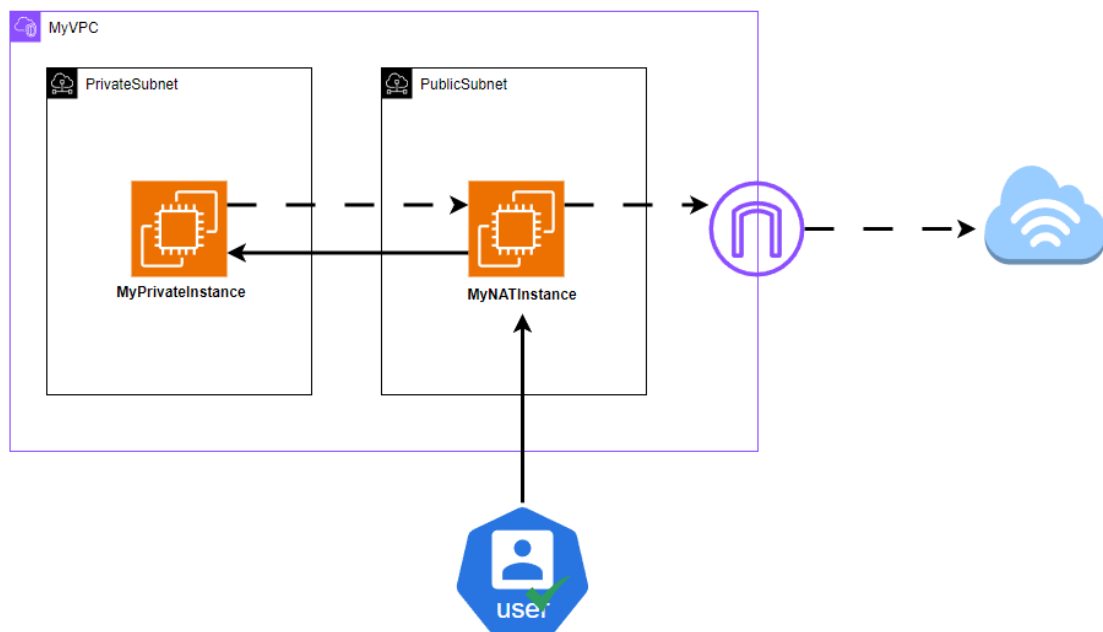


Guided Lab: Configuring a NAT Instance for Secure Connectivity

Description

In modern cloud architectures, ensuring secure and efficient connectivity for instances within private subnets is crucial. Network Address Translation (NAT) instances play a pivotal role in achieving this by allowing instances in private subnets to communicate with the internet for outbound traffic (e.g., software updates) without exposing them to inbound connections. This guided lab will demonstrate the setup and configuration of a NAT instance within an Amazon VPC, simulating a scenario where instances in a private subnet need to connect securely to a NAT instance.



Prerequisites

This lab assumes you have basic knowledge of Amazon EC2 Instance, VPCs & basic network configuration in AWS Cloud.

If you find any gaps in your knowledge, consider taking the following lab:

- Creating an Amazon EC2 instance (Linux)
- Creating a Custom Virtual Private Cloud (VPC) from scratch

Objectives

By the end of this lab, participants will be able to:

- Set up a VPC with public and private subnets.
- Launch and configure EC2 instances in both public and private subnets.
- Create and configure a NAT instance to enable secure communication for instances in the private subnet.

- **Modify route tables to direct traffic appropriately.**
- **Establish a secure connection between a private instance and a NAT instance.**

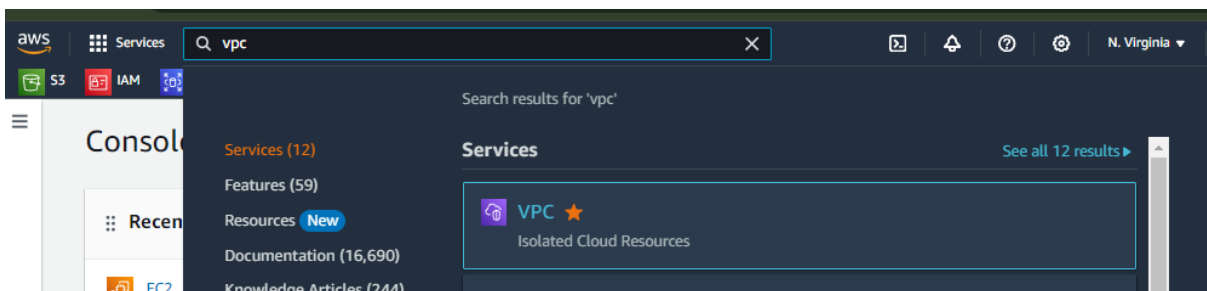
Subscribe to access AWS PlayCloud Labs

Lab Steps

Setup Network Configurations

1. Create a VPC:

- **Go to the VPC dashboard in the AWS Management Console.**



- **Click on Create VPC.**
- **Resource to create: VPC only**
- **Name: MyVPC**
- **CIDR block: 10.0.0.0/16**
- **Tenancy: Default**
- **Click Create.**

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.

MyVPC

IPv4 CIDR block [Info](#)

☒ IPv4 CIDR manual input

☐ IPAM-allocated IPv4 CIDR block

IPv4 CIDR

10.0.0.0/16

CIDR block size must be between /16 and /28.

IPv6 CIDR block [Info](#)

☒ No IPv6 CIDR block

☐ IPAM-allocated IPv6 CIDR block

☐ Amazon-provided IPv6 CIDR block

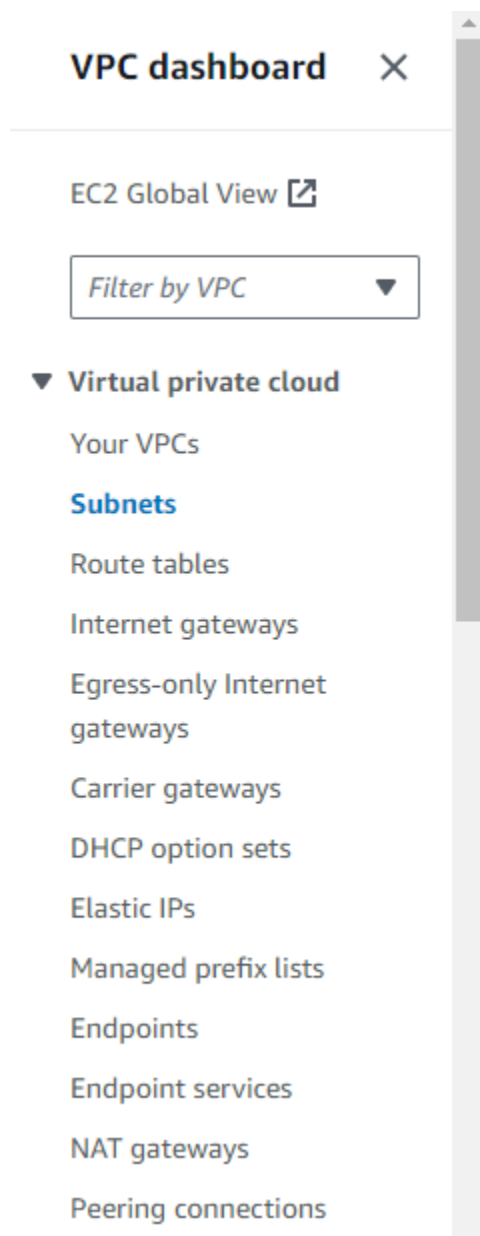
☐ IPv6 CIDR owned by me

Tenancy [Info](#)

Default

2. Create Subnets:

- **Navigate to Subnets in th VPC dashboard.**



- **Create two subnets with the following configurations:**
 - **Public Subnet:**
 - **VPC ID: MyVPC**
 - **Subnet name: PublicSubnet**
 - **Availability Zone: Select one from the list (e.g. us-east-1a)**
 - **IPv4 subnet CIDR block: 10.0.1.0/24**

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.

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.

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.

IPv4 subnet CIDR block

256 IPs

< > ^ v

▼ Tags - optional

Key

X

Value - optional

X

Remove

Add new tag

You can add 49 more tags.

Remove

Add new subnet

Cancel

Create subnet

- - - Add new subnet
 - Private Subnet: (Click on Add new subnet)
 - Name: PrivateSubnet
 - IPv4 subnet CIDR block: 10.0.2.0/24
 - Availability Zone: Same as the public subnet
 - Click Create.

Subnet 2 of 2

Subnet name

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

PrivateSubnet

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.

US East (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.

10.0.0.0/16

IPv4 subnet CIDR block

10.0.2.0/24256 IPs

< > ^ v

▼ Tags - optional

Key

Value - optional

Q NameX

Q PrivateSubnetX

Remove

Add new tag

You can add 49 more tags.

Remove

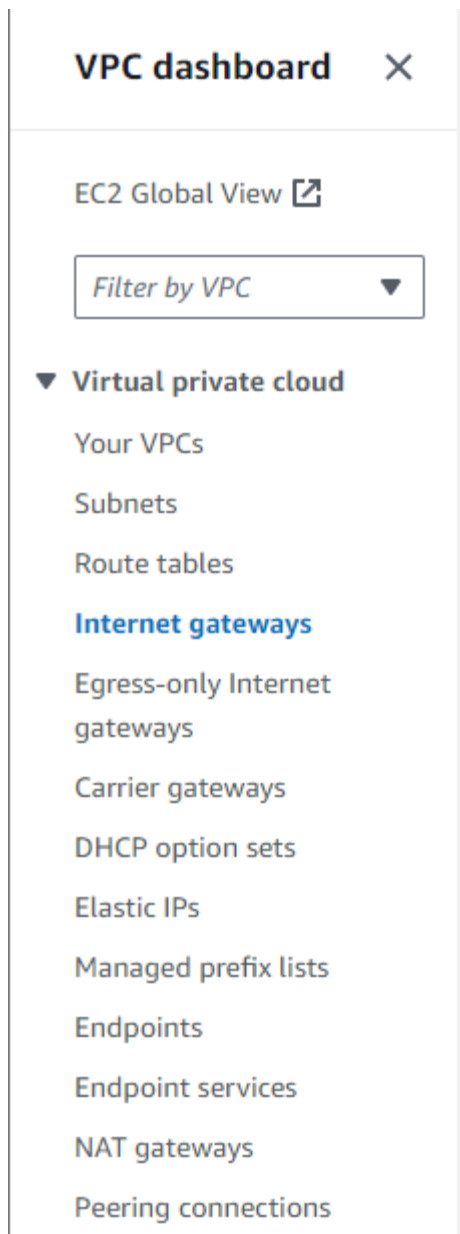
Add new subnet

Cancel

Create subnet

3. Create an Internet Gateway:

- Go to Internet Gateways in the VPC dashboard.



- Click **Create internet gateway**.
- **Name:** MyIGW
- Click **Create**.
- **Attach the Internet Gateway to MyVPC.**

Internet gateway igw-0f3a08b006ab1ca9c successfully attached to vpc-0aa6a3da4f569b288

Notifications 0 0 2 0 0 0

VPC > Internet gateways > igw-0f3a08b006ab1ca9c

igw-0f3a08b006ab1ca9c / MyIGW

Actions

Details Info

Internet gateway ID igw-0f3a08b006ab1ca9c	State Attached	VPC ID vpc-0aa6a3da4f569b288 MyVPC	Owner 767398024881
--	-------------------	--	-----------------------

Tags

Search tags

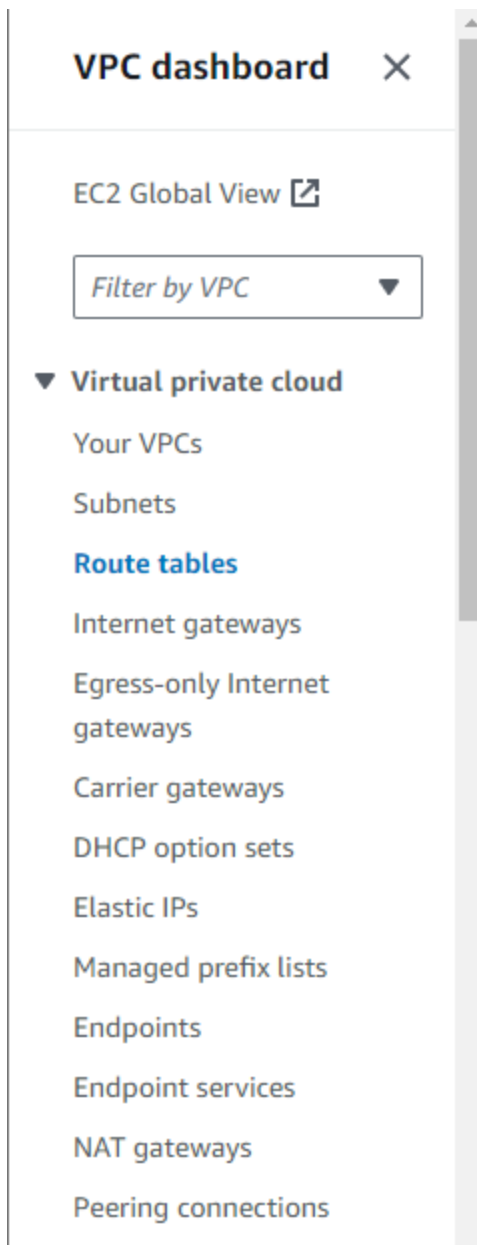
Manage tags

Key	Value
Name	MyIGW

4. Create a new Route Table for the Public Subnet

Take note that there should be a new Route Table created after creating a new VPC.

- Go to the Route Tables section in the VPC dashboard



- Click Create route table
- Name: PublicRT
- VPC: MyVPC
- Click Create route table
- Edit routes: Add route 0.0.0.0/0 to target MyIGW.
- Save changes.

Edit routes

Route 1

Destination
10.0.0.0/16

Target

local ▼

Status

✓ Active

Q local X

Propagated
No

Route 2

Destination

Q 0.0.0.0/0 X

Target

Internet Gateway ▼

Status

-

Q igw-0f3a08b006ab1ca9c X

Use: "igw-0f3a08b006ab1ca9c"

igw-0f3a08b006ab1ca9c (MyIGW)

Propagated
No

Remove

Add route

Cancel

Preview

Save changes

- Edit Explicit subnet associations in the Subnet association tab: Associate PublicSubnet.

You have successfully updated subnet associations for rtb-0a63b657993e2ddd6 / PublicRC.

VPC > Route tables > rtb-0a63b657993e2ddd6

rtb-0a63b657993e2ddd6 / PublicRT Actions ▾

Details Info

Route table ID rtb-0a63b657993e2ddd6	Main No	Explicit subnet associations subnet-01d16a350761fd9e8 / PublicSubnet	Edge associations -
VPC vpc-0aa6a3da4f569b288 MyVPC	Owner ID 767398024881		

Routes Subnet associations Edge associations Route propagation Tags

Explicit subnet associations (1) Edit subnet associations

Find subnet association

Name ▾	Subnet ID ▾	IPv4 CIDR ▾	IPv6 CIDR ▾
PublicSubnet	subnet-01d16a350761fd...	10.0.1.0/24	-

Subnets without explicit associations (1) Edit subnet associations

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

Find subnet association

Name ▾	Subnet ID ▾	IPv4 CIDR ▾	IPv6 CIDR ▾
PrivateSubnet	subnet-029c91c96743d4...	10.0.2.0/24	-

5. Create another Route Table for Private Subnet

- Create another route table: Name: PrivateRouteTable, VPC: MyVPC.
- Edit Explicit subnet associations in the Subnet association tab: Associate PrivateSubnet.

✓ You have successfully updated subnet associations for rtb-03d4b575529f40fb1 / PrivateRT. ✕

VPC > Route tables > rtb-03d4b575529f40fb1

rtb-03d4b575529f40fb1 / PrivateRT Actions ▾

Details [Info](#)

Route table ID rtb-03d4b575529f40fb1	Main No	Explicit subnet associations subnet-029c91c96743d47fc / PrivateSubnet	Edge associations –
VPC vpc-0aa6a3da4f569b288 MyVPC	Owner ID 767398024881		

[Routes](#) | **[Subnet associations](#)** | [Edge associations](#) | [Route propagation](#) | [Tags](#)

Explicit subnet associations (1) Edit subnet associations

< 1 > ⚙️

Name ▾	Subnet ID ▾	IPv4 CIDR ▾	IPv6 CIDR ▾
PrivateSubnet	subnet-029c91c96743d...	10.0.2.0/24	–

Subnets without explicit associations (0) Edit subnet associations

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

< 1 > ⚙️

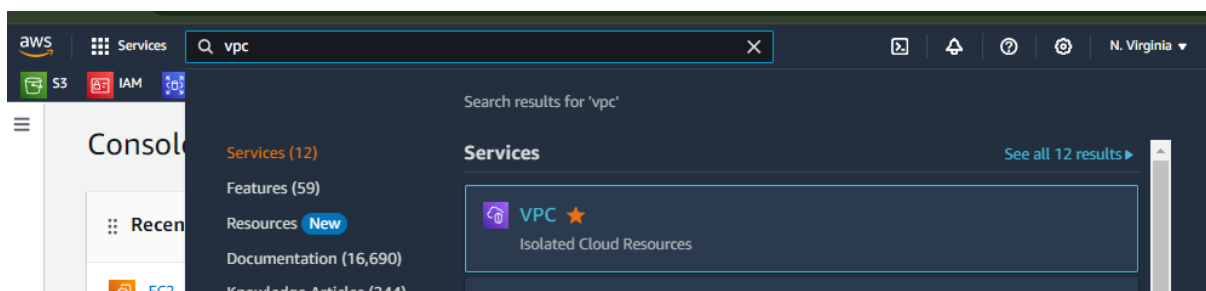
Name ▾	Subnet ID ▾	IPv4 CIDR ▾	IPv6 CIDR ▾
<p>No subnets without explicit associations</p> <p>All your subnets are associated with a route table.</p>			

Lab Steps

Setup Network Configurations

1. Create a VPC:

- Go to the VPC dashboard in the AWS Management Console.



- Click on Create VPC.

- Resource to create: VPC only
- Name: MyVPC
- CIDR block: 10.0.0.0/16
- Tenancy: Default
- Click Create.

[VPC](#) > [Your VPCs](#) > Create VPC

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.

MyVPC

IPv4 CIDR block [Info](#)

☒ IPv4 CIDR manual input
☐ IPAM-allocated IPv4 CIDR block

IPv4 CIDR

10.0.0.0/16

CIDR block size must be between /16 and /28.

IPv6 CIDR block [Info](#)

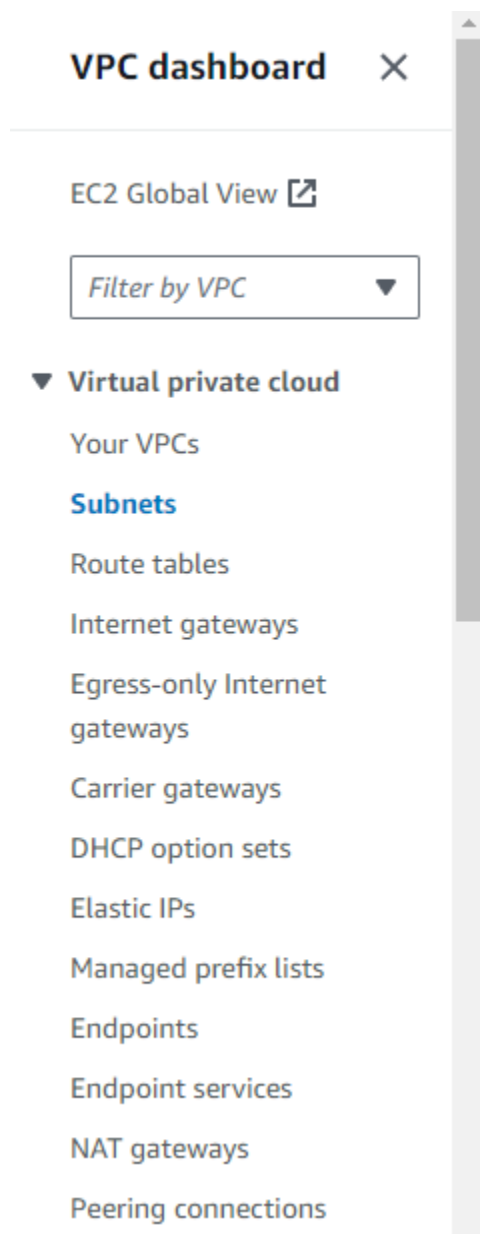
☒ No IPv6 CIDR block
☐ IPAM-allocated IPv6 CIDR block
☐ Amazon-provided IPv6 CIDR block
☐ IPv6 CIDR owned by me

Tenancy [Info](#)

Default ▼

2. Create Subnets:

- Navigate to Subnets in th VPC dashboard.



- Create two subnets with the following configurations:
 - **Public Subnet:**
 - VPC ID: MyVPC
 - Subnet name: PublicSubnet
 - Availability Zone: Select one from the list (e.g. us-east-1a)
 - IPv4 subnet CIDR block: 10.0.1.0/24

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.

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.

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.

IPv4 subnet CIDR block

256 IPs

< > ^ v

▼ Tags - optional

Key

X

Value - optional

X

Remove

Add new tag

You can add 49 more tags.

Remove

Add new subnet

Cancel

Create subnet

- - - Add new subnet
 - Private Subnet: (Click on Add new subnet)
 - Name: PrivateSubnet
 - IPv4 subnet CIDR block: 10.0.2.0/24
 - Availability Zone: Same as the public subnet
 - Click Create.

Subnet 2 of 2

Subnet name

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

PrivateSubnet

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.

US East (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.

10.0.0.0/16

IPv4 subnet CIDR block

10.0.2.0/24256 IPs

< > ^ v

▼ Tags - optional

Key

Value - optional

Q NameX

Q PrivateSubnetX

Remove

Add new tag

You can add 49 more tags.

Remove

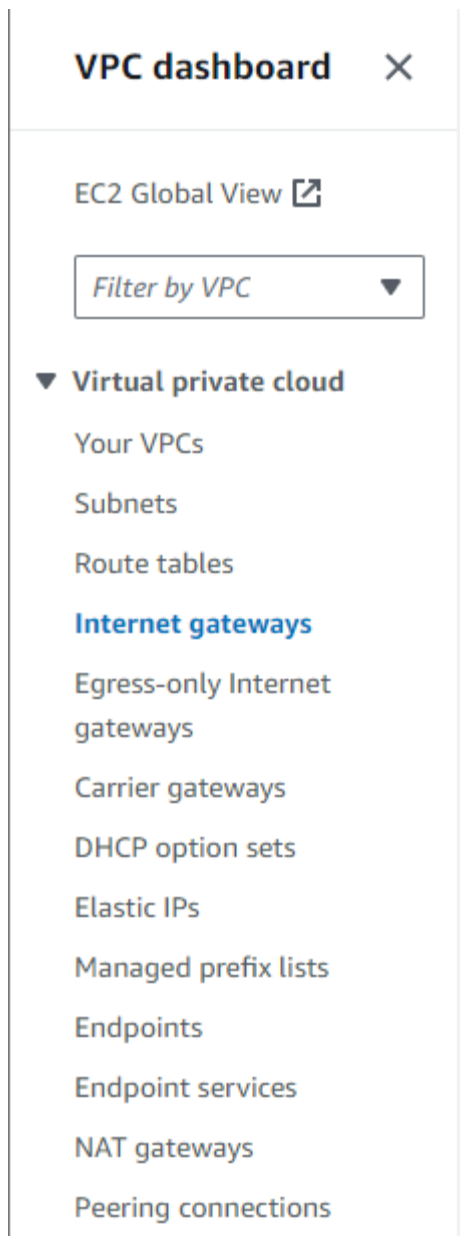
Add new subnet

Cancel

Create subnet

3. Create an Internet Gateway:

- Go to Internet Gateways in the VPC dashboard.



- Click **Create internet gateway**.
- Name: MyIGW
- Click Create.
- Attach the Internet Gateway to MyVPC.

Internet gateway igw-0f3a08b006ab1ca9c successfully attached to vpc-0aa6a3da4f569b288

Notifications 0 0 2 0 0

VPC > Internet gateways > igw-0f3a08b006ab1ca9c

igw-0f3a08b006ab1ca9c / MyIGW

Actions

Details Info

Internet gateway ID
igw-0f3a08b006ab1ca9c

State
Attached

VPC ID
vpc-0aa6a3da4f569b288
MyVPC

Owner
767398024881

Tags

Manage tags

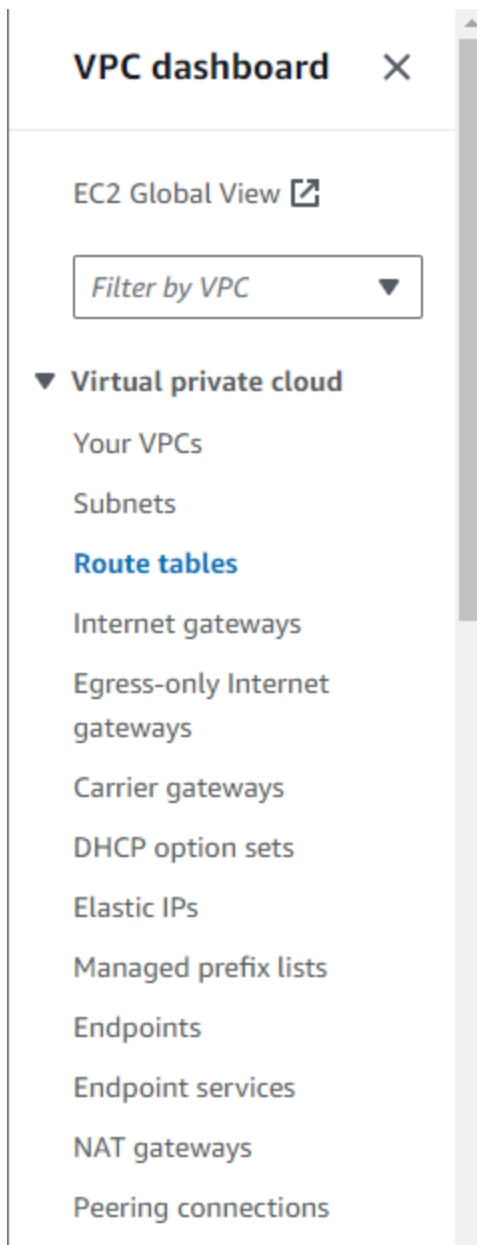
Search tags

Key	Value
Name	MyIGW

4. Create a new Route Table for the Public Subnet

Take note that there should be a new Route Table created after creating a new VPC.

- Go to the Route Tables section in the VPC dashboard



- Click **Create route table**
- Name: PublicRT
- VPC: MyVPC
- Click **Create route table**
- Edit routes: Add route 0.0.0.0/0 to target MyIGW.
- Save changes.

Edit routes

Route 1

Destination
10.0.0.0/16

Target

local

Status

✓ Active

Q local

Propagated
No

Route 2

Destination

Q 0.0.0.0/0

Target

Internet Gateway

Status

-

Q igw-0f3a08b006ab1ca9c

Use: "igw-0f3a08b006ab1ca9c"

igw-0f3a08b006ab1ca9c (MyIGW)

Propagated
No

Remove

Add route

Cancel

Preview

Save changes

- Edit Explicit subnet associations in the **Subnet association tab**: Associate PublicSubnet.

You have successfully updated subnet associations for rtb-0a63b657993e2ddd6 / PublicRC.

VPC > Route tables > rtb-0a63b657993e2ddd6

rtb-0a63b657993e2ddd6 / PublicRT

Actions ▾

Details Info

Route table ID rtb-0a63b657993e2ddd6	Main No	Explicit subnet associations subnet-01d16a350761fd9e8 / PublicSubnet	Edge associations -
VPC vpc-0aa6a3da4f569b288 MyVPC	Owner ID 767398024881		

Routes Subnet associations Edge associations Route propagation Tags

Explicit subnet associations (1) Edit subnet associations

Find subnet association

Name ▾	Subnet ID ▾	IPv4 CIDR ▾	IPv6 CIDR ▾
PublicSubnet	subnet-01d16a350761fd...	10.0.1.0/24	-

Subnets without explicit associations (1) Edit subnet associations

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

Find subnet association

Name ▾	Subnet ID ▾	IPv4 CIDR ▾	IPv6 CIDR ▾
PrivateSubnet	subnet-029c91c96743d4...	10.0.2.0/24	-

5. Create another Route Table for Private Subnet

- Create another route table: Name: PrivateRouteTable, VPC: MyVPC.
- Edit Explicit subnet associations in the **Subnet association tab**: Associate PrivateSubnet.

✓ You have successfully updated subnet associations for rtb-03d4b575529f40fb1 / PrivateRT.

VPC > Route tables > rtb-03d4b575529f40fb1

rtb-03d4b575529f40fb1 / PrivateRT

Actions ▾

Details Info

Route table ID

📄 rtb-03d4b575529f40fb1

VPC

vpc-0aa6a3da4f569b288 | MyVPC

Main

📄 No

Owner ID

📄 767398024881

Explicit subnet associations

subnet-029c91c96743d47fc / PrivateSubnet

Edge associations

–

Routes

Subnet associations

Edge associations

Route propagation

Tags

Explicit subnet associations (1)

Edit subnet associations

🔍 Find subnet association

< 1 > ⚙️

Name ▾

Subnet ID ▾

IPv4 CIDR ▾

IPv6 CIDR ▾

PrivateSubnet

subnet-029c91c96743d...

10.0.2.0/24

–

Subnets without explicit associations (0)

Edit subnet associations

🔍 Find subnet association

< 1 > ⚙️

Name ▾

Subnet ID ▾

IPv4 CIDR ▾

IPv6 CIDR ▾

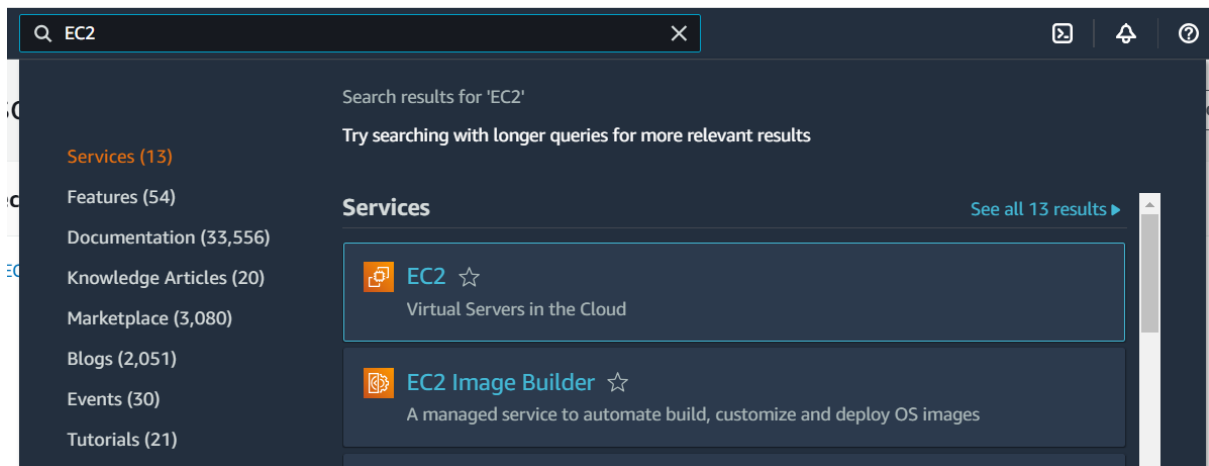
No subnets without explicit associations

All your subnets are associated with a route table.

Launch Instances

1. Launch an EC2 Instance in the Public Subnet:

- Go to the EC2 dashboard.



- Click Launch Instance.
- Name: MyNATInstance
- AMI: **Amazon Linux 2023 AMI**
- Instance Type: t2.micro
- Create key pair: MyKeyPair
- Network Settings:
 - VPC: MyVPC,
 - Subnet: PublicSubnet
- Enable Auto-assign Public IP.

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

MyKeyPair ▼

 [Create new key pair](#)

▼ Network settings [Info](#)

VPC - *required* [Info](#)

vpc-0aa6a3da4f569b288 (MyVPC)
10.0.0.0/16 ▼



Subnet [Info](#)

subnet-01d16a350761fd9e8 PublicSubnet
VPC: vpc-0aa6a3da4f569b288 Owner: 767398024881 Availability Zone: us-east-1a
IP addresses available: 251 CIDR: 10.0.1.0/24 ▼

 [Create new subnet](#) 

Auto-assign public IP [Info](#)

Enable ▼

[Additional charges apply](#) when outside of [free tier allowance](#)

- Firewall (security groups): Select **Create security group**
 - Security group name: MyNATInstanceSG
 - Description – *required*: **Security Group for NATInstance**

Security group name - *required*

MyNATInstanceSG

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and _-./()#,@[]+=&:{}!\$*

Description - *required* [Info](#)

Security Group for NATInstance

- Configure Security Group rules: Allow SSH (port **22**) from your IP and HTTP/HTTPS (port **80**/port **443**) from anywhere.

Inbound Security Group Rules

▼ Security group rule 1 (TCP, 22, 119.111.228.217/32)

Remove

Type	Info	Protocol	Info	Port range	Info
ssh		TCP		22	
Source type	Info	Name	Info	Description - optional Info	
My IP		<input type="text" value="Add CIDR, prefix list or security"/>		<input type="text" value="e.g. SSH for admin desktop"/>	
		<input type="text" value="119.111.228.217/32"/>			

▼ Security group rule 2 (TCP, 80, 0.0.0.0/0)

Remove

Type	Info	Protocol	Info	Port range	Info
HTTP		TCP		80	
Source type	Info	Source	Info	Description - optional Info	
Anywhere		<input type="text" value="Add CIDR, prefix list or security"/>		<input type="text" value="e.g. SSH for admin desktop"/>	
		<input type="text" value="0.0.0.0/0"/>			

▼ Security group rule 3 (TCP, 443, 0.0.0.0/0)

Remove

Type	Info	Protocol	Info	Port range	Info
HTTPS		TCP		443	
Source type	Info	Source	Info	Description - optional Info	
Anywhere		<input type="text" value="Add CIDR, prefix list or security"/>		<input type="text" value="e.g. SSH for admin desktop"/>	
		<input type="text" value="0.0.0.0/0"/>			

- Review and Launch.

2. Launch another EC2 Instance in the Private Subnet:

- Name: MyPrivateInstance
- Follow the same steps, but
 - select PrivateSubnet
 - Key pair: use the same key pair MyKeyPair
 - Network Settings:
 - VPC: MyVPC
 - Subnet: PrivateSubnet
 - Disable Auto-assign Public IP.

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

MyKeyPair ▼

 [Create new key pair](#)

▼ Network settings [Info](#)

VPC - *required* | [Info](#)


vpc-0aa6a3da4f569b288 (MyVPC)
10.0.0.0/16 ▼



Subnet | [Info](#)

subnet-029c91c96743d47fc PrivateSubnet
VPC: vpc-0aa6a3da4f569b288 Owner: 767398024881 Availability Zone: us-east-1a
IP addresses available: 251 CIDR: 10.0.2.0/24 ▼



[Create new subnet](#) 

Auto-assign public IP | [Info](#)

Disable ▼

- Firewall (security groups): Select **Create security group**
 - Security group name: MyPrivateInstanceSG
 - Description – *required*: **Security group for Private Instance**

Security group name - *required*

MyPrivateInstanceSG

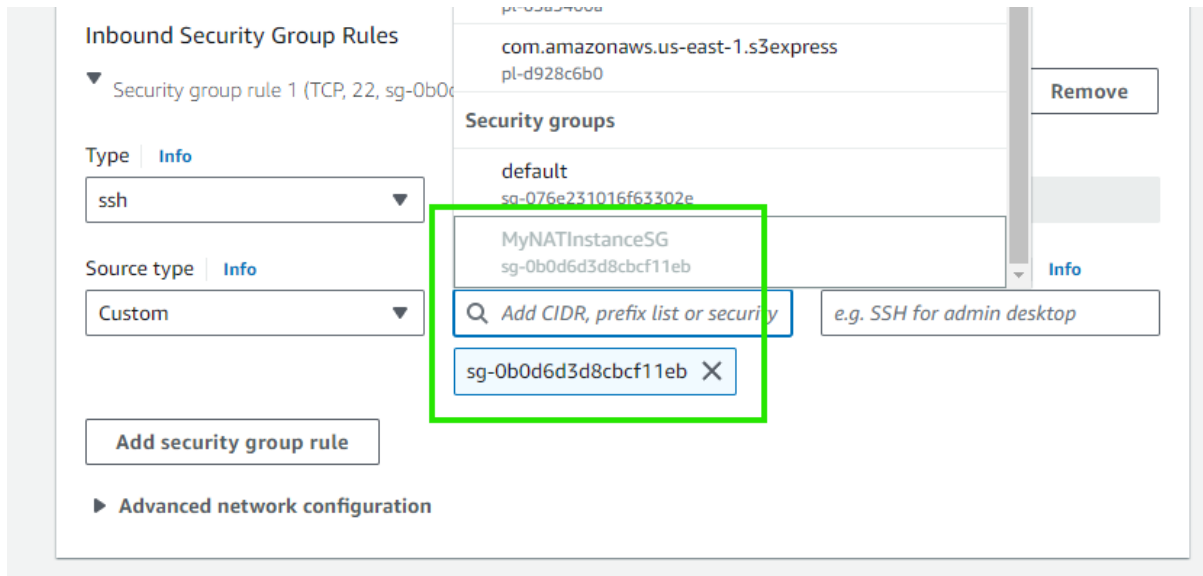
This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and ._-:/()#,@[]+=&:{}!\$*

Description - *required* | [Info](#)

Security group for the Private Instance

- Inbound Security Group Rules: Allow SSH (port 22) from MyNATInstanceSG.

You can also enter the IP subnet CIDR block of the PublicSubnet as an alternative here

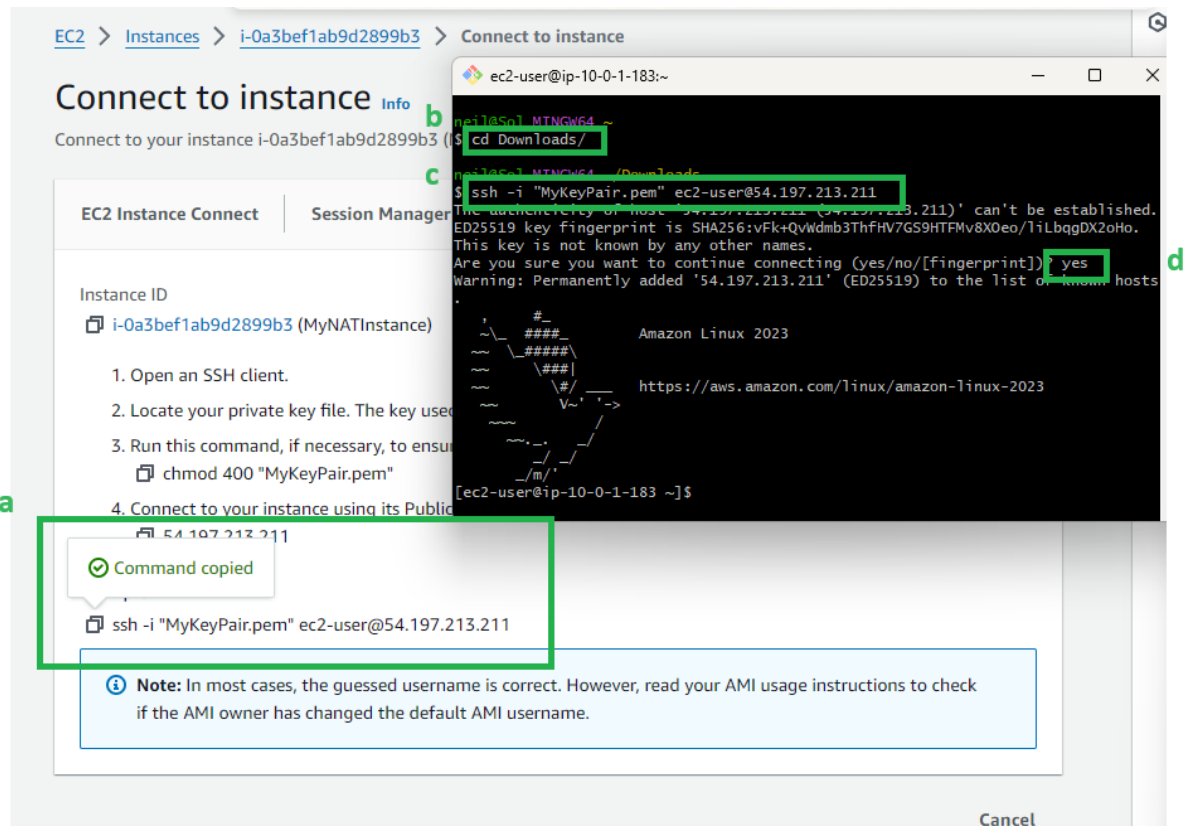


- Review and Launch.

Configure the NAT Instance

1. SSH into the MyNATInstance using your terminal (e.g., Git Bash, PuTTY).

- In AWS console copy the SSH command from EC2 DASHBOARD > select the MyNATInstance > Click on **Connect**.
- Go to your terminal, navigate to your directory where your .pem key is located using the cd command. (For example, in the Downloads/ folder)
- Paste the command into your terminal and press ENTER.
- A prompt confirmation will pop to the terminal, answer yes



- Run the following commands to enable IP forwarding and configure NAT:

- Create a file using vi or vim:

```
sudo vi /etc/sysctl.d/custom-ip-forwarding.conf
```

-

- Next, paste the following:

```
net.ipv4.ip_forward=1
```

-

- Save and exit using the `:wq!` command!

- Then, run the command:

```
sudo sysctl -p /etc/sysctl.d/custom-ip-forwarding.conf
```

```
[ec2-user@ip-10-0-1-183 ~]$ sudo sysctl -p /etc/sysctl.d/custom-ip-forwarding.conf
net.ipv4.ip_forward = 1
```

-

- Next, install the iptables using the command:

```
sudo yum install iptables-services -y
```

```
[ec2-user@ip-10-0-1-183 ~]$ sudo yum install iptables-services -y
Last metadata expiration check: 0:27:01 ago on Tue Aug 6 02:18:11 2024.
Dependencies resolved.
```

Package	Arch	Version	Repository	Size
Installing:				
iptables-services	noarch	1.8.8-3.amzn2023.0.2	amazonlinux	18 k
Installing dependencies:				
iptables-libs	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	401 k
iptables-nft	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	183 k
iptables-utils	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	43 k
libnetfilter_conntrack	x86_64	1.0.8-2.amzn2023.0.2	amazonlinux	58 k
libnfnetlink	x86_64	1.0.1-19.amzn2023.0.2	amazonlinux	30 k
libnftnl	x86_64	1.2.2-2.amzn2023.0.2	amazonlinux	84 k

```
Transaction Summary
Install 7 Packages

Total download size: 816 k
Installed size: 2.9 M
Downloading Packages:
(1/7): iptables-libs-1.8.8-3.amzn2023.0.2.x86_6 5.7 MB/s | 401 kB 00:00
(2/7): iptables-services-1.8.8-3.amzn2023.0.2.n 254 kB/s | 18 kB 00:00
(3/7): iptables-nft-1.8.8-3.amzn2023.0.2.x86_64 2.2 MB/s | 183 kB 00:00
(4/7): iptables-utils-1.8.8-3.amzn2023.0.2.x86_ 2.4 MB/s | 43 kB 00:00
(5/7): libnetfilter_conntrack-1.0.8-2.amzn2023. 2.2 MB/s | 58 kB 00:00
(6/7): libnfnetlink-1.0.1-19.amzn2023.0.2.x86_6 1.3 MB/s | 30 kB 00:00
(7/7): libnftnl-1.2.2-2.amzn2023.0.2.x86_64.rpm 3.7 MB/s | 84 kB 00:00
-----
Total                                         4.7 MB/s | 816 kB 00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing                : 1/1
  Installing               : libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64 1/7
  Installing               : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64 2/7
  Installing               : iptables-libs-1.8.8-3.amzn2023.0.2.x86_64 3/7
  Installing               : libnftnl-1.2.2-2.amzn2023.0.2.x86_64 4/7
  Installing               : iptables-nft-1.8.8-3.amzn2023.0.2.x86_64 5/7
  Running scriptlet: iptables-nft-1.8.8-3.amzn2023.0.2.x86_64 5/7
  Installing               : iptables-utils-1.8.8-3.amzn2023.0.2.x86_64 6/7
  Installing               : iptables-services-1.8.8-3.amzn2023.0.2.noarch 7/7
  Running scriptlet: iptables-services-1.8.8-3.amzn2023.0.2.noarch 7/7
  Verifying                : iptables-libs-1.8.8-3.amzn2023.0.2.x86_64 1/7
```

- - Check the name of the primary network interface using the command:

netstat -i

```
[ec2-user@ip-10-0-1-183 ~]$ netstat -i
```

Interface	MTU	RX-OK	RX-ERR	RX-DRP	RX-OVR	TX-OK	TX-ERR	TX-DRP	TX-OVR	Flg
enX0	9001	3032	0	0	0	3033	0	0	0	BMRU
lo	65536	12	0	0	0	12	0	0	0	LRU

- Take note of this network Interface and use the command following command:

Do not forget to change the <primary_network_interface> placeholder

```
sudo iptables -t nat -A POSTROUTING -o enX0 -j MASQUERADE
```

```
sudo iptables -A FORWARD -i eth0 -o enX0 -m state --state RELATED,ESTABLISHED -j ACCEPT
```

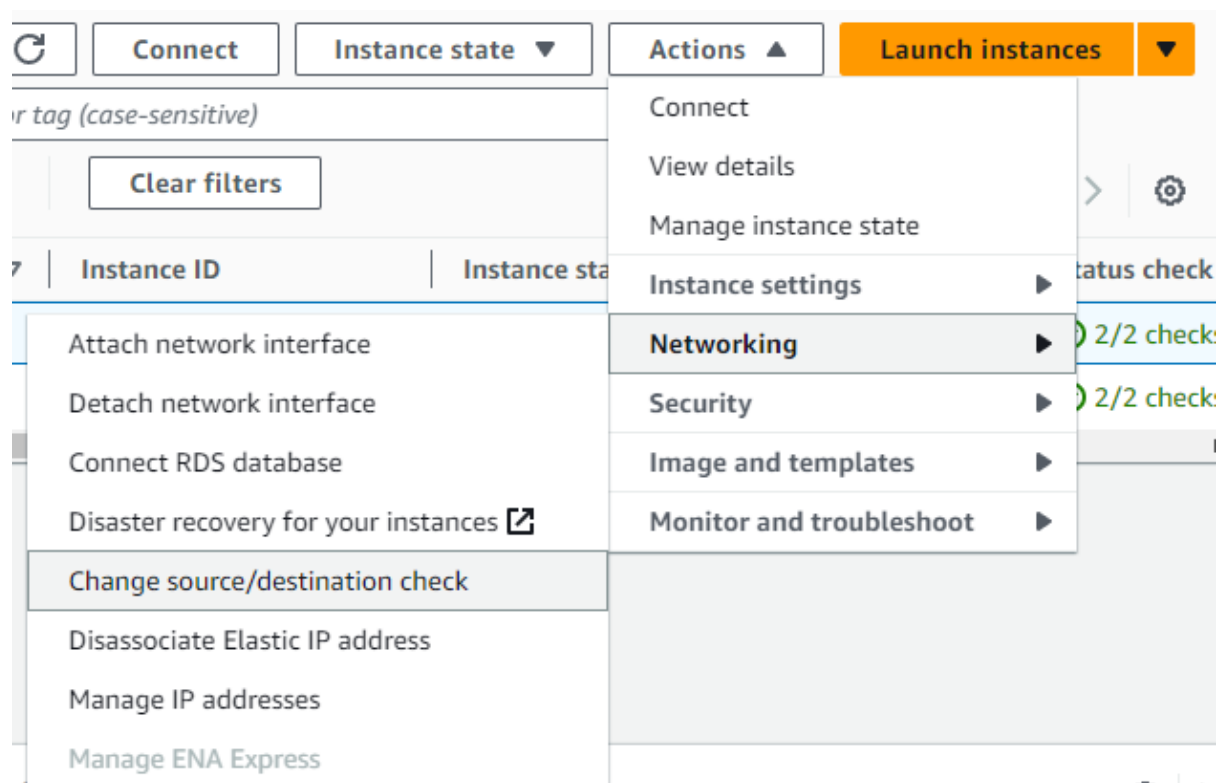
```
sudo iptables -A FORWARD -i eth0 -o enX0 -j ACCEPT
```

```
sudo service iptables save
```

```
sudo service iptables restart
```

```
[ec2-user@ip-10-0-1-183 ~]$ sudo /sbin/iptables -t nat -A POSTROUTING -o enX0 -j MASQUERADE
sudo iptables -A FORWARD -i eth0 -o enX0 -m state --state RELATED,ESTABLISHED -j ACCEPT
sudo iptables -A FORWARD -i eth0 -o enX0 -j ACCEPT
sudo service iptables save
sudo service iptables restart
iptables: Saving firewall rules to /etc/sysconfig/iptables: [ OK ]
Redirecting to /bin/systemctl restart iptables.service
```

2. In the EC2 Dashboard, navigate to your NAT Instance. Click **Actions > Networking > Change source/destination check**



3. Check the checkbox Stop in the **Source / destination checking**

Source / destination checking
Stop to allow your instance to send and receive traffic when the source or destination is not itself.

☒ Stop

Cancel Save

4. Update the Route Table of the Private Subnet, Destination: 0.0.0.0/0 and the Target: Instance (MyNATInstance)

Routes

Subnet associations

Edge associations

Route propagation

Tags

Routes (2)

Both

Edit routes

Filter routes

< 1 >

Destination	Target	Status	Propagated
0.0.0.0/0	eni-084b95...	Active	No
10.0.0.0/16	local	Active	No

5. Update also the Security for the NAT Instance, adding the Type: ICMP - IPv4, Source: 10.0.2.0/24 (The IPv4 subnet CIDR range)

Inbound rule 4

Delete

Security group rule ID

Type

Info

Protocol

Info

-

Custom ICMP - IPv4

All

Port range

Info

Source type

Info

Source

Info

All

Custom

×

10.0.2.0/24

×

Description - optional

Info

Add rule

Access the PrivateInstance

1. Firsts, we need to copy the Private Key (MyKeyPair.pem) to the NAT Instance in our local machine by:

- Exit from your current ssh connection with the NAT Instance using the exit command and go to directory of your .pem key first

```
[ec2-user@ip-10-0-1-183 ~]$ exit
logout
Connection to 54.197.213.211 closed.

neil@Sol MINGW64 ~/Downloads
$ |
```

- Copy, edit, and paste the following command accordingly:
(Since we used the same key pair in both instance)

Do not forget to change the placeholder

scp -i <your_NAT_Instance_Key_Pair> <Your_Private_Instance_Key_Pair> ec2-user@<public_IP_address_of_NAT_Instance>:/home/ec2-user/

```
neil@Sol MINGW64 ~/Downloads
$ scp -i MyKeyPair.pem MyKeyPair.pem ec2-user@54.197.213.211:/home/ec2-user/
MyKeyPair.pem 100% 1678 7.1KB/s 00:00
neil@Sol MINGW64 ~/Downloads
$
```

2. SSH to the NATInstance again.

Do not forget to change the placeholder

ssh -i your-key-pair.pem ec2-user@<NATInstance-Public-IP>

```
neil@Sol MINGW64 ~/Downloads
$ ssh -i "MyKeyPair.pem" ec2-user@54.197.213.211
#_
~\  #####_      Amazon Linux 2023
~\  #####\
~\  \###|
~\  \#/  _____
~\  V~'  ' ->      https://aws.amazon.com/linux/amazon-linux-2023
~\  .-.-
~\  -/-
~\  /m/'

Last login: Tue Aug  6 02:59:07 2024 from 119.111.228.217
[ec2-user@ip-10-0-1-183 ~]$
```

3. From the NAT instance, Change the permission of the MyKeyPair.pem:

- Let's check first if the .pem key was successfully copied using the ls or ls -la command:

```
[ec2-user@ip-10-0-1-183 ~]$ ls
MyKeyPair.pem
[ec2-user@ip-10-0-1-183 ~]$
```

- Use the following command to add permission to the key:

sudo chmod 400 MyKeyPair.pem

```
[ec2-user@ip-10-0-1-183 ~]$ sudo chmod 400 MyKeyPair.pem
```

4. SSH into the Private instance:

Do not forget to change the placeholder

ssh -i MyKeyPair.pem ec2-user@<PrivateInstance-Private-IP>

A prompt confirmation will pop to the terminal, answer yes

