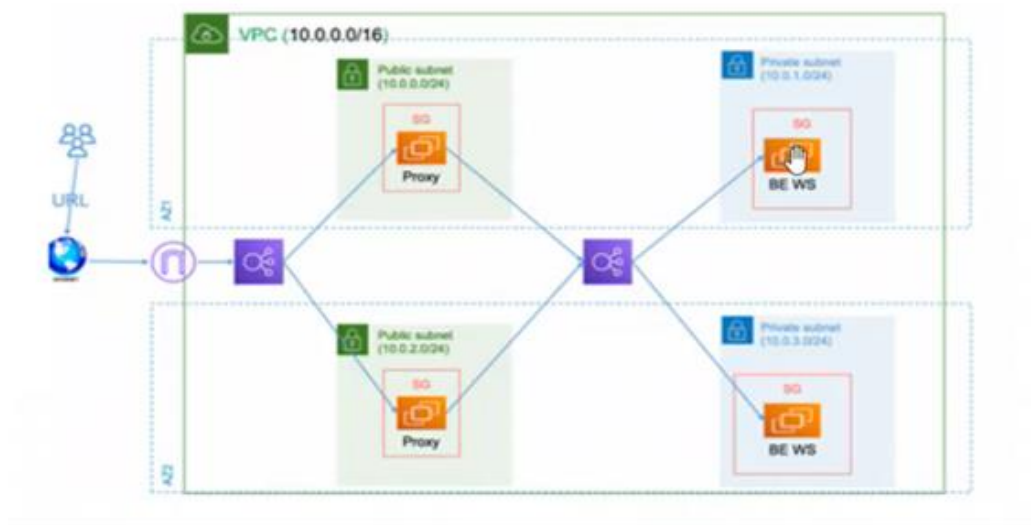


Project Overview

This Terraform project is designed to create an infrastructure with multiple EC2 instances and two load balancers, as shown in the provided diagram. The infrastructure consists of:

1. A Virtual Private Cloud (VPC) with public and private subnets.
2. EC2 instances configured as web servers in the private subnets.
3. Two load balancers:
 - The first one is a public load balancer that forwards traffic to the proxy server.
 - The second one is a private load balancer that forwards traffic to the backend EC2 web servers.

The project uses AWS services, including EC2, Load Balancers and S3 for storing Terraform state files.



Implementation Details

1. **Remote State Configuration:** Configure the backend to store the state file remotely using S3:

```
remote_backend.tf > ...
1 data "aws_s3_bucket" "existing_bucket" {
2   bucket = "my-s3-bucket231"
3 }
4 resource "aws_s3_bucket" "terraform_state" {
5   count = length(data.aws_s3_bucket.existing_bucket.id) == 0 ? 1 : 0
6   bucket = "my-s3-bucket231"
7   force_destroy = true
8   lifecycle {
9     prevent_destroy = true
10  }
11   tags = {
12     Name = "Terraform State Bucket"
13  }
14 }
15 resource "aws_s3_bucket_versioning" "enable" {
16   count = length(data.aws_s3_bucket.existing_bucket.id) == 0 ? 1 : 0
17   bucket = aws_s3_bucket.terraform_state[0].id
18   versioning_configuration {
19     status = "Enabled"
20   }
21 }
22 resource "aws_dynamodb_table" "terraform_locks" {
23   name           = "dynamodb-locks"
24   billing_mode   = "PAY_PER_REQUEST"
25   hash_key       = "LockID"
26
27   attribute {
28     name = "LockID"
29     type = "S"
30   }
31
32   lifecycle {
33     ignore_changes = [name]
34   }
35
36   tags = {
37     Name = "Terraform Lock Table"
38   }
39 }
40
41 terraform {
42   backend "s3" {
43     bucket      = "my-s3-bucket231"
44     key         = "terraform.tfstate"
45     region      = "us-east-1"
46     dynamodb_table = "NM-locks"
47     encrypt     = true
48   }
49 }
50
```

General purpose buckets (1) Info All AWS Regions				
Buckets are containers for data stored in S3.				
<input type="text" value="Find buckets by name"/>				
Name	AWS Region	IAM Access Analyzer	Creation date	
<input type="radio"/> my-s3-bucket231	US East (N. Virginia) us-east-1	View analyzer for us-east-1	October 1, 2024, 23:08:29 (UTC+03:00)	

2. **VPC:** Create a VPC using a custom VPC module.

```
1 resource "aws_vpc" "vpc" {
2   cidr_block = var.vpc_cidr
3   tags = {
4     Name = "VPC"
5   }
6 }
7
8 resource "aws_internet_gateway" "igw" {
9   vpc_id = aws_vpc.vpc.id
10  tags = {
11    Name = "IGW"
12  }
13 }
14
15 resource "aws_eip" "eip" {
16   domain = "vpc"
17 }
18
19 resource "aws_nat_gateway" "nat" {
20   allocation_id = aws_eip.eip.id
21   subnet_id = var.nat_subnet_id
22   tags = {
23     Name = "NAT"
24   }
25   depends_on = [aws_internet_gateway.igw]
26 }
```

```
modules > VPC > outputs.tf > output "vpc_id"
1 output "vpc_id" {
2   value = aws_vpc.vpc.id
3 }
4
5 output "igw_id" {
6   value = aws_internet_gateway.igw.id
7 }
8
9 output "nat_id" {
10  value = aws_nat_gateway.nat.id
11 }
```

```
main.tf ./
main.tf ../VPC
outputs.tf
variables.tf x
modules > VPC > variables.tf > variable "vpc_cidr"
1 variable "vpc_cidr" {
2   description = "VPC CIDR"
3   type = string
4 }
5 variable "nat_subnet_id" {
6   description = "The subnet ID of the public subnet in which to place the gateway"
7   type = string
8 }
```

VPC dashboard

EC2 Global View

Filter by VPC

virtual private cloud

Your VPCs

Subnets

Route tables

Your VPCs (1/4)

Search

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP
custom_vpc	vpc-0232815004252000	Available	10.0.0.0/16	-	dhcp-0
<input checked="" type="checkbox"/> VPC	vpc-00fdd537983d1066a	Available	10.0.0.0/16	-	dhcp-0
custom_vpc	vpc-0f8816f4a220506e5	Available	10.0.0.0/16	-	dhcp-0

vpc-00fdd537983d1066a / VPC

3. EC2 Instances:

- Create EC2 instances using a custom EC2 module.
- Use a data source to get the AMI ID for EC2:
- Create the security group
- Create public, private instance
- Use remote-exec provisioners to install Apache server

```

1 data "aws_ami" "ami_id" {
2   most_recent = true
3   owners      = ["amazon"]
4   filter {
5     name     = "name"
6     values   = ["amzn2-ami-hvm-*x86_64-gp2"]
7   }
8 }
9 resource "aws_security_group" "sg" {
10   vpc_id = var.sg_vpc_id
11   ingress {
12     from_port = 443
13     to_port   = 443
14     protocol  = "tcp"
15     cidr_blocks = ["0.0.0.0/0"]
16   }
17   ingress {
18     from_port = 80
19     to_port   = 80
20     protocol  = "tcp"
21     cidr_blocks = ["0.0.0.0/0"]
22   }
23   ingress {
24     from_port = 22
25     to_port   = 22
26     protocol  = "tcp"
27     cidr_blocks = ["0.0.0.0/0"]
28   }
29   egress {
30     from_port = 0
31     to_port   = 0
32     protocol  = "-1"
33     cidr_blocks = ["0.0.0.0/0"]
34   }
35   tags = {
36     Name = "sg"
37   }
38 }
39
40 resource "aws_instance" "pub-ec2" {
41   count          = length(var.ec2_public_subnet_id)
42   ami            = data.aws_ami.ami_id.id
43   instance_type  = "t2.micro"
44   subnet_id      = var.ec2_public_subnet_id[count.index]
45   security_groups = [aws_security_group.sg.id]
46   associate_public_ip_address = true
47   key_name       = var.key_pair_name
48
49   tags = {
50     Name = "public_ec2_${count.index}"
51   }
52
53   provisioner "remote-exec" {
54     inline = [
55       "set -e",
56       "sleep 10",
57       "sudo yum update -y",
58       "sudo yum install -y httpd",
59       "sudo systemctl start httpd",
60       "sudo systemctl enable httpd",
61       |<<-EOT
62       echo "<html><body><h1>Welcome to Public Ahmed Negm EC2 Instance ${count.index}</h1>
63       </body></html>" | sudo tee /var/www/html/index.html
64     ]
65   }
66   connection {
67     type     = "ssh"
68     host     = self.public_ip
69     user     = "ec2-user"
70     private_key = file("~/Downloads/Nehmkey.pem")
71     timeout  = "5m"
72   }
73 }

```

```

75 | provisioner "local-exec" {
76 |   when      = create
77 |   on_failure = continue
78 |   command = "echo public-ip-${count.index} : ${self.public_ip} >> all-ips.txt"
79 | }
80 |
81 |
82 |
83 | resource "aws_instance" "priv-ec2" {
84 |   count           = length(var.ec2_private_subnet_id)
85 |   ami             = data.aws_ami.ami_id.id
86 |   instance_type   = "t2.micro"
87 |   subnet_id       = var.ec2_private_subnet_id[count.index]
88 |   security_groups = [aws_security_group.sg.id]
89 |   associate_public_ip_address = false
90 |   lifecycle {
91 |     | | create_before_destroy = true
92 |   }
93 |   tags = {
94 |     Name = "private_ec2_${count.index}"
95 |   }
96 |
97 |   user_data = <<EOF
98 |   | | | | | #!/bin/bash
99 |   | | | | | sleep 10
100 |  | | | | | sudo yum update -y
101 |  | | | | | sleep 10
102 |  | | | | | sudo yum install -y httpd
103 |  | | | | | sleep 10
104 |  | | | | | sudo systemctl start httpd
105 |  | | | | | sudo systemctl enable httpd
106 |  | | | | | echo "<html><body><h1>${var.ec2_html[count.index]}</h1>
107 |  | | | | | <p>welcome to Priv${count.index}</p>
108 |  | | | | | </body></html>" | sudo tee /var/www/html/index.html
109 |  | | | | | sudo systemctl restart httpd
110 |  | | | | | EOF
111 |
112 |
113 | provisioner "local-exec" {
114 |   when      = create
115 |   on_failure = continue
116 |   command = "echo private-ip-${count.index} : ${self.private_ip} >> all-ips.txt"
117 | }
118 |
119 |

```

```

modules > ec2 > outputs.tf > output "pub-ips"
1 | output "public_ec2_id" {
2 |   value = aws_instance.pub-ec2[*].id
3 | }
4 | output "private_ec2_id" {
5 |   value = aws_instance.priv-ec2[*].id
6 | }
7 | output "security_group_id" {
8 |   | value = aws_security_group.sg.id
9 | }
10 | output "pub-ips" {
11 |   value = aws_instance.pub-ec2[*].public_ip
12 | }
13 |

```

```
main.tf | variables.tf x
modules > ec2 > variables.tf > variable "key_pair_name" > default
1 variable "sg_vpc_id" {
2   | type = string
3 }
4 variable "priv_lb_dns" {}
5 variable "ec2_public_subnet_id" {
6   | type = list
7 }
8 variable "ec2_private_subnet_id" {
9   | type = list
10 }
11 variable "ec2_html" {
12   | type = list(string)
13   | default = [
14     | "Welcome to Private EC2 Instance 1",
15     | "Welcome to Private EC2 Instance 2"
16   ]
17 }
18 variable "key_pair_name" {
19   | description = "Name of the EC2 Key Pair"
20   | type = string
21   | default = "NetKey"
22 }
23 }
```

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
<input checked="" type="checkbox"/>	private_ec2_0	i-0c130959bf945a528	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-
<input type="checkbox"/>	public_ec2_0	i-0e4ac25571f326454	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-
<input type="checkbox"/>	public_ec2_1	i-05e91c666d669f878	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	-
<input type="checkbox"/>	private_ec2_1	i-0977fcca8853fbb83	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	-

i-0c130959bf945a528 (private_ec2_0)

Details

Status and alarms

Monitoring

Security

Networking

Storage

Tags

▼ Instance summary Info

Instance ID

i-0c130959bf945a528 (private_ec2_0)

IPv6 address

-

Hostname type

IP name: ip-10-0-1-133.ec2.internal

Public IPv4 address

-

Instance state

Running

Private IP DNS name (IPv4 only)

ip-10-0-1-133.ec2.internal

Private IPv4 addresses

10.0.1.133

Public IPv4 DNS

-

4. **Subnets: create public and private subnets using a custom **Subnet** module**
- Define subnets, route tables, and gateways

```
... main.tf x
modules > Subnet > main.tf > resource "aws_subnet" "private_subnets"
1 resource "aws_subnet" "public_subnets" {
2   count           = length(var.pub_subnets)
3   vpc_id          = var.vpc_id
4   cidr_block      = var.pub_subnets[count.index].subnets_cidr
5   availability_zone = var.pub_subnets[count.index].availability_zone
6   tags = {
7     Name = "public_subnet_${count.index}"
8   }
9 }
10 resource "aws_route_table" "public-rt" {
11   vpc_id = var.vpc_id
12   route {
13     cidr_block = "0.0.0.0/0"
14     gateway_id = var.igw_id
15   }
16 }
17 resource "aws_route_table_association" "public-rta" {
18   count           = length(aws_subnet.public_subnets)
19   subnet_id       = aws_subnet.public_subnets[count.index].id
20   route_table_id = aws_route_table.public-rt.id
21 }
22 resource "aws_subnet" "private_subnets" {
23   count           = length(var.priv_subnets)
24   vpc_id          = var.vpc_id
25   cidr_block      = var.priv_subnets[count.index].subnets_cidr
26   availability_zone = var.priv_subnets[count.index].availability_zone
27   tags = {
28     Name = "private_subnet_${count.index}"
29   }
30 }
31
32 resource "aws_route_table" "private-rt" {
33   vpc_id = var.vpc_id
34   route {
35     cidr_block      = "0.0.0.0/0"
36     nat_gateway_id = var.nat_id
37   }
38 }
39
40 resource "aws_route_table_association" "private-rta" {
41   count           = length(aws_subnet.private_subnets)
42   subnet_id       = aws_subnet.private_subnets[count.index].id
43   route_table_id = aws_route_table.private-rt.id
44 }
45
```

```
modules > Subnet > outputs.tf > output "private_subnets_id"
1 output "public_subnets_id" {
2   value = aws_subnet.public_subnets[*].id
3 }
4 output "private_subnets_id" {
5   value = aws_subnet.private_subnets[*].id
6 }
```

Subnets (1/11) Info

Last updated 12 minutes ago

Actions

Create subnet

Find resources by attribute or tag

	Name	Subnet ID	State	VPC	IPv4 CIDR
<input checked="" type="checkbox"/>	private_subnet_1	subnet-0af438e289429742e	Available	vpc-00fdd537983d1066a VPC	10.0.3.0/24
<input type="checkbox"/>	-	subnet-06a6a66acfcf005b3	Available	vpc-0fc466cca59064fa3	172.31.64.0/20
<input type="checkbox"/>	public_subnet_0	subnet-0b7ee889a3abd3e10	Available	vpc-00fdd537983d1066a VPC	10.0.0.0/24

subnet-0af438e289429742e / private_subnet_1

Details | Flow logs | Route table | Network ACL | CIDR reservations | Sharing | Tags

Details

Subnet ID

subnet-0af438e289429742e

Available IPv4 addresses

249

Availability Zone ID

use1-az6

Network ACL

acl-0bcbac14dee3fe7e1

Auto-assign customer-owned IPv4

No

Subnet ARN

arn:aws:ec2:us-east-1:231447665136:subnet/subnet-0af438e289429742e

IPv6 CIDR

-

Network border group

us-east-1

Default subnet

No

State

Available

IPv6 CIDR association ID

-

VPC

vpc-00fdd537983d1066a | VPC

Auto-assign public IPv4 address

No

Outpost ID

-

IPv4 CIDR

10.0.3.0/24

Availability Zone

us-east-1b

Route table

rtb-01d152c5e60b4fd1c

Auto-assign IPv6 address

No

IPv4 CIDR reservations

-

☒ public_subnet_0

subnet-0b7ee889a3abd3e10

Available

vpc-00fdd537983d1066a | VPC

10.0.0.0/24

subnet-0b7ee889a3abd3e10 / public_subnet_0

Details | Flow logs | Route table | Network ACL | CIDR reservations | Sharing | Tags

Details

Subnet ID

subnet-0b7ee889a3abd3e10

Available IPv4 addresses

248

Availability Zone ID

use1-az4

Network ACL

acl-0bcbac14dee3fe7e1

Auto-assign customer-owned IPv4 address

No

Subnet ARN

arn:aws:ec2:us-east-1:231447665136:subnet/subnet-0b7ee889a3abd3e10

IPv6 CIDR

-

Network border group

us-east-1

Default subnet

No

State

Available

IPv6 CIDR association ID

-

VPC

vpc-00fdd537983d1066a | VPC

Auto-assign public IPv4 address

No

Outpost ID

-

IPv4 CIDR

10.0.0.0/24

Availability Zone

us-east-1a

Route table

rtb-0fb398bea1291742f

Auto-assign IPv6 address

No

IPv4 CIDR reservations

-

5. Load Balancers:

- Create a public and private load balancer using a custom module.

```
main.tf x
modules > Load_Balancer > main.tf > resource "aws_lb" "load-balancer"
1 resource "aws_lb_target_group" "tg" {
2   count = 2
3   port = 80
4   protocol = "HTTP"
5   vpc_id = var.lb_vpc_id
6 }
7 resource "aws_lb_target_group_attachment" "public-target-group-attachment" {
8   count = length(var.pub_target_id)
9   target_group_arn = aws_lb_target_group.tg[0].arn
10  target_id = var.pub_target_id[count.index]
11  port = 80
12 }
13 resource "aws_lb_target_group_attachment" "private-target-group-attachment" {
14   count = length(var.priv_target_id)
15   target_group_arn = aws_lb_target_group.tg[1].arn
16   target_id = var.priv_target_id[count.index]
17   port = 80
18 }
19 resource "aws_lb" "load-balancer" {
20   count = 2
21   name = "load-balancer-${count.index}"
22   internal = var.lb_internal[count.index]
23   load_balancer_type = "application"
24   subnets = var.lb_subnets[count.index]
25   security_groups = [var.lb_sg_id]
26 }
27 resource "aws_lb_listener" "lb-listner" {
28   count = 2
29   load_balancer_arn = aws_lb.load-balancer[count.index].id
30   port = "80"
31   protocol = "HTTP"
32   default_action {
33     type = "forward"
34     target_group_arn = aws_lb_target_group.tg[count.index].id
35   }
36 }
37 }
```

```
main.tf x outputs.tf x
modules > Load_Balancer > outputs.tf > output "public_load_balancer_dns"
1 output "public_load_balancer_dns" {
2   value = aws_lb.load-balancer[0].dns_name
3 }
4 output "private_load_balancer_dns" {
5   value = aws_lb.load-balancer[1].dns_name
6 }
7
8
9
```

```
main.tf x variables.tf x
modules > Load_Balancer > variables.tf > variable "lb_vpc_id"
1 variable "lb_vpc_id" {
2   type = string
3 }
4 variable "pub_target_id" {
5   type = list(string)
6 }
7 variable "priv_target_id" {
8   type = list(string)
9 }
10 variable "lb_internal" {
11   type = list(bool)
12 }
13 variable "lb_subnets" {
14   type = list(list(string))
15 }
16 variable "lb_sg_id" {
17 }
```

Load balancers (2)

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"/>	load-balancer-0	load-balancer-0-1494148...	Active	vpc-00fdd537983d10...	2 Availability Zones	application
<input type="checkbox"/>	load-balancer-1	internal-load-balancer-1-2...	Active	vpc-00fdd537983d10...	2 Availability Zones	application

The final Result:

```
main.tf  outputs.tf  remote_backend.tf  all-ips.txt  variables.tf
all-ips.txt
1 private-ip-1 : 10.0.3.131
2 private-ip-0 : 10.0.1.133
3 public-ip-1 : 3.85.130.1
4 public-ip-0 : 34.229.58.194
5

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS
module.LB.aws_lb.load-balancer[1]: Still creating... [2m10s elapsed]
module.LB.aws_lb.load-balancer[0]: Still creating... [2m10s elapsed]
module.LB.aws_lb.load-balancer[0]: Still creating... [2m20s elapsed]
module.LB.aws_lb.load-balancer[1]: Still creating... [2m20s elapsed]
module.LB.aws_lb.load-balancer[0]: Still creating... [2m30s elapsed]
module.LB.aws_lb.load-balancer[1]: Still creating... [2m30s elapsed]
module.LB.aws_lb.load-balancer[0]: Still creating... [2m40s elapsed]
module.LB.aws_lb.load-balancer[1]: Still creating... [2m40s elapsed]
module.LB.aws_lb.load-balancer[0]: Still creating... [2m50s elapsed]
module.LB.aws_lb.load-balancer[1]: Still creating... [2m50s elapsed]
module.LB.aws_lb.load-balancer[0]: Still creating... [3m0s elapsed]
module.LB.aws_lb.load-balancer[1]: Still creating... [3m0s elapsed]
module.LB.aws_lb.load-balancer[0]: Still creating... [3m10s elapsed]
module.LB.aws_lb.load-balancer[1]: Still creating... [3m10s elapsed]
module.LB.aws_lb.load-balancer[0]: Still creating... [3m20s elapsed]
module.LB.aws_lb.load-balancer[1]: Still creating... [3m20s elapsed]
module.LB.aws_lb.load-balancer[0]: Still creating... [3m30s elapsed]
module.LB.aws_lb.load-balancer[1]: Still creating... [3m30s elapsed]
module.LB.aws_lb.load-balancer[0]: Creation complete after 3m31s [id=arn:aws:elasticloadbalancing:us-east-1:231447665136:loadbalancer/app/load-balancer-0-1494148765-us-east-1-elb:82a66f1]
module.LB.aws_lb.load-balancer[1]: Creation complete after 3m33s [id=arn:aws:elasticloadbalancing:us-east-1:231447665136:loadbalancer/app/load-balancer-0-1494148765-us-east-1-elb:f870830]
module.LB.aws_lb.listener.lb-listener[1]: Creating...
module.LB.aws_lb.listener.lb-listener[0]: Creating...
module.LB.aws_lb.listener.lb-listener[0]: Creation complete after 2s [id=arn:aws:elasticloadbalancing:us-east-1:231447665136:listener/app/load-balancer-0-1494148765-us-east-1-elb:2a66f1/fb89b5aef3982896]
module.LB.aws_lb.listener.lb-listener[1]: Creation complete after 2s [id=arn:aws:elasticloadbalancing:us-east-1:231447665136:listener/app/load-balancer-0-1494148765-us-east-1-elb:870830/7a4cea23561a56e4]
Apply complete! Resources: 30 added, 0 changed, 0 destroyed.
```

← ↻ ⚠ Not secure | 3.85.130.1

Welcome to Public Ahmed Negm EC2 Instance 1

← ↻ ⚠ Not secure | 34.229.58.194

Welcome to Public Ahmed Negm EC2 Instance 0

← ↻ ⚠ Not secure | load-balancer-0-1494148765.us-east-1.elb.amazonaws.com

Welcome to Public Ahmed Negm EC2 Instance 0

← ↻ ⚠ Not secure | load-balancer-0-1494148765.us-east-1.elb.amazonaws.com

Welcome to Public Ahmed Negm EC2 Instance 1