Introduction to AWS Virtual Private Cloud

**Lab Details:**

1. This lab walks you through to AWS Virtual Private Cloud (VPC). In this lab, we will create a VPC with private and public subnet, we will create a web applictaion inside the public subnet and a RDS database inside the private subnet,Configuring NACL and Route table to connect them to internet, and testing the RDS database connection on our publicaly available web application.
2. Duration: 00:55:00 Hrs
3. AWS Region: US East (N. Virginia)

**Tasks:**

1. Login to AWS Management Console.
2. Create a VPC.
3. Create private and public subnet for the VPC.
4. Create a EC2 instance in created VPC over public subnet.
5. Create a RDS instace and database in created VPC over private subnet.
6. Create a connection to RDS database on EC2 instance and test.

**Steps:**

1. Launch your lab environment by clicking on **Start Lab** button.
2. Once your lab environment is created successfully your **Console Login**button will be active, Now click on **Console Login** button, this will open your **AWS Console** Account for this lab in a new tab.
3. When you create an Amazon AWS VPC, you specify a set of IP addresses in the form of a Classless Inter-Domain Routing (CIDR) block (take 10.0.0.0/16 for example). For more information about CIDR notation and what "/16" means, see [**Classless Inter-Domain Routing**](http://en.wikipedia.org/wiki/CIDR)**You can assign a single CIDR block to a VPC. The allowed block size is between a /28 netmask and /16 netmask. In other words, the VPC can contain from 16 to 65,536 IP addresses.**.
4. Navigate to VPC by clicking on the “services” menu in the top,then click on “VPC” (in the “Networking & Content Delivery” section).
5. Now on VPC dashboard from the left menu list click on your VPCs.
6. Here you can see the list of all VPC, No need to do anything from existing and default VPC from the list we will create a new VPC for this lab.
7. Click on “**Create VPC**”.
8. **Name tag:**Enter a VPC name for identification to you VPC.
9. **IPv4 CIDR block:**Specify the CIDR block for your VPC. Enter 10.0.0.0/16
10. **IPv6 CIDR block:**No need to change this, make sure "No IPv6 CIDR Block" is checked.
11. **Tenancy:**No need to change this, make sure "Default" is selected.
12. Now click on **Yes,Create** Button.
13. You can see in the the list, your VPC is created successfully with your given CIDR range.
14. Now we are creating subnets.
15. Before we create a subnet, let’s understand the best practices for creating them.   
      
    You should create subnets across multiple availability zones, with each subnet residing within a single zone.   
    Creating subnets in and launching instances across multiple availability zones will ensure a high-availability environment.   
    When creating separate subnets for EC2 and RDS instances, each tier should have at least 2 subnets across availability zones.   
    In this lab we will create 2 subnets for our app EC2 instance and 2 subnets for our RDS database instance   
    in us-east-2a and us-east-2b Availability Zones as follows.   
      
    **App Tier:** 10.0.1.0/24(zone-a), 10.0.2.0/24(zone-b)   
    **Database (RDS):** 10.0.11.0/24(zone-c), 10.0.12.0/24(zone-d)   
      
    **Note:**Always choose the same Amazon Availability Zones for all tiers. For example, if you choose two zones for   
    high availability and use us-east-1a and us-east1b, then maintain those same 1a and 1b zones for all tiers.  
    This will minimize data transfer charges because data transfers between instances within the same availability zone are free.
16. Now Go to Subnets tab from the letf menu and click on **Create subnet**
    1. Create subnet-1 for app tier.
    2. **Name tag:**Enter a name for app subnet in Availability Zone a like "app-subnet-public-a"
    3. **VPC\*:**Select you VPC from the list.
    4. **Availability Zone:**Select us-east-1a
    5. **IPv4 CIDR block\*:**Enter the range 10.0.1.0/24
    6. Click on **Create**

Now 

* 1. Create subnet-2 for app tier.
  2. **Name tag:**Enter a name for app subnet in Availability Zone a like "app-subnet-public-b"
  3. **VPC\*:**Select you VPC from the list.
  4. **Availability Zone:**Select us-east-1b
  5. **IPv4 CIDR block\*:**Enter the range 10.0.2.0/24
  6. Click on **Create**
  7. Create subnet-1 for RDS database.
  8. **Name tag:**Enter a name for app subnet in Availability Zone a like "rds-subnet-private-a"
  9. **VPC\*:**Select you VPC from the list.
  10. **Availability Zone:**Select us-east-1c
  11. **IPv4 CIDR block\*:**Enter the range 10.0.11.0/24
  12. Click on **Create**

Now 

* 1. Create subnet-2 for RDS database.
  2. **Name tag:**Enter a name for app subnet in Availability Zone a like "rds-subnet-private-b"
  3. **VPC\*:**Select you VPC from the list.
  4. **Availability Zone:**Select us-east-1d
  5. **IPv4 CIDR block\*:**Enter the range 10.0.12.0/24
  6. Click on **Create**

1. Now we are creating **Internet Gateway**, By default instances that are launched into a VPC can't communicate with the Internet.   
   However, to enable Internet access we need to attach an Internet gateway to the VPC.
2. Go to Internet Gateways in the navigation pane and click “**Create Internet Gateway**”.
3. Now Enter a **Tag Name :** and click on create.
4. Now we need to attach the gateway to our VPC.
5. Select the Internet gateway you created from the list, and click on Action.
6. From the action menu click on “**Attach to VPC**”
7. Now Select the VPC you created from the list and click on **Attach.**
8. Now Create Route Tables
9. A route table contains a set of rules, called routes, that determine where network traffic is directed.  
   Each subnet in your VPC must be associated with a route table that will control that subnet’s routing. You can associate multiple   
   subnets with a single route table; however, you can only associate a subnet with one route table.  
   Creating a VPC automatically creates a main route table which, by default, enables the instances in your VPC to communicate with one other.
10. As a best practice create separate route tables for each tier.  
    This will provide more control in maintaining the security of each subnet.
11. Go to Route Tables in the navigation pane and click on “**Create Route Table**”.
12. **Name Tag:**Enter a name to identify route table for app tier.
13. **VPC:**Select your VPC from the list.
14. click on Yes,create.
15. Repeat the same steps to create route table for RDS database tier.
16. Now **associate the subnets** to the route tables.
17. Click on one route table and go to the Subnet Associations tab. and click on “**Edit**”.
18. Associate each tier’s subnets separately to the dedicated route tables as follows.
    1. **APP route table**—Associate 10.0.1.0/24 and 10.0.2.0/24.
    2. **RDS route table**—Associate 10.0.11.0/24 and 10.0.12.0/24.
19. Do not associate any subnets with the main route table.

Now navigate to route tables and select main route table and route table for app tier to add a route to allow Internet traffic to the VPC.

1. Go to Routes **App route table** and specify the following values:   
   **Destination:** 0.0.0.0/0  
   **Target:** Select “Internet Gateway” from the dropdown menu.  
   Repeat the same steps for **Main Route table of your VPC**.
2. Create **AWS Security Groups**  
   This process is similar to creating an SG (Security Group) in EC2. Create separate security groups for APP, DB (RDS) instances.  
   Click on **Security Groups** from left menu on VPC dashboard.
   1. **Create Security Group for App server.**, Click on **Create Security Group**

**Name tag**: Enter a Name tag for APP server like**APP\_SG**  
**Description**: Enter Description.  
**VPC**\*: Choose the VPC you created earlier(for this lab) from the list.  
Click on **Yes,Create** button.  
Once your security group created successfully, Select it from the list.  
Now goto **Inbound Rules** tab of security group details.  
Click on **edit** button.

* + 1. For **HTTP**, Click on “Add Another Rule”,  
       Choose Type: Custom TCP Rule  
       Port Range :80   
       Source: Enter 0.0.0.0/0
  1. **Create Security Group for DB server.**, Click on **Create Security Group**

**Name tag**: Enter a Name tag for DB server like**DB\_SG**  
**Description**: Enter Description.  
**VPC**\*: Choose the VPC you created earlier(for this lab) from the list.  
Click on **Yes,Create** button.  
Once your security group created successfully, Select it from the list.  
Now goto **Inbound Rules** tab of security group details.  
Click on **edit** button.

* + 1. To add **MySQL**,   
       Choose Type: Custom TCP Rule  
       Port Range: 3306   
       Source: Select anywhere(0.0.0.0/0)

1. **Launch EC2 App server**   
   Now go to Services->EC2 ->Launch Instance.  
   On the Configure Instance Details page, from the Network list choose the VPC that you created previously.
2. From the Subnet dropdown select one of your app server subnet (10.0.1.0/24 OR 10.0.2.0/24) from the Subnet list.
3. Select the “**Public IP**” check box to request that your app instance receive a public IP address. This is required when you don’t have a NAT instance, but your instance requires Internet access.
4. **Configure Instance Details (Advanced Details)** - In User data Enter the following script, This script is run the first time the instance is launched. It installs a web server on your EC2 instance, and runs an app that can be configured to point to your MySQL RDS instance. After you configure your RDS instance, it will present you can enter you RDS details and test connection.
5. #!/bin/bash -ex
6. sudo yum -y update
7. sudo yum -y install httpd php mysql php-mysql
8. chkconfig httpd on
9. service httpd start
10. cd /var/www/html
11. wget https://s3.amazonaws.com/whizlabs-vpclab-sample/vpc\_sample.zip
12. sudo yum install unzip
13. unzip vpc\_sample.zip
14. chown apache:root /var/www/html/
15. On the Configure Security Group page, select the option “Select an existing security group” and then select the APP\_SG01 (which we created previously). Click “Review and Launch”.
16. Create a Database Subnet Group
    1. When creating a DB instance in a VPC, you must select a DB subnet group.
    2. On the **Services** menu, click **Relational Database Service** or **RDS**.
    3. In the left navigation pane, click **Subnet groups**.
    4. Click Create DB Subnet Group, then configure:
    5. **Name:** Enter a subnet group name.
    6. **Description:** Enter some description.
    7. **VPC:** *Select your created VPC from the list*
    8. Add Your Two subnets(10.0.11.0/24 and 10.0.12.0/24) which we created for RDS (Private Subnets)
    9. Now click on create
17. **Create an Amazon RDS Database**

You are now ready to launch an Amazon RDS database running MySQL.

1. In the left navigation pane, click on **Databases**.
2. Click Create database
   1. On **Step 1**, configure the following:
   2. Click MySQL
   3. Click Next
   4. On **Step 2**, configure the following:
   5. If ask, Click **Dev/Test - MySQL**
   6. Next
   7. For **DB instance class**, select **db.t2.micro - 1 vCPU, 1 GiB RAM** (at the top).
   8. In the **Settings** section, configure:
   9. **DB instance identifier:** Enter a db instance name.
   10. **Master username:** Enter a username.
   11. **Master password:** Enter a password.
   12. **Confirm password:** Confirm password./li>
   13. Click Next
   14. In the **Network Security** section, configure following:
   15. **Virtual Private Cloud (VPC):** *Select your VPC from the list*
   16. Choose the Subnet group you greated in your VCP previously.
   17. **VPC security groups:**
   18. **Choose existing security groups which you created for RDS previously.**
   19. **Database name** Enter a database name.
   20. In the **Backup** section, for **Backup retention period**, select *0 days*
   21. Scroll to the bottom of the page, then click Create database
   22. Click View DB instance details
3. It will take a few minutes for you MySQL database to become available.
   1. In the left navigation pane, click **Instances**.
   2. Click refresh every 60 seconds until the instance status not change to **available**.
4. Connect Your app server to Your Database
5. In this task, you will connect app server (in your Public subnet) to your database (in your Private subnet).
6. Obtain Your MySQL Database Endpoint
7. Before you can connect your app server to your database, you need to know the *endpoint* of the RDS instance, that is the address of your RDS instance.
8. Click your **mydb** instance.
9. Under the **Connect** section, copy the **Endpoint** to your clipboard.

You RDS endpoint should look similar to:

*yourdb.cdsfiu989d.us-west-2.rds.amazonaws.com*

1. **Test and Connect to Your Database with web app**
   1. Return to the browser tab that is displaying your web app, then configure:
   2. **Endpoint:** Enter your RDS MySQL endpoint.
   3. **Database:** Enter your RDS MySQL database name.

**Username:**Enter your RDS MySQL database username.

* 1. **Password:**Enter your RDS MySQL database password.
  2. Click on Submit
  3. Once connected, you should see the message "App Server to RDS Database Connection Establish Successfully."
  4. If you found any error please, re-check you security group pemissions.
  5. Once you see the connection success message, it means you have successfully setup a private subnet for RDS database and public subnet for you web app.

1. You have successfully completed the lab.
2. Once you completed the steps click on End Lab from your whizlabs dashboard.