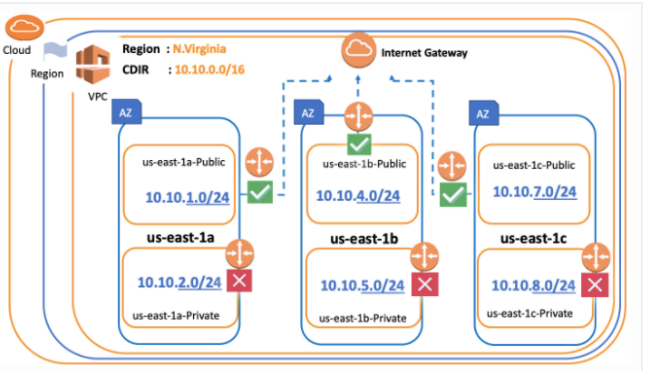
Creating Bastion Host/Jump Box

Review the Scenario

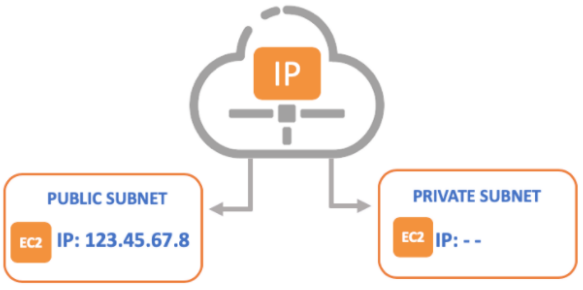


VPC Scenario

Let's Review the Scenario and look at what we have done so far.

- First, we created VPC in N. Virginia Region.
- We used 3 Availability Zones(AZ).
- For each, AZ we created 2 Subnets.
- Of these two subnets; one is Public and the other one is Private.
- We set the rules of Route Table for internet connectivity.
- Then we assigned Private the subnet to another Route Table that we created and blocked the internet connectivity of this Route Table.
- So as you see in the picture above, we have 6 subnets totally. 3 of them are Public Subnets that can be accessed from the Internet and the others are Private Subnets, that are not accessible from the internet.

Public IP of Instances



Public IP of Instances

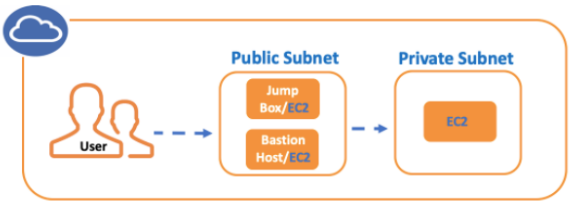
As you remember last lessons, thanks to **Modify Auto-Assign IP** setting, every instance will be created in this subnet will have an IP address.

So, when a user in the outside world wants to go to this IP address of EC2 instance in Public Subnet, it is possible to access this virtual machine from the Internet Gateway.

But when we think about the EC2 virtual machine that we will create in the Private Subnet, it doesn't have internet connectivity. Because in its Route Table, there is no definition about connecting to Internet Gateway and also there is no Public IP allocated.

Therefore, a user other than VPC will not be able to access an EC2 machine in a private subnet. Because it does not have **Public IP** and does not know how to go to the Internet.

Bastion Host/Jump Box



Bastion Host/Jump Box

Now, we have a question:

How do we access this virtual machine in the Private Subnet if there is no public IP?

In such cases, we will install a virtual machine called Jump Box/Bastion Host to one of the public networks.

The purpose of Jump Box is to act as a bridge for the virtual machine in the private subnet and convey the connection to this machine. That's why it's called the Jump Box.

- In this way, we prevent our precious EC2 and database instances the private subnet to be accessed from the outside world in **routine**
- But, on the other hand, we can keep them accessible for updates, monitoring, etc. activities **when we need**.

Now, we will see this as applied in the next pages.

Creating Public Instance as Bastion Host

Let's create Bastion Host instance in the public subnet as we did in EC2 section before.

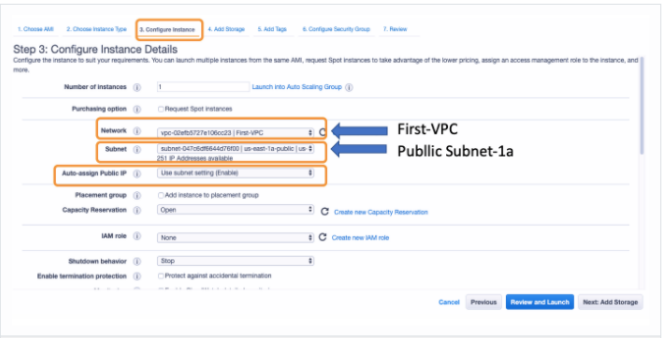
First, go to EC2 Service and select **Instance** section from the left-hand menu. Then click **Launch Instance** tab to create a new instance.

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

Select Amazon Linux 2 AMI (HVM) SSD Volume Type

Step 2: Choose an Instance Type:

We'll continue with **t2.micro** instance type



Step 3: Configure Instance Details

Step 3: Configure Instance Details:

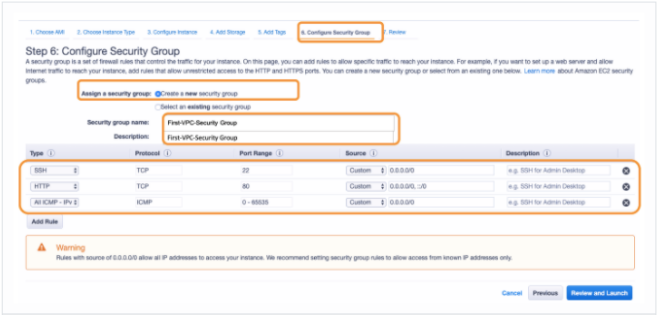
- Let's choose **First-VPC** that we created before as VPC and **public 1-a** as a subnet.
- From **Auto-Assign Public IP** option, we can assign public IP to the virtual machine. As you may remember, we activated this via subnet. That's why we leave it as **Use Subnet Settings** mode.
- Leave the rest as default.

Step 4: Add Storage:

Let's leave the storage part as default.

Step 5: Add Tags:

Let's name the instance as PublicServer or whatever you want.



Step 6: Configure Security Group

Step 6: Configure Security Group:

- We will create a new security group in this section. Because our previous security groups are associated with other VPC. That's why we can't choose it here.
- Let's name it as First-VPC-Security Group
- Let's write the same on the top as Description.
- Add the rule of **SSH** and **HTTP** for access from the outside world and **All ICMP IPv4** for the ping.

Step 7: Review Instance Launch:

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

Public Instance is ready. Let's create a private one.

Creating Private Instance

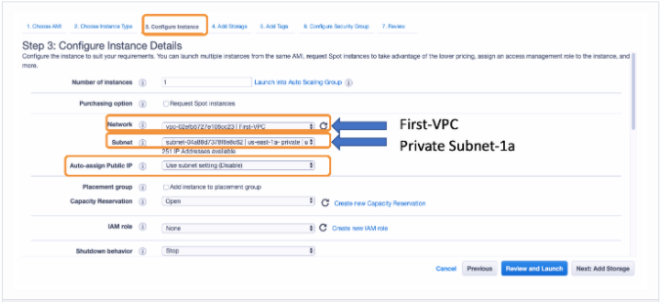
First, go to EC2 Service and select **Instance** section from the left-hand menu. Then click **Launch Instance** tab to create a new instance.

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

Select Amazon Linux 2 AMI (HVM) SSD Volume Type

Step 2: Choose an Instance Type:

We'll continue with **t2.micro** instance type



Step 3: Configure Instance Details

Step 3: Configure Instance Details:

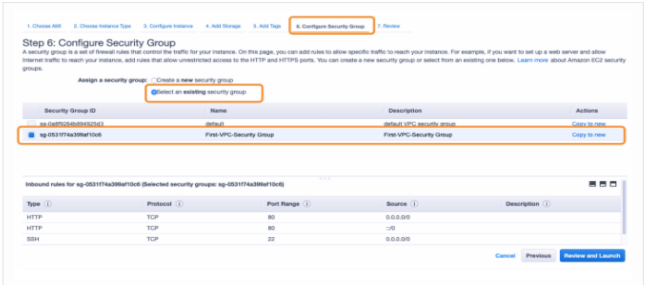
- Let's choose **First-VPC** that we created before as VPC and **private 1-a** as a subnet.
- We leave the **Auto Assign Public IP** option again as **Use Subnet Settings**. However, this time we can see that it is in a **Disable** state. Because we did not activate Public IP assignment for the private subnets as you remember.
- Leave the rest as default.

Step 4: Add Storage:

Let's leave the storage part as default.

Step 5: Add Tags:

Let's name the instance as a Private Server or whatever you want.



Step 6: Configure Security Group

Step 6: Configure Security Group:

- This time we don't need to create a new security group. Because we have already created a security group associated with First- VPC before. So 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.

Private Instance is ready.

Ping The Private Instance via Bastion Host

It's time to try whether Private Instance has connectivity or not. So we will first connect from CLI to the Bastion Host then ping Private Instance indirectly.

- First, connect to Public Instance (Bastion Host).
- You'll see the following script on your Bash monitor.
- If you were not able to connect to the instance, check the command from the former lessons of EC2

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

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-10-10-1-78 ~]$
```

- After connecting to the instance:
  - First, get the root privileges,
  - Then write the **IP of Private Instance** after **ping** command,

```
[ec2-user@ip-10-10-1-78 ~]$ sudo su
[root@ip-10-10-1-78 ec2-user]# ping 10.10.2.194
```

Finally, you'll see the following script as shown below on your Bash monitor.

```
PING 10.10.2.194 (10.10.2.194) 56(84) bytes of data.
64 bytes from 10.10.2.194: icmp_seq=1 ttl=255 time=0.485 ms
64 bytes from 10.10.2.194: icmp_seq=2 ttl=255 time=0.503 ms
64 bytes from 10.10.2.194: icmp_seq=3 ttl=255 time=0.511 ms
64 bytes from 10.10.2.194: icmp_seq=4 ttl=255 time=0.576 ms
```

- Press **CTRL + C** to stop after a while.
- You'll see this script below. It means ping is successful and you have a connection with Private Instance in Private Subnet.

```
--- 10.10.2.194 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 20471ms
rtt min/avg/max/mdev = 0.485/0.549/0.619/0.051 ms
[root@ip-10-10-1-78 ec2-user]#
```

We succesfuly ping the Private Instance via Bastion Host

SSH Connection to The Private Instance via Bastion Host

Let's try to make SSH to the Private Instance from CLI via the Bastion Host.

- First open BASH and go to the directory of the key file.
- Then connect to Private Instance (Bastion Host) with the command following respectively.

⚠Avoid ! :

- The following commands for connecting to EC2 are different from the commands we learned in the former lesson. So while connecting, be sure to use the same commands seen below
- "key.pem" is the name of the key file. While creating our instance we determine this name. So it can be different for everybody. Also, IP will be different. Be sure to write here your key file name and IP number of instance correctly

```
user$ cd desktop/key

eval $(ssh-agent -s)  or  eval "$(ssh-agent)"
ssh-add -K key.pem
ssh -A ec2-user@54.234.128.151
```

You'll see the following script as shown below on your Bash monitor. It means you are in the instance of Bastion Host.

```
Last login: Tue Jan 28 14:32:25 2020 from 95.173.224.230

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

https://aws.amazon.com/amazon-linux-2/
-bash: warning: setlocale: LC_CTYPE: cannot change locale (UTF-8): No such file

[ec2-user@ip-10-10-1-111 ~]$
```

Let's jump here to the private instance with the command seen below.

- First copy the IP of Private Instance,
- Paste it after the following command (ssh ec2-user@IP)

```
[ec2-user@ip-10-10-1-111 ~]$ ssh ec2-user@10.10.2.18
```

- Then you'll see the script seen below.

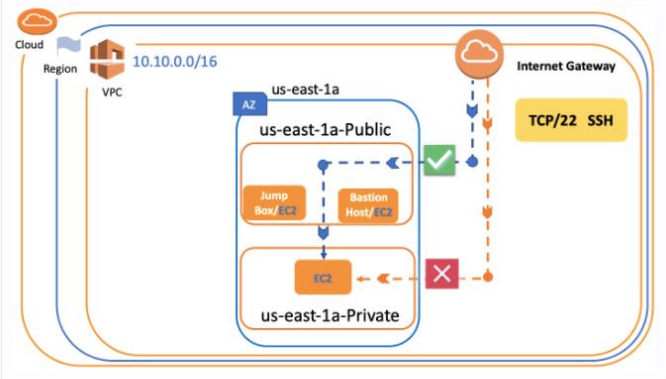
```
The authenticity of host '10.10.2.18 (10.10.2.18)' can't be established.
ECDSA key fingerprint is SHA256:dLsMlOxpZ2ccEmrZLGjwIO/7QNDjTbHPF3jhcqC4Few.
ECDSA key fingerprint is MD5:d3:2e:38:9e:8e:02:d5:4e:4f:89:b5:9c:82:81:d1:6d.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.2.18' (ECDSA) to the list of known hosts.

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

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-10-10-2-18 ~]$
```

Congratulations! You successfully connect to Private Instance via SSH with the help of the Bastion Host.

Conclusion: Bastion Host/Jump Box



Bastion Host/Jump Box Diagram

Complementary Lesson about How to create a Bastion Host/Jump Box