Terraform Configuration for AWS EC2 Instance and Security Group

Created by Grok 3

June 18, 2025

1 Overview

This document provides a Terraform configuration to provision an AWS EC2 instance and a security group. It demonstrates Terraform's core concepts: Providers, Resources, State, Variables, Modules, Data Sources, and Outputs. The configuration is split into multiple files for modularity and includes commands to execute the script.

2 Terraform Files

2.1 provider.tf

```
provider "aws" {
  region = var.region
}
```

2.2 variables.tf

```
variable "region" {
  description = "AWS region for resource deployment"
  type        = string
  default     = "us-east-1"
}

variable "instance_type" {
  description = "EC2 instance type"
  type        = string
  default     = "t2.micro"
}

variable "allowed_ports" {
  description = "List of ports to allow in security group"
  type        = list(number)
  default     = [22, 80]
}
```

2.3 modules/security_group/main.tf

```
resource "aws_security_group" "main" {
  name        = var.sg_name
  description = "Security group for EC2 instance"
  vpc_id      = var.vpc_id

  dynamic "ingress" {
    for_each = var.allowed_ports
    content {
      from_port   = ingress.value
      to_port     = ingress.value
      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 = var.sg_name
  }
}

variable "sg_name" {
  description = "Name of the security group"
  type        = string
}

variable "vpc_id" {
  description = "VPC ID for the security group"
  type        = string
}

variable "allowed_ports" {
  description = "List of ports to allow"
  type        = list(number)
}
```

2.4 main.tf

```
# Fetch the default VPC
data "aws_vpc" "default" {
  default = true
}

```

```
# Fetch the latest Ubuntu AMI
data "aws_ami" "latest_ubuntu" {
  most_recent = true
  filter {
    name   = "name"
    values = ["ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-
       server-*"]
  }
  owners = ["099720109477"] # Canonical
}

# Call the security group module
module "security_group" {
  source        = "./modules/security_group"
  sg_name       = "ec2-security-group"
  vpc_id        = data.aws_vpc.default.id
  allowed_ports = var.allowed_ports
}

# Create the EC2 instance
resource "aws_instance" "web" {
  ami                    = data.aws_ami.latest_ubuntu.id
  instance_type          = var.instance_type
  vpc_security_group_ids = [module.security_group.
     aws_security_group.main.id]
  subnet_id              = data.aws_subnet_ids.available.subnets
     [0]
  associate_public_ip    = true

  tags = {
    Name = "web-server"
  }
}

# Fetch available subnets in the default VPC
data "aws_subnet_ids" "available" {
  vpc_id = data.aws_vpc.default.id
}
```

2.5 outputs.tf

```
output "instance_public_ip" {
  description = "Public IP of the EC2 instance"
  value       = aws_instance.web.public_ip
}

output "security_group_id" {
  description = "ID of the security group"
  value       = module.security_group.aws_security_group.main.id
}
```

3 State Management

The state is stored locally in `terraform.tfstate` by default. For team collaboration, configure a remote backend (e.g., S3):

```
terraform {
  backend "s3" {
    bucket = "my-terraform-state"
    key    = "state/terraform.tfstate"
    region = "us-east-1"
  }
}
```

4 Execution Commands

Run the following commands in order:

1. `terraform init` – Initialize the working directory and download providers.
2. `terraform validate` – Validate the configuration.
3. `terraform fmt` – Format the configuration files.
4. `terraform plan` – Preview the changes.
5. `terraform apply` – Apply the configuration to provision resources.
6. `terraform output` – View the outputs (e.g., public IP).
7. `terraform destroy` – Destroy the resources when done.

5 Core Concepts Demonstrated

- **Providers**: The AWS provider is configured in `provider.tf`.
- **Resources**: EC2 instance and security group are defined as resources.
- **State**: Managed locally or via an S3 backend.
- **Variables**: Defined in `variables.tf` for parameterization.
- **Modules**: Security group is encapsulated in a reusable module.
- **Data Sources**: Used to fetch the default VPC, subnets, and latest Ubuntu AMI.
- **Outputs**: Public IP and security group ID are output for reference.