

PROJECT

Create below infra using terraform

1. Create two virtual machines in east us (web servers)
2. Configure load balancer for above servers

I use this code

- # Configure the AWS provider
- provider "aws" {
- region = "us-east-2"
- }
-
- # Create a VPC
- resource "aws_vpc" "web_vpc" {
- cidr_block = "10.0.0.0/16"
- enable_dns_hostnames = true
-
- tags = {
- Name = "web-vpc"
- }
- }

- # Create an Internet Gateway
- resource "aws_internet_gateway" "web_igw" {
- vpc_id = aws_vpc.web_vpc.id
-
- tags = {
- Name = "web-igw"
- }
- }
-
- # Create a subnet in us-east-2a
- resource "aws_subnet" "web_subnet_1" {
- vpc_id = aws_vpc.web_vpc.id
- cidr_block = "10.0.1.0/24"
- availability_zone = "us-east-1a"
-
- tags = {
- Name = "web-subnet-1"
- }
- }

- # Create a subnet in us-east-2b
- resource "aws_subnet" "web_subnet_2" {
- vpc_id = aws_vpc.web_vpc.id
- cidr_block = "10.0.2.0/24"
- availability_zone = "us-east-1b"
-
- tags = {
- Name = "web-subnet-2"
- }
- }
-
- # Create a route table
- resource "aws_route_table" "web_route_table" {
- vpc_id = aws_vpc.web_vpc.id
-
- route {
- cidr_block = "0.0.0.0/0"
- gateway_id = aws_internet_gateway.web_igw.id
- }
-
- tags = {
- Name = "web-route-table"
- }
- }

- # Associate the route table with subnet 1
- resource "aws_route_table_association" "web_route_assoc_1" {
 - subnet_id = aws_subnet.web_subnet_1.id
 - route_table_id = aws_route_table.web_route_table.id
- }
-
- # Associate the route table with subnet 2
- resource "aws_route_table_association" "web_route_assoc_2" {
 - subnet_id = aws_subnet.web_subnet_2.id
 - route_table_id = aws_route_table.web_route_table.id
- }
-
- # Create a security group for the web servers
- resource "aws_security_group" "web_sg" {
 - name = "web-sg"
 - description = "Security group for web servers"
 - vpc_id = aws_vpc.web_vpc.id

- ingress {
- from_port = 80
- to_port = 80
- protocol = "tcp"
- cidr_blocks = ["0.0.0.0/0"]
- }
-
- ingress {
- from_port = 22
- to_port = 22
- protocol = "tcp"
- cidr_blocks = ["0.0.0.0/0"] # Note: In production, restrict this to your IP
- }
-
- egress {
- from_port = 0
- to_port = 0
- protocol = "-1"
- cidr_blocks = ["0.0.0.0/0"]
- }
-
- tags = {
- Name = "web-sg"
- }
- }

- # Create two EC2 instances (web servers)
- resource "aws_instance" "web_server" {
- count = 2
- ami = ""
- instance_type = "t2.micro"
- key_name = "kumar-tf.pem"
- vpc_security_group_ids = [aws_security_group.web_sg.id]
- subnet_id = count.index == 0 ? aws_subnet.web_subnet_1.id : aws_subnet.web_subnet_2.id
-
- user_data = <<-EOF
- #!/bin/bash
- yum update -y
- yum install -y httpd
- systemctl start httpd
- systemctl enable httpd
- echo "<h1>Hello from \$(hostname -f)</h1>" > /var/www/html/index.html
- EOF
-
- tags = {
- Name = "web-server-\${count.index + 1}"
- }
- }

- # Create an Application Load Balancer
- resource "aws_lb" "web_alb" {
- name = "web-alb"
- internal = false
- load_balancer_type = "application"
- security_groups = [aws_security_group.web_sg.id]
- subnets = [aws_subnet.web_subnet_1.id, aws_subnet.web_subnet_2.id]
-
- tags = {
- Name = "web-alb"
- }
- }
-
- # Create a target group for the ALB
- resource "aws_lb_target_group" "web_tg" {
- name = "web-tg"
- port = 80
- protocol = "HTTP"
- vpc_id = aws_vpc.web_vpc.id
-
- health_check {
- path = "/"
- healthy_threshold = 2
- unhealthy_threshold = 10
- }
- }

- # Attach the EC2 instances to the target group
- resource "aws_lb_target_group_attachment" "web_tg_attachment" {
 - count = 2
 - target_group_arn = aws_lb_target_group.web_tg.arn
 - target_id = aws_instance.web_server[count.index].id
 - port = 80
- }
-
- # Create a listener for the ALB
- resource "aws_lb_listener" "web_listener" {
 - load_balancer_arn = aws_lb.web_alb.arn
 - port = "80"
 - protocol = "HTTP"
-
- default_action {
 - type = "forward"
 - target_group_arn = aws_lb_target_group.web_tg.arn
- }
- }

- # Output the public IPs of the EC2 instances
- output "web_server_public_ips" {
 - value = aws_instance.web_server[*].public_ip
 - }
 -
- # Output the DNS name of the load balancer
- output "alb_dns_name" {
 - value = aws_lb.web_alb.dns_name
 - description = "The DNS name of the Application Load Balancer"
 - }

- I have facing some errors at the terraform apply
- For creating the EC2 Instances
- The error was shone like this
- Error: creating EC2 Instance: operation error EC2: RunInstances, https response error StatusCode: 400, RequestID: 15a6f75e-575b-4561-9766-a80f6eb5b4ff, api error InvalidKeyPair.NotFound: The key pair 'dhoni.pem' does not exist

```
aws_lb.web_alb: Still creating... [1m30s elapsed]
aws_lb.web_alb: Still creating... [1m40s elapsed]
aws_lb.web_alb: Still creating... [1m50s elapsed]
aws_lb.web_alb: Still creating... [2m0s elapsed]
aws_lb.web_alb: Still creating... [2m10s elapsed]
aws_lb.web_alb: Still creating... [2m20s elapsed]
aws_lb.web_alb: Still creating... [2m30s elapsed]
aws_lb.web_alb: Still creating... [2m40s elapsed]
aws_lb.web_alb: Creation complete after 2m41s [id=arn:aws:elasticloadbalancing:us-east-2:891376934605:loadbalancer/app/web-alb/a1e6935c0e3aa2b8]
aws_lb_listener.web_listener: Creating...
aws_lb_listener.web_listener: Creation complete after 0s [id=arn:aws:elasticloadbalancing:us-east-2:891376934605:listener/app/web-alb/a1e6935c0e3aa2b8/563add27e549056c]

Error: creating EC2 Instance: operation error EC2: RunInstances, https response error StatusCode: 400, RequestID: ac462f93-efd0-4f26-ae83-ccd55ac21567, api error InvalidKeyPair.NotFound: The key pair 'dhoni.pem' does not exist

    with aws_instance.web_server[1],
      on main.tf line 106, in resource "aws_instance" "web_server":
   106: resource "aws_instance" "web_server" {

Error: creating EC2 Instance: operation error EC2: RunInstances, https response error StatusCode: 400, RequestID: 1dbbc39b-0758-4bd0-896c-efa84d2e7e52, api error InvalidKeyPair.NotFound: The key pair 'dhoni.pem' does not exist










    with aws_instance.web_server[0],
      on main.tf line 106, in resource "aws_instance" "web_server":
   106: resource "aws_instance" "web_server" {

root@ip-172-31-46-78:~/terraform# ^C
root@ip-172-31-46-78:~/terraform# vi main.tf
root@ip-172-31-46-78:~/terraform# terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
```



- I solved the errors
- I got out put but

I get the instances

Instances (4) Info									
<input type="text" value="Find Instance by attribute or tag (case-sensitive)"/>					All states ▼		< 1 > ⚙		
<input type="checkbox"/>	Name 	Instance ID	Instance state ▼	Instance type ▼	Status check	Alarm status	Availability Zone ▼	Public IP	
<input type="checkbox"/>	dhoni	i-035187fef053b4c89	✔ Running  	t2.micro	✔ 2/2 checks passed	View alarms +	us-east-2c	ec2-3-11	
<input type="checkbox"/>	web-server-2	i-0c60b84830cb2de6c	✔ Running  	t2.micro	⌚ Initializing	View alarms +	us-east-2b	-	
<input type="checkbox"/>	dhoni-web	i-044567609d1f2d829	⊖ Terminated  	t2.micro	-	View alarms +	us-east-2c	-	
<input type="checkbox"/>	web-server-1	i-01c6cd51bf46b6a18	✔ Running  	t2.micro	⌚ Initializing	View alarms +	us-east-2a	-	

Load balancer

The screenshot shows the AWS Management Console interface for the 'Load balancers' page. The top navigation bar includes a search bar, user profile, and region. The left sidebar lists various AWS services. The main content area shows the 'Load balancers' section with a table of existing load balancers. One load balancer, 'web-alb', is listed with a DNS name, active state, VPC ID, and two availability zones. Below the table, a message indicates that no load balancers are currently selected.

Services [Alt+S] Ohio RaviKumar

EC2 > Load balancers

Load balancers (1) Actions Create load balancer

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

< 1 > ⚙

<input type="checkbox"/>	Name	DNS name	State	VPC ID	Availability Zones	Type
<input type="checkbox"/>	web-alb	web-alb-543214551.us-ea...	Active	vpc-0e0df5991585c14...	2 Availability Zones	application

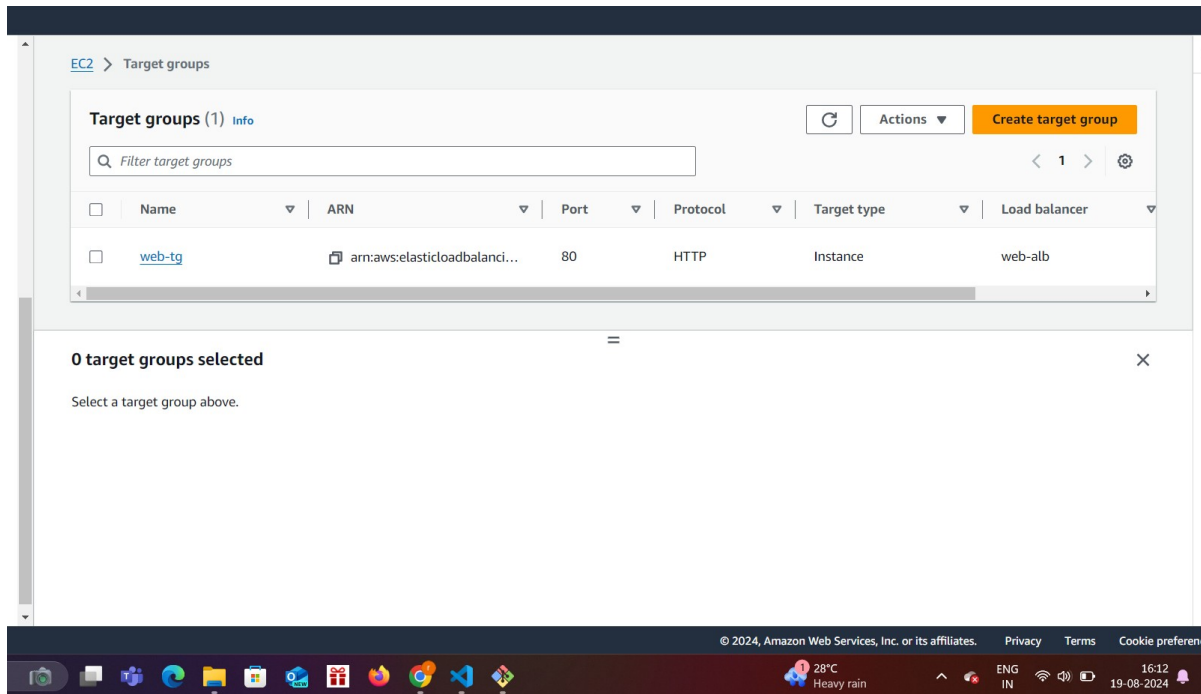
0 load balancers selected

Select a load balancer above.

Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

search 28°C Heavy rain ENG IN 16:12 19-08-2024

Target group



EC2 > Target groups

Target groups (1) [Info](#)

[Create target group](#)

<input type="checkbox"/>	Name	ARN	Port	Protocol	Target type	Load balancer
<input type="checkbox"/>	web-tg	arn:aws:elasticloadbalanci...	80	HTTP	Instance	web-alb

0 target groups selected

Select a target group above.

© 2024, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

28°C Heavy rain 16:12 ENG IN 19-08-2024

VPC

The screenshot displays the AWS Management Console for a VPC. At the top, a summary table provides key details:

Property	Value
Default VPC	No
IPv4 CIDR	10.0.0.0/16
IPv6 pool	-
IPv6 CIDR	-
Network Address Usage metrics	Disabled
Route 53 Resolver DNS Firewall rule groups	-
Owner ID	891376934605

Below the summary, the **Resource map** tab is selected, showing a visual overview of the VPC resources:

- Subnets (2):** Subnets within this VPC. Includes **us-east-2a** (web-subnet-1) and **us-east-2b** (web-subnet-2).
- Route tables (2):** Route network traffic to resources. Includes **web-route-table** (rtb-034b2002c63aa5b89).
- Network connections (1):** Connections to other networks. Includes **web-igw**.

Connections are shown between web-subnet-1 and web-route-table, and between web-subnet-2 and web-route-table. The web-route-table is connected to web-igw.

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

28°C Heavy rain 16:13 19-08-2024 ENG IN

Success to do terraform apply

```
saved the plan to: myplan.tfplan

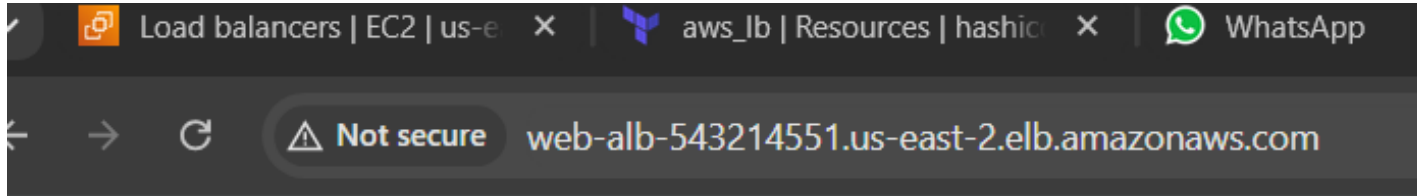
To perform exactly these actions, run the following command to apply:
  terraform apply "myplan.tfplan"
root@ip-172-31-46-78:~/terraform# terraform apply myplan.tfplan
aws_instance.web_server[1]: Creating...
aws_instance.web_server[0]: Creating...
aws_instance.web_server[1]: Still creating... [10s elapsed]
aws_instance.web_server[0]: Still creating... [10s elapsed]
aws_instance.web_server[1]: Still creating... [20s elapsed]
aws_instance.web_server[0]: Still creating... [20s elapsed]
aws_instance.web_server[1]: Still creating... [30s elapsed]
aws_instance.web_server[0]: Still creating... [30s elapsed]
aws_instance.web_server[0]: Creation complete after 32s [id=i-01c6cd51bf46b6a18]
aws_instance.web_server[1]: Creation complete after 32s [id=i-0c60b84830cb2de6c]
aws_lb_target_group_attachment.web_tg_attachment[0]: Creating...
aws_lb_target_group_attachment.web_tg_attachment[1]: Creating...
aws_lb_target_group_attachment.web_tg_attachment[0]: Creation complete after 0s [id=arn:aws:elasticloadbalancing:us-east-2:891376934605:targetgroup/web-tg/5b07fcc070fc99e7-20240819103759875400000003]
aws_lb_target_group_attachment.web_tg_attachment[1]: Creation complete after 0s [id=arn:aws:elasticloadbalancing:us-east-2:891376934605:targetgroup/web-tg/5b07fcc070fc99e7-20240819103759951400000004]

Apply complete! Resources: 4 added, 0 changed, 0 destroyed.

Outputs:
alb_dns_name = "web-alb-543214551.us-east-2.elb.amazonaws.com"
web_server_public_ips = [
  "",
  ""
]
root@ip-172-31-46-78:~/terraform# top
top - 10:44:18 up 41 min,  1 user,  load average: 0.00, 0.02, 0.03
Tasks: 105 total,  1 running, 104 sleeping,  0 stopped,  0 zombie
%Cpu(s):  0.3 us,  0.0 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  957.4 total,  339.4 free,  329.9 used,  445.3 buff/cache
MiB Swap:   0.0 total,   0.0 free,   0.0 used.  627.5 avail Mem
```



Success



- I have successfully created two instancer and load balancer by using terraform