Terraform Mini Project2

[3Tier-Structure-Design]

ELB(ALB|NLB) - ASG(EC2 x 2) - DB(클러스터)

김 유 진

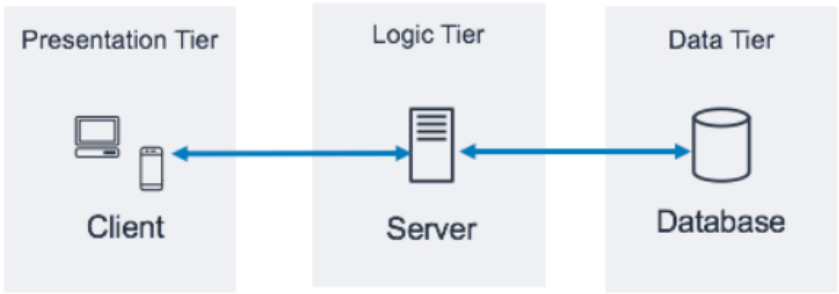
목차

1. 개요 2
1-1. 3-Tier 아키텍처 소개 2
1-2. 아키텍처 구성 3
2. 아키텍처 구현 4
3. 생성 확인 37

1. 3-Tier 아키텍처

3-Tier 아키텍처란?

어떠한 플랫폼이나 애플리케이션을 3계층으로 나누어 별도의 논리적/물리적인 장치에 구축 및 운영하는 형태이다. 통 프레젠테이션 계층, 어플리케이션 계층, 데이터 계층으로 나눈다.

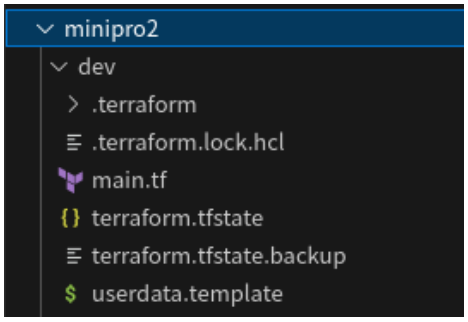


- 프젠테이션 계층
  - 사용자가 애플리케이션과 상호작용하는 인터페이스
  - 일반적으로 HTML, JS, CSS 등이 이 계층에 포함되며, 프론트엔드라고 불린다.
- 애플리케이션 계층
  - 요청되는 정보를 어떠한 규칙에 따라 처리하고 가공한다.
  - 백엔드로 불린다.
- 데이터 계층
  데이터 베이스와 데이터 베이스에 접근하여 데이터를 CRUD 한다.

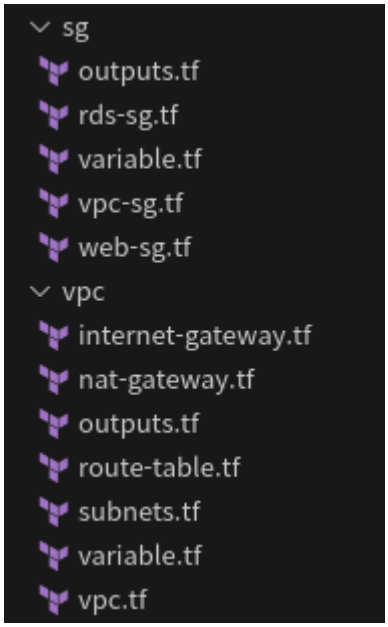
3-Tier 아키텍처 구성

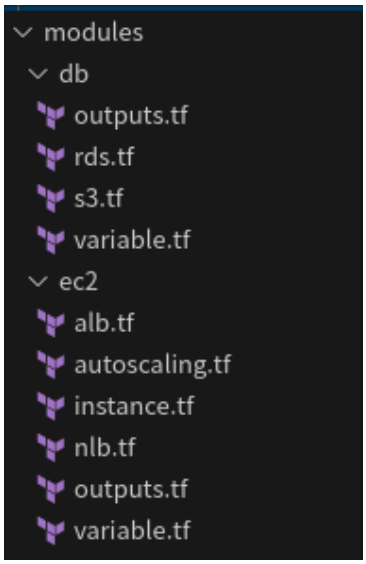
폴더 구조

- **dev/** : Terraform 구성이 정의된 파일

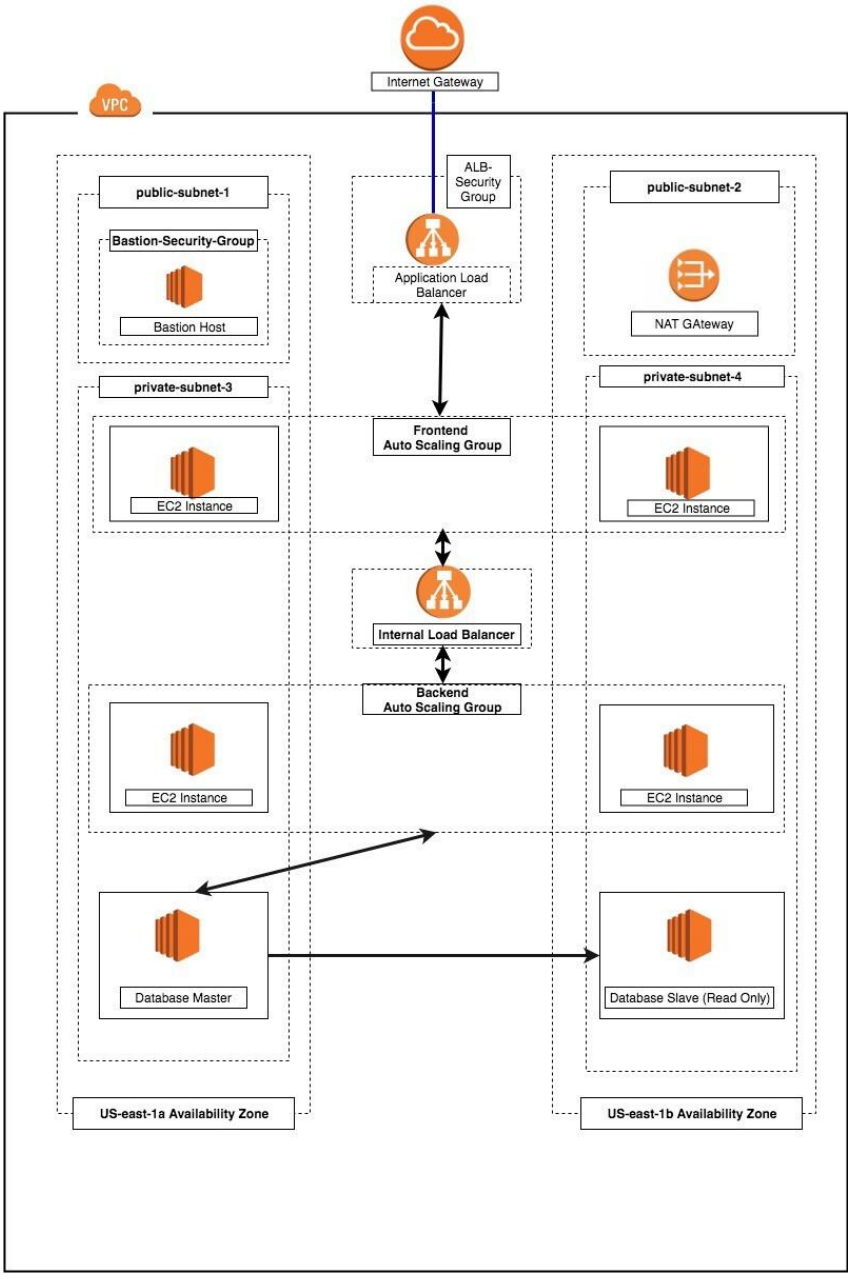


- **modules/** : 리소스 집합





2. 구현



[sg] 보안 그룹

* **Source**: 출발지, cidr_blocks 또는 Security Group

* **Port**: Source에서 들어오는 Port Number

- vpc-security-group

VPC SG	Type	Protocol	Port	Source
INGRESS	SSH	TCP	22	내 IP
	HTTP	TCP	80	내 IP
	ICMP	ICMP	-1	내 IP

- web-security-group

WEB SG	Type	Protocol	Port	Source
INGRESS	SSH	TCP	22	VPC SG
	HTTP	TCP	80	0.0.0.0/0

- was-security-group

WEB SG	Type	Protocol	Port	Source
INGRESS	SSH	TCP	22	VPC SG
	Custom TCP	TCP	8080	10.0.11.0/24 10.0.12.0/24

- rds-security-group

RDS SG	Type	Protocol	Port	Source
INGRESS	MYSQL	TCP	3306	WAS SG
	Custom TCP	TCP	8080	LB SG

- variable.tf

```
variable "vpc-id" {

  description = "VPC ID"

  type        = string

}
```

```
variable "vpc-sg-id" {
   description = "VPC Security Group ID"
   type        = string
}
```

- outputs.tf

```
output "vpc_security_group_id" {
   description = "VPC Security Group ID"
   value = aws_security_group.vpc_security_group.id
}

output "web_security_group_id" {
   description = "WEB Security Group ID"
   value = aws_security_group.web_security_group.id
}
output "was_security_group_id" {
   description = "WAS Security Group ID"
   value = aws_security_group.was_security_group.id
}

output "rds_security_group_id" {
   description = "RDS Security Group ID"
   value = aws_security_group.rds_security_group.id
}
```

```
#####################################
#     VPC Security Group
#####################################
resource "aws_security_group" "vpc_security_group" {
  name            = "vpc_security_group"
  description = "Allow SSH/HTTP/ICMP inbound traffic"
  vpc_id          = var.vpc-id

  ingress {
    description         = "Allow SSH from VPC"
    from_port           = 22
    to_port             = 22
    protocol            = "tcp"
    cidr_blocks         = ["0.0.0.0/0"]
  }

  ingress {
    description         = "Allow HTTP from VPC"
    from_port           = 80
    to_port             = 80
    protocol            = "tcp"
    cidr_blocks         = ["0.0.0.0/0"]
  }
```

```
  ingress {
    description      = "Allow ICMP from VPC"
    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 = "vpc_security_group"
  }
}

#####################################
#    WEB Security Group
#####################################
resource "aws_security_group" "web_security_group" {
```

```
name            = "web_security_group"
description = "Allow SSH/HTTP inbound traffic"
vpc_id          = var.vpc-id


ingress {
   description          = "Allow SSH from VPC"
   from_port             = 22
   to_port               = 22
   protocol              = "tcp"
   security_groups = [var.vpc-sg-id]
}


ingress {
   description          = "Allow HTTP from VPC"
   from_port             = 80
   to_port               = 80
   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"]
  }

  tags = {
    Name = "web_security_group"
  }
}



####################################
#    WAS Security Group
####################################
resource "aws_security_group" "was_security_group" {
  name            = "was_security_group"
  description = "Allow SSH inbound traffic"
  vpc_id         = var.vpc-id

  ingress {
    description      = "Allow SSH from VPC"
    from_port        = 22
    to_port          = 22
    protocol         = "tcp"
    security_groups = [var.vpc-sg-id]
  }
```

```
  ingress {
    from_port     = 8080
    to_port       = 8080
    protocol      = "tcp"
    cidr_blocks = ["10.0.11.0/24", "10.0.12.0/24"]
  }

  egress {
    from_port            = 0
    to_port              = 0
    protocol             = "-1"
    cidr_blocks          = ["0.0.0.0/0"]
  }

  tags = {
    Name = "was_security_group"
  }
}
#####################################
#     RDS Security Group
#####################################
resource "aws_security_group" "rds_security_group" {
  name             = "rds_security_group"
```

```
  description = "RDS Security Group"
  vpc_id          = var.vpc-id

  ingress {
    from_port     = 3306
    to_port       = 3306
    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"]
  }

  tags = {
    Name = "rds_security_group"
  }
}
```

[vpc]

- variable.tf

```
variable "vpc-id" {
```

```
  description = "VPC ID"
  type        = string
}
variable "pub-sub1-id" {
  description = "Public Subnet1 ID"
  type        = string
}
variable "pub-sub2-id" {
  description = "Public Subnet2 ID"
  type        = string
}

variable "pri-sub1-id" {
  description = "Private Subnet1 ID"
  type        = string
}
variable "pri-sub2-id" {
  description = "Private Subnet2 ID"
  type        = string
}
variable "pub-rt-id" {
  description = "Public Route Table ID"
  type        = string
}
```

```
variable "pri-rt-id" {
   description = "Private Route Table ID"
   type        = string
}


variable "igw-id" {
   description = "Internet Gateway ID"
   type        = string
}
variable "ngw-id" {
   description = "Nat Gateway ID"
   type        = string
}
variable "eip-id" {
   description = "Elastic ID"
   type        = string
}
```

- outputs.tf

```
output "vpc_id" {
   description = "VPC ID"
   value = aws_vpc.vpc.id
}
output "public_subnet1_id" {
   description = "Public Subnet1 ID"
```

```
  value = aws_subnet.public_subnet1.id
}
output "public_subnet2_id" {
  description = "Public Subnet2 ID"
  value = aws_subnet.public_subnet2.id
}
output "private_subnet1_id" {
  description = "Private Subnet1 ID"
  value = aws_subnet.private_subnet1.id
}
output "private_subnet2_id" {
  description = "Private Subnet2 ID"
  value = aws_subnet.private_subnet2.id
}
output "public_route_table_id" {
  description = "Public Route Table ID"
  value = aws_route_table.public_route_table.id
}
output "private_route_table_id" {
  description = "Private Route Table ID"
  value = aws_route_table.private_route_table.id
}

output "internet_gateway_id" {
```

```
  description = "Internet Gateway ID"
  value = aws_internet_gateway.internet_gateway.id
}
output "nat_gateway_id" {
  description = "Nat Gateway ID"
  value = aws_nat_gateway.nat_gateway.id
}
output "elastic_ip_id" {
  description = "Elastic IP ID"
  value = aws_eip.elastic_ip.id
}
```

- vpc (aws_vpc)

```
resource "aws_vpc" "vpc" {
  cidr_block           = "10.0.0.0/16"
  instance_tenancy = "default"
  enable_dns_hostnames = true

  tags = {
    Name = "vpc"
  }
}
```

- 서브넷 (aws_subnet)

Public -> public-subnet1, public-subnet2

Private -> private -subnet1, private-subnet2

```
#####################################
#     Public Subnet
#####################################
resource "aws_subnet" "public_subnet1" {
   vpc_id        = var.vpc-id
   cidr_block = "10.0.1.0/24"
   availability_zone = "us-east-2a"

   tags = {
      Name = "public_subnet1"
   }
}
resource "aws_subnet" "public_subnet2" {
   vpc_id        = var.vpc-id
   cidr_block = "10.0.2.0/24"
   availability_zone = "us-east-2b"
   map_public_ip_on_launch = true

   tags = {
      Name = "public_subnet2"
   }
}

#####################################
```

```
#     Private Subnet
#####################################
resource "aws_subnet" "private_subnet1" {
   vpc_id        = var.vpc-id
   cidr_block = "10.0.11.0/24"
   availability_zone = "us-east-2a"

   tags = {
      Name = "private_subnet1"
   }
}
resource "aws_subnet" "private_subnet2" {
   vpc_id        = var.vpc-id
   cidr_block = "10.0.12.0/24"
   availability_zone = "us-east-2b"

   tags = {
      Name = "private_subnet2"
   }
}
```

- 인터넷 게이트웨이 (aws_internet_gateway)

```
resource "aws_internet_gateway" "internet_gateway" {
   vpc_id = var.vpc-id
   tags = {
```

```
    Name = "internet-gateway"
  }
}
```

- 네트워크 게이트웨이 (aws_nat_gateway)

```
#####################################
#     Elastic IPs
#####################################
resource "aws_eip" "elastic_ip" {
  # instance = aws_instance.web.id
  domain     = "vpc"
}

#####################################
#     Nat Gateway
#####################################
resource "aws_nat_gateway" "nat_gateway" {
  allocation_id = var.eip-id
  subnet_id         = var.pub-sub1-id
  tags = {
    Name = "nat_gateway"
  }
}
```

- 라우팅 테이블 (aws_route_table)

라우팅 테이블은 ip 주소에 대한 라우팅 경로를 설정한다. 이는 Public과 Private을 나눠 설정하고

각각 Public, Private 서브넷에 연결한다. 외부에서 내부로의 접근이 가능해야 하므로 Public 서브넷만 인터넷 게이트와 연결한다.

public-route-table, private-route-table

```
####################################
#    Public Route Table
####################################
resource "aws_route_table" "public_route_table" {
  vpc_id = var.vpc-id


  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = var.igw-id
  }


  tags = {
    Name = "public_route_table"
  }
}


####################################
#    Association
####################################
resource "aws_route_table_association" "public-rt-association1" {
  subnet_id      = var.pub-sub2-id
  route_table_id = var.pub-rt-id
```

```
}
resource "aws_route_table_association" "public-rt-association2" {
  subnet_id      = var.pub-sub1-id
  route_table_id = var.pub-rt-id
}

####################################
#    Private Route Table
####################################
resource "aws_route_table" "private_route_table" {
  vpc_id = var.vpc-id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = var.ngw-id
  }

  tags = {
    Name = "private_route_table"
  }
}

####################################
#    Association
####################################
```

```
resource "aws_route_table_association" "private-rt-association1" {
   subnet_id      = var.pri-sub1-id
   route_table_id = var.pri-rt-id
}
resource "aws_route_table_association" "private-rt-association2" {
   subnet_id      = var.pri-sub2-id
   route_table_id = var.pri-rt-id
}
```

[ec2]

- variable.tf

```
variable "vpc-sg-id" {
   description = "VPC Security Group ID"
   type = string
}
variable "web-sg-id" {
   description = "WEB Security Group ID"
   type = string
}

variable "was-sg-id" {
   description = "WAS Security Group ID"
   type = string
}
```

```
variable "vpc-id" {
   description = "VPC ID"
   type = string
}

variable "pub-sub1-id" {
   description = "Public Subnet1 ID"
   type = string
}
variable "pub-sub2-id" {
   description = "Public Subnet2 ID"
   type = string
}

variable "pri-sub1-id" {
   description = "private Subnet1 ID"
   type = string
}
variable "pri-sub2-id" {
   description = "private Subnet2 ID"
   type = string
}
```

- outputs.tf

```
output "bastion_instance_id" {
  description = "Bastion Instance ID"
  value = aws_instance.bastion_instance.id
}
```

- 오토 스케일링 (aws_autoscaling_group)

Bastion_instance, web_instance * 2, was_instance * 2

```
data "aws_ami" "ubuntu" {
  most_recent = true

  filter {
    name   = "name"
    values = ["ubuntu/images/hvm-ssd/ubuntu-jammy-22.04-amd64-server-*"]
  }

    filter {
    name   = "virtualization-type"
    values = ["hvm"]
  }

  owners = ["099720109477"] # Canonical
}

####################################
#    SSH Key
```

```
####################################

resource "aws_key_pair" "deployer" {

key_name = "deployer-key"

public_key = file("~/.ssh/testkey.pub")

}


####################################

#     WEB Launch Configuration

####################################

resource "aws_launch_configuration" "web-configuration" {

  name                 = "web_config"

  image_id          = data.aws_ami.ubuntu.id

  instance_type = "t2.micro"


  user_data = <<-EOF

                #!/bin/bash

                echo "ssh-rsa" >> /home/.ssh/testkey.pub

                EOF


  security_groups = [var.web-sg-id]


  lifecycle {

    create_before_destroy = true

  }
```

```
}


####################################
#     WAS Launch Configuration
####################################
resource "aws_launch_configuration" "was-configuration" {
   name                 = "was_config"
   image_id          = data.aws_ami.ubuntu.id
   instance_type = "t2.micro"


   user_data = file("userdata.template")
   security_groups = [var.was-sg-id]


   lifecycle {
      create_before_destroy = true
   }
}


####################################
#     Bastion Instance
####################################
resource "aws_instance" "bastion_instance"{
   ami = data.aws_ami.ubuntu.id
   instance_type = "t2.micro"
```

```
  subnet_id = var.pub-sub1-id
  associate_public_ip_address = "true"

  key_name = aws_key_pair.deployer.key_name
  vpc_security_group_ids = [var.vpc-sg-id]
  tags = { Name = "bastion_instance" }
}
####################################
#     WEB Autoscaling
####################################
resource "aws_autoscaling_group" "web-autoscaling" {
  name                        = "web-autoscaling"
  max_size                    = 4
  min_size                    = 2
  desired_capacity            = 2

  health_check_grace_period = 300
  health_check_type           = "ELB"
  force_delete                = true

  launch_configuration        = aws_launch_configuration.web-configuration.name
  vpc_zone_identifier         = [var.pub-sub1-id, var.pub-sub2-id]
  target_group_arns = [aws_lb_target_group.alb-target-group.arn]
```

```
  tag {
    key                 = "Name"
    value               = "web-autoscaling"
    propagate_at_launch = false
  }
  lifecycle {
    create_before_destroy = true
  }
}

#####################################
#     WAS Autoscaling
#####################################
resource "aws_autoscaling_group" "was-autoscaling" {
  name                      = "was-autoscaling"
  max_size                  = 4
  min_size                  = 2
  desired_capacity          = 2

  health_check_grace_period = 300
  health_check_type         = "ELB"
  force_delete              = true

  launch_configuration      = aws_launch_configuration.was-configuration.name
```

```
  vpc_zone_identifier       = [var.pri-sub1-id, var.pri-sub2-id]
  target_group_arns = [aws_lb_target_group.nlb-target-group.arn]

  tag {
    key                 = "Name"
    value               = "was-autoscaling"
    propagate_at_launch = false
  }
  lifecycle {
    create_before_destroy = true
  }
}
```

- 로드벨런서 (aws_lb)

alb → HTTP 및 HTTPS 트래픽 로드밸런싱

인터넷 게이트웨이로 들어온 트래픽을 WEB으로 분산시킨다. 가용영역을 public 서브넷으로 설정하고 vpc 보안그룹을 사용한다. HTTP 프로토콜의 80포트를 타고 타겟을 찾아가도록 설정하였으며, 타겟 그룹은 private 서브넷으로 설정한다.

Public -> application-load-balancing

내부 WEB으로 들어온 트래픽을 WAS로 분산시킨다. 가용영역은 private 서브넷(web instance)로 설정하고, web 보안그룹 사용을 타겟 그룹은 private 서브넷(was instance)으로 설정한다.

Private -> network-load-balancing

internal 외부에서 접근이 가능한지 여부

```
#####################################
#     Application Load Balancing
#####################################
```

```
resource "aws_lb" "application-load-balancing" {
  name               = "application-load-balancing"
  internal           = false
  load_balancer_type = "application"
  security_groups    = [var.web-sg-id]
  subnets            = [var.pub-sub1-id, var.pub-sub2-id]

  tags = {
    Name = "application-load-balancing"
  }
}

####################################
#     Load Balancing Listener
####################################
resource "aws_lb_listener" "alb-listener" {
  load_balancer_arn = aws_lb.application-load-balancing.arn
  port              = "80"
  protocol          = "HTTP"

  default_action {
    type = "fixed-response"

    fixed_response {
```

```
      content_type = "text/plain"
      message_body = "404: page not found"
      status_code  = 404
    }
  }
}

#####################################
#    Load Balancing Target Group
#####################################
resource "aws_lb_target_group" "alb-target-group" {
  name        = "alb-target-group"
  port        = 80
  protocol    = "HTTP"
  vpc_id      = var.vpc-id

  health_check {
    path                = "/"
    protocol            = "HTTP"
    matcher             = "200"
    interval            = 15
    timeout             = 3
    healthy_threshold   = 2
    unhealthy_threshold = 2
```

```
  }
}

#####################################
#     Load Balancing Rule
#####################################
resource "aws_lb_listener_rule" "alb-rule" {
  listener_arn = aws_lb_listener.alb-listener.arn
  priority     = 100

  action {
    type = "forward"
    target_group_arn = aws_lb_target_group.alb-target-group.arn
  }

  condition {
    path_pattern {
      values = ["*"]
    }
  }
}
#####################################
#     Network Load Balancing
#####################################
```

```
resource "aws_lb" "network-load-balancing" {
  name                = "network-load-balancing"
  internal            = true
  load_balancer_type = "network"
  security_groups     = [var.was-sg-id]
  subnets             = [var.pri-sub1-id, var.pri-sub2-id]

  tags = {
    Name = "network-load-balancing"
  }
}

####################################
#     Load Balancing Listener
####################################
resource "aws_lb_listener" "nlb-listener" {
  load_balancer_arn = aws_lb.network-load-balancing.arn
  port              = "8080"
  protocol          = "TCP"

  default_action{
    type = "forward"
    target_group_arn = aws_lb_target_group.nlb-target-group.arn
  }
```

```
}


####################################
#     Load Balancing Target Group
####################################
resource "aws_lb_target_group" "nlb-target-group" {
   name        = "nlb-target-group"
   port        = 8080
   protocol = "TCP"
   vpc_id     = var.vpc-id


   health_check {
      path                          = "/"
      protocol                    = "HTTP"
      matcher                     = "200"
      interval                    = 15
      timeout                     = 3
      healthy_threshold     = 2
      unhealthy_threshold = 2
   }
}
```

[db]

```
- variable.tf
variable "my-bucket-id" {
```

```
  description = "S3 Bucket ID"
  type = string
}


variable "vpc-id" {
  description = "VPC ID"
  type        = string
}
variable "rds-sg-id" {
  description = "RDS Security Group ID"
  type        = string
}
variable "pri-sub1-id" {
  description = "Private Subnet1 ID"
  type        = string
}
variable "pri-sub2-id" {
  description = "Private Subnet2 ID"
  type        = string
}
```

- outputs.tf

```
output "my_bucket_id" {
  description = "S3 Bucket ID"
  value = aws_s3_bucket.my-bucket.id
```

```
}
```

- db 서브넷 그룹 (aws_db_subnet_group)

```
resource "aws_db_subnet_group" "db-subnet-group" {
  name       = "main"
  subnet_ids = [var.pri-sub1-id, var.pri-sub2-id]

  tags = {
    Name = "My DB subnet group"
  }
}
```

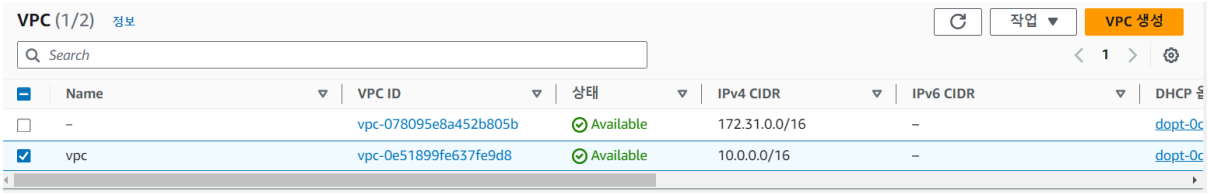
- db 클러스터 (aws_rds_cluster)

```
####################################
#    RDS Cluster
####################################
resource "aws_rds_cluster" "rds-cluster" {
  cluster_identifier      = "rds-cluster"
  engine                  = "aurora-mysql"
  engine_version          = "5.7.mysql_aurora.2.07.9"
  availability_zones      = ["us-east-2a", "us-east-2b"]
  db_subnet_group_name = aws_db_subnet_group.db-subnet-group.name
  vpc_security_group_ids = [var.rds-sg-id]
  master_username         = "tf"
  master_password         = "soldesk1."
  skip_final_snapshot = true
```

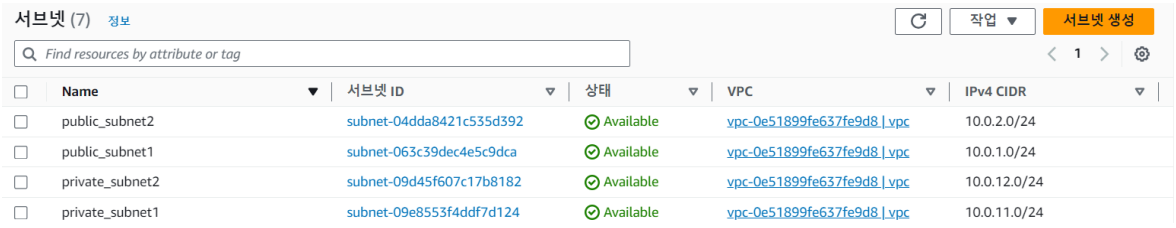
```
}

#####################################

#     RDS Instance

#####################################

resource "aws_rds_cluster_instance" "rds-instance" {

   count                  = 2

   identifier             = "rds-cluster-${count.index}"

   cluster_identifier = aws_rds_cluster.rds-cluster.id

   instance_class       = "db.t3.medium"

   engine                 = aws_rds_cluster.rds-cluster.engine

   engine_version       = aws_rds_cluster.rds-cluster.engine_version

}
```

3. 생성 확인

- VPC



- 서브넷



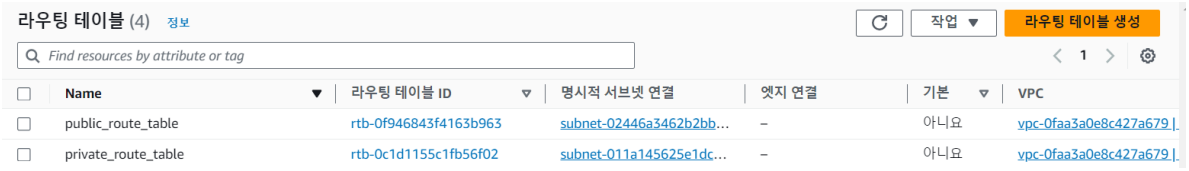
서브넷 (7) 정보

Find resources by attribute or tag

작업 | 서브넷 생성

Name	서브넷 ID	상태	VPC	IPv4 CIDR
public_subnet2	subnet-04dda8421c535d392	Available	vpc-0e51899fe637fe9d8 | vpc	10.0.2.0/24
public_subnet1	subnet-063c39dec4e5c9dca	Available	vpc-0e51899fe637fe9d8 | vpc	10.0.1.0/24
private_subnet2	subnet-09d45f607c17b8182	Available	vpc-0e51899fe637fe9d8 | vpc	10.0.12.0/24
private_subnet1	subnet-09e8553f4ddf7d124	Available	vpc-0e51899fe637fe9d8 | vpc	10.0.11.0/24

- 라우팅 테이블



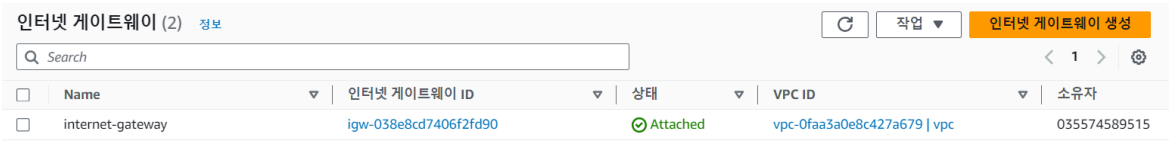
라우팅 테이블 (4) 정보

Find resources by attribute or tag

작업 | 라우팅 테이블 생성

Name	라우팅 테이블 ID	명시적 서브넷 연결	엣지 연결	기본	VPC
public_route_table	rtb-0f946843f4163b963	subnet-02446a3462b2bb...	–	아니요	vpc-0faa3a0e8c427a679 |
private_route_table	rtb-0c1d1155c1fb56f02	subnet-011a145625e1dc...	–	아니요	vpc-0faa3a0e8c427a679 |

- 인터넷 게이트웨이



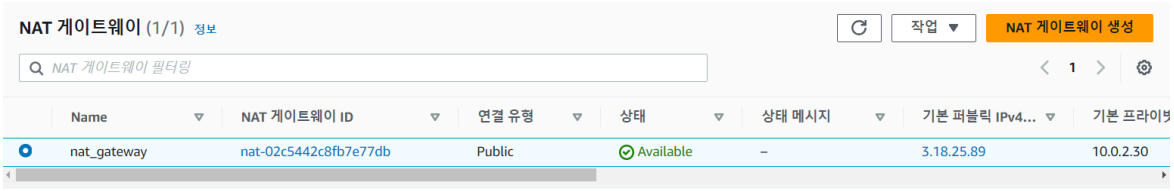
인터넷 게이트웨이 (2) 정보

Search

작업 | 인터넷 게이트웨이 생성

Name	인터넷 게이트웨이 ID	상태	VPC ID	소유자
internet-gateway	igw-038e8cd7406f2fd90	Attached	vpc-0faa3a0e8c427a679 | vpc	035574589515

- NAT 게이트웨이



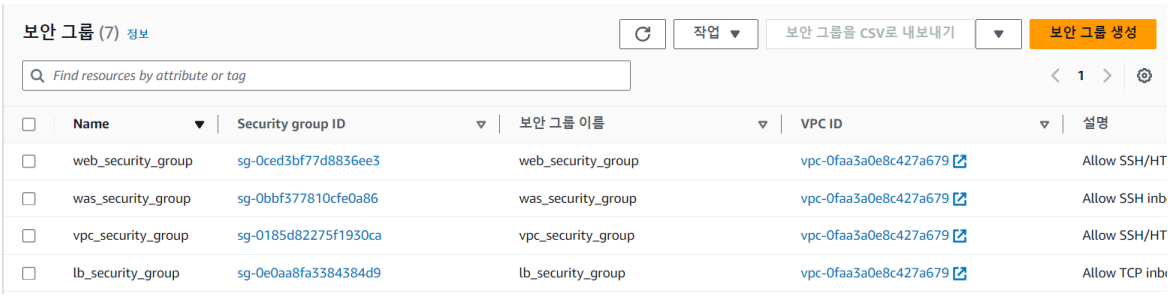
NAT 게이트웨이 (1/1) 정보

NAT 게이트웨이 필터링

작업 | NAT 게이트웨이 생성

Name	NAT 게이트웨이 ID	연결 유형	상태	상태 메시지	기본 퍼블릭 IPv4...	기본 프라이빗
nat_gateway	nat-02c5442c8fb7e77db	Public	Available	–	3.18.25.89	10.0.2.30

- 보안 그룹



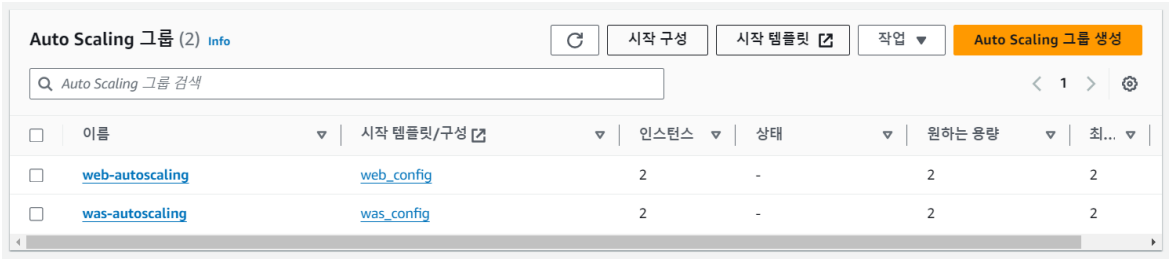
보안 그룹 (7) 정보

Find resources by attribute or tag

작업 | 보안 그룹을 CSV로 내보내기 | 보안 그룹 생성

Name	Security group ID	보안 그룹 이름	VPC ID	설명
web_security_group	sg-0ced3bf77d8836ee3	web_security_group	vpc-0faa3a0e8c427a679	Allow SSH/HT
was_security_group	sg-0bbf377810cfe0a86	was_security_group	vpc-0faa3a0e8c427a679	Allow SSH inb
vpc_security_group	sg-0185d82275f1930ca	vpc_security_group	vpc-0faa3a0e8c427a679	Allow SSH/HT
lb_security_group	sg-0e0aa8fa3384384d9	lb_security_group	vpc-0faa3a0e8c427a679	Allow TCP inb

- 오토스케일링



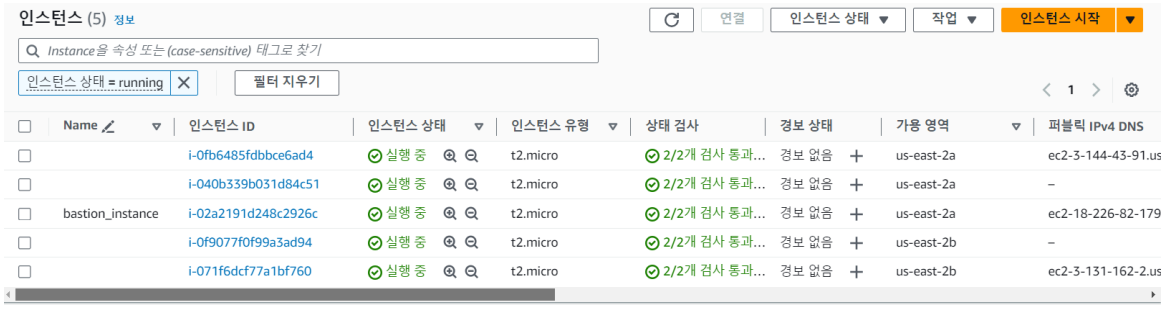
Auto Scaling 그룹 (2) Info

Auto Scaling 그룹 검색

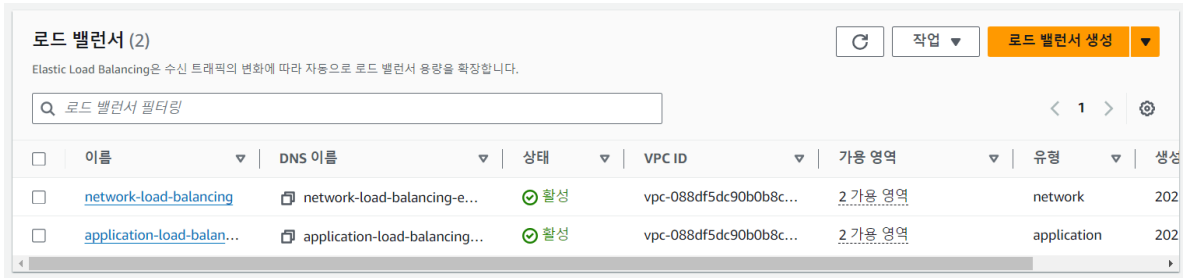
시작 구성 | 시작 템플릿 | 작업 | Auto Scaling 그룹 생성

이름	시작 템플릿/구성	인스턴스	상태	원하는 용량	최...
web-autoscaling	web_config	2	-	2	2
was-autoscaling	was_config	2	-	2	2

- 인스턴스



- 로드밸런서



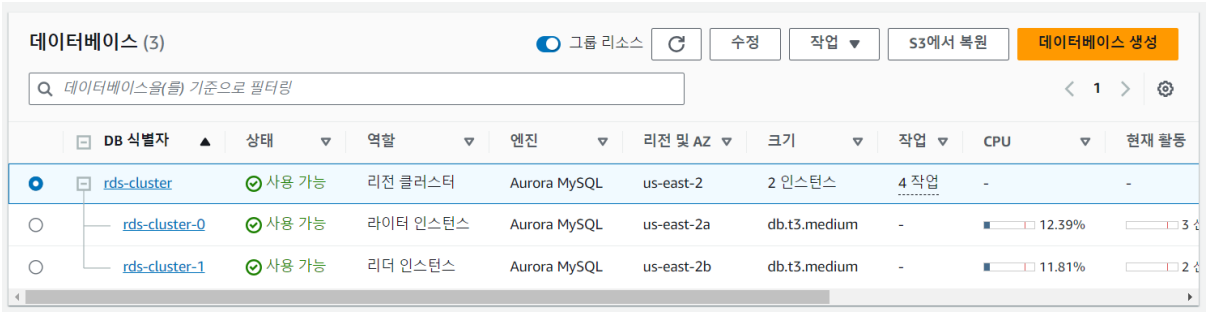
- 연결 확인

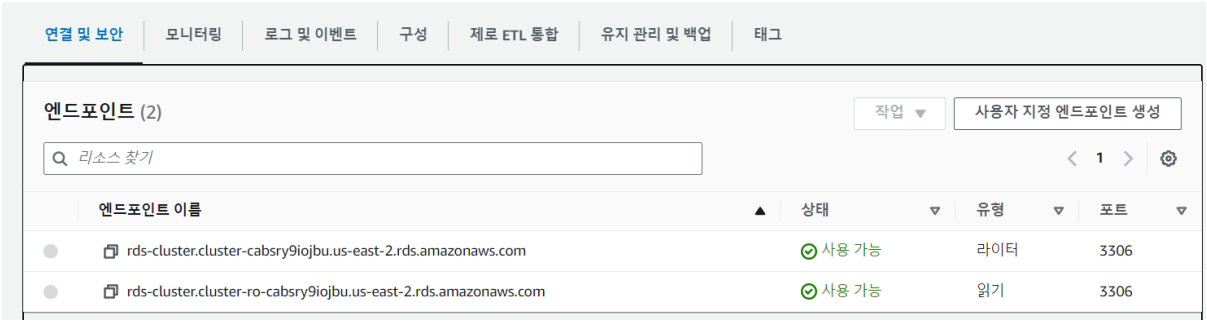
```
[tf@main ~]$ ssh -i ~/.ssh/testkey ubuntu@3.138.101.224
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-1016-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sat Nov 25 20:55:10 UTC 2023

  System load:  0.0               Processes:             96
  Usage of /:   20.8% of 7.57GB   Users logged in:       0
  Memory usage: 21%               IPv4 address for eth0: 10.0.1.47
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


Last login: Sat Nov 25 20:55:11 2023 from 116.36.38.87
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-10-0-1-47:~$ ping 8.8.8.8 -c 4
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=56 time=10.5 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=56 time=10.5 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=56 time=10.5 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=56 time=10.5 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 10.478/10.512/10.538/0.023 ms
ubuntu@ip-10-0-1-47:~$
```

- S3 버킷


버킷 (1) 정보
버킷은 S3에 저장되는 데이터의 컨테이너입니다. 자세히 알아보기
ARN 복사
비어 있음
삭제
버킷 만들기
이름으로 버킷 찾기
1
이름
AWS 리전
액세스
생성 날짜
kyj-bucket
미국 동부(오하이오) us-east-2
버킷 및 객체가 퍼블릭이 아님
2023. 11. 26. am 2:45:48 AM KST

- db 클러스터


데이터베이스 (3)
그룹 리소스
수정
작업
S3에서 복원
데이터베이스 생성
데이터베이스을(를) 기준으로 필터링
1
DB 식별자
상태
역할
엔진
리전 및 AZ
크기
작업
CPU
현재 활동
rds-cluster
사용 가능
리전 클러스터
Aurora MySQL
us-east-2
2 인스턴스
4 작업
rds-cluster-0
사용 가능
라이터 인스턴스
Aurora MySQL
us-east-2a
db.t3.medium
12.39%
rds-cluster-1
사용 가능
리더 인스턴스
Aurora MySQL
us-east-2b
db.t3.medium
11.81%


연결 및 보안
모니터링
로그 및 이벤트
구성
제로 ETL 통합
유지 관리 및 백업
태그
엔드포인트 (2)
작업
사용자 지정 엔드포인트 생성
리소스 찾기
1
엔드포인트 이름
상태
유형
포트
rds-cluster.cluster-cabsry9iojbu.us-east-2.rds.amazonaws.com
사용 가능
라이터
3306
rds-cluster.cluster-ro-cabsry9iojbu.us-east-2.rds.amazonaws.com
사용 가능
읽기
3306