**Course-end Project 2**

Implement, manage, and monitor on Azure environment

David Ahn

# Problem Statement

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.

# Proposed Solution

Per the requirements above I prose the following solution:

* The web servers will be placed in a private subnet behind a load balancer to secure the web servers in the backend pool from unauthorized access from the Internet
* A NAT Gateway will be implemented to allow outbound access to the Internet from the application subnet
* To further secure the architecture, the web servers will only be accessible via a bastion host

# Architecture Provisioning

* As per the requirements, all resource provisioning was performed using the Azure CLI from within Cloud Shell

## Virtual Network

* Create a resource group named RandWebApp-rg in the EastUS region
* Create virtual network named RandVNet with subnet prefix of 10.1.0.0/16 and Application Subnet with subnet prefix of 10.1.0.0/24
* Create a public IP to be used for the load balancer

## Load Balancer

* Create load balancer named RandLoadBalancer
* Create health probe called HTTPHealthProbe to ensure traffic only gets load balanced between healthy VMs in the backend pool on port 80
* Create load balancer rule allowing TCP port 80 traffic through the load balancer to the backend poolcle

## Network Security Group

* Create NSG named BackendPoolNSG for the VMs in the load balancer backend pool
* Create an inbound rule allowing HTTP port 80 trafficclear

## Bastion

* Create public IP for bastion host
* Create subnet for bastion called AzureBastionSubnet with subnet prefix of 10.1.1.0/27
* Deploy bastion host

## Backend Servers

* Create NICs for the 2 VMs
* Deploy 2 VMS (VM1, VM2)
* Add the VMs to the LBBackendPool of the load balancer
* Install IIS

## NAT Gateway

* Create NAT Gateway IP (NATgatewayIP)
* Create NAT Gateway (NATgateway)
* Associate NAT gateway with ApplicationSubnet

# Testing

* Successfully accessed public IP of load balancer and verified traffic was being routed to VM1 and VM2
* Verified that I cannot RDP to VM1 and VM2 directly
* Verified that I can RDP to VM1 and VM2 via Azure Bastion