. The 'Subnet name' is 'Private_NAT'. The 'Availability Zone' dropdown is highlighted with a red box and has 'No preference' selected. The 'IPv4 VPC CIDR block' dropdown is set to '192.168.0.0/16'. The 'IPv4 subnet CIDR block' dropdown is set to '192.168.128.64/18'. A red arrow points to the '192.168.128.64/18' entry.

Here in the both you can set same availability zone also

Imp – if we need set a private subnet that's we need set that subnet auto assign public IP to be Disabled or off let me show you

For public Subnet we will set On because we need that subnet accessible for internet

The screenshot shows the AWS VPC Subnet Details page. The subnet ID is subnet-0ad60e2f8e81d060d. Key details include:

- Subnet ID:** subnet-0ad60e2f8e81d060d
- IPv4 CIDR:** 192.168.128.0/18
- Availability Zone:** us-east-1f
- Route table:** rtb-02a6fe6c8c9bfe06e
- Auto-assign IPv6 address:** No
- IPv4 CIDR reservations:** -
- Resource name DNS A record:** Disabled
- Subnet ARN:** arn:aws:ec2:us-east-1:351352951218:subnet/subnet-0ad60e2f8e81d060d
- State:** Available
- IPv6 CIDR:** -
- Availability Zone ID:** us-east-1az5
- Network ACL:** acl-0712045810eac1874
- Auto-assign customer-owned IPv4 address:** No
- IPv6 CIDR reservations:** -
- Resource name DNS AAAA record:** Disabled
- Default subnet:** No
- Customer-owned IPv4 pool:** -
- IPv6-only:** No
- DNS64:** Disabled

The 'Edit subnet settings' option in the Actions menu is highlighted.

The screenshot shows the 'Edit subnet settings' page. The subnet ID is subnet-0dd6b0d4930354934. The 'Name' field is set to Public_NAT. The 'Auto-assign IP settings' section is highlighted with a red box and a red arrow pointing to the 'Enable auto-assign public IPv4 address' checkbox.

Auto-assign IP settings
Enable AWS to automatically assign a public IPv4 or IPv6 address to a new primary network interface for an instance in this subnet.

Enable auto-assign public IPv4 address
 Enable auto-assign customer-owned IPv4 address
Option disabled because no customer owned pools found.

Resource-based name (RBN) settings
Specify the hostname type for EC2 instances in this subnet and optional RBN DNS query settings.

Enable resource name DNS A record on launch
 Enable resource name DNS AAAA record on launch
Hostname type

And Click on Save.

Now we have check for private subnet is auto assign IP is enabled or off .Need to set off because we are making private server using that subnet lets check It.

subnet-Oad60e2f8e81d060d / Private_NAT

Details

Subnet ID	subnet-Oad60e2f8e81d060d
IPv4 CIDR	192.168.128.0/18
Availability Zone	us-east-1f
Route table	rtb-02a6fe6c8c9bfe06e
Auto-assign IPv6 address	No
IPv4 CIDR reservations	-
Resource name DNS A record	Disabled
Subnet ARN	arn:aws:ec2:us-east-1:351352951218:subnet/subnet-Oad60e2f8e81d060d
Available IPv4 addresses	16379
Availability Zone ID	use1-az5
Network ACL	acl-0712045810eac1874
Auto-assign customer-owned IPv4 address	No
IPv6 CIDR reservations	-
Resource name DNS AAAA record	Disabled
State	Available
IPv6 CIDR	-
Network border group	us-east-1
Default subnet	No
Customer-owned IPv4 pool	-
IPv6-only	No
DNS64	Disabled
Block Public Access	Off
IPv6 CIDR association ID	-
VPC	vpc-039630cd94add2198 NAT-VPC
Auto-assign public IPv4 address	<input checked="" type="checkbox"/>
Outpost ID	-
Hostname type	IP name
Owner	351352951218

If here shows Yes go to actions and edit subnet settings and uncheck the box as shown in the snap.

Edit subnet settings

Subnet

Subnet ID	subnet-Oad60e2f8e81d060d	Name	Private_NAT
Auto-assign IP settings <small>Info</small>			
Enable AWS to automatically assign a public IPv4 or IPv6 address to a new primary network interface for an instance in this subnet.			
<input type="checkbox"/> Enable auto-assign public IPv4 address <small>Info</small> <input type="checkbox"/> Enable auto-assign customer-owned IPv4 address <small>Info</small> <small>Option disabled because no customer-owned pools found.</small>			
Resource-based name (RBN) settings <small>Info</small>			
Specify the hostname type for EC2 instances in this subnet and optional RBN DNS query settings.			
<input type="checkbox"/> Enable resource name DNS A record on launch <small>Info</small> <input type="checkbox"/> Enable resource name DNS AAAA record on launch <small>Info</small>			
Hostname type <small>Info</small>			

When create subnets in VPC it automatically gives us a route table as you can see in the snap.

Your VPCs

Resource map

Enabled	default	default	default
Main network ACL	acl-0712045810eac1874	Default VPC	dopt-058c9890e15e8aa56
IPv6 CIDR (Network border group)	-	Network Address Usage metrics	rtb-02a6fe6c8c9bfe06e
Route 53 Resolver DNS Firewall rule groups	-	IPv6 pool	IPv6 pool
Owner ID	351352951218	Route tables (1)	Route tables (1)

Resource map

- VPC** Show details Your AWS virtual network
NAT-VPC
- Subnets (2)** Subnets within this VPC
us-east-1f
Private_NAT
Public_NAT
- Route tables (1)** Route network traffic to resources
rtb-02a6fe6c8c9bfe06e
- Network connections (0)** Connections to other networks

But you can see in the snap we don't have any internet gateway because we have created own vpc if we use default VPC we get automatically attached Internet gateway to our VPC .So now **we have to create a internet gateway for our VPCs and for its Subnets which connects by route table.**

The screenshot shows the AWS VPC dashboard with the 'Internet gateways' section selected. The table lists one existing Internet gateway: 'igw-014b7c1da638c3742' (Attached, VPC ID: 'vpc-0dbbb05820b87165c'). A red box highlights the 'Create internet gateway' button at the top right of the table.

The screenshot shows the 'Create internet gateway' wizard. Step 1: Internet gateway settings. The 'Name tag' input field contains 'NAT-IGW'. Step 2: Tags - optional. The 'Add new tag' button is highlighted with a red box. A red box also highlights the 'Create internet gateway' button at the bottom right.

Now we need to attach that Internet gateway to our VPC

The screenshot shows the AWS VPC dashboard with the 'Internet gateways' section selected. A red box highlights the 'NAT-IGW' row. A context menu is open over the row, with the 'Attach to VPC' option highlighted with a red box.

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.

Q vpc-039630cd94add2198 X

Use: "vpc-039630cd94add2198"
AWS Command Line Interface command
vpc-039630cd94add2198 - NAT-VPC

Cancel Attach internet gateway

We have connect internet gateway to route table which we provide internet to out subnet later

VPC dashboard < Enabled default dopt-058c9890e15e8aa56 rtb-02a6fe6c8c9bfe06e

Main network ACL acf-0712045810eac1874 Default VPC IPv4 CIDR 192.168.0.0/16 IPv6 pool -

IPv6 CIDR (Network border group) - Network Address Usage metrics Disabled Route 53 Resolver DNS Firewall rule groups -

Owner ID 351352951218

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

Resource map | CIDs | Flow logs | Tags | Integrations

Resource map Info

VPC Show details Your AWS virtual network NAT-VPC

Subnets (2) Subnets within this VPC us-east-1f Private_NAT Public_NAT

Route tables (1) Route network traffic to resources rtb-02a6fe6c8c9bfe06e

Network connections (1) Connections to other networks NAT-IGW

Got to route table ID - rtb-02a6fe6c8c9bfe06e

VPC dashboard < rtb-02a6fe6c8c9bfe06e Actions ▾

Details Info

Route table ID rtb-02a6fe6c8c9bfe06e Main Explicit subnet associations Edge associations

VPC vpc-039630cd94add2198 | NAT-VPC Owner ID 351352951218

Routes Subnet associations Edge associations Route propagation Tags

Routes (1)

Filter routes Destination Target Status Propagated

192.168.0.0/16 local Active No

Both ▾ Edit routes

The screenshot shows the 'Edit routes' page for a specific route table. A red box highlights the 'Add route' button at the bottom left. The table has one row:

Destination	Target	Status	Propagated
192.168.0.0/16	local	Active	No

Buttons at the bottom right include 'Cancel', 'Preview', and 'Save changes'.

The screenshot shows the 'Edit routes' page for a specific route table. A red box highlights the '192.168.0.0/17' destination field. The table has two rows:

Destination	Target	Status	Propagated
192.168.0.0/16	local	Active	No
192.168.0.0/17	Internet Gateway	-	No

Buttons at the bottom right include 'Cancel', 'Preview', and 'Save changes'.

Here we have added specific IPs which shows internet only route on or over specific IPs or subnets. Because we have to isolate our server if you are testing you can set like default all (0.0.0.0/0) its your choice.

Save changes.

The screenshot shows the 'Edit routes' page for a specific route table. A red box highlights the '0.0.0.0/0' destination field. The table has two rows:

Destination	Target	Status	Propagated
192.168.0.0/16	local	Active	No
0.0.0.0/0	Internet Gateway	-	No

Buttons at the bottom right include 'Cancel', 'Preview', and 'Save changes'.

The screenshot shows the 'VPC dashboard' page. A green banner at the top says 'Updated routes for rtb-02a6fe6c8c9bfe06e successfully'. Below it is the 'rtb-02a6fe6c8c9bfe06e' route table details:

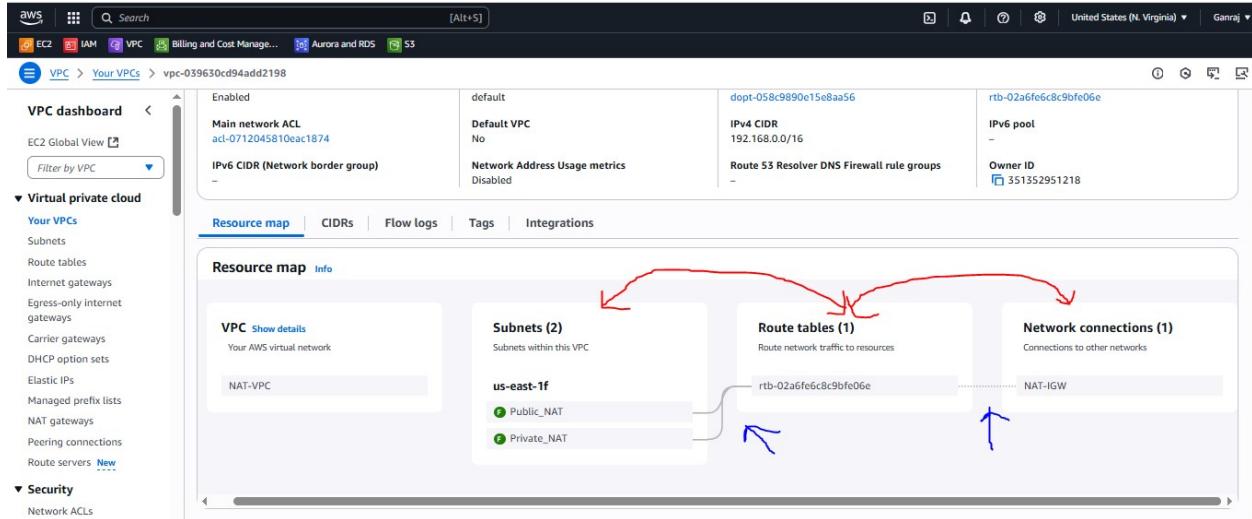
Details		Main	Explicit subnet associations	Edge associations
Route table ID	rtb-02a6fe6c8c9bfe06e	Yes	-	-
VPC	vpc-039630cd94add2198 NAT-VPN	Owner ID	351352951218	-

The 'Routes' tab shows two entries:

Destination	Target	Status	Propagated
0.0.0.0/0	igw-0d34d81981e0a061b	Active	No
192.168.0.0/16	local	Active	No

Buttons at the bottom right include 'Both', 'Edit routes', and navigation arrows.

As we can see in the below snap connection between internet gateway to route table to subnets are made see but note if we made specific subnet ip connection then its will not same as below.



Now we will create NAT gateway

The screenshot shows the AWS VPC dashboard under the 'NAT gateways' section. The sidebar includes 'Your VPCs', 'Subnets', 'Route tables', 'NAT gateways' (which is highlighted in yellow), and 'Security'. The main area shows a table with no results found. At the top right, there is a button labeled 'Create NAT gateway' which is highlighted with a red box.

NAT gateway only Attached to Public Subnet because internet access is only available for Public subnet

The screenshot shows the 'Create NAT gateway' wizard. It has three steps: 'Create NAT gateway info', 'NAT gateway settings', and 'Review and launch'. The 'NAT gateway settings' step is currently active. It includes fields for 'Name' (set to 'NAT-Internet'), 'Subnet' (set to 'subnet-0dd6b0d4930354934 (Public_NAT)'), and 'Connectivity type' (set to 'Public'). The 'Subnet' field is highlighted with a red box.

For NAT we need Elastic IP because it works on Static IP which is **fixed public IP**. Access the internet (outbound) but **Remain unreachable from the internet (inbound)** — for security

NAT gateway settings

Name - optional
Create a tag with a key of 'Name' and a value that you specify.
NAT-Internet

The name can be up to 256 characters long.

Subnet
Select a subnet in which to create the NAT gateway.
subnet-0dd6b0d4930354934 (Public_NAT)

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

Elastic IP allocation ID **eipalloc-0a73c561b5df40605**

Allocate Elastic IP

NAT gateway nat-0803fa2e34495c4b6 | NAT-Internet was created successfully.

Details

NAT gateway ID nat-0803fa2e34495c4b6	Connectivity type Public	State Pending	State message —
NAT gateway ARN arn:aws:ec2:us-east-1:135135295121:natgateway/nat-0803fa2e34495c4b6	Primary public IPv4 address	Primary private IPv4 address —	Primary network interface ID —
VPC vpc-039630cd94add2198 / NAT-VPC	Subnet subnet-0dd6b0d4930354934 / Public_NAT	Created Monday, July 21, 2025 at 20:10:01 GMT+5:30	Deleted —

Secondary IPv4 addresses | Monitoring | Tags

Secondary IPv4 addresses

Here in the below snap we can see public subnet have internet but not for private because of route table for that we need to create another route table to make connection through NAT

VPC dashboard

Enabled
Main network ACL
ac-0712045810eac1874

IPv6 CIDR (Network border group)
—

default
Default VPC
No

IPV4 CIDR
192.168.0.0/16

Route 53 Resolver DNS Firewall rule groups
—

rtb-02a6fe6c8c9bfe06e
IPv6 pool
—

Owner ID
351352951218

Virtual private cloud

Your VPCs
vpc-039630cd94add2198

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**

Resource map

Subnets (2)
Subnets within this VPC
NAT-VPC

us-east-1f
Public_NAT
Private_NAT

Route tables (1)
Route network traffic to resources
rtb-02a6fe6c8c9bfe06e

Network connections (2)
Connections to other networks
NAT-IGW
NAT-Internet

Now creating new route table for private

Route tables (2) Info

Name	Route table ID	Explicit subnet assoc...	Edge associations	Main	VPC	Owner ID
-	rtb-02a6fe6c8c9bfe06e	-	-	Yes	vpc-039630cd94add2198 NAT...	351352951218
-	rtb-0828c3c2ce62491e0	-	-	Yes	vpc-0dbbb05820b87165c	351352951218

Select a route table

Create route table

A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.

Route table settings

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

VPC
The VPC to use for this route table.
vpc-039630cd94add2198 (NAT-VPN)

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** **Remove**

Add new tag
You can add 49 more tags.

Create route table

Now we have connect our private route table associated with only with Private Subnet later on NAT also

VPC dashboard

Resource map

Subnets

- NAT-VPN
- us-east-1f
 - Public_NAT
 - Private_NAT

Route tables

- rtb-02a6fe6c8c9bfe06e
- rtb-0828c3c2ce62491e0

Network connections

- NAT-IGW
- NAT-Internet

Now we will use private route to do subnet association and add NAT to be routed with Private Route table

The screenshot shows the AWS VPC Resource Map interface. On the left, there's a sidebar with 'Virtual private cloud' sections for 'Your VPCs', 'Subnets', 'Route tables', and 'Security'. The main area displays a 'Resource map' with four panels: 'VPC' (showing details like 'Your AWS virtual network' and 'NAT-VPN'), 'Subnets (2)' (listing 'us-east-1f' with 'Public_NAT' and 'Private_NAT' subnets), 'Route tables (2)' (listing 'rtb-02a6fe6c8c9bfe06e' and 'rtb-0be70f9857b4e93b7'), and 'Network connections (2)' (listing 'NAT-IGW' and 'NAT-Internet'). A red box highlights the 'Private_NAT' subnet and its association with the 'rtb-02a6fe6c8c9bfe06e' route table.

VPC dashboard

- Enabled: default
- Main network ACL: acl-0712045810ea1874
- IPv6 CIDR (Network border group): -
- Route 53 Resolver DNS Firewall rule groups: -

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
- Security

Resource map

Subnets (2)

- Subnets within this VPC
- us-east-1f
 - Public_NAT
 - Private_NAT (192.168.128.0/18 No IPv6)

Route tables (2)

- Route network traffic to resources
- rtb-02a6fe6c8c9bfe06e (Private)
- rtb-0be70f9857b4e93b7

Network connections (2)

- Connections to other networks
- NAT-IGW
- NAT-Internet

Route tables

Explicit subnet associations (0)

No subnet associations

Subnets without explicit associations (2)

The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
Private_NAT	subnet-0ad60e2f8e81d060d	192.168.128.0/18	-
Public_NAT	subnet-0dd6b0d4930354934	192.168.0.0/17	-

Here we have done the subnet association with route table for exclusive private route table will associated with Private Subnet (192.168.128.0/18)

The screenshot shows the 'Edit subnet associations' dialog. It has two main sections: 'Available subnets (1/2)' and 'Selected subnets'. In the 'Available subnets' section, 'Private_NAT' is selected (indicated by a checked checkbox). In the 'Selected subnets' section, 'subnet-0ad60e2f8e81d060d / Private_NAT' is listed. At the bottom right, a red box highlights the 'Save associations' button.

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (1/2)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
Private_NAT	subnet-0ad60e2f8e81d060d	192.168.128.0/18	-	Main (rtb-02a6fe6c8c9bfe06e)
Public_NAT	subnet-0dd6b0d4930354934	192.168.0.0/17	-	Main (rtb-02a6fe6c8c9bfe06e)

Selected subnets

subnet-0ad60e2f8e81d060d / Private_NAT

Cancel Save associations

You have successfully updated subnet associations for rtb-0be70f9857b4e93b7 / Private.

Details Info

Route table ID rtb-0be70f9857b4e93b7	Main No	Explicit subnet associations subnet-0ad60e2f8e81d060d / Private_NAT	Edge associations -
VPC vpc-039630cd94add2198 NAT-VPC	Owner ID 351352951218		

Subnet associations

Explicit subnet associations (1)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
Private_NAT	subnet-0ad60e2f8e81d060d	192.168.0.0/18	-

Subnets without explicit associations (1)

The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
Public_NAT	subnet-0dd6b0d493034934	192.168.0.0/17	-

Now we will route the NAT towards Private subnet using Route table (Private) as you can see below snap

Resource map

- VPC** Show details Your AWS virtual network
- Subnets (2)** Subnets within this VPC
 - us-east-1f
 - Public_NAT
 - Private_NAT
- Route tables (2)** Route network traffic to resources
 - rtb-02a6fe6c8c9bfe06e
 - Private
- Network connections (2)** Connections to other networks
 - NAT-IGW
 - NAT-Internet

So go to route table's Routes

rtb-0be70f9857b4e93b7 / Private

Routes (1)

Destination	Target	Status	Propagated
192.168.0.0/16	local	Active	No

Actions

The screenshot shows the 'Edit routes' page for a specific route table. A route entry for '0.0.0.0/0' is selected and points to a 'NAT Gateway' target. The target is identified by its ID: 'nat-0803fa2e34495cb6'. A red box highlights this target entry. At the bottom right, there is a 'Save changes' button, which is also highlighted with a red box.

The screenshot shows the 'Resource map' interface for a VPC. It displays four main components: VPC, Subnets, Route tables, and Network connections. Arrows indicate the flow of traffic from Subnets through Route tables to Network connections. A red arrow points from the 'Subnets (2)' section to the 'Route tables (2)' section, specifically highlighting the connection between them.

Its will show connection like this also its depends on representation of AWS.

This screenshot of the 'Resource map' interface is similar to the one above, but it includes a large red arrow pointing from the 'Subnets (2)' section to the 'Route tables (2)' section, emphasizing the flow of traffic from subnets to route tables.

See we have made connection between NAT to Internet now we have create Instance to check whether our process giving internet of Public server to private server and the private server access only through public server

Note : - Private Server

- Cannot be accessed directly (no public IP, no IGW route)
- Can only go out to the internet using NAT Gateway
- Must be accessed via:
 - Bastion Host (jump box in public subnet)
 - Public-facing server (with SSH access + internal connectivity)
 - VPN / Direct Connect if you want private access from outside AWS

Now lets create Public and private servers in their specific subnets using VPC

Here while selecting VPC click on our VPC and change subnet to public subnet it important change for our process also check Auto assign IP is Enable because we creating Public server.

Create security group which have allowed inbound rule for internet and SSH protocol for accessing Linux type OS and launch it.

Now We will create private server

Name and tags

Name: Private_Server

Software image (AMI)

Amazon Linux 2023 AMI 2023.8.2... read more

Virtual server type (instance type)

t2.micro

Summary

Number of instances: 1

Software image (AMI)

Amazon Linux 2023 AMI 2023.8.2... read more

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GB

Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs. 750 hours per month of public IPv4 address usage, 30 GB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Launch Instance

Here also we have change Default VPC to our VPC that we have created and Private Subnet

Network settings

VPC - required: vpc-039630cd94add2198

Auto-assign public IP: Disable

Firewall (security groups): Create new security group

Summary

Number of instances: 1

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GB

Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs. 750 hours per month of public IPv4 address usage, 30 GB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Launch Instance

Here we can see auto assign IP need to disabled because we have set our private server to isolated from direct access through internet.

Network settings

VPC - required: vpc-039630cd94add2198 (NAT-VPC)

Auto-assign public IP: Disable

Firewall (security groups): Create new security group

Common security groups: NAT sg-0006ecd374bf7ada9

Number of instances: 1

t2.micro

Firewall (security group)

NAT

Storage (volumes)

1 volume(s) - 8 GB

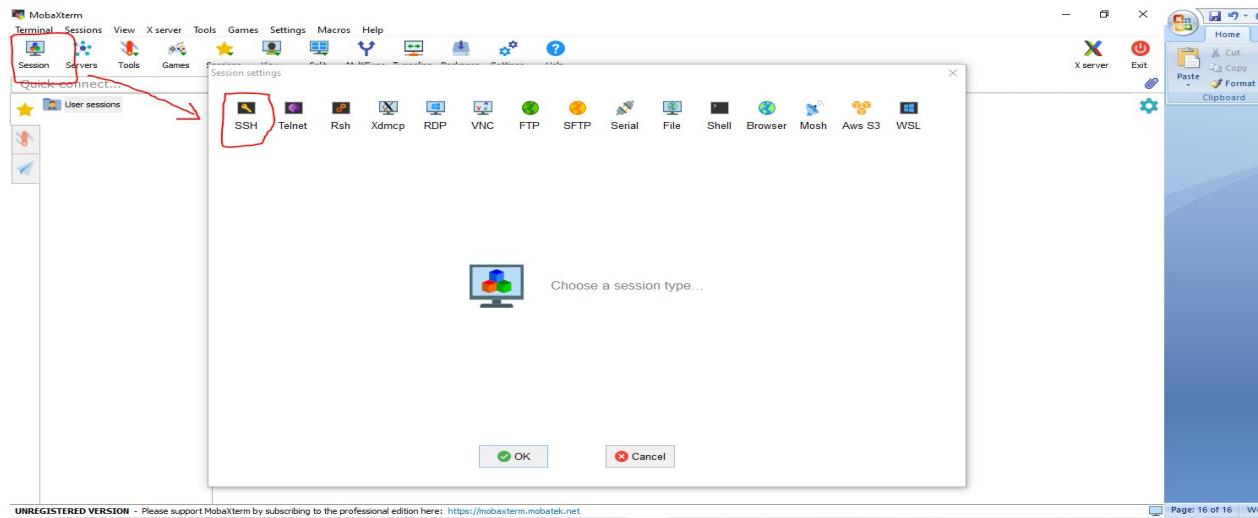
Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs. 750 hours per month of public IPv4 address usage, 30 GB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Launch Instance

See we have created Public server and Private Server but if we want to access our private we need to go through Public so let's do process using Mobaxterm because we need to add Private Key to our Public server which will access to Private Server

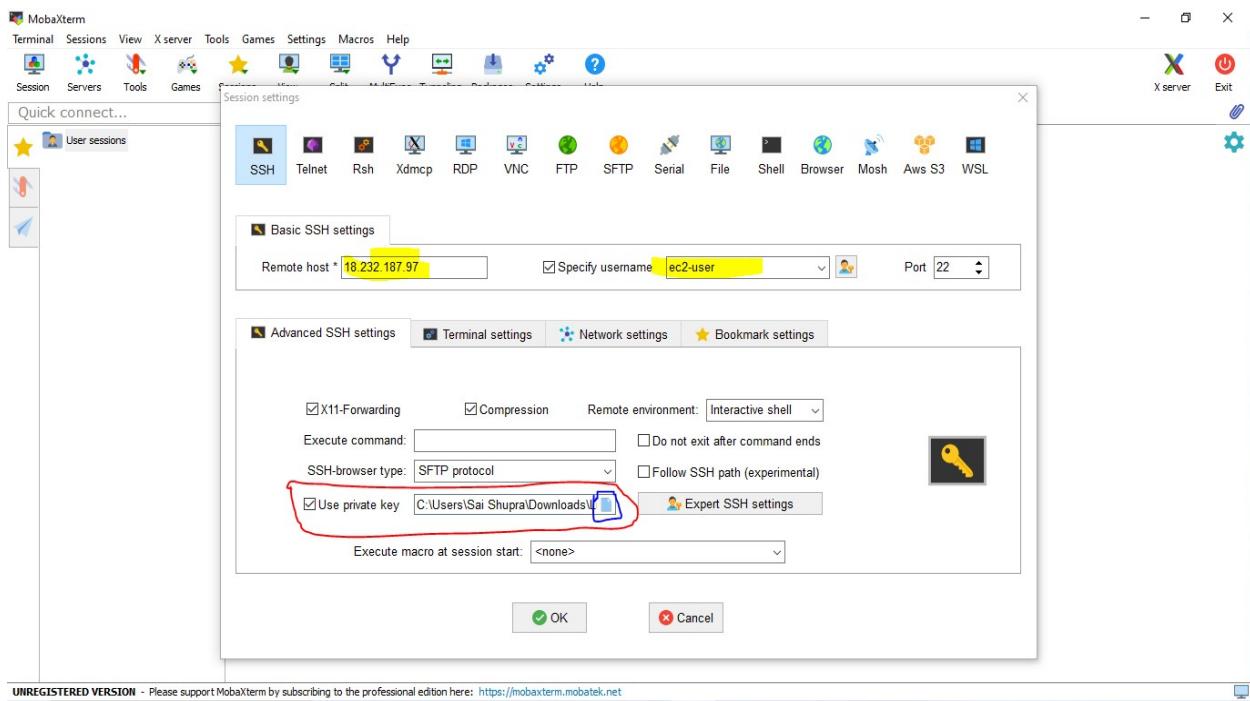
You can download MobaXterm from here Free

<https://mobaxterm.mobatek.net/download.html>

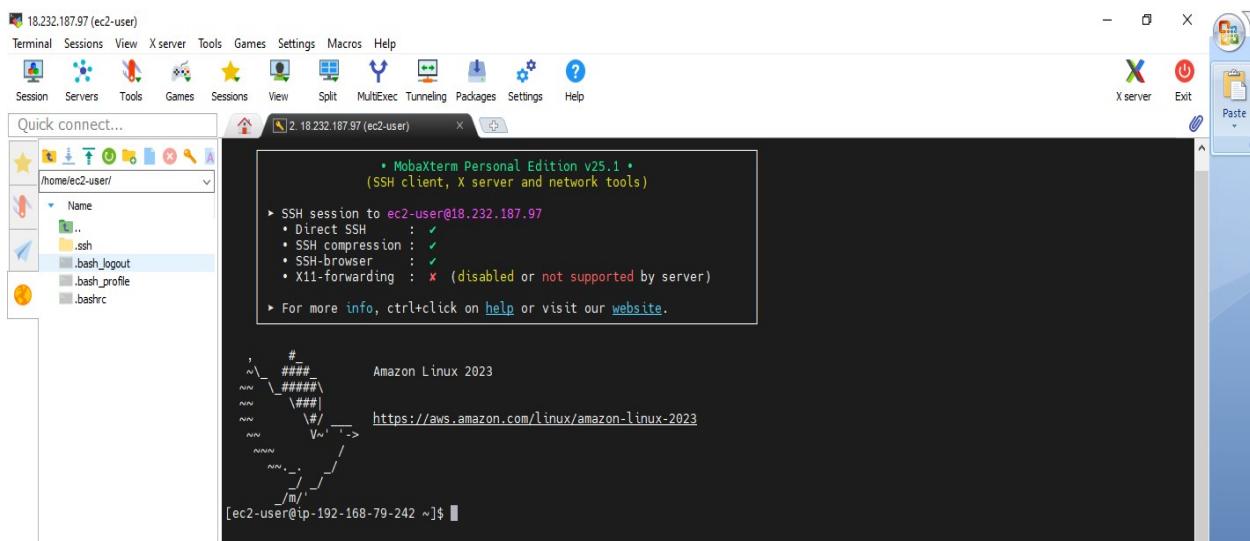


Instance summary for i-049ccd11ea87d9510 (Public_Server)	
Updated 1 minute ago	Info
Instance ID	i-049ccd11ea87d9510
IPv6 address	-
Hostname type	IP name: ip-192-168-79-242.ec2.internal
Answer private resource DNS name	-
Auto-assigned IP address	18.232.187.97 [Public IP]
IAM Role	-
IMDSv2	Required
Operator	-
Private IP4 address	18.232.187.97 open address
Instance state	Running
Private IP DNS name (IPv4 only)	ip-192-168-79-242.ec2.internal
Instance type	t2.micro
VPC ID	vpc-039630cd94add2198 (NAT-VPC)
Subnet ID	subnet-0dd6b0d4930354934 (Public_NAT)
Instance ARN	arn:aws:ec2:us-east-1:351352951218:instance/i-049ccd11ea87d9510
Private IPv4 addresses	192.168.79.242
Public DNS	-
Elastic IP addresses	-
AWS Compute Optimizer finding	Opt-in to AWS Compute Optimizer for recommendations.
Auto Scaling Group name	-
Managed	false

Copy the Public IP of Public server mention specific user for amazon we put ec2-user and in the advance option of MobXterm open saved Private key of public server then we can access public server using MobaXterm .



Click OK.

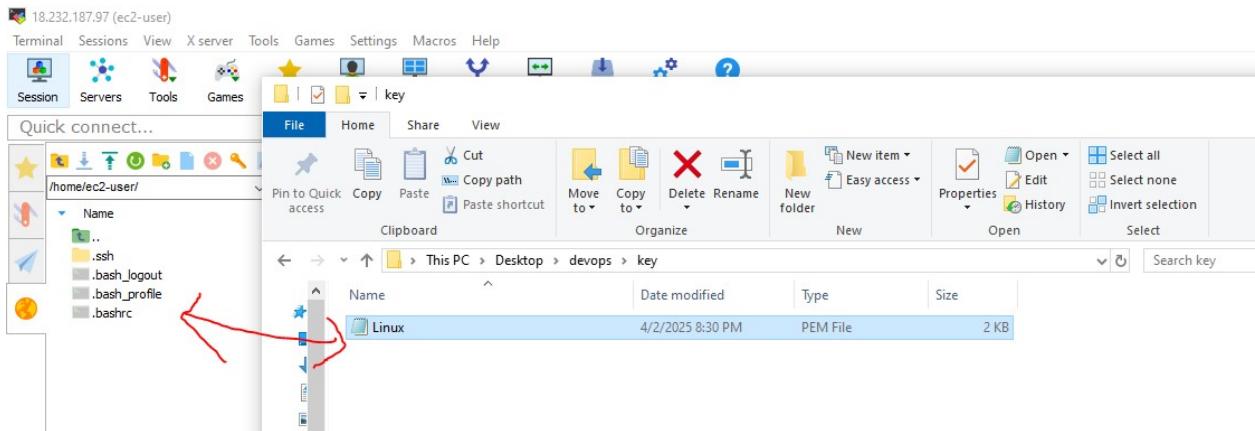


See we now connected to our public server which has internet access now we have connect to Private server using a command

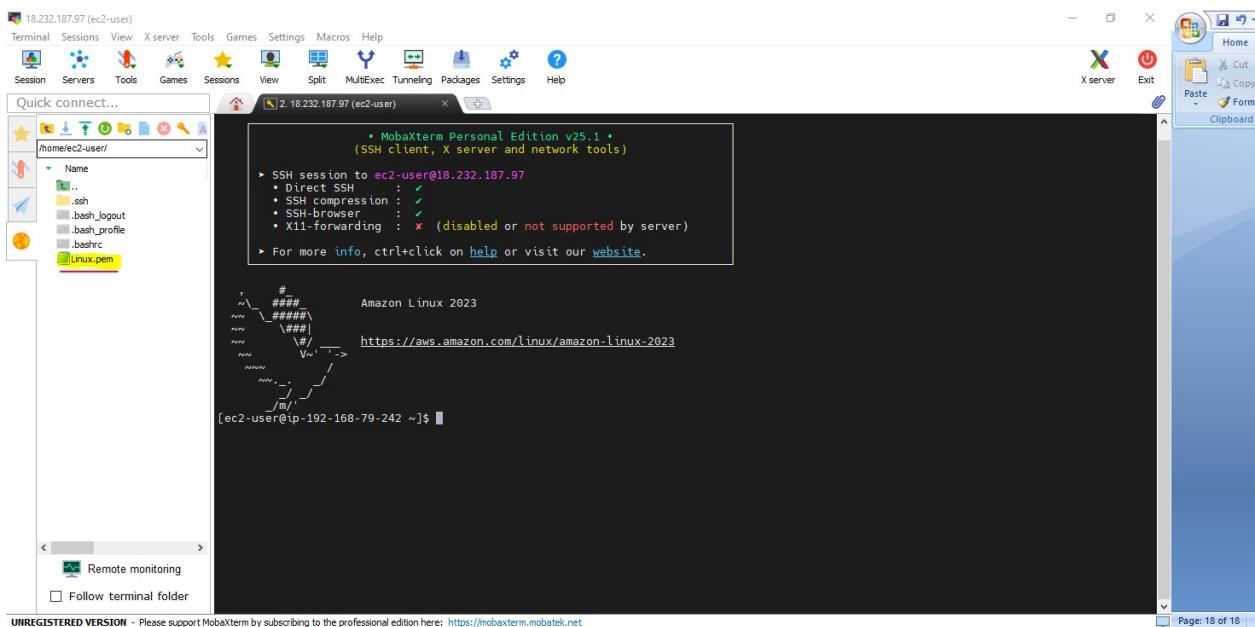
`ssh -i 'Private-key' ec2-user@private-IP`

for that we have to add private server's private key lets add it

In the MobaXterm we have one advantage that we can drag drop data files from our local machine to VM(EC2 Instances)



As you can see copied it from our machine to EC2 instance

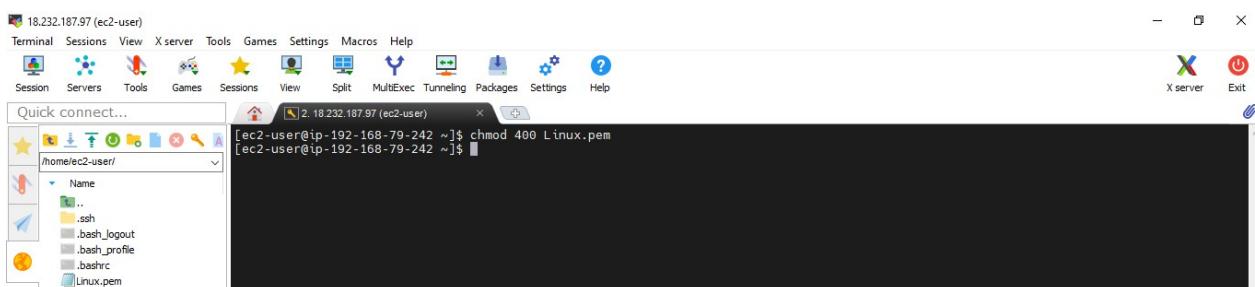


Now we will use that command and connect Private Server on Public Server

But first we need to give access to that file for Public server user

Using :- chmod 400 privatekeyname

chmod 400 Linux.pem



Now use this command to connect private server

ssh -i 'Linux.pem' ec2-user@192.168.176.241

The screenshot shows the AWS Management Console with the EC2 service selected. In the left sidebar, 'Instances' is expanded, showing various instance-related options like Launch Templates, Spot Requests, and Reserved Instances. The main content area displays the 'Instance summary for i-00ebd09e33e90f664 (Private_Server)'. Key details shown include:

- Private IP4 address:** 192.168.176.241 (highlighted with a red box)
- Instance state:** Running
- Private IP DNS name (IPv4 only):** ip-192-168-176-241.ec2.internal
- Instance type:** t2.micro
- VPC ID:** vpc-039630cd94add2198 (NAT-VPC)
- Subnet ID:** subnet-0ad60e2f8e81d060d (Private_NAT)
- Instance ARN:** arn:aws:ec2:us-east-1:351352951218:instance/i-00ebd09e33e90f664

Copying related private IP from Private server Instance and Private key name

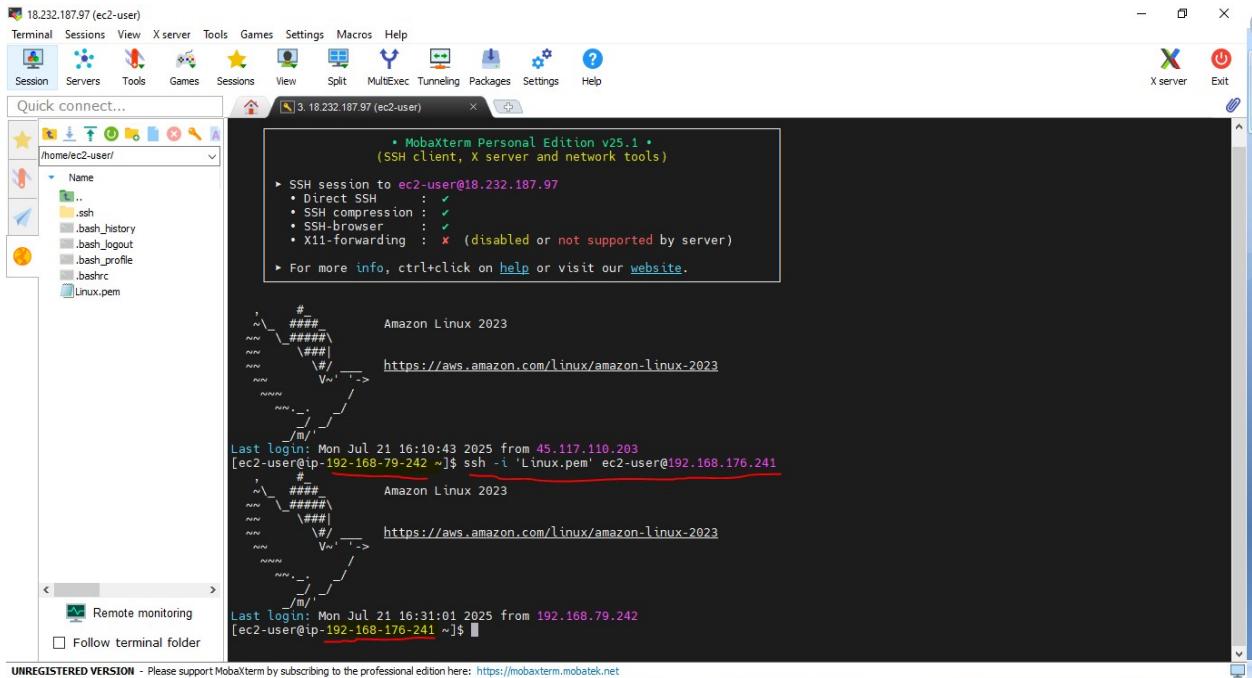
ssh -i 'Linux.pem' ec2-user@192.168.176.241

The screenshot shows a MobaXterm window titled '18.232.187.97 (ec2-user)'. The terminal session is connected to the private IP 192.168.176.241. The command entered is:

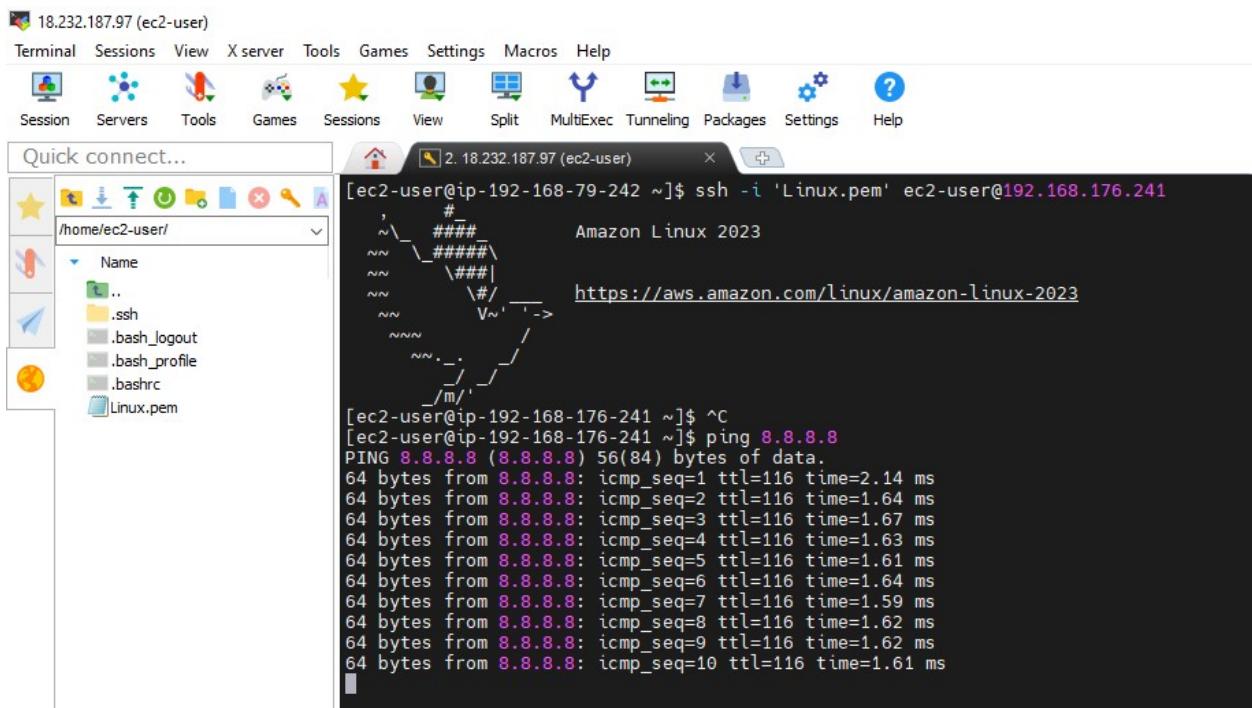
```
[ec2-user@ip-192-168-79-242 ~]$ ssh -i 'Linux.pem' ec2-user@192.168.176.241
```

The terminal output shows the user is connected to an Amazon Linux 2023 instance, with the URL <https://aws.amazon.com/linux/amazon-linux-2023> displayed.

This screenshot shows reconnection of Private server using Public server



As you can see in the snap we have connected to private server using Public server, Now we will check Internet Access .



See in this scenario we have achieved process of NAT which provides internet access Public Server(EC2) to Private Server(EC2) and The access will provide only through Public Server