**Create a VM Scale Set in Azure with Terraform**

**Introduction**

In this lab, we will cover these 4 objectives:

1. First, we will log into the Azure Portal, configure the Cloud Shell, and download and run the lab setup script to setup the lab.
2. Second, we will import the resource group.
3. Third, we will add our VM scale set to our configuration and set CPU thresholds to allow for autoscaling.
4. And for the fourth objective, we will add a jumpbox to our configuration to give us a server to connect to that we can use to connect to our cluster. We will then deploy our VM cluster resources.

**Solution**

**Set Up Cloud Shell and the Lab Environment**

1. In the Azure Portal, click the Cloud Shell icon (>\_) in the upper right. Take note of the **Location** for the lab environment.
2. Select **Bash**.
3. Click **Show advanced settings**.
4. If needed, change the **Cloud Shell region** to match the **Location** of the lab environment.
5. For **Storage account**, select **Use existing**.
6. For **File share**, select **Create new** and give it a name of *terraform*.
7. Click **Create storage**.

**Import the Resource Group into Terraform**

1. Using az group list, find and copy everything within the quotes for the subscription id:

az group list

1. Using the resource name and label from the networking.tf file, as well as the subscription id that was just copied, import the Azure resource group:

terraform import azurerm\_resource\_group.comics <SUBSCRIPTION\_ID>

1. Open the networking.tf file:

vim networking.tf

1. In the Azure Portal, copy the **Resource group** name.
2. In the Cloud Shell, on line 2 of the networking.tf file, uncomment the line by deleting the # and replace <RESOURCE\_GROUP\_NAME> with the **Resource group** name that was just copied.
3. On line 3, uncomment the line by deleting the # and replace <RESOURCE\_GROUP\_LOCATION> with the location for the lab environment, using westus for West US, centralus for Central US, and so on.
4. Save and exit the file:

ESC :wq!

**Define the Azure VM Scale Set**

1. Create a new file named vmss.tf:

vim vmss.tf

1. In the file, paste the following configuration:

resource "azurerm\_virtual\_machine\_scale\_set" "comics" {

name = "vmscaleset"

location = var.location

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

upgrade\_policy\_mode = "Manual"

sku {

name = "Standard\_DS1\_v2"

tier = "Standard"

capacity = 2

}

storage\_profile\_image\_reference {

publisher = "Canonical"

offer = "UbuntuServer"

sku = "16.04-LTS"

version = "latest"

}

storage\_profile\_os\_disk {

name = ""

caching = "ReadWrite"

create\_option = "FromImage"

managed\_disk\_type = "Standard\_LRS"

}

storage\_profile\_data\_disk {

lun = 0

caching = "ReadWrite"

create\_option = "Empty"

disk\_size\_gb = 10

}

os\_profile {

computer\_name\_prefix = "vmlab"

admin\_username = var.admin\_user

admin\_password = var.admin\_password

custom\_data = file("./web.conf")

}

os\_profile\_linux\_config {

disable\_password\_authentication = false

}

network\_profile {

name = "terraformnetworkprofile"

primary = true

ip\_configuration {

name = "IPConfiguration"

subnet\_id = azurerm\_subnet.comics.id

load\_balancer\_backend\_address\_pool\_ids = [azurerm\_lb\_backend\_address\_pool.bpepool.id]

primary = true

}

}

tags = var.tags

}

1. Save and exit the file:

ESC :wq!

**Define the Jumpbox Configuration**

1. Create a new file named vmss.tf:

vim jumpbox.tf

1. In the file, paste the following configuration:

resource "azurerm\_public\_ip" "jumpbox" {

name = "jumpbox-public-ip"

location = var.location

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

allocation\_method = "Static"

domain\_name\_label = "${random\_string.fqdn.result}-ssh"

tags = var.tags

}

resource "azurerm\_network\_interface" "jumpbox" {

name = "jumpbox-nic"

location = var.location

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

ip\_configuration {

name = "IPConfiguration"

subnet\_id = azurerm\_subnet.comics.id

private\_ip\_address\_allocation = "dynamic"

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

}

tags = var.tags

}

resource "azurerm\_virtual\_machine" "jumpbox" {

name = "jumpbox"

location = var.location

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

network\_interface\_ids = [azurerm\_network\_interface.jumpbox.id]

vm\_size = "Standard\_DS1\_v2"

storage\_image\_reference {

publisher = "Canonical"

offer = "UbuntuServer"

sku = "16.04-LTS"

version = "latest"

}

storage\_os\_disk {

name = "jumpbox-osdisk"

caching = "ReadWrite"

create\_option = "FromImage"

managed\_disk\_type = "Standard\_LRS"

}

os\_profile {

computer\_name = "jumpbox"

admin\_username = var.admin\_user

admin\_password = var.admin\_password

}

os\_profile\_linux\_config {

disable\_password\_authentication = false

}

tags = var.tags

}

1. Save and exit the file:

ESC :wq!

**Create the Variables and Outputs, then Deploy**

1. Create a new file named variables.tf:

vim variables.tf

1. In the file, paste the following:

variable "location" {

default = "southcentralus"

description = "Location where resources will be created"

}

variable "tags" {

description = "Map of the tags to use for the resources that are deployed"

type = map(string)

default = {

environment = "Staging"

}

}

variable "application\_port" {

description = "Port that you want to expose to the external load balancer"

default = 80

}

variable "admin\_user" {

description = "User name to use as the admin account on the VMs that will be part of the VM scale set"

default = "azureuser"

}

variable "admin\_password" {

description = "Default password for admin account"

}

1. Save and exit the file:

ESC :wq!

1. Create a new file named outputs.tf:

vim outputs.tf

1. In the file, paste the following:

output "vmss\_public\_ip\_fqdn" {

value = azurerm\_public\_ip.comics.fqdn

}

output "jumpbox\_public\_ip\_fqdn" {

value = azurerm\_public\_ip.jumpbox.fqdn

}

output "jumpbox\_public\_ip" {

value = azurerm\_public\_ip.jumpbox.ip\_address

}

1. And finally, create a new file named web.conf:

vim web.conf

1. In the file, paste the following:

#cloud-config packages: - nginx

1. Save and exit the file:

ESC :wq!

1. Format the files:

terraform fmt

1. Validate the syntax. When validating, you may receive an Argument is deprecated warning. This can be ignored:

terraform validate

1. Do a dry run of the configuration. When prompted, enter a new password to set the admin\_password variable:

terraform plan

1. Apply the configuration, using the admin\_password that was just set:

terraform apply

1. When prompted, enter yes to proceed.
2. Under Outputs, copy the vmss\_public\_ip\_fqdn output URL.
3. In a new tab, navigate to the URL and verify the cluster is up and running.