CS312 Project 2

Background

What will we do? We will write infrastructure provisioning scripts using **Terraform** to set up AWS resources.

Requirements

- Install Terraform
- Install AWS CLI

Credentials Required

- Use your **AWS Academy Learner Lab**.
- Click **"AWS Details"** in the top right of your Learner Lab page to get your access keys.

Environment Configuration

Follow the steps in the link to configure Terraform:

Terraform AWS Setup Guide

Terraform Configuration for Minecraft Server - `main.tf`

Terraform Block

```
terraform {
 required_providers {
 aws = {
 source = "hashicorp/aws"
 version = "~> 4.16"
 }
 }

 required_version = ">= 1.2.0"
}
```

AWS Provider Configuration

```
provider "aws" {
 region = "us-west-2"
 shared_credentials_files = ["~/.aws/credentials"]
}
```

EC2 Instance Resource

```
resource "aws_instance" "app_server" {
 ami = "ami-04999cd8f2624f834"
 instance_type = "t2.large"
 key_name = "minecraft"
 subnet_id = "subnet-0f116fa7a42d0a000"
 associate_public_ip_address = true

 provisioner "remote-exec" {
 script = "restart-w-reboot.sh"
 }

 connection {
 type = "ssh"
 user = "ec2-user"
 private_key = file("~/downloads/minecraft.pem")
 host = self.public_ip
 }

 tags = {
 Name = "Minecraft Server"
 }
}
```

`restart-w-reboot.sh` script

create this scipt in the same direcotry as your terraform configuration

```
#!/bin/bash

# Install Java
sudo amazon-linux-extras enable corretto11
sudo yum clean metadata
sudo yum install -y java-11-amazon-corretto

# Make sure the Minecraft directory exists
mkdir -p /home/ec2-user/minecraft

# Upload the systemd service file
cat <<EOF | sudo tee /etc/systemd/system/minecraft.service
```

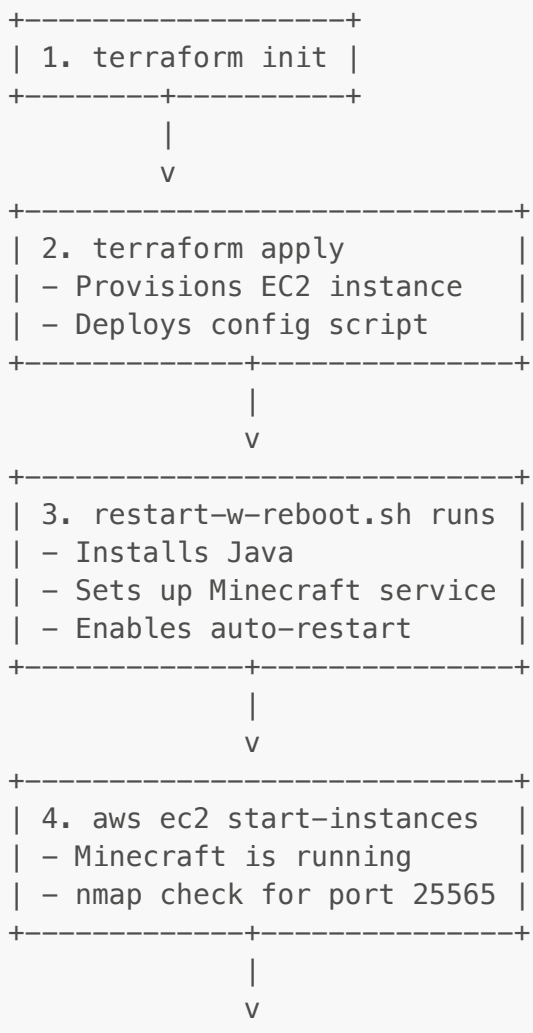
```
[Unit]
Description=Minecraft Server
After=network.target

[Service]
WorkingDirectory=/home/ec2-user/minecraft
ExecStart=/usr/bin/java -Xmx1024M -Xms1024M -jar /home/ec2-
user/minecraft/minecraft_server.jar nogui
Restart=always
User=ec2-user

[Install]
WantedBy=multi-user.target
EOF

# Reload systemd and enable the service
sudo systemctl daemon-reload
sudo systemctl enable minecraft
sudo systemctl start minecraft
```

Pipeline Diagram



```
+----------------------------+
| 5. aws ec2 reboot-instances |
| - Validate auto-restart     |
| - Wait, recheck with nmap   |
+-------------+--------------+
              |
              v
+----------------------------+
| 6. Client connects to IP:25565 |
+----------------------------+
```

Command List with Explanations

1. `terraform init`

- Initialize the working directory containing Terraform configuration files

2. `terraform fmt`

- Format and validate configuration files for readability and consistency

3. `terraform validate`

- Validate the configuration files and check for syntax or logical errors

4. `terraform apply`

- Apply the configuration — you'll be prompted to enter "yes" to proceed

5. `aws ec2 start-instances --instance-ids i-09b8a9856f287a703 --region us-west-2`

- Starts the EC2 instance

6. `nmap -sV -Pn -p T:25565 35.81.205.57`

- Confirms the host is up and port 25565 is open

7. `aws ec2 reboot-instances --instance-ids i-09b8a9856f287a703 --region us-west-2`

- Reboot the EC2 instance and check if the automatic restart works

8. `nmap -sV -Pn -p T:25565 35.81.205.57`

- The state should be closed

9. Wait 1–2 minutes

10. `nmap -sV -Pn -p T:25565 35.81.205.57`

- After waiting: host should be up and port open again

How to Connect to the Minecraft Server

Once the server is running and port **25565** is open (confirmed via `nmap`), you can connect using the Minecraft client by entering the **public IP address** of the EC2 instance followed by `:25565`.

Resources and References

- Lab Week 9: Infrastructure as Code
- Terraform AWS Setup Guide