Building a VPC with a Bastion Host
Technical Documentation
Written by: Jane Pierre

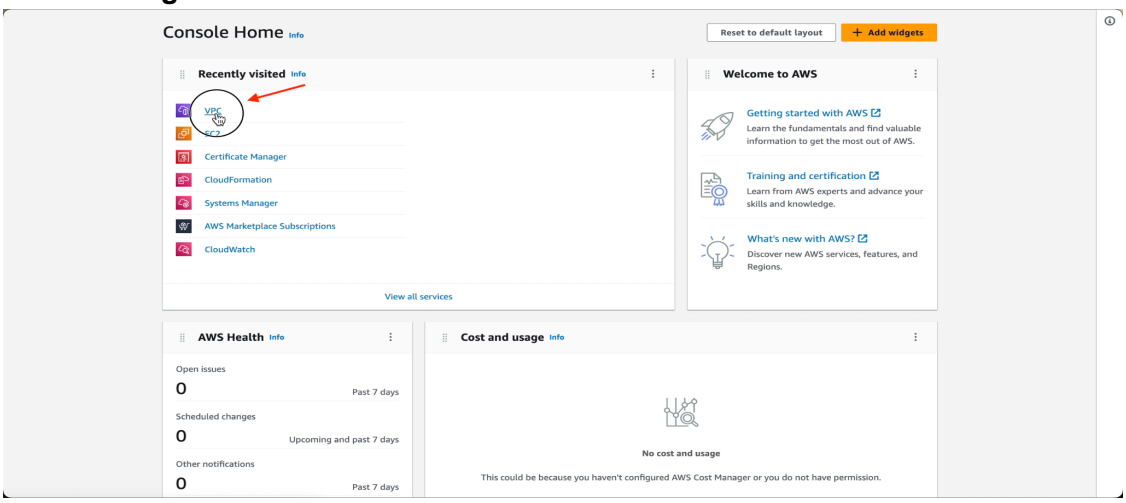
Before proceeding with this lesson you should already:

- have an AWS account
- have a basic understanding of what EC2s are and how they work
- have a basic understanding of how CIDR works
- know the address of 4 CIDR blocks (2 for the private and 2 for the public subnets)

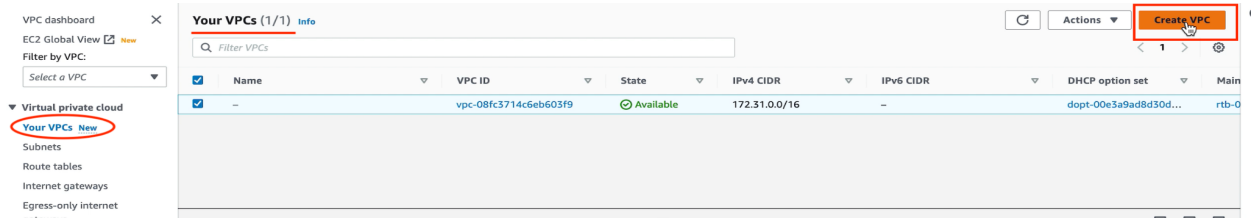
Part 1: Creating VPC

A Virtual Private Cloud also known as VPC is a virtual network that you can create and customize to meet your needs. It enables you to build and launch other AWS resources such as EC2 instances and RDS databases, which you, the user designs and configures. VPC Peering is a secure way to connect two VPCs to each other so that they can communicate through their private IP addresses. It allows each VPC to communicate with the resources in the other VPC as if they were on the same network. In concert, VPC and VPC Peering are an integral part of creating a secure platform by providing segmentation, isolation defense-in-depth and compliance with industry and regulatory standards.

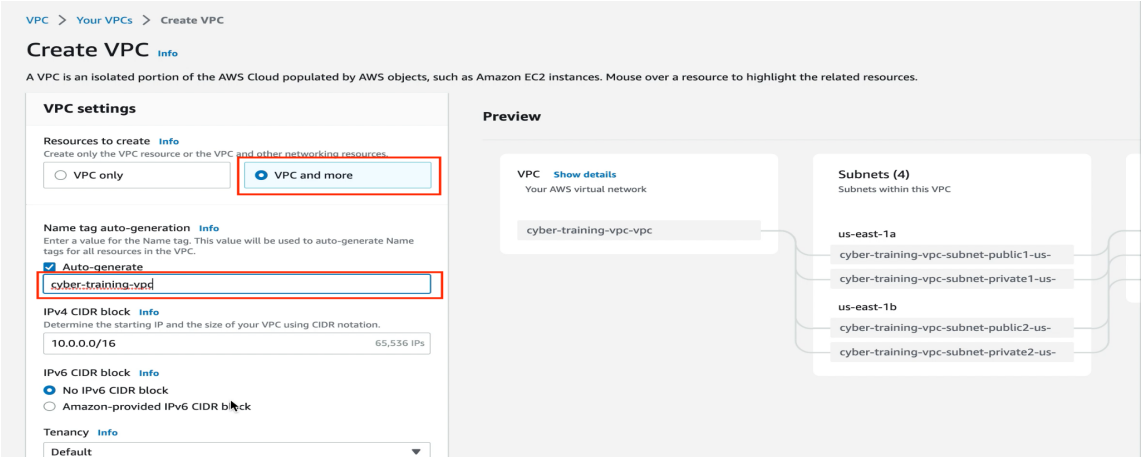
1. **Navigate to the AWS Console Home and select VPC**



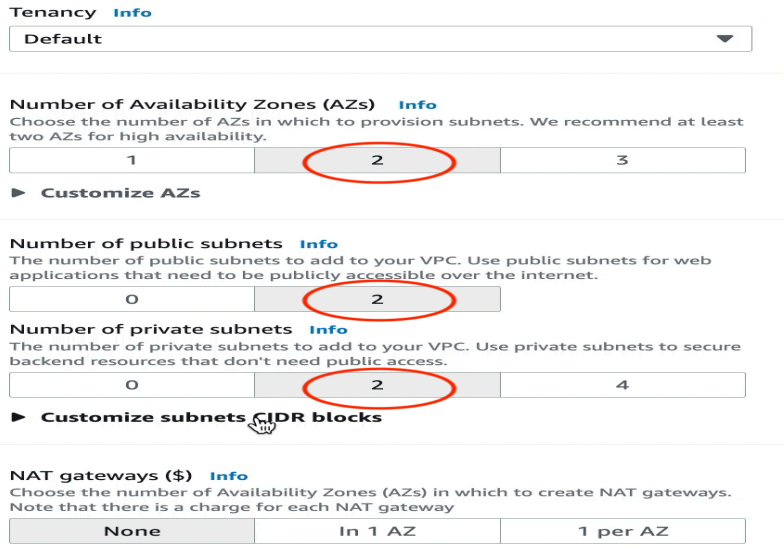
2. **Click "Your VPCs" and select "Create VPC."**



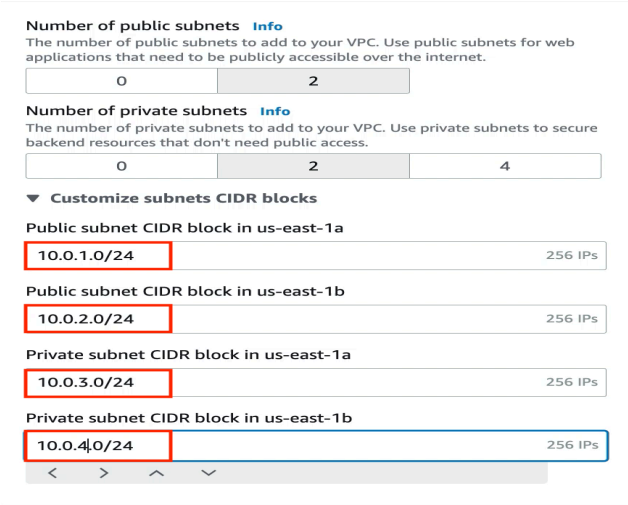
3. **Name your VPC and select, select "VPC and More."** Selecting "VPC and more" will allow you to add additional subnets and availability zones.



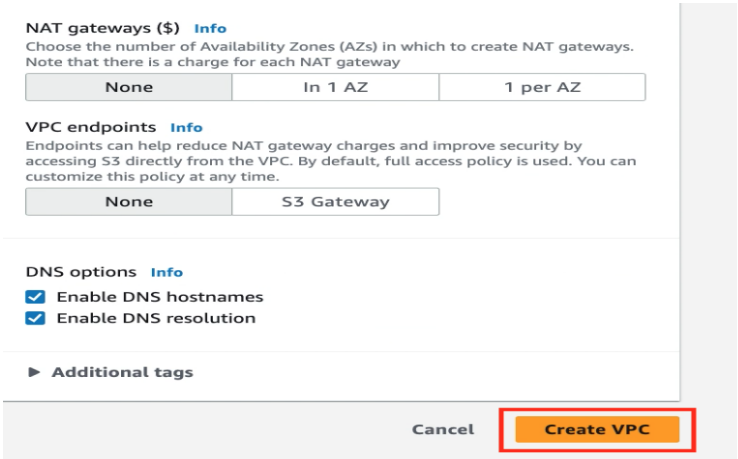
4. **Use the options in the VPC Wizard to add a private and a public subnet.** Make sure that the Availability Zone and number of private and public subnets are set to 2.



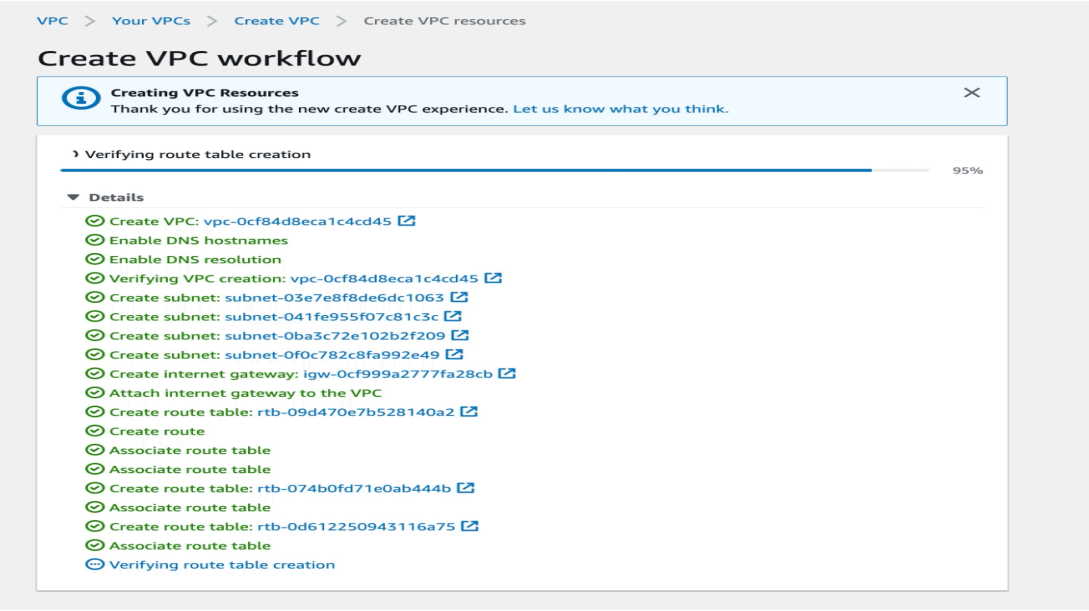
5. **Click "Customize subnets CIDR blocks."** You will use this to set the CIDR range for each subnet in the availability zones.



6. **Check over remaining VPC options.** Click "Create VPC



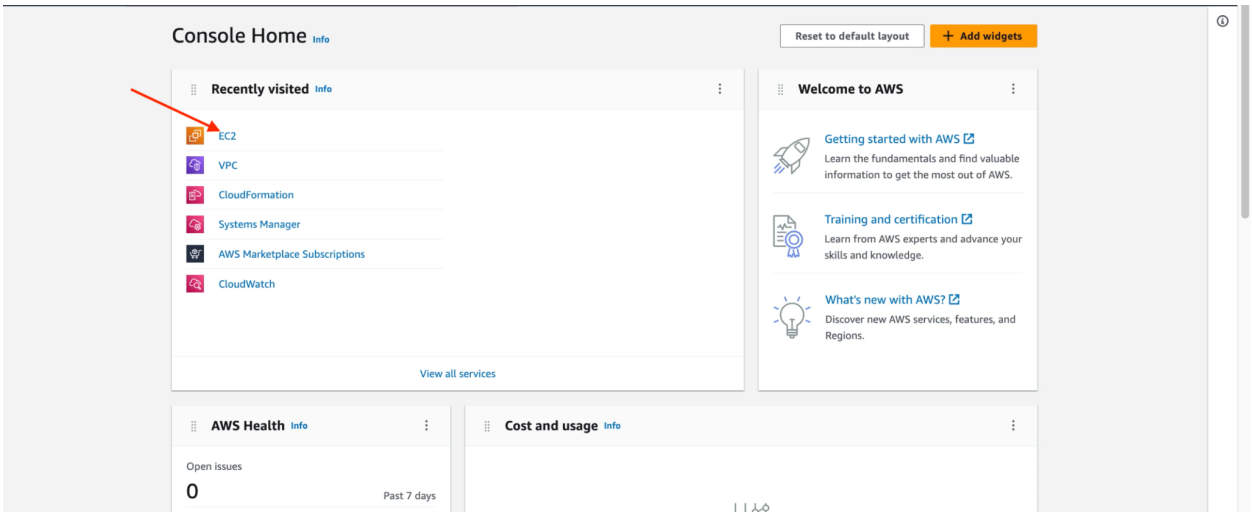
7. **Wait for the VPC workflow to complete.** *Do not close the window during this process.*



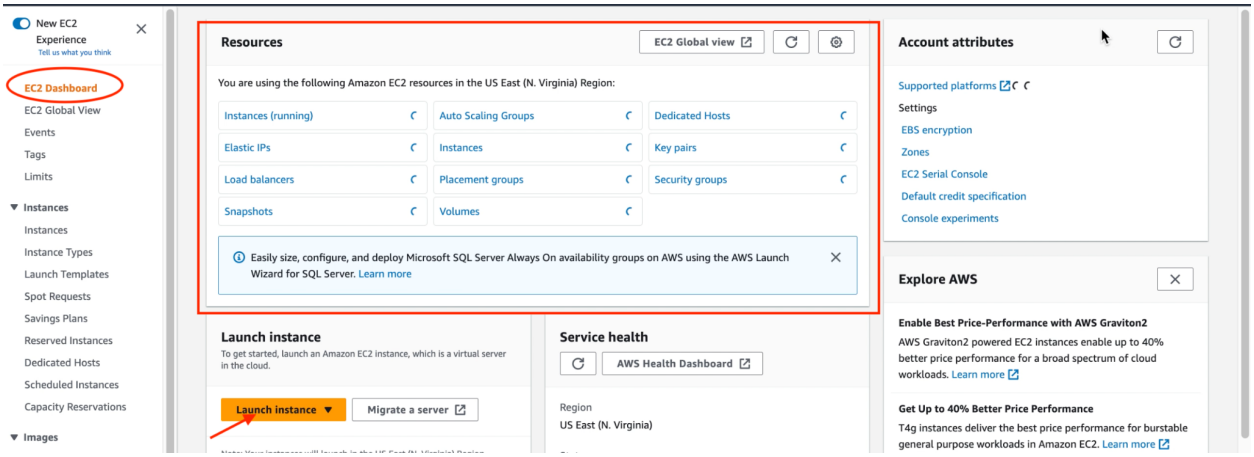
Part 2: Creating a Bastion Host

A bastion host is an essential component of a secure VPC architecture, especially when you need to provide secure access to resources within a private subnet. A bastion host is an EC2 that will act as a server that is exposed to the public internet and is used to securely access resources within a private subnet of a VPC. Also, a bastion host works as a gateway, enabling users to securely SSH into a private instance (usually due to its location on a private subnet) within a VPC. Using a bastion host can improve security by reducing the exposure of private instances to the internet and allows you to limit SSH access to the private subnet to a single, secure entry point. To create a bastion host, we will have to create an EC2 instance and place it on the public subnet of the VPC you created in Part 1.

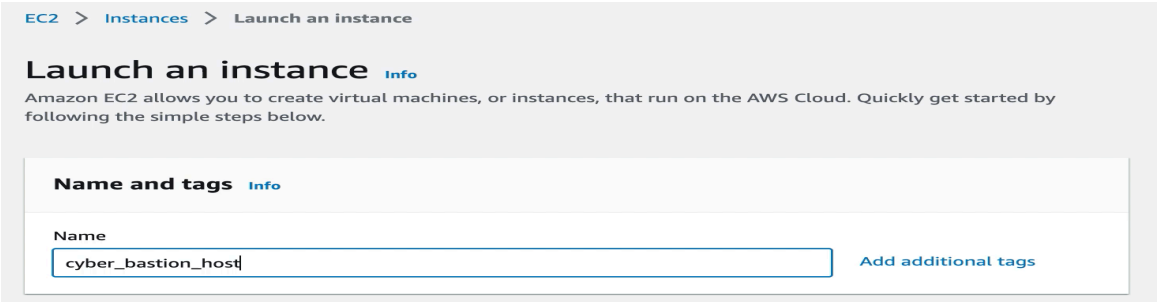
1. **Log in to the AWS Management Console and navigate to the EC2 dashboard.**



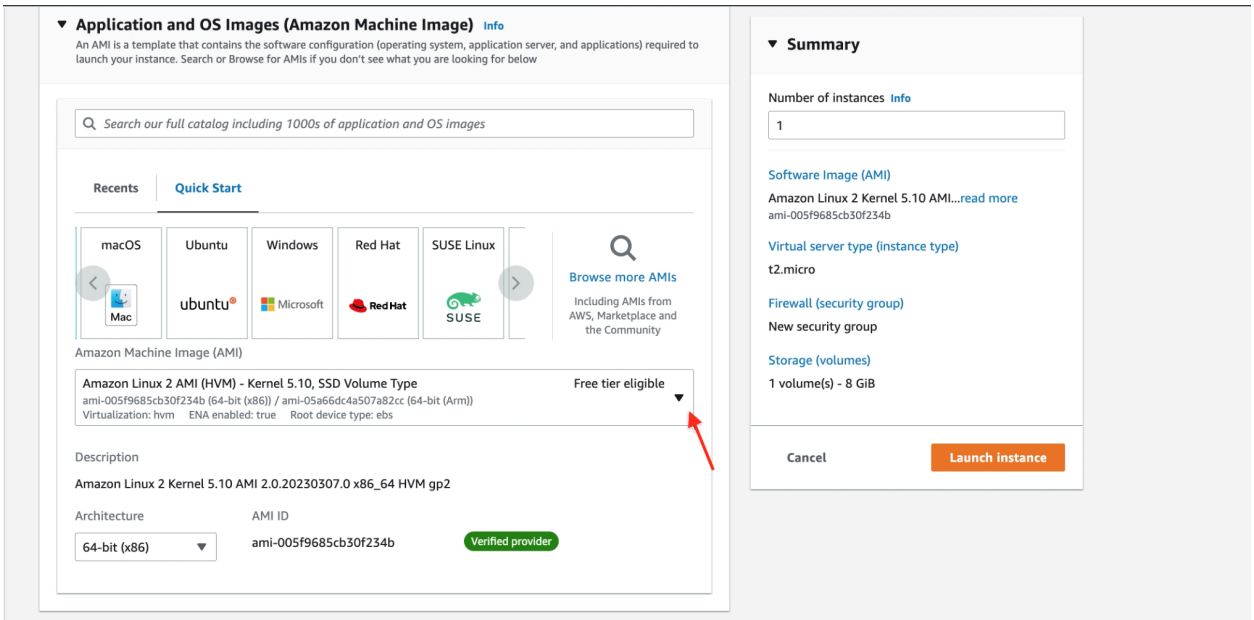
2. **Click on the "Launch Instance" button to begin launching a new EC2 instance.** (This new EC2 instance will become your bastion host.)



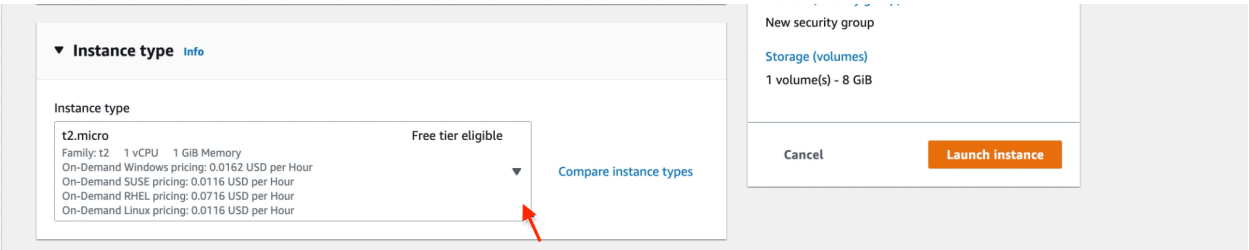
3. **Type a name for your bastion host EC2 instance.**



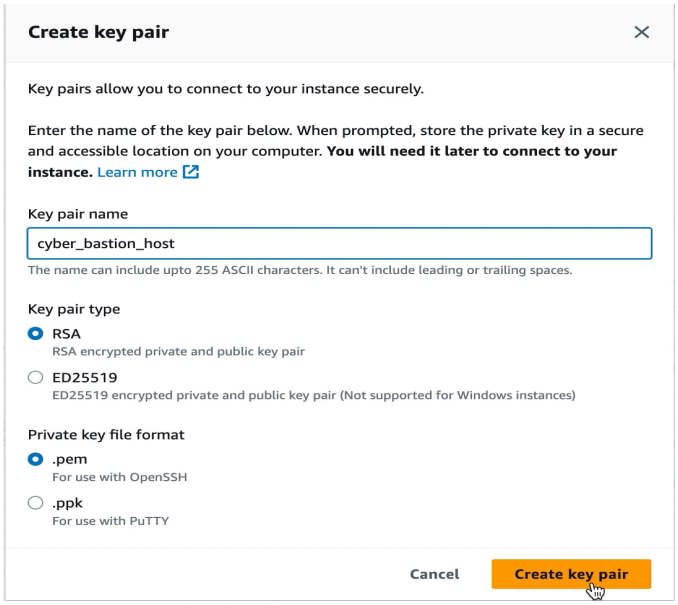
4. **Choose the Amazon Linux 2 AMI as the AMI for the bastion host.**



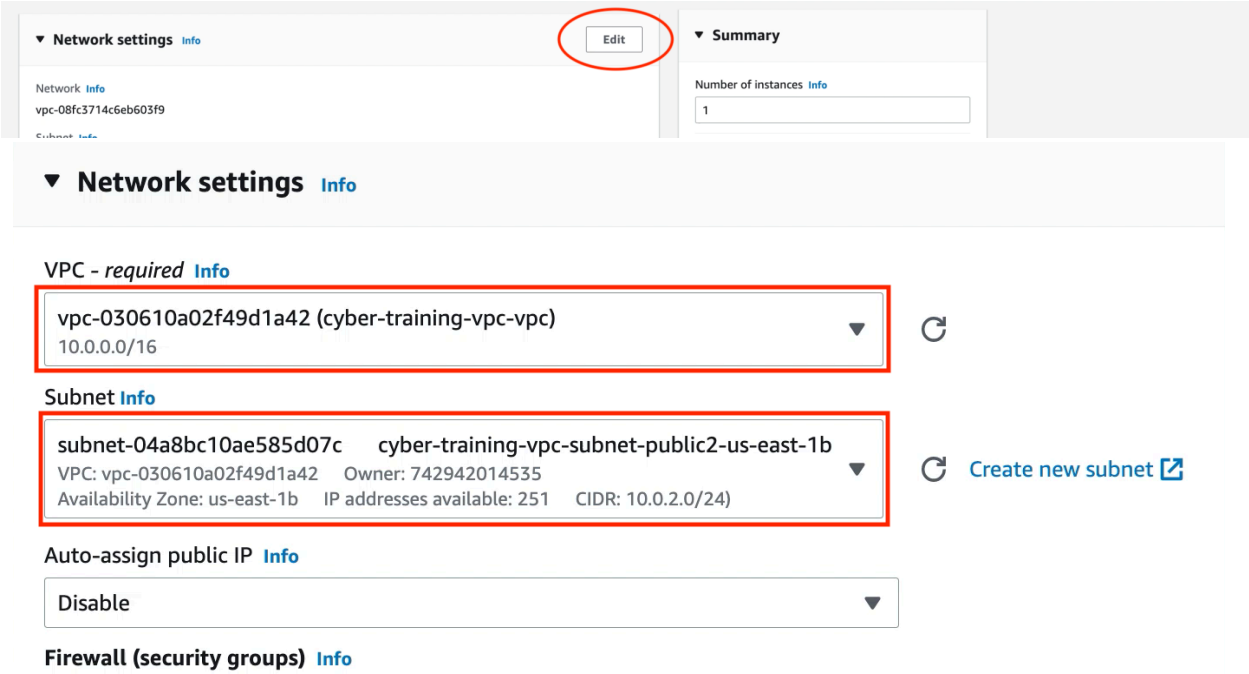
5. **Choose the t2.micro, (which is part of the AWS free tier) as your bastion host instance type.**



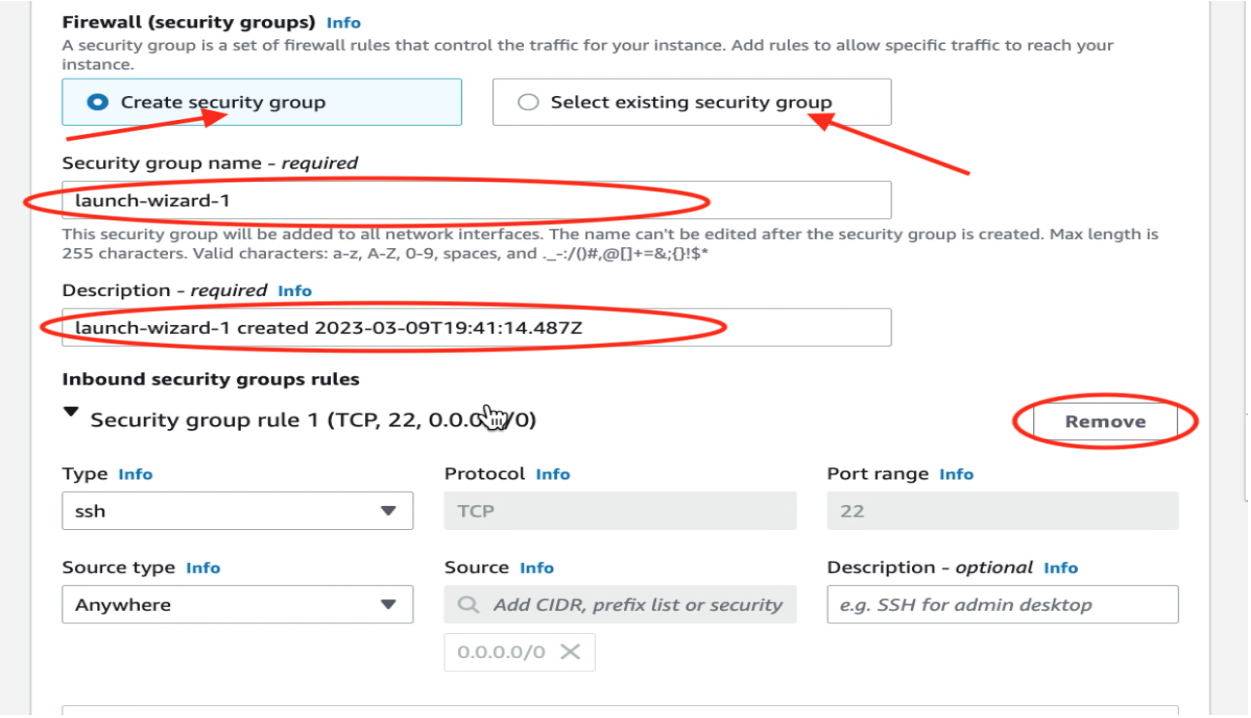
6. **Choose an existing key pair or create a new key pair.** This key pair will be used later in Steps 19 thru 24 to SSH into your instance.

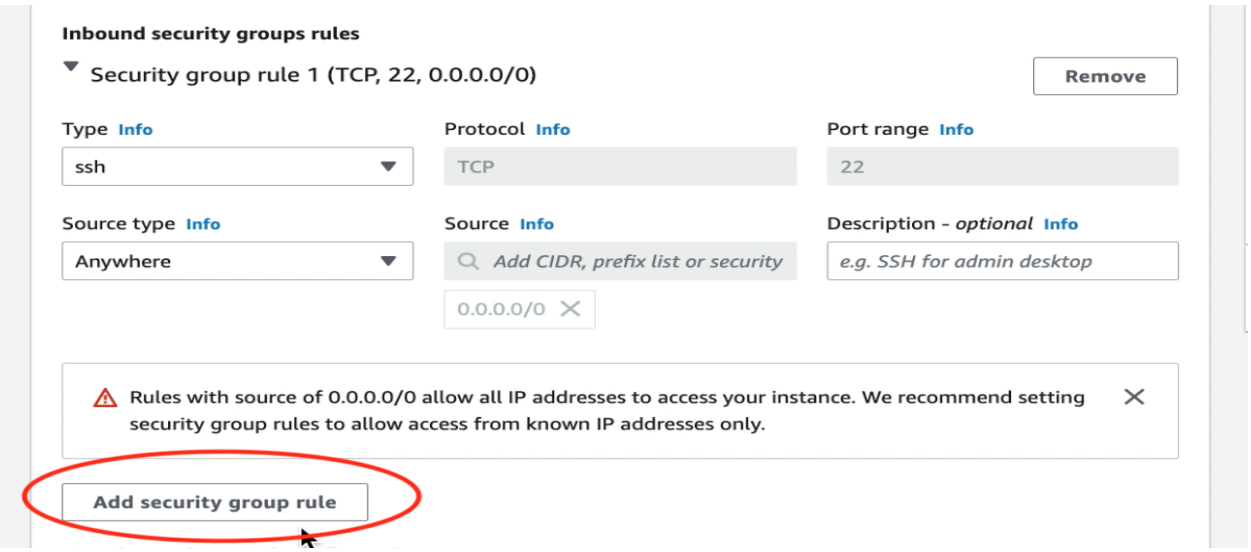


7. **In the "Network Settings" section, click the "Edit" button.** Select the VPC you made in Part 1 and a public subnet for the bastion host to reside on.

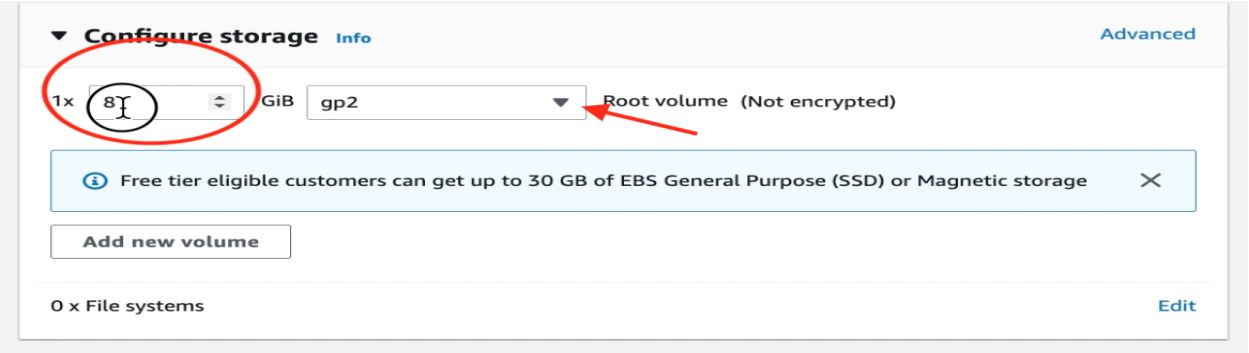


8. **While in the "Network Settings" section, review your Security Groups settings.** Here you can create and configure a new security group or select an already existing security group. You can use the "Add security group rule" button to add or the "Remove" button to delete a security group rule.

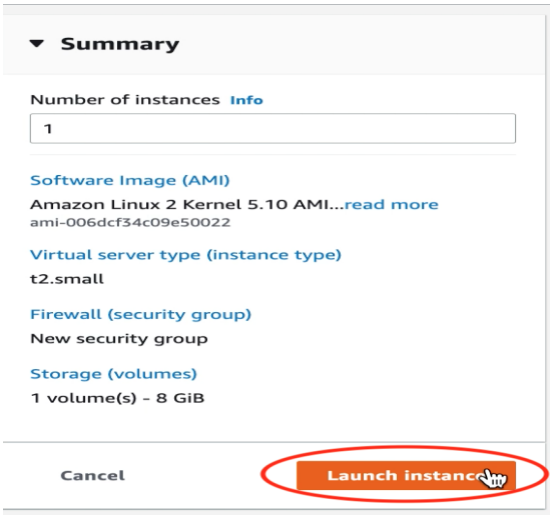




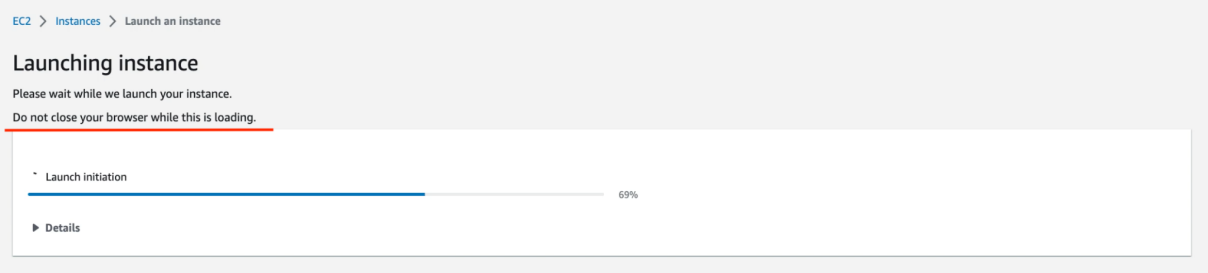
9. **Review and if needed configure your storage options.** For the purposes of a bastion host it isn't necessary to change the default storage options.



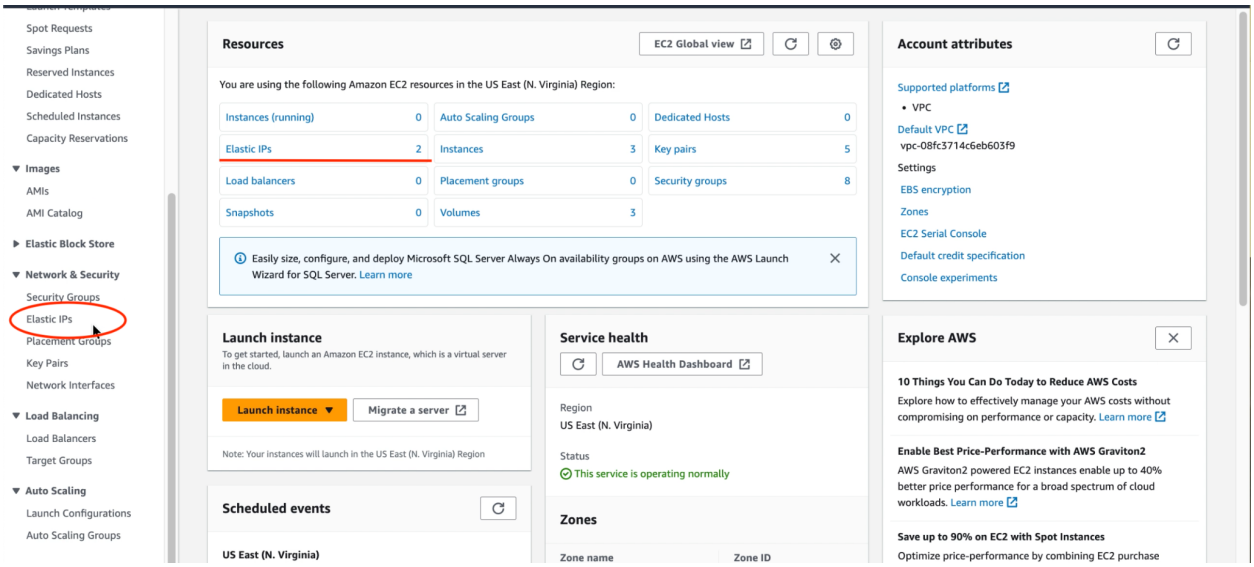
10. **Review your instance details and click on "Launch" to launch your instance.**



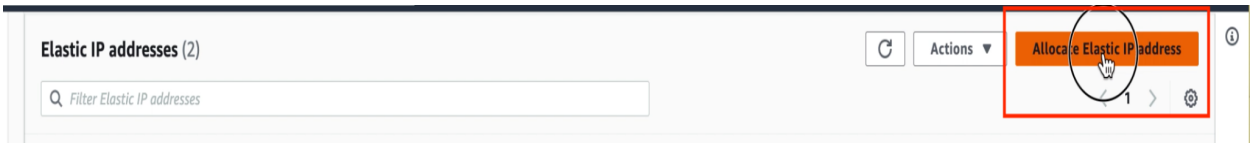
11. **Wait for your instance to launch.** *Do not close the browser window while the instance is preparing to launch.*



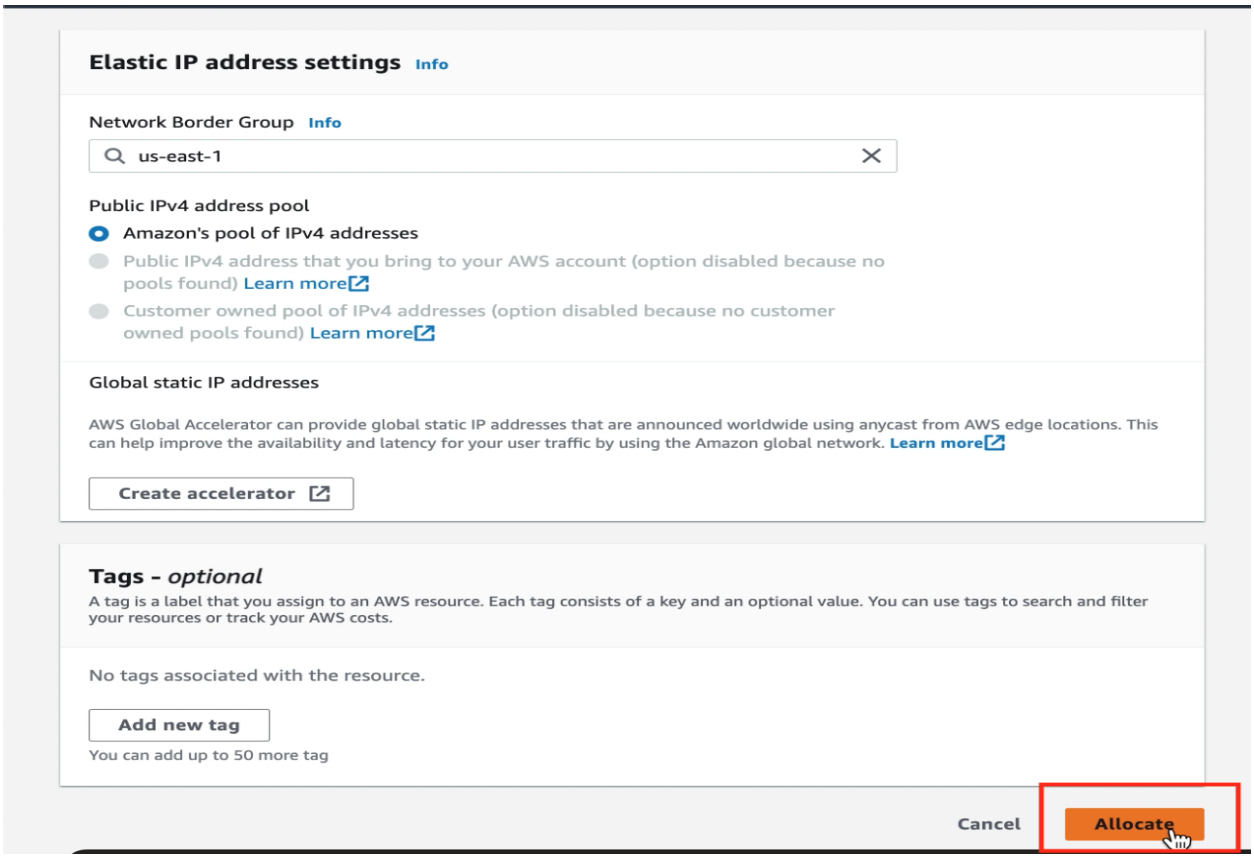
12. **To remotely access the bastion host you will have to use an SSH connection. One such way is to allocate and attach an Elastic IP.** In order to do this, return to the EC2 Dashboard and select Elastic IP.



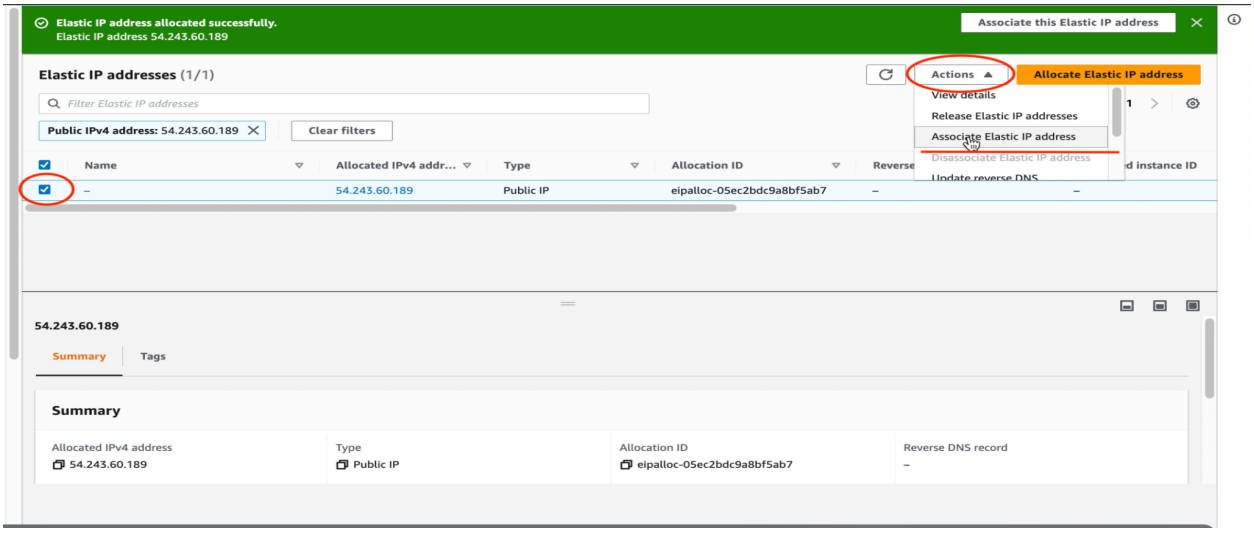
13. **Select "Allocate Elastic IP Address"**



14. **Under "Elastic IP address settings" choose your Network Border Group, Public IPv4 option and Tags.** Click "Allocate."

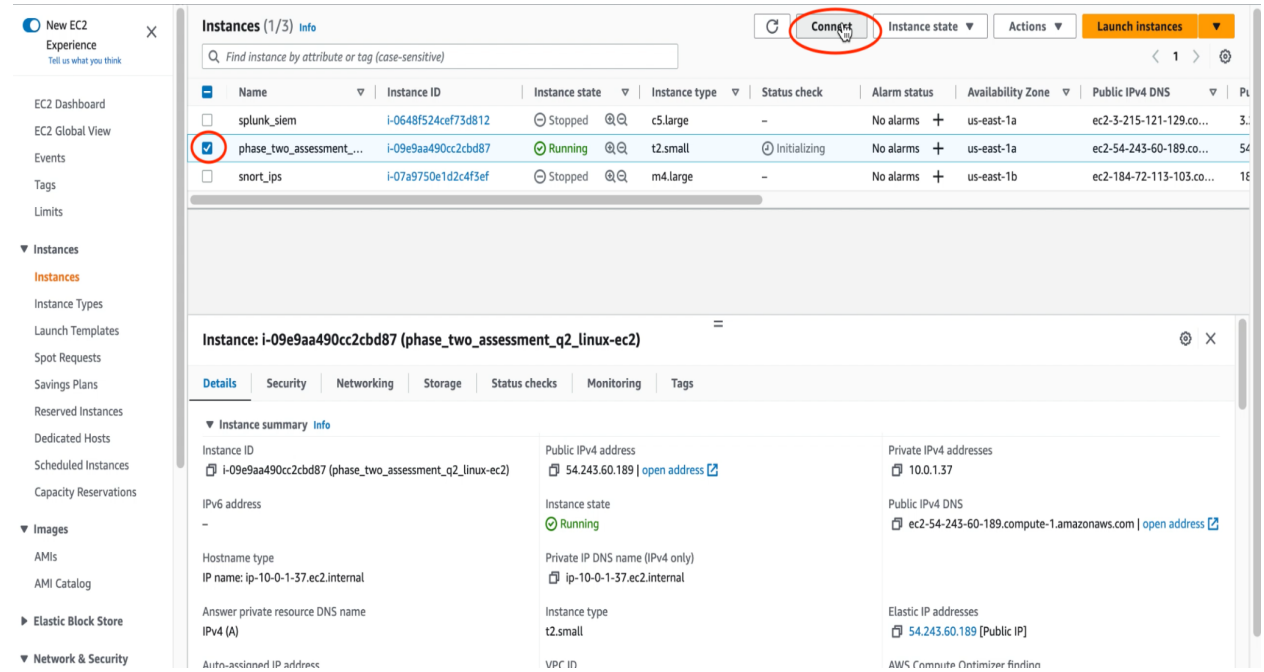


15. **Usually, you will be redirected to Elastic IP menu, however, if you are not, you can return to the EC2 Dashboard.** Here, you will attach the Elastic IP address to the bastion host EC2 you have just created. First, make sure that you have the correct newly allocated Elastic IP selected with its checkbox marked. Then, click on the "Actions" button and select "Associate Elastic IP address."

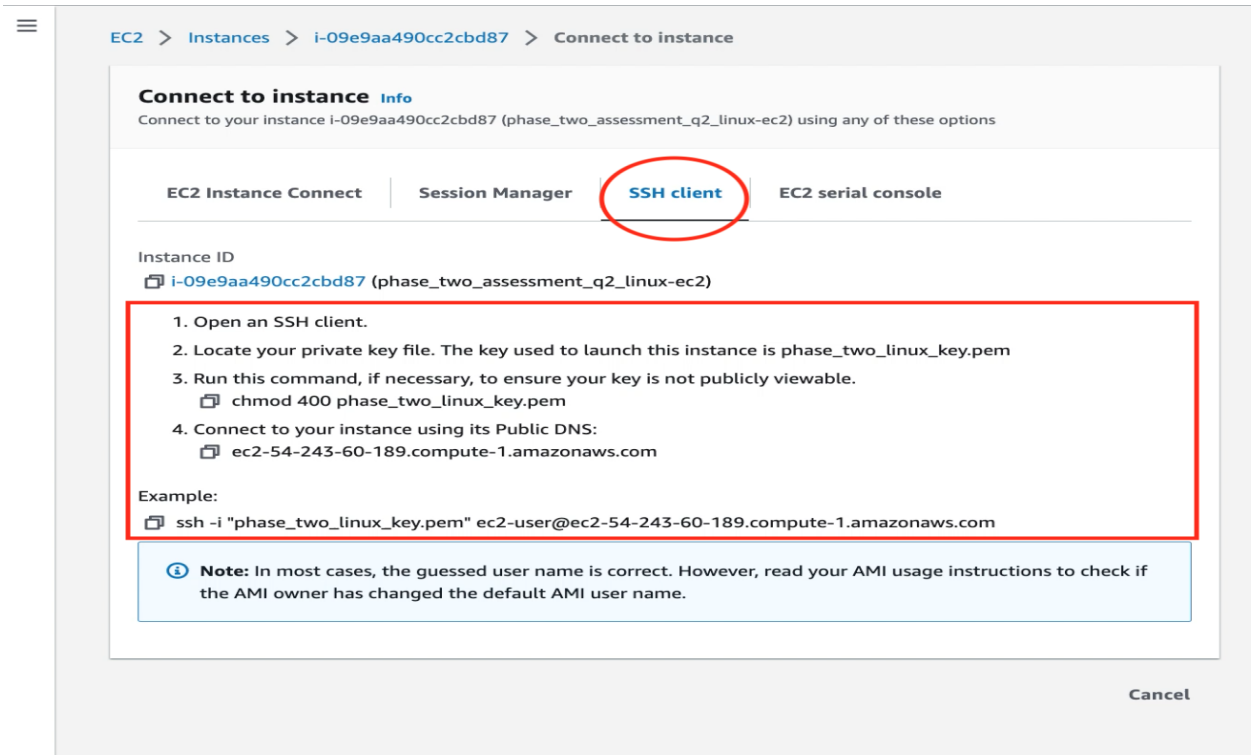


16. **Within "Associate Elastic IP address" options, under "Resource type" select "Instance."** Click under "Instance" and a drop menu showing your bastion host will appear. Select your bastion host instance. Review your selections and click "Associate" to attach the Elastic IP to your EC2 Instance.

17. **Return to the EC2 Dashboard.** Click on the checkbox next to your EC2 Instance name and click "Connect."

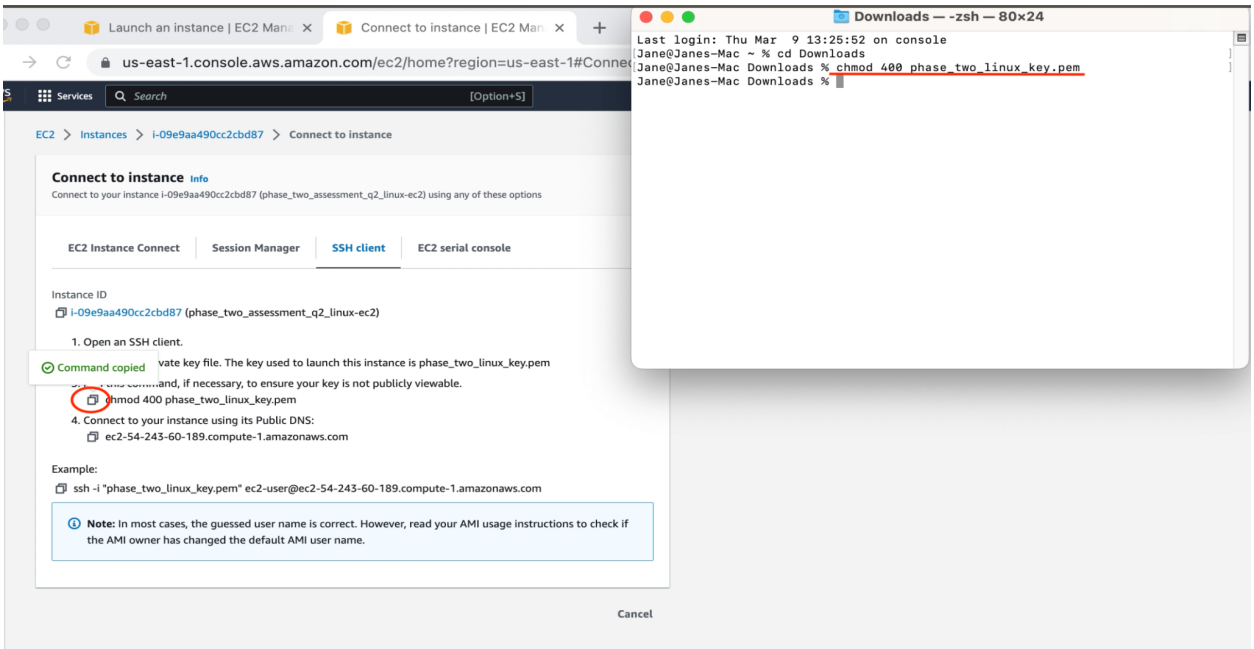


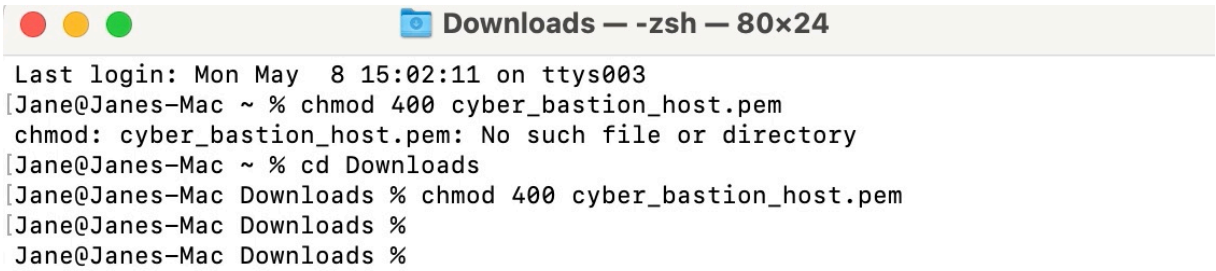
18. Under "Connect to instance," click on "SSH client" tab for instructions on how to SSH into the EC2 Instance.



19. Open the application that will allow you to SSH into the EC2 Instance, in this case, it is a Mac terminal. Use the “cd ..” (without the quotation marks) command to move from directory to directory until you are in your Downloads folder.

20. From the “SSH client” tab, copy the command that will ensure your key pair is not publicly viewable and paste it into the terminal CLI. Press the Enter key.





21. First run the command eval ‘ssh-agent’ this will initialize an SSH agent and set the necessary environment variables so that subsequent SSH operations, such as adding keys with ssh-add

or using SSH-based tools, can interact with the agent and leverage the stored keys for authentication.

```
Downloads — -zsh — 80×24
Last login: Mon May  8 15:02:11 on ttys003
Jane@Janes-Mac ~ % chmod 400 cyber_bastion_host.pem
chmod: cyber_bastion_host.pem: No such file or directory
Jane@Janes-Mac ~ % cd Downloads
Jane@Janes-Mac Downloads % chmod 400 cyber_bastion_host.pem
Jane@Janes-Mac Downloads % chmod 400 cyber_training_test.pem
Jane@Janes-Mac Downloads % eval 'ssh-agent'
SSH_AUTH_SOCK=/var/folders/5g/zldhbcls43g2wzxy53b888b00000gp/T//ssh-dZFVRnoFM1nX
/agent.29034; export SSH_AUTH_SOCK;
SSH_AGENT_PID=29035; export SSH_AGENT_PID;
echo Agent pid 29035;
Jane@Janes-Mac Downloads % ssh-add -K ./cyber_bastion_host.pem
WARNING: The -K and -A flags are deprecated and have been replaced
         by the --apple-use-keychain and --apple-load-keychain
         flags, respectively.  To suppress this warning, set the
         environment variable APPLE_SSH_ADD_BEHAVIOR as described in
         the ssh-add(1) manual page.
Identity added: ./cyber_bastion_host.pem (./cyber_bastion_host.pem)
Jane@Janes-Mac Downloads %
```

22. Then run the command ssh-add ./<pemkey.pem> (when you're using the MacOS terminal you have to run the command with an additional -K. For example `ssh-add ./cyber_bastion_host.pem` (or in my case `ssh-add -K ./cyber_bastion_host.pem`.) This will add private key identities to the SSH authentication agent and enable agent forwarding during SSH connections.


```
Downloads — -zsh — 80×24
Last login: Mon May  8 15:02:11 on ttys003
Jane@Janes-Mac ~ % chmod 400 cyber_bastion_host.pem
chmod: cyber_bastion_host.pem: No such file or directory
Jane@Janes-Mac ~ % cd Downloads
Jane@Janes-Mac Downloads % chmod 400 cyber_bastion_host.pem
Jane@Janes-Mac Downloads % chmod 400 cyber_training_test.pem
Jane@Janes-Mac Downloads % eval 'ssh-agent'
SSH_AUTH_SOCK=/var/folders/5g/zldhbcls43g2wzxy53b888b00000gp/T//ssh-dZFVRnoFM1nX
/agent.29034; export SSH_AUTH_SOCK;
SSH_AGENT_PID=29035; export SSH_AGENT_PID;
echo Agent pid 29035;
Jane@Janes-Mac Downloads % ssh-add -K ./cyber_bastion_host.pem
WARNING: The -K and -A flags are deprecated and have been replaced
         by the --apple-use-keychain and --apple-load-keychain
         flags, respectively.  To suppress this warning, set the
         environment variable APPLE_SSH_ADD_BEHAVIOR as described in
         the ssh-add(1) manual page.
Identity added: ./cyber_bastion_host.pem (./cyber_bastion_host.pem)
Jane@Janes-Mac Downloads %
```


23. From the "SSH client" tab, copy the command under "Example" that will allow you to securely connect to and SSH into the EC2 Instance with your key pair and paste it into the terminal

CLI. Press the Enter key.After displaying a message regarding the security of the host, you will be prompted to answer the question, “Are you sure you want to continue connecting (yes/no/[fingerprint])?” Type yes as your answer and press the Enter key.

```
Downloads — ec2-user@ip-10-0-2-242:~ — ssh -A -i cyber_bastion_host...
          flags, respectively.  To suppress this warning, set the
          environment variable APPLE_SSH_ADD_BEHAVIOR as described in
          the ssh-add(1) manual page.
Identity added: ./cyber_training_test.pem (./cyber_training_test.pem)
Jane@Janes-Mac Downloads % ssh -A -i "cyber_bastion_host.pem" ec2-user@ec2-52-20
6-104-53.compute-1.amazonaws.com
The authenticity of host 'ec2-52-206-104-53.compute-1.amazonaws.com (2607:7700:0
:1e:0:1:34ce:6835)' can't be established.
ED25519 key fingerprint is SHA256:dEER7TFyL6NwqCvIjftmzo8i6tm1fFh7qfRrUUl2MTk.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-52-206-104-53.compute-1.amazonaws.com' (ED25519)
 to the list of known hosts.
   ,     #_
   ~\_  ####_          Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___    https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-10-0-2-242 ~]$
```

24. To test that the bastion host is functioning and has internet connectivity, you can ping to a website like www.google.com

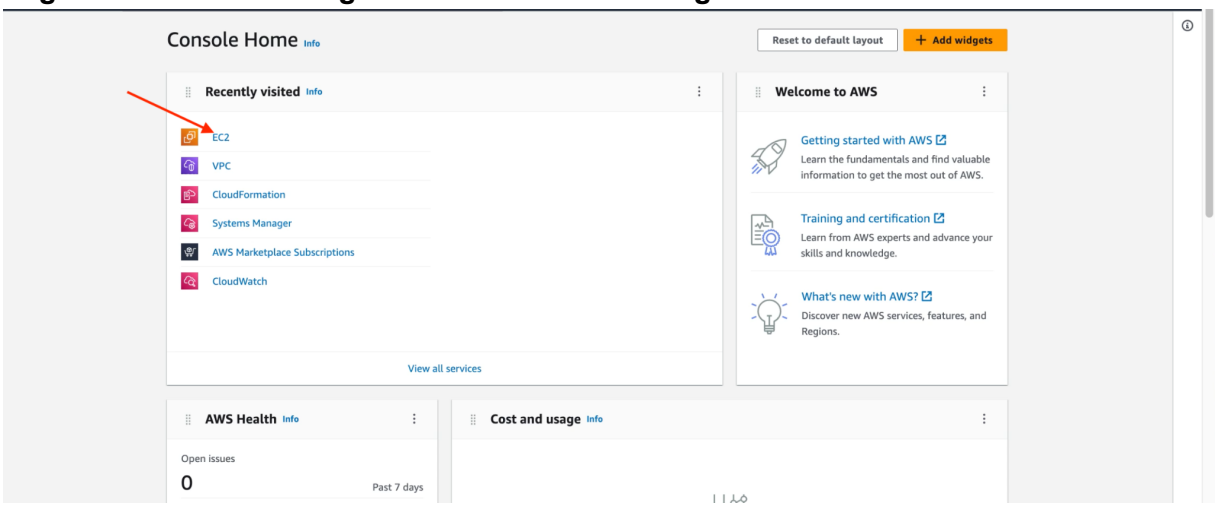
```
Downloads — ec2-user@ip-10-0-2-242:~ — ssh -A -i cyber_bastion_host...
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-10-0-2-242 ~]$ ping www.google.com
PING www.google.com (142.251.16.106) 56(84) bytes of data.
64 bytes from bl-in-f106.1e100.net (142.251.16.106): icmp_seq=1 ttl=96 time=1.94
 ms
64 bytes from bl-in-f106.1e100.net (142.251.16.106): icmp_seq=2 ttl=96 time=1.96
 ms
64 bytes from bl-in-f106.1e100.net (142.251.16.106): icmp_seq=3 ttl=96 time=1.95
 ms
64 bytes from bl-in-f106.1e100.net (142.251.16.106): icmp_seq=4 ttl=96 time=1.96
 ms
64 bytes from bl-in-f106.1e100.net (142.251.16.106): icmp_seq=5 ttl=96 time=1.94
 ms
64 bytes from bl-in-f106.1e100.net (142.251.16.106): icmp_seq=6 ttl=96 time=1.95
 ms
^C
--- www.google.com ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
rtt min/avg/max/mdev = 1.935/1.949/1.964/0.010 ms
[ec2-user@ip-10-0-2-242 ~]$
```

You have now successfully created a Bastion Host EC2…now lets put it to the test.

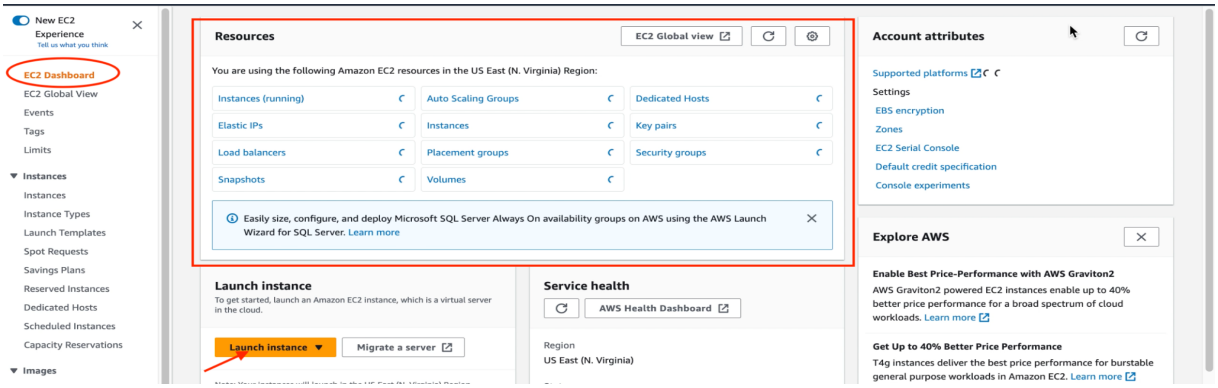
Part 3: Creating a Test EC2 on a Private Subnet

A bastion host, acts as an intermediary server between your local machine and resources in a VPC on AWS. It serves as an entry point to access your private resources securely. Testing a bastion host in AWS is important to ensure its functionality, security and proper configuration. To test the bastion host, an EC2 instance must be created, but this time placed on a private subnet of the VPC you made in Part 1.

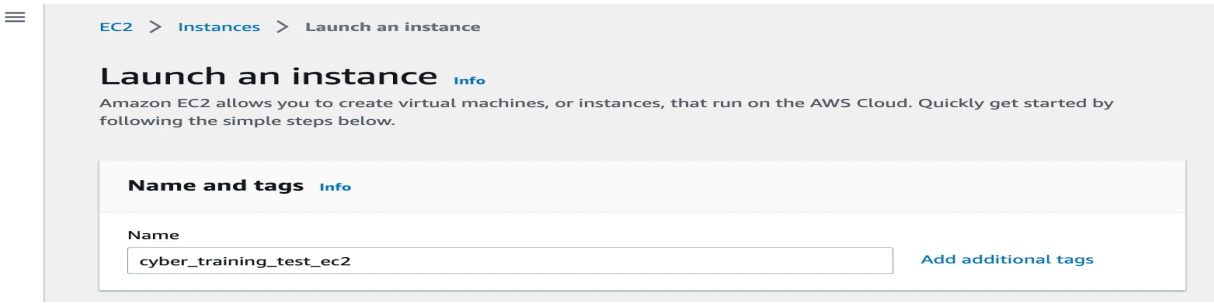
1. **Log in to the AWS Management Console and navigate to the EC2 dashboard.**



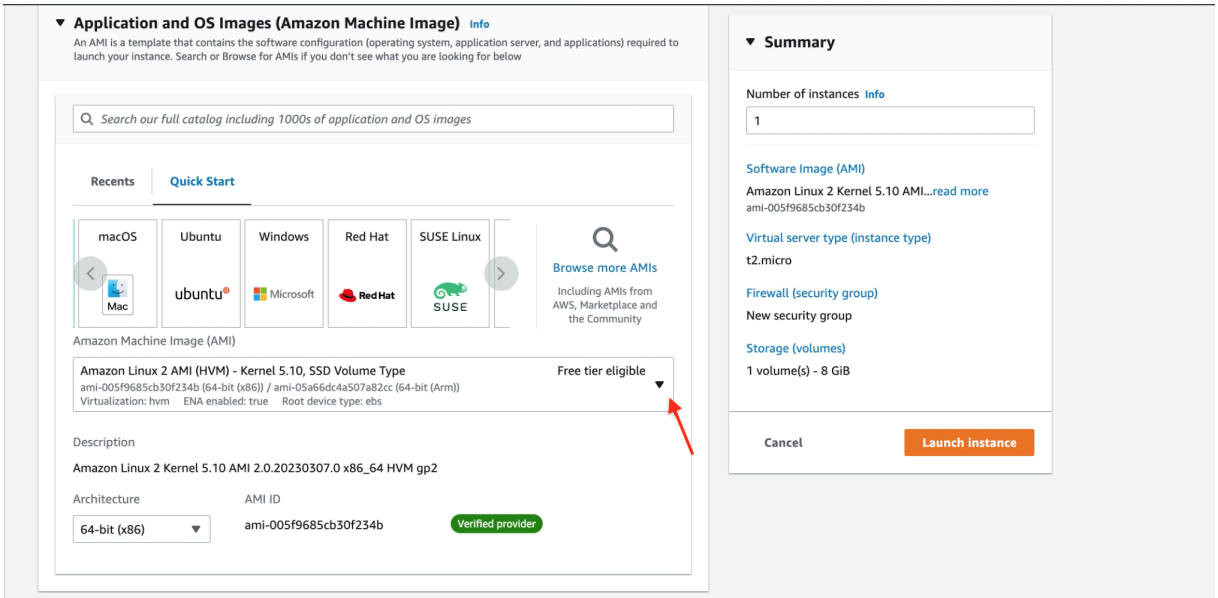
2. **Click on the "Launch Instance" button to begin launching a new EC2 instance.**



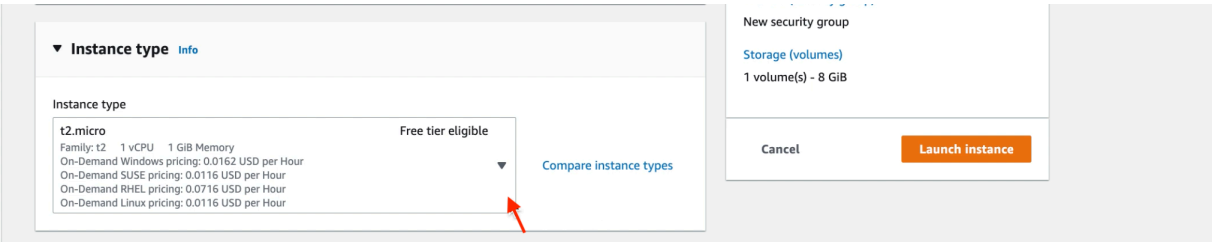
3. **Type a name for your test EC2 instance.**



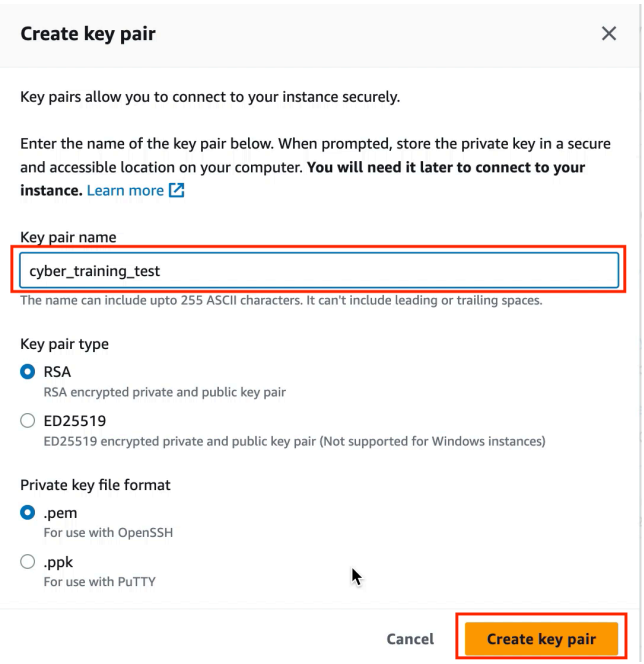
4. **Choose the Amazon Linux 2 AMI as the AMI for the EC2.**



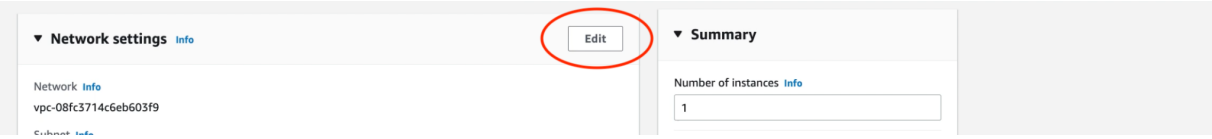
5. **Choose the t2.micro, (which is part of the AWS free tier) as your instance type.**

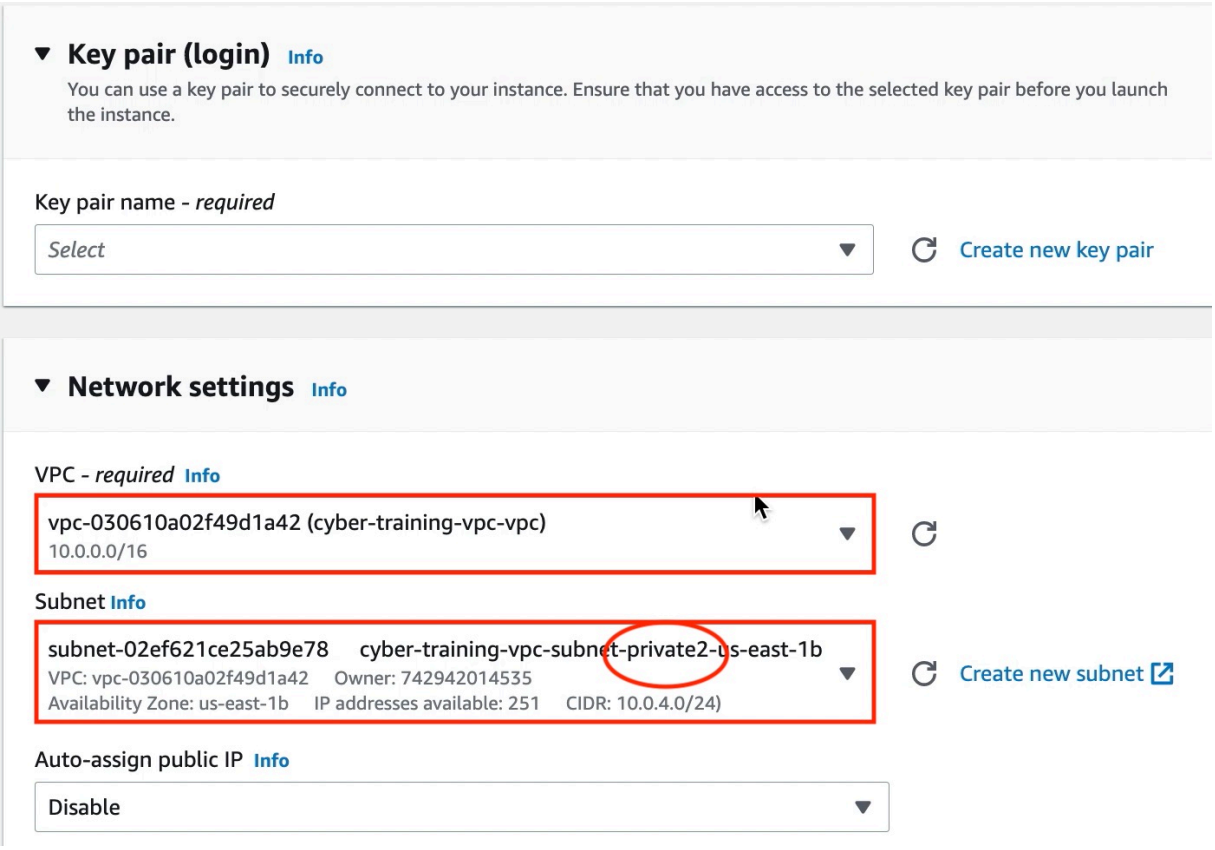


6. **Choose an existing key pair or create a new key pair.** This key pair will be used later to SSH into this instance from the bastion host..



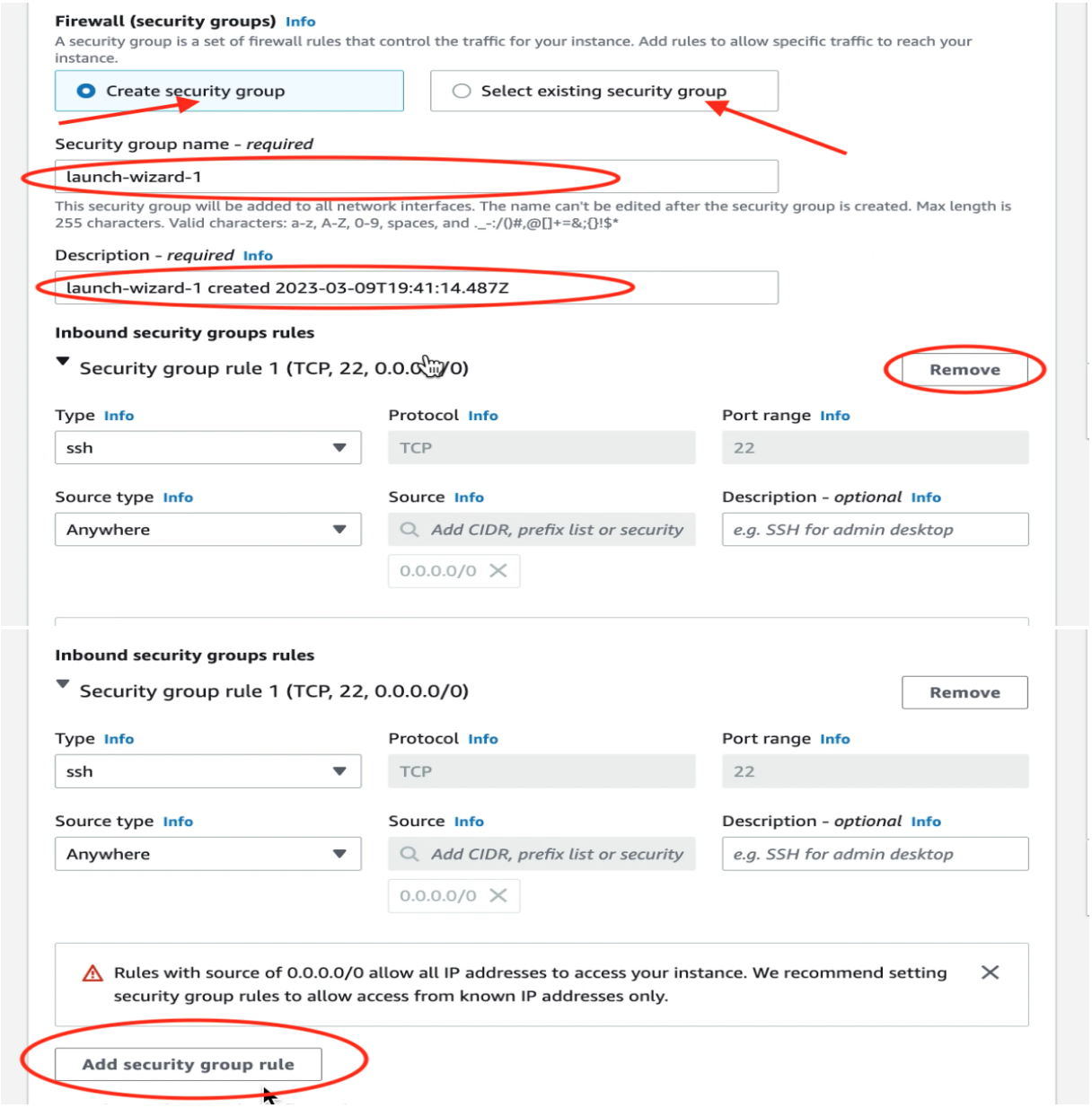
7. **In the "Network Settings" section, click the “Edit” button.** Select the VPC you made in Part 1 and a *private* subnet for the EC2 to reside on.



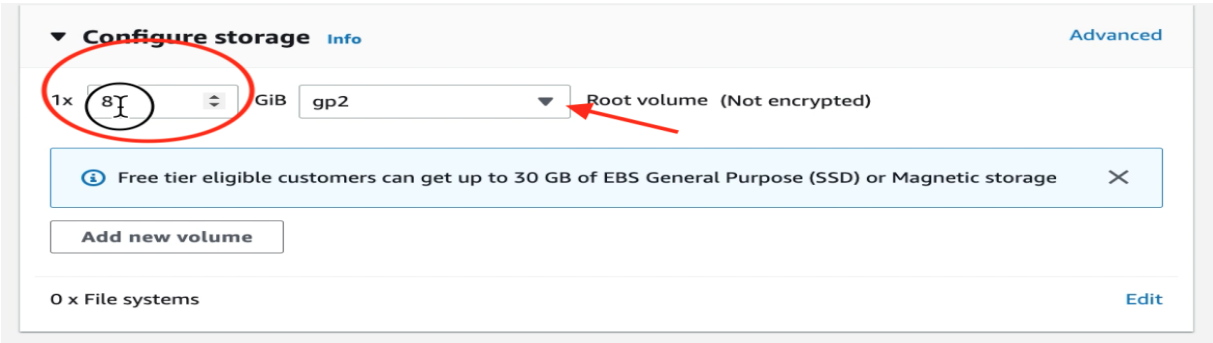


8. **While in the "Network Settings" section, review your Security Groups settings.** Here you can create and configure a new security group or select an already existing

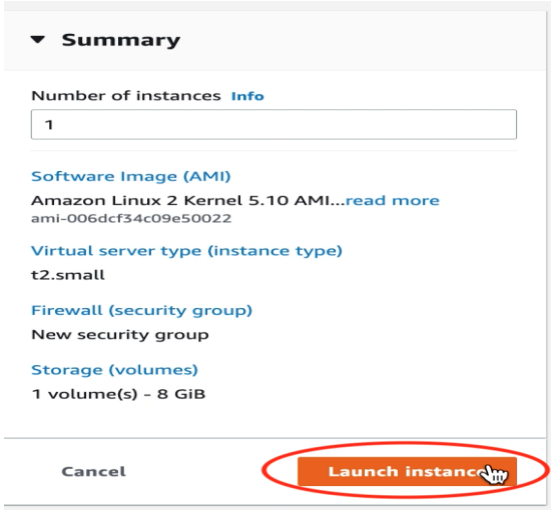
security group. You can use the “Add security group rule” button to add or the “Remove” button to delete a security group rule.



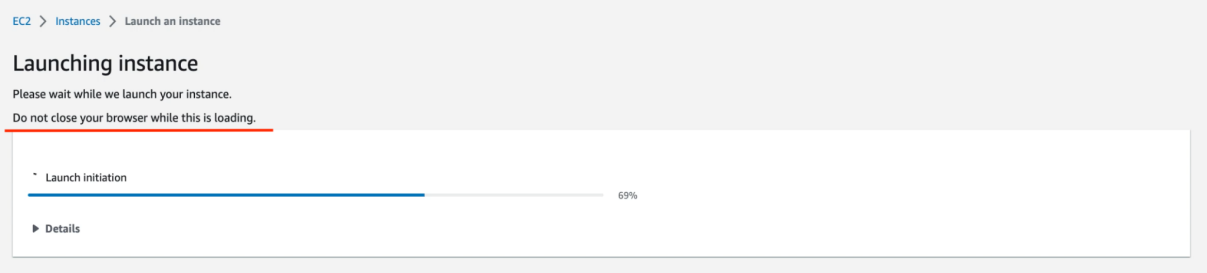
9. **Review and if needed configure your storage options.** For the purposes of a bastion host it isn’t necessary to change the default storage options.



10. **Review your instance details and click on "Launch" to launch your instance.**



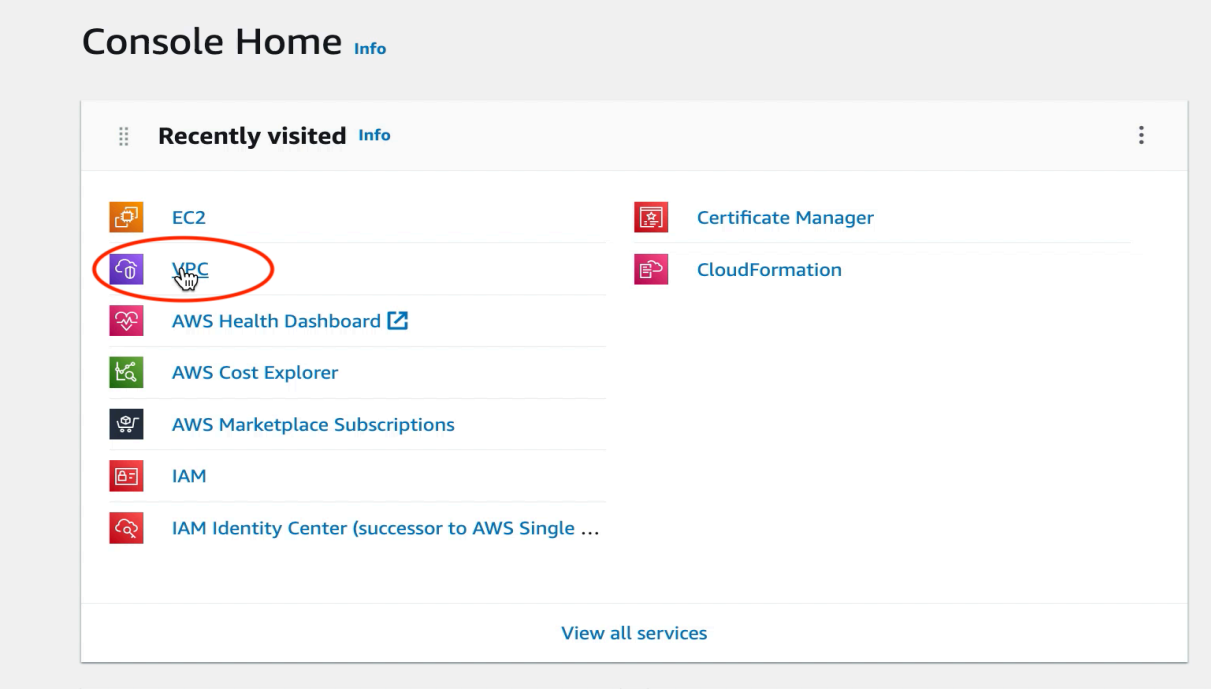
11. **Wait for your instance to launch.** *Do not close the browser window while the instance is preparing to launch.*



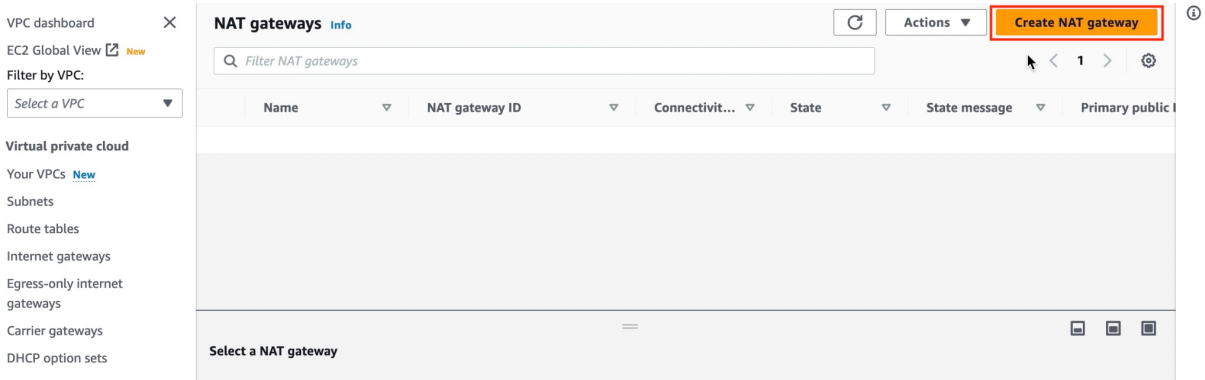
Part 4: Creating a NAT Gateway

After creating the EC2 in a private subnet that will be used to test the bastion host, a Network Address Translation gateway must be created linking the private subnet to the internet. A Network Address Translation or NAT gateway is a managed service that provides outbound internet connectivity for resources within private subnets of a VPC. It allows instances within your private subnets to access the internet for tasks such as downloading software updates, interacting with external APIs, or retrieving data from external sources. From there, the bastion host will SSH into the EC2 instance on the private subnet ensuring that the bastion host is functioning properly.

1. **From the console, navigate back to the VPC Dashboard.** On the left hand side, in the Virtual Private Cloud drop down menu, select “NAT gateways.”

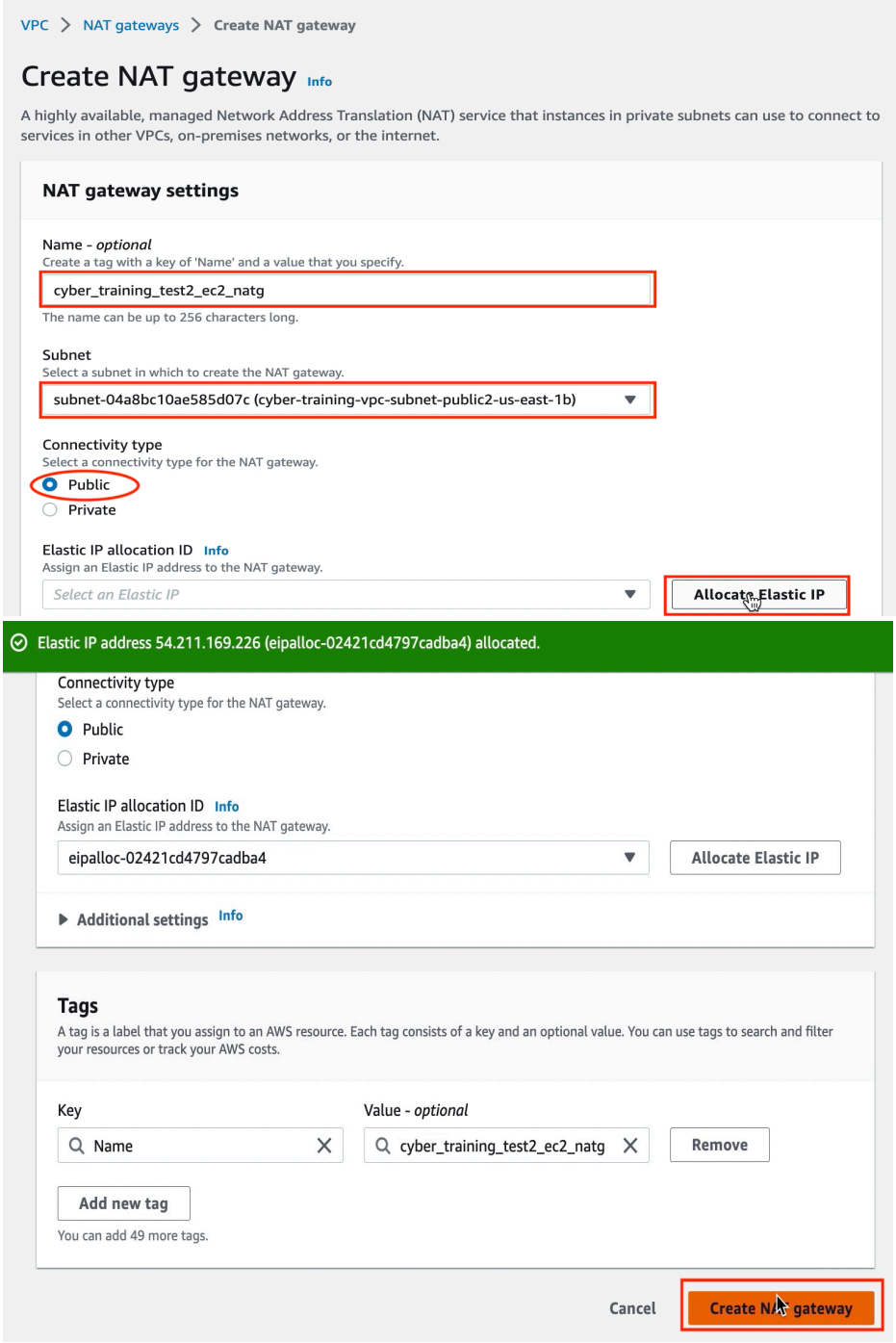


2. **Click "Create NAT Gateway"**



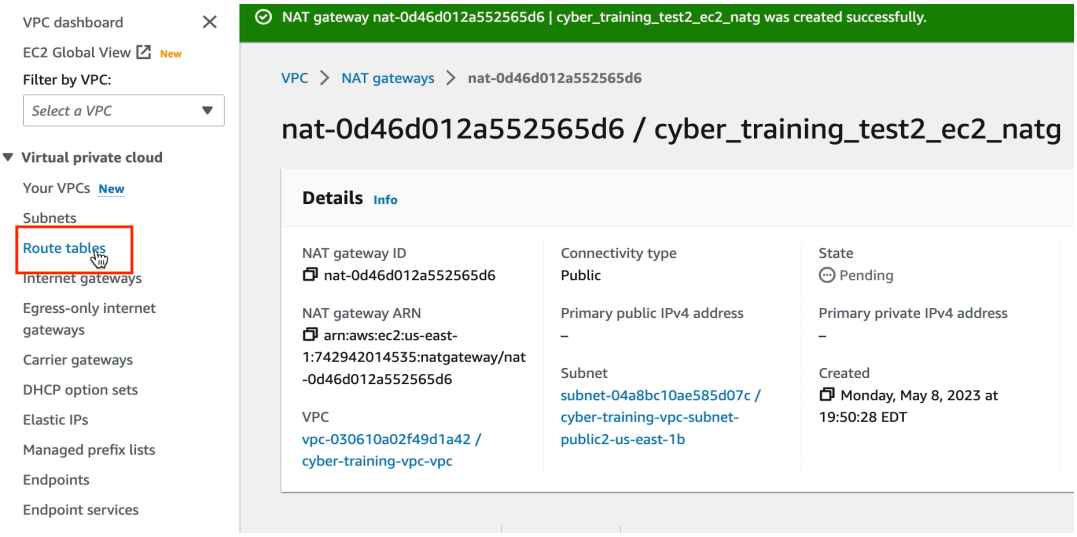
3. **In the Create NAT gateway window, type in the name of your NAT gateway and select the private subnet the test EC2 is located in.** Then select "Public" as the "Connectivity type." Since you've selected Public as the connectivity type, you will have to click "Allocate Elastic IP" allocate and attach

an Elastic IP address to the NAT gateway. After reviewing your settings, click "Create NAT gateway."

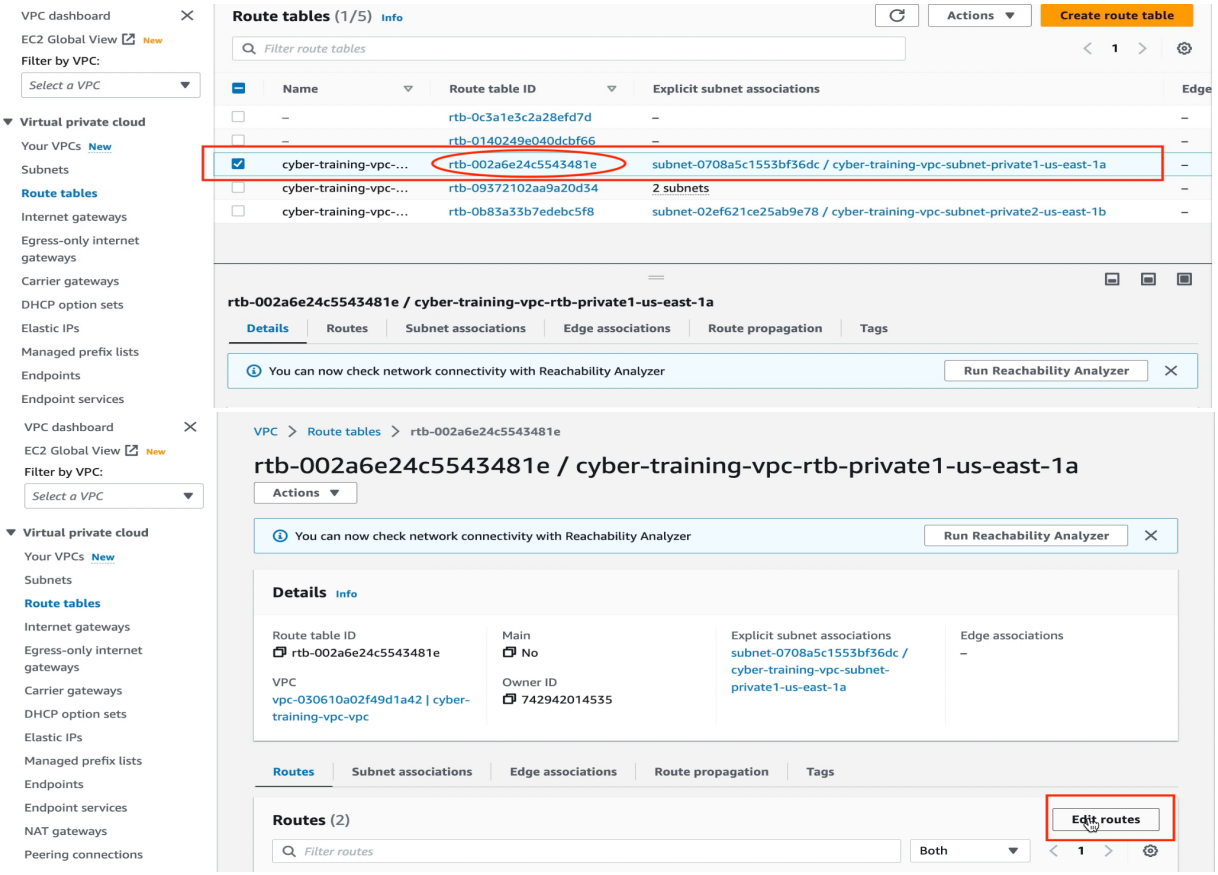


4. **After seeing the confirmation message that the NAT gateway has been created, you will need to associate the NAT gateway with the route table that the private subnet (holding the test EC2) is on.** To do this, click

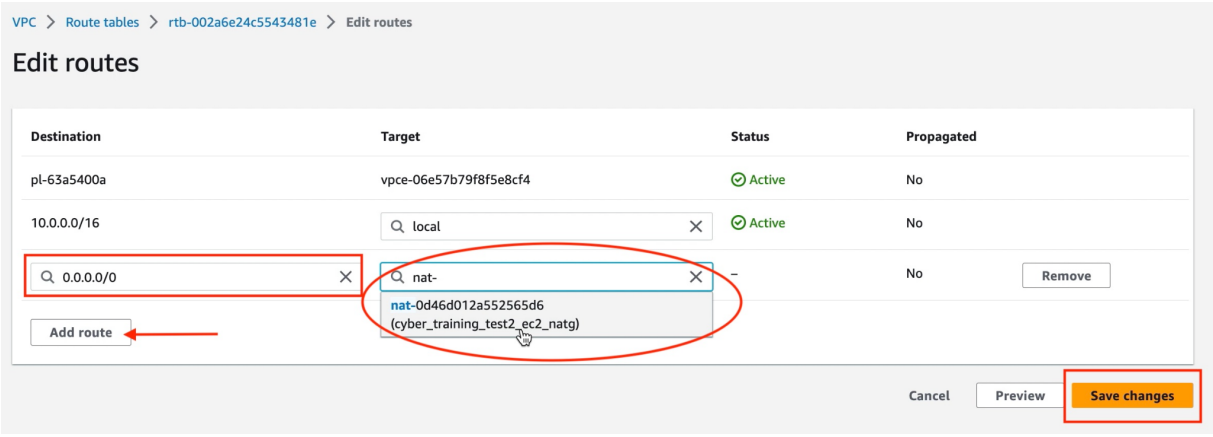
“Route Tables” in the VPC quick access menu.



5. Then click the route table that the private subnet containing the test EC2 is on. Once you’re in the route table window, select “Edit route”



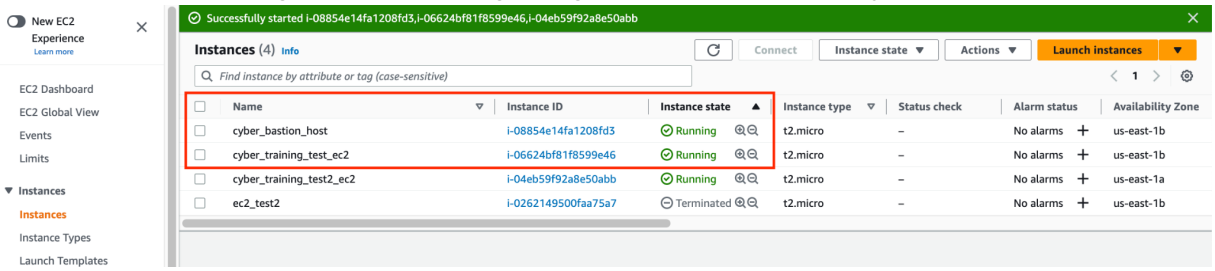
6. Click “Add Route.” In the Destination box type, 0.0.0.0/0. In the Target Box, select NAT gateway and select the NAT gateway you made earlier. Click “Save changes.”



Part 5: Testing the Bastion Host

Once the NAT gateway has been created and attached to the corresponding route table, there is now a literal (virtual) path from the private subnet to the internet. To ensure this functioning correctly, you will now SSH into the bastion host, and from the bastion host instance, SSH into the private test EC2. This is known as "jumpboxing" or using the bastion host as a "jump box."

1. **Return to the EC2 Dashboard.** Ensure the bastion host and test EC2 instances are on and running prior to SSH-ing/using the bastion host as a jump box.



2. **Open the terminal associated with your local machine.** Ensure that SSH-agent forwarding is enabled for not only your bastion host keypair, but the test EC2 keypair as well. This is done by running the command ssh-add ./<pemkey.pem> (when you're using the MacOS terminal you have to run the command with an additional -K. For example `ssh-add ./cyber_testing_host.pem` (or in my case `ssh-add -`

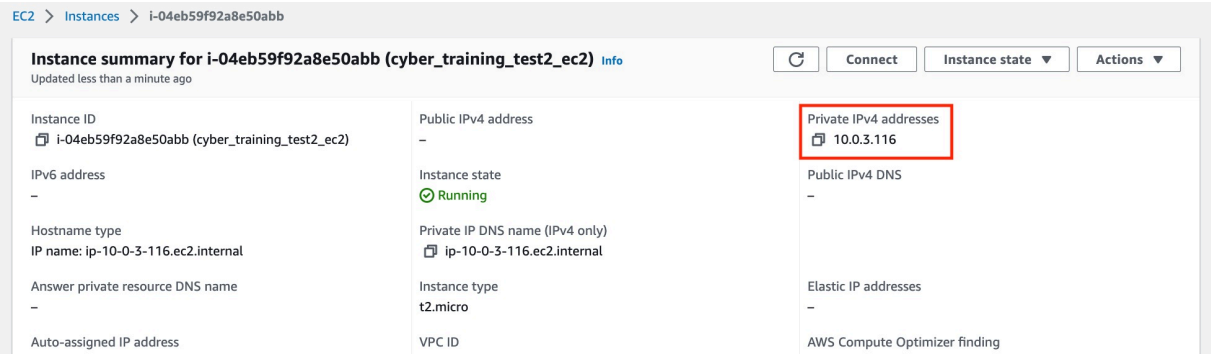
`K ./cyber_testing_host.pem`.) This will add private key identities to the SSH authentication agent and enable agent forwarding during SSH connections.

```
Downloads — -zsh — 80×24
chmod: cyber_bastion_host.pem: No such file or directory
Jane@Janes-Mac ~ % cd Downloads
Jane@Janes-Mac Downloads % chmod 400 cyber_bastion_host.pem
Jane@Janes-Mac Downloads % chmod 400 cyber_training_test.pem
Jane@Janes-Mac Downloads % eval 'ssh-agent'
SSH_AUTH_SOCK=/var/folders/5g/zldhbcls43g2wzxy53b888b00000gp/T//ssh-dZFVRnoFM1nX
/agent.29034; export SSH_AUTH_SOCK;
SSH_AGENT_PID=29035; export SSH_AGENT_PID;
echo Agent pid 29035;
Jane@Janes-Mac Downloads % ssh-add -K ./cyber_bastion_host.pem
WARNING: The -K and -A flags are deprecated and have been replaced
         by the --apple-use-keychain and --apple-load-keychain
         flags, respectively.  To suppress this warning, set the
         environment variable APPLE_SSH_ADD_BEHAVIOR as described in
         the ssh-add(1) manual page.
Identity added: ./cyber_bastion_host.pem (./cyber_bastion_host.pem)
Jane@Janes-Mac Downloads % ssh-add -K ./cyber_training_test.pem
WARNING: The -K and -A flags are deprecated and have been replaced
         by the --apple-use-keychain and --apple-load-keychain
         flags, respectively.  To suppress this warning, set the
         environment variable APPLE_SSH_ADD_BEHAVIOR as described in
         the ssh-add(1) manual page.
Identity added: ./cyber_training_test.pem (./cyber_training_test.pem)
Jane@Janes-Mac Downloads %
```

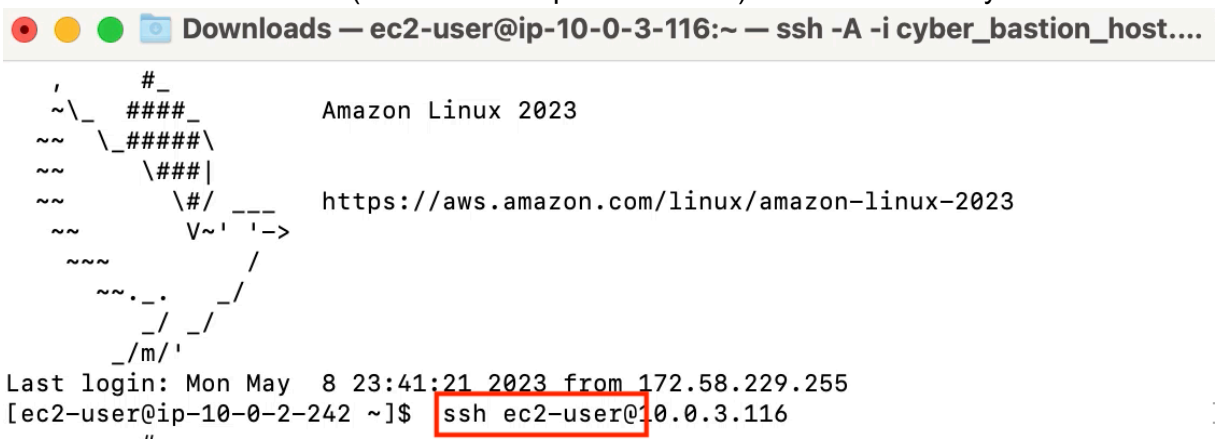
3. **With the terminal opened, input the command that will allow you to SSH into the bastion host instance.**

```
Jane@Janes-Mac Downloads % ssh -A -i "cyber_bastion_host.pem" ec2-user@ec2-52-20
6-104-53.compute-1.amazonaws.com
       #_
       ~\_  ####_        Amazon Linux 2023
      ~~  \_#####\
      ~~     \###|
      ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
       ~~       V~' '->
        ~~~         /
          ~~._.   _/
             _/ _/
           _/m/'
Last login: Mon May  8 23:41:21 2023 from 172.58.229.255
[ec2-user@ip-10-0-2-242 ~]$
```

4. **Return to the test EC2 instance information page to copy and paste the Private IP address that the bastion host will "jumpbox" into.**



5. **In the terminal window, in your bastion host instance, type the following command to jumpbox into the test EC2 instance:** `ssh ec2-user@<Private IP address>`**.** In this example the command is: `ssh ec2-user@10.0.3.116` as it utilizes the Private IP address of the test EC2 instance (which is in the private subnet). Press "Enter" key.



Once you see that the ec2-user IP address has changed from the IP address used by the bastion host to the IP address used by the test EC2, you have now SSH into the test EC2.

```
Downloads — ec2-user@ip-10-0-3-116:~ — ssh -A -i cyber_bastion_host....
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       /m/'

[ec2-user@ip-10-0-2-242 ~]$  ssh ec2-user@10.0.3.116
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login:                               from 10.0.2.242
[ec2-user@ip-10-0-3-116 ~]$
```

Congratulations! You have now successfully jumpboxed from the bastion host instance on the public subnet to the test EC2 on the private subnet. Great work!