Lab 4: Creating Network Security Group (NSG) and Security Rules in Microsoft Azure using Terraform

Author: Dr. Sandeep Kumar Sharma
Level: Beginner
Platform: Ubuntu Linux + Microsoft Azure
Prerequisite: Lab 1 (Setup), Lab 2 (Resource Group), Lab 3 (VNet + Subnet)

Learning Objective

Participants will learn how to:

- Understand the purpose of Network Security Group (NSG)
- Create an NSG using Terraform
- Create inbound security rules
- Attach NSG to a subnet
- Control network traffic using security rules

Learning Outcome

After completing this lab, participants will be able to:

- Secure Azure network resources
- Manage inbound traffic rules
- Use Terraform to implement network security
- Understand traffic control at subnet level

Concept Overview

What is a Network Security Group (NSG)?

A Network Security Group (NSG) is used to **control network traffic** in Azure.

It works using **security rules** that allow or deny traffic based on: - Source - Destination - Port - Protocol - Direction (Inbound/Outbound)

Hands-On Lab

Step 1: Go to Terraform Directory

```
cd terraform-azure-lab
```

Step 2: Create Terraform File for NSG

```
touch nsg.tf
```

Step 3: Open File

```
nano nsg.tf
```

Step 4: Write Terraform Code for NSG

```
resource "azurerm_network_security_group" "nsg1" {
  name                = "nsg-terraform-lab"
  location            = azurerm_resource_group.rg1.location
  resource_group_name = azurerm_resource_group.rg1.name
}
```

Step 5: Create Security Rules

Allow SSH (Port 22)

```
resource "azurerm_network_security_rule" "allow_ssh" {
  name                        = "allow-ssh"
  priority                    = 100
  direction                   = "Inbound"
  access                      = "Allow"
  protocol                    = "Tcp"
  source_port_range           = "*"
```

```
  destination_port_range      = "22"
  source_address_prefix       = "*"
  destination_address_prefix  = "*"
  resource_group_name         = azurerm_resource_group.rg1.name
  network_security_group_name = azurerm_network_security_group.nsg1.name
}
```

Allow HTTP (Port 80)

```
resource "azurerm_network_security_rule" "allow_http" {
  name                        = "allow-http"
  priority                    = 200
  direction                   = "Inbound"
  access                      = "Allow"
  protocol                    = "Tcp"
  source_port_range           = "*"
  destination_port_range      = "80"
  source_address_prefix       = "*"
  destination_address_prefix  = "*"
  resource_group_name         = azurerm_resource_group.rg1.name
  network_security_group_name = azurerm_network_security_group.nsg1.name
}
```

Step 6: Attach NSG to Subnet

```
resource "azurerm_subnet_network_security_group_association"
"nsg_subnet_attach" {
  subnet_id                 = azurerm_subnet.subnet1.id
  network_security_group_id = azurerm_network_security_group.nsg1.id
}
```

Step 7: Initialize Terraform

```
terraform init
```

Step 8: Preview Changes

```
terraform plan
```

Step 9: Apply Configuration

```
terraform apply
```

Type:

```
yes
```

Step 10: Verify in Azure Portal

Go to:

Resource Group → `rg-terraform-lab`

Verify: - NSG: `nsg-terraform-lab` - Inbound rules: SSH (22), HTTP (80) - NSG attached to subnet `subnet-terraform-lab`

Step 11: Verify using Azure CLI

```
az network nsg list -o table
```

```
az network nsg rule list --resource-group rg-terraform-lab --nsg-name nsg-terraform-lab -o table
```

```
az network vnet subnet show --resource-group rg-terraform-lab --vnet-name vnet-terraform-lab --name subnet-terraform-lab
```

Step 12: Cleanup

```
terraform destroy
```

Type:

```
yes
```