# VPC PEERING (by TERRAFORM Script)

Creating one ec2 instance manually

Connecting to the instance

apt install unzip

curl "https://awscli.amazonaws.com/awscli-exe-linux-x86\_64.zip" -o "awscliv2.zip"

unzip awscliv2.zip

sudo ./aws/install

aws configure 🡪 give the details mentioned below frm the IAM🡪 user

access key

secret access key

region –us-east-2

formate –table

mkdir terraform

cd terraform

vi terraformblock.tf

terraform {

required\_providers {

aws = {

source = "hashicorp/aws"

version = "5.66.0"

}

}

}

Vi provider.tf

Provider “aws” {

Profile = “default’

Region = “ us-east-2”

}

VPC PEERING SCRIPT (Peering.tf)

# Create VPC 1

resource "aws\_vpc" "vpc1" {

cidr\_block = "10.0.0.0/16"

tags = {

Name = "VPC1"

}

}

# Create Subnet for VPC 1

resource "aws\_subnet" "subnet1" {

vpc\_id = aws\_vpc.vpc1.id

cidr\_block = "10.0.1.0/24"

availability\_zone = "us-east-2a"

map\_public\_ip\_on\_launch = true

tags = {

Name = "Subnet1"

}

}

# Create Internet Gateway for VPC 1

resource "aws\_internet\_gateway" "igw1" {

vpc\_id = aws\_vpc.vpc1.id

tags = {

Name = "IGW1"

}

}

# Create Route Table for VPC 1 and associate with the subnet

resource "aws\_route\_table" "rt1" {

vpc\_id = aws\_vpc.vpc1.id

route {

cidr\_block = "0.0.0.0/0"

gateway\_id = aws\_internet\_gateway.igw1.id

}

}

resource "aws\_route\_table\_association" "rta1" {

subnet\_id = aws\_subnet.subnet1.id

route\_table\_id = aws\_route\_table.rt1.id

}

# Create Security Group for VPC 1

resource "aws\_security\_group" "sg1" {

name = "vpc1-sg"

vpc\_id = aws\_vpc.vpc1.id

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

}

# Create EC2 Instance in VPC 1

resource "aws\_instance" "instance1" {

ami = "ami-085f9c64a9b75eed5" # Replace with a valid AMI ID

instance\_type = "t2.micro"

subnet\_id = aws\_subnet.subnet1.id

key\_name = "ohio" # Replace with your AWS key pair name

vpc\_security\_group\_ids = [aws\_security\_group.sg1.id]

associate\_public\_ip\_address = true

tags = {

Name = "Instance1"

}

}

# Create VPC 2

resource "aws\_vpc" "vpc2" {

cidr\_block = "10.1.0.0/16"

tags = {

Name = "VPC2"

}

}

# Create Subnet for VPC 2

resource "aws\_subnet" "subnet2" {

vpc\_id = aws\_vpc.vpc2.id

cidr\_block = "10.1.1.0/24"

availability\_zone = "us-east-2b"

map\_public\_ip\_on\_launch = true

tags = {

Name = "Subnet2"

}

}

# Create Internet Gateway for VPC 2

resource "aws\_internet\_gateway" "igw2" {

vpc\_id = aws\_vpc.vpc2.id

tags = {

Name = "IGW2"

}

}

# Create Route Table for VPC 2 and associate with the subnet

resource "aws\_route\_table" "rt2" {

vpc\_id = aws\_vpc.vpc2.id

route {

cidr\_block = "0.0.0.0/0"

gateway\_id = aws\_internet\_gateway.igw2.id

}

}

resource "aws\_route\_table\_association" "rta2" {

subnet\_id = aws\_subnet.subnet2.id

route\_table\_id = aws\_route\_table.rt2.id

}

# Create Security Group for VPC 2

resource "aws\_security\_group" "sg2" {

name = "vpc2-sg"

vpc\_id = aws\_vpc.vpc2.id

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

}

# Create EC2 Instance in VPC 2

resource "aws\_instance" "instance2" {

ami = "ami-085f9c64a9b75eed5" # Replace with a valid AMI ID

instance\_type = "t2.micro"

subnet\_id = aws\_subnet.subnet2.id

key\_name = "ohio" # Replace with your AWS key pair name

vpc\_security\_group\_ids = [aws\_security\_group.sg2.id]

associate\_public\_ip\_address = true

tags = {

Name = "Instance2"

}

}

# Create VPC Peering Connection

resource "aws\_vpc\_peering\_connection" "peer" {

vpc\_id = aws\_vpc.vpc1.id

peer\_vpc\_id = aws\_vpc.vpc2.id

auto\_accept = true

tags = {

Name = "VPC1-to-VPC2-Peering"

}

}

# Update Route Tables to include Peering Connection

resource "aws\_route" "route\_to\_vpc2" {

route\_table\_id = aws\_route\_table.rt1.id

destination\_cidr\_block = aws\_vpc.vpc2.cidr\_block

vpc\_peering\_connection\_id = aws\_vpc\_peering\_connection.peer.id

}

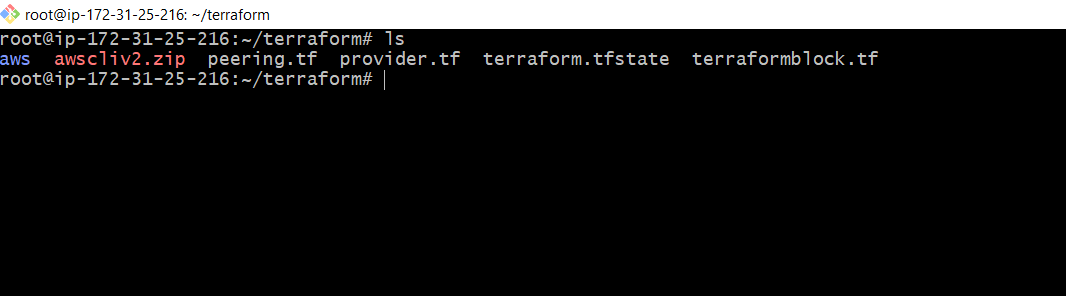
resource "aws\_route" "route\_to\_vpc1" {

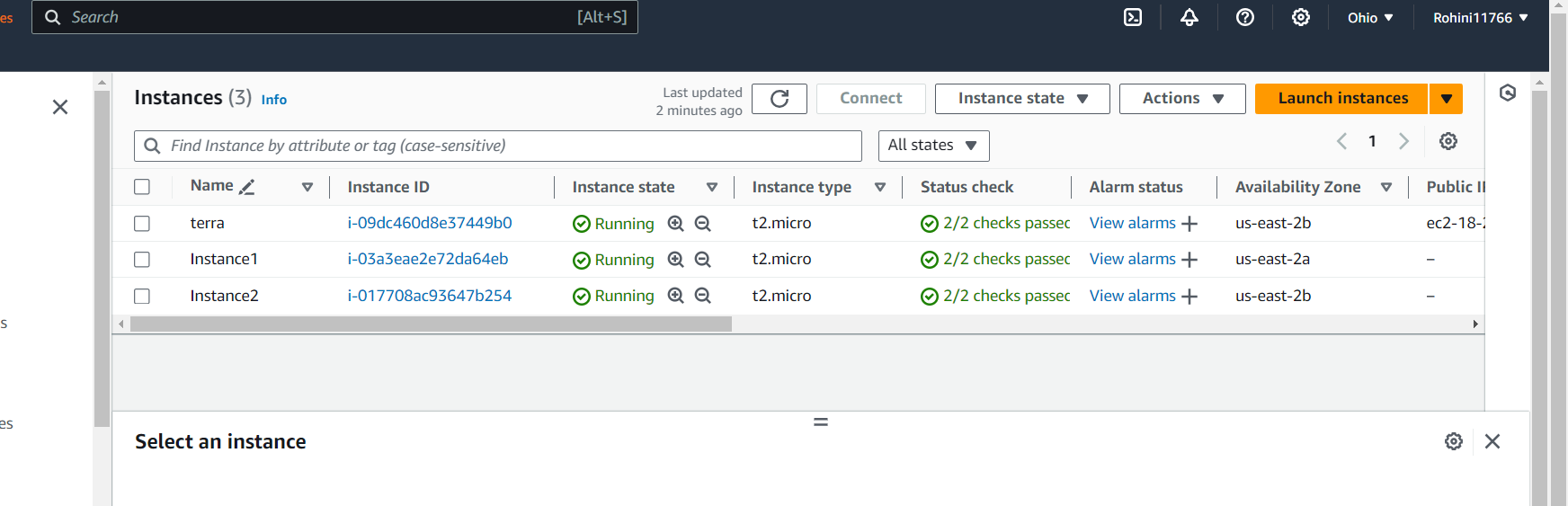
route\_table\_id = aws\_route\_table.rt2.id

destination\_cidr\_block = aws\_vpc.vpc1.cidr\_block

vpc\_peering\_connection\_id = aws\_vpc\_peering\_connection.peer.id

}





Terraform init

Terraform validate

Terraform apply

