## Darshan Nikam Date: 09/03/2024

### **Virtual private cloud (Vpc)**

A Virtual Private Cloud (VPC) in AWS is like having your own private piece of the internet. It's a virtual network that you can set up within AWS to run your applications and services. A VPC allows you to create a space where you can control and manage your own resources.

Within your VPC, you can launch various AWS resources such as virtual servers (EC2 instances), databases (RDS), and storage (S3) in an environment that is isolated from other users' resources. This isolation provides security and privacy for your applications and data.

You have control over who can enter and exit, and you can set up security groups to protect your VPC from outside incoming and outgoing traffic.

VPC needs ways to connect to the internet or other AWS services. You can set up gateways to make these connections safely.

Definition: AWS VPC is a virtual network environment within AWS that allows users to launch and manage AWS resources in a private, isolated section of the cloud.

Key Components

Subnets: Segments of IP addresses within the VPC's CIDR block.

Internet Gateway (IGW): Allows communication between instances in the VPC and the internet.

NAT Gateway/NAT Instance: Provides outbound internet access for instances in private subnets.

Route Tables: Direct traffic between subnets and gateways.

Security Groups: Act as virtual firewalls for the Instance, controlling inbound and outbound traffic.

Network Access Control Lists (NACLs): Provide subnet-level traffic control.

Benefits

Isolation: Each VPC is logically isolated, providing security and privacy.

Customization: Users can define their own IP address ranges, subnets, and routing tables.

Scalability: Easily scale resources up or down within the VPC as needed.

Security: Security Groups and NACLs provide granular control over traffic flow.

Connectivity

Internet Access: Achieved through Internet Gateway or NAT Gateway (NAT For instance not have a Public IP Address)

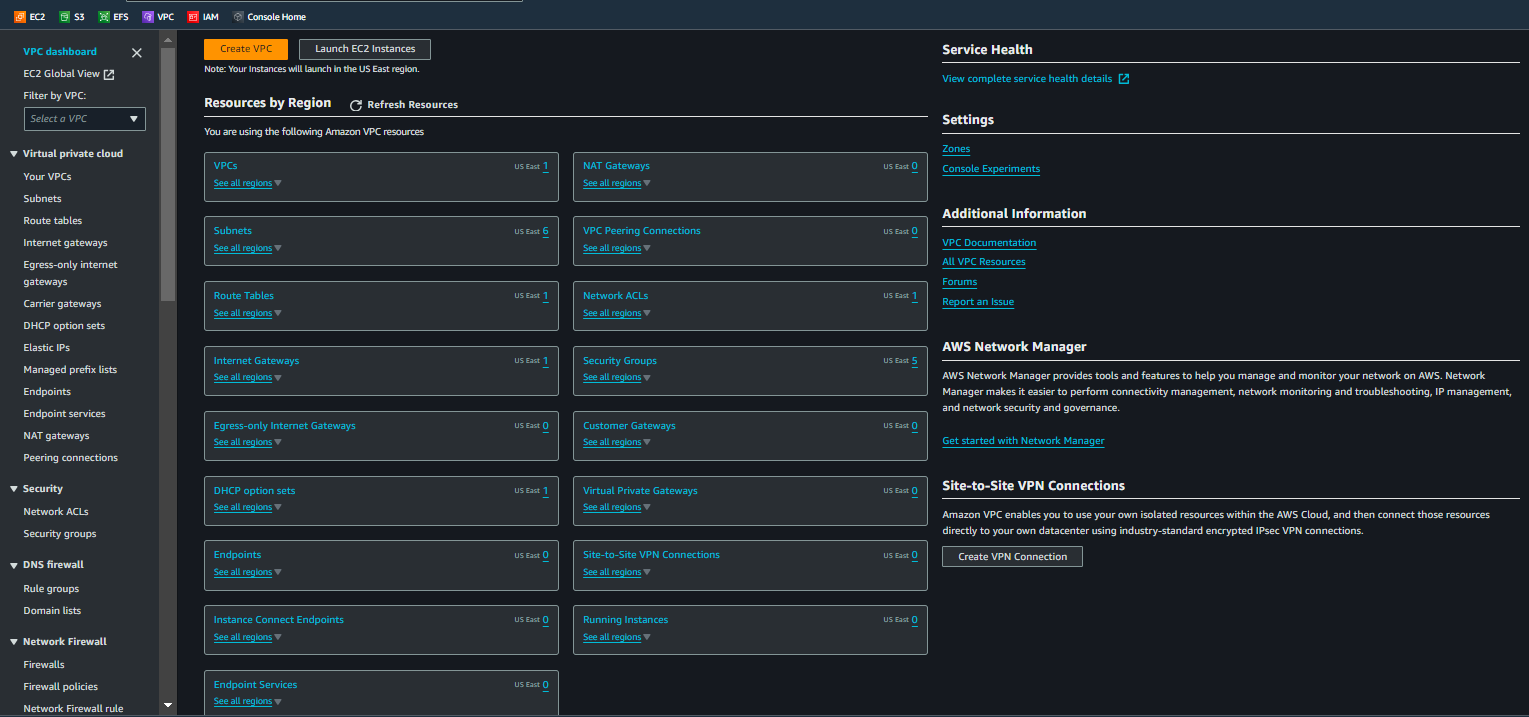
VPC Peering: Connects two VPCs privately to facilitate communication.

VPN Connections: Establish encrypted connections between VPC and on-premises networks.

VPC Endpoints: Privately connect VPC to AWS services without internet access.

Now we create the VPC and launch the instance in it.

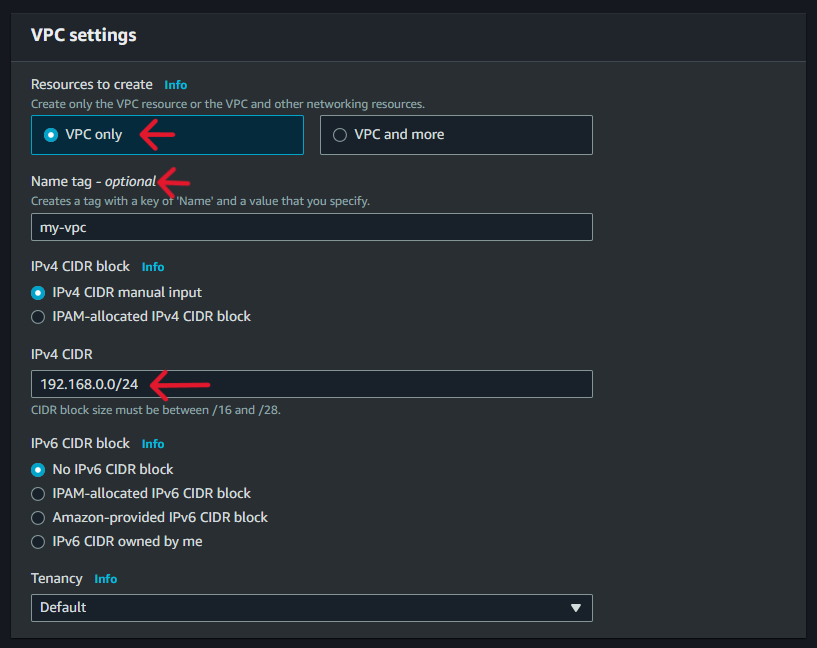
1. Search VPC in the AWS console and Select. This will take you to the VPC dashboard.



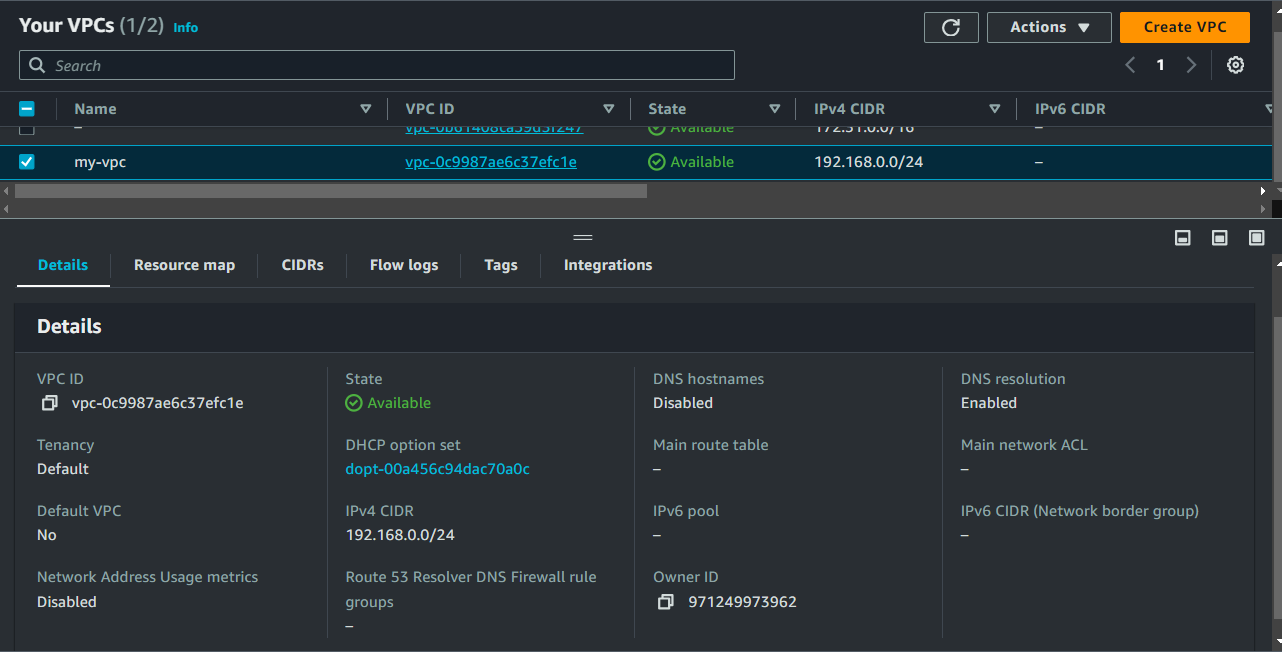
1. Simply click on the Create VPC Button and select the only VPC option.
2. Enter the VPC name that you want, Specify the IPv4 CIDR block to define the IP address range within VPC.

**In AWS we will get CIDR block size between**

**/16 and /28**

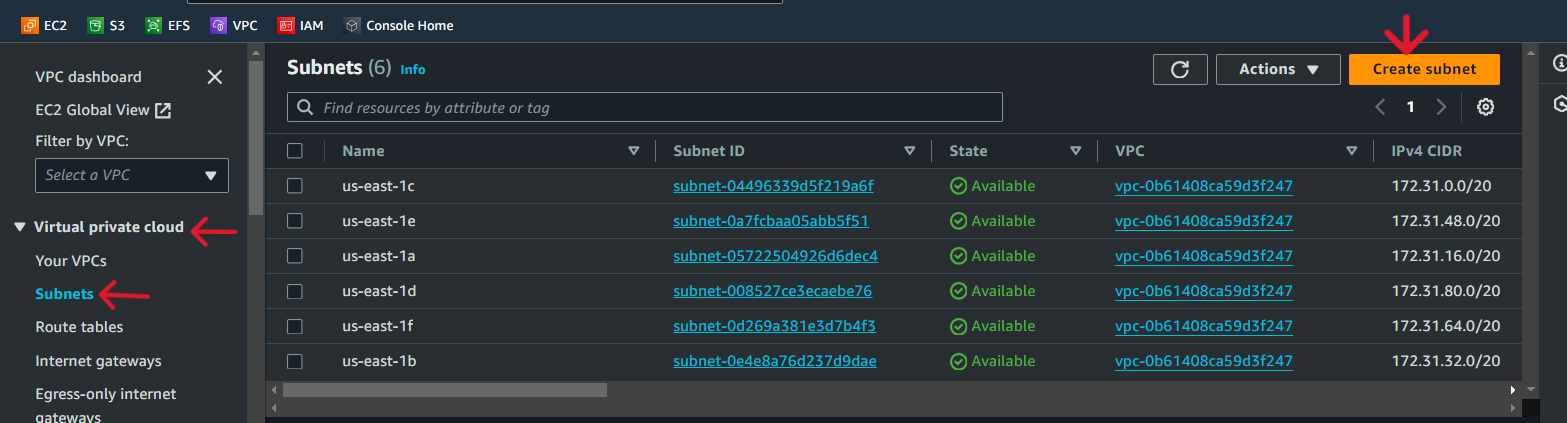


1. Click on the Create VPC button. Your custom VPC will be created.



1. Now we need to create a subnet within our custom VPC, Subnets are segments of the VPC's IP address range.

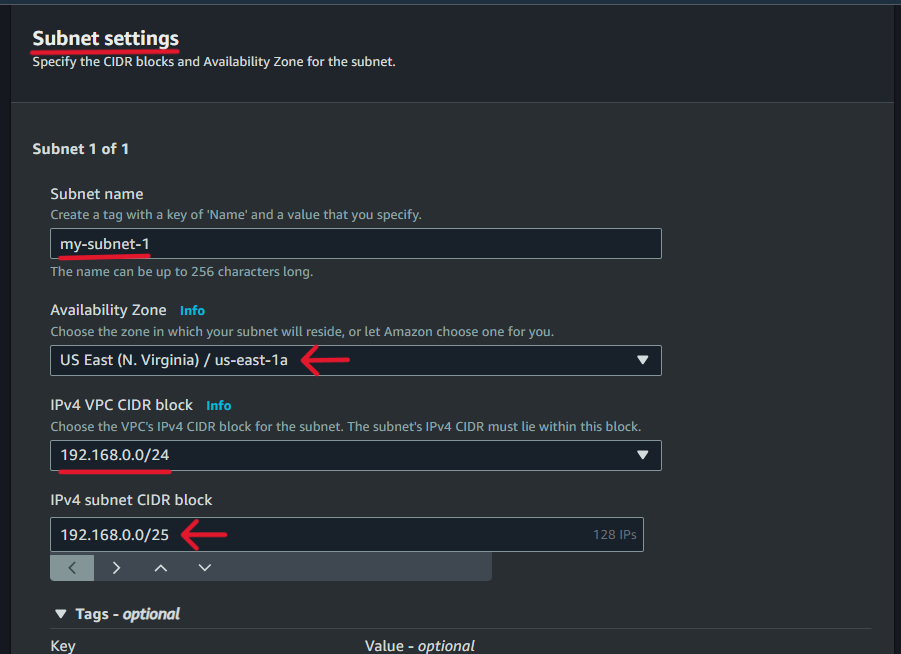
* Select the Subnet option in the VPC Dashboard Menu. Click on the Create Subnet button.



1. Select our custom VPC ID.
2. Enter the Subnet name, select the availability zone, define the CIDR block, and Click the Create Subnet button.

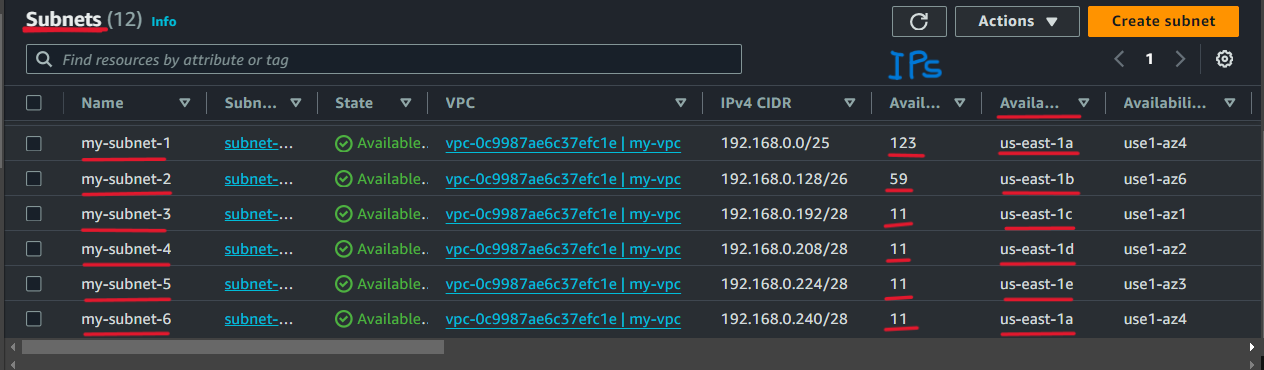
We specified CIDR block 192.168.0.0/24, and its IP range from 192.168.0.0 to 192.168.0.255.

There are a total of 232-24 = 28 = 256 IP addresses in this block



Our first subnet’s CIDR is /25, IPs = 232-25= 27= 128 IPS, so the IP address range is 192.168.0.0 to 192.168.0.127.

Now follow the above step and create more subnets in different availability zones.



In AWS, when you create a subnet within a VPC, AWS reserves 5 IP addresses of each subnet for its own use. Typically, AWS reserves the following five IP addresses within each subnet:

1. The first IP address in a subnet is reserved for the network address.
2. The second IP address is reserved for DNS resolution and other AWS services.
3. The third IP address is reserved for future use by AWS.
4. The fourth IP address is reserved for the VPC router. Its IP address is used as the default gateway for instances within the subnet to route traffic outside the VPC.
5. The fifth IP address is reserved for the broadcast address, although it's not used in AWS.
6. Now we need to set up an Internet Gateway to our custom VPS is crucial for

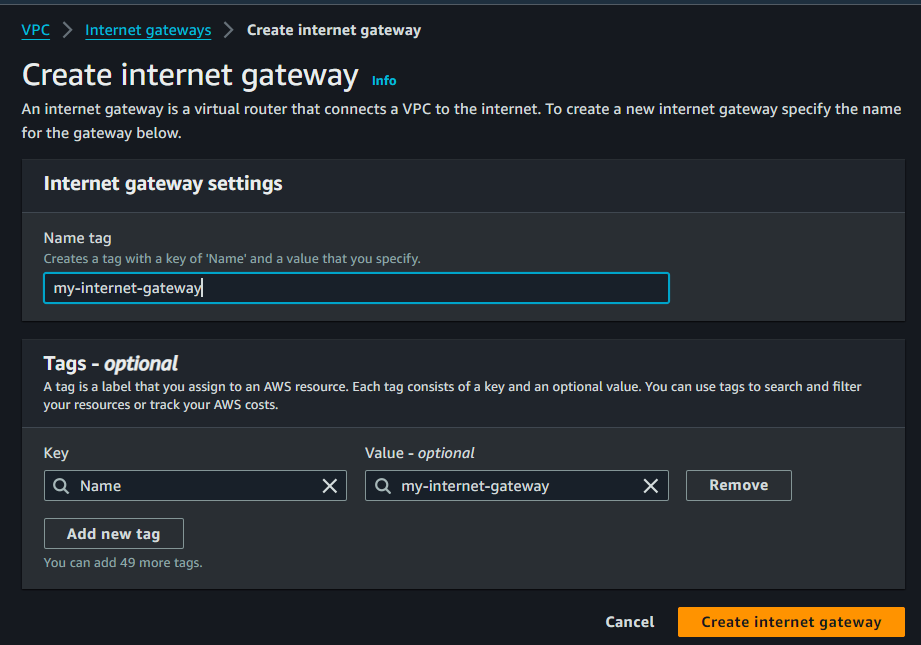
* Internet Access: Enables resources within the VPC to connect to the internet.
* Public Facing Services: Allows hosting of services (e.g., web servers, APIs) accessible from the internet.
* Updates and Patches: Facilitates fetching updates and patches from external repositories.
* Hybrid Cloud Configurations: Enables communication between on-premises infrastructure and cloud resources.
* External APIs and Services: Allows accessing external APIs or services hosted on the internet.
* Outbound Traffic Monitoring and Filtering: Enables security measures such as traffic filtering, monitoring, and logging for outbound traffic.

Steps to create an Internet gateway:

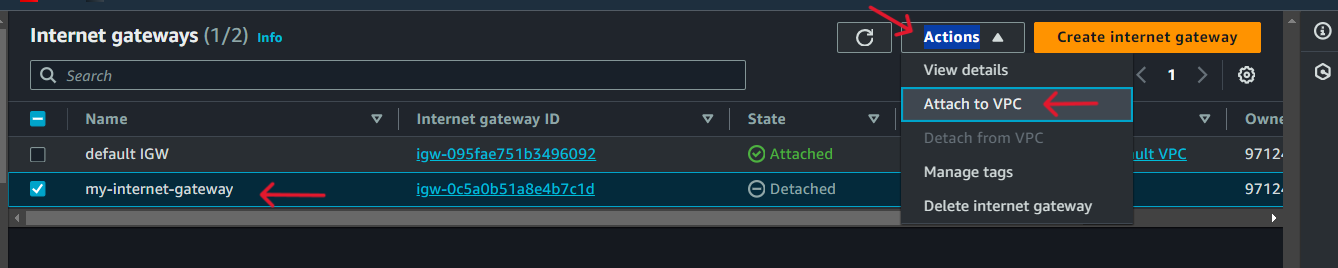
1. In the VPC dashboard select the Internet Gateway option and click on the Create Internet Gateway button.



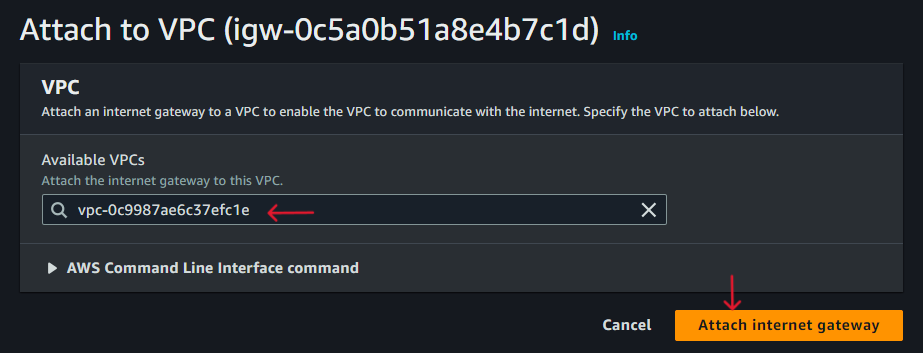
1. Name the Internet gateway, and click on the Create Internet Gateway button.



1. Select Internet Gateway, click on the action button, and select Attach to the VPC.



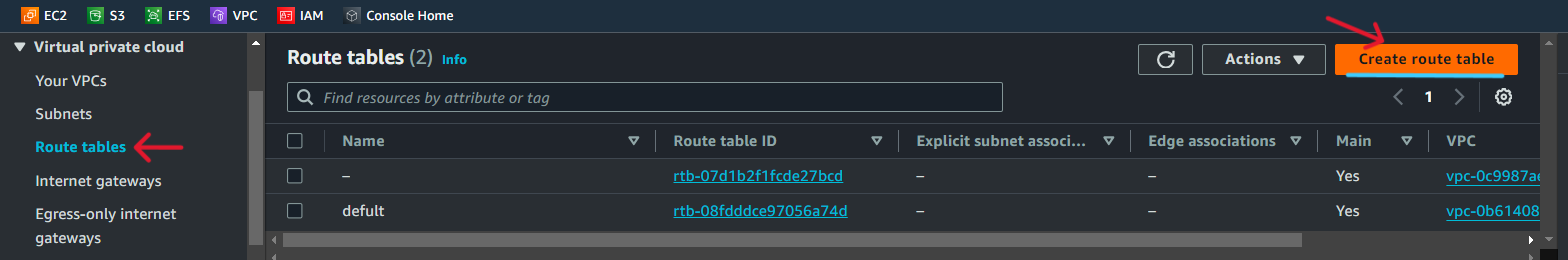
1. Select our custom VPC, and Attach the IGW to the VPC.



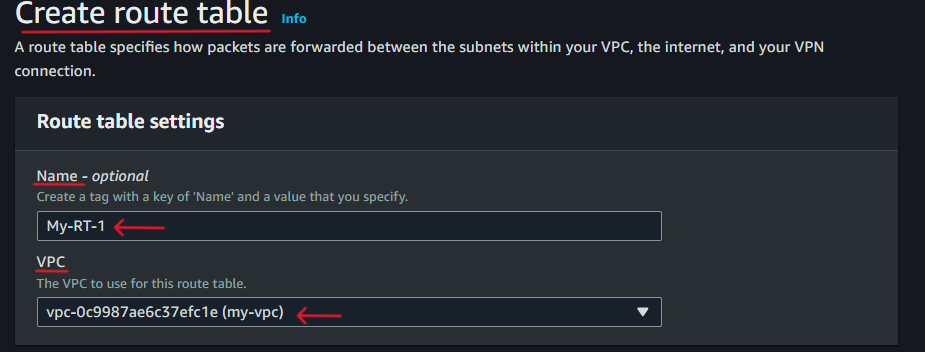
1. Default Route table is automatically created when we create VPC. So we can edit it or create a custom route table.

Creating custom route tables for a VPC is necessary to achieve granular control, define specific routing policies, enable internet and VPN connectivity, accommodate diverse networking requirements, optimize performance, and enhance security within the VPC environment.

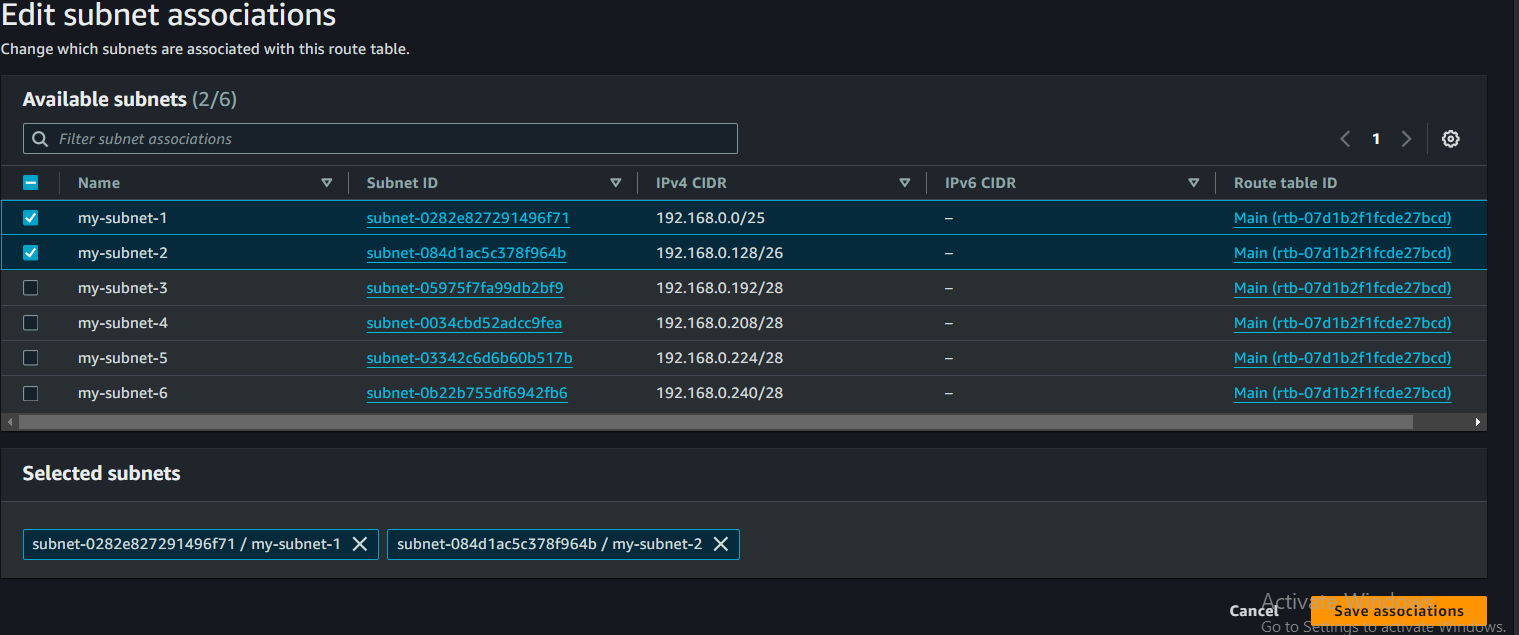
* Select Route Table in VPC Dashboard, and click on the create route table.



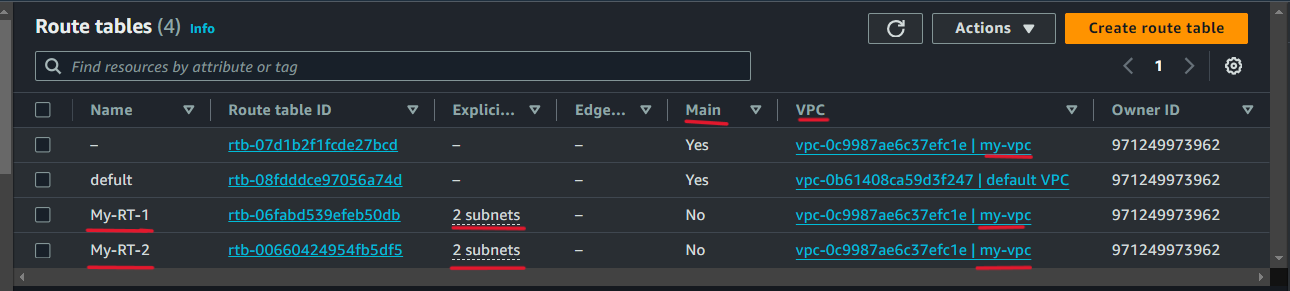
* Name Route Table, select our custom VPC, and click on the Create route table button.



* Select the custom route table, click on the action button then click the Edit Subnet association option.
* Select subnets that you want to associate with this custom route table and click the save association button.

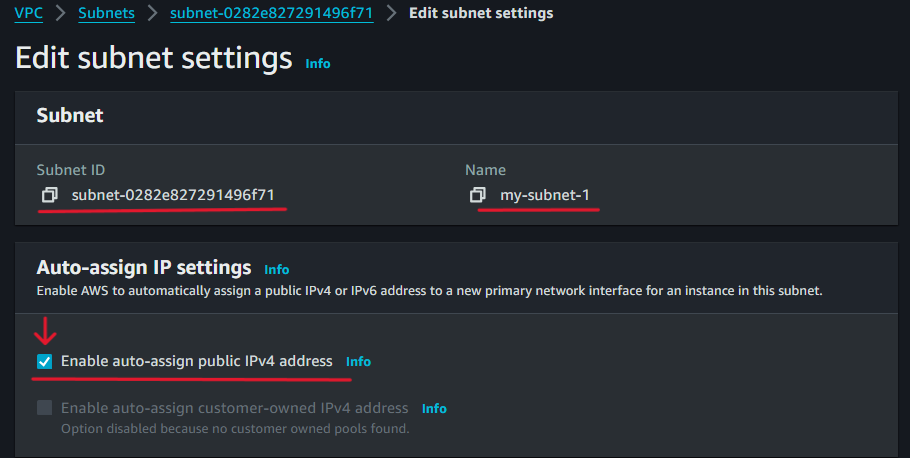


* Now create another route table with associate different subnets.

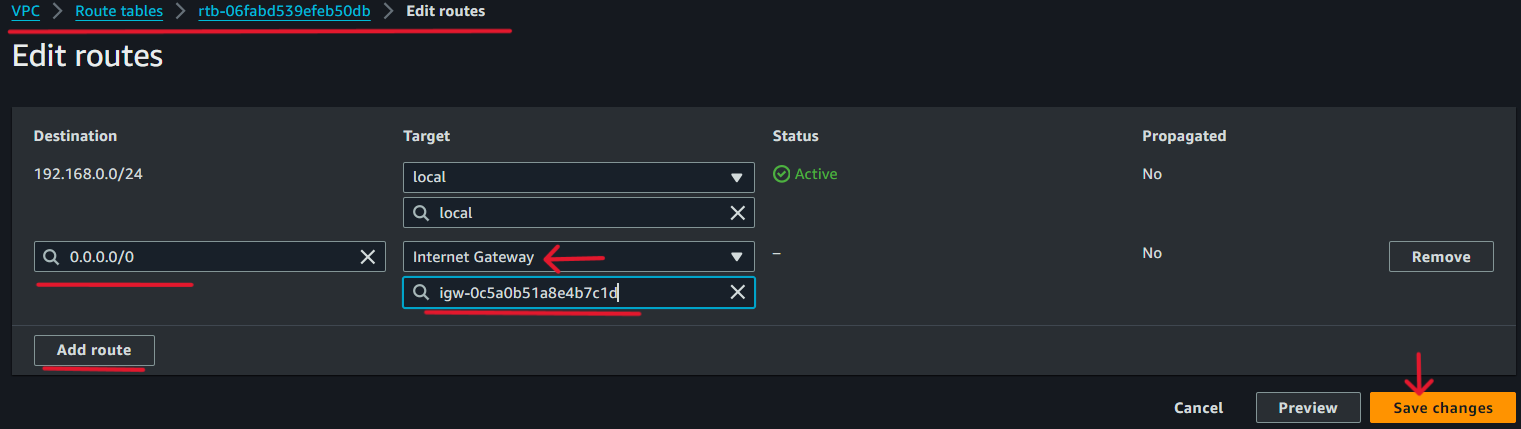


Step 10) Our all subnets are by default disabled to auto-assign public IPv4 addresses by AWS. This means that by default, instances launched in those subnets won't receive public IP addresses. So here we enable this option for subnets under our My-RT-1 route table. (Also there is an option to enable or disable auto-assign public IPv4 address when you launch the instance)

* Go to the subnets menu, select subnet, click on the action button then select the Edit subnet setting option.
* Check the box of Enable Auto-assign IPv4 address, and click on the save button.



Step 11) After creating a route table we need to edit it to add a route for internet access by setting up the target Internet gateway.



Adding destination 0.0.0.0/0 represents the default route. This effectively allows resources within your VPC to access the internet. Any traffic destined for external addresses (including websites, APIs, etc.) will be routed through the Internet Gateway, enabling internet connectivity for your VPC.

Now we have created custom VPC, Subnets within custom VPC, Internet gateway and attached to VPC, then created route table and subnet associated with it, then added the route in route table by setting up target IGW with destination 0.0.0.0/0.

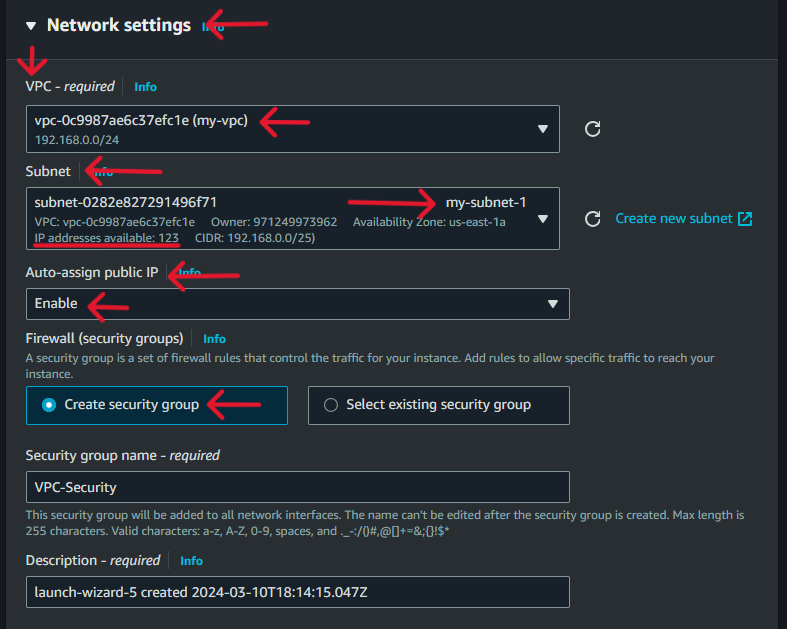
That means we have successfully set up basic network infrastructure within our custom VPC in AWS.

Step 12) Now we launch 2 Instances within our custom VPC. 1st ***with public IPv4 address and 2nd is without Public IPv4 address****.*

|  |  |
| --- | --- |
| ***Instance with Public IPv4 Address:***   * *This instance has a public IPv4 address assigned to it, allowing it to communicate directly with the internet.* * *It can be accessed from the internet using its public IP address.* * *Useful for instances that need to serve content or provide services directly to external clients over the internet.* * *Commonly used for web servers, public-facing APIs, and other services that require direct internet access.* | ***Instance without Public IPv4 Address:***   * *This instance does not have a public IPv4 address assigned to it, making it inaccessible directly from the internet.* * *It can still communicate with other instances within the same VPC or with resources in other VPCs connected via VPN or Direct Connect.* * *Accessible from the internet indirectly via services like* ***NAT Gateway or Load Balancer in the public subnet****.* * *Suitable for instances that do not need direct internet access or are part of an internal network, such as database servers, internal APIs, or backend services.* |

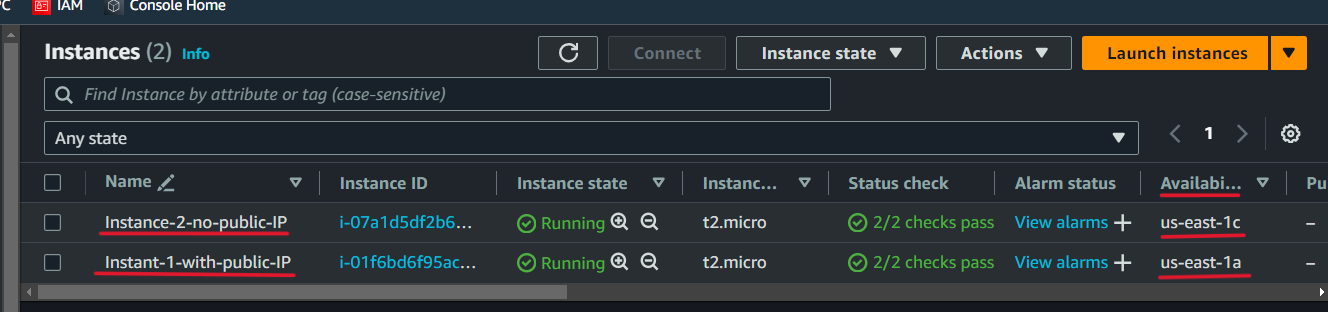
1. *Launch 1st Instance with Public IPv4 Address*

* *At launching the Instance edit the network setting, select our custom VPC, then select my-subnet-1. Ensure that Auto-assign public IP is enabled. Create a security group allowing SSH and HTTP rules.*



1. *Now launch 2nd Instance in a different subnet and disable Auto-assign Public IP Address. In the security group allow SSH and HTTP rules.*

*We will see that both instances have launched.*



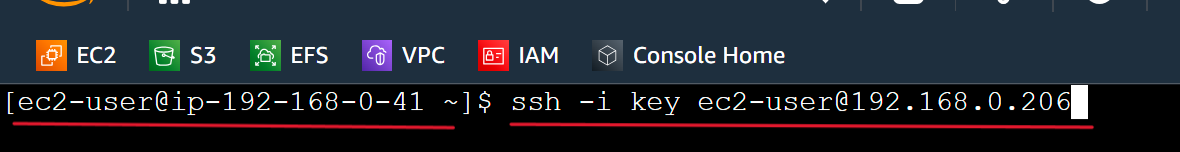
Step 13) Now you can connect to the 1st Instance having a public IP, But the 2nd Instance does not have a Public IP we can’t connect it using EC2 Connect.

To Access 2nd Instance does not have a public by using the following Methods.

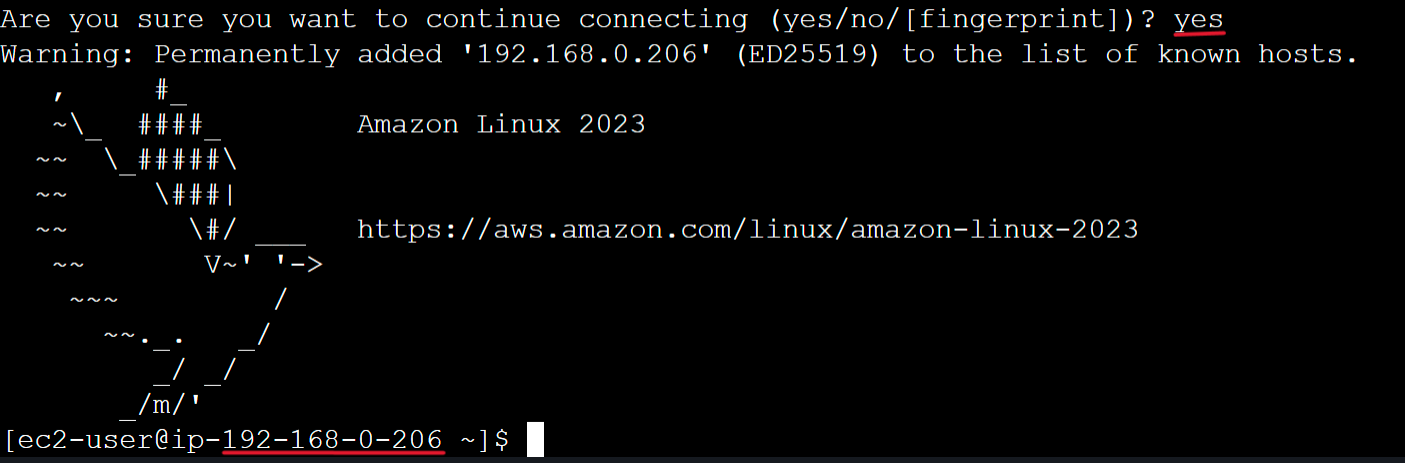
1. Using a Bastion Host or Jump Server: Connect to a separate instance with a public IP address (bastion host or jump server) in a public subnet, then use that instance to SSH into the target instance in the private subnet.
2. Using Elastic IP (EIP): Assign an Elastic IP (EIP) to the instance. This gives it a static public IP address, allowing direct access from the internet. However, this method is less secure because it exposes the instance directly to the internet.
3. Using AWS Systems Manager Session Manager: Use AWS Systems Manager Session Manager, a service that provides secure shell (SSH) access to your instances without the need for a public IP address or opening inbound ports in security groups. This method requires the AWS Systems Manager agent to be installed on the instance.
4. Using VPN or Direct Connect: Connect to your VPC using a VPN connection or AWS Direct Connect, then access the instance via its private IP address.
5. Using NAT Gateway: Configure port forwarding on a NAT Gateway to allow SSH traffic from a specific port on the NAT Gateway's public IP address to the instance's private IP address.
6. Using AWS Client VPN: Set up an AWS Client VPN endpoint to provide secure access to your VPC from remote clients. Once connected, access the instance via its private IP address.

Now we use Bastion Host or Jump Sever Method to access our 2nd Instance does not have a public IP address.

* Connect to the Instance have a public IP address.
* Open the private key file of the 2nd Instance in the notepad and copy all the text from it.
* Go to the connected instance CLI, create a new file, and paste the copied text of the key file into a new file then save it.
* The key is not publicly viewable is necessary. When we have created a key file in our Instance is default permission 644.
* Run the command “chmod 400 key.file.name” to set only read permission for the owner.
* Now copy or 2nd instance Private IPv4 address and run SSH client command on 1st instance CLI.
* **ssh -i key.name ec2-user@192.168.0.206**

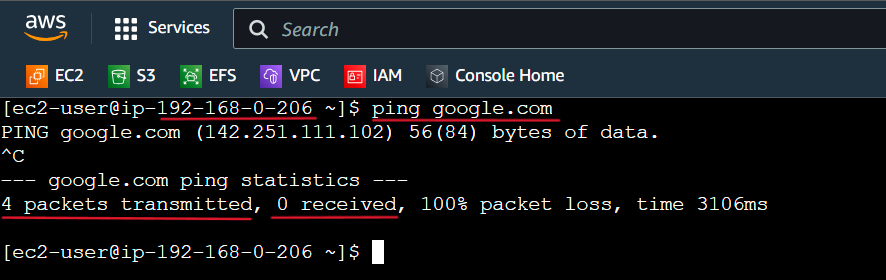
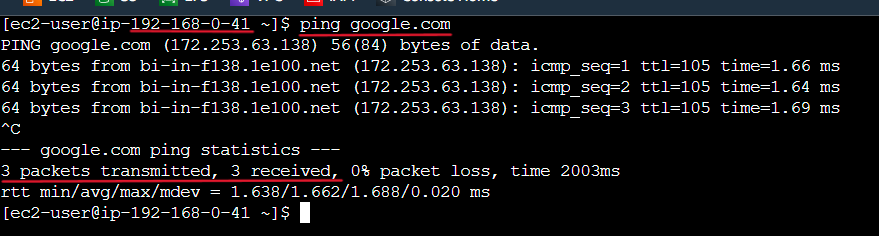


192.168.0.41 is our instance having public IP and we are connecting our accessing to 192.168.0.206 Instance does not have a private IP address. Here we can see we connected to 2nd instance..



Stay connected with it, and open another tab to connect to 1st Instance again. Now we have access of both Instances but only 1st instance can access the internet and 2nd does not. As we can see in bellow Images.

In the above image, 1st instance can packets transmitted and received means accessing the internet



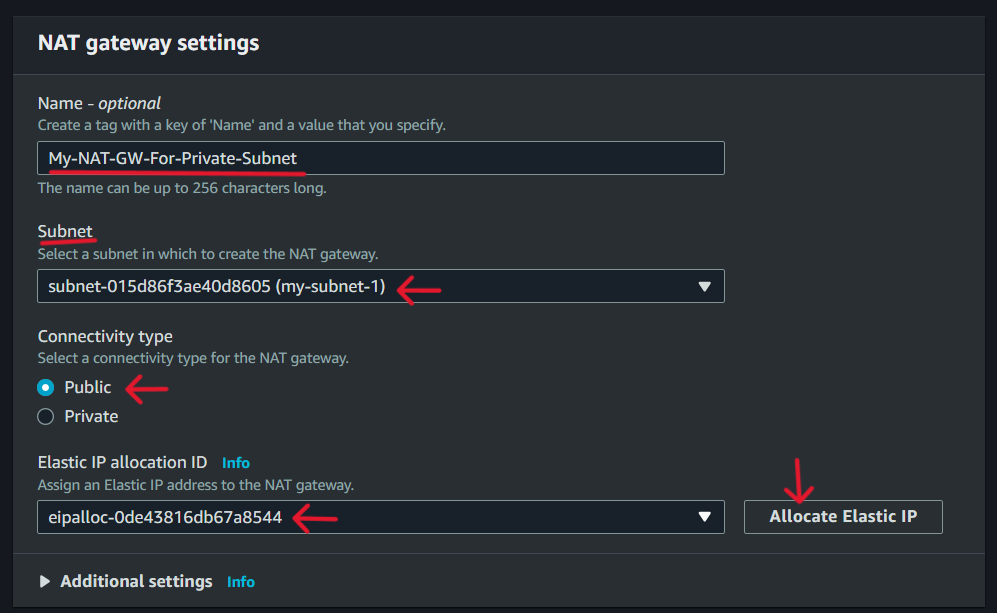
In this 2nd Instance, packets transmitted but not received means not accessing the internet.

This happens because we do not create a NAT gate for the private subnet where we created our 2nd Instance.

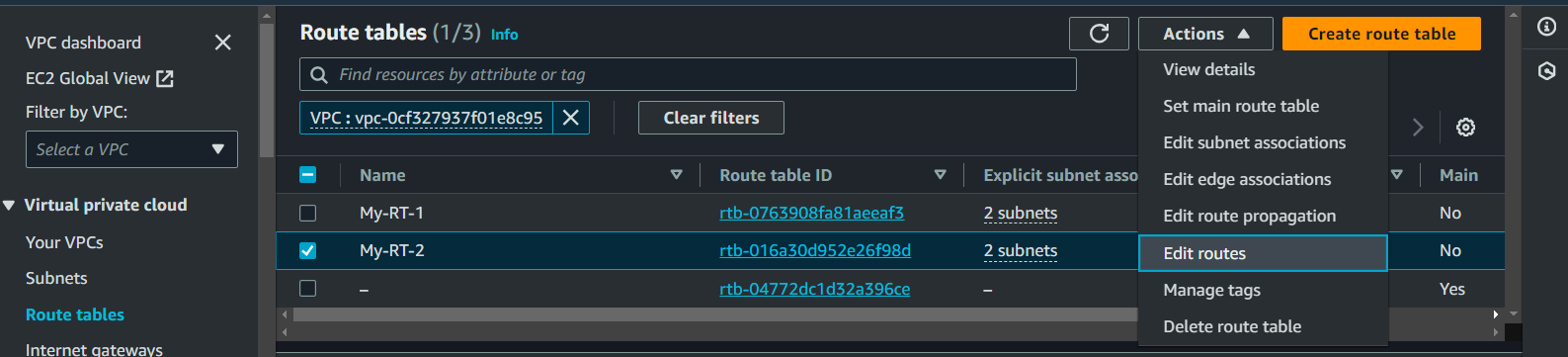
NAT Gateway provides a secure and scalable solution for instances in private subnets to access the internet while maintaining control over inbound traffic and ensuring the privacy and security of the network environment.

Step 14) Create a NAT Gateway.

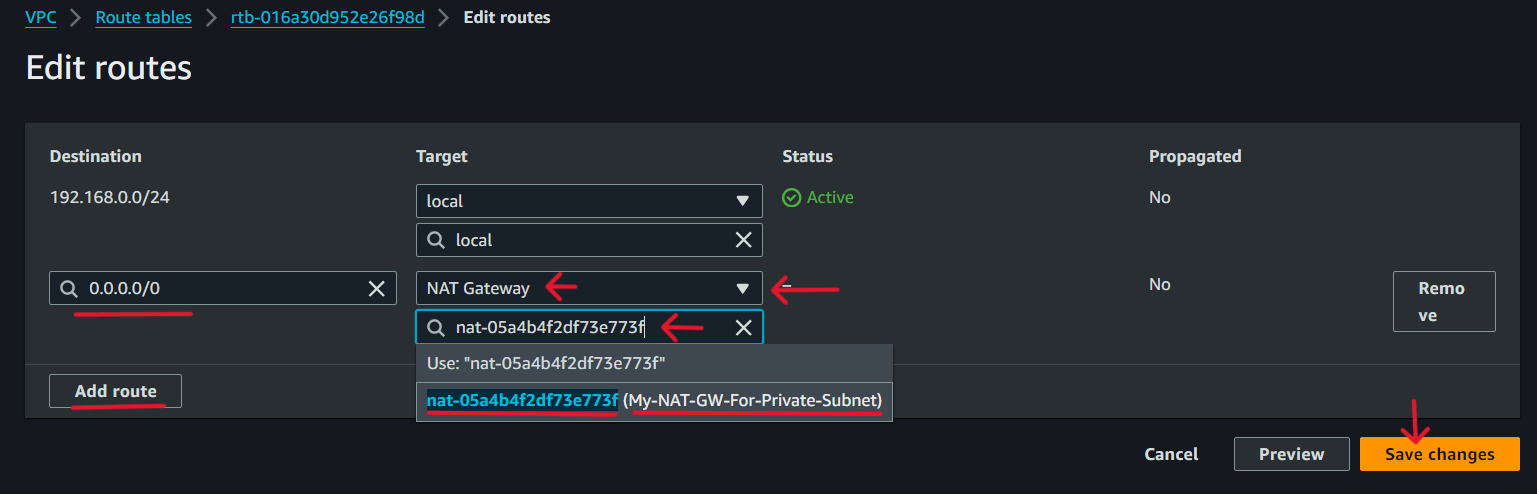
* In the VPC dashboard click on NAT Gateway, then click on the Create NAT gateway button.
* Name the NAT Gateway, select the Public subnet, and select the connectivity type as Public, (Creating a NAT Gateway in a public subnet, it uses the public subnet's routing table to route traffic to the internet. Instances in the private subnet are configured to send their outbound internet-bound traffic to the NAT Gateway)
* Allocate Elastic IP (This EIP provides the public IP address for the NAT Gateway. Placing the NAT Gateway in a public subnet ensures it has access to the internet and can communicate using this public IP address).
* Click on the Create NAT Gateway button.



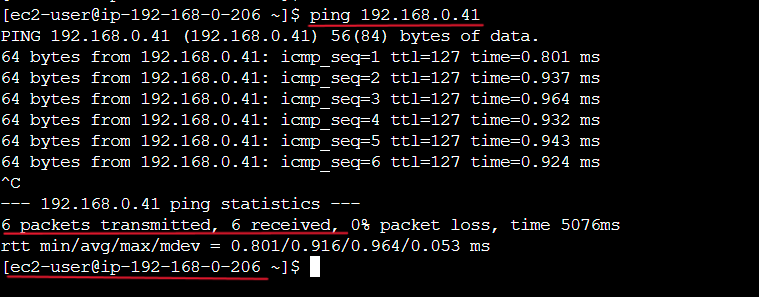
* After creating the NAT gateway we need to add a route into the route table for internet access by setting up the target NAT gateway
* Go to the route table, Select the route table, click on the Action button, then click on the edit route button.



* Click on the Add Route button, enter destination 0.0.0.0/0, select target NAT Gateway, then choose our NAT target, and click on Save Changes.



Now you our 2nd Instance can access the internet. Ping google.com to verify.



We can see it working.

Step 15) Add an ICMP IPv4 rule in the security groups of both Instances.

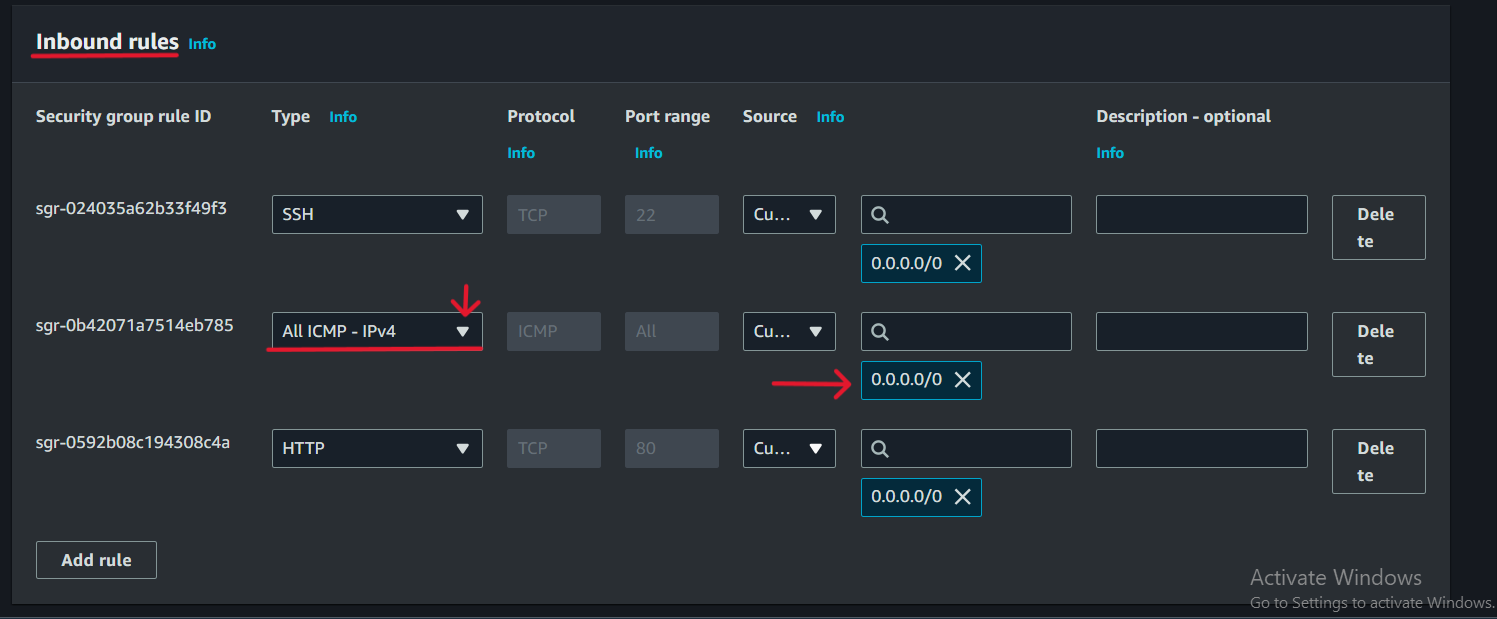
ICMP (Internet Control Message Protocol) is a network protocol used for various network diagnostics and error reporting purposes. It's commonly associated with IPv4, although it's also used in IPv6 networks.

Ping: The most common use of ICMP is for the ping command, which sends ICMP Echo Request packets to a destination host and waits for ICMP Echo Reply packets to confirm connectivity.

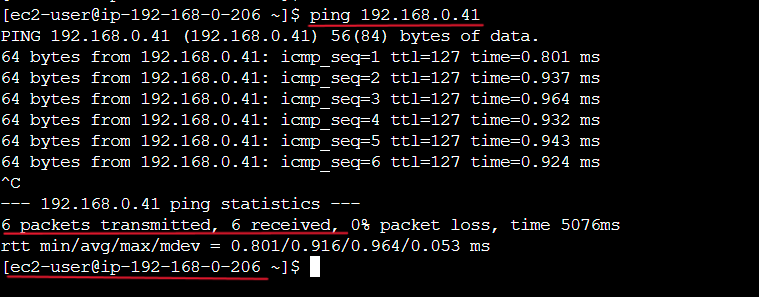
Firewall Considerations: ICMP traffic is often blocked by firewalls for security reasons. However, it's important to allow certain ICMP types, such as Echo Request/Reply, for network troubleshooting and monitoring.

Diagnostic Tool: ICMP is a crucial tool for network administrators for diagnosing network issues, such as determining whether a host is reachable, identifying routing problems, and troubleshooting network congestion.

* Edit Security rule, and add the inbound rule ICMP – Ipv4 and save the changes.



Now we ping a public Instance from a private Instance using its Public IP address.



It is working.

Resource Map of our custom VPC

