

## Building a First Non-Default VPC

---

edureka!

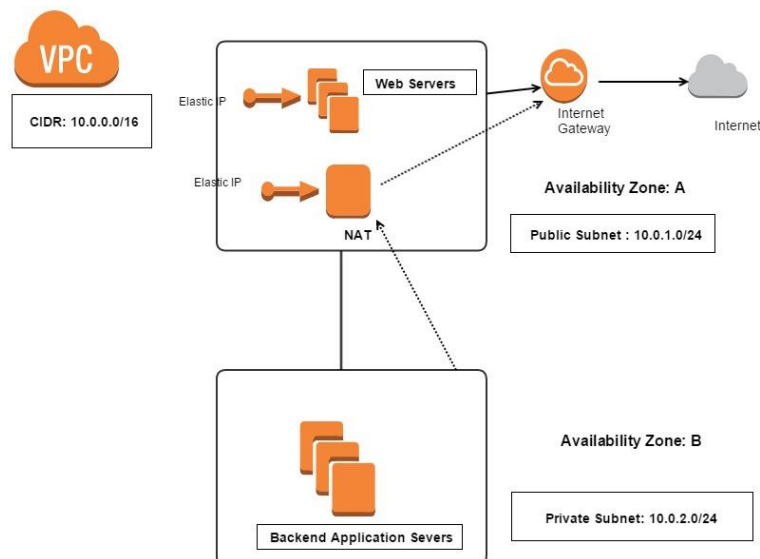
**edureka!**

© Brain4ce Education Solutions Pvt. Ltd.

## Overview

In this lab session, you create a basic virtual private cloud (VPC) and then extend it to produce a customized result. You do all of this with the AWS Management Console.

This diagram illustrates what we will build:



The overall VPC is designed to incorporate these basic features:

- It spans two Availability Zones (AZs), in order that later you can distribute applications across these zones in order to architect for application durability and availability.
- Within each Availability Zone (AZ) there are two subnets: one “public” subnet is connected directly to the Internet. The other “private” subnet is able to communicate with any other subnet within the VPC; however, there is no access to them from the Internet. The dashed line demarcates this isolation.
- You will walk through two alternatives to allowing access to servers that are in the private subnets.

### Step 1: Create a VPC with CIDR 10.0.0.0/16 with tenancy as Default

1. Click **VPC** within Networking section. This will take you to VPC Dashboard.

## Networking

**VPC**  
Isolated Cloud Resources**Direct Connect**  
Dedicated Network Connection to AWS**Route 53**  
Scalable DNS and Domain Name Registration2. Select **Your VPCs** in the navigation pane.

**VPC Dashboard**

Filter by VPC: None

Virtual Private Cloud

**Your VPCs**

Subnets

Route Tables

Internet Gateways

DHCP Options Sets

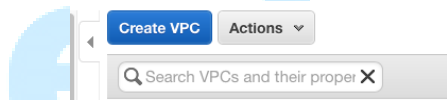
**Resources**

[Start VPC Wizard](#) [Launch EC2 Instances](#)

Note: Your Instances will launch in the US West (N. California) region.

You are using the following Amazon VPC resources in the US West (N. California) region:

2 VPCs	1 Internet Gateway
4 Subnets	2 Route Tables
2 Network ACLs	0 Elastic IPs
0 VPC Peering Connections	0 Endpoints
0 Nat Gateways	63 Security Groups
0 Running Instances	0 VPN Connections

3. Click **Create VPC**.4. This will open another window for you to define and create a VPC. Over here, we will name it as "My VPC", choose **CIDR block** as 10.0.0.0/16 and select **Tenancy** as Default. Click **Yes, Create**.

**Create VPC**

A VPC is an isolated portion of the AWS cloud populated by AWS objects, such as Amazon EC2 instances. Use the Classless Inter-Domain Routing (CIDR) block format to specify your VPC's contiguous IP address range, for example, 10.0.0.0/16. You cannot create a VPC larger than /16.

Name tag:

CIDR block:

Tenancy: Default

[Cancel](#) [Yes, Create](#)

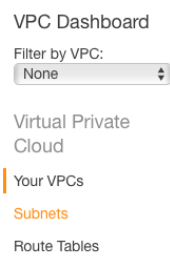
→ This will eventually add your non-default VPC, which would now show up in the dashboard.

<input type="checkbox"/>	Name	VPC ID	State	VPC CIDR	DHCP options set	Route table	Network ACL	Tenancy
<input checked="" type="checkbox"/>	My VPC	vpc-45b39a20	available	10.0.0.0/16	dopt-13cb0976	rtb-34a3ad51	acl-9beeeefe	Default
<input type="checkbox"/>		vpc-1771e672	available	172.31.0.0/16	dopt-13cb0976	rtb-b19efed4	acl-8098fce5	Default

## Step 2: Add Subnets to this VPC

Now we must add two different subnets: one Public and another Private to this VPC we just created.

- Click **Subnets** in the navigation pane.



- Click **Create Subnet**. Over here, we would be naming it as Public Subnet, associate this to one of the listed AZs, and assign it a CIDR range of 10.0.1.0/24.

**Create Subnet**

Use the CIDR format to specify your subnet's IP address block (e.g., 10.0.0.0/24). Note that block sizes must be between a /16 netmask and /28 netmask. Also, note that a subnet can be the same size as your VPC.

Name tag: Public Subnet

VPC: vpc-45b39a20 (10.0.0.0/16) | My VPC

Availability Zone: us-west-1b

CIDR block: 10.0.1.0/24

Cancel Yes, Create

- Click **Yes, create** to create this subnet.
- Similarly, we will create another subnet (Private) with a CIDR range of 10.0.2.0/24 while associating it to a different AZ this time.

**Create Subnet**

Use the CIDR format to specify your subnet's IP address block (e.g., 10.0.0.0/24). Note that block sizes must be between a /16 netmask and /28 netmask. Also, note that a subnet can be the same size as your VPC.

Name tag: Private Subnet

VPC: vpc-45b39a20 (10.0.0.0/16) | My VPC

Availability Zone: us-west-1c

CIDR block: 10.0.2.0/24

Cancel Yes, Create

### Step 3: Create an Internet Gateway and attach it VPC

An Internet Gateway creates a gateway to the internet. We should create and attach it to our virtual private cloud.

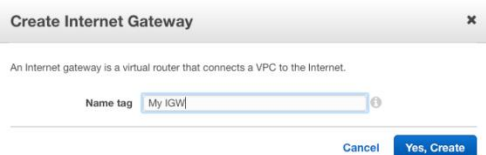
9. Select **Internet Gateways** in the navigation pane.

Virtual Private  
Cloud  
Your VPCs  
Subnets  
Route Tables  
**Internet Gateways**  
DHCP Options Sets

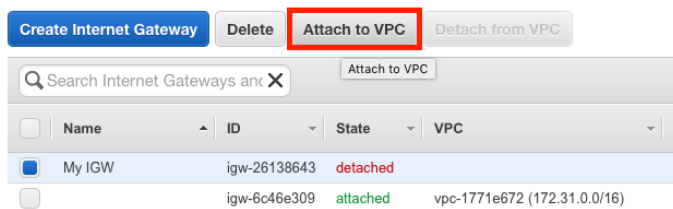
10. Click **Create Internet Gateway**.



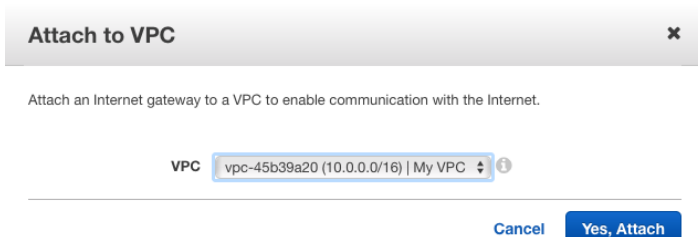
11. Assign a name to it and click **Yes, Create**.



12. Once created, we have to attach it our VPC by clicking on **Attach to VPC**.



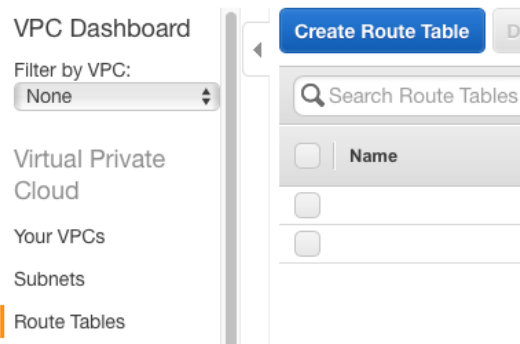
13. Select appropriate VPC in the list and click **Yes, Attach**.



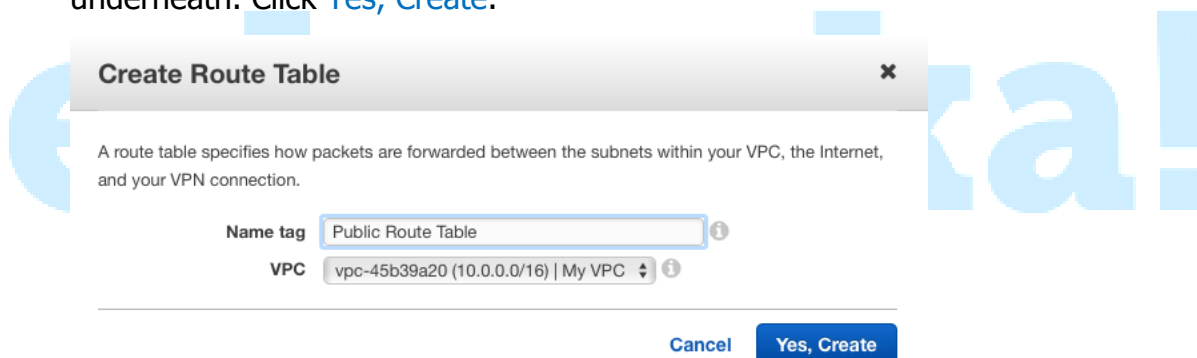
#### Step 4: Create a Route Table containing Internet Gateway within its route

It's time for us to create a route table especially for the subnet that we want to make it as Public.

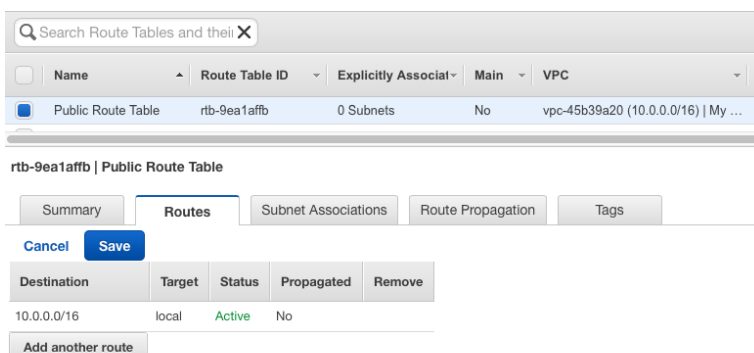
14. Select **Route Tables** in the navigation pane and click **Create Route Table**.



15. Over here, we name it as "Public Route Table" and select an appropriate VPC underneath. Click **Yes, Create**.



16. Select this route table in the list, navigate to **Routes** and click **Add Another Route**.



17. Add another route with destination as 0.0.0.0/0 and choose Target as your Internet Gateway. This will ensure that any traffic that flows to and fro between your VPC and internet goes through this Internet Gateway that you have attached to your VPC. Click **Save** to apply this change you made to this route.

rtb-9ea1affb | Public Route Table

Summary Routes Subnet Associations Route Propagation Tags

Cancel Save

Destination	Target	Status	Propagated	Remove
10.0.0.0/16	local	Active	No	
0.0.0.0/0	igw-26138643	No	No	✕

Add another route

### Step 5: Associate this Route table to the subnet to make it "Public"

Once the route table is created, we have to assign the same to one of our subnets we intend to make it as a Public one.

18. Select your Public Subnet, select its **Route Table** and click **Edit**.

VPC Dashboard

Filter by VPC: None

Virtual Private Cloud

Your VPCs

Subnets

Route Tables

Internet Gateways

DHCP Options Sets

Elastic IPs

Endpoints

NAT Gateways

Peering Connections

Security

Network ACLs

Create Subnet Subnet Actions

Search Subnets and their pro X

Name	Subnet ID	State	VPC	CIDR
	subnet-c0c07a99	available	vpc-1771e672 (172.31.0.0/16)	172.3
	subnet-6c387d09	available	vpc-1771e672 (172.31.0.0/16)	172.3
Private Subnet	subnet-762fc712	available	vpc-45b39a20 (10.0.0.0/16)   My ...	10.0.2
Public Subnet	subnet-a20533fb	available	vpc-45b39a20 (10.0.0.0/16)   My ...	10.0.1

subnet-a20533fb (10.0.1.0/24) | Public Subnet

Summary Route Table Network ACL Flow Logs Tags

Edit

Route Table: rtb-34a3ad51

Destination	Target
10.0.0.0/16	local

19. Select route table in the list and click **Save** to apply this change. Now, we have associated our newly created route table to this subnet, and therefore made it as a "Public Subnet" of our VPC.

subnet-a20533fb (10.0.1.0/24) | Public Subnet

Summary Route Table Network ACL Flow Logs Tags

Cancel Save

Current Route Table: rtb-34a3ad51

Change to: rtb-9ea1affb | Public Route Table

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	igw-26138643

### Step 6: Launch instances in each of the subnets

- We can now launch instances in both these subnets. In this example, we have launched two Ubuntu instances, one in Public Subnet and another one in Private Subnet of our VPC. Over here, we have to make sure that we allow HTTP and HTTPS traffic from all sources while defining rules in security groups being assigned to both these instances.

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below.

Learn more about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group ☐ Select an existing security group

Security group name:

Description:

Type	Protocol	Port Range	Source
SSH	TCP	22	Anywhere 0.0.0.0/0
HTTP	TCP	80	Anywhere 0.0.0.0/0
HTTPS	TCP	443	Anywhere 0.0.0.0/0

- Once launched, we can see these instances in EC2 dashboard.

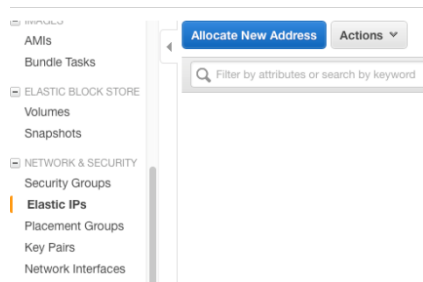
Filter by tags and attributes or search by keyword							
<input type="checkbox"/>	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status
<input type="checkbox"/>	Public Instance	i-f4eeba41	t2.micro	us-west-1b	running	2/2 checks ...	None
<input checked="" type="checkbox"/>	Private Instance	i-e365fda6	t2.micro	us-west-1c	running	2/2 checks ...	None

### Step 7: Assign Elastic IPs to instances in Public Subnet

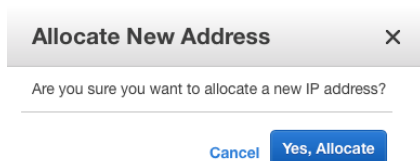
As we have launched these instances in Non-Default VPC, they will never get Public IPs assigned to them. Therefore, we have to assign Elastic IPs to all instances running in Public Subnet of our VPC.



20. Go to EC2 dashboard and click **Elastic IPs** in the navigation pane. Click **Allocate New Address**.



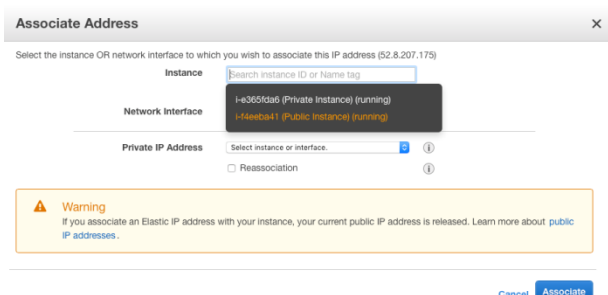
21. Click **Yes, Allocate** to allocate an elastic IP address.



22. Select this newly allocated IP address, click **Actions** and choose **Associate Address**.



23. Choose the instance launched in Public Subnet of your VPC and click **Associate**.



→ This associates this elastic IP address to public subnet's instance.

Filter by attributes or search by keyword

<<

>

1 to 1 of 1

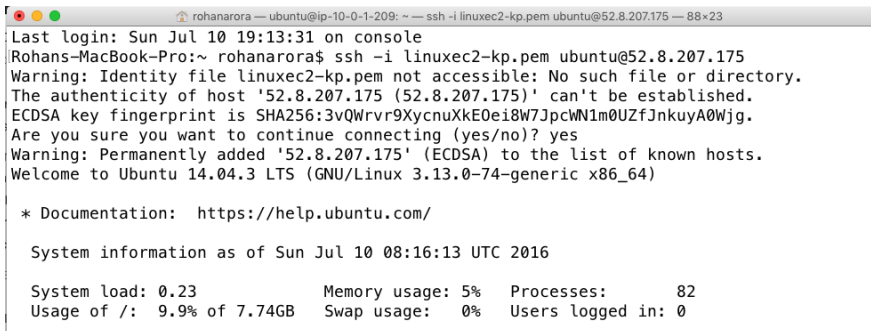
>>

	Elastic IP	Allocation ID	Instance	Private IP Address	Scope	Public DNS
	52.8.207.175	eipalloc-b003938a	i-4feeba41 (Public Instance)	10.0.1.209	vpc-45b39a20	

## Step 8: Connect to instances via SSH/RDP

It's time for us to initiate SSH sessions towards both these instances running in two different subnets of our non-default VPC.

### 24. Initiate an SSH session and login into instance in Public Subnet.



```

Last login: Sun Jul 10 19:13:31 on console
Rohans-MacBook-Pro:~ rohanarora$ ssh -i linuxec2-kp.pem ubuntu@52.8.207.175
Warning: Identity file linuxec2-kp.pem not accessible: No such file or directory.
The authenticity of host '52.8.207.175 (52.8.207.175)' can't be established.
ECDSA key fingerprint is SHA256:3vQWrvr9XycnuXkE0ei8W7JpcWN1m0UZfJnkuyA0Wjg.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '52.8.207.175' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-74-generic x86_64)

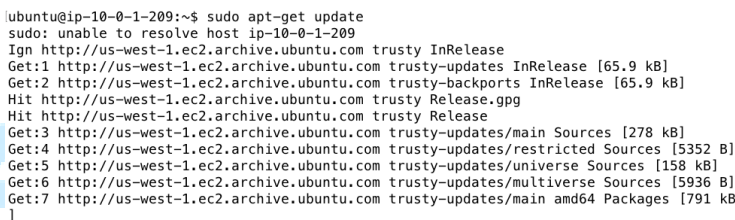
 * Documentation:  https://help.ubuntu.com/

System information as of Sun Jul 10 08:16:13 UTC 2016

System load: 0.23           Memory usage: 5%    Processes:      82
Usage of /:  9.9% of 7.74GB Swap usage:   0%    Users logged in: 0

```

### 25. Try installing updates by running command `sudo apt-get update`.



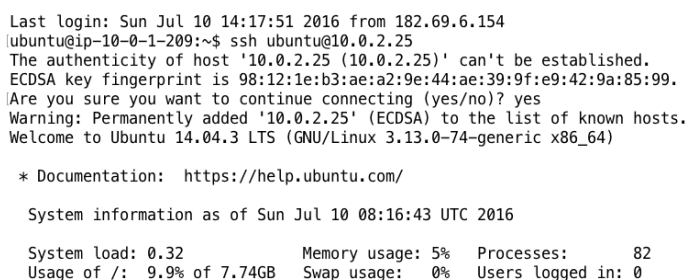
```

ubuntu@ip-10-0-1-209:~$ sudo apt-get update
sudo: unable to resolve host ip-10-0-1-209
Ign http://us-west-1.ec2.archive.ubuntu.com trusty InRelease
Get:1 http://us-west-1.ec2.archive.ubuntu.com trusty-updates InRelease [65.9 kB]
Get:2 http://us-west-1.ec2.archive.ubuntu.com trusty-backports InRelease [65.9 kB]
Hit http://us-west-1.ec2.archive.ubuntu.com trusty Release.gpg
Hit http://us-west-1.ec2.archive.ubuntu.com trusty Release
Get:3 http://us-west-1.ec2.archive.ubuntu.com trusty-updates/main Sources [278 kB]
Get:4 http://us-west-1.ec2.archive.ubuntu.com trusty-updates/restricted Sources [5352 B]
Get:5 http://us-west-1.ec2.archive.ubuntu.com trusty-updates/universe Sources [158 kB]
Get:6 http://us-west-1.ec2.archive.ubuntu.com trusty-updates/multiverse Sources [5936 B]
Get:7 http://us-west-1.ec2.archive.ubuntu.com trusty-updates/main amd64 Packages [791 kB]
]

```

→ This installs all updates to this Ubuntu instance through Internet Gateway at the backend, which is associated to Public Subnet containing this very server.

### 26. Let us now connect to instance launched in Private Subnet through this instance in Public one.



```

Last login: Sun Jul 10 14:17:51 2016 from 182.69.6.154
ubuntu@ip-10-0-1-209:~$ ssh ubuntu@10.0.2.25
The authenticity of host '10.0.2.25 (10.0.2.25)' can't be established.
ECDSA key fingerprint is 98:12:1e:b3:ae:a2:9e:44:ae:39:9f:e9:42:9a:85:99.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.2.25' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-74-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

System information as of Sun Jul 10 08:16:43 UTC 2016

System load: 0.32           Memory usage: 5%    Processes:      82
Usage of /:  9.9% of 7.74GB Swap usage:   0%    Users logged in: 0

```

### 27. We will try to install updates. As expected, it will not be a successful installation because we haven't established any source for the Private Subnet to reach the internet.

```
Documents — ubuntu@ip-10-0-2-25: ~ — ssh -A ubuntu@52.8.207.175 — 80x24
ubuntu@ip-10-0-2-25:~$ sudo apt-get update
sudo: unable to resolve host ip-10-0-2-25
0% [Connecting to us-west-1.ec2.archive.ubuntu.com (54.177.73.145)] [Connecting
```

## Step 9: Create a NAT Security Group

Prior launching a NAT instance, we need to create a security group that would be assigned to this instance during its launching process.

28. For Inbound rules, we have to allow web traffic (HTTP and HTTPS) from CIDR range belonging to Private Subnet.

The screenshot shows the 'Create Security Group' dialog in the AWS Management Console. The 'Security group name' is 'NAT SG' and the 'Description' is 'This security group is for our NAT instance'. The 'VPC' is 'vpc-45b39a20 (10.0.0.0/16) | My VPC'. Under 'Security group rules', the 'Inbound' tab is selected. It shows two rules: HTTP (TCP, port 80) and HTTPS (TCP, port 443), both with a source of 'Anywhere' and '10.0.2.0/24'. There are 'Add Rule', 'Cancel', and 'Create' buttons at the bottom.

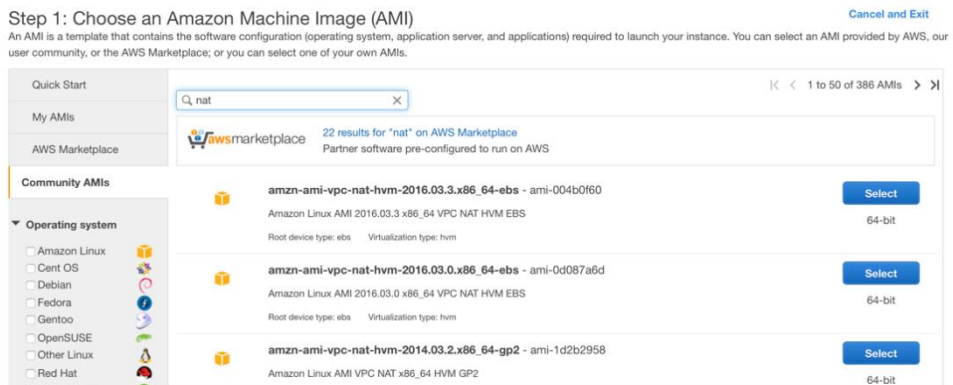
29. For outbound rules, the same traffic would be forwarded to all destinations i.e. 0.0.0.0/0. Click [Create](#) to form this security group.

This screenshot shows the 'Outbound' tab of the 'Create Security Group' dialog. It displays two rules: HTTP (TCP, port 80) and HTTPS (TCP, port 443), both with a destination of 'Anywhere' and '0.0.0.0/0'. There is an 'Add Rule' button at the bottom left.

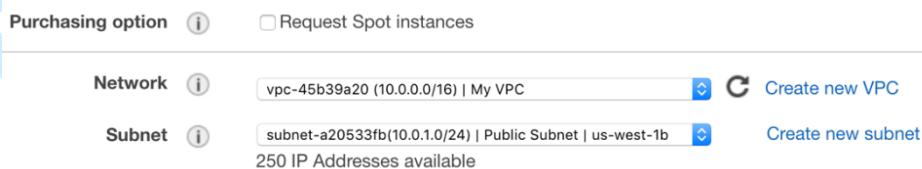
## Step 10: Launch a NAT Instance

The NAT instance will be launched in Public Subnet, and it will allow all instances of Private Subnet to download patches from the internet securely.

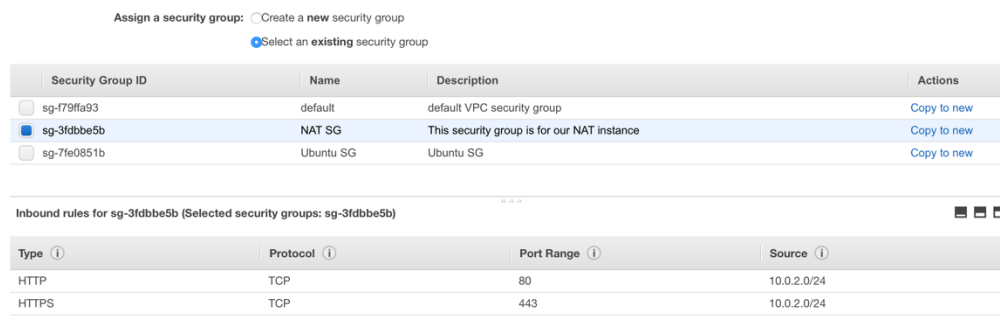
30. To get the right AMI for your NAT instance, go to **Community AMIs** and type **NAT** in the search bar. This will present you with different options to choose from. Select any one of these to launch NAT instance.



31. We need to ensure that we launch our NAT instance in Public Subnet.



32. Apply NAT Security Group by choosing **Select an existing security group**.



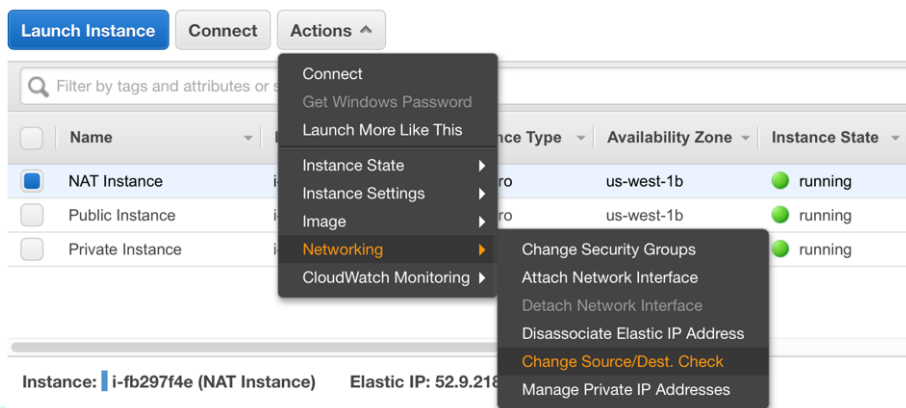
33. Assign an Elastic IP address to this instance.

<input checked="" type="checkbox"/>	Elastic IP	Allocation ID	Instance	Private IP Address	Scope
<input checked="" type="checkbox"/>	52.9.218.205	eipalloc-9cd142a6	i-fb297f4e (NAT Instance)	10.0.1.131	vpc-45b39a20

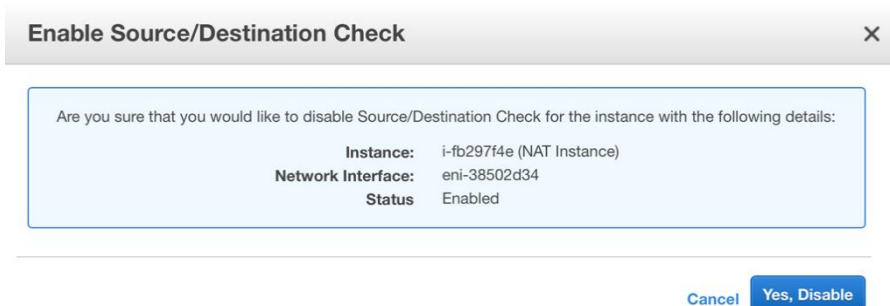
### Step 11: Disable Source/Destination Check for NAT instance

Each EC2 instance performs source/destination checks by default. This means that the instance must be the source or destination of any traffic it sends or receives. However, a NAT instance must be able to send and receive traffic when the source or destination is not itself. Therefore, you must disable source/destination checks on the NAT instance.

34. Select the NAT instance, choose **Actions**, select **Networking**, and then select **Change Source/Dest. Check**.



35. For the NAT instance, verify that this attribute is disabled. Otherwise, choose **Yes, Disable**.



### Step 12: Associate NAT instance to private subnet

As all database and backend instances in private subnet will get access to internet through NAT instance, we need to associate the same to the subnet through a route table.

36. Go back to VPC Dashboard and click **Route Tables** in the navigation pane.

37. Create a new route table that would be assigned to private subnet of VPC.

**Create Route Table**
✕

A route table specifies how packets are forwarded between the subnets within your VPC, the Internet, and your VPN connection.

Name tag

VPC vpc-45b39a20 (10.0.0.0/16) | My VPC

Cancel
Yes, Create

38. Add another route to this route table by allowing access to all destinations i.e. 0.0.0.0/0 through the NAT instance. Select [Save](#).

☒ Private Route Table
rtb-14313871
0 Subnets
No
vpc-45b39a20 (10.0.0.0/16) | My ...

**rtb-14313871 | Private Route Table**

Summary
Routes
Subnet Associations
Route Propagation
Tags

Cancel
Save

Destination	Target	Status	Propagated	Remove
10.0.0.0/16	local	Active	No	
<input type="text" value="0.0.0.0/0"/>	<input type="text" value="igw-26138643   My IGW"/>		No	✕

Add another route

igw-26138643 | My IGW  
 i-fb297f4e | NAT Instance

#### Step 14: Associate route table to the private subnet

We need to associate this route table to the private subnet now.

39. Select Private Subnet from the [Subnet](#) list and change its route table from default to the one we created in our previous step. Click [Save](#).

subnet-762fc712 (10.0.2.0/24) | Private Subnet

Summary
Route Table
Network ACL
Flow Logs
Tags

Cancel
Save

Current Route Table: rtb-34a3ad51

Change to: rtb-14313871 | Private Route Table

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	eni-38502d34 / i-fb297f4e

### Step 13: Test the setup

Finally, it's time for us to test this entire VPC setup.

40. Login to the instance in public subnet via SSH/RDP. From here initiate another SSH/RDP session towards another instance residing in private subnet.

41. We will now try installing some updates to private subnet's instance.

```
Last login: Mon Jul 11 10:45:18 2016 from 10.0.1.209
ubuntu@ip-10-0-2-25:~$ sudo apt-get update
sudo: unable to resolve host ip-10-0-2-25
Ign http://us-west-1.ec2.archive.ubuntu.com trusty InRelease
Get:1 http://us-west-1.ec2.archive.ubuntu.com trusty-updates InRelease [65.9 kB]
Get:2 http://us-west-1.ec2.archive.ubuntu.com trusty-backports InRelease [65.9 kB]
Hit http://us-west-1.ec2.archive.ubuntu.com trusty Release.gpg
Hit http://us-west-1.ec2.archive.ubuntu.com trusty Release
Get:3 http://us-west-1.ec2.archive.ubuntu.com trusty-updates/main Sources [278 kB]
Get:4 http://us-west-1.ec2.archive.ubuntu.com trusty-updates/restricted Sources [5352 B]
Get:5 http://us-west-1.ec2.archive.ubuntu.com trusty-updates/universe Sources [158 kB]
Get:6 http://us-west-1.ec2.archive.ubuntu.com trusty-updates/multiverse Sources [5036 B]
```

→ Consequently, all updates and packages install successfully as all this internet traffic is flowing through and managed by NAT instance at the backend.

edureka!