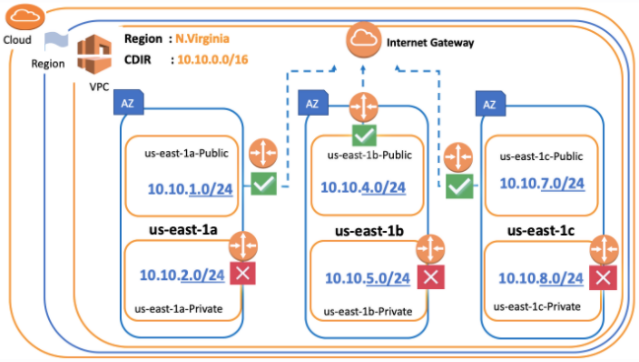
Creating NAT Gateway&NAT Instance

Need of NAT Gateway&NAT Instance



VPC Scenario

Now, before going to NAT Gateways from the VPC menu, Let's look at our scenario.

We created 3 public and 3 private subnets when we created VPCs and defined them by creating subnets.

Virtual machines in **Public Subnet** can go on the internet, and traffic can come to these machines over the internet.

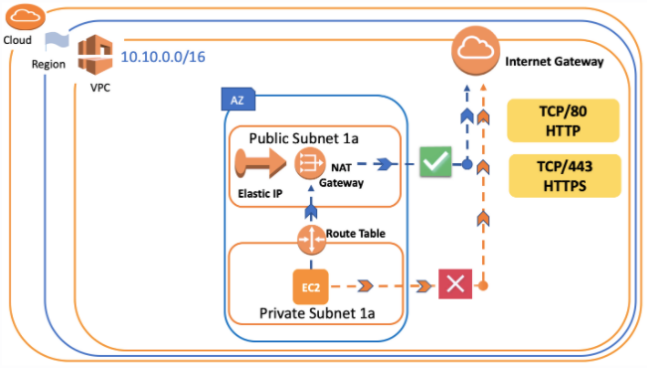
However, instances in **Private Subnet** here are able to communicate only with each other. Neither they can get traffic from the internet nor connect to the internet.

We solved the problem of accessing **from outside to Private instance** that is in with Bastion Host / Jump Box.

However, how can we solve the problem of accessing **from Private instance to outside**?

In some cases, we might want the virtual machines in Private Subnets to access the internet. For example,

- When we want to perform a software update on the virtual machines, we put in Private Subnets or,
- When we want this virtual machine to pull a package from the internet or download a file, etc.



NAT Gateway

In such cases, NAT Gateway or NAT Instance is what we will use to do this as you see in the picture which shows the NAT Gateway workflow.

NAT Gateway Creating Process

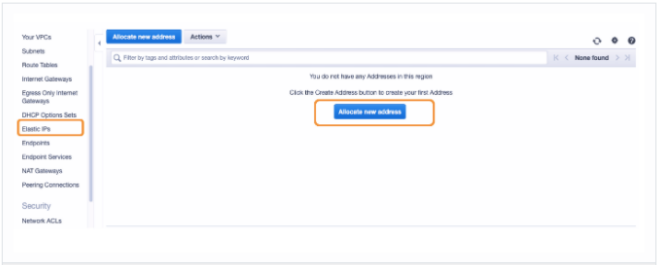
To Create NAT Gateway:

- First Allocate Elastic IP,
- Then create a NAT Gateway and put it in one of the **Public Subnet**,
- After that, go to the **Route Table of Private Subnets** and set the rule to ensure that packages are delivered via **NAT Gateway**.
- Therefore, the machines in Private Subnets can access the internet through these NAT Gateways.

Let's do it on the AWS console.

⚠Avoid ! :
- Since NAT Gateway is not an Internet Gateway, these machines cannot be accessed from the outside world.

Step-1 Allocate Elastic IP for NAT Gateway

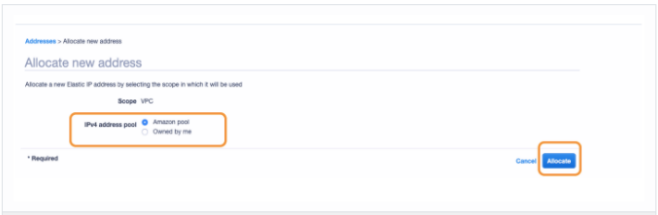


Elastic IP Page

First of all, we need to allocate an Elastic IP before creating a NAT Gateway. Elastic IP is a kind of static IP.

In fact, while creating a NAT Gateway you can also create new Elastic IP but, we prefer to create before to see the process.

- Let's click the Elastic IPs tab from the left-hand menu on VPC service first,
- Then, click the **Allocate New Address** tab on the screen that opens as you see in the picture above,
- After that hit the **Allocate** tab.



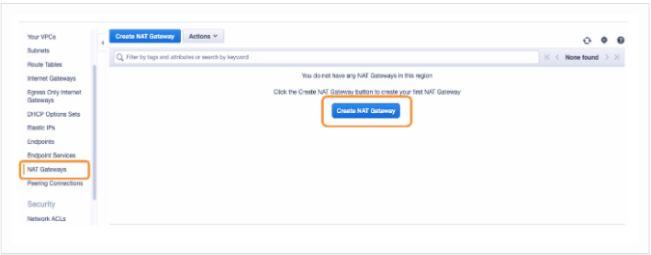
Allocate Elastic IP

- Here we are asked whether we want IPv4 from the **Amazon Pool** or the **Owned by me** option.
- Let's select **Amazon Pool** option because we don't have owned IP right now.
- Then Click **Allocate** and Elastic IP addresses will be created.

Now, as long as we don't delete this IP address here, we can use this IP address as much as we want. We can associate the Elastic IP address to any resources from the **Actions Menu**

⚠Avoid ! :
- Elastic IPs are totally free as long as they are being used by an instance. However, Amazon will charge you $0.01/hr for each EIP that you reserve and do not use. So don't forget to terminate the Elastic IP or associated component such as NAT Gateway if you'll not use anymore in the short term.

Step-2 Create NAT Gateway



NAT Gateway

Let's click on the NAT Gateway tab on the left-hand in the VPC menu and click on the **Create NAT Gateway** tab as you see in the picture above.

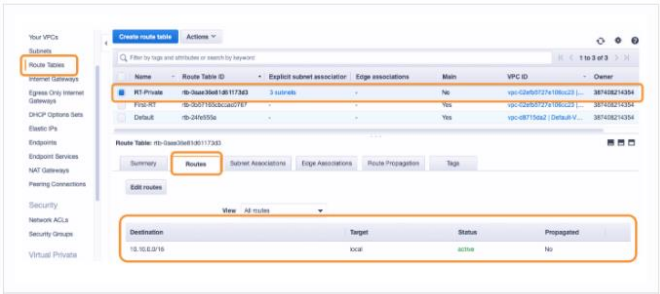
Create NAT Gateway

On the screen that opens:

- First, we select in which Public Subnet we will create. Let's choose **1a- Public Subnet**.
- Then we need to define an Elastic IP address. We select the newly created Elastic IP address, or we can continue by Creating a New Elastic IP address with the **Create New EIP** option.

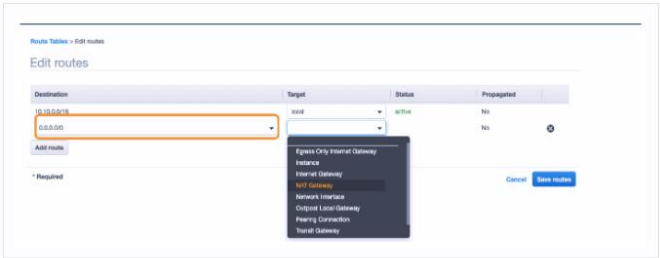
Let's click **Create NAT Gateway** and complete the process.

Step-3 Set the Route Table of Private Subnet



Route Table of Private Subnet

- First, let's move on to the **Route Tables** section on left-hand menu in VPC services.
- Select the **Route Table** we created for the Private Subnets (RT-Private)
- After selecting the **Routes** tab from here, click **Edit Routes**.



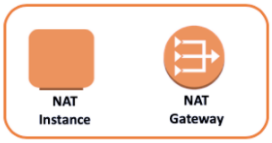
Route Table of Private Subnet

You'll see the page as seen above.

- Just as we did for Public subnets, let's write the IP which indicates **0.0.0.0/0**, which means any IP address, to the **Destination** section.
- In the **Target** section, let's choose **NAT Gateway** and determine the delivery of the package that will be released to the Internet to NAT Gateway.
- Finally, save the process by clicking **Save Routes**

That's it. We have created NAT Gateway and associated with private subnet via Route Table.

NAT Instance Instead of NAT Gateway



NAT Instance vs. NAT Gateway

NAT Gateways are not the only way for our machines in the private subnet to access the internet safely. Instead of NAT Gateway, we can use the **NAT Instance** also.

We prefer NAT Gateways, because NAT Gateways are managed by AWS and we don't get involved in anything.

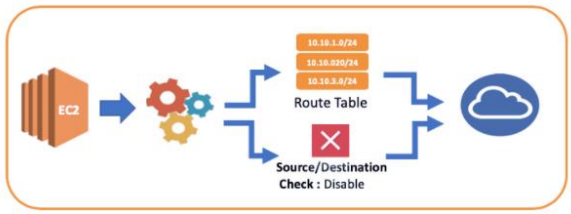
But, if your system consists of hundreds of machines and you need some TCP based adjustments, NAT Gateway may not support your needs.

Therefore, we need to use the NAT Instance instead of NAT Gateway.

Also, we have one more reason to choose NAT Instances. It's the price.

NAT Instances are cheaper than NAT Gateway.

NAT Instance Creating Process



Creating a NAT Instance

For Creating a NAT Instance;

- First, we create an EC2 Instance on Public Subnet,
- Then we set the EC2 Instance via **Route Table** and **Action Menu**
- Finally, we use this NAT Instance for accessing the internet.

Now let's see this as applied on the AWS console.

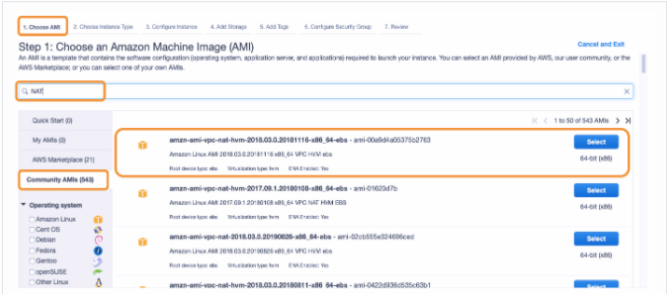
Step-1 Launch an Instance

First, go to EC2 Service and select **Instance** section from the left-hand menu. Then click **Launch Instance** tab to create a new instance.

Step 1: Choose an Amazon Machine Image(AMI):

We can not use standard AMI for the NAT Instance. Thus we need to choose **NAT AMI** of Community as you see in the picture below.

- First click **Community AMIs** tab,
- Then search the Community AMIs tab with the keyword **NAT**.
- Select NAT AMI of Amazon which is at the top of the search results and suitable for us.



Step 1: Choose an Amazon Machine Image(AMI)

Step 2: Choose an Instance Type:

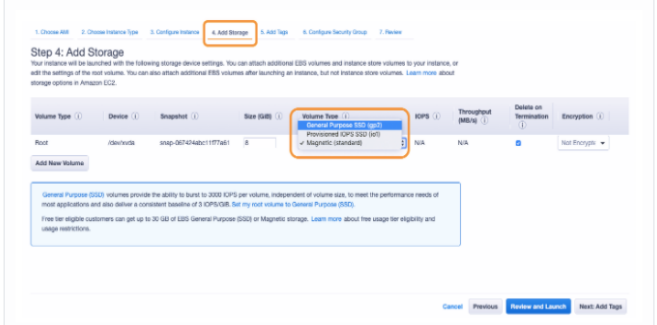
For now, we'll continue with **t2.micro** instance type. But imagine there are 200 EC2 instances, this NAT Gateway may not handle the traffic. So in these cases, we will have to choose a larger instance.

Step 3: Configure Instance Details:

- Let's choose **First-VPC** that we created before as VPC and **public 1-a** as a subnet.
- Leave the rest as default.

Step 4: Add Storage:

- 8 GB disk is enough for **Storage Size**,
- But, **Magnetic Volume** option selected as default is not suitable. Let's change it to **General Purpose SSD** so that it gets faster.



Step 4: Add Storage

Step 5: Add Tags:

Let's name the instance as a **NAT Instance** or whatever you want.

Step 6: Configure Security Group:

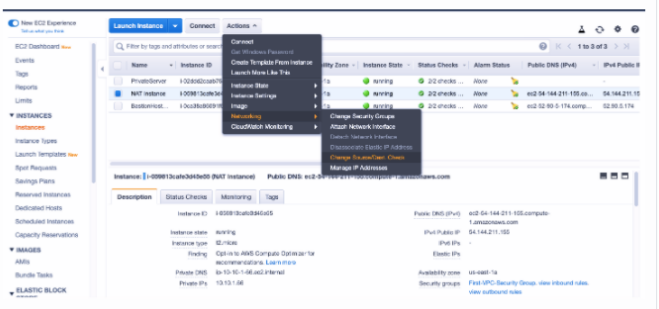
Click **Existing Security Group** option then select **First-VPC Security Group**.

Step 7: Review Instance Launch:

After Review & Launch confirm that you have the key-pair.

Click **Launch Instance**.

Step-2 Setting the NAT Instance



Setting the NAT Instance

If we have created a NAT Instance, the first thing to do is change the **Source / Destination Check** function as you see in the picture above. So,

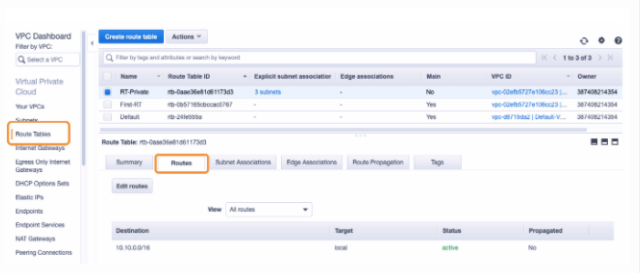
- First of all, we choose our NAT Instance.
- Then click **Networking** from the Actions menu and then go to **Change Source/ Destination Check**
- We click **Disable** option from the window that opens.

But, why do we change this setting?

Normally, virtual machines control which source the package comes from and which source it goes to. This is called the Source/ Destination Check function. If it detects that package does not match its address, it drops the package.

If we do not disable this function, NAT Instance will drop the package because the package doesn't belong to Instance that is located in the Private Subnet.

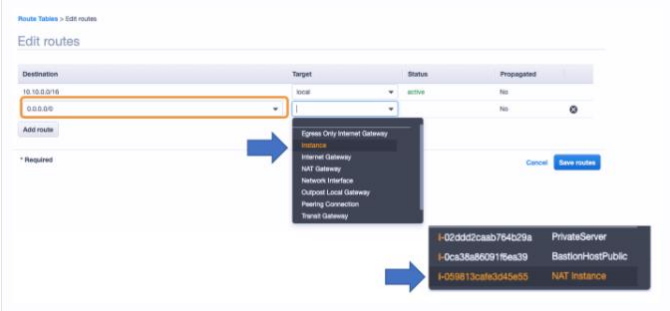
Step-3 Configure the Route Tables



Route Tables

So far, we have created Nat Instance and changed its settings. Let's configure the route table seen above.

- First, go to VPC Dashboard,
- Then, select the **Route Tables** section on the left-hand menu,
- After that, Click the route table we created for Private Subnets,
- Then, Select the **Routes** section below and click **Edit Routes**.

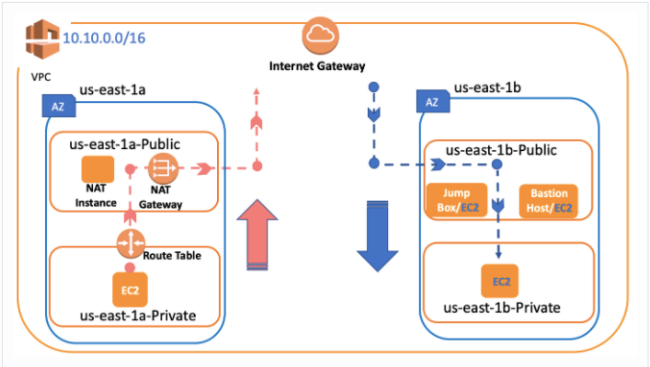


Configure the Route Tables

- After opening the window as you see in the picture above, first, delete the NAT Gateway route if you didn't do it after the last lesson. Because instead of NAT Gateway this time we will define the NAT Instance we created now.
- We enter **0.0.0.0/0** as **Destination** which means anywhere in outside.
- As for the **Target**, first we select **Instance** then click **NAT Instance** from a window pop up at the bottom. So we define the target to deliver the package to NAT Instance.

It's ready to use. We provide connectivity of instance in Private Subnet by NAT Instance in addition to NAT Gateway.

Conclusion : NAT Gateway&NAT Instance vs. Bastion Host&Jumpbox



NAT Gateway&NAT Instance vs. Bastion Host&Jumpbox

Complementary Lesson about Creating NAT Gateway