Step-01: Introduction

- We will create the below Azure Resources using Terraform

1. Azure Resource Group
2. Azure Virtual Network
3. Azure Subnet
4. Azure Public IP
5. Azure Network Interface
6. Azure Linux Virtual Machine
7. `random_string` Resource

- We will use Azure `custom_data` argument in `azurerm_linux_virtual_machine` to install a simple webserver during the creation of VM.
- Terraform file Function
- Terraform filebase64 Function

Step-02: Create SSH Keys for Azure Linux VM

```
# Create Folder
cd terraform-manifests/
mkdir ssh-keys

# Create SSH Key
cd ssh-ekys
ssh-keygen \
    -m PEM \
    -t rsa \
    -b 4096 \
    -C "azureuser@myserver" \
    -f terraform-azure.pem
Important Note: If you give passphrase during generation, during everytime you login to VM, you also need to provide pa

# List Files
ls -lrt ssh-keys/

# Files Generated after above command
Public Key: terraform-azure.pem.pub -> Rename as terraform-azure.pub
Private Key: terraform-azure.pem

# Permissions for Pem file
chmod 400 terraform-azure.pem
```

Step-03: c1-versions.tf - Create Terraform & Provider Blocks

- Create Terraform Block
- Create Provider Block
- Create Random Resource Block

```
# Terraform Block
terraform {
  required_version = ">= 1.0.0"
  required_providers {
    azurerm = {
      source = "hashicorp/azurerm"
      version = ">= 2.0"
    }
    random = {
      source = "hashicorp/random"
```

```
      version = ">= 3.0"
    }
  }
}

# Provider Block
provider "azurerm" {
 features {}
}

# Random String Resource
resource "random_string" "myrandom" {
  length = 6
  upper = false
  special = false
  number = false
}
```

Step-04: c2-resource-group.tf

```
# Resource-1: Azure Resource Group
resource "azurerm_resource_group" "myrg" {
  name = "myrg-1"
  location = "East US"
}
```

Step-05: c3-vritual-network.tf - Virtual Network Resource

```
# Create Virtual Network
resource "azurerm_virtual_network" "myvnet" {
  name                = "myvnet-1"
  address_space       = ["10.0.0.0/16"]
  location            = azurerm_resource_group.myrg.location
  resource_group_name = azurerm_resource_group.myrg.name
}
```

Step-06: c3-vritual-network.tf - Azure Subnet Resource

```
# Create Subnet
resource "azurerm_subnet" "mysubnet" {
  name                 = "mysubnet-1"
  resource_group_name  = azurerm_resource_group.myrg.name
  virtual_network_name = azurerm_virtual_network.myvnet.name
  address_prefixes     = ["10.0.2.0/24"]
}
```

Step-07: c3-vritual-network.tf - Azure Public IP Resource

```
# Create Public IP Address
resource "azurerm_public_ip" "mypublicip" {
  name                = "mypublicip-1"
  resource_group_name = azurerm_resource_group.myrg.name
  location            = azurerm_resource_group.myrg.location
  allocation_method   = "Static"
  domain_name_label = "app1-vm-${random_string.myrandom.id}"
  tags = {
```

```
    environment = "Dev"
  }
}
```

Step-08: c3-vritual-network.tf - Network Interface Resource

```
# Create Network Interface
resource "azurerm_network_interface" "myvmnic" {
  name                = "vmnic"
  location            = azurerm_resource_group.myrg.location
  resource_group_name = azurerm_resource_group.myrg.name

  ip_configuration {
    name                          = "internal"
    subnet_id                     = azurerm_subnet.mysubnet.id
    private_ip_address_allocation = "Dynamic"
    public_ip_address_id = azurerm_public_ip.mypublicip.id
  }
}
```

Step-09: c4-linux-virtual-machine.tf

```
# Resource: Azure Linux Virtual Machine
resource "azurerm_linux_virtual_machine" "mylinuxvm" {
  name                = "mylinuxvm-1"
  computer_name       = "devlinux-vm1" # Hostname of the VM
  resource_group_name = azurerm_resource_group.myrg.name
  location            = azurerm_resource_group.myrg.location
  size                = "Standard_DS1_v2"
  admin_username      = "azureuser"
  network_interface_ids = [
    azurerm_network_interface.myvmnic.id
  ]
  admin_ssh_key {
    username   = "azureuser"
    public_key = file("${path.module}/ssh-keys/terraform-azure.pub")
  }
  os_disk {
    name = "osdisk"
    caching              = "ReadWrite"
    storage_account_type = "Standard_LRS"
  }
  source_image_reference {
    publisher = "RedHat"
    offer     = "RHEL"
    sku       = "83-gen2"
    version   = "latest"
  }
  custom_data = filebase64("${path.module}/app-scripts/app1-cloud-init.txt")
}
```

Step-10: app1-cloud-init.txt

```
#cloud-config
package_upgrade: false
packages:
  - httpd
write_files:
  - owner: root:root
    path: /var/www/html/index.html
    content: |
```

```
      <h1>Welcome to Hashicorp Terraform - APP-1</h1>
  - owner: root:root
    path: /var/www/html/app1/index.html
    content: |
      <!DOCTYPE html> <html> <body style="background-color:rgb(250, 210, 210);"> <h1>Welcome to Hashicorp Terraform - A
runcmd:
  - sudo systemctl start httpd
  - sudo systemctl enable httpd
  - sudo systemctl stop firewalld
  - sudo mkdir /var/www/html/app1
  - [sudo, curl, -H, "Metadata:true", --noproxy, "*", "http://169.254.169.254/metadata/instance?api-version=2020-09-01"
```

Step-11: Execute Terraform commands to Create Resources using Terraform

```
# Initialize Terraform
terraform init

# Terraform Validate
terraform validate

# Terraform Plan
terraform plan

# Terraform Apply
terraform apply
```

Step-12: Verify the Resources

- Verify Resources

1. Azure Resource Group
2. Azure Virtual Network
3. Azure Subnet
4. Azure Public IP
5. Azure Network Interface
6. Azure Virtual Machine

```
# Connect to VM and Verify
ssh -i ssh-keys/terraform-azure.pem azureuser@<PUBLIC-IP>

# Access Application
http://<PUBLIC_IP>
http://<PUBLIC_IP>/app1
http://<PUBLIC_IP>/app1/metadata.html
```

Step-13: Destroy Terraform Resources

```
# Destroy Terraform Resources
terraform destroy

# Remove Terraform Files
rm -rf .terraform*
rm -rf terraform.tfstate*
```

References

1. Azure Resource Group
2. Azure Virtual Network
3. Azure Subnet
4. Azure Public IP
5. Azure Network Interface
6. Azure Virtual Machine