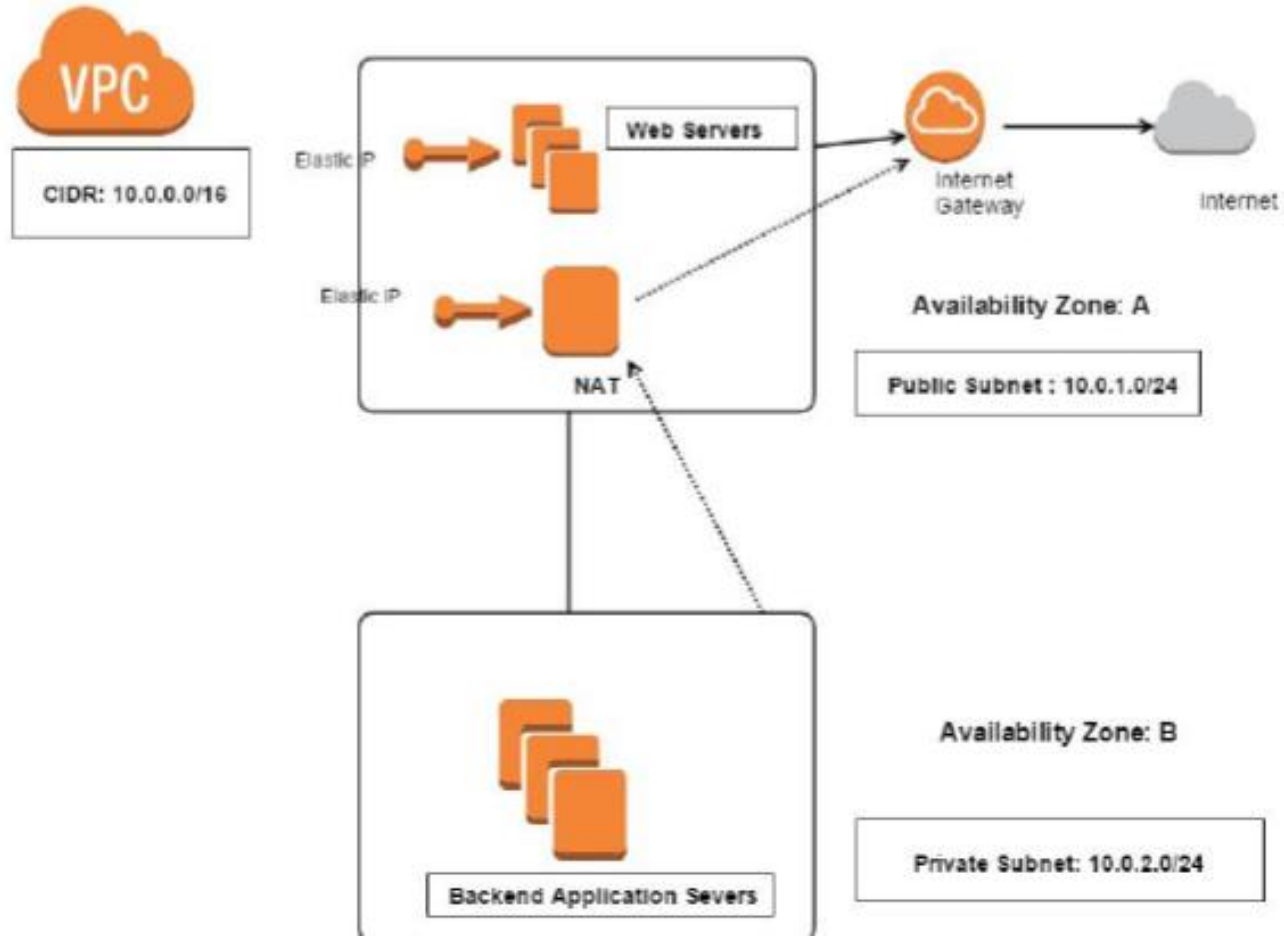
Building a First Non-Default VPC

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

2 Click Create VPC.

3. This will open another window for you to define and create a VPC. Ove 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.

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

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

4.. Select Your VPCs in the navigation pane.

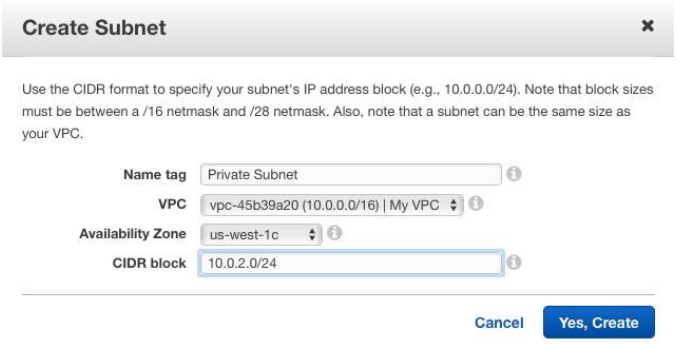
Step 2: Add Subnets to this VPC

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

5. Click Subnets in the navigation pane.

6. 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.

7. Click Yes, create to create this subnet.
8. 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.



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.

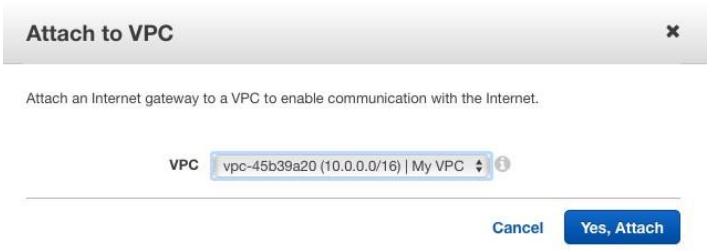
9. Select Internet Gateways in the navigation pane.

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.



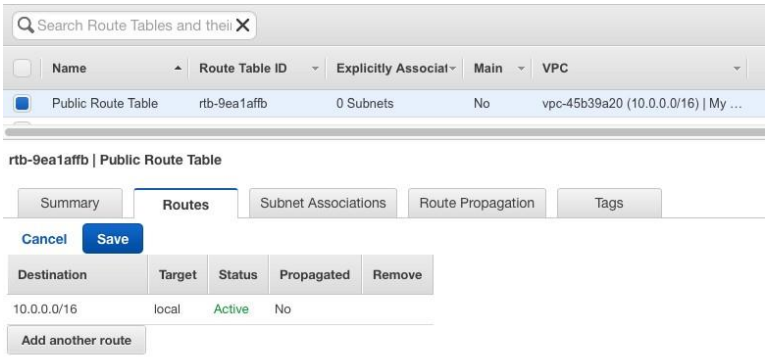
Step 4: Create a Route Table containing Internet Gateway within its route

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

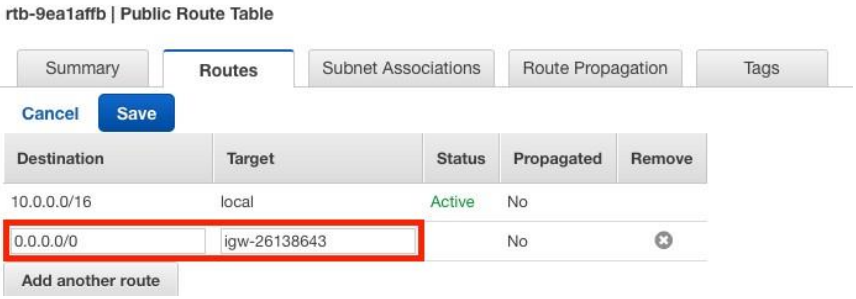
14. Select Route Tables in the navigation pane and click Create Route Table.

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 the through this Internet Gateway that you have attached to your VPC. Click Save to apply this change you made to this 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.

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

Step 6: Launch instances in each of the subnets

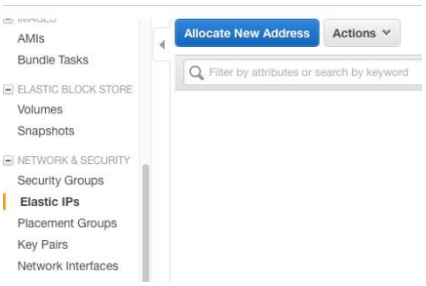
We can now launch instances in both these subnets. In this example, we have launched two Ubuntu instances, one in Public Subnet and another one in Private Subnet of our VPC. Over here, we have to make sure that we allow HTTP and HTTPS traffic from all sources while defining rules in security groups being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

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

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.

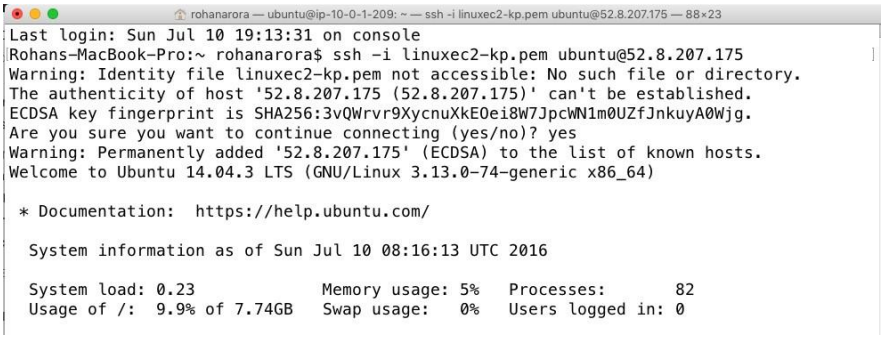
23. Choose the instance launched in Public Subnet of your VPC and click Associate.

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

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.



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

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.

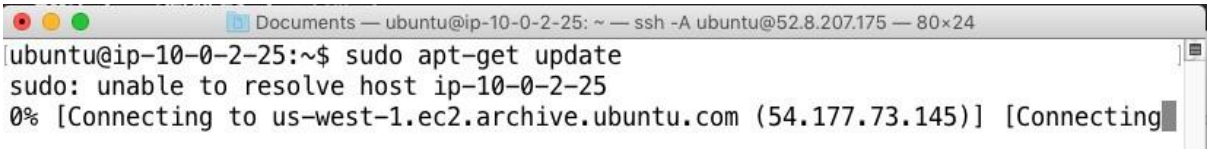
```
Last login: Sun Jul 10 14:17:51 2016 from 182.69.6.154
ubuntu@ip-10-0-1-209:~$ ssh ubuntu@10.0.2.25
The authenticity of host '10.0.2.25 (10.0.2.25)' can't be established.
ECDSA key fingerprint is 98:12:1e:b3:ae:a2:9e:44:ae:39:9f:e9:42:9a:85:99.
Are you sure you want to continue connecting (yes/no)? 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.



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.

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

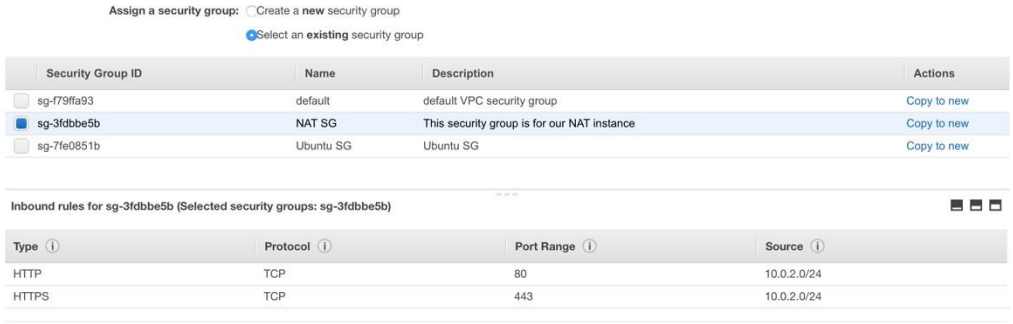
Step 10: Launch a NAT Instance

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.

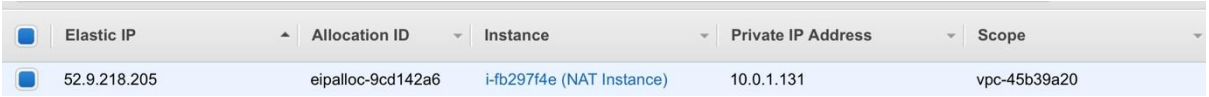
30. To get the right AMI for your NAT instance, go to Community AMIs and type NAT in the search bar. This will present you with different options to choose from. Select any one of these to launch NAT instance.

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.



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.

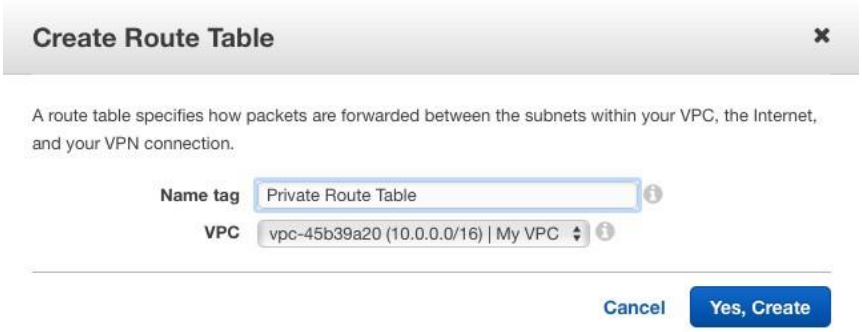
35. For the NAT instance, verify that this attribute is disabled. Otherwise, choose Yes, Disable.

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 pan

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

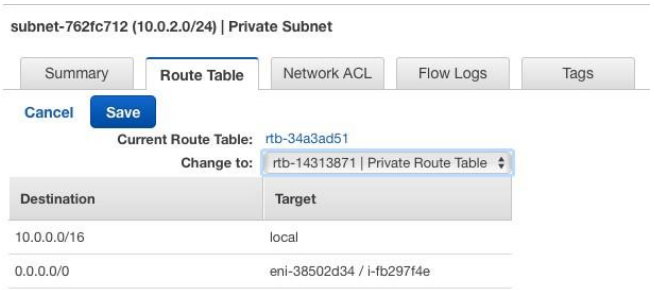


38. Add another route to this route table by allowing access to all destinations i.e. 0.0.0.0/0 through the NAT instance. Select Save.

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



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.

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