To configure virtual machines or cloud instances with different network configurations (e.g subnets, VPCs, security groups and network access control lists) AWS the following prerequisites must be in place.

Terraform must be installed on your computer.

An AWS account must be set up.

In this case, we will use AWS CLI to implement the actions.

Link to install terraform [Terraform](https://developer.hashicorp.com/terraform/install)

Link to download AWS CLI <https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>

**Step 1: Define Your Provider and Initialize Terraform**

First, make sure your AWS CLI is configured correctly and that Terraform is installed and initialized.

Create a file named main.tf and define your provider:

hcl

Copy code

provider "aws" {

region = "us-east-1"

}

**Step 2: Create a VPC**

Define a VPC resource in your main.tf:

hcl

Copy code

resource "aws\_vpc" "main\_vpc" {

cidr\_block = "10.0.0.0/16"

tags = {

Name = "main\_vpc"

}

}

**Step 3: Create Subnets**

Add subnets to the VPC:

hcl

Copy code

resource "aws\_subnet" "public\_subnet" {

vpc\_id = aws\_vpc.main\_vpc.id

cidr\_block = "10.0.1.0/24"

availability\_zone = "us-east-1a"

map\_public\_ip\_on\_launch = true

tags = {

Name = "public\_subnet"

}

}

resource "aws\_subnet" "private\_subnet" {

vpc\_id = aws\_vpc.main\_vpc.id

cidr\_block = "10.0.2.0/24"

availability\_zone = "us-east-1a"

tags = {

Name = "private\_subnet"

}

}

**Step 4: Create an Internet Gateway**

Create an internet gateway to allow traffic to and from the public internet:

hcl

Copy code

resource "aws\_internet\_gateway" "main\_igw" {

vpc\_id = aws\_vpc.main\_vpc.id

tags = {

Name = "main\_igw"

}

}

**Step 5: Create Route Tables**

Create and associate route tables:

hcl

Copy code

resource "aws\_route\_table" "public\_route\_table" {

vpc\_id = aws\_vpc.main\_vpc.id

route {

cidr\_block = "0.0.0.0/0"

gateway\_id = aws\_internet\_gateway.main\_igw.id

}

tags = {

Name = "public\_route\_table"

}

}

resource "aws\_route\_table\_association" "public\_subnet\_association" {

subnet\_id = aws\_subnet.public\_subnet.id

route\_table\_id = aws\_route\_table.public\_route\_table.id

}

**Step 6: Create Security Groups**

Create security groups for public and private instances:

hcl

Copy code

resource "aws\_security\_group" "public\_sg" {

vpc\_id = aws\_vpc.main\_vpc.id

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

ingress {

from\_port = 80

to\_port = 80

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

ingress {

from\_port = -1

to\_port = -1

protocol = "icmp"

cidr\_blocks = ["0.0.0.0/0"]

}

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

tags = {

Name = "public\_sg"

}

}

resource "aws\_security\_group" "private\_sg" {

vpc\_id = aws\_vpc.main\_vpc.id

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["10.0.1.0/24"]

}

ingress {

from\_port = 80

to\_port = 80

protocol = "tcp"

cidr\_blocks = ["10.0.1.0/24"]

}

ingress {

from\_port = -1

to\_port = -1

protocol = "icmp"

cidr\_blocks = ["10.0.1.0/24"]

}

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

tags = {

Name = "private\_sg"

}

}

**Step 7: Create EC2 Instances**

Create public and private EC2 instances:

hcl

Copy code

resource "aws\_instance" "public\_instance" {

ami = "ami-0c55b159cbfafe1f0" # Amazon Linux 2 AMI

instance\_type = "t2.micro"

subnet\_id = aws\_subnet.public\_subnet.id

security\_groups = [aws\_security\_group.public\_sg.name]

tags = {

Name = "public\_instance"

}

}

resource "aws\_instance" "private\_instance" {

ami = "ami-0c55b159cbfafe1f0" # Amazon Linux 2 AMI

instance\_type = "t2.micro"

subnet\_id = aws\_subnet.private\_subnet.id

security\_groups = [aws\_security\_group.private\_sg.name]

tags = {

Name = "private\_instance"

}

}

**Step 8: Apply the Terraform Configuration**

Run the following commands to apply the configuration:

sh

Copy code

terraform init

terraform plan

terraform apply

**Step 9: Test Network Connectivity**

1. **SSH into the Public Instance:**
   * Get the public IP of the public instance from the AWS console or Terraform output.
   * Use an SSH client (like PuTTY or the terminal) to connect:

sh

Copy code

ssh -i your\_key.pem ec2-user@public\_instance\_ip

1. **Ping the Private Instance:**
   * SSH into the public instance.
   * Use the private IP of the private instance to test connectivity:

sh

Copy code

ping private\_instance\_ip

1. **Telnet:**
   * Ensure telnet is installed on the public instance:

sh

Copy code

sudo yum install -y telnet

* + Use telnet to test connectivity to a specific port:

sh

Copy code

telnet private\_instance\_ip 80

After creating the separate .tf files, the next step is to initialize terraform by running the code below

terraform init

A screenshot of a computer program

Description automatically generated

To be able to successfully run the configuration, I need to configure Terraforn to use AWS CLI. This can be done on the same CLI window.

**Verify AWS Credentials:** Ensure that you have valid AWS Access Key ID and Secret Access Key.

**Set Up AWS CLI:** Configure the AWS CLI to make sure your credentials are working correctly. Run the following command in your terminal:

A computer screen shot of a black screen

Description automatically generated

To use the AWS CLI console, access keys need to be created for the user account.

**Creating Access Keys**

To create a new access key for an IAM user:

1. **Sign in to the AWS Management Console:** Go to the [AWS Management Console](https://aws.amazon.com/console/).
2. **Navigate to the IAM Console:** In the services menu, search for and select **IAM**.
3. **Select Users:** In the IAM dashboard, select **Users** from the navigation pane.
4. **Choose the User:** Click on the IAM user for which you want to create an access key.
5. **Security Credentials Tab:** Go to the **Security credentials** tab.
6. **Create Access Key:** Scroll down to the **Access keys** section and click **Create access key**. Follow the prompts to create a new access key.
7. **Download the Credentials:** Download the credentials file or copy the Access Key ID and Secret Access Key to a secure location. You will not be able to see the Secret Access Key again after this step, so make sure to save it securely.

A screenshot of a computer

Description automatically generated

After successfully applying the terraform configuration, the instance will be created.

A summary of the configuration will be displayed for confirmation.

The instance ID generated for this task is : id=i-0ab4c21be962806ac]

A screenshot of a computer

Description automatically generated

Web Instance ID: id=i-0d8249ba6a728ebe0 from the AWS Management Console

A screenshot of a computer

Description automatically generated

To test connectivity from one instance to another.

Connect to an Instance using any method. For this task, I will be connecting to my instance via RDP

A blue background with white lines

Description automatically generated

The request keeps timing out A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated