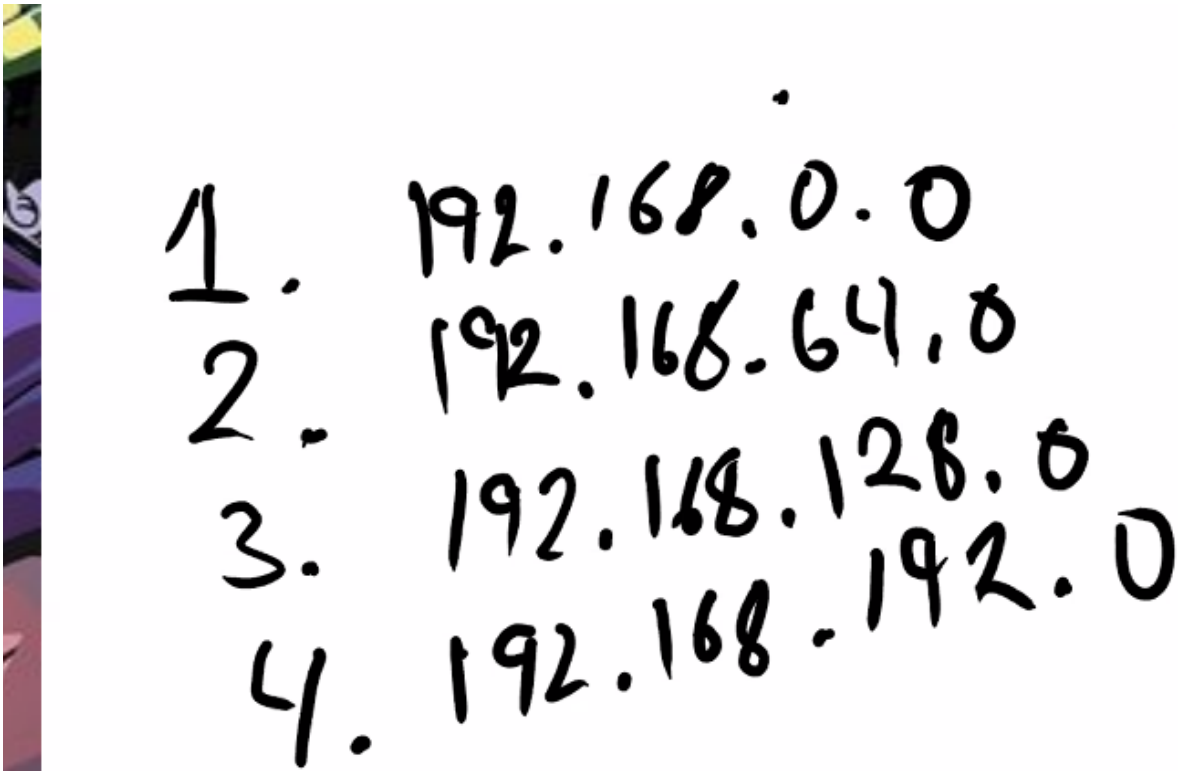
Create a vpc

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR
kura-vpc	vpc-0b8c66f9619175e9d	Available	192.168.0.0/16	–
–	vpc-bfba24d4	Available	172.31.0.0/16	–

Create 4 subnets with different IPv4 CIDR block and CIDR block size 18. For every number more than VPC CIDR block of 16, the vpc gets 2 subnets. Since its cut in 4 it will be CIDR block 18.,
16=1
17=2
18=4 subnets
19=6 subnets



Name	Subnet ID	State	VPC	IPv4 CIDR
public02	subnet-032d541c6b2b191fe	Available	vpc-0b8c66f9619175e9d | kur...	192.168.64.0/18
private02	subnet-0d3c78ea21ddd1bcb	Available	vpc-0b8c66f9619175e9d | kur...	192.168.192.0/18
public01	subnet-029df4ece29d0fbb4	Available	vpc-0b8c66f9619175e9d | kur...	192.168.0.0/18
private01	subnet-05ec11df09d56cb64	Available	vpc-0b8c66f9619175e9d | kur...	192.168.128.0/18

Then create an internet gateway called kura-IG, which is basically the router for your vpc. Your vpc will know exactly what route to take to go to the outside world

Name	Internet gateway ID	State	VPC ID
-	igw-03d5a06270d0aad10	Attached	vpc-bfba24d4
kura-ig	igw-09d00fd0c22c4ac76	Detached	–

Then attach the internet gateway to the kura-vpc.

Name	Internet gateway ID	State	VPC ID
-	igw-03d5a06270d0aad10	Attached	vpc-bfba24d4
kura-ig	igw-09d00fd0c22c4ac76	Attached	vpc-0b8c66f9619175e9d | kura-vpc

After that go to route tables and there should be one attached to kura-vpc. Its unnamed so name it whatever. We named it Kura_RT

Name	Route table ID	Explicit subnet associat...	Edge associations	Main	VPC
Kura_RT	rtb-0270ab86199901a2b	–	–	Yes	vpc-0b8c66f9619175e9d | kur.
Default-rt	rtb-aaaf88c1	–	–	Yes	vpc-bfba24d4

Our vpc has the internet gateway attached to it but there is something else. We need to tell our VPC, that if you want to send traffic out onto the outside internet, send it to the internet gateway.To do this go to routes and click edit routes and add the internet gateway. Now, Ec2 are able to route traffic to the internet.

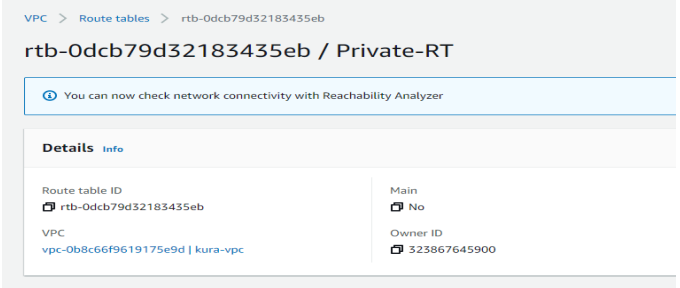
An Internet Gateway (IGW) allows resources within your VPC to access the internet, and vice versa. In order for this to happen, there needs to be a routing table entry allowing **a subnet to access** the IGW. ... The internet at large cannot get through your NAT to your private resources unless you explicitly allow it.

Edit routes

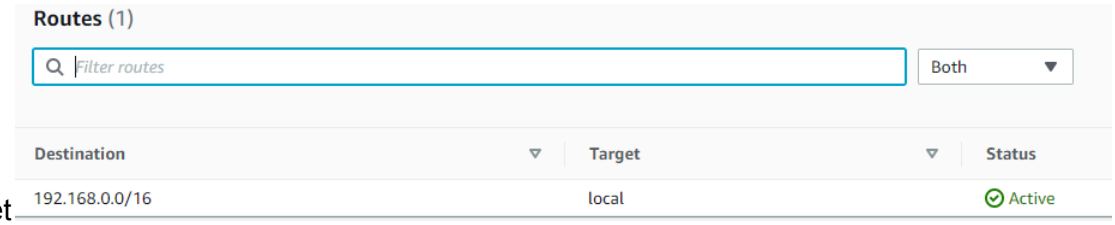
Destination	Target	Status	Propagated
192.168.0.0/16	local	Active	No
0.0.0.0/0	igw-09d00fd0c22c4ac76	-	No

Add route

Then make a private routing table to handle more private information.



This private-rt only knows how to route traffic to the vpc. It doesn't know how to traffic to outside internet

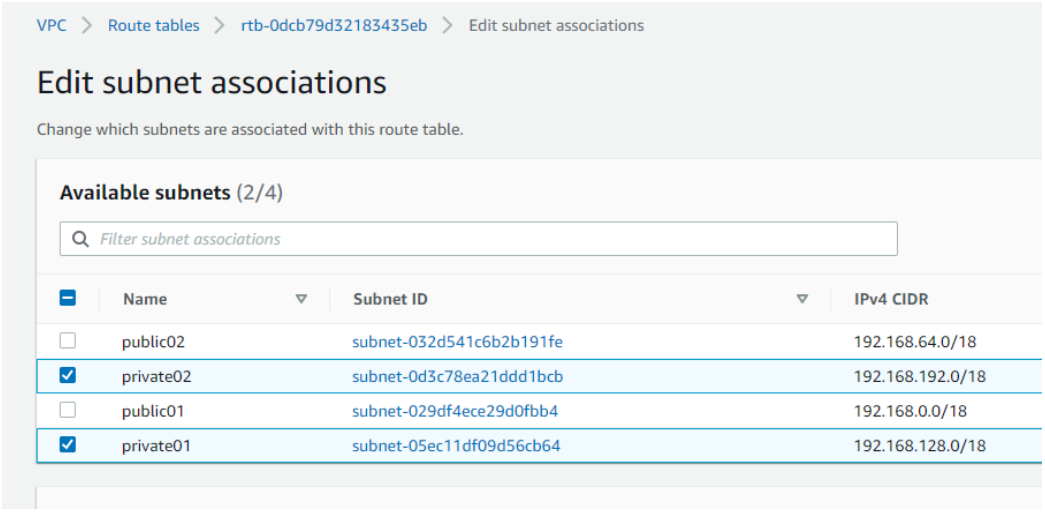


The VPC has a control list of what's allowed. Using the internet gateway(router) it allows the VPC to connect to the outside internet. The routing table Kura-RT allows subnets to be routed to the vpc and the outside internet by being attached to the internet gateway..

The subnets have routing tables which is a control list of what's allowed
The EC2 instance would have security groups of whats allowed

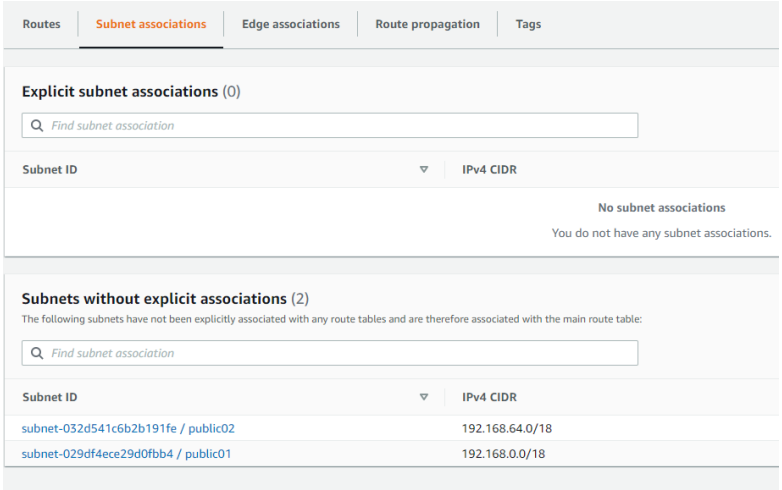
The rules you set for vpc is network firewall\
The rules for your Ec2 is like your host firewall

After that go to route tables→ Private-RT, here you click subnet associations and edit. And this will allow you to associate the private subnets to Private-RT.

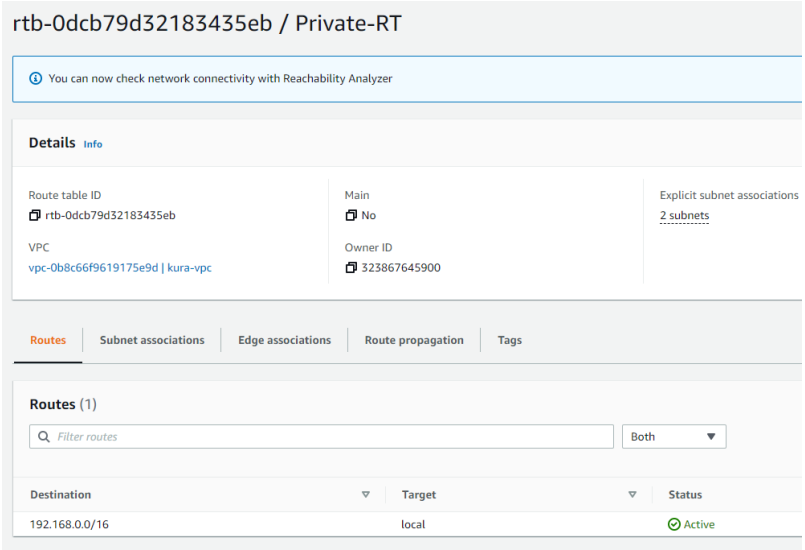


We don’t want all our devices be exposed to the outside internet. So we want to control what communication happens in our vpc especially for the private subnets. So for these private subnets, we are more restrictive. For this we have created a routing table called Private-RT.Anything associated with this private-rt, only jknows how to send traffic to the VPC. Kura-rt knows how to route traffic to the vpc and it knows how to route traffic to the internet.

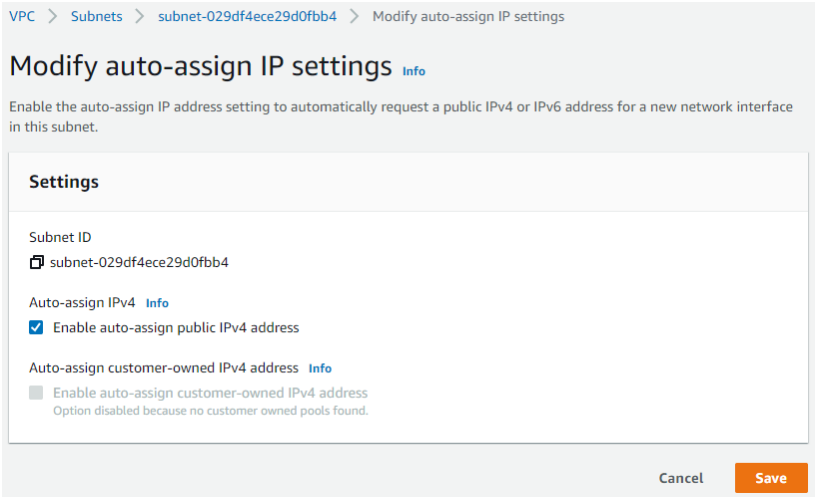
The kura-rt subnet associations doesn't have any associations because kura-rt is for your entire VPC. Anything that's not placed in the private-part is going to be able to leave the network. So anything associated with kura-rt is public. ANd anything associated with private rt will not be able to leave the network but only stay in it. Basically anything not place in your private rt is public.
Public(kura-rt subnet associations)



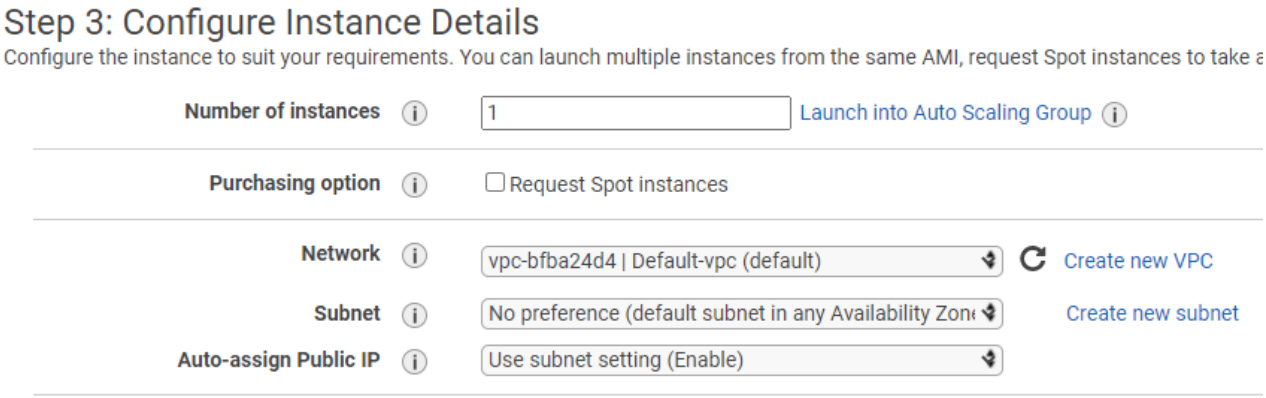
Private(Private-rt subnet associations)

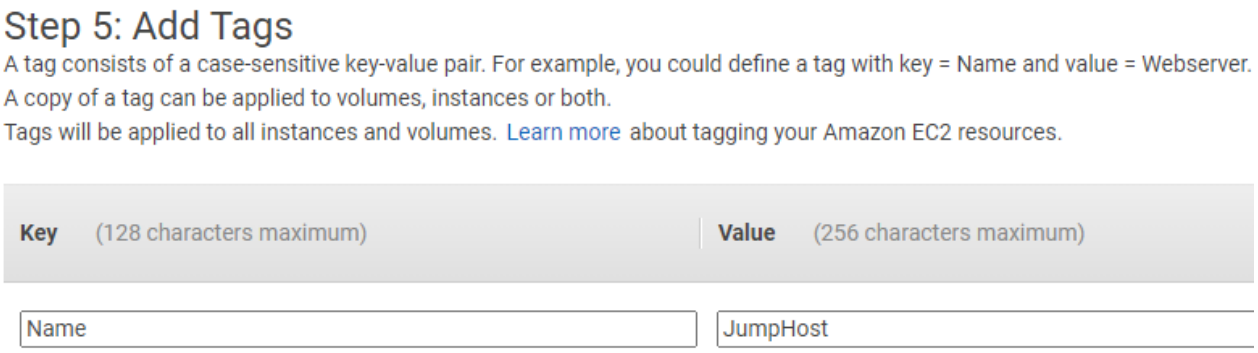


Then go to routing tables and click on public-02 and go to actions and modify auto assign ip setting and enable auto assign. This auto assigns the public ip. Do this with the other public-02 as well

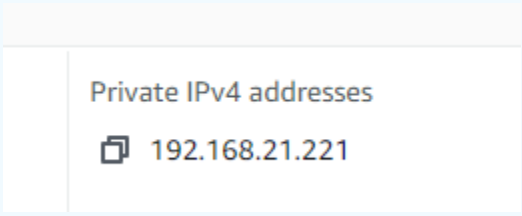


Then make the public ec2 called Jumphost using these settings. Subnet set it to 2a public01. And with auto-assign Public IP your vpc will give your ec2 a public ip. This is because you modified it before to do so for public subnets.





Your private IPV4 should be a ip between your public subnet's ipv4 CIDRs So between 92.168.0.0/18 and 192.168.64.0/18, ANd if I check I check its just that.



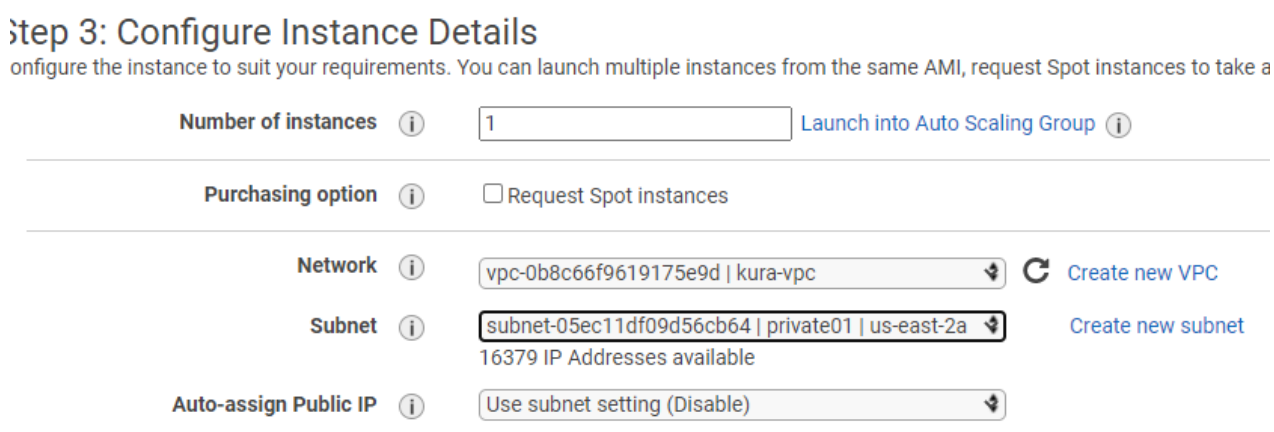
Now check if you can connect to the outside world with this public Jumphost EC2

```
PS C:\Users\Bishajit Lodh\downloads> ssh -i "Python.pem" ec2-user@18.219.112.243
The authenticity of host '18.219.112.243 (18.219.112.243)' can't be established.
ECDSA key fingerprint is SHA256:Fi6VswvxSx1KydkykOsCgwzgq9h/elSm051KdihTK2c.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '18.219.112.243' (ECDSA) to the list of known hosts.

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-192-168-21-221 ~]$
```

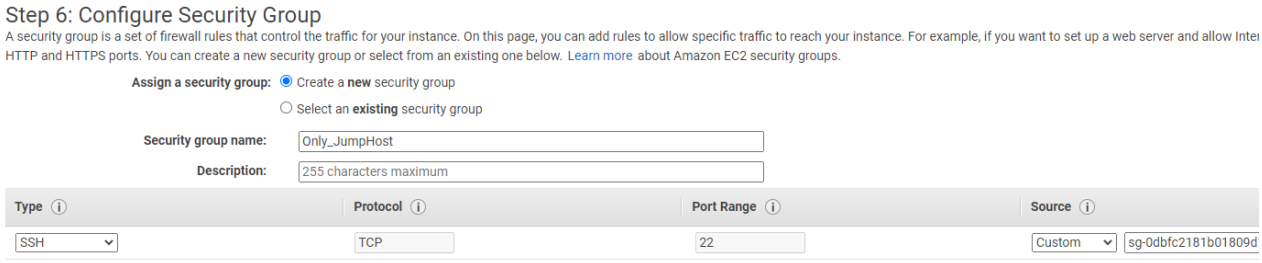
If you are able to ping 8.8.8.8, your able to connect to the outside world, it works. This means traffic coming in and out of the VPC

```
[ec2-user@ip-192-168-21-221 ~]$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=97 time=11.3 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=97 time=11.4 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=97 time=11.3 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=97 time=11.4 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=97 time=11.3 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=97 time=11.4 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=97 time=11.9 ms
```

Now create your private ec2



Configure security group to have (SSH_public) the old security group to only have access to the ec2.SSH public was the old security group that was created for the Jumpost ec2(the ec2 that was set to public-01 subnet so it could take all public traffic.



For security purposes have a separate key for each EC2 but for now due to it being learning purpose you can have one key for all the ec2s.

Go back to your Jumpost Ec2 and create linex.pem file and put your rsa private key inside it

```
[ec2-user@ip-192-168-21-221 ~]$ nano linux.pem
[ec2-user@ip-192-168-21-221 ~]$ [ec2-user@ip-192-168-21-221 ~]$ ls
linux.pem
```

```
GNU nano 2.9.8                                    linex.pem

-----BEGIN RSA PRIVATE KEY-----
IIEpAIBAAKCAQEAqt9ZF9s21xTvJ3oD4XbP78+kKdxizcTPIXwFFq0YEd7THe1Y
8TMLG4+PwSO8TXGyWew7UsSjpZdOO9qabKyaAmIcpTP41n5Lw6bRv0Otkg4V/FJ
CsFxvxfB//YTXFxK/s1BtSGOVpvOniBC/ra8k0yI1QD2TsHNOdFEjSzRepsrJgc
X3n/QHWH1Y1UOxsfPjX+d3s27t1ymNGpWQxVVhBrAdA5emmI38AbaYwoR+FDxNN
LX0jD0QIIF3/+IE3u5BbhGALzSdMSB5VwxLCO0tlt/JPyjPKJyY8BErSKOHd0Nm
uMcDBxI9AgivGaNnk1KLcDUz9fOMhB0ogkH8QIDAQABAoIBAH46bLig6oxNHLcO
hrrCIVh4SZDdoIH5/QWiDPOPz/EWjV4kfsH6MTR1A39shMi5O0YOq2xjc3j6o4K
5OhvQX6NOBgaorLhrdqlO8GNn4WL/1XeBU4AxaSjS19qS7cpPwWoX+gNF/eHNkW
E3/TQs+V9DgO8+YOaStTGs8n0r/wFTKq34t072oWz62SRYnJhEbTx1Z0Y1/neb5
dvXb6YoGI/auTOjC1NHNcco9FtqmVa8TGJAU2uLO0EHMny3f/2MHNpzzgAOybVm
Y0ucAoHE1vDjkPsdvCPkWc4s420PYX4FAtN4icGgNiO8V1owdLn8nMKo4x8Lq6Q
yTOFAECgYEA81noQOLCicU8RL3D6B8K1z1/7te6yJZM7ubHTp+VNyOZLXyUB+Ct
rkfsukxRmPhn+JQVuJgR0OsZIR8VLWWFe3FBQ1HCl182UVWG0bkOenvd5iww4ib
Yy9BwYWMv9FuqcIQVS1JRJDiPZwkKZqtwtetW8UHACtl2cOlnD6ZZECgYEAtH71
kf5goP6WxTDo1dBK2ehb0pBdPHDUzRTLHBRT9jZuaYvOkGAlLdb1sugKRu70t9M
M7M5U11CCSvLCyO+15aInsaAaNUaQC9GcyvZ7MoaEJGMWcuTHMB0ECKNdV1prAS
MoQP/vtNB0xUSaWKMSh++R/UYCUgyeHFqYpzGECgYEA0vtOZF6E+aeJp1aGHKgZ
j0KCQYqB8JBS6IBVQM8xfnpul3ACXOiCvJrFDhKu3rsy2Iciucc5i2WARQNoT1M
d7PLGk9PVrR+9yO78Qxn/F21emJyVA9qeYCEgFMhi0RqUMMvsNP2gwiWzQgHu2h
aSqWM1Q9mfot+vyGSpV8uECgYAZW2Z+94OXEWF/oDgn3185cNKeJulxO3G8Y7J3
MPbmtRjn3QtE/yzZ9sQHwU+ITZO1q4Mr8TbRfc2oKi3ZP81L1IsW+z4Y/C826y1
isqP2Ii24Su9hbsCJAtqJOCIpuoQj4vpZeI1aSqWJuQEgKwcYdsMtqsIYs7rbbt
```

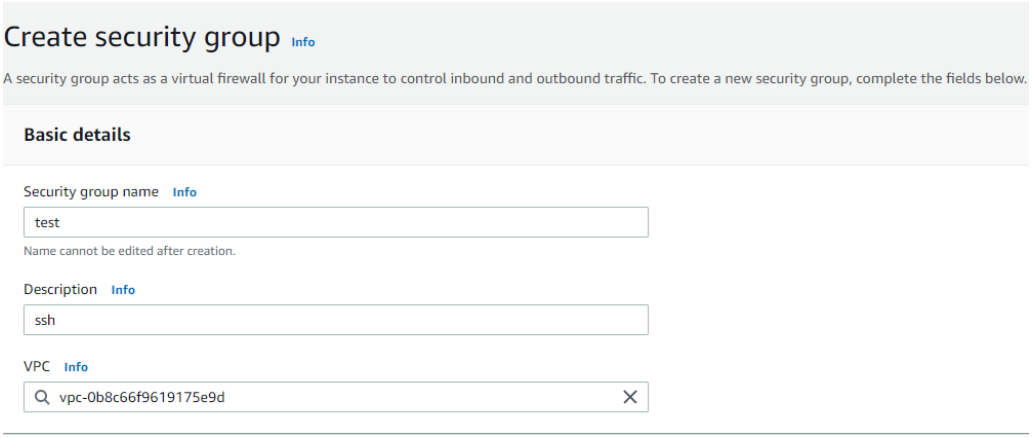
Chmod 400 the file

```
[ec2-user@ip-192-168-21-221 ~]$ [ec2-user@ip-192-168-21-221 ~]$ chmod 400 linex.pem
```

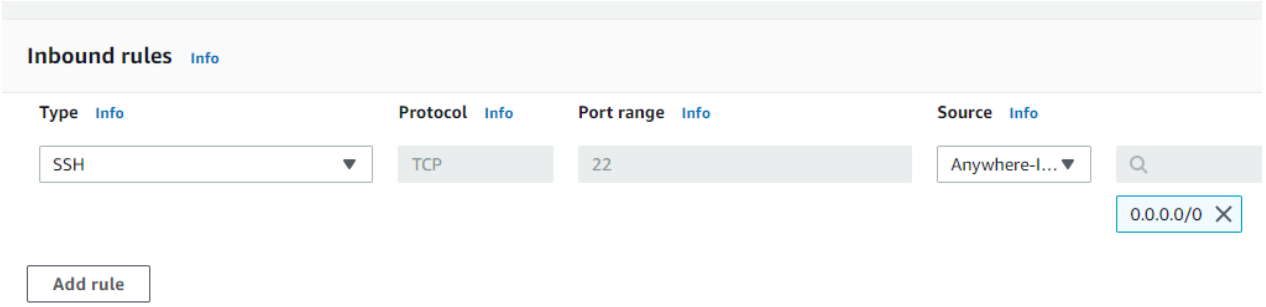
Now you should be able to ssh into the private ec2 using linex.pem

```
[ec2-user@ip-192-168-21-221 ~]$ ssh -i linex.pem ec2-user@192.168.171.110
The authenticity of host '192.168.171.110 (192.168.171.110)' can't be established.
ECDSA key fingerprint is SHA256:wruzharY186CMDDNE4eotmNz5GTexkr3PCIV174krTs.
ECDSA key fingerprint is MD5:1e:40:36:f7:83:95:4b:a6:b7:4d:a6:5f:a7:f7:9d:98.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.171.110' (ECDSA) to the list of known hosts.

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-192-168-171-110 ~]$
```

Create a new security group



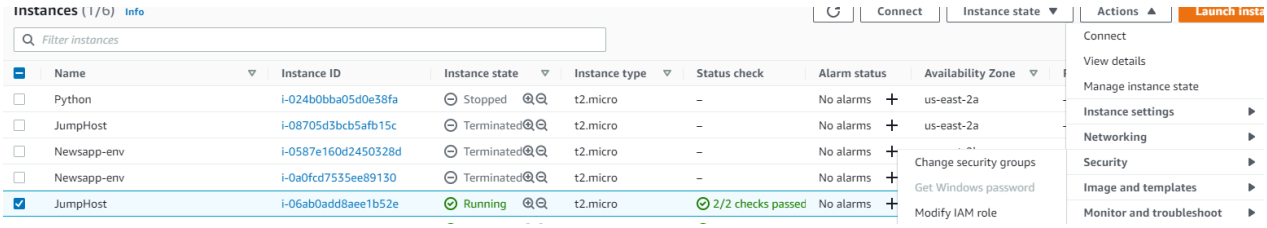
Set your ec2 inbound rules to let you ssh into anywhere from ipv4

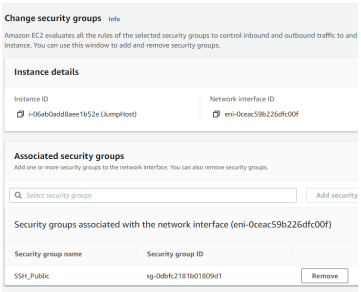


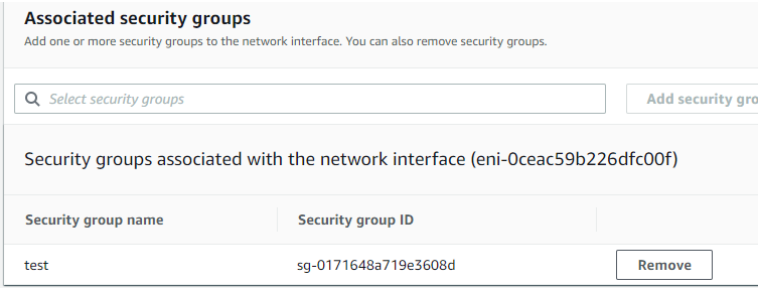
And set outbound rules to this

Name	Security group rule...	IP version	Type	Protocol	Port range	Destination
–	sgr-0fb18ee520cc10acb	IPv4	All traffic	All	All	0.0.0.0/0

Change your security group for Jumphost by removing ssh _public and addding test







Changing the security group will make it so you can't ssh into the privatee01 ec2 from the jumphost ec2 that is configured to the public-01 subnet.

To be able to get back into to your private excc2 you would need a NAT gateway

NAT Gateway

NAT Gateway is **a highly available AWS managed service** that makes it easy to connect to the Internet from instances within a private subnet in an Amazon Virtual Private Cloud (Amazon VPC). Previously, you needed to launch a NAT instance to enable NAT for instances in a private subnet

VPC > NAT gateways > Create NAT gateway

Create NAT gateway Info

A highly available, managed Network Address Translation (NAT) service that instances in private subnets can use to connect services in other VPCs, on-premises networks, or the internet.

NAT gateway settings

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

kura-nat

The name can be up to 256 characters long.

Subnet
Select a subnet in which to create the NAT gateway.

subnet-029df4ece29d0fbb4 (public01)

Connectivity type
Select a connectivity type for the NAT gateway.

- (●) Public
- ( ) Private

Elastic IP allocation ID Info
Assign an Elastic IP address to the NAT gateway.

eipalloc-058e03e95c372c8d9 | Allocate Elastic IP

Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - *optional*	
Name	kura-nat	Remove

Add new tag

You can add 49 more tags.

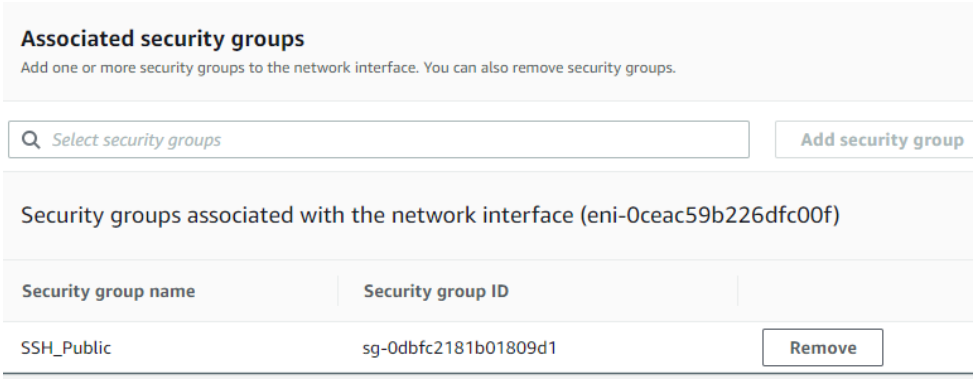
Then go to add a route for your private-rt so you can add your NAT gateway

Edit routes

Destination	Target	Status	Propagated
192.168.0.0/16	local	Active	No
0.0.0.0/0	nat-0d6ff876c19d7f0ff	-	No

Add route

Go back to jumpohost security groups and remove test and add back ssh_public



Now you are able to ssh into the private again

There was a reason we add ed nat gateway to the private-rt because without it wont be able to reachout to the outside world or internet. For example if you ping it you get this

```
[ec2-user@ip-192-168-171-110 ~]$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
```

Now with the NAT gateway added as a route in the private-rt, your private-02 instance will be able to reach out to the outside internet.

```
[ec2-user@ip-192-168-171-110 ~]$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=392 ttl=96 time=12.0 ms
64 bytes from 8.8.8.8: icmp_seq=393 ttl=96 time=11.5 ms
64 bytes from 8.8.8.8: icmp_seq=394 ttl=96 time=11.5 ms
64 bytes from 8.8.8.8: icmp_seq=395 ttl=96 time=11.5 ms
64 bytes from 8.8.8.8: icmp_seq=396 ttl=96 time=11.6 ms
64 bytes from 8.8.8.8: icmp_seq=397 ttl=96 time=11.6 ms
64 bytes from 8.8.8.8: icmp_seq=398 ttl=96 time=11.5 ms
64 bytes from 8.8.8.8: icmp_seq=399 ttl=96 time=11.6 ms
64 bytes from 8.8.8.8: icmp_seq=400 ttl=96 time=11.5 ms
64 bytes from 8.8.8.8: icmp_seq=401 ttl=96 time=11.6 ms
```

Then follow these instructions
https://github.com/kura-labs-org/DEPLOY5_AWS/blob/main/Deployment%235.pdf

Step1:You start the instruction with following the commands below. Then in your private01 use these commands
sudo amazon-linux-extras install java-openjdk11

sudo amazon-linux-extras install epel

sudo wget -O /etc/yum.repos.d/jenkins.repo \
https://pkg.jenkins.io/redhat-stable/jenkins.repo

sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key

sudo yum upgrade

sudo yum install epel-release java-11-openjdk-devel

sudo yum install jenkins

sudo systemctl start jenkins

Step 2 :Create a target group, my one was called test1

Protocol: HTTP

Port: 8080

VPC

Select the VPC with the instances that you want to include in the target group.

kura-vpc
vpc-0b8c66f9619175e9d
IPv4: 192.168.0.0/16

Protocol version

- HTTP1
  Send requests to targets using HTTP/1.1. Supported when the request protocol is HTTP/1.1 or HTTP/2.
- HTTP2
  Send requests to targets using HTTP/2. Supported when the request protocol is HTTP/2 or gRPC, but gRPC-specific features are not available.
- gRPC
  Send requests to targets using gRPC. Supported when the request protocol is gRPC.

Health checks

The associated load balancer periodically sends requests, per the settings below, to the registered targets to test their status.

Health check protocol

HTTP

Health check path

Use the default path of "/" to ping the root, or specify a custom path if preferred.

/login

Up to 1024 characters allowed.

▶ **Advanced health check settings**

▶ **Tags - *optional***

Consider adding tags to your target group. Tags enable you to categorize your AWS resources so you can more easily manage them.

▼ **Advanced health check settings** Restore defaults

Port

The port the load balancer uses when performing health checks on targets. The default is the port on which each target receives traffic from the load balancer, but you can specify a different port.

- ○ Traffic port
- ◉ Override

8080

1-65535

Register targets

Available instances (1/2)

Filter resources by property or value

	Instance ID	Name	State	Security groups
☐	i-06ab0add8aee1b52e	JumpHost	running	SSH_Public
☑	i-01e42a1563da99563	Private01	running	Only_JumpHost

1 selected

Targets (1)

All | Filter resources by property or value

Remove	Health status	Instance ID	Name	Port	State	Security groups
×	Pending	i-01e42a1563da99563	Private01	8080	running	Only_JumpHost

Then create a load balancer as instructed in step 2

VPC Info

Select the virtual private cloud (VPC) for your targets. Only VPCs with an internet gateway are enabled for selection. The selected VPC cannot be changed after the load balancer is created. To confirm the VPC for your targets, view your target groups.

kura-vpc
vpc-0b8c66f9619175e9d
IPv4: 192.168.0.0/16

Mappings Info

Select at least two Availability Zones and one subnet per zone. The load balancer routes traffic to targets in these Availability Zones only. Availability Zones that are not supported by the load balancer or the VPC are not available for selection. Subnets cannot be removed after the load balancer is created, but additional subnets can be added. Availability Zones that are not supported by the load balancer or the VPC are disabled. At least two subnets must be specified.

us-east-2a

Subnet

subnet-029df4ece29d0fbb4 public01

IPv4 settings

Assigned by AWS

us-east-2b

Subnet

subnet-032d541c6b2b191fe public02

IPv4 settings

Assigned by AWS

Listeners and routing Info

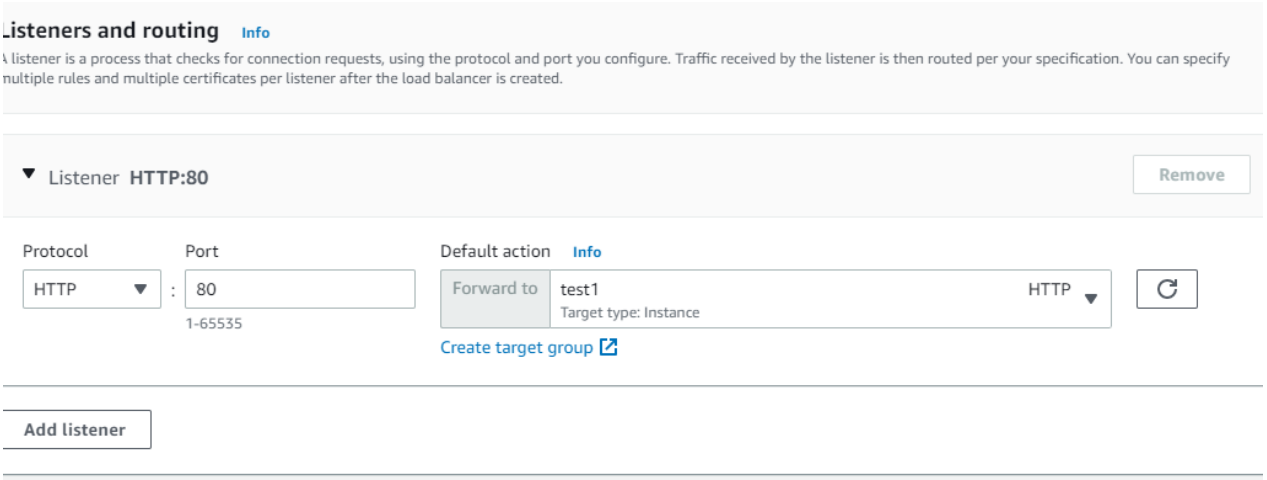
A listener is a process that checks for connection requests, using the protocol and port you configure. Traffic received by the listener is then routed per your specification. You can specify multiple rules and multiple certificates per listener after the load balancer is created.

Listener **HTTP:80** Remove

Protocol	Port	Default action Info
HTTP	80 (1-65535)	Forward to *Select a target group*

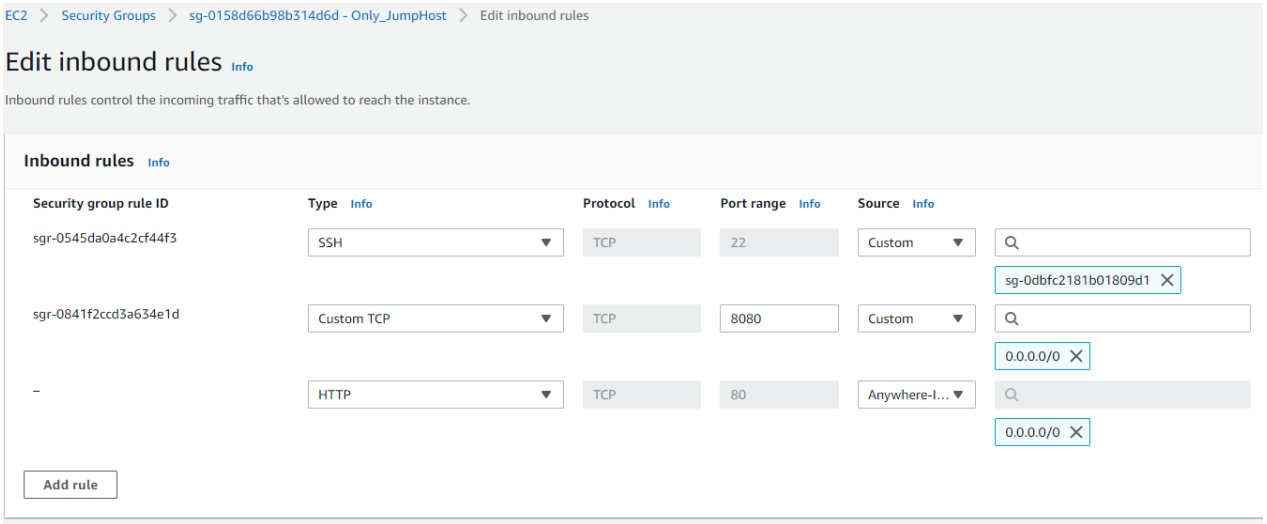
Create target group

Add listener

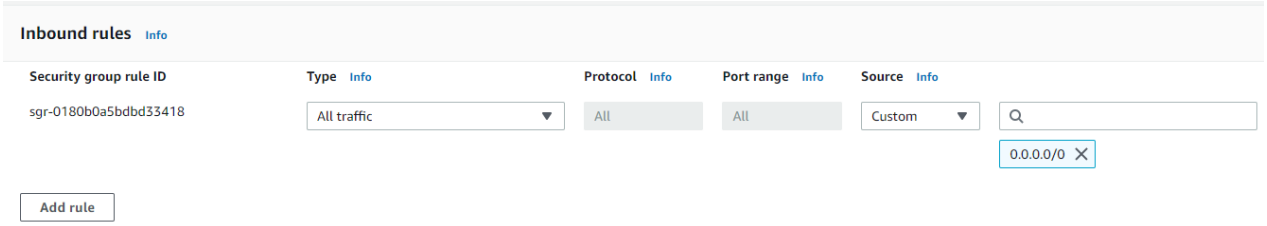


Then click Load balancer. In a few minutes, you will see load balancer as healthy and the target group is healthy as well\

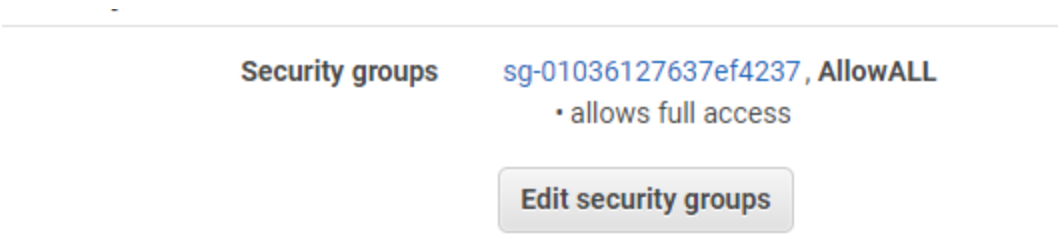
However, it didn't work for me because my target group kept saying unhealthy so I had to change my Only_Jumphost to allow both custom TCP 8080 and HTTP 80.



Then I tried to go into the DNS link and it failed for me. So I had to create a new security group for my Load balancer called Allow All.

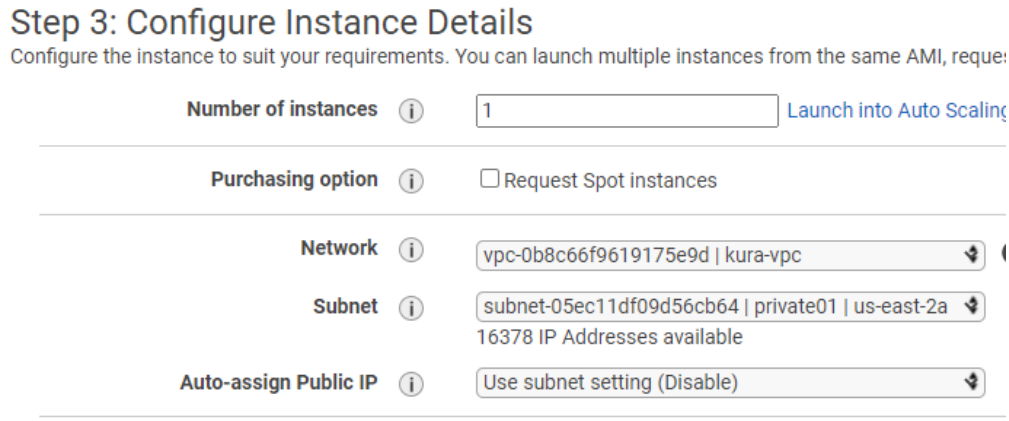


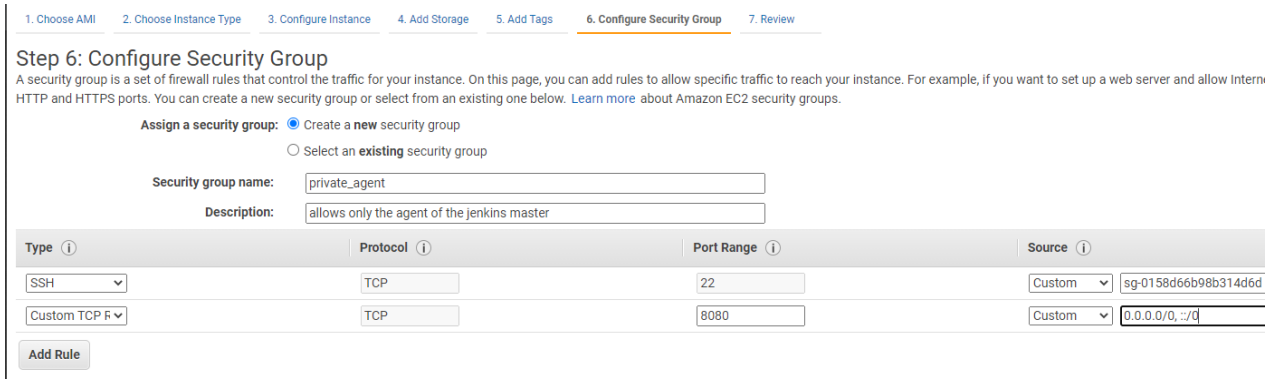
And then I edited my Load balancer where I took out the SSh_Public security group and replaced it with the All traffic ssh group.



http://alb-1078231393.us-east-2.elb.amazonaws.com/
This allows you to create a new jenkins

Step 4: After that create a new Jenkins with the same





Put the security group that has jenkins and another custom tcp rule that would allow all ipv4 and 6.

Then go back to your private01 ec2 instance by ssh into it. If you got out of it then go into public (jumphost) ec2 instance and then ssh into the private.

One in the private ec2, type nano linex.pem

```
ec2-user@ip-192-168-171-110:~
GNU nano 2.9.8                                                    linux.p
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAqt9ZF9s21xTvJ3oD4XbP78+kKdxizcTPIXwFFq0YEd7THelY
8TMLG4+PwSO8TXGyWew7UsSjpZdOO9qabKyaAmIcpTP41n5Lw6bRv0Otkg4V/FJ
CsFxvxfB//YTXFxK/s1BtSGOVpvOniBC/ra8k0yIlQD2TsHNOdFEjSzRepsrJgc
X3n/QHWH1Y1UOxsfPjX+d3s27t1ymNGpWQxVVhBrAdA5emmI38AbaYwoR+FDxNN
LX0jD0QIIF3/+IE3u5BbhGALzSdMSB5VwxLCO0t1t/JPyjPKJyY8BErSKOHd0Nm
uMcDBxI9AgivGaNnk1KLcDUz9fOMhB0ogkH8QIDAQABAoIBAH46bLig6oxNHLcO
hrrCIVh4SZDdoIH5/QWiDPOPz/EWjV4kfsH6MTR1A39shMi5O0YOq2xjc3j6o4K
5OhvQX6NOBgaorLhrdqlO8GNn4WL/lXeBU4AxaSjS19qS7cpPwWoX+gNF/eHNkW
E3/TQs+V9DgO8+YOaStTGs8n0r/wFTKq34t072oWz62SRYnJhEbTxlZ0Y1/neb5
dvXb6YoGI/auTOjC1NHNcco9FtqmVa8TGJAU2uLO0EHMny3f/2MHNpzzgAOybVm
Y0ucAoHE1vDjkPsdvCPkWc4s420PYX4FAtN4icGgNiO8V1owdLn8nMKo4x8Lq6Q
yTOFAECgYEA8lnoQOLCicU8RL3D6B8K1z1/7te6yJZM7ubHTp+VNyOZLXyUB+Ct
rkfsukxRmPhn+JQVuJgR0OsZIR8VLWWFe3FBQ1HCl182UVWG0bkOenvd5iww4ib
Yy9BwYWMv9FuqcIQVSlJRJDiPZwkKZqtwtetW8UHACtl2cOlnD6ZZECgYEAtH71
kf5goP6WxTDo1dBK2ehb0pBdPHDUzRTLHBRT9jZuaYvOkGA1Ldb1sugKRu70t9M
M7M5U11CCSvLCyO+l5aInsaAaNUaQC9GcyvZ7MoaEJGMWcuTHMB0ECKNdV1prAS
MoQP/vtNB0xUSaWKMSh++R/UYCUgyeHFqYpzGECgYEA0vtOZF6E+aeJplaGHKgZ
j0KCQYqB8JBS6IBVQM8xfnpul3ACXOiCvJrFDhKu3rsy2IciucC5i2WARQNoT1M
d7PLGk9PVrR+9yO78Qxn/F21emJyVA9qeYCEgFMhi0RqUMMvsNP2gwiWzQgHu2h
```

And then crtl O and the crtl x.Then chmod 400 linex.pem After that ssh into the private ssh -i linux.pem ec2-user@192.168.146.135

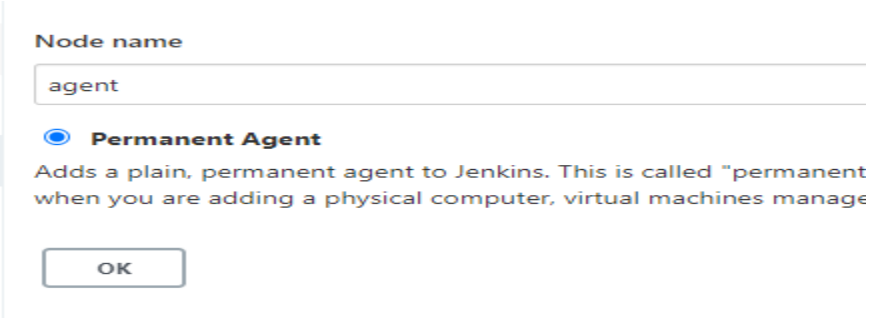
```
[ec2-user@ip-192-168-171-110 ~]$ ssh -i linux.pem ec2-user@192.168.146.135
The authenticity of host '192.168.146.135 (192.168.146.135)' can't be established.
ECDSA key fingerprint is SHA256:8mo3RTieJpSaPYoH08J+GxkI4/vA6XIlwRPtu9jiBkk.
ECDSA key fingerprint is MD5:4f:83:8f:b8:ec:ca:70:97:20:88:1b:9d:41:ba:b8:e5.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.146.135' (ECDSA) to the list of known hosts.
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0664 for 'linux.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "linux.pem": bad permissions
Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
[ec2-user@ip-192-168-171-110 ~]$ chmod 400 linux.pem
[ec2-user@ip-192-168-171-110 ~]$ ssh -i linux.pem ec2-user@192.168.146.135

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-192-168-146-135 ~]$
```

After that install Jenkins into the new agent, same way it was installed in private

Then click on the DNS link from Load balancer and set up jenkins with install plugins.

And then do these steps

Step 5: Configure the Jenkins master to SSH into the Agent - Select build executor status - Select new node - Create a name for the node and select permanent agent - Create a name and description - Enter 2 for executors - enter {/home/ec2-user/jenkins} for remote root directory - Create a label - Select use this node as much as possible - Select launch agent via ssh - Enter the private IP address of the Agent - Add SSH credentials (username: ec2-user | key: the private key you used to ssh into the agent) - Select non verifying verification strategy - save and then look at the logs to see if your setup was successful



Enter ip address of the agent ec2 in Host

Name

Private-agent

Description

agent

Number of executors

2

Remote root directory

{/home/ec2-user/jenkins}

⚠ **Are you sure you want to use a relative path for the FS root? N recommended.**

Labels

agent-linux

Usage

Use this node as much as possible

Launch method

Launch agents via SSH

After doing the above, for username, put ec2-user and for private key put directly so you dnt need a password

Username

ec2-user

☐ Treat username as secret

Private Key

◉ Enter directly

Key


```
uKRQOQKBgQDhHYNXkMqD13agJZh1WN9BY3OJZAa9RKABZg9QWeAChyJyYIgIgZ0B
Wofm35D6M4C0PmxkhzPlyvZDgCthwXnGdX4QG+s7u5z7rHfO3TBnDLXM4plToTGx
kB1WqhibWZDbbTMAaDXQB8xiQ+XUMKQjGl9XqBuu5lG0DjEzOE2OQg==
-----END RSA PRIVATE KEY-----
```


Credentials

ec2-user (jenkins ssh cred for agent) | Add

Host Key Verification Strategy

Non verifying Verification Strategy

Availability

Keep this agent online as much as possible

Node Properties

☐ **Disable deferred wipeout on this node**

☐ **Environment variables**

☐ **Tool Locations**

Save

Save

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space
	agent		N/A	N/A	N/A	N/A
	master	Linux (amd64)	In sync	5.90 GB	0 B	5.90 GB
	Data obtained	**36 min**	**36 min**	**36 min**	**36 min**	**36 min**

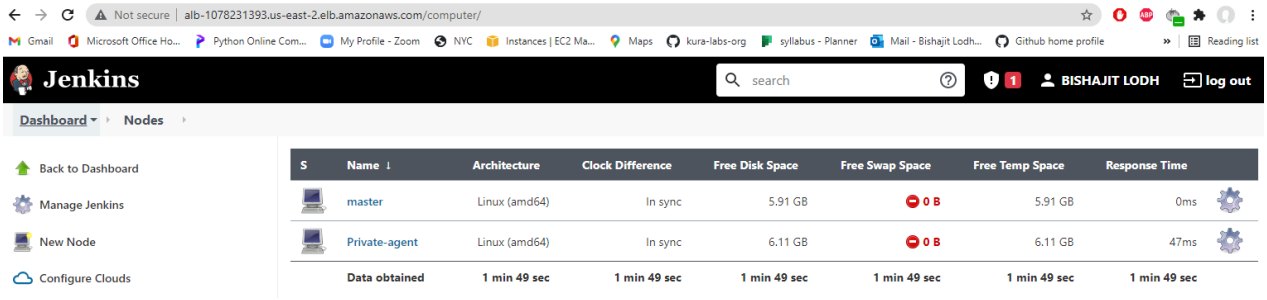
Click on agent and then logs to see if its built successfully.

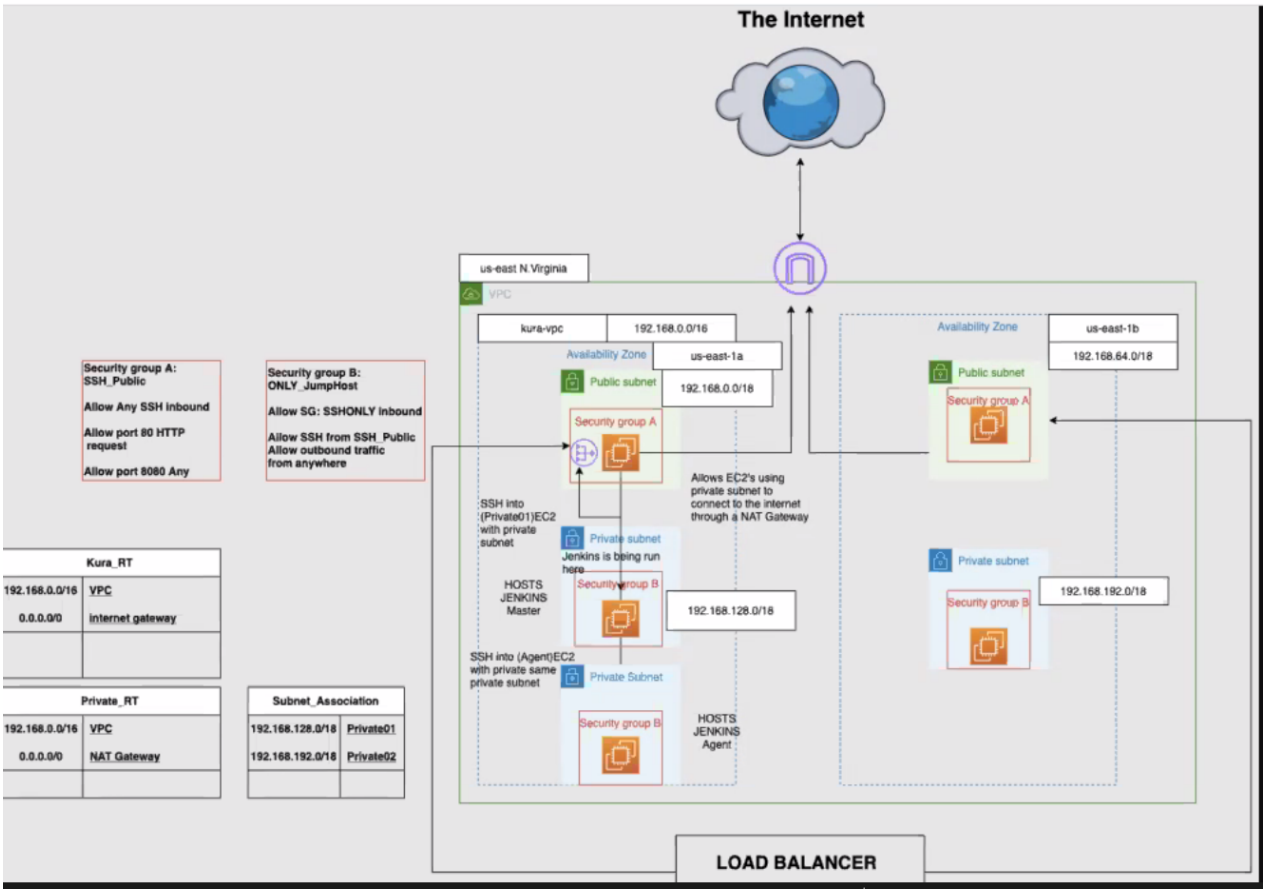
```
USER=ec2-user
XDG_RUNTIME_DIR=/run/user/1000
XDG_SESSION_ID=5
_=/etc/bashrc
Checking Java version in the PATH
bash: java: command not found
Java is not in the PATH nor configured with the javaPath setting, Jenkins will try to guess where is Java, this guess wi
[09/18/21 09:30:01] [SSH] Checking java version of /home/ec2-user/jenkins/jdk/bin/java
Couldn't figure out the Java version of /home/ec2-user/jenkins/jdk/bin/java
bash: /home/ec2-user/jenkins/jdk/bin/java: No such file or directory

[09/18/21 09:30:01] [SSH] Checking java version of java
Couldn't figure out the Java version of java
bash: java: command not found

[09/18/21 09:30:01] [SSH] Checking java version of /usr/bin/java
Couldn't figure out the Java version of /usr/bin/java
bash: /usr/bin/java: No such file or directory

[09/18/21 09:30:01] [SSH] Checking java version of /usr/java/default/bin/java
Couldn't figure out the Java version of /usr/java/default/bin/java
bash: /usr/java/default/bin/java: No such file or directory

[09/18/21 09:30:01] [SSH] Checking java version of /usr/java/latest/bin/java
Couldn't figure out the Java version of /usr/java/latest/bin/java
bash: /usr/java/latest/bin/java: No such file or directory

[09/18/21 09:30:01] [SSH] Checking java version of /usr/local/bin/java
Couldn't figure out the Java version of /usr/local/bin/java
bash: /usr/local/bin/java: No such file or directory

[09/18/21 09:30:01] [SSH] Checking java version of /usr/local/java/bin/java
Couldn't figure out the Java version of /usr/local/java/bin/java
bash: /usr/local/java/bin/java: No such file or directory

java.io.IOException: Java not found on hudson.slaves.SlaveComputer@42132d6a. Install Java 8 or Java 11 on the Agent.
        at hudson.plugins.sshslaves.JavaVersionChecker.resolveJava(JavaVersionChecker.java:84)
        at hudson.plugins.sshslaves.SSHLauncher$1.call(SSHLauncher.java:453)
        at hudson.plugins.sshslaves.SSHLauncher$1.call(SSHLauncher.java:421)
        at java.base/java.util.concurrent.FutureTask.run(FutureTask.java:264)
        at java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1128)
        at java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:628)
        at java.base/java.lang.Thread.run(Thread.java:829)
[09/18/21 09:30:02] Launch failed - cleaning up connection
[09/18/21 09:30:02] [SSH] Connection closed.
```

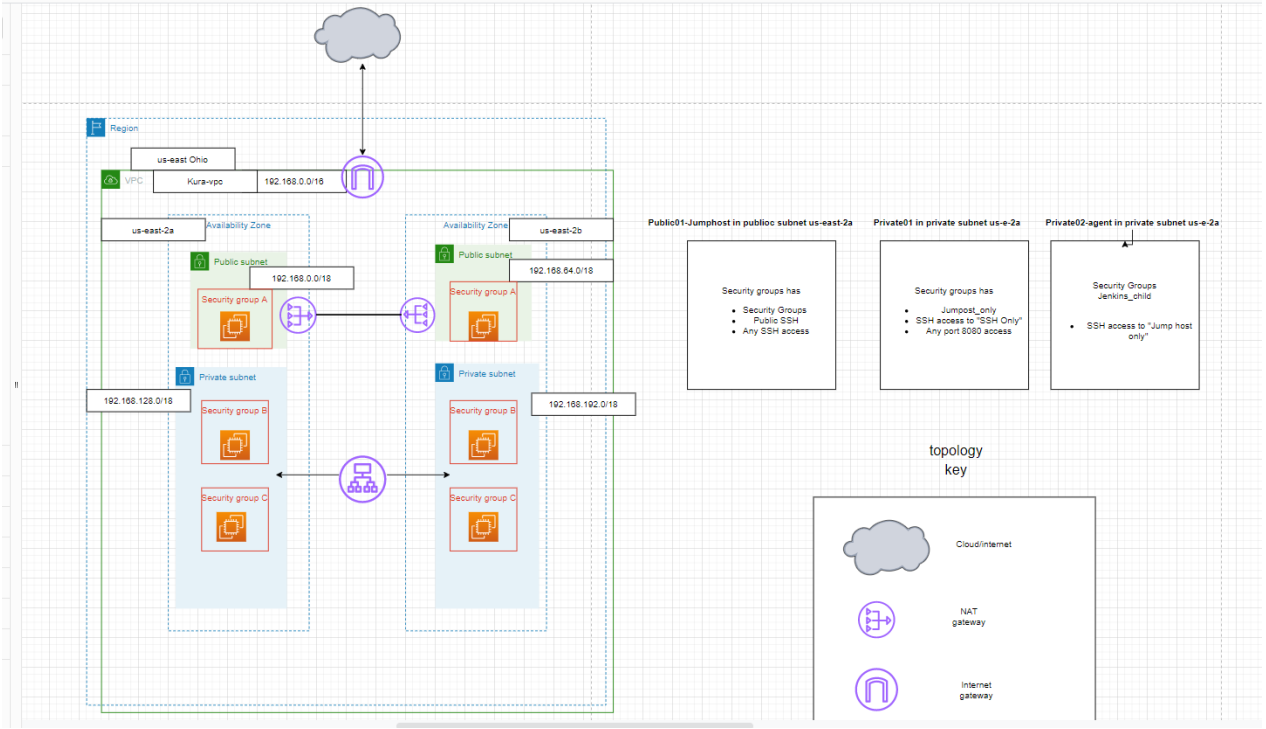
To fix this, go into your ec2 instance for agent and use these commands

sudo yum install maven

sudo yum install git


Not secure | alb-1078231393.us-east-2.elb.amazonaws.com/computer/
Jenkins
search
BISHAJIT LODH
log out
Dashboard
Nodes
Back to Dashboard
Manage Jenkins
New Node
Configure Clouds
S
Name ↓
Architecture
Clock Difference
Free Disk Space
Free Swap Space
Free Temp Space
Response Time
master
Linux (amd64)
In sync
5.91 GB
0 B
5.91 GB
0ms
Private-agent
Linux (amd64)
In sync
6.11 GB
0 B
6.11 GB
47ms
Data obtained
1 min 49 sec


The Internet
us-east N.Virginia
VPC
kura-vpc
192.168.0.0/16
Availability Zone
us-east-1a
Public subnet
192.168.0.0/18
Security group A
Security group A: SSH_Public
Allow Any SSH inbound
Allow port 80 HTTP request
Allow port 8080 Any
Security group B: ONLY_JumpHost
Allow SG: SSHONLY inbound
Allow SSH from SSH_Public
Allow outbound traffic from anywhere
Allows EC2's using private subnet to connect to the internet through a NAT Gateway
SSH into (Private01)EC2 with private subnet
Private subnet
Jenkins is being run here
HOSTS JENKINS Master
Security group B
192.168.128.0/18
SSH into (Agent)EC2 with private same private subnet
Private Subnet
Security group B
HOSTS JENKINS Agent
Availability Zone
us-east-1b
192.168.64.0/18
Public subnet
Security group A
Private subnet
Security group B
192.168.192.0/18
Kura_RT
192.168.0.0/16 VPC
0.0.0.0/0 internet gateway
Private_RT
192.168.0.0/16 VPC
0.0.0.0/0 NAT Gateway
Subnet_Association
192.168.128.0/18 Private01
192.168.192.0/18 Private02
LOAD BALANCER



VPC-Amazon Virtual Private Cloud (VPC) gives you complete control over your virtual networking environment including resource placement, connectivity, and security.

NAT Gateway is **a highly available AWS managed service** that makes it easy to connect to the Internet from instances within a private subnet in an Amazon Virtual Private Cloud (Amazon VPC). Previously, you needed to launch a NAT instance to enable NAT for instances in a private subnet.