MS Azure Administrator Course End Project
Esther Jennings
2024-08-31

Course-end Project 2

## Project Agenda

Create Highly Available Architecture by Distributing Incoming Traffic among Healthy Service Instances in Cloud Services or Virtual Machines in a Load Balanced Set with the Help of Command-Line Interface

### Description

The Rand Enterprises Corporation wants to deploy a web application in a highly available environment so that only the healthy instances will be serving the traffic so end users will not be facing any downtime. They have decided to work on an Azure public load balancer to implement the functionality.

The operations team at Rand decides to define the entire architecture using the load balancer and its backend pool, once that's in place they intend to create the frontend IP and health probe along with virtual machines housing their application.

Rand Enterprises works extensively on delivering highly available web applications for their users in a secure way by avoiding directly exposing the virtual machines hosting the applications to the public internet. The communication from the application in the VM to the end-user must take place via the Load Balancer.

The expectation of the operation team is to create a reusable method that can be used for automation if in the future we need to deploy the same kind of infrastructure. So, rather than deploying resources in the Azure portal, they should leverage the command-line interface to deploy the resources so that in the future these commands can be used

As a security measure, you need to ensure that only the health instances of the virtual machine will be serving the traffic.

### Expected Deliverables

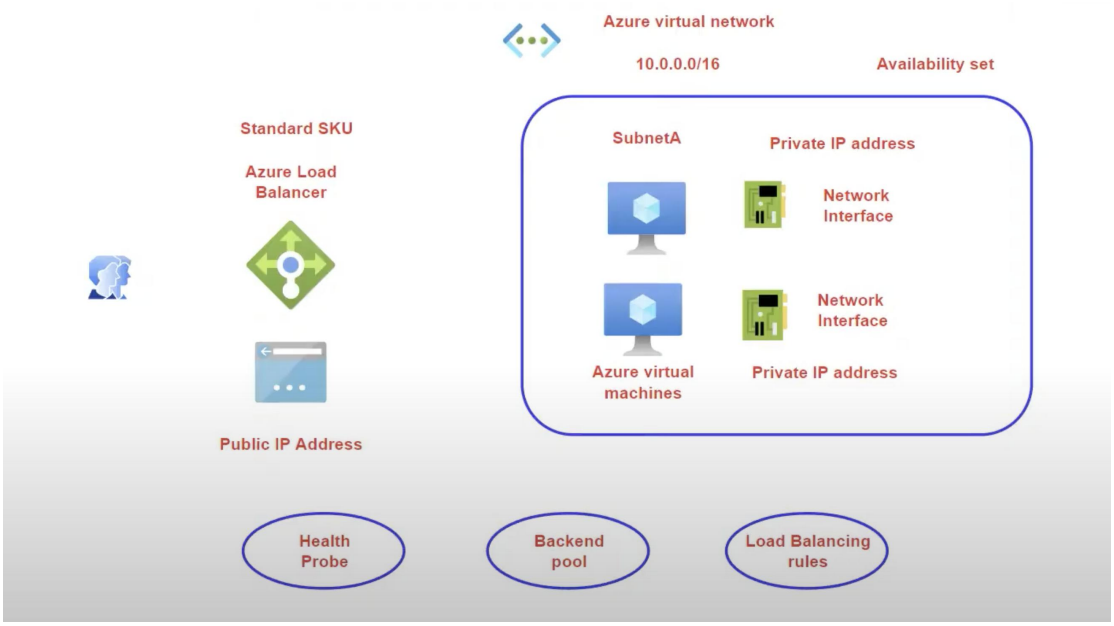
- Identify Virtual machines and Networking
- Configure the load balancer
- Extend the load balancer with backend pool and frontend IP
- Define the Health probe

## Solution

The following resources were created in addition to achieve 2. objective:
Network and compute resources:

- Create load balancer with public IP-Address and domain name in public subnet. The load balancer check the health of the assigned VMs server http-requests.
- Create scale set to manage private VMs with web-server in public subnet and avoid downtime of handling http-requests.
- Allow users to use http to access resources in public subnet
- Refactor Terraform commands into different files grouped by resource categories: iam, loadbalancer, network, and vm to be reused in future projects.

See Guide at Create a public load balacer using Terraform. The following figure shows the infrastructure we are setting up.




Terraform lets us define and deploy cloud infrastructure, in HCL syntax. The needed Terrafor source code files are the following:

1. Providers.tf: this specifies the cloud provider as Microsoft Azure
2. Main.tf: this files creates the resources needed for a public load balancer.
   a) Create resource group
   b) Create virtual network with address space 10.0.0.0/16
   c) Create subnet in the virtual network with address prefix 10.0.1.0/24
   d) hCreate network security group
   e) Create public IP (to be used by public load balancer)
   f)Create network interface (NIC)
   g) Create virtual machine(s), create 2 Linux servers to serve content.
   h) Enable virtual machine extension to install Nginx (web server for Linux)

i) Create public load balancer (this is the front-end)

3. Variables.tf: use this file to specify resource location, username, virtual network name, subnet name, public ip name, security group name, network interface name, virtual machine name, virtual machine size, load balancer name. In this way, main.tf is reusable for future projects. We configure:

a) Add both VMs to backend pool

b) Add heath probe (to probe the backend pool)

c) Specify load balancing rules

4. Outputs.tf: this specifies the URL users an specify to access the web content to be served by the web server.

### Step 1. Initialize Terraform

Log into Microsoft Azure account. Bring up a command console and enter the command:

Terraform init -upgrade

This command downloads he Azure provider required to manage our Azure resources.

### Step 2. Create a Terraform Execution Plan

To create an execution plan, use the command:

Terraform plan -out main.tfplan

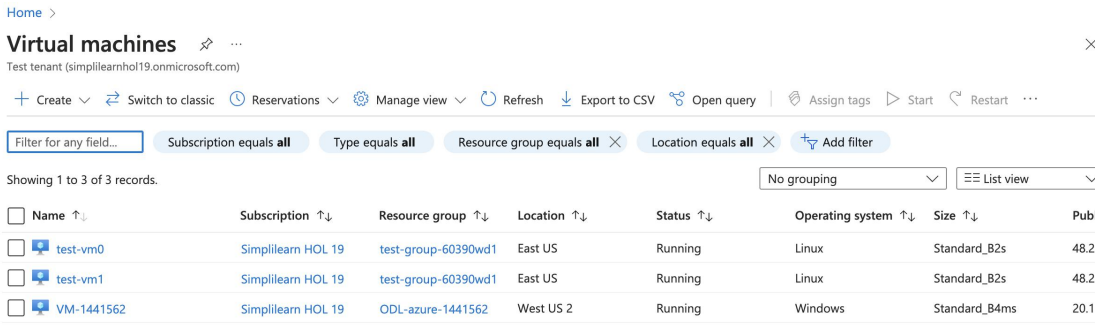
### Step 3. Apply a Terraform Execution Plan

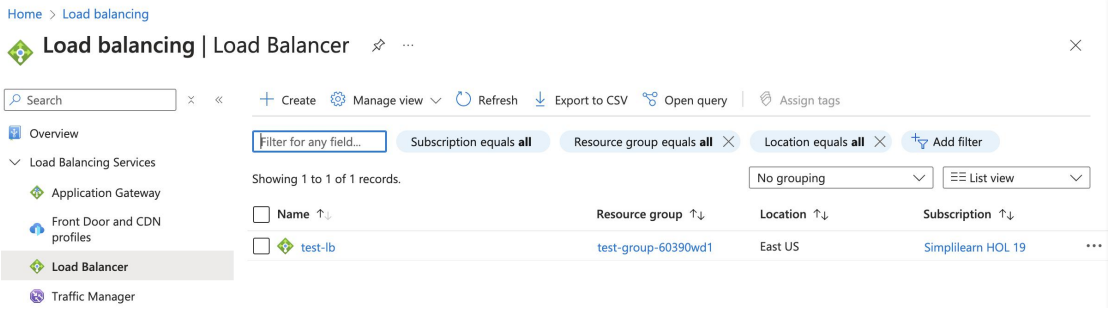
To apply the execution plan, do:

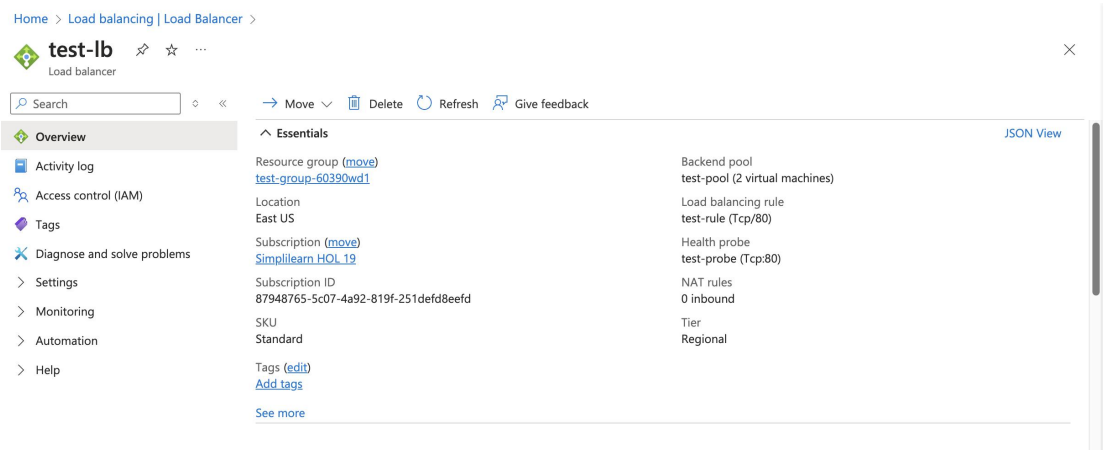
Terraform apply main.tfplan

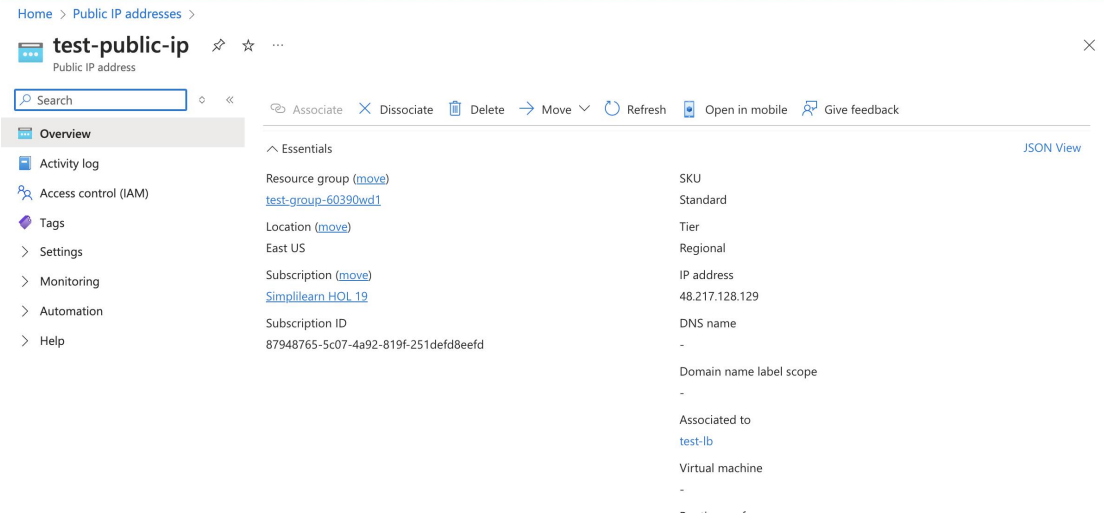
```
-251defd8eefd/resourceGroups/test-group-60390wd1/providers/Microsoft.Comput
azurerm_virtual_machine_extension.my_vm_extension[1]: Still creating... [1m
azurerm_virtual_machine_extension.my_vm_extension[1]: Still creating... [1m
azurerm_virtual_machine_extension.my_vm_extension[1]: Still creating... [1m
azurerm_virtual_machine_extension.my_vm_extension[1]: Creation complete aft
f-251defd8eefd/resourceGroups/test-group-60390wd1/providers/Microsoft.Compu

Apply complete! Resources: 20 added, 0 changed, 0 destroyed.

Outputs:

public_ip_address = "http://48.217.128.129"
```

Verify that the resources have been created.


Home
Virtual machines
Test tenant (simplilearnhol19.onmicrosoft.com)
Create
Switch to classic
Reservations
Manage view
Refresh
Export to CSV
Open query
Assign tags
Start
Restart
Filter for any field...
Subscription equals all
Type equals all
Resource group equals all
Location equals all
Add filter
Showing 1 to 3 of 3 records.
No grouping
List view
Name
Subscription
Resource group
Location
Status
Operating system
Size
Pub
test-vm0 Simplilearn HOL 19 test-group-60390wd1 East US Running Linux Standard_B2s 48.2
test-vm1 Simplilearn HOL 19 test-group-60390wd1 East US Running Linux Standard_B2s 48.2
VM-1441562 Simplilearn HOL 19 ODL-azure-1441562 West US 2 Running Windows Standard_B4ms 20.1


Home
Load balancing
Load balancing | Load Balancer
Search
Overview
Load Balancing Services
Application Gateway
Front Door and CDN profiles
Load Balancer
Traffic Manager
Create
Manage view
Refresh
Export to CSV
Open query
Assign tags
Filter for any field...
Subscription equals all
Resource group equals all
Location equals all
Add filter
Showing 1 to 1 of 1 records.
No grouping
List view
Name
Resource group
Location
Subscription
test-lb test-group-60390wd1 East US Simplilearn HOL 19


Home
Load balancing | Load Balancer
test-lb
Load balancer
Search
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Settings
Monitoring
Automation
Help
Move
Delete
Refresh
Give feedback
Essentials
JSON View
Resource group (move)
test-group-60390wd1
Location
East US
Subscription (move)
Simplilearn HOL 19
Subscription ID
87948765-5c07-4a92-819f-251defd8eefd
SKU
Standard
Tags (edit)
Add tags
See more
Backend pool
test-pool (2 virtual machines)
Load balancing rule
test-rule (Tcp/80)
Health probe
test-probe (Tcp:80)
NAT rules
0 inbound
Tier
Regional

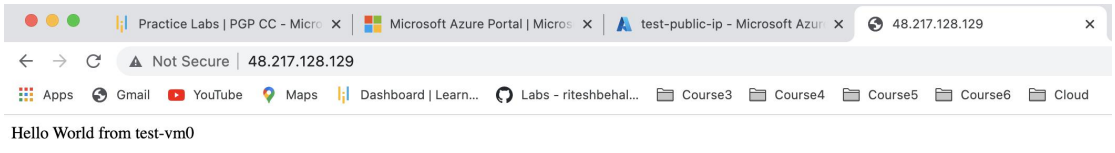

Home
Public IP addresses
test-public-ip
Public IP address
Search
Overview
Activity log
Access control (IAM)
Tags
Settings
Monitoring
Automation
Help
Associate
Dissociate
Delete
Move
Refresh
Open in mobile
Give feedback
Essentials
JSON View
Resource group (move)
test-group-60390wd1
Location (move)
East US
Subscription (move)
Simplilearn HOL 19
Subscription ID
87948765-5c07-4a92-819f-251defd8eefd
SKU
Standard
Tier
Regional
IP address
48.217.128.129
DNS name
-
Domain name label scope
-
Associated to
test-lb
Virtual machine
-

Verify the public IP address.

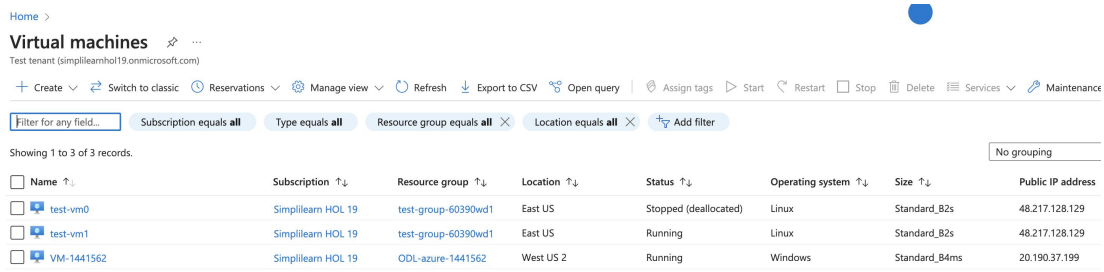
```
public_ip_address = "http://48.217.128.129"
odl_user [ ~ ]$ echo $(terraform output -raw public_ip_address)
http://48.217.128.129
odl_user [ ~ ]$
```

## Step 4. Test the Load Balancer

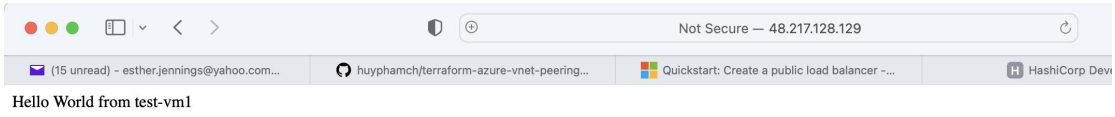
Use a different browser (than the one used for MS Azure). Go to the URL above.




I refreshed this a few times, but it seens that the load balancer is either preferring vm0. Next, I stopped vm0.




Trying again shows that load balancer is now serving the content from vm1.




Restarting vm0 again and re-trying, the load balancer is serving the content from vm0 again.

## Step 5. Clean up -- create a plan to destroy resources

The command is: terraform plan -destroy -out main.destroy.tfplan

## Step 6. Clean up -- execute the destroy resource plan

The command is: terraform apply main.destroy.tfplan

Verify that the resources have been destroyed.

--- end of project.