# Lab 10 – Creating AWS Security Groups with Terraform (Beginner to Intermediate)

**Creator:** Sandeep Kumar Sharma

---

## Learning Objectives

- Understand what a Security Group is in AWS.
- Learn why Security Groups are required for EC2 instances.
- Learn how to create Security Groups using Terraform.
- Learn how to define ingress (inbound) and egress (outbound) rules.
- Attach a Security Group to an EC2 instance using Terraform.

---

## Learning Outcome

By the end of this lab, the learner will be able to: - Explain Security Groups in simple terms. - Write Terraform code to create Security Groups. - Configure inbound and outbound rules. - Attach a Security Group to an EC2 instance. - Deploy a secure EC2 environment using Terraform.

---

## Concept Explanation (Natural Style)

Let’s understand the Security Group in a very simple way.

Imagine you purchased a new smartphone. Before using the phone, what's the first thing you do? You set up a **screen lock**. Why? To protect your data.

Security Groups work exactly like that in AWS.

### 👉What is a Security Group?

A Security Group is like a **virtual firewall** for your EC2 instance.

It controls: - Who can enter (inbound rules) - Who can leave (outbound rules)

### 👉Why do we need a Security Group?

Because without it: - Anyone could access your EC2 instance - Your server would be exposed to the internet - Your data and applications become vulnerable

### 👉Inbound Rules

Inbound rules define what type of traffic can enter your EC2 instance. Example: - Allow SSH (port 22) from your IP - Allow HTTP (port 80) from anywhere (for a website)

### 👉Outbound Rules

Outbound rules define where your EC2 instance can connect to. By default, AWS allows **all outbound traffic**, which is safe for beginners.

---

# 😠Part 1: Create the Project Structure

```
mkdir terraform-lab10-security-groups
cd terraform-lab10-security-groups
```

Create a file:

```
touch main.tf
```

---

# 😠Part 2: Write Terraform Code

## main.tf

```
provider "aws" {
  region = "ap-south-1"
}

# ------------------------------
# Create Security Group
# ------------------------------
resource "aws_security_group" "lab10_sg" {
  name        = "terraform-lab10-sg"
  description = "Security group for Lab 10 EC2 instance"
  vpc_id      = null  # Uses default VPC

  # Inbound Rule: Allow SSH
  ingress {
    from_port   = 22
    to_port     = 22
```

```
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  # Inbound Rule: Allow HTTP
  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  # Outbound Rule: Allow all
  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags = {
    Name = "Terraform-Lab10-SG"
  }
}

# ------------------------------
# Create EC2 Instance with SG
# ------------------------------
resource "aws_instance" "lab10_ec2" {
  ami           = "ami-052c08d70def0ac62"
  instance_type = "t2.micro"

  vpc_security_group_ids = [aws_security_group.lab10_sg.id]

  tags = {
    Name = "Terraform-Lab10-EC2"
  }
}

# Outputs
output "instance_public_ip" {
  value = aws_instance.lab10_ec2.public_ip
}

output "security_group_id" {
  value = aws_security_group.lab10_sg.id
}
```

---

# 😠Part 3: Initialize Terraform

```
terraform init
```

This downloads the AWS provider plugin.

---

# 😠Part 4: Run Terraform Plan

```
terraform plan
```

Terraform will show: - 1 Security Group to create - 1 EC2 Instance to create

---

# 😠Part 5: Apply the Configuration

```
terraform apply
```

Type **yes**.

Terraform will: - Create the Security Group - Create an EC2 instance - Attach the Security Group to the instance - Display Public IP and SG ID

---

# 😠Part 6: Validate in AWS Console

### Check Security Group

- Go to **EC2 → Security Groups**
- Search for **terraform-lab10-sg**
- Inspect inbound and outbound rules

### Check EC2

- Go to **EC2 → Instances**
- Find **Terraform-Lab10-EC2**
- Verify Security Group attached
- Verify Public IP

---

# 😠Part 7: Connect via SSH (Optional)

If you have a key pair, you can SSH into the server:

```
ssh ec2-user@<public_ip>
```

---

# 😠Part 8: Destroy (Optional)

```
terraform destroy
```

Type **yes**.

This safely removes: - Security Group - EC2 instance

---

## Summary

In this lab, you learned: - What AWS Security Groups are and why they matter. - How to create a Security Group with Terraform. - How to define inbound and outbound rules. - How to attach a Security Group to an EC2 instance. - How Terraform deploys secure infrastructure.

In the next lab, we will go one level deeper and create **VPC, Subnets, Route Tables**, and attach EC2 with full networking.

---

**End of Lab 10**