LSEG-Application Services

Assignment

Tilan de Silva

Contents

1. Basic Requirements

In this document, Steps that have been used to complete the assignment have been documented. Basically, the following tools and platforms were used to complete tasks.

- Terraform
- Ansible
- AWS

First, a Linux workstation machine was needed to create to run the tasks. For that either an ec2 instance on AWS or a VM running on your machine can be used and Linux distribution should be Redhat. In my case, a Centos VM running on my laptop was used. Before using any tool or platform, there are some dependency packages that need to be installed on the workstation machine. The following packages were installed on my workstation machine.

- Terraform
- Ansible
- AWS CLI
- Python3
- Python3-pip
- Boto3

The instructions have been provided on the README file to install the above packages and a bash script file has been provided to install all the packages automatically. After that, an AWS account or IAM user account with an access key and secret keys are needed to configure AWS CLI on the workstation machine and run the terraform files. After configured AWS CLI, all files that need to complete the tasks were copied to the one directory under the root user home. All the tasks should be executed under the root user home directory with root permission. An RSA key was generated to establish a secure connection with the web server. The key sharing with the web server is done by Terraform. After completing basic requirements, AWS resources can be provisioned using the Terraform.

2. Clould Insfrastructure

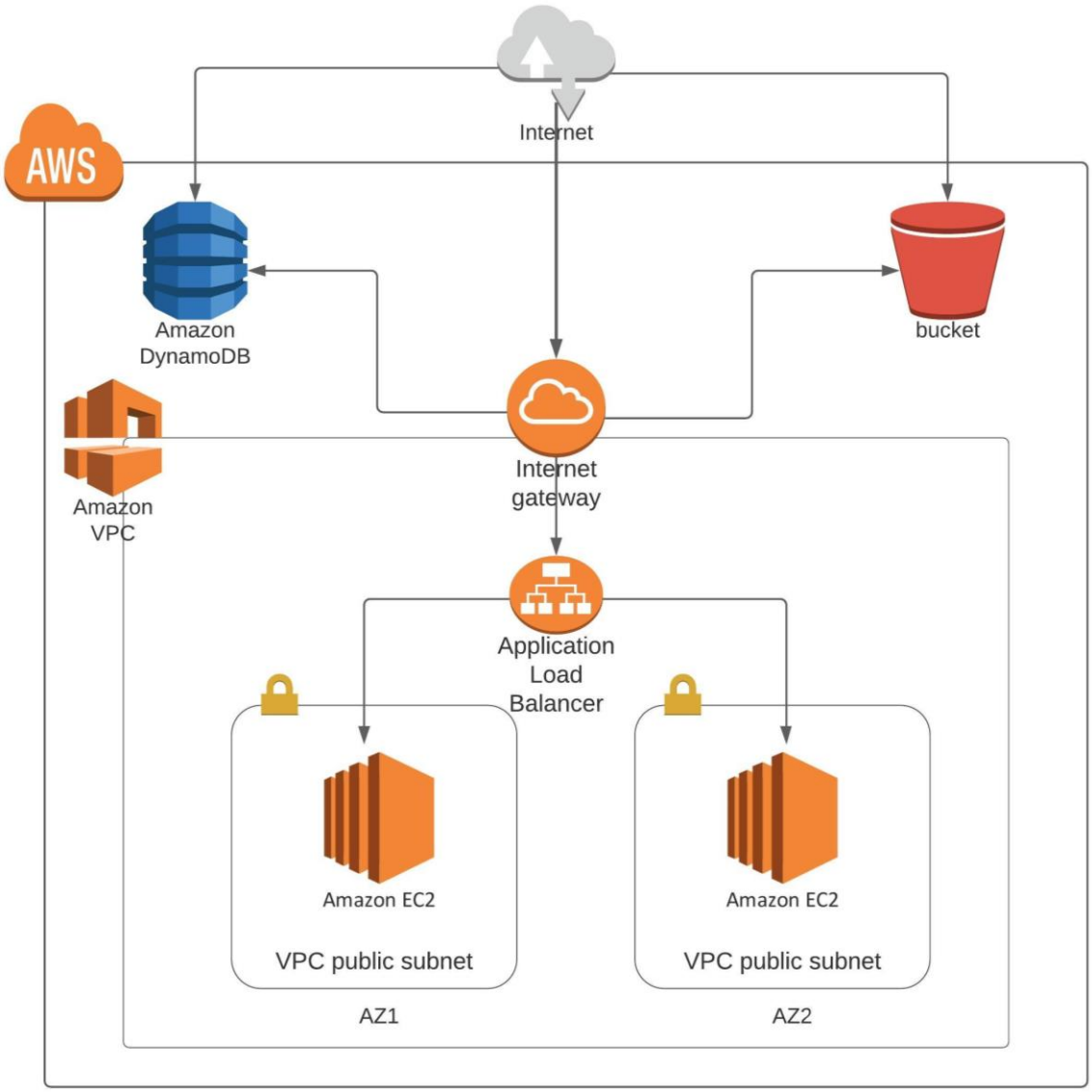

Figure 1: System Infrastructure Diagram

The resources that need to carry out this assignment are built on the AWS cloud platform. Figure 1 illustrates the infrastructure diagram of the resources. Following resources were used to build the infrastructure according to accomplish the requirement of the assignment,

- VPC
- Internet Gateway
- Public Subnets
- EC2 Instances
- Routing Table
- DynamoDB
- S3 Bucket

3. Execution and Results

To provision, the above resources using a single click of a button as per the supplementary requirement, terraform which is a software tool for changing building and versioning infrastructure safely and efficiently has been used. All the resources are created on the Asia Pacific (Mumbai)ap-south-1 region according to the instructions given in terraform file except the S3 bucket and DynamoDB since both are cannot be created specifying a region. After executing the terraform files,

- a VPC (name "main") will be created with a network address "10.0.0.0/22",
- internet gateway will be created by associating with the VPC,
- Two public subnets will be created on availability zones (ap-south-1a, ap-south-1b) in the region by associating with the VPC,
- a routing table will be created associating with VPC and IGW,
- subnets and routing table will be associate,
- a key pair will be generated to connect to the EC2 instance but need to provide a public key of the workstation machine,
- a security group will be created allowing only incoming traffic from ports 22 and 80 to provide more security,
- and finally, two ec2 instances will be created on availability zones of the region to provide the high availability by associating with other resources mentioned above.

Except for these resources, an S3 bucket and DynamoDB will be created in order to save the logs of web servers. Even if an application load balancer has been mentioned in figure 1, it will not be created in actual setup.

After applied the terraform plan, to complete other tasks, Ansible has been used. Before implementing the ansible files there are some configurations that need to be done. The manage node's IPs need to be mentioned in the inventory file in order to apply the tasks which are executed through the ansible. The EC2 instances which are created using terraform have public IPs. These IPs need to be put automatically into the inventory file in order to minimize human intervention. This task also established using a terraform local provisioner. But in this setup, only one EC2 instance's IP is copied to the inventory file, and all other tasks were carried out using that EC2 instance. By default, ansible has its own log system but it has been disabled. In order to activate it and add the inventory file path, an ansible.cfg file should be created where

all the ansible playbooks are executed. Following entries should be included in the ansible.cfg file.

- [defaults]
- inventory = /root/answer/inventory
- log_path = /var/log/ansible.log

To install the webserver (Apache), create the web content, start the webserver and add cron jobs to execute the other tasks as mentioned in the assignment, an Ansible playbook was created, and the following ansible modules were used to create it.

- Yum
- Copy
- Service
- Cron

Yum, copy, service modules are executed on the EC2 instance while cron module runs on the workstation machine. It has done using the “delegate_to: localhost” option.

After executing, the above mentioned Ansible playbooks, As mentioned in the assignment, periodically (Every 30min) run these tasks; checking the webserver is running or not and start it if it is not, checking the server is serving the expected content (checking whether it returns 200 status code or not), saving results of the service checks in AWS DynamoDB along with the timestamps, and notifying to the App Support team via an email if the scrip detects any errors. The above tasks have also been completed using an ansible playbook and the following modules were used,

- Service
- Uri
- Command
- Mail

The result saving task is completed using the command module, and it is executed the bash script which includes the AWS command that is used to insert the items into DynamoDB. Both the command module and mail module run on the workstation machine. A fake email address has been used in the mail module.

In order to carry out the following tasks; collecting log files and content of the web server daily and create one compressed file, moving the compressed file to the same location as the script is running, Uploading the compressed file to an S3 bucket, and removing the compressed file if the uploading is successful and informing to the support team if it is not successful, an Ansible playbook has been used. Following Ansible modules has been used,

- Uri
- Shell
- Copy
- Archive
- Fetch
- amazon.aws.aws_s3
- file
- mail

The amazon.aws.aws_s3 module, file module, and mail module are executed on the workstation machine. Regarding all the modules that have been used in all the playbooks and tasks of modules are commented on playbooks. After successfully executing all the terraform and ansible files the assignment's requirements could be accomplished. The following files are available on this link:

- README file
- dependecies.sh file
- dynamodb1.sh file
- dynamodb2.sh file
- dynamodb3.sh file
- user-data.sh file
- main.tf file
- output.tf file
- provider.tf file
- variables.tf file
- apache-install.yml
- log-save.yml file
- webserver-status-check.yml file

- Ansible.cfg file
- inventory file

README file provides all the instruction how to setup the environment and instructions how to execute the scripts.