Topic : <u>Get started with Azure Load Balancer by using the Azure portal</u>

Contents:

Problem-statement:

Introduction

The Internal load balancer manages load for a private network with any inbound access from the public platform. As in the diagram below, the primary load balancer managing load from the internet is a public-type load balancer. But, the VMs communication to storage or database is managed through a type-internal load balancer.

Solution/ Architecture

Azure Load Balancer is a service provided by Microsoft Azure that enables the distribution of incoming traffic across multiple backend resources such as virtual machines, virtual machine scale sets, or availability sets. Load balancing improves application availability, increases scalability, and enhances resiliency by detecting unhealthy resources and redirecting traffic to healthy ones.

. In this blog, we will walk through the steps required to set up a load balancer in Azure using the Azure portal.

Prerequisites:

- An active Azure subscription
- At least two backend resources (virtual machines, virtual machine scale sets, or availability sets) in the same Azure region
- An Azure Virtual Network and subnet for the backend resources

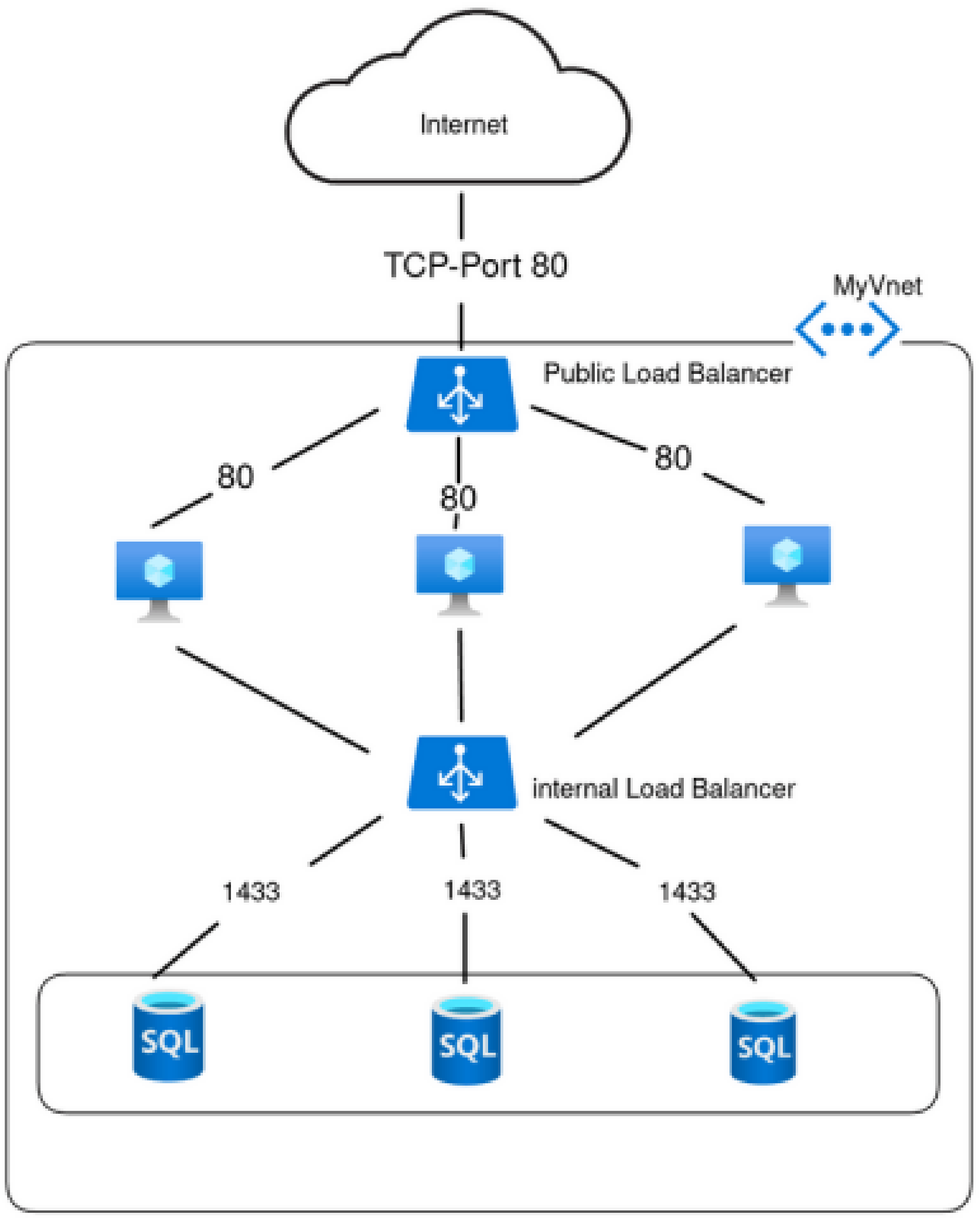
Steps to create an Azure Load Balancer:

1. Open the Azure portal and navigate to the Load balancers page.
2. Click on the "Add" button to create a new load balancer.
3. In the "Basics" tab, enter a name for the load balancer and select the subscription, resource group, and region.
4. Choose the "Public" or "Internal" load balancer type based on your requirements. For this guide, we will choose "Public."
5. In the "Configure load balancing" section, select "Availability set," "Virtual machine scale set," or

"Virtual machine" as the backend pool type. Then, select the backend pool and add the backend resources that you want to load balance.

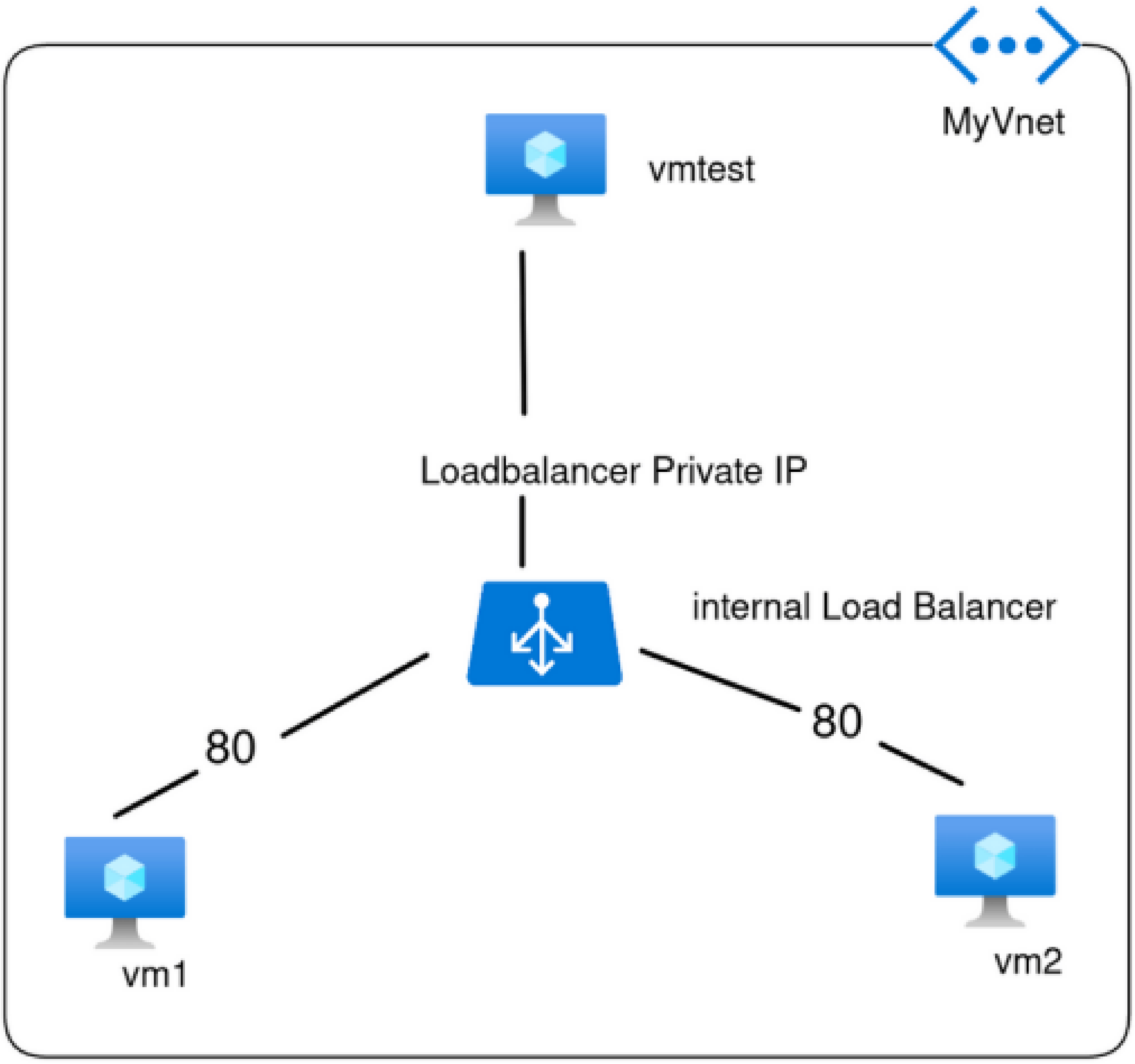
6. In the "Frontend IP configuration" section, create a public IP address or select an existing one. This IP address will be used as the entry point for incoming traffic.
7. In the "Health probes" section, configure the settings for the health probes that will be used to check the availability of the backend resources. You can set the probe protocol, port, and interval, among other settings.
8. In the "Load balancing rules" section, create a new rule to define how the traffic will be distributed across the backend resources. You can choose the protocol, port, and backend port, among other settings.
9. Review the settings and click on the "Review + create" button.
10. Once the validation is complete, click on the "Create" button to create the load balancer.
11. After the load balancer is created, you can verify its operation by accessing the public IP address in a web browser or using a tool such as Telnet to connect to the backend resources through the load balancer.

Hence, creating an Azure Load Balancer in the Azure portal involves selecting the load balancer type, configuring the backend resources, defining the health probes and load balancing rules, and reviewing the settings before creating the load balancer.



Internal Load Balancer Setup

Let us see our setup which we will explain here. We have 3 virtual machines, 1 virtual network, and 1 Load Balancer of type-internal. We will use the **vmtest** to test if we can balance the load for traffic from the same network. The load will be on web servers deployed on two VMs: **vm1** and **vm2**.



Create a common resource group in the Azure cloud for this demonstration. We are taking the name “rg-lb” and the rest of the default configuration.

Step 1: Creating the Virtual Network

We used the following configurations:

- Name: myVnet
- region: East US
- Address: 10.0.0.0/16
- subscription: Azure subscription 1
- resource group: rg-lb
- subnet name: mySubnet
- Subnet Address range: 10.0.0.0/24

Step 2: Creating the virtual machines

Create 3 VMs with similar configurations only name changes

- Name: vm1 / vm2 / vmtest

subscription: Azure subscription 1

resource group: rg-lb

region: East US

Availability options: Availability set (because distribute VMs across multiple fault domains but zone remains same)

Availability set: (create a new set and leave the fault domains:2 and update domains:5 as they are): myavailabilityset

Image: Windows Server 2016 Datacenter (windows servers are good for practice)

size: Standard_B2s

Username: azureuser

Password: yourown

Confirm Password: yourown

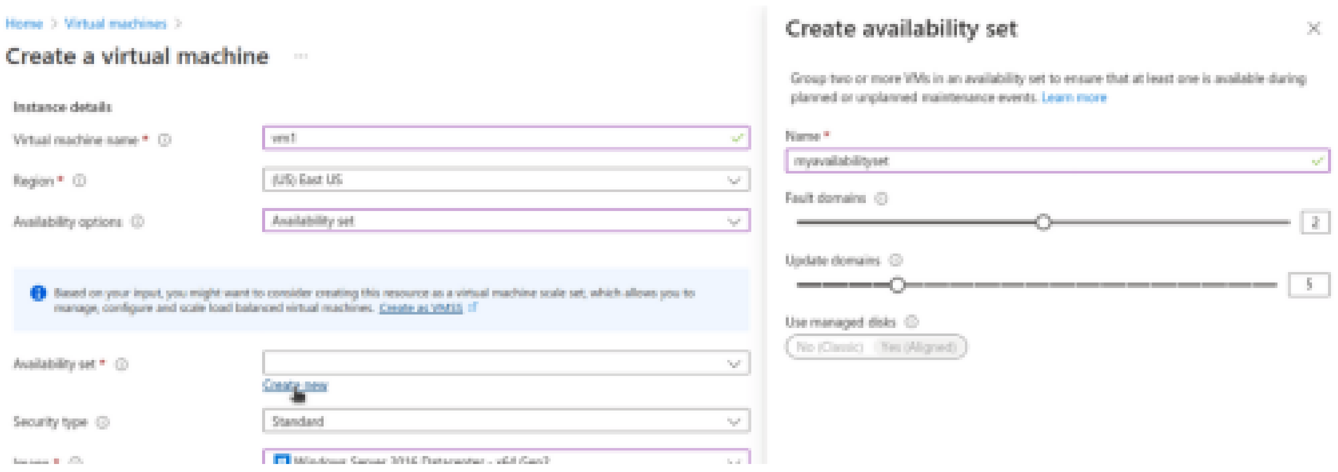
inbound ports: HTTP,HTTPS,RDP

OS disk type: Standard SSD

Boot diagnostics: Off (to stop from creating a storage account)

Virtual network: myVnet

subnet: mySubnet



Home > Virtual machines >

Create a virtual machine

Instance details

Virtual machine name *	vm1
Region *	(US) East US
Availability options	Availability set

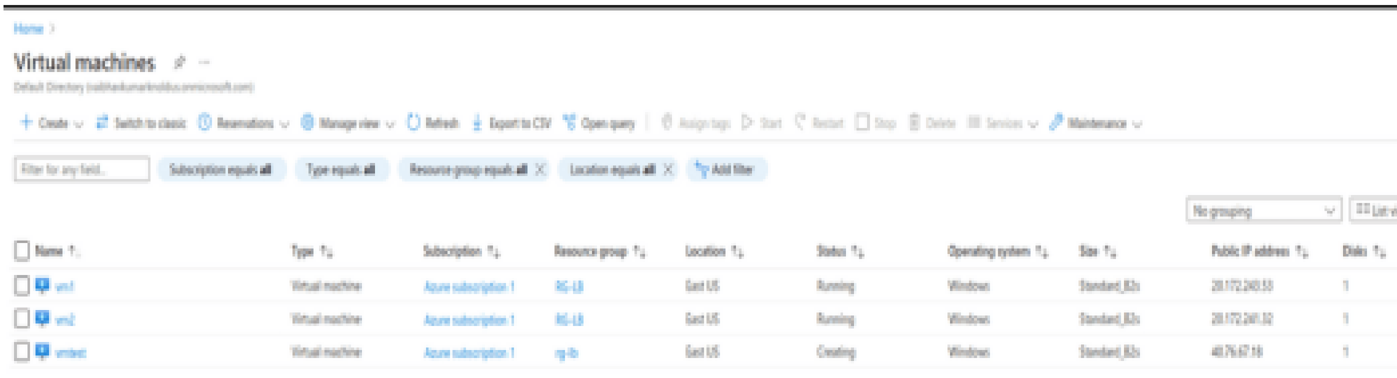
Based on your input, you might want to consider creating this resource as a virtual machine scale set, which allows you to manage, configure and scale load balanced virtual machines. Create as VMSS

Availability set *	Create new
Security type	Standard
Image *	Windows Server 2016 Datacenter - x64 Gen2

Create availability set

Group two or more VMs in an availability set to ensure that at least one is available during planned or unplanned maintenance events. Learn more

Name *	myavailabilityset
Fault domains	2
Update domains	5
Use managed disks	No (Classic) Yes (Aligned)



Home >

Virtual machines

Default Directory

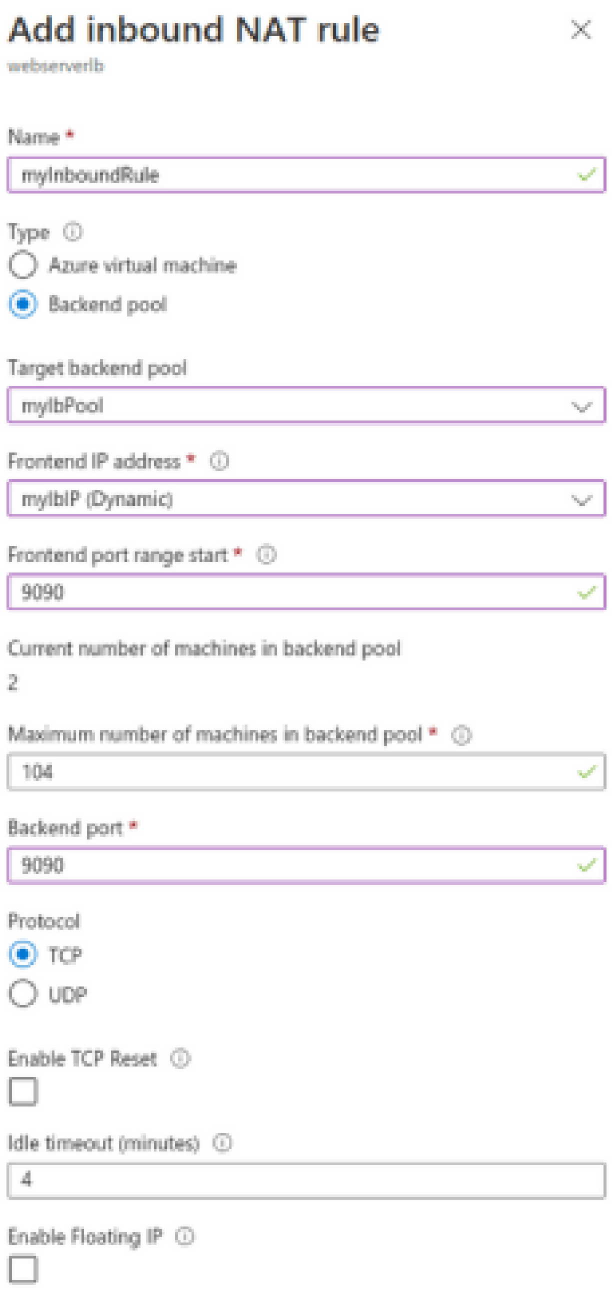
Name	Type	Subscription	Resource group	Location	Status	Operating system	Size	Public IP address	Disks
vm1	Virtual machine	Azure subscription 1	RG-LB	East US	Running	Windows	Standard_B2s	20.172.240.53	1
vm2	Virtual machine	Azure subscription 1	RG-LB	East US	Running	Windows	Standard_B2s	20.172.241.32	1
vmtest	Virtual machine	Azure subscription 1	rg-lb	East US	Creating	Windows	Standard_B2s	40.76.67.18	1

Step 3: Creating a Load Balancer

We will be creating a load balancer with the following configurations:

- name: webserverlb

- type: Internal
- SKU: Standard
- virtual network: myVnet
- subnet: mysubnet
- resource group: rg-lb
- location: East US
- tier: Regional
- frontend IP
  - Name: mylbIP
  - Virtual Network: myNet
  - Subnet: mySubnet(10.0.0.0/24)
  - Assignment: Dynamic
- Backend pool
  - Name: mylbPool
  - Backend Pool Configuration: IP address
  - IP addresses: select private IPs of vm1 and vm2
- load balancing group:
  - Name: lb-rule
  - IP Version: IPv4
  - Backend pool: mylbPool
  - Protocol: TCP
  - Port: 80
  - Backend Port: 80
  - health probe (create new)
    - Name: mylbProbe
    - protocol: HTTP
    - Interval: 5 seconds
    - Path: /
    - Port: 80
  - ideal time out 4 minute
  - TCP reset: disabled
  - Floating IP: disabled


Add inbound NAT rule

webserverlb

Name *
myInboundRule

Type
Azure virtual machine
[✓ Backend pool]

Target backend pool
myIbPool

Frontend IP address *
myIbIP (Dynamic)

Frontend port range start *
9090

Current number of machines in backend pool
2

Maximum number of machines in backend pool *
104

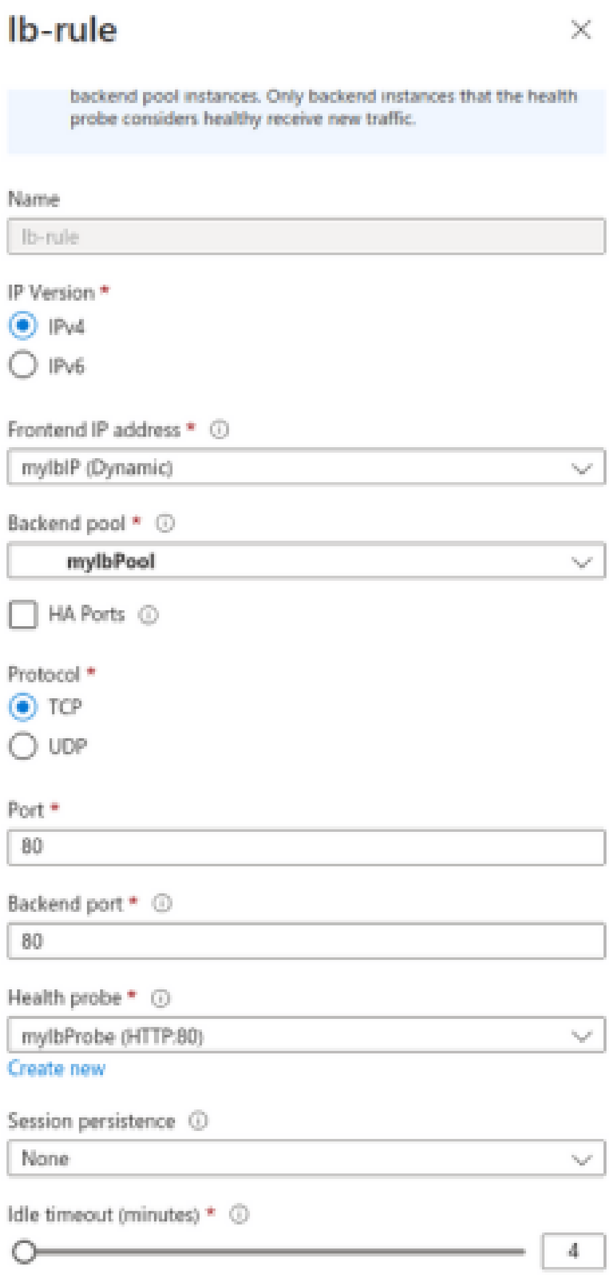
Backend port *
9090

Protocol
[✓ TCP]
UDP

Enable TCP Reset
☐

Idle timeout (minutes)
4

Enable Floating IP
☐


lb-rule

backend pool instances. Only backend instances that the health probe considers healthy receive new traffic.

Name: lb-rule

IP Version *: [✓ IPv4] IPv6

Frontend IP address *: myIbIP (Dynamic)

Backend pool *: myIbPool

HA Ports: ☐

Protocol *: [✓ TCP] UDP

Port *: 80

Backend port *: 80

Health probe *: myIbProbe (HTTP:80)

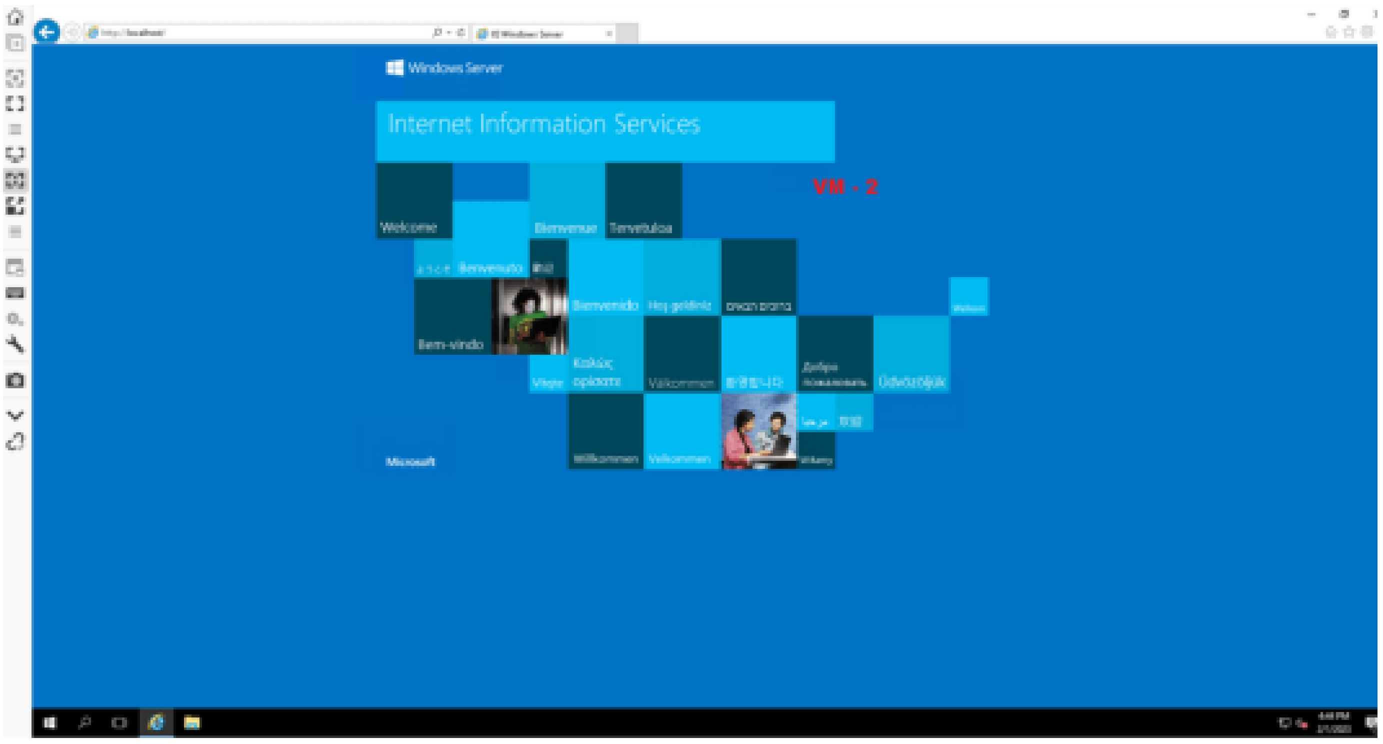
Create new

Session persistence: None

Idle timeout (minutes) *: 4

Step 4: Web Server installation on VMs

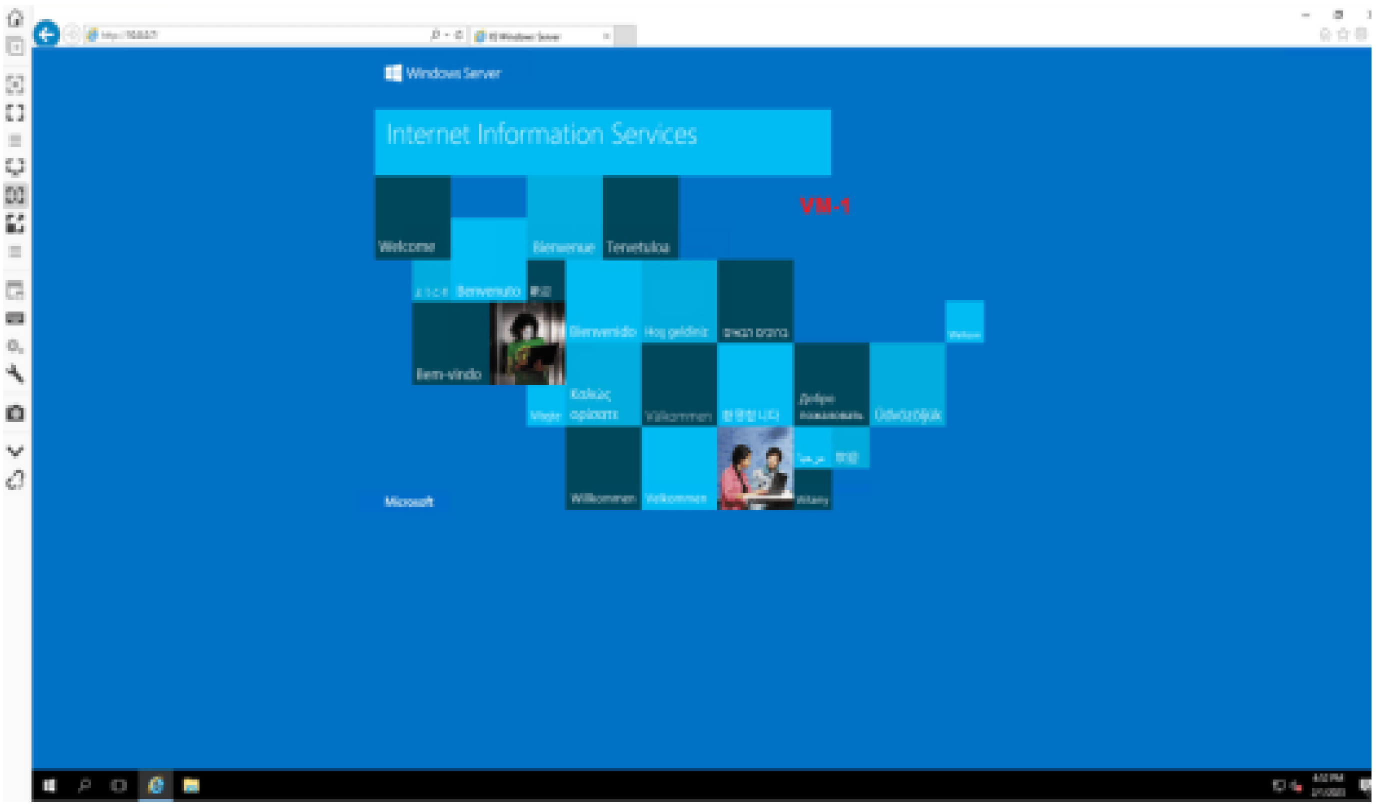
Download the RDP file of **vm1** and **vm2**. We are using **Remina Connect** to run the virtual machines locally with an RDP connection. You can use **Remote Desktop Connection** in Windows. Install the IIS web server on vm1 and vm2 using server manager. And set up the default pages in a way that distinguishes both pages like we have done.



Step 5: Testing the connection

Download the RDP file of **vmtest** and open using the RDP connector. Copy the Private IP of the load balancer from the frontend IP configuration. Open the private IP on the browser, and you would get the default page of one of the VMs. As the idle time to wait is 4 minutes, then refresh you would get the other VM's default page as shown here.

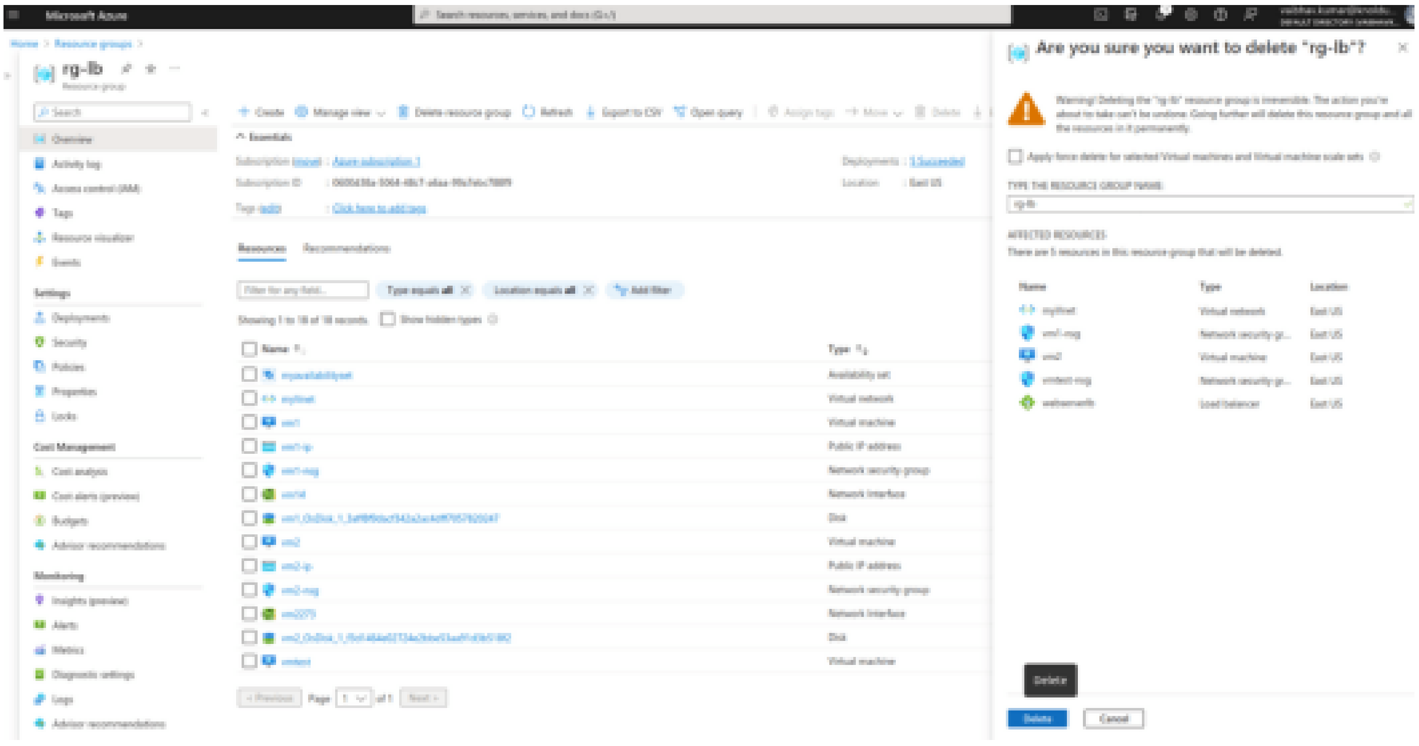
webserverlb | Frontend IP configuration



Conclusion:

This was to practice using an internal load balancer. We can create a complete network infrastructure as stated in the starting diagram. You can refer to [azure docs](#) for understanding.

Note: Do not forget to delete the resource group to delete all resources created.



Subscription (move) : Azure subscription 1

Subscription ID : [illegible]

Tags (edit) : Click here to add tags

Deployments : 1 Succeeded

Location : East US

Showing 1 to 13 of 13 records.

Name	Type
myavailabilityset	Availability set
myVNet	Virtual network
vm1	Virtual machine
vm1-ip	Public IP address
vm1-nsg	Network security group
vm1[illegible]	Network interface
vm1_OsDisk_1_[illegible]	Disk
vm2	Virtual machine
vm2-ip	Public IP address
vm2-nsg	Network security group
vm2[illegible]	Network interface
vm2_OsDisk_1_[illegible]	Disk
vmtest	Virtual machine

Page 1 of 1

Are you sure you want to delete "rg-lb"?

Warning! Deleting the "rg-lb" resource group is irreversible. The action you're about to take can't be undone. Going further will delete this resource group and all the resources in it permanently.

Apply force delete for selected Virtual machines and Virtual machine scale sets

TYPE THE RESOURCE GROUP NAME: rg-lb

AFFECTED RESOURCES

There are 5 resources in this resource group that will be deleted.

Name	Type	Location
myVNet	Virtual network	East US
vm1-nsg	Network security gr...	East US
vm2	Virtual machine	East US
vmtest-nsg	Network security gr...	East US
[illegible]	Load balancer	East US

Use Cases of Azure Load Balancer :

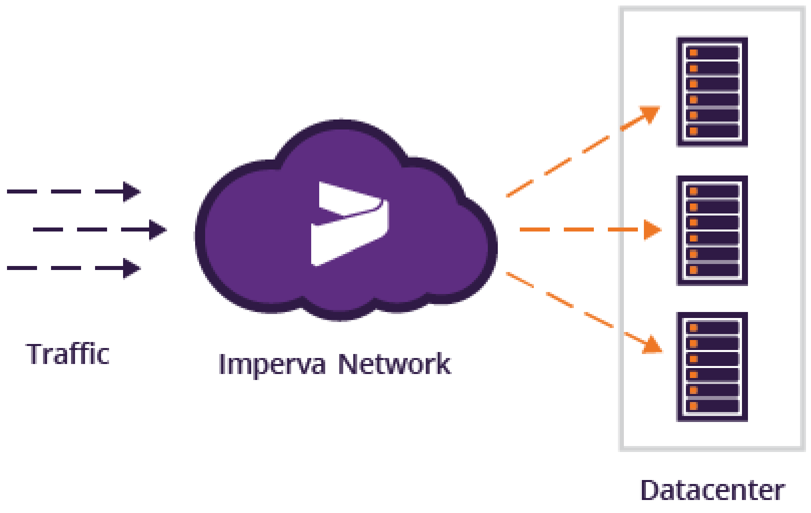
Use Case 1:

Single Data Center with Multiple Origin Servers – All Active

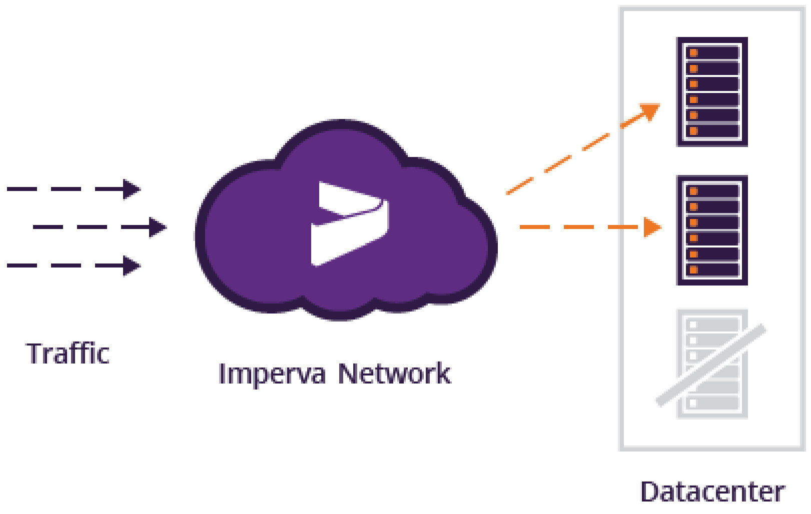
Description: If you have a single data center with multiple servers, select the Multiple Origin Servers (Single Data Center) topology setting. If all servers should be active at all times, select the Active Server setting for each server that you add.

You can configure each server to have a separate public IP address, or you can configure only one server with a public IP address, and route traffic to the other servers using port forwarding.

If one or more servers are identified as down, traffic is routed to the remaining active servers



Multiple Active Servers: Load Balancing among All Servers



Multiple Active Servers: Some Servers Down - Load Balancing among Functioning Servers

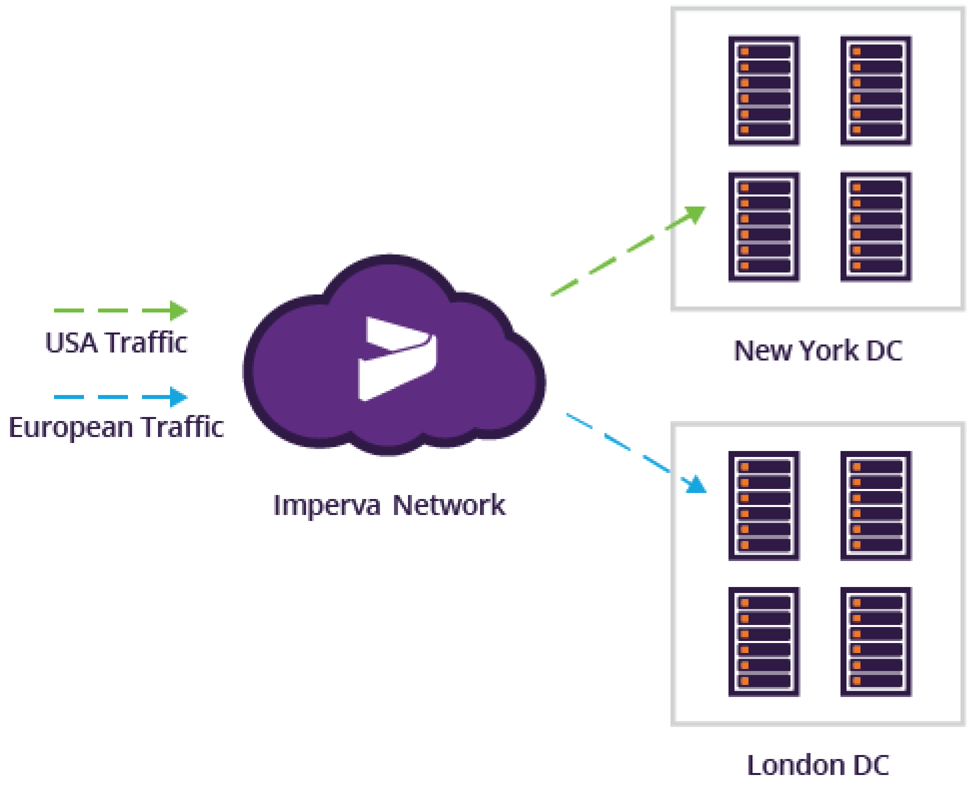
Availability: Purchase of the Load Balancing add-on is required for setups of three or more servers.

Use Case 2: Multiple Data Centers for Localized Content

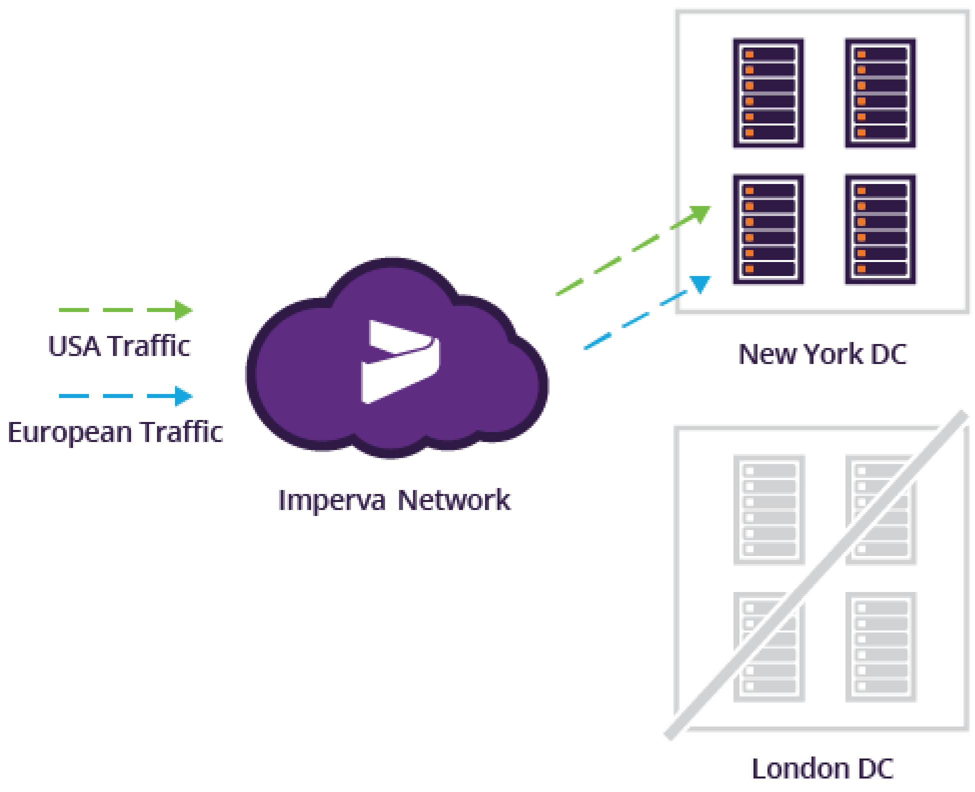
Description: Some websites provide localized content, depending on the location from which the site is accessed. For instance, users from the West Coast of the US might see different articles or advertisements than users from the East Coast.

To support this use case, select the Multiple Data Centers topology setting, and select the Geo-Targeting Preferred mode. This means that the load balancing mechanism will serve users from the data center that is geographically closest to them, as long as that data center is active. However, if the closest data center is down, user requests will be served from other active data centers, according to the Best Connection Time mode.

To work in Geo-Targeting Preferred mode, you must assign each geographical area to a specific data center. The last area assigned to a data center will always be Rest of the World. This "catch all" data center will handle requests from all areas not explicitly assigned to another data center



Geo-Targeting Preferred: Users Served from Targeted Data Center



Geo-Targeting Preferred: If Data Center Down, Users Served from Active Data Center with Best Connection Time

Availability: Purchase of the Load Balancing add-on is required.

Technical Details and Implementation of solution

Azure Load Balancer is a layer-4 load balancer that distributes incoming traffic across multiple backend resources, such as virtual machines, virtual machine scale sets, or availability sets. It works by assigning each backend resource a unique IP address and forwarding incoming traffic to the appropriate backend resource based on the load balancing rules and health probes.

Technical Details:

- Azure Load Balancer operates at the transport layer (layer 4) of the OSI model, which means that it forwards traffic based on the source IP address, destination IP address, protocol, and port number.
- Azure Load Balancer can operate in two modes: public and internal. In public mode, it provides a public IP address that can be used to access the backend resources over the internet. In internal mode, it provides a private IP address that can be used to access the backend resources within the same virtual network.
- Azure Load Balancer supports various load balancing algorithms, including round-robin, source IP

affinity, and least connections. Round-robin distributes traffic evenly across all healthy backend resources, while source IP affinity directs traffic from the same client IP address to the same backend resource. Least connections directs traffic to the backend resource with the fewest active connections.
- Azure Load Balancer uses health probes to check the availability of the backend resources. Health probes are sent to the backend resources at regular intervals, and if a resource fails to respond, it is marked as unhealthy and traffic is redirected to a healthy resource.

Implementation Steps:

1. Create a virtual network and subnet for the backend resources if one does not already exist.
2. Create or select the backend resources that you want to load balance. These can be virtual machines, virtual machine scale sets, or availability sets.
3. Create a public or internal load balancer in the Azure portal. Specify the load balancer name, subscription, resource group, and region.
4. Specify the backend pool type (availability set, virtual machine scale set, or virtual machine) and add the backend resources to the pool.
5. Create a frontend IP address and specify the load balancer type (public or internal).
6. Configure the health probes to check the availability of the backend resources.
7. Create one or more load balancing rules to define how traffic is distributed across the backend resources. Specify the protocol, port, and backend port for each rule.
8. Review the load balancer settings and click on the "Create" button to create the load balancer.
9. Once the load balancer is created, test it by accessing the public IP address in a web browser or using a tool such as Telnet to connect to the backend resources through the load balancer.

CHALLENGES IN IMPLEMENTING THE SOLUTION.

Technical Details and Implementation of solution

While implementing Azure Load Balancer using the Azure portal, there are several challenges that one may face, including:

1. **Understanding Load Balancer Concepts:** Understanding the concept of load balancing and how Azure Load Balancer works can be challenging for beginners. It is essential to have a clear understanding of the backend resources, health probes, and load balancing rules to configure the load balancer correctly.
2. **Configuring Backend Resources:** Configuring the backend resources, including virtual machines, virtual machine scale sets, or availability sets, can be complex, especially if they are in different availability zones or regions. Proper network configuration and firewall rules may also need to be set up to allow traffic to pass through.
3. **Security Considerations:** Load balancing may require opening ports and exposing resources to

the internet, which can pose security risks. It is crucial to ensure that security measures are in place to protect against potential threats.

4. **Load Balancer Configuration Errors:** Misconfiguration of the load balancer can result in traffic not being distributed as intended or backend resources not being accessible. Careful attention should be paid to the configuration of health probes and load balancing rules to ensure that traffic is distributed correctly.
5. **Cost Considerations:** Azure Load Balancer is a paid service, and the cost depends on the number of frontend and backend connections, data processing, and network data transfer. It is essential to consider the cost implications when setting up the load balancer.

In summary, challenges in implementing Azure Load Balancer using the Azure portal may include understanding load balancer concepts, configuring backend resources, considering security implications, avoiding configuration errors, and considering the cost of the service. Proper planning and careful attention to configuration can help overcome these challenges.

Business Benefit:

Azure Load Balancer is a powerful tool that can help businesses improve their application performance and availability by distributing incoming traffic across multiple virtual machines (VMs) or backend services. When it comes to getting started with Azure Load Balancer, the Azure portal is an easy and effective way to manage your load balancer resources. Here are some business benefits of using Azure Load Balancer through the Azure portal:

1. **Simplified management:** The Azure portal provides a user-friendly interface that simplifies the management of Azure Load Balancer resources. Users can easily create, configure, and manage load balancers, backend pools, health probes, and rules without needing to use command-line tools or write complex scripts.
2. **Improved application performance:** Azure Load Balancer can distribute traffic across multiple backend servers, which can improve application performance by reducing server load and improving response times. The Azure portal allows users to configure load balancing algorithms and health probes to ensure that traffic is distributed evenly and efficiently.
3. **Enhanced availability:** By using Azure Load Balancer, businesses can improve the availability of their applications by distributing traffic across multiple backend servers. If one server fails or becomes unavailable, traffic can be automatically redirected to healthy servers, ensuring that the application remains available to users.
4. **Scalability:** Azure Load Balancer can also help businesses scale their applications by distributing traffic across multiple VMs or backend services. The Azure portal allows users to easily add or remove VMs from a backend pool, and the load balancer will automatically distribute traffic across the available servers.
5. **Cost-effectiveness:** Azure Load Balancer is a cost-effective solution for businesses that need to improve their application performance and availability. The Azure portal provides users with real-time insights into load balancer performance, allowing them to optimize their configurations and reduce costs by using only the resources they need.