Load Balancer

Create Load balancer.

1.

SKU

* Standard (Distribute traffic to backend resources)

Type

* Public

Tier

* Regional

1. **SKU: Standard**
   * **Purpose**: Distributes traffic to backend resources.
   * **Features**: Offers higher performance and more features compared to the Basic SKU, including support for Availability Zones, more backend pool instances, and better security.
2. **Type: Public**
   * **Purpose**: Distributes incoming internet traffic to your Azure resources.
   * **Use Case**: Ideal for applications that need to be accessible from the internet.
3. **Tier: Regional**
   * **Scope**: Operates within a single Azure region.
   * **Benefits**: Provides low-latency and high-performance load balancing within the specified region.
   * **Backend Pool**: Only virtual machines (VMs) and other resources within that specific Azure region can be added to the backend pool of the load balancer. This ensures that the load balancer distributes traffic only to resources located within the same region.
   * **Access to Load Balancer**: The load balancer itself can be accessed from anywhere, including from other regions or the internet, if it's a public load balancer. However, it will only distribute traffic to the backend resources within its own region.

A screenshot of a computer

Description automatically generated

Configure Frontend IP for the load balancer.

A screenshot of a computer

Description automatically generated

Configure Backend either by NIC if in same VNET or can do it by by adding their IP.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Add Load balancing rule.

A screenshot of a computer

Description automatically generated

Define which frontend ip client will hit for getting response.

Frontend port and backend port.

Which backend pool request will hit.

A screenshot of a computer

Description automatically generated

Define health prob.

A screenshot of a computer screen

Description automatically generated

Define session persistence for session stickiness.

A screenshot of a computer

Description automatically generated

Let's break down the differences between an **inbound NAT rule** and a **load balancing rule** for an Azure Load Balancer:

**Inbound NAT Rule**

* **Purpose**: Used for port forwarding. It forwards traffic from a specific frontend IP address and port combination to a specific backend instance and port.
* **Use Case**: Typically used to provide direct access to individual virtual machines (VMs) in the backend pool. For example, you might use an inbound NAT rule to allow remote desktop access to a specific VM.
* **Functionality**: When traffic arrives at the load balancer's frontend IP and port, the inbound NAT rule directs it to a designated VM and port in the backend pool

1

2

.

* **No Health Probe**: Unlike load balancing rules, inbound NAT rules do not require a health probe to function

1

.

**Load Balancing Rule**

* **Purpose**: Distributes incoming traffic across multiple backend instances to ensure high availability and reliability.
* **Use Case**: Ideal for balancing traffic for web applications, APIs, or any service where you want to distribute the load evenly across multiple VMs.
* **Functionality**: When traffic arrives at the load balancer's frontend IP and port, the load balancing rule distributes it across all healthy instances in the backend pool based on the configured load balancing algorithm (e.g., round-robin, hash-based distribution)

1

2

.

* **Health Probe**: Requires a health probe to monitor the health of backend instances and ensure traffic is only sent to healthy VMs

1

.

**Summary**

* **Inbound NAT Rule**: Directs traffic to a specific VM and port, used for scenarios like remote access.
* **Load Balancing Rule**: Distributes traffic across multiple VMs, used for balancing load and ensuring high availability.

In our case we don’t required NAT-inbound.

Outbound configuration.

An **outbound rule** for an Azure Load Balancer is used to manage and control outbound network traffic from your backend instances to the internet or other external endpoints. Here are the key points about outbound rules:

**Purpose**

* **Source Network Address Translation (SNAT)**: Outbound rules define how the source IP addresses of outbound traffic from your backend instances are translated to the public IP address of the load balancer

1

.

* **Simplified Management**: They help simplify the management of outbound connectivity by reducing the number of public IP addresses needed and making it easier to manage allowlists

1

.

**Functionality**

* **Outbound Connectivity**: Ensures that instances in the backend pool can communicate with external endpoints, such as the internet, using the load balancer's public IP address

2

.

* **Idle Timeout and TCP Reset**: Allows you to configure the idle timeout for outbound connections and decide whether to send a TCP reset when the connection times out

1

.

* **Port Allocation**: Provides control over how SNAT ports are allocated among backend instances, which can be useful for optimizing performance and managing port usage

1

.

**Use Cases**

* **Internet Access**: Enables VMs in a private network to access the internet without needing individual public IP addresses.
* **Security**: Helps in securing outbound traffic by controlling which public IP addresses are used for outbound connections, making it easier to manage firewall rules and allowlists

1

.

**Summary**

* **Outbound Rule**: Manages outbound traffic, translating internal IP addresses to the load balancer's public IP, and provides control over connection settings and port allocation.

If you have any more questions or need further clarification, feel free to ask!

For now we don’t required this as well.

Create. And browse by lb frontend ip.

A screenshot of a computer

AI-generated content may be incorrect.

**If any old basic load balancer how to upgrade to standard or any?**

->

**Here are the steps you can follow to upgrade your Basic Load Balancer with PowerShell:**

**1. Install the upgrade module**

Run

Install-Module -Name AzureBasicLoadBalancerUpgrade -Scope CurrentUser -Repository PSGallery -Force

**2. Connect to the subscription of the Basic Load Balancer**

Run

Set-AzContext -Subscription abe10fc9-2089-4a76-8fb7-e9a93870bafc

**3. Upgrade from a Basic Load Balancer to a Standard Load Balancer**

Run

Start-AzBasicLoadBalancerUpgrade -ResourceGroupName Nacha-Test -BasicLoadBalancerName LB1

For more detailed information, you can refer to the following resources:

* [Upgrade a basic load balancer with PowerShell](https://go.microsoft.com/fwlink/?linkid=2260415)
* [Azure Load Balancer Basic to Standard Migration walk-through video](https://learn-video.azurefd.net/vod/player?id=8e203b99-41ff-4454-9cbd-58856708f1c6)

A screenshot of a computer

AI-generated content may be incorrect.

Run those cmd on cloudshell you will be all done.

>> az network lb show --resource-group Nacha-Test --name LB1 --query "sku.name"

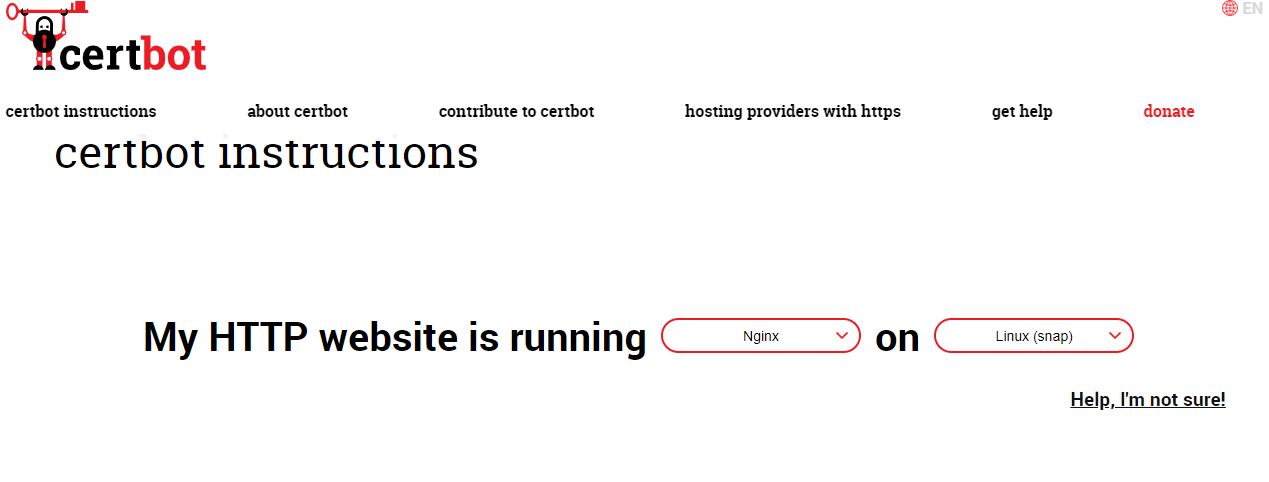
A screenshot of a computer

AI-generated content may be incorrect.

Now site is not secured. Create and assign SSL certificate to make the site secure.

Go to Certboat site.

url- [Certbot](https://certbot.eff.org/)



During installation it will fetch the domain name for the server. we can go with go daddy or any other domain name provider.

For now we can give a try by using azure private DNS.

A screen shot of a computer

AI-generated content may be incorrect.

Without domain name assigned SSL certificate will fail.

Create private DNS zone and assig the name to the server

A screenshot of a computer

AI-generated content may be incorrect.

Create record for the server.

A screenshot of a computer

AI-generated content may be incorrect.

Now try to get by domain name. as it’s the private DNS, so vm in same vnet and subnet can browse by domain name.

Create a windows vm in the same and try.

Getting by IP but, not by the domain name.

A screenshot of a computer

AI-generated content may be incorrect.

Create virtual network integration.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a network link

AI-generated content may be incorrect.

So, the vm’s in same vnet can communicate.

A screenshot of a computer

AI-generated content may be incorrect.

If we want to get it from our vm. Then from that integrated vnet do VPN connection to our server.

Let’s try to create SSL certificate again from certbot.

No, it don’t get for private DNS.

A computer screen with white text

AI-generated content may be incorrect.

We must go with Godaddy or any public DNS.