VPC : Virtual Private Cloud
++++++++++++++++++++++++++
-> We are creating Virtual Machines in AWS to setup our infrastructure

-> In order to create Virtual Machine mainley we will use 3 services

1) EC2 (Elastic Compute Cloud)
2) EBS (Elastic Block Storage)
3) VPC (Virtual Private Cloud)

Note: Without using these services we can't launch Virtual VM

-> In AWS we have total 200+ Services

-> Those 200 services we can categorize them into 2 types

1) AWS Managed Services (SaaS)
2) Customer Managed Services (IaaS & PaaS)

Note: VPC is one of the customer managed service in AWS

Example

Take a 2 BHK flat
(Land, House (BR-1, BR-2, Kitchen, Hall))

Land Gate (To enter into our land)
Hall Gate (To enter into our house)
Kitchen Gate, BR-1 gate and BR-2 gate

-> Consider VPC also like above house

-> For every room seperate door (In VPC seperate network (Subnets))

-> We need main gate to enter into house (In VPC we will use Internet Gateway)

-> To enter into hall we will use hall gate (In VPC we will use Routing Table)

-> People will come into our house and sit in hall for discussion then they will leave
    (we will be monitoring manually)

Note: To avoid public access we will keep our components in private subnets

(Webservers, Appservers, DB Servers)

-> Public subnet means Internet connection will be there

-> Private subnet means no internet connection

-> If our components are in private subnet then how to take code or update code ?

Note: NAT gateway comes into picture (it will be created in public subnet)

-> Our public and private subnets will be connected to NAT Gateway

Q) What is the difference between Ineternet Gateway and NATGateway ?

-> IG will have both incoming and outgoing

-> Nategateway will have only outgoing

Note: Outside ppl can't access NAT Gateway

VPC

Subnets (private & public)
Internet Gateway (Ingres & Agress)
Route Table (Connections to Subnets)
NAT Gateways (no incoming only outgoing)

##########################
Then how to provide security ?
##########################

-> In houses manually we will be monitoring who is entering into our house and who is entering into rooms

-> In VPC we have to configure security rules

-> In VPC to provide security we have 2 options

1) Security Group
2) NACL (N/W Access Control List)

Note: Who can access which subnet we will configure in NACL

-> Every subnet will have its own NACL (both private and public)
-> NACL will work at subnet level

-> Security group is used to open ports and who can access those ports
-> Security groups works at resource level

-> If we want to apply same rule for multiple resources then we will go for security group and will add that for all resources

-> If we want to apply security policy for entire subnet then we will use NACL

-> In security group we can only allow the traffic
-> In NACL we can do both allow and deny the traffic

Note : For our aprtment we will have security to notedown incoming and outgoing ppl/vehicles recrods

-> If we want same functionality for our VPC then we can use VPC FLOW Logs
-> VPC Flow Logs will track incoming and outgoing traffic of our VPC
-> By default VPC is disabled (we can enable)

Note: Check default VPC configurations Route Tables and NACL

--

-> To construct house we will take land measurements (length & width)

-> How to take measurements in VPC Networking ?

-> We will do VPC sizing based on IPS

Example

If we take 100 ip's then in VPC we will allocate those 100 ip's to 2 subnets

Subnet-1 : 50 ips
Subnet-2 : 50 ips

-> All my 100 ips got completed. In future if i get new requirement i m running out of ips

-> To overcome these problems we don't allocate all ips to subnets
   (we will allocate few ips and will keep few ips in reserve mode)

-> Before entering into ip sizing lets understand IPs first

Note: We can create VPC is very easily but we will get multiple errors while creating subnets and ip allocation.

Note: there are several websites will help us in ip division

-> Lets understand ips

-> I have 2 subnets connected with LAN

```
                              private ip
        subnet-1 <----------------------> subnet-2
```

-> If 2 systems not connected with LAN then we should go for public ip

```
                               public ip
        System-1 <----------------------> System-2
```

-> IPs are divided into 2 types

1) IPV4
2) IPV6

Note: Earlier we have only IPV4

-> In IPV4 we have 5 classes
(A, B, C, D, E)

Note: Class E is reserved for researching purpose

-> IP ranges we will do with CIDR

-> Class less inter domain range

Note: These CIDR classes are required for only public ips and not required for private ips

-> Daily billions of new devices launching and they are using internet

-> If a device wants to use internet then ip is mandatory (we are running out of ips)

-> To overcome this 1PV4 problem IPV6 came into picture

IPV4
++++
10 . 0 . 0 . 1
32-Bits
We have 4 parts here
Every part is called as one octect
In every octect we have 8 bits
4 parts * 8 bits = 32 bits

IPV6
++++
10.4.1 - - -
128 bits
We have 8 parts
In evary part we have 16 bits
8 parts * 16 bits = 128 bits

++++++++++++++++

```
IP ranges in IPV4
+++++++++++++++++
-> IP ranges will be calculated in 2 power

-> If we give range as /16

10.0.0.1/16 = > 2 power (32-16) => 2 Power 16 => 65536
10.0.0.1/32 = > 2 power (32-32) => 2 Power 0  => 1
10.0.0.1/31 = > 2 power (32-31) => 2 Power 1  => 2
10.0.0.1/30 = > 2 power (32-30) => 2 Power 2  => 4
10.0.0.1/29 = > 2 power (32-29) => 2 Power 3  => 8
10.0.0.1/28 = > 2 power (32-28) => 2 Power 4  => 16 (AWS supports from /28)

=> /29  to /32 AWS won't support

-> We can start giving subnet ranges from /16 to /28

-> Recommended to use /24

        10.0.0.1/24 = > 2 power (32-24) => 2 Power 8 => 256

Note: /24 we will get 256 IPs those are sufficient for our usecases in realtime


#################
VPC Lab Task
#################

-> Create VPC

-> CIDR Blok : 10.0.0.0/16

-> Select No IPV6 CIDR blok

-> Select Default Tenancy

-> Create VPC

Note: After creating VPC verify its details

(DNS hostnames -> Disabled)

#############
Create Subnets
################

Note:  By default 3 subnets will be available under default vpc


Create Subnet-1
----------------------
-> Create Subnet

   Name : public-subnet-az-1

-> select availability zone as 1a

-> CIDR Block : 10.0.0.0/24  (It will take 256 ips)

Create Subnet-2
-------------------------
-> Create Subnet

   Name : private-subnet-az-2
```

-> select availability zone as 1b

-> CIDR Block : 10.0.1.0/24 (It will take 256 ips)

Note: Every subnet will have Route Table and NACL

-> AWS will reserve 5 ips in every subnet (we will get only 251)

Create Internet gateway

Note: By default one IGW will be available and it will be attached to default VPC

-> Create custom Internet Gateway (demo-vpc-igw)

-> Attach this IGW to VPC (we can attach IGW to only one VPC)

Create Route Table

-> Create one Route Table (public-rt-demo-vpc)
-> Choose vpc and create it

Now We have 2 route tables

Note : Make previous route table as 'private-rt-demo-vpc'

-> Goto route table and attach only one subnet

Private Route Table should have Private Subnet
Public Route Table should have Public Subnet

Making Subnet as public
+++
-> Goto public Route Table -> Edit Routes

-> Add Destination as 0.0.0.0/0 and Target as IGW -> Save

-> Subnet Associations -> Edit SNET -> Select Public Subnet

Create EC2 (Public EC2)

-> Choose AMI
-> Select VPC
-> Select Public Subnet
-> Assign Public IP as Enable
-> Add SSH and Http Protocols
-> Download KeyPair
-> Launch Instance

Note: Goto VPC and Enable DNS Host Enable

Create EC2 (Private EC2)

-> Choose AMI
-> Select VPC
-> Select Private Subnet
-> Assign Public IP as Enable
-> Add SSH (source : custom, Range : 10.0.0.0/16)
-> Download KeyPair
-> Launch Instance

Open Mobaxterm
++++++++++++++
-> Connect to Public EC2
-> Connect from Public Ec2 to private EC2 using SSH client command
(upload keypair file to execute ssh command)

A Virtual Private Cloud (VPC) is a private network that is isolated from other private networks in AWS. It allows the user to customize the IP ranges used, the security and networking settings.

Each region has a default VPC in which the user can deploy EC2 instances. This allows a plug-and-play operation. On the backend, few things happen transparently to the user when the resources are launched in the default VPC.

However, if the user wants to have as much control as possible on the resources used in AWS, more likely a custom VPC will be required.

At the time of creation, the user must specify a set of IP addresses that will be used with that VPC. This set of IP addresses is called CIDR (which cannot be changed after VPC creation).

The CIDR can be split into subnets into which the EC2 instances are launched.

A public subnet means that the EC2 instances from that subnet have internet access, whereas the EC2 instances from the private subnet do not have internet access(there are few ways how to provide internet access to the EC2 instances from the private subnet)

Sometimes it is required to provide connectivity between VPCs because:

1) Two departments that require full access to each otherâ€™s resources

2) Multiple customers that must not be able to talk to each other, but still be able to provide access to your resources to each customer

To solve this problem, some sort of logical connection is required between the VPCs. We can achieve that using VPC Peering.

What is VPC Peering
++++++++++++++++++++
VPC Peering: IPV4 or IPV6 traffic routes between VPCs created to establish communication between one or more multiple VPCs.

AWS definition:
+++++++++++++++
â€œA VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses. Instances in either VPC can communicate with each other as if they are within the same network. â€œ

1) Through VPC Peering, traffic stays within the AWS network and not go over the internet.

2) Non-overlapping CIDRs â€“ The 2 VPCs you are trying to peer, must have a mutually exclusive set of IP ranges.

3) Transitive VPC Peering â€“ not allowed i.e

(If VPC A & B have peered and VPC A & C have peered, VPC B & C cannot share contents until there is an exclusive peering done between VPC B & C)

Will VPC Cost me?
++++++++++++++++++++
No. VPC itself wonâ€™t cost you, however, the resources deployed inside the VPC and data transfers are done will cost you.

How to establish peering between 2 VPCs in AWS
++

Since every AWS region comes with a default VPC, letâ€™s connect a default and custom VPC through peering for this article.

Create a custom VPC
++++++++++++++++++++
Name : ashokit_custom_vpc

Name your VPC and use the IPV4 CIDR range of your choice, I have used 10.0.0.0/16 for this example.

Note: Following components would be auto-created when you are creating a new VPC

Route Table
Security Group
NACL â€“ Network Access Control List

Creating a Subnet in VPC
+++++++++++++++++++++++++
VPC (Virtual Private Cloud) is your private cloud inside AWS as we mentioned earlier. typically we have to divide our big network into multiple subnets so we can place different resources.

For example, A Subnet allowed to connect to a public network for webserver hosting, A Subnet for a database with no public access and monitored IN/OUT connections (private subnets) etc.

Subnet Name : ashokit_aws_subnet

Creating EC2 instances
++++++++++++++++++++++
Letâ€™s create EC2 instances under each VPC (default-vpc and custom-vpc).

For the instance we are creating under the Custom VPC add the following User_Data, A Startup script to install the apache HTTPD service

```
#!/bin/bash

yum update -y
yum install -y httpd.x86_64
systemctl start httpd.service
systemctl enable httpd.service
echo "Hello world from $(hostname -f)" > /var/www/html/index.html
```

Creating Internet Gateway for the Custom VPC created
++
As we have mentioned earlier, we are going to use the EC2 instance we have created under the custom VPC for hosting an Apache HTTP server.

In order to allow traffic, we must create a Gateway.

For the EC2 under Custom VPC, create an Internet Gateway and make sure to add the following entries in its Route Table

(all traffic with internet-gateway) (range : 0.0.0.0/0)

And associate the Custom VPC Public Subnet to it

Note: Security Group of both EC2 instances are set to allow SSH Connection from Global (anywhere) for testing purpose.

Now login to both the EC2 instances using their Public IP from Putty or Mobaxterm softwares.

To prove there is no connectivity between Instance A and Instance B ,

letâ€™s access webserver which is installed in Custom-VPC-EC2-Instance
(since we had added user-data and setup an apache httpd web server)

$ curl localhost:80   (It should give response)

Try to access custom-vpc-ec2-instance webserver from default-vpc-ec2-instance with below command

$ curl <private-ip>:80 (It should not connect)

Letâ€™s create VPC Peering to enable communication
++
To establish the connection, lets create VPC peering from both VPCs.

On the left navigation panel under VPC -> Peering Connections:

VPC (Requester) = ashokit_aws_custom_vpc
VPC (Accepter) = default_vpc

Now you would see the status Pending Acceptance which means, Requestor has sent a request to the peer
now target VPC needs to accept the request.

Go to VPC Peering -> Click on Actions -> Accept Request

Now we need to make entries in Route Tables

ashokit_aws_custom_vpc IP Range: 10.0.0.0/16
default_vpc IP Range: 172.31.0.0/16

Now navigate to Route Tables, in Default VPC RT(Route Table) -> Edit routes

Default VPC Route Table should have 3 routes
(Local + all traffic with IGW + ashokit_aws_custom_vpc IP Range)

172.31.0.0/16 - local
0.0.0.0/0 - Internet-gateway
10.0.0.0/16 - vpc

Do the same for CustomVPC Route Table and it would look as below
(Local + All traffic with IGW + Default VPC IP Range)

10.0.0.0/16 - local
0.0.0.0/0 - Internet-gateway
172.31.0.0/16 - vpc

Now try to check the connectivity

Note : Allow Traffic in VPC Security Groups.

Edit Security Group of Default and Custom VPC to allow traffic from each other

Default VPC Security Group looks like
SSH - 22 - all
HTTP - 80 - Custom VPC IP Range (10.0.0.0/16)

Custom VPC Security Group would look like
SSH - 22 - all
HTTP - 80 - default VPC IP Range (172.31.0.0/16)

Now it works like a charm using the PRIVATE IP of the destination VPC !!!

Access Webserver that is available in Custom VPC from Default-vpc-ec2-instance with below command
$ curl <private-ip>:80
