**Create a VM Cluster in Azure with Terraform**

**Introduction**

In this lab, you are asked to start building out an app server cluster for a comic reader that you are working on for your employer for digital comics. To do this, you will configure Cloud Shell and run the lab setup script; import the resource group; deploy the Azure Storage account configuration; and add the VM cluster configuration and deploy your cluster.

**Solution**

**Set Up Cloud Shell and Lab Environment**

1. Click the Cloud Shell icon (>\_) in the upper right.
2. Select **Bash**.
3. Click **Show advanced settings**, and set the following values:
   * **Cloud Shell region**: Select **Central US**.
   * **Storage account**: Select **Use existing**.
   * **File share**: Select **Create new**, and give it a name of *terraform*.
4. Click **Create storage**.

**Import the Resource Group**

1. Open and review providers.tf:

vim providers.tf

You'll see it has the required version of Terraform and required providers.

1. Exit the file by typing :q.
2. Open and review lb.tf:

vim lb.tf

You'll see it's a basic load balancer configuration.

1. Exit the file by typing :q.
2. Open and review networking.tf:

vim networking.tf

You'll see it has the networking configuration information.

1. Exit the file by typing :q.
2. Pull down the provider plugins:

terraform init

It will take a few minutes to finish running.

1. Get the subscription ID:

az group list

1. Copy the subscription ID (everything after "id" within quotation marksm, including the slash at the beginning) and paste it into a text file, as you'll need it next.
2. Import the resource group, replacing <SUBSCRIPTION\_ID> with the ID you just copied:

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

You should see a success message after a minute or so.

1. Open networking.tf:

vim networking.tf

1. Un-comment name and location by deleting the # before them.
2. Replace <RESOURCE\_GROUP> with your resource group name (you can copy it from the portal above Cloud Shell).
3. Replace <LOCATION> with centralus.
4. Save and quit the file by pressing **Escape** followed by :wq!.

**Define Your Cluster Config and Output Variables**

1. Create a file called vms.tf:

vim vms.tf

1. Add the following to the file:

resource "azurerm\_network\_interface" "comics" {

count = 2

name = "nic${count.index}"

location = azurerm\_resource\_group.comics.location

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

ip\_configuration {

name = "comicsConfiguration"

subnet\_id = azurerm\_subnet.comics.id

private\_ip\_address\_allocation = "dynamic"

}

}

resource "azurerm\_managed\_disk" "comics" {

count = 2

name = "datadisk\_existing\_${count.index}"

location = azurerm\_resource\_group.comics.location

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

storage\_account\_type = "Standard\_LRS"

create\_option = "Empty"

disk\_size\_gb = "1023"

}

resource "azurerm\_availability\_set" "comics" {

name = "avset"

location = azurerm\_resource\_group.comics.location

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

platform\_fault\_domain\_count = 2

platform\_update\_domain\_count = 2

managed = true

}

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

count = 2

name = "vm${count.index}"

location = azurerm\_resource\_group.comics.location

availability\_set\_id = azurerm\_availability\_set.comics.id

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

network\_interface\_ids = [element(azurerm\_network\_interface.comics.\*.id, count.index)]

vm\_size = "Standard\_DS1\_v2"

storage\_image\_reference {

publisher = "Canonical"

offer = "UbuntuServer"

sku = "16.04-LTS"

version = "latest"

}

storage\_os\_disk {

name = "myosdisk${count.index}"

caching = "ReadWrite"

create\_option = "FromImage"

managed\_disk\_type = "Standard\_LRS"

}

# Optional data disks

storage\_data\_disk {

name = "datadisk\_new\_${count.index}"

managed\_disk\_type = "Standard\_LRS"

create\_option = "Empty"

lun = 0

disk\_size\_gb = "1023"

}

storage\_data\_disk {

name = element(azurerm\_managed\_disk.comics.\*.name, count.index)

managed\_disk\_id = element(azurerm\_managed\_disk.comics.\*.id, count.index)

create\_option = "Attach"

lun = 1

disk\_size\_gb = element(azurerm\_managed\_disk.comics.\*.disk\_size\_gb, count.index)

}

os\_profile {

computer\_name = "hostname"

admin\_username = "comicsadmin"

admin\_password = "Password1234!"

}

os\_profile\_linux\_config {

disable\_password\_authentication = false

}

tags = {

environment = "staging"

}

}

1. Save and exit the file by pressing **Escape** followed by :wq!.
2. Create a file called output.tf:

vim output.tf

1. Add the following to the file:

output "resource\_group\_name" {

value = azurerm\_resource\_group.comics.name

}

output "vm0\_private\_ip" {

value = azurerm\_network\_interface.comics.0.private\_ip\_address

}

output "vm1\_private\_ip" {

value = azurerm\_network\_interface.comics.1.private\_ip\_address

}

output "public\_ip\_address" {

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

}

output "location" {

value = azurerm\_resource\_group.comics.location

}

1. Save and exit the file by pressing **Escape** followed by :wq!.

**Add the VM Cluster Terraform Config and Deploy**

1. Check the formatting of the files:

terraform fmt

1. Fix any formatting errors if they exist.
2. Check the code syntax of the files:

terraform validate

1. Preview the changes:

terraform plan

1. Deploy the resources:

terraform apply

1. Enter yes at the prompt. It will take a minute or so to finish after that point.
2. Note the output after it's successfully deployed the resources. It should include a public IP address and two private IP addresses.
3. List all the resources Terraform is currently managing:

terraform state list

You should see a list of resources now deployed.

1. In the Azure portal, navigate to **All resources**. There, you should see the same list of resources.