Terraform Task:

Create a terraform module for aws ec2 service. a. Able to create key pairs and assign to ec2 instances.

b. Able to get ami id using filter string and use it for ec2.

c. Able to create ec2 instances using asg / standard way..

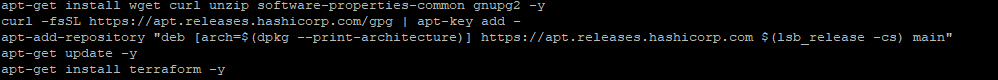
d. Able to create an auto scaling group and load balancer.

e. Able to attach elastic ip to ec2 instance.

Prerequires: 1.terraform

2.awscli

Using below command to install the terraform.



Then check the terraform installed are not with using “terraform -v” command



Then install awscli with using below command



— Now i give versions for terraform 1.2 and aws provider for 5.8.0

— In my main.tf file using different resource like

– aws\_key\_pair,aws\_iam,aws\_instances,aws\_launch\_configuration,etc..

Main.tf –

resource "aws\_key\_pair" "key" {

key\_name = "my-key-pair"

public\_key = file("/root/my-key-pair.pub")

}

data "aws\_ami" "modi\_ami" {

most\_recent = true

filter {

name = "name"

values = ["ubuntu/images/hvm-ssd/ubuntu-bionic-18.04-amd64-server-\*"]

}

filter {

name = "virtualization-type"

values = ["hvm"]

}

owners = var.ami\_owners

}

resource "aws\_instance" "my-instance" {

ami = data.aws\_ami.modi\_ami.id

instance\_type = var.instance\_type

key\_name = aws\_key\_pair.key.key\_name

}

resource "aws\_launch\_configuration" "asg\_launch" {

name = "launch\_config"

image\_id = data.aws\_ami.modi\_ami.id

instance\_type = var.instance\_type

security\_groups = var.security\_groups

key\_name = aws\_key\_pair.key.key\_name

}

resource "aws\_autoscaling\_group" "modi\_asg" {

name = "example\_asg"

min\_size = 1

max\_size = 3

desired\_capacity = 2

launch\_configuration = aws\_launch\_configuration.asg\_launch.name

vpc\_zone\_identifier = ["subnet-0b697dbf1d9956bd4"]

}

resource "aws\_lb" "load\_balancer" {

name = var.load\_balancer\_name

load\_balancer\_type = var.load\_balancer\_type

subnets = var.subnets

}

resource "aws\_lb\_target\_group" "target\_group" {

name = "awstarget"

port = var.target\_group\_port

protocol = var.target\_group\_protocol

vpc\_id = var.vpc\_id

}

resource "aws\_lb\_listener" "listner" {

load\_balancer\_arn = aws\_lb.load\_balancer.arn

port = var.listner\_port

protocol = var.listner\_protocol

default\_action {

type = "forward"

target\_group\_arn = aws\_lb\_target\_group.target\_group.arn

}

}

resource "aws\_eip" "elastic\_ip" {

instance = aws\_instance.my-instance.id

}

output "load\_balancer\_dns" {

value = aws\_lb.load\_balancer.dns\_name

}

Now i am using different variables to create infrastructure some variables

Listed here.

— instance\_type, ami,owners

— ami\_filters, security\_groups

— load\_balancer\_name, load\_balancer\_type

— etc..

Variable.tf –

variable "instance\_type" {

description = "EC2 instance type"

type = string

default = "t2.micro"

}

variable "ami\_owners" {

description = "list of ami owners"

type = list(string)

default = [ "099720109477" ]

}

variable "ami\_filter" {

description = "filter string t find desired AMI"

type = list(string)

default = ["ami-07db5101f3471fab2"]

}

variable "security\_groups" {

description = "list the security groups"

type = set(string)

default = ["sg-02d6acd0dc7850342", "sg-01f0befa6c9eeffcf"]

}

variable "load\_balancer\_name" {

description = "name of the load balanceer"

type = string

default = "modi123"

}

variable "load\_balancer\_type" {

description = "type of the load balancer"

type = string

default = "application"

}

variable "subnets" {

description = "ID of the subnet where the EC2 instance will be deploye"

type = list(string)

default = ["subnet-0b697dbf1d9956bd4", "subnet-0263045bb428921be"]

}

variable "target\_group\_port" {

description = "port for the taget group"

type = number

default = 80

}

variable "target\_group\_protocol" {

description = "protocol for the target group"

type = string

default = "HTTP"

}

variable "vpc\_id" {

description = "ID of the vpc"

type = string

default = "vpc-0783b06692f125be3"

}

variable "listner\_port" {

description = "port for the load balancer listener"

type = number

default = 80

}

variable "listner\_protocol" {

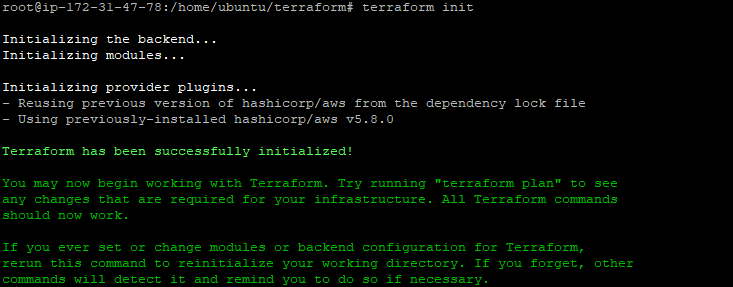
description = "protocol for the load balancer listener"

type = string

default = "HTTP"

}

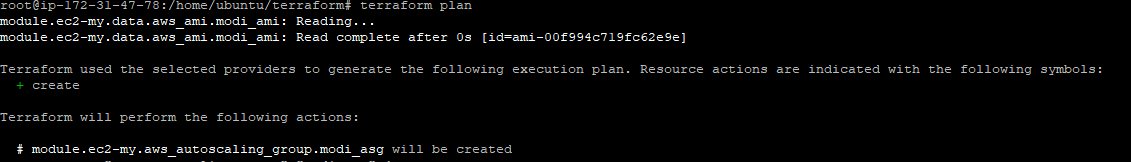
— then using ‘terraform init’ command its setup necessary components and download the provided plugins



– then check the any syntax error with using terraform validate command



– After using the ‘terraform plan’ command it gives the execution plan.



– After the 'terraform apply’ command is used to apply the changes to define your terraform configuration file, create and modify the associated infrastructure resources.

