**Steps**

**1. Creating a Resource Group**

1. **Define Resource Group:**
   * Create a Terraform configuration file to define the resource group name and location.
   * Example:

hcl

Copy code

resource "azurerm\_resource\_group" "example" {

name = "example-resources"

location = "East US"

}

1. **Push to Repo:**
   * Commit and push the Terraform configuration to your Azure Repo.
2. **Pipeline Trigger:**
   * The Azure Pipeline is triggered automatically upon code push.
3. **Approval and Deployment:**
   * Approval is requested from Abdul Hamed.
   * Once approved, the resource group is created in Azure.

**2. Creating a Virtual Network (VNet) and Subnets**

1. **Define VNet and Subnets:**
   * Create a Terraform configuration for the VNet and its subnets.
   * Example:

hcl

Copy code

resource "azurerm\_virtual\_network" "example" {

name = "example-vnet"

address\_space = ["10.0.0.0/16"]

location = azurerm\_resource\_group.example.location

resource\_group\_name = azurerm\_resource\_group.example.name

}

resource "azurerm\_subnet" "example" {

name = "example-subnet"

resource\_group\_name = azurerm\_resource\_group.example.name

virtual\_network\_name = azurerm\_virtual\_network.example.name

address\_prefixes = ["10.0.1.0/24"]

}

1. **Push to Repo and Trigger Pipeline:**
   * Commit and push the Terraform configuration.
   * The Azure Pipeline triggers automatically.
2. **Approval and Deployment:**
   * Approval from Abdul Hamed is required.
   * After approval, the VNet and subnets are created in Azure.

**3. Creating Network Security Groups (NSG)**

1. **Define NSG and Rules:**
   * Create a Terraform configuration for NSGs and their rules.
   * Example:

hcl

Copy code

resource "azurerm\_network\_security\_group" "example" {

name = "example-nsg"

location = azurerm\_resource\_group.example.location

resource\_group\_name = azurerm\_resource\_group.example.name

security\_rule {

name = "Allow-SSH"

priority = 1001

direction = "Inbound"

access = "Allow"

protocol = "Tcp"

source\_port\_range = "\*"

destination\_port\_range = "22"

source\_address\_prefix = "\*"

destination\_address\_prefix = "\*"

}

}

1. **Push to Repo and Trigger Pipeline:**
   * Commit and push the Terraform configuration.
   * The Azure Pipeline triggers automatically.
2. **Approval and Deployment:**
   * Approval from Abdul Hamed is required.
   * After approval, the NSGs and rules are created in Azure.

**4. Creating Application Gateways**

1. **Define Application Gateway:**
   * Create Terraform configurations for internal and external application gateways.
   * Example:

hcl

Copy code

resource "azurerm\_application\_gateway" "example" {

name = "example-appgw"

location = azurerm\_resource\_group.example.location

resource\_group\_name = azurerm\_resource\_group.example.name

gateway\_ip\_configuration {

name = "example-gateway-ip-config"

subnet\_id = azurerm\_subnet.example.id

}

frontend\_ip\_configuration {

name = "example-frontend-ip"

public\_ip\_address\_id = azurerm\_public\_ip.example.id

}

...

}

1. **Push to Repo and Trigger Pipeline:**
   * Commit and push the Terraform configuration.
   * The Azure Pipeline triggers automatically.
2. **Approval and Deployment:**
   * Approval from Abdul Hamed is required.
   * After approval, the application gateways are created in Azure.

**5. Creating Public IPs**

1. **Define Public IPs:**
   * Create Terraform configurations for public IP addresses.
   * Example:

hcl

Copy code

resource "azurerm\_public\_ip" "example" {

name = "example-pip"

location = azurerm\_resource\_group.example.location

resource\_group\_name = azurerm\_resource\_group.example.name

allocation\_method = "Dynamic"

}

1. **Push to Repo and Trigger Pipeline:**
   * Commit and push the Terraform configuration.
   * The Azure Pipeline triggers automatically.
2. **Approval and Deployment:**
   * Approval from Abdul Hamed is required.
   * After approval, the public IPs are created in Azure.

**6. Creating Load Balancers**

1. **Define Load Balancers:**
   * Create Terraform configurations for external and internal load balancers.
   * Example:

hcl

Copy code

resource "azurerm\_lb" "example" {

name = "example-lb"

location = azurerm\_resource\_group.example.location

resource\_group\_name = azurerm\_resource\_group.example.name

frontend\_ip\_configuration {

name = "example-frontend-ip"

public\_ip\_address\_id = azurerm\_public\_ip.example.id

}

...

}

1. **Push to Repo and Trigger Pipeline:**
   * Commit and push the Terraform configuration.
   * The Azure Pipeline triggers automatically.
2. **Approval and Deployment:**
   * Approval from Abdul Hamed is required.
   * After approval, the load balancers are created in Azure.

**7. Creating a Private DNS Zone**

1. **Define Private DNS Zone:**
   * Create a Terraform configuration for the private DNS zone.
   * Example:

hcl

Copy code

resource "azurerm\_private\_dns\_zone" "example" {

name = "example.private"

resource\_group\_name = azurerm\_resource\_group.example.name

}

1. **Push to Repo and Trigger Pipeline:**
   * Commit and push the Terraform configuration.
   * The Azure Pipeline triggers automatically.
2. **Approval and Deployment:**
   * Approval from Abdul Hamed is required.
   * After approval, the private DNS zone is created in Azure.

**8. Configuring VNet Peering**

1. **Define VNet Peering:**
   * Create Terraform configurations for VNet peering.
   * Example:

hcl

Copy code

resource "azurerm\_virtual\_network\_peering" "example" {

name = "example-vnet-peering"

resource\_group\_name = azurerm\_resource\_group.example.name

virtual\_network\_name = azurerm\_virtual\_network.example.name

remote\_virtual\_network\_id = azurerm\_virtual\_network.remote.id

}

1. **Push to Repo and Trigger Pipeline:**
   * Commit and push the Terraform configuration.
   * The Azure Pipeline triggers automatically.
2. **Approval and Deployment:**
   * Approval from Abdul Hamed is required.
   * After approval, the VNet peering is configured in Azure.

**9. Azure Site Recovery Setup**

1. **Create Recovery Vault:**
   * Create an Azure Recovery Vault using the Azure GUI and configure it.
2. **Define Storage Account:**
   * Create a Terraform configuration for the storage account.
   * Example:

hcl

Copy code

resource "azurerm\_storage\_account" "example" {

name = "examplestorageacct"

resource\_group\_name = azurerm\_resource\_group.example.name

location = azurerm\_resource\_group.example.location

account\_tier = "Standard"

account\_replication\_type = "LRS"

}

1. **Enable Hyper-V Site Replication:**
   * Configure replication for the Hyper-V site using the Recovery Vault.

**10. Storing Terraform Code in Azure Repos**

1. **Version Control:**
   * Store all Terraform configuration files in Azure Repos.
   * Use Git for version control to manage changes in the code.

**11. Setting Up Azure Pipeline for Terraform**

1. **Pipeline Configuration:**
   * Configure the Azure Pipeline to use the Terraform state file backend.
   * Specify the Azure subscription and resource details.
   * Ensure Abdul Hamed is set as the approver for critical changes.

**12. PowerShell Script for DNS Record Creation**

1. **Create PowerShell Script:**
   * Write a PowerShell script to automate DNS record creation using VM names and IPs.
   * Example:

powershell

Copy code

$dnsZoneName = "example.private"

$resourceGroupName = "example-resources"

$records = Import-Csv "vm-details.csv"

foreach ($record in $records) {

$vmName = $record.'SERVER NAME'

$ipAddress = $record.IP

New-AzPrivateDnsRecordSetA -ResourceGroupName $resourceGroupName -ZoneName $dnsZoneName -Name $vmName -IPv4Address $ipAddress

}

1. **Run Script:**
   * Execute the PowerShell script to update the DNS records in the private DNS zone.