Course-End Project 2

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

Steps to be followed:

1. Creating a Resource group
2. Configuring a virtual network
3. Creating backend Servers
4. Creating a public IP address
5. Creating a Load Balancer
6. Creating an Outbound Rule Configuration
7. Installing IIS
8. Testing the Load Balancer

Step 1: Creating a Resource group

1.1 Launch Azure Cloud shell **Bash terminal**

1.2 Create a resource group

```
az group create \
  --name lbResourceGroup \
  --location eastus
```

```
odl_user@Azure:~$ az group create \
>     --name lbResourceGroup \
>     --location eastus
{
  "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup",
  "location": "eastus",
  "managedBy": null,
  "name": "lbResourceGroup",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null,
  "type": "Microsoft.Resources/resourceGroups"
}
odl_user@Azure:~$
```

Step 2: Configuring a virtual network

2.1 Create a virtual network using **az network vnet create**

```
az network vnet create \
  --resource-group lbResourceGroup \
  --location eastus \
  --name myVNet \
  --address-prefixes 10.1.0.0/16 \
  --subnet-name myBackendSubnet \
  --subnet-prefixes 10.1.0.0/24
```

```
odl_user@Azure:~$ az network vnet create \
>     --resource-group lbResourceGroup \
>     --location eastus \
>     --name myVNet \
>     --address-prefixes 10.1.0.0/16 \
>     --subnet-name myBackendSubnet \
>     --subnet-prefixes 10.1.0.0/24
{
  "newVNet": {
    "addressSpace": {
      "addressPrefixes": [
        "10.1.0.0/16"
      ]
    },
    "bgpCommunities": null,
    "ddosProtectionPlan": null,
    "dhcpOptions": {
      "dnsServers": []
    },
    "enableDdosProtection": false,
    "enableVmProtection": null,
    "encryption": null,
    "etag": "W/\"adc00b27-c009-4ee2-ab0e-36a31713c03b\"",
    "extendedLocation": null,
    "flowTimeoutInMinutes": null,
```

2.2 Create a public IP address for a bastion host

```
az network public-ip create \
  --resource-group lbResourceGroup \
  --name myBastionIP \
  --sku Standard
```

```
odl_user@Azure:~$ az network public-ip create \
>     --resource-group lbResourceGroup \
>     --name myBastionIP \
>     --sku Standard
[Coming breaking change] In the coming release, the default behavior will be changed as follows when sku is Standa
ons, you will get a zone-redundant IP indicated by zones:["1","2","3"]; For non-zonal regions, you will get a non
{
  "publicIp": {
    "ddosSettings": null,
    "deleteOption": null,
    "dnsSettings": null,
    "etag": "W/\"bd245108-7f10-4cb7-a843-d59a6b40392e\"",
    "extendedLocation": null,
    "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.
    "idleTimeoutInMinutes": 4,
    "ipAddress": "20.124.176.208",
    "ipConfiguration": null,
    "ipTags": [],
    "linkedPublicIpAddress": null,
    "location": "eastus",
    "migrationPhase": null,
    "name": "myBastionIP",
    "natGateway": null,
```

2.3 Create a bastion subnet

```
az network vnet subnet create \
  --resource-group lbResourceGroup \
  --name AzureBastionSubnet \
  --vnet-name myVNet \
  --address-prefixes 10.1.1.0/24
```

```
odl_user@Azure:~$ az network vnet subnet create \
>     --resource-group lbResourceGroup \
>     --name AzureBastionSubnet \
>     --vnet-name myVNet \
>     --address-prefixes 10.1.1.0/24
{
  "addressPrefix": "10.1.1.0/24",
  "addressPrefixes": null,
  "applicationGatewayIpConfigurations": null,
  "delegations": [],
  "etag": "W/\"8c65e5a6-8d7a-4ba0-9cf6-282a0c9d7366\"",
  "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/p
astionSubnet",
  "ipAllocations": null,
  "ipConfigurationProfiles": null,
  "ipConfigurations": null,
  "name": "AzureBastionSubnet",
  "natGateway": null,
  "networkSecurityGroup": null,
  "privateEndpointNetworkPolicies": "Enabled",
  "privateEndpoints": null,
  "privateLinkServiceNetworkPolicies": "Enabled",
  "provisioningState": "Succeeded",
  "purpose": null,
```

2.4 Create a bastion host

```
az network bastion create \
  --resource-group lbResourceGroup \
  --name myBastionHost \
  --public-ip-address myBastionIP \
  --vnet-name myVNet \
  --location eastus
```

```
odl_user@Azure:~$ az network bastion create \
>     --resource-group lbResourceGroup \
>     --name myBastionHost \
>     --public-ip-address myBastionIP \
>     --vnet-name myVNet \
>     --location eastus
Command group 'network bastion' is in preview and under development. Reference and support levels: https://aka.ms/CLI
{
  "disableCopyPaste": false,
  "dnsName": "bst-ae234116-88f0-4ce5-93b2-fe4448ddd4a6.bastion.azure.com",
  "enableFileCopy": false,
  "enableIpConnect": false,
  "enableShareableLink": false,
  "enableTunneling": false,
  "etag": "W/\"d3b8370e-3532-4cc2-ae35-cfe86ccc9258\"",
  "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.Netwo
  "ipConfigurations": [
    {
      "etag": "W/\"d3b8370e-3532-4cc2-ae35-cfe86ccc9258\"",
      "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.N
ations/bastion_ip_config",
      "name": "bastion_ip_config",
      "privateIpAllocationMethod": "Dynamic",
      "provisioningState": "Succeeded",
      "publicIpAddress": {
```

2.5 Create a network security group

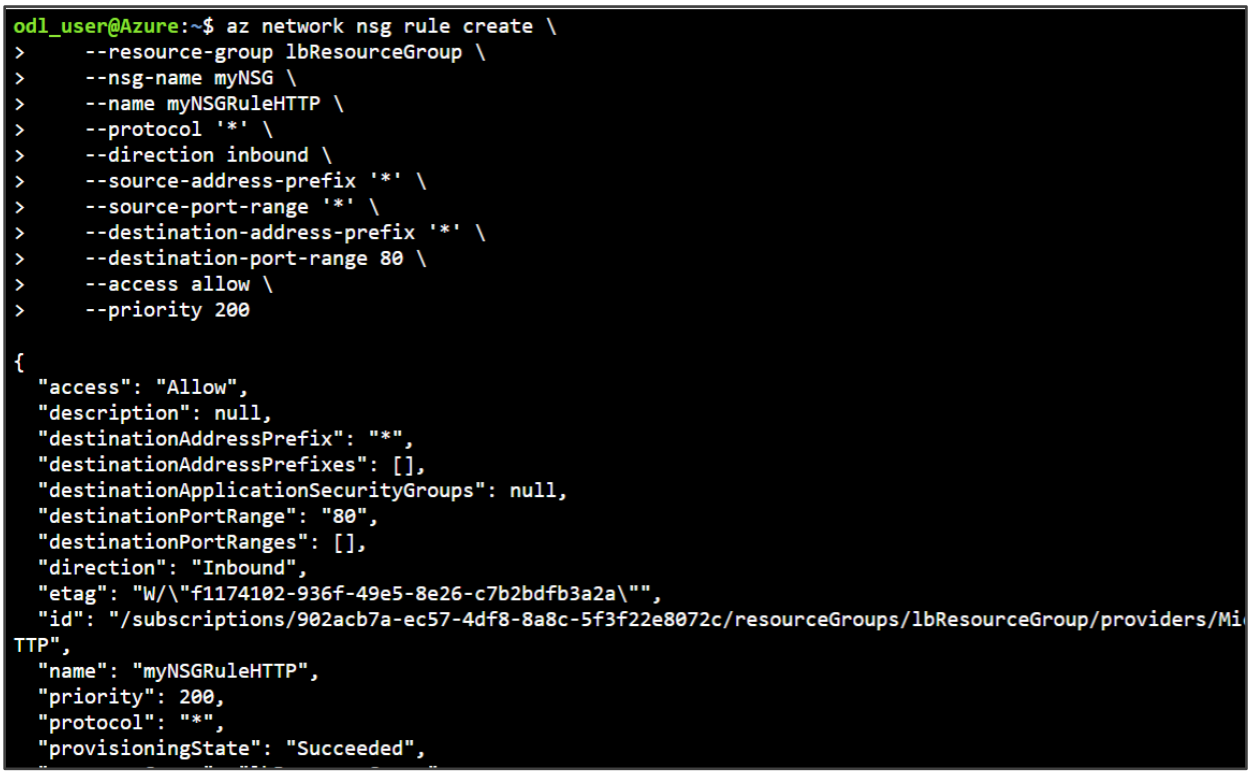
```
az network nsg create \
  --resource-group lbResourceGroup\
  --name myNSG
```

```
odl_user@Azure:~$ az network nsg create \
>     --resource-group lbResourceGroup\
>     --name myNSG

{
  "NewNSG": {
    "defaultSecurityRules": [
      {
        "access": "Allow",
        "description": "Allow inbound traffic from all VMs in VNET",
        "destinationAddressPrefix": "VirtualNetwork",
        "destinationAddressPrefixes": [],
        "destinationApplicationSecurityGroups": null,
        "destinationPortRange": "*",
        "destinationPortRanges": [],
        "direction": "Inbound",
        "etag": "W/\"b198e614-5708-4054-afe9-4e4b17376e03\"",
        "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/AllowVnetInBound",
        "name": "AllowVnetInBound",
        "priority": 65000,
        "protocol": "*",
        "provisioningState": "Succeeded",
        "resourceGroup": "lbResourceGroup",
        "sourceAddressPrefix": "VirtualNetwork",
        "sourceAddressPrefixes": [],
        "sourceApplicationSecurityGroups": null,
        "sourcePortRange": "*",
        "sourcePortRanges": [],
        "type": "Microsoft.Network/networkSecurityGroups/defaultSecurityRules"
      },
```

2.6 Create a network security group rule

```
az network nsg rule create \
  --resource-group lbResourceGroup \
  --nsg-name myNSG \
  --name myNSGRuleHTTP \
  --protocol '*' \
  --direction inbound \
  --source-address-prefix '*' \
  --source-port-range '*' \
  --destination-address-prefix '*' \
  --destination-port-range 80 \
  --access allow \
  --priority 200
```



Step 3: Creating backend servers

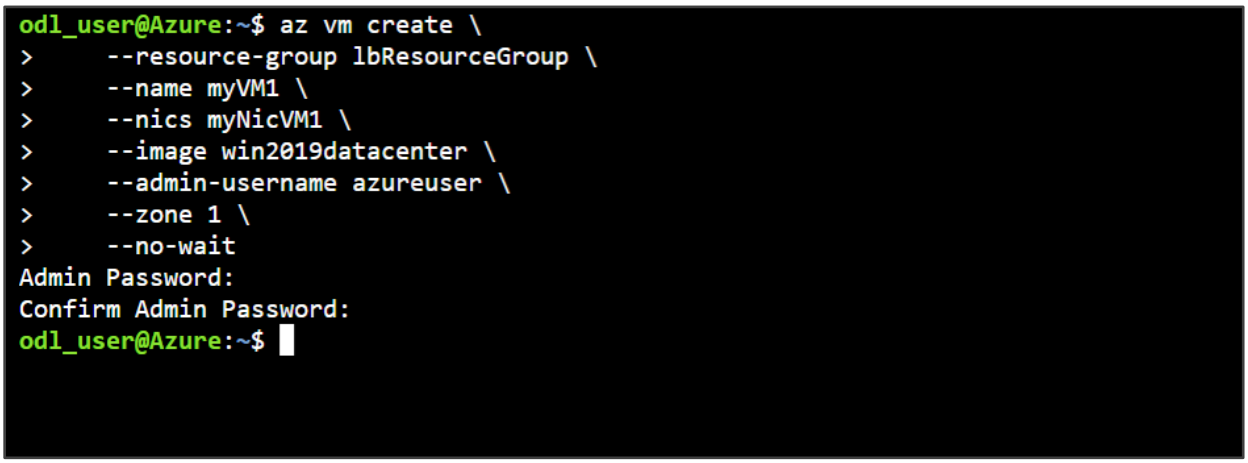
3.1 Create three virtual network interfaces for virtual machines

```
array=(myNicVM1 myNicVM2 myNicVM3)
 for vmnic in "${array[@]}"
 do
  az network nic create \
     --resource-group lbResourceGroup \
     --name $vmnic \
     --vnet-name myVNet \
     --subnet myBackEndSubnet \
     --network-security-group myNSG
 done
```

```
odl_user@Azure:~$ array=(myNicVM1 myNicVM2 myNicVM3)
odl_user@Azure:~$   for vmnic in "${array[@]}"
>   do
>     az network nic create \
>         --resource-group lbResourceGroup \
>         --name $vmnic \
>         --vnet-name myVNet \
>         --subnet myBackEndSubnet \
>         --network-security-group myNSG
>   done
{
  "NewNIC": {
    "dnsSettings": {
      "appliedDnsServers": [],
      "dnsServers": [],
      "internalDnsNameLabel": null,
      "internalDomainNameSuffix": "rlooituzgc0urmyanfdikxrdab.bx.internal.cloudapp.net",
      "internalFqdn": null
    },
    "dscpConfiguration": null,
    "enableAcceleratedNetworking": false,
    "enableIpForwarding": false,
    "etag": "W/\"7de2e7fa-67d4-46fd-a84a-aebc2d6cdd50\"",
    "extendedLocation": null,
    "hostedWorkloads": [],
    "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.Network/networkInterfaces/myNicVM1",
    "ipConfigurations": [
      {
        "applicationGatewayBackendAddressPools": null,
        "applicationSecurityGroups": null,
        "etag": "W/\"7de2e7fa-67d4-46fd-a84a-aebc2d6cdd50\"",
        "gatewayLoadBalancer": null,
```

3.2 Create virtual machines with **az vm create**

```
az vm create \
  --resource-group lbResourceGroup \
  --name myVM1 \
  --nics myNicVM1 \
  --image win2019datacenter \
  --admin-username azureuser \
  --zone 1 \
  --no-wait
```




```
az vm create \
  --resource-group lbResourceGroup \
  --name myVM2 \
  --nics myNicVM2 \
  --image win2019datacenter \
  --admin-username azureuser \
  --zone 2 \
  --no-wait
```

```
odl_user@Azure:~$ az vm create \
>     --resource-group lbResourceGroup \
>     --name myVM2 \
>     --nics myNicVM2 \
>     --image win2019datacenter \
>     --admin-username azureuser \
>     --zone 2 \
>     --no-wait
Admin Password:
Confirm Admin Password:
odl_user@Azure:~$
```

```
az vm create \
  --resource-group  lbResourceGroup \
  --name myVM3 \
  --nics myNicVM3 \
  --image win2019datacenter \
  --admin-username azureuser \
  --zone 3 \
  --no-wait
```

```
odl_user@Azure:~$ az vm create \
>     --resource-group  lbResourceGroup \
>     --name myVM3 \
>     --nics myNicVM3 \
>     --image win2019datacenter \
>     --admin-username azureuser \
>     --zone 3 \
>     --no-wait
Admin Password:
Confirm Admin Password:
odl_user@Azure:~$
```

Step 4: Creating a public IP address

4.1 Create a zone-redundant public IP

```
az network public-ip create \
  --resource-group lbResourceGroup \
  --name myPublicIP \
  --sku Standard
```

```
odl_user@Azure:~$ az network public-ip create \
>     --resource-group lbResourceGroup \
>     --name myPublicIP \
>     --sku Standard
[Coming breaking change] In the coming release, the default behavior will be changed as follows when sk
et a zone-redundant IP indicated by zones:["1","2","3"]; For non-zonal regions, you will get a non zone
{
  "publicIp": {
    "ddosSettings": null,
    "deleteOption": null,
    "dnsSettings": null,
    "etag": "W/\"be82ea24-ab7d-42e5-9237-4b522b14c2cc\"",
    "extendedLocation": null,
    "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers
    "idleTimeoutInMinutes": 4,
    "ipAddress": "52.191.27.222",
    "ipConfiguration": null,
    "ipTags": [],
    "linkedPublicIpAddress": null,
    "location": "eastus",
    "migrationPhase": null,
    "name": "myPublicIP",
    "natGateway": null,
```

Step 5: Creating a Load Balancer

5.1 Create a public load balancer with **az network lb create:**

- Named **myLoadBalancer**
- A frontend pool named **myFrontEnd**
- A backend pool named **myBackEndPool**
- Associated with the public IP address **myPublicIP** created in the preceding step

```
az network lb create \
  --resource-group lbResourceGroup \
  --name myLoadBalancer \
  --sku Standard \
  --public-ip-address myPublicIP \
  --frontend-ip-name myFrontEnd \
  --backend-pool-name myBackEndPool
```

```
odl_user@Azure:~$ az network lb create \
>     --resource-group lbResourceGroup \
>     --name myLoadBalancer \
>     --sku Standard \
>     --public-ip-address myPublicIP \
>     --frontend-ip-name myFrontEnd \
>     --backend-pool-name myBackEndPool
{
  "loadBalancer": {
    "backendAddressPools": [
      {
        "etag": "W/\"27cc0f77-2afa-4533-9b60-0892d2dbdea5\"",
        "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.N
ls/myBackEndPool",
        "name": "myBackEndPool",
        "properties": {
          "loadBalancerBackendAddresses": [],
          "provisioningState": "Succeeded"
        },
        "resourceGroup": "lbResourceGroup",
        "type": "Microsoft.Network/loadBalancers/backendAddressPools"
      }
    ],
    "frontendIPConfigurations": [
      {
        "etag": "W/\"27cc0f77-2afa-4533-9b60-0892d2dbdea5\"",
        "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.N
rations/myFrontEnd",
        "name": "myFrontEnd",
        "properties": {
          "privateIPAllocationMethod": "Dynamic",
```

5.2 Create a health probe with **az network lb probe create**

**az network lb probe create **

**--resource-group lbResourceGroup **

**--lb-name myLoadBalancer **

**--name myHealthProbe **

**--protocol tcp **

--port 80

```
odl_user@Azure:~$ az network lb probe create \
>     --resource-group lbResourceGroup \
>     --lb-name myLoadBalancer \
>     --name myHealthProbe \
>     --protocol tcp \
>     --port 80
{
  "etag": "W/\"61c07791-7e67-4dbd-be8d-fe8be561b2de\"",
  "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.Network/
  "intervalInSeconds": 15,
  "loadBalancingRules": null,
  "name": "myHealthProbe",
  "numberOfProbes": 2,
  "port": 80,
  "protocol": "Tcp",
  "provisioningState": "Succeeded",
  "requestPath": null,
  "resourceGroup": "lbResourceGroup",
  "type": "Microsoft.Network/loadBalancers/probes"
}
```

5.3 Create a load balancer rule

```
az network lb rule create \
  --resource-group lbResourceGroup \
  --lb-name myLoadBalancer \
  --name myHTTPRule \
  --protocol tcp \
  --frontend-port 80 \
  --backend-port 80 \
  --frontend-ip-name myFrontEnd \
  --backend-pool-name myBackEndPool \
  --probe-name myHealthProbe \
  --disable-outbound-snat true \
  --idle-timeout 15 \
  --enable-tcp-reset true
```

```
odl_user@Azure:~$ az network lb rule create \
>     --resource-group lbResourceGroup \
>     --lb-name myLoadBalancer \
>     --name myHTTPRule \
>     --protocol tcp \
>     --frontend-port 80 \
>     --backend-port 80 \
>     --frontend-ip-name myFrontEnd \
>     --backend-pool-name myBackEndPool \
>     --probe-name myHealthProbe \
>     --disable-outbound-snat true \
>     --idle-timeout 15 \
>     --enable-tcp-reset true
{
  "backendAddressPool": {
    "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Mi
yBackEndPool",
    "resourceGroup": "lbResourceGroup"
  },
  "backendAddressPools": [
    {
      "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/
/myBackEndPool",
      "resourceGroup": "lbResourceGroup"
    }
  ],
  "backendPort": 80,
  "disableOutboundSnat": true,
  "enableFloatingIp": false,
  "enableTcpReset": true,
  "etag": "W/\"fe5e58a0-5aca-4df9-abc0-4913f278df48\"",
```

5.4 Add virtual machines to the load balancer backend pool

```
array=(myNicVM1 myNicVM2 myNicVM3)
 for vmnic in "${array[@]}"
 do
  az network nic ip-config address-pool add \
   --address-pool myBackendPool \
   --ip-config-name ipconfig1 \
   --nic-name $vmnic \
   --resource-group lbResourceGroup \
   --lb-name myLoadBalancer
 done
```

```
odl_user@Azure:~$ array=(myNicVM1 myNicVM2 myNicVM3)
odl_user@Azure:~$   for vmnic in "${array[@]}"
>   do
>     az network nic ip-config address-pool add \
>      --address-pool myBackendPool \
>      --ip-config-name ipconfig1 \
>      --nic-name $vmnic \
>      --resource-group lbResourceGroup \
>      --lb-name myLoadBalancer
>   done

{
  "applicationGatewayBackendAddressPools": null,
  "applicationSecurityGroups": null,
  "etag": "W/\"5e1855d2-0b39-49da-ae03-968d04f5bd86\"",
  "gatewayLoadBalancer": null,
  "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.Network/ne
1",
  "loadBalancerBackendAddressPools": [
    {
      "backendIpConfigurations": null,
      "etag": null,
      "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.Networ
/myBackEndPool",
      "inboundNatRules": null,
      "loadBalancerBackendAddresses": null,
      "loadBalancingRules": null,
      "location": null,
      "name": null,
      "outboundRule": null,
      "outboundRules": null,
      "provisioningState": null,
```

Step 6: Creating an Outbound Rule Configuration

6.1 Use **az network public-ip create** to create a single IP for the outbound connectivity

```
az network public-ip create \
  --resource-group lbResourceGroup \
  --name myPublicIPOutbound \
  --sku Standard
```

```
odl_user@Azure:~$ az network public-ip create \
>     --resource-group lbResourceGroup \
>     --name myPublicIPOutbound \
>     --sku Standard
[Coming breaking change] In the coming release, the default behavior will be changed as follows when sku
et a zone-redundant IP indicated by zones:["1","2","3"]; For non-zonal regions, you will get a non zone-r
{
  "publicIp": {
    "ddosSettings": null,
    "deleteOption": null,
    "dnsSettings": null,
    "etag": "W/\"6f5abd4f-2def-48fa-a68b-90f1929f6d7d\"",
    "extendedLocation": null,
    "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/M
    "idleTimeoutInMinutes": 4,
    "ipAddress": "13.90.25.121",
    "ipConfiguration": null,
    "ipTags": [],
    "linkedPublicIpAddress": null,
    "location": "eastus",
    "migrationPhase": null,
    "name": "myPublicIPOutbound",
    "natGateway": null,
    "provisioningState": "Succeeded",
    "publicIpAddressVersion": "IPv4",
    "publicIpAllocationMethod": "Static",
    "publicIpPrefix": null,
    "resourceGroup": "lbResourceGroup",
    "resourceGuid": "bc096171-e262-48d3-977b-4d4b6d609b3e",
    "servicePublicIpAddress": null,
    "sku": {
```

6.2 Use **az network public-ip prefix create** to create a public IP prefix for the outbound connectivity

**az network public-ip prefix create **
**--resource-group lbResourceGroup **
**--name myPublicIPPrefixOutbound **
--length 28

```
odl_user@Azure:~$ az network public-ip prefix create \
>     --resource-group lbResourceGroup \
>     --name myPublicIPPrefixOutbound \
>     --length 28
{
  "customIpPrefix": null,
  "etag": "W/\"4a5310b3-b729-4d79-b7ee-653914a624eb\"",
  "extendedLocation": null,
  "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microso
  "ipPrefix": "52.190.10.240/28",
  "ipTags": [],
  "loadBalancerFrontendIpConfiguration": null,
  "location": "eastus",
  "name": "myPublicIPPrefixOutbound",
  "natGateway": null,
  "prefixLength": 28,
  "provisioningState": "Succeeded",
  "publicIpAddressVersion": "IPv4",
  "publicIpAddresses": null,
  "resourceGroup": "lbResourceGroup",
  "resourceGuid": "a87293fe-d7e0-492a-bb5f-0630309c92cc",
  "sku": {
    "name": "Standard",
    "tier": "Regional"
  },
  "tags": null,
  "type": "Microsoft.Network/publicIPPrefixes",
  "zones": null
}
```

6.3 Create outbound frontend IP configuration

Public IP:

```
az network lb frontend-ip create \
   --resource-group lbResourceGroup \
   --name myFrontEndOutbound \
   --lb-name myLoadBalancer \
   --public-ip-address myPublicIPOutbound
```

```
odl_user@Azure:~$ az network lb frontend-ip create \
>     --resource-group lbResourceGroup \
>     --name myFrontEndOutbound \
>     --lb-name myLoadBalancer \
>     --public-ip-address myPublicIPOutbound
{
  "etag": "W/\"f3a91599-10e3-48be-857f-6ea562baf879\"",
  "gatewayLoadBalancer": null,
  "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.Network/loadBalance
s/myFrontEndOutbound",
  "inboundNatPools": null,
  "inboundNatRules": null,
  "loadBalancingRules": null,
  "name": "myFrontEndOutbound",
  "outboundRules": null,
  "privateIpAddress": null,
  "privateIpAddressVersion": null,
  "privateIpAllocationMethod": "Dynamic",
  "provisioningState": "Succeeded",
  "publicIpAddress": {
    "ddosSettings": null,
    "deleteOption": null,
```

Public IP Prefix:

```
az network lb frontend-ip create \
   --resource-group lbResourceGroup \
   --name myFrontEndOutbound \
   --lb-name myLoadBalancer \
   --public-ip-prefix myPublicIPPrefixOutbound
```

```
odl_user@Azure:~$ az network lb frontend-ip create \
>     --resource-group lbResourceGroup \
>     --name myFrontEndOutbound \
>     --lb-name myLoadBalancer \
>     --public-ip-address myPublicIPOutbound
{
  "etag": "W/\"f3a91599-10e3-48be-857f-6ea562baf879\"",
  "gatewayLoadBalancer": null,
  "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.Network/loadBalancers/
  "inboundNatPools": null,
  "inboundNatRules": null,
  "loadBalancingRules": null,
  "name": "myFrontEndOutbound",
  "outboundRules": null,
  "privateIpAddress": null,
  "privateIpAddressVersion": null,
  "privateIpAllocationMethod": "Dynamic",
  "provisioningState": "Succeeded",
  "publicIpAddress": {
    "ddosSettings": null,
    "deleteOption": null,
    "dnsSettings": null,
    "etag": null,
```

6.4 Create an outbound pool

```
az network lb address-pool create \
  --resource-group lbResourceGroup \
  --lb-name myLoadBalancer \
  --name myBackendPoolOutbound
```

```
odl_user@Azure:~$ az network lb address-pool create \
>     --resource-group lbResourceGroup \
>     --lb-name myLoadBalancer \
>     --name myBackendPoolOutbound
{
  "backendIpConfigurations": null,
  "etag": "W/\"11539187-f9ae-4aa6-9e98-11e57ef5ad90\"",
  "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsof
  "inboundNatRules": null,
  "loadBalancerBackendAddresses": [],
  "loadBalancingRules": null,
  "location": null,
  "name": "myBackendPoolOutbound",
  "outboundRule": null,
  "outboundRules": null,
  "provisioningState": "Succeeded",
  "resourceGroup": "lbResourceGroup",
  "tunnelInterfaces": null,
  "type": "Microsoft.Network/loadBalancers/backendAddressPools"
}
odl_user@Azure:~$
```

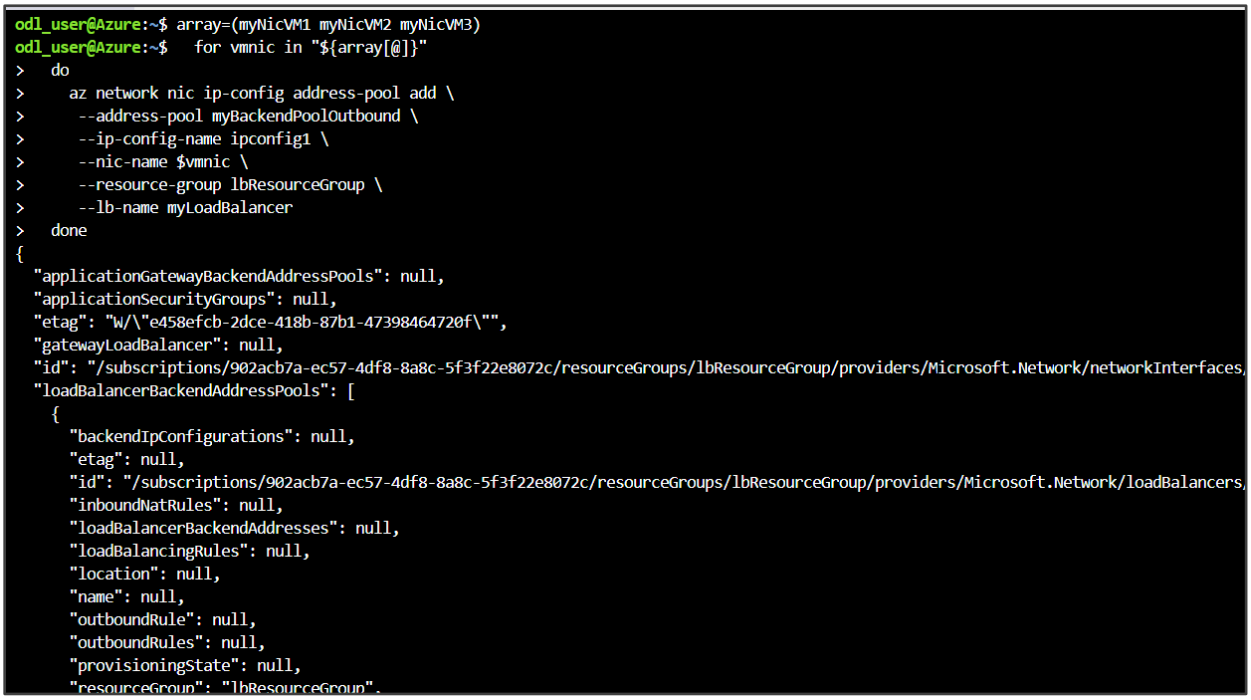
6.5 Create an outbound rule

```
az network lb outbound-rule create \
  --resource-group lbResourceGroup \
  --lb-name myLoadBalancer \
  --name myOutboundRule \
  --frontend-ip-configs myFrontEndOutbound \
  --protocol All \
  --idle-timeout 15 \
  --outbound-ports 10000 \
  --address-pool myBackEndPoolOutbound
```

```
odl_user@Azure:~$ az network lb outbound-rule create \
>     --resource-group lbResourceGroup \
>     --lb-name myLoadBalancer \
>     --name myOutboundRule \
>     --frontend-ip-configs myFrontEndOutbound \
>     --protocol All \
>     --idle-timeout 15 \
>     --outbound-ports 10000 \
>     --address-pool myBackEndPoolOutbound
{
  "allocatedOutboundPorts": 10000,
  "backendAddressPool": {
    "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.Network/loa
    "resourceGroup": "lbResourceGroup"
  },
  "enableTcpReset": false,
  "etag": "W/\"82390102-95a4-4600-b90b-d6df886a9e91\"",
  "frontendIpConfigurations": [
    {
      "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.Network/l
      "resourceGroup": "lbResourceGroup"
    }
  ],
  "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Microsoft.Network/loadB
  "idleTimeoutInMinutes": 15,
  "name": "myOutboundRule",
  "protocol": "All",
  "provisioningState": "Succeeded",
  "resourceGroup": "lbResourceGroup",
  "type": "Microsoft.Network/loadBalancers/outboundRules"
}
odl_user@Azure:~$
```

6.6 Add virtual machines to the outbound pool

```
array=(myNicVM1 myNicVM2 myNicVM3)
 for vmnic in "${array[@]}"
 do
  az network nic ip-config address-pool add \
   --address-pool myBackendPoolOutbound \
   --ip-config-name ipconfig1 \
   --nic-name $vmnic \
   --resource-group lbResourceGroup \
   --lb-name myLoadBalancer
 done
```



Step 7: Installing IIS

7.1 Use the **az vm extension** set to install IIS on virtual machines

```
array=(myVM1 myVM2 myVM3)
  for vm in "${array[@]}"
  do
   az vm extension set \
    --publisher Microsoft.Compute \
    --version 1.8 \
    --name CustomScriptExtension \
    --vm-name $vm \
    --resource-group lbResourceGroup \
    --settings '{"commandToExecute":"powershell Add-WindowsFeature Web-Server;
powershell Add-Content -Path \"C:\\inetpub\\wwwroot\\Default.htm\" -Value
$($env:computername)"}'
  done
```

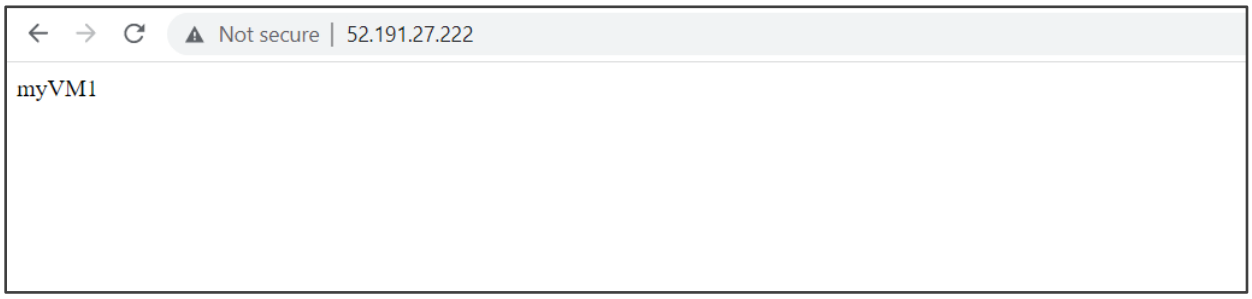
```
odl_user@Azure:~$ array=(myVM1 myVM2 myVM3)
odl_user@Azure:~$     for vm in "${array[@]}"
>     do
>      az vm extension set \
>        --publisher Microsoft.Compute \
>        --version 1.8 \
>        --name CustomScriptExtension \
>        --vm-name $vm \
>        --resource-group lbResourceGroup \
>        --settings '{"commandToExecute":"powershell Add-WindowsFeature Web-Server; powershell Add-Content
>   done
{
  "autoUpgradeMinorVersion": true,
  "enableAutomaticUpgrade": null,
  "forceUpdateTag": null,
  "id": "/subscriptions/902acb7a-ec57-4df8-8a8c-5f3f22e8072c/resourceGroups/lbResourceGroup/providers/Micr
  "instanceView": null,
  "location": "eastus",
  "name": "CustomScriptExtension",
  "protectedSettings": null,
  "protectedSettingsFromKeyVault": null,
  "provisioningState": "Succeeded",
  "publisher": "Microsoft.Compute",
  "resourceGroup": "lbResourceGroup",
  "settings": {
    "commandToExecute": "powershell Add-WindowsFeature Web-Server; powershell Add-Content -Path \"C:\\inetp
  },
  "suppressFailures": null,
  "tags": null,
  "type": "Microsoft.Compute/virtualMachines/extensions",
  "typeHandlerVersion": "1.8",
  "typePropertiesType": "CustomScriptExtension"
}
```

Step 8: Testing the Load balancer

8.1 Get the public IP address of the load balancer and use **az network public-ip show**

```
az network public-ip show \
--resource-group lbResourceGroup \
--name myPublicIP \
--query ipAddress \
--output tsv
```

```
odl_user@Azure:~$ az network public-ip show \
>     --resource-group lbResourceGroup \
>     --name myPublicIP \
>     --query ipAddress \
>     --output tsv
52.191.27.222
odl_user@Azure:~$
```



As we can see from the diagram, when we are hitting the load balancer's public IP, the request is going to one of the virtual machines that we added to the backend pool. Subsequent hits on this IP will go to different virtual machines and the load balancer automatically does the load balancing and sends the request to the backend pool accordingly.