**Transform Data Using Apache Spark on Amazon EMR

06:45 AM

Configure a Subnet for EMR Cluster
Globomantics is an analytics firm that processes raw data/logs to reap rich insights. Your role as a Data Engineer is to pre-process/massage data, so that the analytics team can use the data for further data modelling. You will use the Spark ETL stack on Amazon EMR cluster to transform the data, in this challenge you will set up a subnet that will be used by Amazon EMR cluster.

Note: It is recommended that you wait for this lab to finish loading the environment prior to beginning (about 3-5 minutes). We have adjusted the lab time to account for this wait.

Go to AWS Management Console and in Services search for VPC under the Networking & Content Delivery section.

Go to VPC, in the left panel, click on Your VPCs, and click on Create VPC.

Under Name tag give a name to your VPC EMR_VPC.

Under IPv4 CIDR block enter 172.31.0.0/16 and click on Create VPC. Shortly afterwards you'll see a green notification appear at the top that says you successfully created a VPC.

In the left panel, click on Internet Gateways and then click on Create internet gateway.

Under Name tag enter the name EMR_IGW and click on Create internet gateway.

Click on Actions, select Attach to VPC, under available VPCs select the VPC ending with the name EMR_VPC and click on Attach internet gateway.

In the left panel click on Route Tables, select the Route table ID whose VPC ID ends with the name EMR_VPC, select the tab Routes and click on Edit routes.

Click on Add route, under Destination enter 0.0.0.0/0, under Target the select Internet Gateway with the name ending EMR_IGW and click on Save changes.

In the left panel click on Subnets, click on Create subnet, under VPC ID select the VPC ending with the name EMR_VPC, under Subnet name enter the name as EMR_public_subnet. Select the Availability Zone as us-west-2c, under IPv4 CIDR block enter 172.31.0.0/20 and then click on Create subnet.

You will observe the message highlighted in green that You have successfully created 1 subnet with your subnet id and you will find your subnet state as Available in green. You have created a public subnet which is a subnet that's associated with a route table that has a route to an Internet gateway. This connects the VPC to the internet and to AWS EMR cluster which we will create in the next challenge.

##

Configure an Amazon EMR Cluster to Run Spark Jobs
In this challenge you will create an Amazon EMR cluster to run the data engineering jobs using Spark stack

Use the search bar at the top of the AWS console to navigate to the EMR service.

Click on Create cluster, then click Go to advanced options.

Under Release select emr-5.31.0, check the box for Spark 2.4.6 and click on Next.

In Networking click the Network dropdown and select the VPC ending with the name EMR_VPC. Its subnet will be selected automatically in the EC2 Subnet dropdown.

Under Cluster Nodes and Instances, change each Instance type to m4.large (click the pencil icon next to each), then click Next.

Click Next again. On the Security Options page, set the EC2 key pair to pluralsight, then click Create cluster.

The cluster should take 5-10 minutes to start, but could take longer. Wait until the cluster state changes from Starting to Waiting. The status will change from Starting to Running before it reaches Waiting.

Under the Security and access section you will find Security groups for Master. Click on the link beside that, which will open a new tab. In the new tab, select the option with a Security group name of ElasticMapReduce-master, then go to the Inbound rules tab and click on Edit inbound rules. Scroll down and click Add rule.

Set the Type as SSH, change the Source type from Custom to Anywhere.

Click Save rules, then you can close that tab and go back to the EMR Cluster page.

You will observe your cluster in the Waiting state. In the Network and hardware section you will see your nodes are running. You will use the master node to run your spark jobs and apply data transformations which will coordinate with the slave nodes to execute the job in the distributed cluster environment and fetch the results.