Lab 3 - Virtual Private Cloud and migration of your existing application into the VPC.

# Create Virtual Private Cloud Network, create subnets across availability zones, understand connectivity within and between subnets. Understanding NAT, ACLs and Routing Tables.

# -70 points

* Deliverables: 5 screenshots (3a-3e).png

1. Migrate application developed earlier on elastic beanstalk to VPC. Use both public and private cloud (Hybrid Cloud): Load balancer in the public cloud and the application along with the database resides in the private cloud.Migrating your existing application on Beanstalk into the VPC -30 points

* Deliverables: 1 screenshot (3f)

# Create Virtual Private Cloud and learn its features

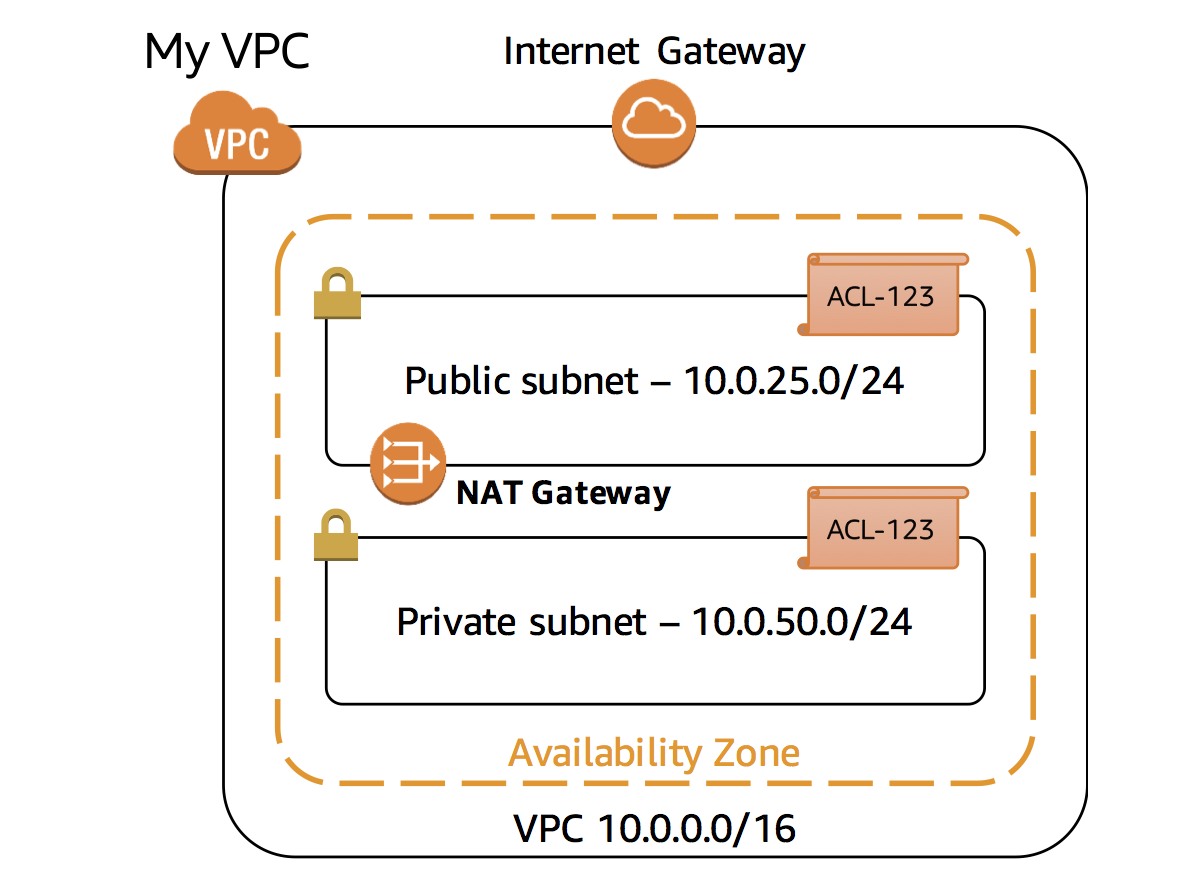
* **Reading –** 20 mins
  + [Virtual Private Cloud](https://aws.amazon.com/vpc/?vpc-blogs.sort-by=item.additionalFields.createdDate&vpc-blogs.sort-order=desc) : Make sure to go through benefits and use cases
  + [What is Amazon VPC?](https://docs.aws.amazon.com/vpc/latest/userguide/what-is-amazon-vpc.html) : Understand Key concepts for VPCs

## Task 1: Create an Elastic IP address - -5 min

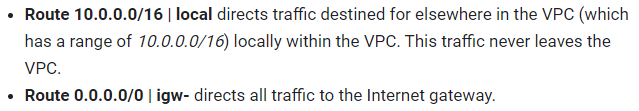
* Under services in the AWS management console, click on VPC
* Allocate an elastic ip address with default settings.
* Take a screen shot with the allocated elastic ip address (3a.png)

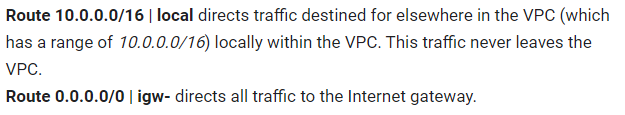
## Task 2: Create a VPC -35 mins

* Go to the VPC dashboard and launch the VPC wizard. The wizard makes your life much easier.
* Understand all the 4 VPC configurations
* Understand NAT gateways: [NAT gateways - Amazon Virtual Private Cloud](https://docs.aws.amazon.com/vpc/latest/userguide/vpc-nat-gateway.html)
* Create a **VPC with public and private subnets (Launch VPC wizard as shown in the demo video)**
* Configure the following:
  + VPC name: (your\_name)\_VPC
  + Public subnet IPv4 [CIDR](https://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing) : 10.0.x.0 /24 (x: any number )
  + Private subnet IPv4 [CIDR](https://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing): 10.0.y.0/24 (y :any number)
  + Note: The availability zones should be the first in the list
  + Allocate the elastic ip address you created earlier.
* Create your VPC and click ok. Go to **Your VPCs** section and look at the details of the VPC you just created. Take a screenshot (3b.png)
* Internet gateway connects your VPC to the internet. Check if the gateway is attached to your VPC.
* Go to **subnets** and make sure to see if you have two subnets one public and one private.
* Take a screenshot of the subnets along with their ipv4 CIDR addresses. (3c.png)
* Go to the public network and take a screen shot(3d.png) and also analyse the details.



## Task 3: Analyse your VPC -10 mins

* Each subnet is associated with a routing table which specifies the outbound traffic
* Routing tables for both your public and private networks
* The routing table for public network should have the following configuration: 
* The routing table for your private network should look like this:



* Go to **Network ACL**:
* A network access control list (**ACL**) is an optional layer of security for your VPC that acts as a firewall for controlling traffic in and out of one or more subnets.
* Edit the inbound rule in your VPC to allow only http traffic Take a screen shot when done (3e.png)

# Task b: -40 mins

# Migrating your existing application on Beanstalk into the VPC

(Note: Refer the pictorial representation below)

* Create a snapshot of your elastic beanstalk environment by saving your environment's configuration as an object in Amazon Simple Storage Service (Amazon S3)
* Make sure you have a snapshot of the RDS database which works along with your application.
* Refer [this](https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/environment-configuration-savedconfig.html) to create a saved configuration of your environment in beanstalk.
* Now, go to your saved configurations under your web app and choose **Launch environment.**
* Note: We are now going to launch this environment into our VPC created earlier.
* Give the new environment name: pes[yourSRN]-env
* Upload your application code by importing from S3 or locally.
* Click configure more options and modify network so that the app sits on the VPC you created.
* Assign your load balancer to a subnet in each Availability Zone (AZ) in which your application runs.
* Choose a subnet in each AZ for the instances that run your application. To avoid exposing your instances to the Internet, run your instances in private subnets and load balancer in public subnets.
* Now that you have your elastic beanstalk application in your VPC, take a screenshot (3f) clearly showing your environment name and the network in which your application resides.
* Make sure you can access your application through the URL provided in the beanstalk dashboard.
* Try editing your database (in configuration) and restore an existing snapshot of the RDS SQL database.
* Validate that your VPC and the beanstalk application if it’s configured properly.
* You are Done! But don’t forget to delete your beanstalk application and your VPC.

## Task 4: Delete your VPC -10 mins

* Terminate all your beanstalk environments and the application.
* Delete the NAT gateway (only the one associated with your VPC)
* Detach the internet gateway associated with your VPC after deleting the private and public subnet attached with the VPC.
* Delete the VPC (all subnets and routing tables are deleted in this process)
* Note: Make sure deletion is done in order to prevent any dependency errors.
* Release the elastic IP that you allocated.

Pictorial representation for your understanding


      Elastic Beanstalk and VPC topology with Amazon RDS
    

Reference: https://docs.amazonaws.cn/en\_us/elasticbeanstalk/latest/dg/vpc-rds.html