THISARA KANDAGE

UNDERGRADUATE - SLIIT

E-mail LinkedIn GitHub Website

Infrastructure Provisioning on Microsoft Azure Using Terraform

Project Description

This project demonstrates infrastructure provisioning on Microsoft Azure using Terraform, an Infrastructure as Code (IaC) tool. The primary goal was to automate the creation and management of essential Azure resources through declarative configuration files, enabling reproducible and consistent infrastructure deployments.

The Terraform configuration includes defining a resource group, a storage account, a virtual network with a subnet, a public IP address, a network interface, and a Linux virtual machine. The resource group serves as a logical container that organizes all the resources within a specific Azure region. The storage account was created with standard performance and locally-redundant storage replication to provide reliable and cost-effective cloud storage. The virtual network and subnet allow for secure and isolated communication between cloud resources. The public IP address and network interface facilitate network connectivity for the virtual machine, which runs Ubuntu Linux and is provisioned with specified sizing and administrative access.

This infrastructure setup mimics a basic cloud environment that can host applications or services, demonstrating core Azure networking and compute capabilities. All resources were defined in Terraform configuration files, and the infrastructure was provisioned using Terraform CLI commands. The use of Terraform enabled efficient resource management, version control of infrastructure, and repeatable deployments across different environments.

THISARA KANDAGE

UNDERGRADUATE - SLIIT

E-mail LinkedIn GitHub Website

```
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project> Set-ExecutionPolicy Bypass -Scope Process -Force; `
>> [System.Net.ServicePointManager]::SecurityProtocol = `
>> [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; `
>> iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))
WARNING: 'choco' was found at 'C:\ProgramData\chocolatey\bin\choco.exe'.
WARNING: An existing Chocolatey installation was detected. Installation will not continue. This
script will not overwrite existing installations.
If there is no Chocolatey installation at 'C:\ProgramData\chocolatey', delete the folder and attempt
 the installation again.
```

- **Install terraform**

```
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project> choco install terraform -y
Chocolatey v2.4.3
Installing the following packages:
terraform
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
```

```
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project> terraform -v
Terraform v1.12.2
on windows_amd64
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project>
```

- **Installing azure cli**

```
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project> choco install azure-cli -y
Chocolatey v2.4.3
Installing the following packages:
azure-cli
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading azure-cli 2.75.0... 100%
```

THISARA KANDAGE

UNDERGRADUATE - SLIIT

E-mail LinkedIn GitHub Website

```
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project> az -v
azure-cli                         2.75.0

core                              2.75.0
telemetry                          1.1.0

Dependencies:
msal                            1.33.0b1
azure-mgmt-resource               23.3.0

Python location 'C:\Program Files\Microsoft SDKs\Azure\CLI2\python.exe'
Config directory 'C:\Users\thisa\.azure'
Extensions directory 'C:\Users\thisa\.azure\cliextensions'

Python (Windows) 3.12.10 (tags/v3.12.10:0cc8128, Apr  8 2025, 12:21:36) [MSC v.1943 64 bit (AMD64)]

Legal docs and information: aka.ms/AzureCliLegal


Your CLI is up-to-date.
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project>
```

- **Provides configuration details for Terraform**

main.tf

```hcl
# Provides configuration details for Terraform
terraform {
  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "~>2.31.1"
    }
  }
}
```

- **Provides configuration details for the Azure Terraform provider**

```hcl
# Provides configuration details for the Azure Terraform provider
provider "azurerm" {
  features {}
}
```

- **Provides the Resource Group to Logically contain resources**

```
# Provides the Resource Group to Logically contain resources
resource "azurerm_resource_group" "rg" {
  name     = "Thisara-test1"
  location = "southcentralus"
  tags = {
    environment = "dev"
    name        = "Terraform"
  }
}
```

- **Terraform init**

```
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/azurerm versions matching "~> 2.31.1"...
- Installing hashicorp/azurerm v2.31.1...
- Installed hashicorp/azurerm v2.31.1 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!
```

- **Terraform format**

```
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project> terraform fmt
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project>
```

- **Terraform validate**

```
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project> terraform validate
Success! The configuration is valid.

PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project>
```

THISARA KANDAGE

UNDERGRADUATE - SLIIT

E-mail LinkedIn GitHub Website

- **Terraform plan**

```
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project> terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # azurerm_resource_group.rg will be created
  + resource "azurerm_resource_group" "rg" {
      + id       = (known after apply)
      + location = "southcentralus"
      + name     = "Thisara-test1"
      + tags     = {
          + "environment" = "dev"
```

- **Terraform apply**

```
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project> terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # azurerm_resource_group.rg will be created
  + resource "azurerm_resource_group" "rg" {
      + id       = (known after apply)
      + location = "southcentralus"
      + name     = "Thisara-test1"
      + tags     = {
```

```
Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

azurerm_resource_group.rg: Creating...
azurerm_resource_group.rg: Creation complete after 6s [id=/subscriptions/e135e03c-672c-4d80-97e8-dac4633b1197/resourceGroups/Thisara-test1]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

- **Add Azure Storage Account via terraform**

```
resource "azurerm_storage_account" "storage" {
  name                     = "thisarateststorage"         # Must be globally unique
  resource_group_name      = azurerm_resource_group.rg.name
  location                 = azurerm_resource_group.rg.location
  account_tier             = "Standard"
  account_replication_type = "LRS"

  tags = {
    environment = "dev"
  }
}
```

- **After apply terraform**

```
azurerm_storage_account.storage: Creating...
azurerm_storage_account.storage: Still creating... [00m10s elapsed]
azurerm_storage_account.storage: Still creating... [00m20s elapsed]
azurerm_storage_account.storage: Still creating... [00m30s elapsed]
azurerm_storage_account.storage: Creation complete after 37s [id=/subscriptions/e135e03c-
ageAccounts/thisarateststorage]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project>
```

- **After all done apply terraform destroy**

```
PS C:\Users\thisa\OneDrive\Desktop\TF-Azure project> terraform destroy
azurerm_resource_group.rg: Refreshing state... [id=/subscriptions/e135e03
```

THISARA KANDAGE

UNDERGRADUATE - SLIIT

E-mail LinkedIn GitHub Website

Summary of Learning

Through this project, I gained practical experience with Terraform and Microsoft Azure cloud services, enhancing my skills in Infrastructure as Code and cloud automation. I learned how to write Terraform configuration files to define various Azure resources, including resource groups, storage accounts, virtual networks, and compute instances. I also became proficient in managing dependencies between resources and understanding the lifecycle of cloud infrastructure from code to deployment.

Additionally, I learned how to handle resource import into Terraform state for existing Azure resources, ensuring infrastructure consistency between Terraform and the actual cloud environment. I explored different Azure VM sizes and understood the importance of selecting resource configurations based on regional availability and quota limits.

Working with the Azure CLI alongside Terraform improved my ability to troubleshoot and validate resource states, and I became familiar with concepts such as SKU selection for public IPs and subnet configurations within virtual networks.

Overall, this project strengthened my understanding of cloud infrastructure provisioning and automation, preparing me for real-world cloud engineering challenges. It also emphasized the value of declarative infrastructure management in improving scalability, reliability, and collaboration within DevOps workflows.