Azure Virtual Machine

Expected Outcome

In this challenge, you will create a Azure Virtual Machine running Windows Server.

You will gradually add Terraform configuration to build all the resources needed to be able to login to the Azure Virtual Machine.

The resources you will use in this challenge:

- Resource Group
- Virtual Network
- Subnet
- Network Interface
- Virtual Machine
- Public IP Address

How to

Connect with Service Principal

Should you have more than one Subscription, you can specify the Subscription to use via the following command:

```
$ az account set --subscription="SUBSCRIPTION_ID"
```

We can now create the Service Principal which will have permissions to manage resources in the specified Subscription using the following command:

```
$ az ad sp create-for-rbac --role="Contributor" --
scopes="/subscriptions/SUBSCRIPTION_ID"
```

This command will output 5 values:

```
{
  "appId": "00000000-0000-0000-0000-000000000000", #client_id
  "displayName": "azure-cli-2017-06-05-10-41-15",
  "name": "http://azure-cli-2017-06-05-10-41-15",
  "password": "0000-0000-0000-0000-000000000000",  # client_secret
  "tenant": "00000000-0000-0000-0000-000000000000" # tenant_id
}
```

Replace the provider as follows:

```
provider "azurerm" {
  # Whilst version is optional, we /strongly recommend/ using it to pin the version
of the Provider being used
  version = "=1.38.0"

  subscription_id = "00000000-0000-0000-0000-000000000000"
  client_id       = "00000000-0000-0000-0000-000000000000"
  client_secret   = "${var.client_secret}"
  tenant_id       = "00000000-0000-0000-0000-000000000000"
}
```

When storing the credentials as Environment Variables, for example:

```
$ export ARM_CLIENT_ID="00000000-0000-0000-0000-000000000000"
$ export ARM_CLIENT_SECRET="00000000-0000-0000-0000-000000000000"
$ export ARM_SUBSCRIPTION_ID="00000000-0000-0000-0000-000000000000"
$ export ARM_TENANT_ID="00000000-0000-0000-0000-000000000000"
```

Create the base Terraform Configuration

Change directory into a folder specific to this challenge.

For example: `cd solution/`.

We will start with a few of the basic resources needed.

Create a `main.tf` file to hold our configuration.

Create Variables

Create a few variables that will help keep our code clean:

```
variable "prefix" {}

variable "location" {}
```

Create a Resource Group

Now create a Resource Group to contain all of our infrastructure using the variables to interpolate the parameters:

```
resource "azurerm_resource_group" "main" {
  name     = "${var.prefix}-vm-rg"
  location = var.location
}
```

Create Virtual Networking

In order to create an Azure Virtual Machine we need to create a network in which to place it.

Create a Virtual Network and Subnet using a basic CIDR block to allocate an IP block:

```
resource "azurerm_virtual_network" "main" {
  name                = "${var.prefix}-vnet"
  address_space       = ["10.0.0.0/16"]
  location            = azurerm_resource_group.main.location
  resource_group_name = azurerm_resource_group.main.name
}

resource "azurerm_subnet" "main" {
  name                 = "${var.prefix}-subnet"
  resource_group_name  = azurerm_resource_group.main.name
  virtual_network_name = azurerm_virtual_network.main.name
  address_prefix       = "10.0.1.0/24"
}
```

> Notice that we use the available metadata from the `azurerm_resource_group.main` resource to populate the parameters of other resources.

Pass in Variables

Create a file called 'terraform.tfvars' and add the folowing variables:

```
prefix   = ""
location = ""
```

> Note: Be sure to add a unique prefix, and an appropriate value for location. Do you know where to find valid 'locations'?

Run Terraform Workflow

Run `terraform init` since this is the first time we are running Terraform from this directory.

Run `terraform plan` where you should see the plan of two new resources, namely the Resource Group and the Virtual Network.

▶ View Output

```
$ terraform plan

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # azurerm_resource_group.main will be created
  + resource "azurerm_resource_group" "main" {
      + id       = (known after apply)
      + location = "centralus"
      + name     = "PREFIX-vm-rg"
      + tags     = (known after apply)
    }

  # azurerm_subnet.main will be created
  + resource "azurerm_subnet" "main" {
      + address_prefix       = "10.0.1.0/24"
      + id                   = (known after apply)
      + ip_configurations    = (known after apply)
      + name                 = "PREFIX-vm-subnet"
      + resource_group_name  = "PREFIX-vm-rg"
      + virtual_network_name = "PREFIX-vm-vnet"
    }

  # azurerm_virtual_network.main will be created
  + resource "azurerm_virtual_network" "main" {
      + address_space       = [
          + "10.0.0.0/16",
        ]
      + id                  = (known after apply)
      + location            = "centralus"
      + name                = "PREFIX-vm-vnet"
      + resource_group_name = "PREFIX-vm-rg"
      + tags                = (known after apply)

      + subnet {
          + address_prefix = (known after apply)
          + id             = (known after apply)
```

```
          + name           = (known after apply)
          + security_group = (known after apply)
        }
    }

Plan: 3 to add, 0 to change, 0 to destroy.
```

If your plan looks good, go ahead and run `terraform apply` and type "yes" to confirm you want to apply. When it completes you should see:

```
Apply complete! Resources: 3 added, 0 changed, 0 destroyed.
```

Create the Linux Azure Virtual Machine

Now that we have base networking in place, we will add a Network Interface and Virtual Machine. We will create a VM with an Azure Marketplace Image for Ubuntu 18.04.

Create the Network Interface resource:

```
resource "azurerm_network_interface" "linux" {
  name                = "${var.prefix}-linuxnic"
  location            = azurerm_resource_group.main.location
  resource_group_name = azurerm_resource_group.main.name

  ip_configuration {
    name                          = "config1"
    subnet_id                     = azurerm_subnet.main.id
    private_ip_address_allocation = "dynamic"
  }
}
```

Create the Virtual Machine resource:

```
resource "azurerm_virtual_machine" "linux" {
  name                  = "${var.prefix}-linuxvm"
  location              = azurerm_resource_group.main.location
  resource_group_name   = azurerm_resource_group.main.name
  network_interface_ids = [azurerm_network_interface.linux.id]
  vm_size               = "Standard_A2_v2"

  storage_os_disk {
    name              = "${var.prefix}linuxvm-osdisk"
    caching           = "ReadWrite"
    create_option     = "FromImage"
    managed_disk_type = "Standard_LRS"
  }

  os_profile {
    computer_name  = "${var.prefix}linuxvm"
    admin_username = "testadmin"
    admin_password = "Password1234!"
  }

  storage_image_reference {
    publisher = "Canonical"
    offer     = "UbuntuServer"
    sku       = "18.04-LTS"
    version   = "latest"
```

```
  }
  os_profile_linux_config {
    disable_password_authentication = false
  }
}
```

Take note of the OS image:

```
  storage_image_reference {
    publisher = "Canonical"
    offer     = "UbuntuServer"
    sku       = "18.04-LTS"
    version   = "latest"
  }
```

Run a plan and apply to create both these resources.

Add a Public IP

At this point you should have a running Virtual Machine in Azure running Linux Server, however you have no way to access it. To do this we must do two things, create the Public IP Resource and configure the Network Interface to use it.

Create a Public IP Address that will assign an IP address:

```
resource "azurerm_public_ip" "linux" {
  name                = "${var.prefix}-linux-pubip"
  location            = azurerm_resource_group.main.location
  resource_group_name = azurerm_resource_group.main.name
  allocation_method   = "Static"
}
```

Update the IP Configuration parameter of the Network Interface to attach the Public IP:

```
resource "azurerm_network_interface" "linux" {
  ...

  ip_configuration {
    ...
    public_ip_address_id  = azurerm_public_ip.linux.id
  }
}
```

Terraform Plan

Running `terraform plan` should contain something like the following:

```
# azurerm_network_interface.main will be updated in-place
  ~ resource "azurerm_network_interface" "linux" {
        ...

      ~ ip_configuration {
            ...
          + public_ip_address_id                          = (known after apply)
        }
    }
```

```
  # azurerm_public_ip.main will be created
  + resource "azurerm_public_ip" "linux" {
      + allocation_method            = "Static"
      + fqdn                         = (known after apply)
      + id                           = (known after apply)
      + idle_timeout_in_minutes      = 4
      + ip_address                   = (known after apply)
      + ip_version                   = "IPv4"
      + location                     = "centralus"
      + name                         = "PREFIX-linuxvm-pubip"
      + public_ip_address_allocation = (known after apply)
      + resource_group_name          = "PREFIX-vm-rg"
      + sku                          = "Basic"
      + tags                         = (known after apply)
    }

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

> Notice that there is a new resource being added and one being updated.

Run `terraform apply` to apply the changes.

Outputs

You now have all the infrastructure in place and can now SSH into the Linux Server VM we just stood up.

But wait, the Public IP was dynamically created, how do I access it?

You could check the value in the Azure Portal, however let's instead add an output to get that information.

Add the following output:

```
output "linux-private-ip" {
  value       = azurerm_network_interface.linux.private_ip_address
  description = "Linux Private IP Address"
}

output "linux-public-ip" {
  value       = azurerm_public_ip.linux.ip_address
  description = "Linux Public IP Address"
}
```

Now run a `terraform refresh`, which will refresh your state file with the real-world infrastructure and resolve the new outputs you just created.

The output should look similar to this:

```
$ terraform refresh
azurerm_resource_group.main: Refreshing state... (ID:
/subscriptions/.../resourceGroups/challenge03-rg)
azurerm_virtual_network.main: Refreshing state... (ID:
/subscriptions/.../virtualNetworks/challenge03-vnet)
azurerm_public_ip.main: Refreshing state... (ID:
/subscriptions/.../publicIPAddresses/challenge03-pubip)
azurerm_subnet.main: Refreshing state... (ID:
/subscriptions/.../subnets/challenge03-subnet)
azurerm_network_interface.main: Refreshing state... (ID:
/subscriptions/.../networkInterfaces/challenge03-nic)
azurerm_virtual_machine.main: Refreshing state... (ID:
/subscriptions/.../virtualMachines/challenge03-vm)
```

```
Outputs:

private-ip = 10.0.1.4
public-ip = 168.61.55.117
```

> Note: you can also run `terraform output` to see just these outputs without having to run refresh again.

Connect (optional)

Using the Public IP output value, SSH into the VM using the username and password created.

Success! You have now stood up a Virtual Machine in Azure using Terraform!

Windows VM (extra credit)

Add the Terraform needed to create another VM to the same subnet, but using a Windows Base image. This will require a slightly different `azurerm_virtual_machine` resource block, can you determine what needs to change based on the online docs?

Hint:

```
publisher = "MicrosoftWindowsServer"
offer     = "WindowsServer"
sku       = "2016-Datacenter"
version   = "latest"
```

Scaling the Virtual Machine (extra credit)

Utilize the count meta arguemnt to allow for the scaling of VM's.

Clean up

When you are done, run `terraform destroy` to remove everything we created.

Advanced areas to explore

1. Extract secrets into required variables.
2. Add a data disk.
3. Add a DNS Label to the Public IP Address.
4. Search for Marketplace Images. (hint: use the Azurel CLI and start with `az vm image -h`)

Resources

- Azure Resource Group
- Azure Virtual Network
- Azure Subnet
- Azure Network Interface
- Azure Virtual Machine
- Public IP Address