ECS Fargate Nginx/Wordpress Server Deployment with Terraform (Task 4)

Zaeem Attique Ashar

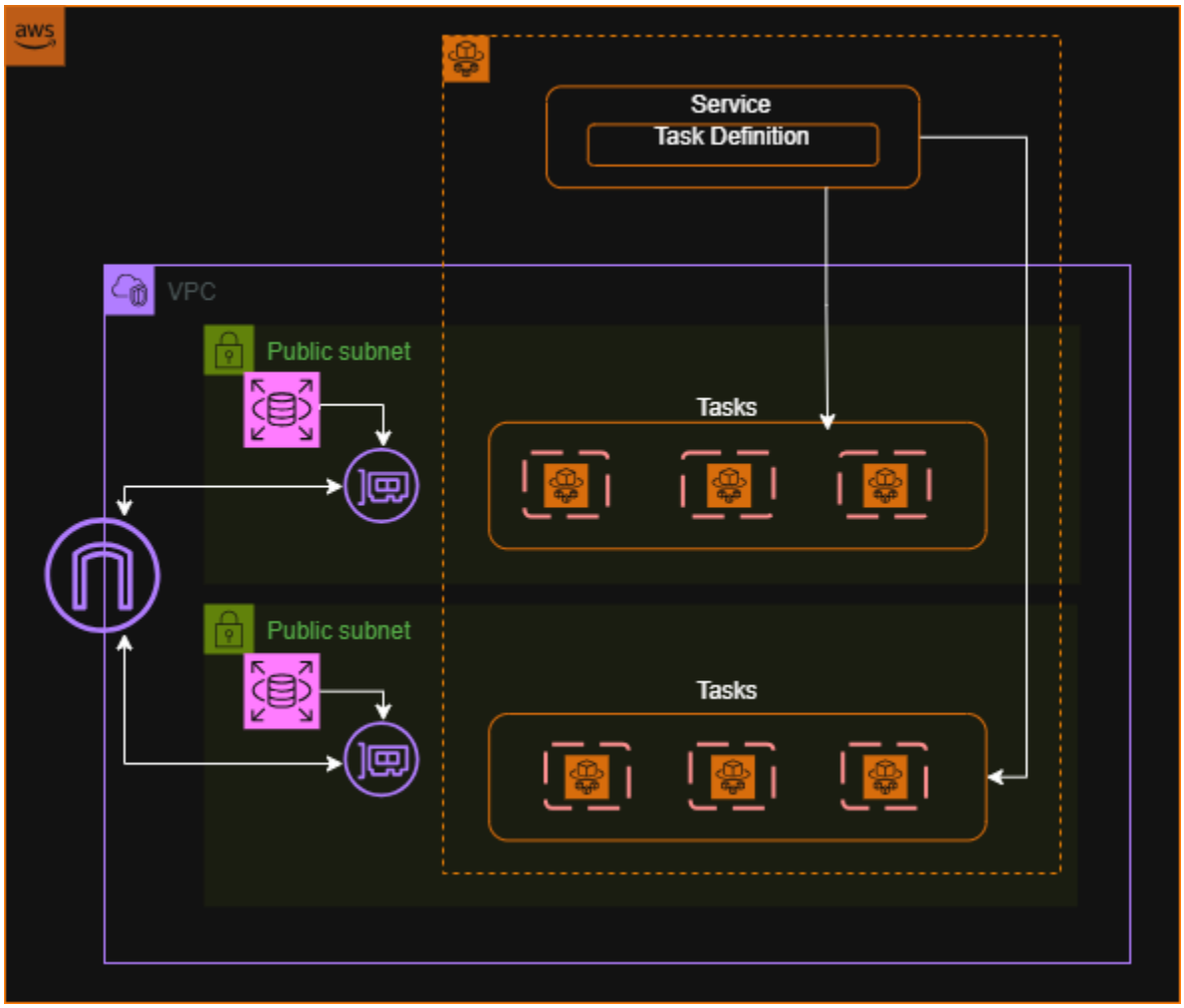
Cloud Intern

Task Description:

This is a guide on deploying a containerized Nginx/Wordpress server on an ECS Cluster. The infrastructure will be deployed through terraform code and will be using many different components such as EC2 Fargate Cluster, Execution Roles, Task Roles, as well as basic infrastructure such as VPC and Subnets.

Architecture Diagram: 2
Task4.1: Define a custom VPC with public subnets 3
Task4.2: Create ECS Cluster Using Terraform 4
Task4.3: Configure an ECS Service for the Cluster 6
Task4.4: Setup IAM Role and Execution Policy 7
Task4.5: Verification of active Wordpress Server Container 8
Task4.6: Challenges Faced 8

Architecture Diagram:

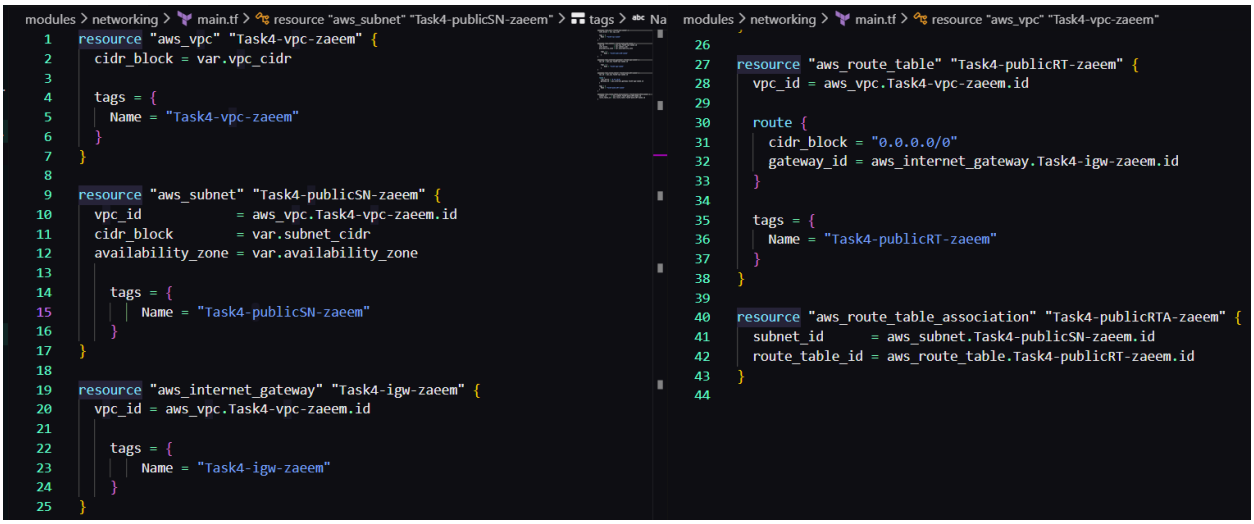


Task4.1: Define a custom VPC with public subnets

This section takes care of deploying the basic infrastructure needed.

- VPC Configuration:
  - CIDR: 10.0.0.0/16.
- Subnets Configuration:
  Availability Zone A:
  - CIDR Block for subnet: 10.0.1.0/24.
  - Availability Zone: us-west-2a.

  Availability Zone B:
  - CIDR Block for subnet: 10.0.2.0/24.
  - Availability Zone: us-west-2a.
- Internet Gateway Configuration:
  - VPC ID.
- Route Table (Public):
  - VPC ID.
  - Route: Destination 0.0.0.0/0, Target Internet Gateway.
- Security Group (Public):
  - Inbound SSH Traffic: Port 22, source 0.0.0.0/0.
  - Inbound HTTP Traffic: Port 80, source 0.0.0.0/0.



Task4.2: Create ECS Cluster Using Terraform

- ECS Cluster:
  - Name (Unique).
- ECS Task Definition:
  - Family: Task4FamilyZaeem.
  - Network Mode: awsvpc (cluster will be deployed inside the VPC).
  - Compatibility Requirement: Fargate (Serverless architecture).
  - CPU and Memory: 512, 1024 (Resources that the task will possess entirely).
  - Execution Role: ECS Task Execution Role (IAM Role for executing the task).
  - Container Definition:
    - Name: Wordpress-Container (Name of the Task/Container).
    - Image: wordpress:6.8.3-php8.1-apache (Docker Hub image that will be pulled to run).
    - CPU: 256 (CPU allocation for each task).
    - Memory = 512 (Memory allocation for each task).
    - Port Mappings: containerPort = 80, hostPort = 80, protocol = "tcp" (Task/Container port 80 will be exposed on the instance port 80

- A database is also needed for the wordpress to function normally. In this task, an RDS database is provisioned and connected to the wordpress container through environment variables.
- Configuration:
    - identifier: task4-mysql
    - allocated_storage: 20
    - engine: mysql
    - engine_version: 8.0
    - instance_class: db.t3.micro
    - db_name: wordpressdb
    - username: root
    - Password: YourPassword123!
    - db_subnet_group_name: aws_db_subnet_group.rds_subnet_group.name
    - vpc_security_group_ids: aws_security_group.rds_sg.id
- The RDS instance needs 2 more resources:
    - Database Subnet Group:
        - Subnet IDs: Subnet-A ID, Subnet-B ID
    - VPC Security Group:
        - Ingress: port 3306, target ECS Security group
        - Egress: port any, CIDR 0.0.0.0/0

```
resource "aws_db_instance" "task4_mysql" {
  identifier           = "task4-mysql"
  allocated_storage    = 20
  engine               = "mysql"
  engine_version       = "8.0"
  instance_class       = "db.t3.micro"
  db_name              = "wordpressdb"
  username             = "root"
  password             = "YourPassword123!"
  db_subnet_group_name = aws_db_subnet_group.rds_subnet_group.name
  vpc_security_group_ids = [aws_security_group.rds_sg.id]

  skip_final_snapshot = true
  publicly_accessible = false
}
```

```
resource "aws_security_group" "rds_sg" {
  name        = "task4-rds-sg"
  description = "Allow access from ECS tasks"
  vpc_id      = var.vpc_id

  ingress {
    description      = "MySQL from ECS"
    from_port        = 3306
    to_port          = 3306
    protocol         = "tcp"
    security_groups  = [var.sg_id]  # ECS task SG
  }

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

resource "aws_db_subnet_group" "rds_subnet_group" {
  name       = "task4-rds-subnet-group"
  subnet_ids = [var.public_subnet_id_A, var.public_subnet_id_B]

  tags = {
    Name = "task4-rds-subnet-group"
  }
}
```

Task4.3: Configure an ECS Service for the Cluster

- ECS Service (For monitoring the task and keeping it to the desired state):
  - Name: Task4-ECS-Service-Zaeem.
  - Cluster: Cluster ID (ID of the cluster it will be a part of).
  - Task Definition: Task Definition ID (Task Definition according to which the task will be deployed).
  - Desired Count: 2 (Number of tasks that are desired to be running).
  - Launch Type: Fargate (Serverless/AWS Managed instances).
  - Network Configuration:
    - Subnets: Subnet AZ-A ID, Subnet AZ-B ID (Subnets inside which the fargate instances will be launched to run the tasks).
    - Security Group: Security Group ID (Security Group that will control the ingress/egress traffic of the task).

```
32  resource "aws_ecs_service" "Task4-ECS-Service-Zaeem" {
33    name            = "Task4-ECS-Service-Zaeem"
34    cluster         = aws_ecs_cluster.Task4-ECS-Cluster-Zaeem.id
35    task_definition = aws_ecs_task_definition.Task4-TaskDefinition-Zaeem.arn
36    desired_count   = 1
37    launch_type     = "FARGATE"
38
39
40    network_configuration {
41      subnets         = [var.public_subnet_id]
42      security_groups = [var.sg_id]
43      assign_public_ip = true
44    }
45
46  }
```

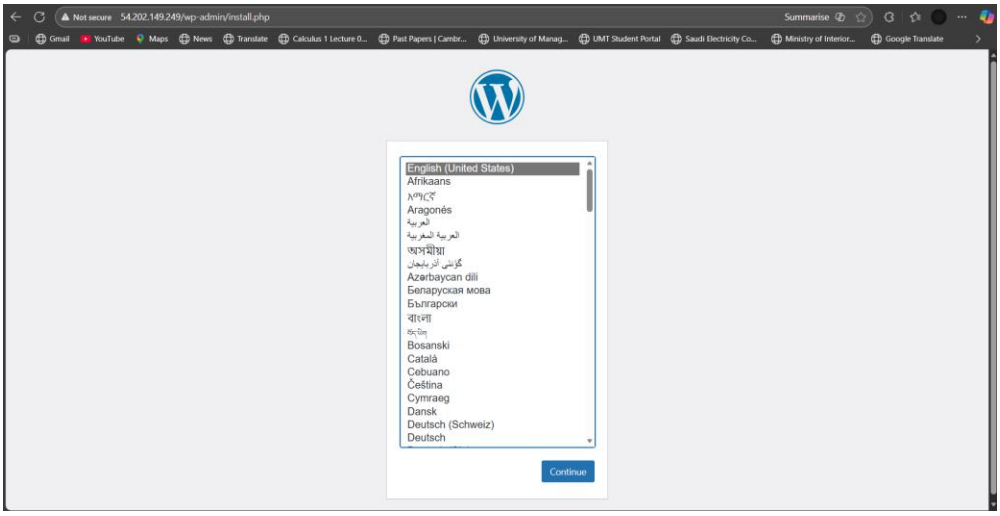
Task4.4: Setup IAM Role and Execution Policy

- IAM Execution Role:
  - Name: ecsTaskExecutionRoleZaeem
  - Policy Attached: AmazonECSTaskExecutionRolePolicy
  - No Task Policy is required for the current task.

```
48  resource "aws_iam_role" "ecs_task_execution_role" {
49    name = "ecsTaskExecutionRoleZaeem"
50
51    assume_role_policy = jsonencode({
52      Version = "2012-10-17"
53      Statement = [{
54        Effect = "Allow"
55        Principal = {
56          Service = "ecs-tasks.amazonaws.com"
57        }
58        Action = "sts:AssumeRole"
59      }]
60    })
61  }
62
63  resource "aws_iam_role_policy_attachment" "ecs_execution_role_policy" {
64    role       = aws_iam_role.ecs_task_execution_role.name
65    policy_arn = "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
66  }
```

Task4.5: Verification of active Wordpress Server Container

The Wordpress homepage should be available on Port 80 of the public IP of the instance. We can find the public IP in the Tasks > Networking dashboard.



Task4.6: Challenges Faced

- The network configuration that we require will be defined under the service block and not the cluster block or the task definition.
- The security group should have appropriate outbound rules to be able to pull the image from Docker Hub and inbound rules to interact with the Wordpress open port.
- The wordpress container needs a Database Instance to start the wordpress service. We will need to provision a database externally as it does not come with the wordpress image. It can be either a database resource or a container running an database Image.