Microsoft

Designing and Implementing Microsoft Azure Networking Solutions

Exam Ref AZ-700

Charles Pluta

FREE SAMPLE CHAPTER

Microsoft

Exam Ref AZ-700 Designing and Implementing Microsoft Azure Networking Solutions

Charles Pluta

Exam Ref AZ-700 Designing and Implementing Microsoft Azure Networking Solutions

Published with the authorization of Microsoft Corporation by:
Pearson Education, Inc.

Copyright © 2022 by Pearson Education, Inc.

All rights reserved. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, request forms, and the appropriate contacts within the Pearson Education Global Rights & Permissions Department, please visit www.pearson.com/permissions.

No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained herein.

ISBN-13: 978-0-13-768277-5
ISBN-10: 0-13-768277-8

Library of Congress Control Number: 2022933634

ScoutAutomatedPrintCode

TRADEMARKS

Microsoft and the trademarks listed at http://www.microsoft.com on the "Trademarks" webpage are trademarks of the Microsoft group of companies. All other marks are property of their respective owners.

WARNING AND DISCLAIMER

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an "as is" basis. The author, the publisher, and Microsoft Corporation shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the programs accompanying it.

SPECIAL SALES

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact intlcs@pearson.com.

CREDITS

EDITOR-IN-CHIEF
Brett Bartow

EXECUTIVE EDITOR
Loretta Yates

SPONSORING EDITOR
Charvi Arora

DEVELOPMENT EDITOR
Songlin Qiu

MANAGING EDITOR
Sandra Schroeder

SENIOR PROJECT EDITOR
Tracey Croom

COPY EDITOR
Scout Festa

INDEXER
Timothy Wright

PROOFREADER
Gill Editorial Services

TECHNICAL EDITOR
Thomas Palathra

EDITORIAL ASSISTANT
Cindy Teeters

COVER DESIGNER
Twist Creative, Seattle

COMPOSITOR
codeMantra

Pearson's Commitment to Diversity, Equity, and Inclusion

Pearson is dedicated to creating bias-free content that reflects the diversity of all learners. We embrace the many dimensions of diversity, including but not limited to race, ethnicity, gender, socioeconomic status, ability, age, sexual orientation, and religious or political beliefs.

Education is a powerful force for equity and change in our world. It has the potential to deliver opportunities that improve lives and enable economic mobility. As we work with authors to create content for every product and service, we acknowledge our responsibility to demonstrate inclusivity and incorporate diverse scholarship so that everyone can achieve their potential through learning. As the world's leading learning company, we have a duty to help drive change and live up to our purpose to help more people create a better life for themselves and to create a better world.

Our ambition is to purposefully contribute to a world where:

- Everyone has an equitable and lifelong opportunity to succeed through learning.
- Our educational products and services are inclusive and represent the rich diversity of learners.
- Our educational content accurately reflects the histories and experiences of the learners we serve.
- Our educational content prompts deeper discussions with learners and motivates them to expand their own learning (and worldview).

While we work hard to present unbiased content, we want to hear from you about any concerns or needs with this Pearson product so that we can investigate and address them.

- Please contact us with concerns about any potential bias at https://www.pearson.com/report-bias.html.

Contents at a glance

Contents

Acknowledgments

I would like to thank my wife, Jennifer, for being supportive and putting up with the odd hours getting this book finished. To Elias Mereb and Brian Svidergol, thank you for the years of friendship, conferences, dinners, and everything else. And to my friends and colleagues Ed Gale, Joshua Waddell, and Aaron Lines, thank you for your friendship, mentorship, and advice the last couple of years. To all the cloud professionals and readers of this book, thank you for taking the time to read, explore, learn, test, and "play around" with these technologies while you are learning. Keep it up, and good luck!

About the Author

CHARLES PLUTA is a technical consultant and Microsoft Certified Trainer who has authored several certification exams, lab guides, and learner guides for various technology vendors. As a technical consultant, Charles has assisted small, medium, and large organizations in deploying and maintaining their IT infrastructure. He is also a speaker, staff member, or trainer at several large industry conferences every year. Charles has a degree in Computer Networking and holds over 25 industry certifications. He makes a point to leave the United States to travel to a different country once every year. When not working on training or traveling, he plays pool in Augusta, Georgia.

Introduction

This book takes a high-level approach to the list of topics and skills measured on the Designing and Implementing Microsoft Azure Networking Solutions (AZ-700) exam, which is required to obtain the Microsoft Certified: Azure Network Engineer Associate certification. This exam focuses on networking topics in Microsoft Azure, but it does require additional knowledge of the Azure Portal and related services such as virtual machines, storage accounts, monitoring tools, and more. If you are not already familiar with general Azure services, I suggest that you begin with Azure Fundamentals (AZ-900) or Azure Administrator (AZ-104) before focusing on this exam.

This book provides step-by-step examples of configuring the Azure services that are outlined in the topic list using the Azure Portal. However, the certification exam could also ask you questions about PowerShell or CLI commands that perform the same actions as those in the portal. You should use this book as a supplement to your learning journey and practice other methods of configuring these services in addition to what is outlined in the book.

This book covers every major topic area found on the exam, but it does not cover every exam question. Only the Microsoft exam team has access to the exam questions, and Microsoft regularly adds new questions to the exam, making it impossible to cover specific questions. You should consider this book a supplement to your relevant real-world experience and other study materials. If you encounter a topic in this book that you do not feel completely comfortable with, use the “Need more review?” links you'll find in the text to find more information, and take the time to research and study the topic. Great information is available on MSDN, on TechNet, and in blogs and forums.

Organization of this book

This book is organized by the “Skills measured” list published for the exam. The “Skills measured” list is available for each exam on the Microsoft Learn website: *http://aka.ms/examlist*. Each chapter in this book corresponds to a major topic area in the list, and the technical tasks in each topic area determine a chapter’s organization. If an exam covers six major topic areas, for example, the book will contain six chapters.

Microsoft certifications

Microsoft certifications distinguish you by proving your command of a broad set of skills and experience with current Microsoft products and technologies. The exams and corresponding certifications are developed to validate your mastery of critical competencies as you design and develop, or implement and support, solutions with Microsoft products and technologies both on-premises and in the cloud. Certification brings a variety of benefits to the individual and to employers and organizations.

NEED MORE REVIEW? **ALL MICROSOFT CERTIFICATIONS**

For information about Microsoft certifications, including a full list of available certifications, go to *http://www.microsoft.com/learn*.

Check back often to see what is new!

Errata, updates, and book support

We've made every effort to ensure the accuracy of this book and its companion content. You can access updates to this book—in the form of a list of submitted errata and their related corrections—at:

MicrosoftPressStore.com/ExamRefAZ700/errata

If you discover an error that is not already listed, please submit it to us at the same page.

For additional book support and information, please visit

MicrosoftPressStore.com/Support

Please note that product support for Microsoft software and hardware is not offered through the previous addresses. For help with Microsoft software or hardware, go to *http://support.microsoft.com*.

Stay in touch

Let's keep the conversation going! We're on Twitter: *http://twitter.com/MicrosoftPress*.

This page intentionally left blank

CHAPTER 3

Design and implement routing

Routing traffic is a core component of any type of application deployment in your Azure environment. This can be an “east-west” type of traffic within the subscription, from one virtual network to another network or service. Or it can be a “north-south” type of traffic that flows in and out of the application.

The traffic flow in your virtual network can be manipulated by using user-defined routes (UDRs). Then, depending on the scenario, incoming traffic can be distributed across multiple backend resources using load balancers, application gateways, Azure Front Door, or Azure Traffic Manager. For outgoing traffic, you can represent multiple virtual machines with a single public IP using a NAT gateway. All of these services will be discussed in this chapter.

Skills in this chapter:

- Skill 3.1: Design, implement, and manage virtual network routing
- Skill 3.2: Design and implement an Azure load balancer
- Skill 3.3: Design and implement Azure Application Gateway
- Skill 3.4: Implement Azure Front Door
- Skill 3.5: Implement an Azure Traffic Manager profile
- Skill 3.6: Design and implement an Azure Virtual Network NAT

Skill 3.1: Design, implement, and manage virtual network routing

Routing is one of the core network infrastructure components that define how traffic leaves point A and arrives at point B. When you create a virtual machine in Azure, by default it can communicate outbound to the internet. In most organizations, there are security or compliance policies that require this traffic to be inspected or audited before going out to the internet. In these scenarios, it is required to create custom route tables to modify the traffic flow based on the destination. In this skill section, we discuss creating these UDRs and how to configure custom routing for a virtual network.

This skill covers how to:

- Design and implement user-defined routes
- Associate a route table with a subnet
- Configure forced tunneling
- Diagnose and resolve routing issues

Design and implement user-defined routes

When you create a virtual network, there are default system routes that define how the resources attached to the virtual network can communicate with Azure services and the internet. Table 3-1 outlines the default system routes for a virtual network.

TABLE 3-1 Virtual network default routes

Destination address range	Next hop
Address space of the network	Virtual network
0.0.0.0/0	Internet
10.0.0.0/8	None
192.168.0.0/16	None
100.64.0.0/10	None

The first default route is determined by the address space that has been configured for the virtual network. Next, as long as the traffic is not destined to a reserved network from RFC 1918 or RFC 6598, then the traffic is sent to the internet. If you add any address ranges that are by default set to None to the virtual network, then it will change the next hop to *virtual network*.

If you add virtual network peering or a virtual network gateway or you configure service endpoints, the default routes are modified to include the service that you configure. When you configure virtual network peering, a default route is added for the address space of the peered virtual network, which is the underlying reason why peered virtual networks cannot have overlapping address spaces. Figure 3-1 outlines the bi-directional communication with virtual network peering.

Let’s assume that in Figure 3-1, VM1 has an IP address of 10.0.0.4 and VM2 has an IP address of 10.1.0.4. The traffic flow would resemble this path:

1. Packets leave VM1 destined to VM2 through the default gateway of 10.0.0.1.
2. The gateway has a route to 10.1.0.0/16 through peering and forwards the packet to 10.1.0.1.

3. The virtual network at 10.1.0.0/16 recognizes that the packet is destined to VM2 and forwards accordingly.
4. VM2 receives the packet from VM1.

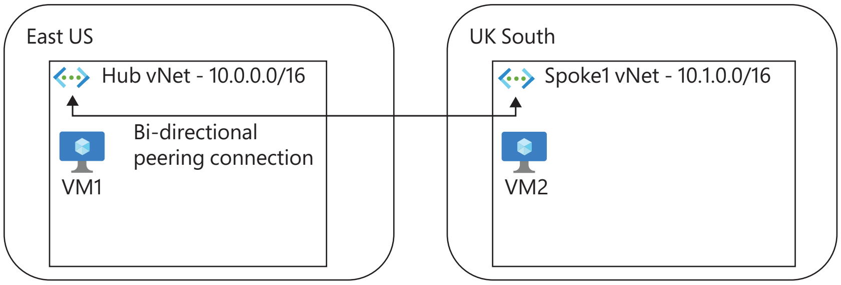


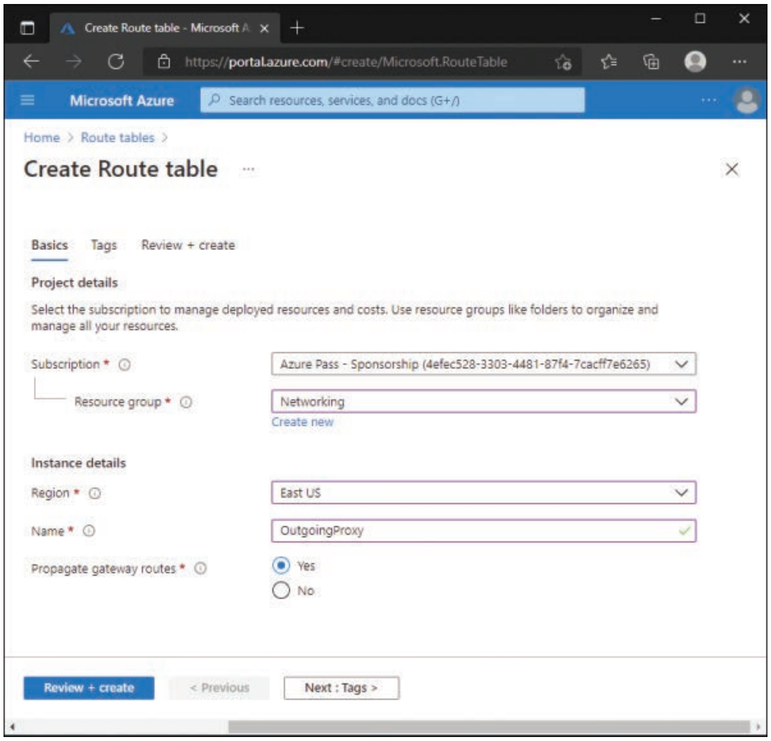
FIGURE 3-1 Bi-directional communication with virtual network peering

By default, when you deploy a virtual machine, it can and will communicate with the internet outbound through an Azure IP address, even if the virtual network does not have a firewall associated or if the VM does not have a public IP address assigned. The IP address that the VM would display to internet services would vary depending on the region.

There are several scenarios in which an organization will want to change this traffic flow and not allow VMs to communicate directly with the internet. Or, even if you decide to peer two virtual networks, instead of allowing all default communication, the traffic is first forwarded through a firewall or other type of network virtual appliance. The override mechanism for these scenarios is named a UDR, which you configure by using an Azure route table.

To create a route table, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Route tables**.
3. On the Route tables page, click **Create**.
4. In the Subscription dropdown, select the subscription to associate with the route table. This must be the same subscription as the virtual network.
5. In the Resource group dropdown, select a resource group to associate with the resource, such as **Networking**.
6. In the Region dropdown list, select the region to associate with the route table. This must also be the same location as the virtual network. In this example, select **East US**.
7. In the Name field, provide a name for the route table, such as **OutgoingProxy**.
8. Leave the default of **Yes** selected for Propagate gateway routes.
9. Click **Review + Create**, and then click **Create**. Figure 3-2 displays the completed configuration.


Create Route table - Microsoft A
https://portal.azure.com/#create/Microsoft.RouteTable
Microsoft Azure
Search resources, services, and docs (G+/)
Home > Route tables >
Review + create
< Previous
Next : Tags >

Create Route table

Basics Tags Review + create

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * Azure Pass - Sponsorship (4efec528-3303-4481-87f4-7cacff7e6265)

Resource group * Networking

Create new

Instance details

Region * East US

Name * OutgoingProxy

Propagate gateway routes * [✓ Yes] No

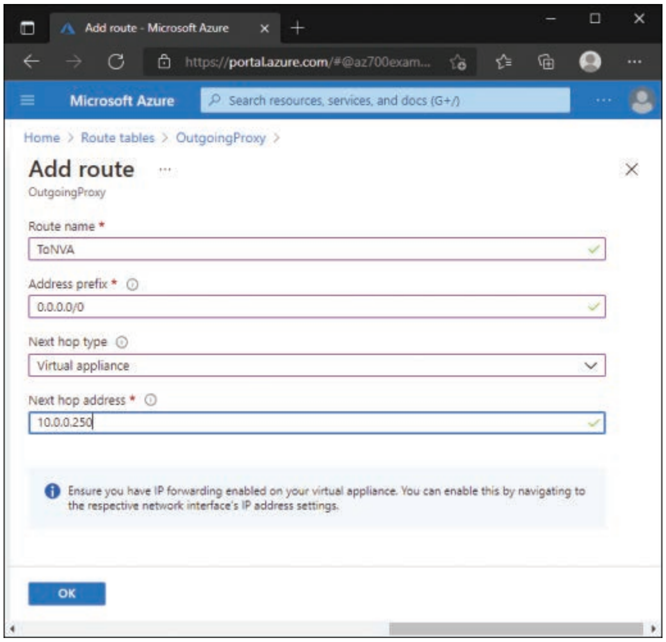
FIGURE 3-2 Create Route table

This creates a route table object in the specified subscription and region. Based on the name we provided for the route table, OutgoingProxy, we'll use the scenario of two VMs in different virtual networks and force outbound traffic through a network virtual appliance.

After creating the route table, there are two more steps to make the desired effect of changing the traffic flow: creating a route within the route table and associating the table with a subnet. To create a route within the table, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Route tables**.
3. Select the **OutgoingProxy** route table that was previously created.
4. From the table, click the **Routes** blade.
5. On the Routes blade, click **Add**.
6. In the Route name field, name the route **ToNVA**.
7. In the Address prefix field, enter the destination prefix that you want to modify traffic flow for. For example, to force all traffic through an NVA, specify **0.0.0.0/0**.
8. In the Next hop type dropdown, select **Virtual appliance**.
9. In the Next hop address field, specify the IP address of the NVA that will act as the proxy. For example, enter **10.0.0.250**.

10. Click **OK**. Figure 3-3 displays the completed route entry.


Add route - Microsoft Azure
https://portal.azure.com/#@az700exam...
Microsoft Azure
Search resources, services, and docs (G+/)
Home > Route tables > OutgoingProxy >

Add route

OutgoingProxy

Route name *: ToNVA

Address prefix *: 0.0.0.0/0

Next hop type: Virtual appliance

Next hop address *: 10.0.0.250

Ensure you have IP forwarding enabled on your virtual appliance. You can enable this by navigating to the respective network interface's IP address settings.

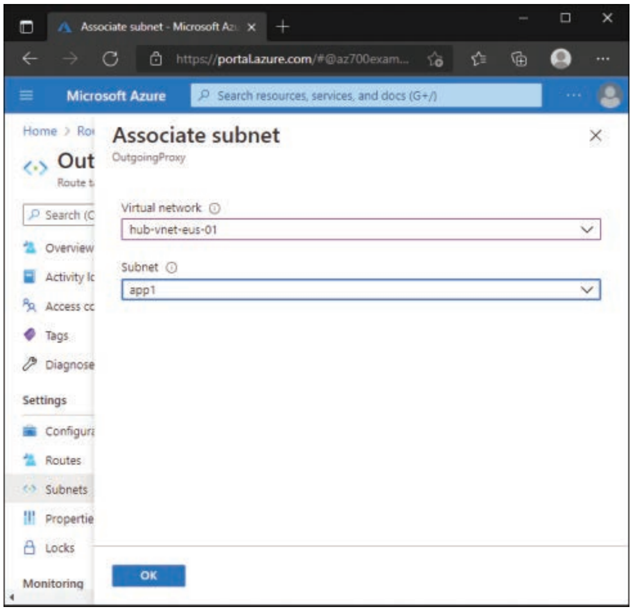
FIGURE 3-3 Add route entry

Associate a route table with a subnet

After you create the route table and add a route entry to the table, the final step to change the traffic flow is to associate the table with the subnets that you want to change the flow of traffic from.

To associate the table with a subnet, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Route tables**.
3. Select the **OutgoingProxy** route table that was previously created.
4. From the table, click the **Subnets** blade.
5. On the Subnets blade, click **Associate**.
6. In the Virtual network dropdown, select the network with the subnet to associate with the table; for example, select **hub-vnet-eus-01**.
7. In the Subnet dropdown, select the subnet to associate with the route table and modify the traffic flow for, such as **app1**.
8. Click **OK**. Figure 3-4 displays the completed configuration.



Associate subnet
OutgoingProxy

Virtual network: hub-vnet-eus-01

Subnet: app1

FIGURE 3-4 Associate subnet

In the scenario that the traffic from VM1 needs to flow through the NVA before communicating with VM2, we have made the route table entry and association if VM1 is in the app1 subnet. Table 3-2 outlines the IP address for the VMs in Figure 3-5.

TABLE 3-2 IP addresses

VM	IP address
VM1	10.0.0.4
NVA	10.0.0.250
VM2	10.1.0.4

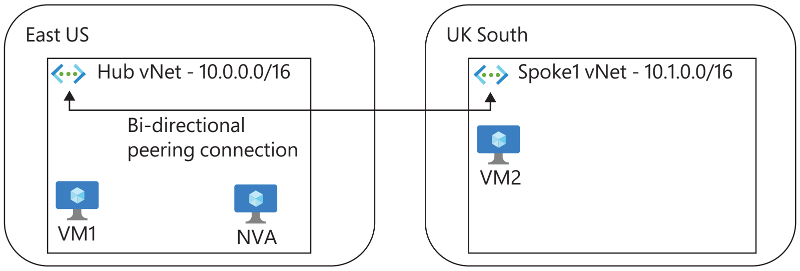


FIGURE 3-5 NVA proxy diagram

With the route table association to the app1 subnet, assuming VM1 is in the app1 subnet, then any traffic destined to VM2 will flow through the NVA. This NVA could be a proxy, a firewall, or any third-party appliance from Azure Marketplace.

Configure forced tunneling

Forced tunneling is a similar concept to using a network virtual appliance to route all traffic through as a proxy, but it incorporates either a site-to-site VPN or ExpressRoute to route any virtual network traffic back on-premises. This is typically designed in regulated or high-security environments where using the existing infrastructure is more convenient short-term than duplicating the requirements in the cloud.

For a forced tunneling design, the route table entry is similar to using an NVA. You change the next hop IP address to the on-premises address that is accessible through the VPN tunnel. Figure 3-6 represents the design of forced tunneling to on-premises.

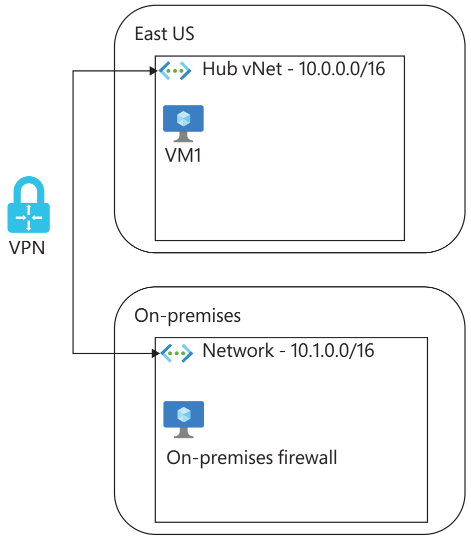


FIGURE 3-6 Forced tunneling diagram

If you are using ExpressRoute, either in addition to or instead of a VPN, the concept is the same except that the default route should be advertised as part of the BGP peering sessions. This does not require any additional route entry into a route table.

Diagnose and resolve routing issues

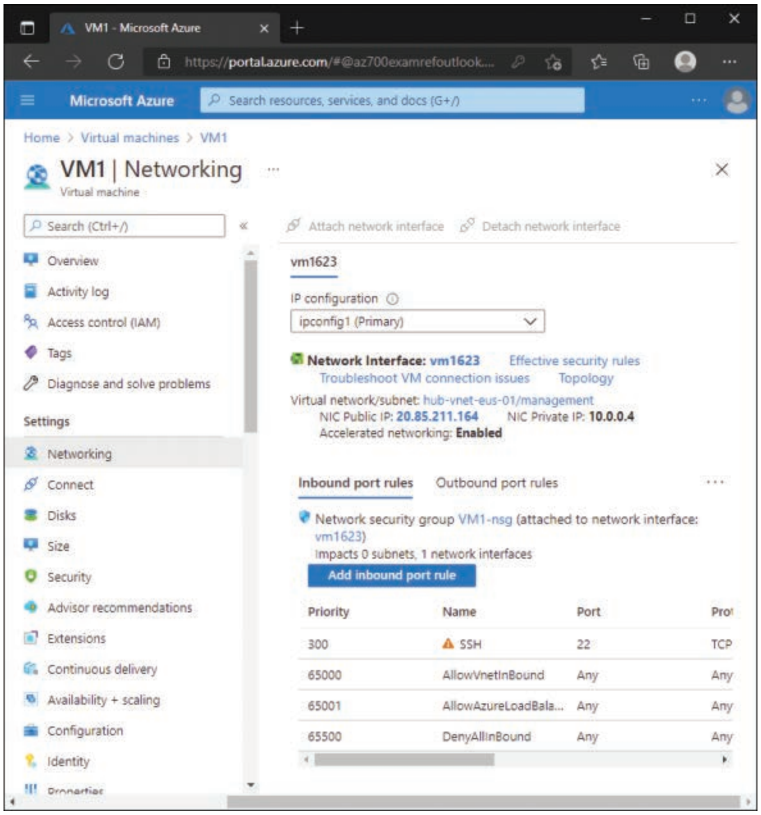
Routes in Azure are processed in the order of longest prefix match for any destination that has been defined. If a route has more than one match, then the processing priority is:

1. User-defined routes
2. BGP routes
3. System routes

If you need to troubleshoot the routing from a virtual machine perspective, there are two built-in tools in the Azure portal that can assist. From a VM, you can view the effective routes that apply to the VM based on the subnet that the NIC is associated with.

To view the effective routes, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Virtual machines**.
3. Select a virtual machine that you have deployed, for example **VM1**.
4. From the VM, click the **Networking** blade.
5. On the Networking blade, click the name of the network interface attached to the VM. Figure 3-7 shows an example of a NIC named **vm1623**.



IP configuration: ipconfig1 (Primary)

Network Interface: vm1623

Virtual network/subnet: hub-vnet-eus-01/management

NIC Public IP: 20.85.211.164   NIC Private IP: 10.0.0.4

Accelerated networking: Enabled

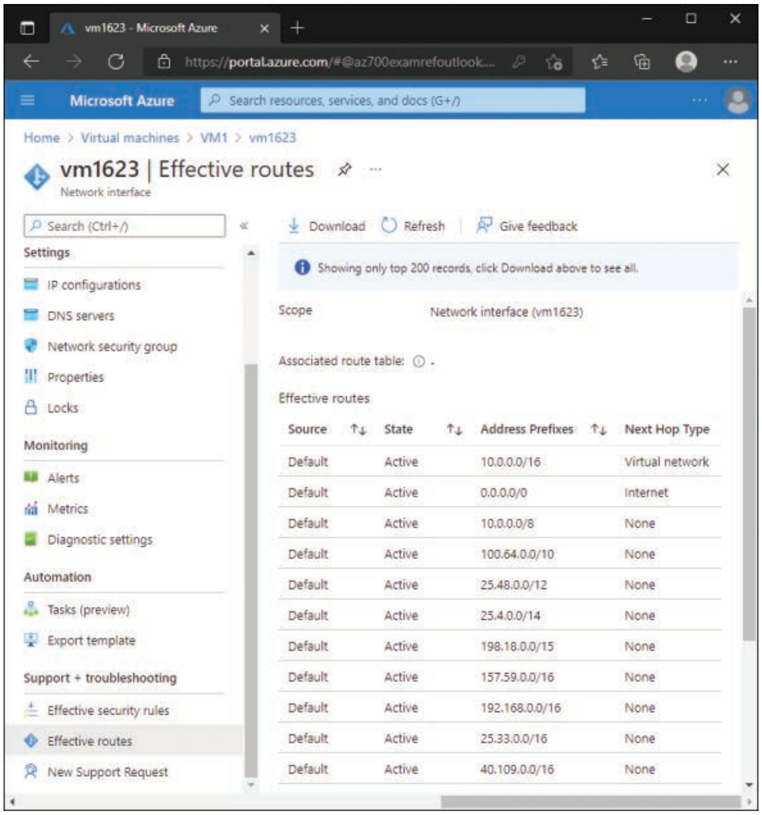
Network security group VM1-nsg (attached to network interface: vm1623)

Impacts 0 subnets, 1 network interfaces

Priority	Name	Port	Prot
300	SSH	22	TCP
65000	AllowVnetInBound	Any	Any
65001	AllowAzureLoadBala...	Any	Any
65500	DenyAllInBound	Any	Any

FIGURE 3-7 VM Networking

6. From the network interface, click the **Effective routes** blade. Figure 3-8 displays the effective routes for the network interface.



Showing only top 200 records, click Download above to see all.

Scope Network interface (vm1623)

Associated route table: -

Effective routes

Source	State	Address Prefixes	Next Hop Type
Default	Active	10.0.0.0/16	Virtual network
Default	Active	0.0.0.0/0	Internet
Default	Active	10.0.0.0/8	None
Default	Active	100.64.0.0/10	None
Default	Active	25.48.0.0/12	None
Default	Active	25.4.0.0/14	None
Default	Active	198.18.0.0/15	None
Default	Active	157.59.0.0/16	None
Default	Active	192.168.0.0/16	None
Default	Active	25.33.0.0/16	None
Default	Active	40.109.0.0/16	None

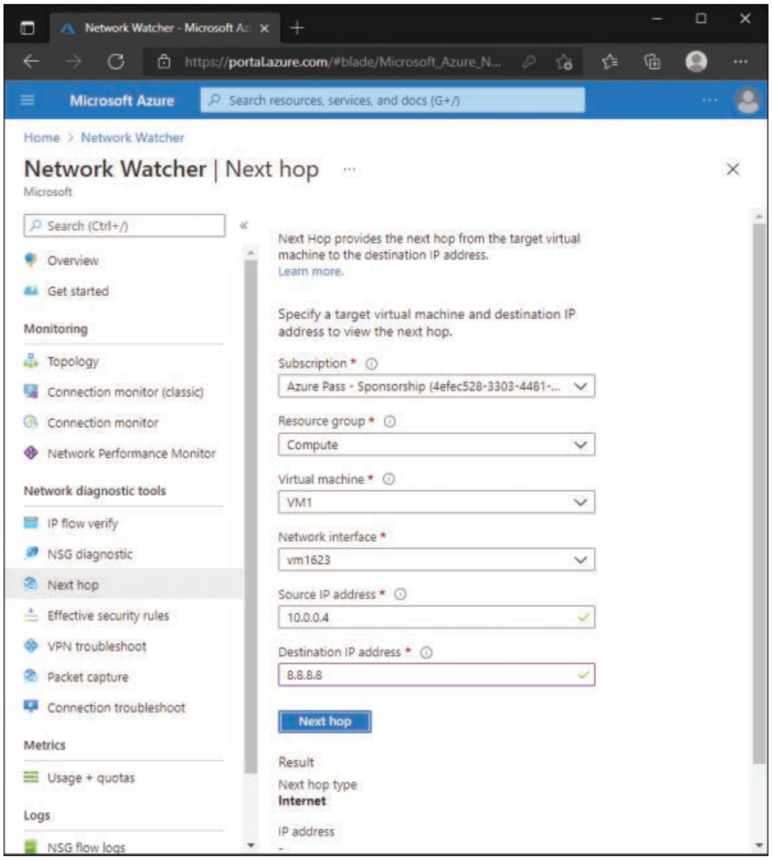
FIGURE 3-8 Effective routes

The Effective routes blade on the network interface card is a useful tool for an overview of the routes that are associated with the subnet that the NIC is associated with. Another method of troubleshooting the routing from a VM is to use Network Watcher.

Network Watcher is an Azure monitoring resource that is built into the Azure portal and is a collection of tools that can be used for troubleshooting. One of these tools, named Next hop, allows you to select a VM and identify the next hop for a specific address.

To use the Next hop tool, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Network Watcher**.
3. From Network Watcher, select the **Next hop** blade.
4. Complete the fields to select the desired virtual machine and network interface card.
5. In the Destination IP address field, specify the IP address that you want to test to identify what the configured next hop would be. For example, specify **8.8.8.8**.
6. Click **Next hop**. Figure 3-9 shows the completed form and that the next hop type is the internet based on the effective routes.



Network Watcher | Next hop

Next Hop provides the next hop from the target virtual machine to the destination IP address.
Learn more.

Specify a target virtual machine and destination IP address to view the next hop.

Subscription *: Azure Pass - Sponsorship (4efec528-3303-4481-...

Resource group *: Compute

Virtual machine *: VM1

Network interface *: vm1623

Source IP address *: 10.0.0.4

Destination IP address *: 8.8.8.8

Next hop

Result

Next hop type
Internet

IP address
-

FIGURE 3-9 Next hop

Figure 3-9 shows that from the network interface card vm1623, which has a source IP address of 10.0.0.4 on the management subnet, if the VM were to communicate with the 8.8.8.8 destination IP address, the traffic would be forwarded to the internet.

Skill 3.2: Design and implement an Azure load balancer

There are a few options in Azure if you need to distribute incoming traffic to a pool of back-end resources. These options include Azure Load Balancer, Azure Application Gateway, and Azure Front Door. Each service has its own set of features and functionality that sets it apart. Azure Load Balancer is the simplest of the three options; it provides core layer 4 load balancing functionality for both public and internal scenarios.

This skill covers how to:

- Choose an Azure Load Balancer SKU
- Choose between public and internal
- Create and configure an Azure load balancer
- Implement a load balancing rule
- Create and configure inbound NAT rules
- Create explicit outbound rules for a load balancer

Choose an Azure Load Balancer SKU

As you are designing a solution that includes an Azure load balancer, there are two SKU options to select from: Basic and Standard. In general, the best practice is to use a Standard SKU load balancer for any production deployment. After you deploy a load balancer, you cannot change the SKU and must deploy a new resource. Table 3-3 outlines the feature differences between the SKUs.

TABLE 3-3 Load balancer SKUs

Feature	Standard load balancer	Basic load balancer
Backend pool size	Up to 1,000 instances	Up to 300 instances
Backend pool type	Virtual machines, VM scale sets	Availability sets or scale sets
Health probe protocols	TCP, HTTP, HTTPS	TCP, HTTP
Health probe failure behavior	Existing TCP connections stay alive	Single instance failures stay alive, all instance failures drop
Availability zones	Zone-redundant and zonal front-ends can be configured	Not available
Diagnostics	Azure Monitor	Not available
HA ports	With internal IP addresses	Not available
Security defaults	Closed by default	Open by default
Outbound rules	Outbound NAT	Not available
TCP Reset on Idle	Configured on rules	Not available
Multiple front-ends	Inbound and outbound	Inbound only
SLA	99.99%	None
Global virtual network peering	Supported with internal IP address	Not available

Of the feature differences outlined in Table 3-3, the standouts would be the number of instances, HTTPS as a health probe, and the SLA for the Standard SKU. For smaller development, test, quality assurance, or other non-production type environments, a Basic SKU might be acceptable.

Choose between public and internal

Determining whether a load balancer is public or internal is an architecture decision regarding where the load balancer is placed as part of the solution. Both Basic and Standard load balancers support internal and public IP addresses, so there is no placement restriction based on the SKU that you select. Both types of load balancer SKUs support both scenarios of internal or public. Whether you select internal or public determines which type of traffic the load balancer can accept traffic from. Internal load balancers can accept traffic only from private IP addresses, whereas public load balancers can accept traffic only from public IP addresses.

A likely scenario for public load balancers is to also perform outbound NAT for the backend pool virtual machines. However, only the Standard SKU load balancer supports configuring outbound NAT rules. Thus, most public load balancer scenarios would require a Standard SKU load balancer.

Other than the feature differences provided by the two types of SKUs, there is no performance benefit or restriction by choosing to configure a load balancer with a public or private IP address.

Create and configure an Azure load balancer

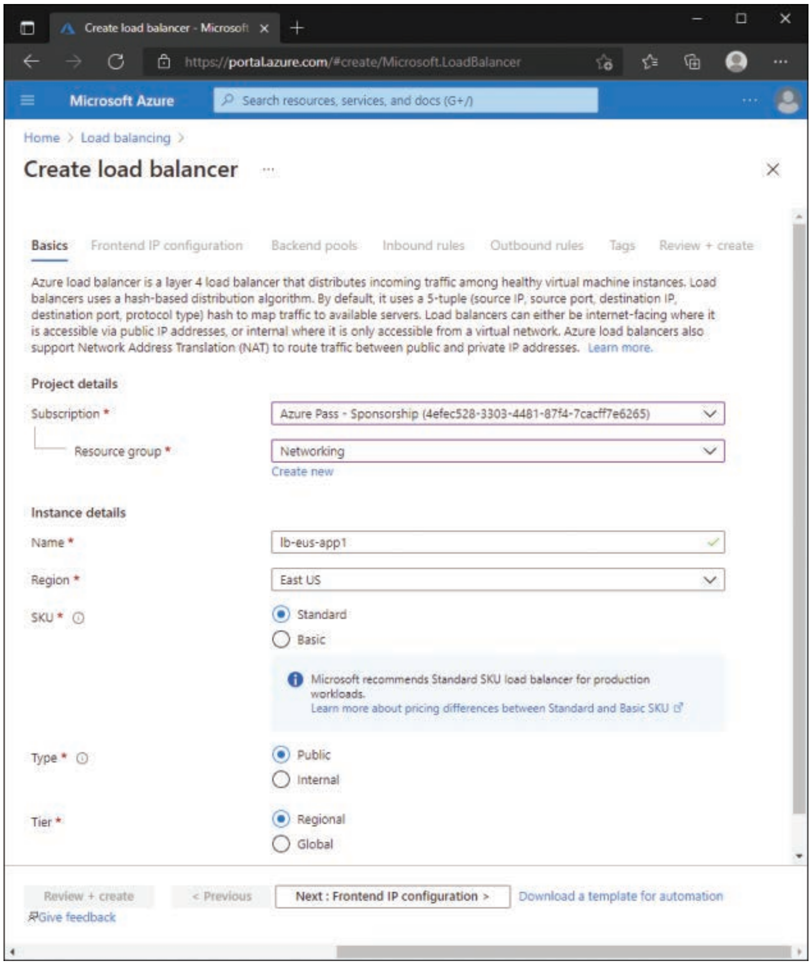
After you choose which SKU to use, and whether your load balancer will be internal or public-facing, you are ready to create the load balancer. To complete the configuration, you will also need virtual machines for the backend pool. There are a few configuration components that make up a complete load balancer configuration:

- Frontend IP address
- Backend pool
- Health probes
- Load balancer rules

In this section, we focus on configuring the frontend IP address and backend pools as we create the load balancer. We'll focus on load balancer health probes and rules in the next section. To create an Azure load balancer, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Load Balancers**.
3. On the Load Balancer page, click **Create**.
4. In the Resource group dropdown, select the desired resource group, such as **Networking**.
5. In the Name field, provide a name for the load balancer, such as **lb-eus-app1**.

6. In the Region dropdown menu, select the desired Azure region, such as **East US**.
7. For the SKU selection, select **Standard**.
8. For the Type selection, select **Public**.
9. For the Tier selection, select **Regional**.
10. Click **Next: Frontend IP Configuration**. Figure 3-10 shows the completed configuration on the Basics tab.


Create load balancer - Microsoft

https://portal.azure.com/#create/Microsoft.LoadBalancer

Microsoft Azure

Search resources, services, and docs (G+/)

Home > Load balancing >

Create load balancer

Basics | Frontend IP configuration | Backend pools | Inbound rules | Outbound rules | Tags | Review + create

Azure load balancer is a layer 4 load balancer that distributes incoming traffic among healthy virtual machine instances. Load balancers uses a hash-based distribution algorithm. By default, it uses a 5-tuple (source IP, source port, destination IP, destination port, protocol type) hash to map traffic to available servers. Load balancers can either be internet-facing where it is accessible via public IP addresses, or internal where it is only accessible from a virtual network. Azure load balancers also support Network Address Translation (NAT) to route traffic between public and private IP addresses. Learn more.

Project details

Subscription *	Azure Pass - Sponsorship (4efec528-3303-4481-87f4-7cacff7e6265)
Resource group *	Networking
	Create new

Instance details

Name *	lb-eus-app1
Region *	East US
SKU *	[✓ Standard] Basic
Type *	[✓ Public] Internal
Tier *	[✓ Regional] Global

Microsoft recommends Standard SKU load balancer for production workloads.
Learn more about pricing differences between Standard and Basic SKU

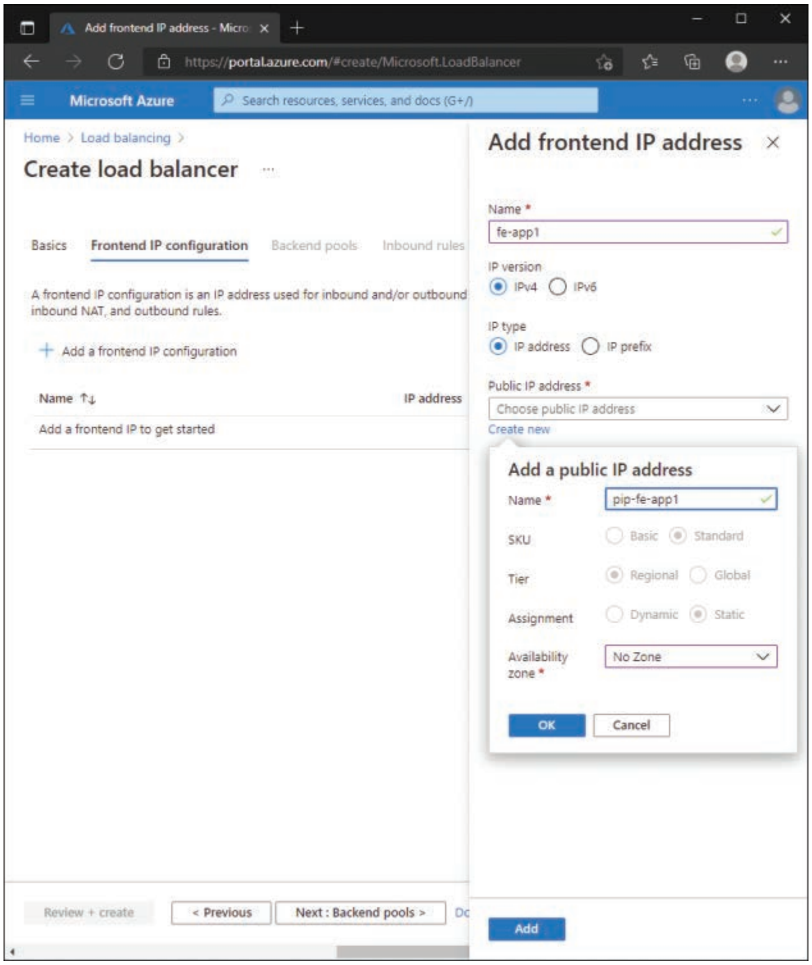
Review + create | < Previous | Next : Frontend IP configuration > | Download a template for automation

Give feedback

FIGURE 3-10 Load balancer basics

11. On the Frontend IP configuration tab, click **Add a frontend IP configuration**.
12. In the Name field, specify a name for the frontend IP address, such as **fe-app1**.
13. For the IP version, select **IPv4**.
14. For the IP type, select **IP address**.
15. For the Public IP address, click **Create new**.

16. In the Name field, provide a name for the IP address, such as **pip-fe-app1**.
17. Click **OK**, and then click **Add**. Figure 3-11 shows the example configuration.



Add frontend IP address

Name *: fe-app1

IP version: [✓ IPv4] IPv6

IP type: [✓ IP address] IP prefix

Public IP address *: Choose public IP address

Add a public IP address

Name *: pip-fe-app1

SKU: Basic [✓ Standard]

Tier: [✓ Regional] Global

Assignment: Dynamic [✓ Static]

Availability zone *: No Zone

Create load balancer

A frontend IP configuration is an IP address used for inbound and/or outbound inbound NAT, and outbound rules.

Add a frontend IP to get started

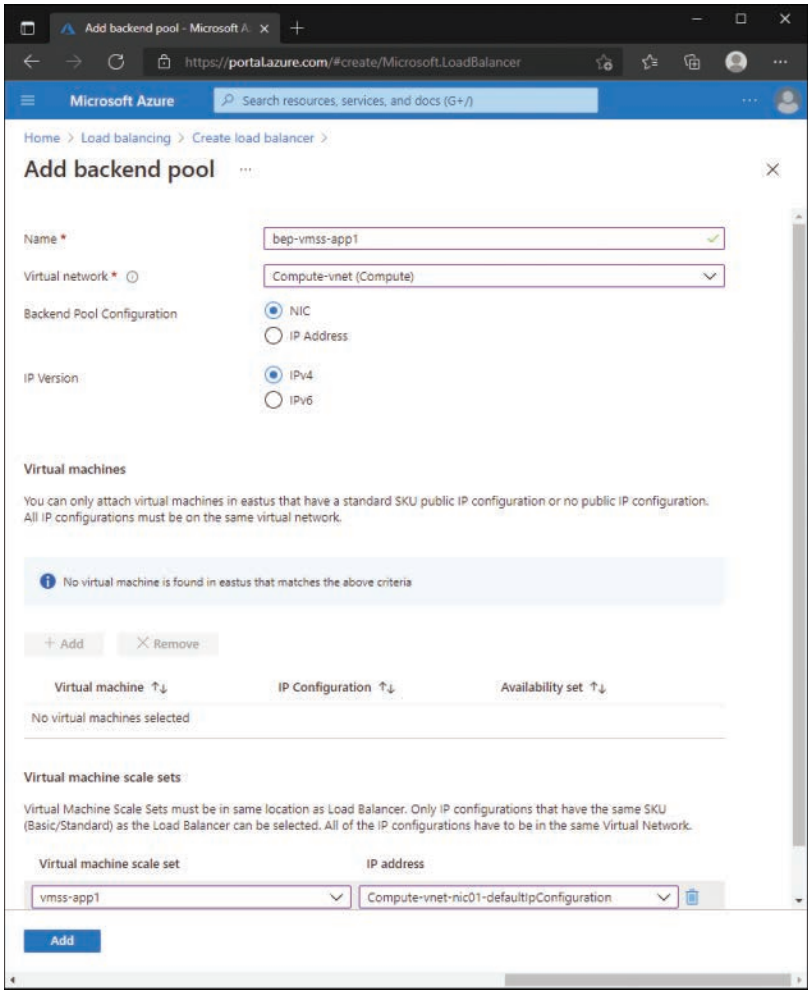
FIGURE 3-11 Load balancer frontend IP configuration

18. Click **Next: Backend pools**.

NOTE **COMPUTE DEPLOYMENT**

These steps assume that you already have compute deployed, either individual virtual machines or a virtual machine scale set. Deploying compute options is not in the exam outline and is not covered in this book.

19. On the Backend pools tab, click **Add a backend pool**.
20. In the Name field, provide a name for the pool, such as **bep-vmss-app1**.
21. In the Virtual network dropdown menu, select the virtual network that the backend pool is connected to, such as **Compute-vnet**.
22. For the Backend Pool Configuration, select **NIC**.
23. For the IP Version, select **IPv4**.
24. Select the backend compute where the application is being run, whether individual virtual machines or a virtual machine scale set. In this example, select a virtual machine scale set named **vmss-app1**. Figure 3-12 displays the completed Add backend pool page.


Add backend pool - Microsoft A
https://portal.azure.com/#create/Microsoft.LoadBalancer
Microsoft Azure
Search resources, services, and docs (G+/)
Home > Load balancing > Create load balancer >

Add backend pool

Name * : bep-vmss-app1

Virtual network * : Compute-vnet (Compute)

Backend Pool Configuration: [✓ NIC] IP Address

IP Version: [✓ IPv4] IPv6

Virtual machines

You can only attach virtual machines in eastus that have a standard SKU public IP configuration or no public IP configuration. All IP configurations must be on the same virtual network.

No virtual machine is found in eastus that matches the above criteria

Virtual machine	IP Configuration	Availability set
No virtual machines selected		

Virtual machine scale sets

Virtual Machine Scale Sets must be in same location as Load Balancer. Only IP configurations that have the same SKU (Basic/Standard) as the Load Balancer can be selected. All of the IP configurations have to be in the same Virtual Network.

Virtual machine scale set	IP address
vmss-app1	Compute-vnet-nic01-defaultIpConfiguration

FIGURE 3-12 Load balancer backend pool

25. Click **Add**.
26. Click **Review + create**, and then click **Create**.

Adding at least one frontend IP address and one backend pool are the minimum requirements for creating a load balancer. For the load balancer to be functional, it will also need rules created.

Implement a load balancing rule

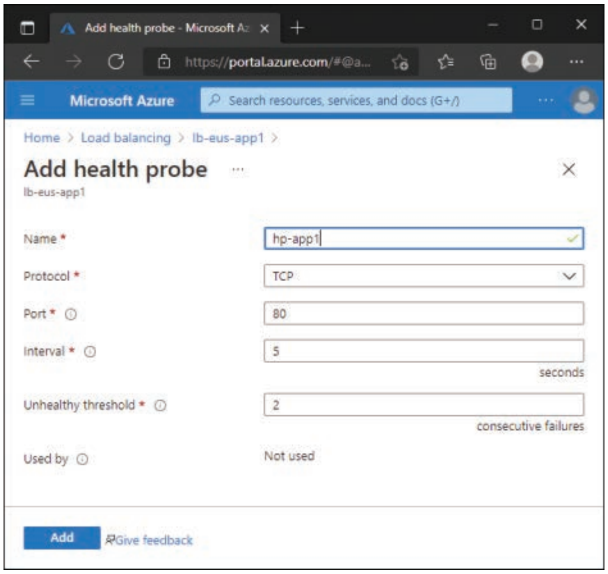
Load balancer rules are the configuration that ties together the frontend IP address, backend pool, health probe, protocol, and port number that you want to accept and distribute traffic on. Basic SKU load balancers can have up to 250 rules configured, and Standard SKU load balancers can have up to 1,500 rules.

Rules also allow you to configure port address translation. For example, if you need to allow port 443 (HTTPS) connections externally, but need to translate that to a custom port, 8443, internally, the load balancer rule can be configured to do this translation.

To create a load balancer rule, you must have a health probe that checks the status of the backend pool. To create a health probe for an existing load balancer, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Load Balancers**.
3. On the Load Balancer page, select an existing load balancer, for example **lb-eus-app1**.
4. On the selected load balancer, click the **Health probes** blade.
5. On the Health probes blade, click **Add**.
6. In the Name field, provide a name such as **hp-app1**.
7. Leave the remaining fields set to the defaults:
   - Protocol: **TCP**
   - Port: **80**
   - Interval: **5**
   - Unhealthy threshold: **2**
8. Click **Save**. Figure 3-13 shows the completed configuration.

The health probe that is configured in Figure 3-13 will communicate with the defined port number, TCP 80, every five seconds. The backend pool that the health probe is associated with through the load balancer rule will be deemed "Healthy" as long as the port number is accessible. If the communication times out or is otherwise unreachable for two consecutive attempts, or in this case ~10 seconds, then the resource in the backend pool would be marked "Unhealthy" and connections would not be forwarded to that resource.


Add health probe - Microsoft Az
https://portal.azure.com/#@a...
Microsoft Azure
Search resources, services, and docs (G+/)
Home > Load balancing > lb-eus-app1 >

Add health probe

lb-eus-app1

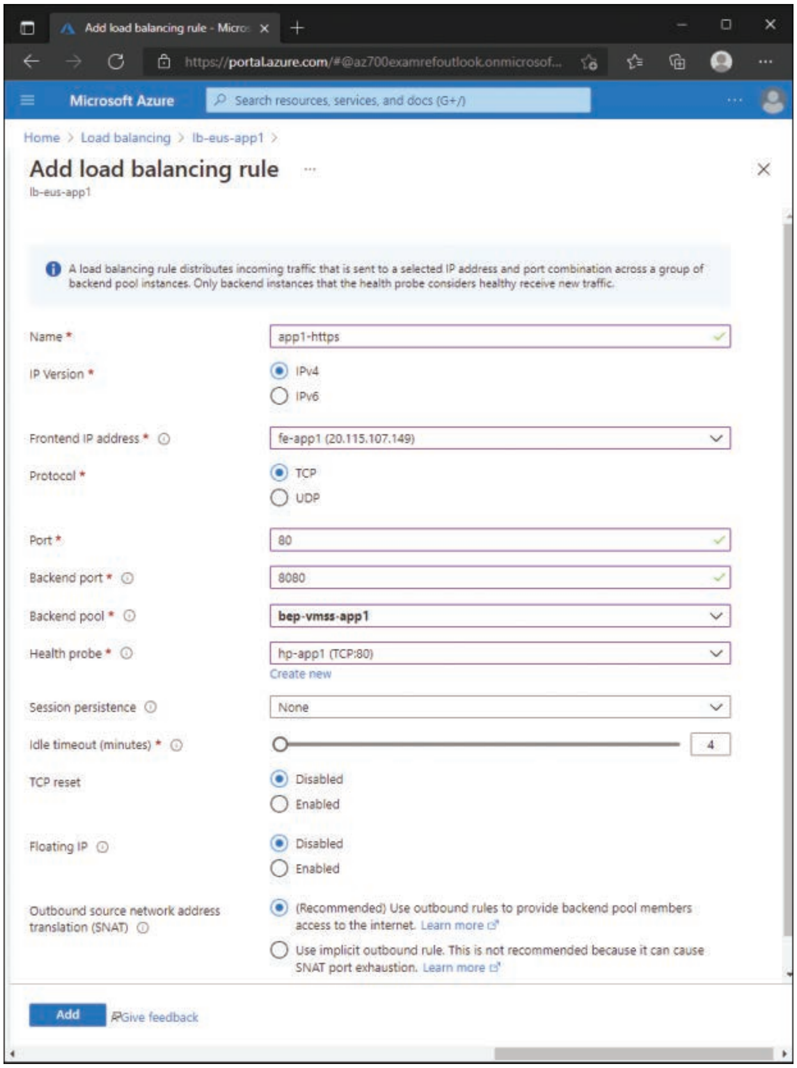
Name *	hp-app1
Protocol *	TCP
Port *	80
Interval *	5 seconds
Unhealthy threshold *	2 consecutive failures
Used by	Not used

FIGURE 3-13 Add health probe

To create a rule that associates these components together for an existing load balancer, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Load Balancers**.
3. On the Load Balancer page, select an existing load balancer, for example **lb-eus-app1**.
4. On the selected load balancer, click the **Load balancing rules** blade.
5. On the Load balancing rules page, click **Add**.
6. In the Name field, provide a name for the rule, for example **app1-https**.
7. In the IP Version field, leave the default **IPv4** selected.
8. In the Frontend IP address dropdown, select the desired IP address to accept connections on. For example, select **fe-app1**.
9. In the Protocol field, leave the default of **TCP** selected.
10. In the Port field, specify the external port number to accept connections on. For example, specify **80**.
11. In the Backend port field, specify the internal or translated port number to forward the connection to on the backend pool. For example, specify **8080**.
12. In the Backend pool dropdown, select the desired backend pool. For example, select **bep-vmss-app1**.
13. In the Health probe dropdown, select the previously created health probe. For example, select **hp-app1**.

14. Leave the remaining fields set to their default values:
   - Idle timeout: **4 minutes**
   - TCP reset: **Disabled**
   - Floating IP: **Disabled**
   - Outbound source NAT: **(Recommended) Use outbound rules to provide backend pool members access to the internet.**
15. Click **Add**. Figure 3-14 displays the completed configuration.


Add load balancing rule - Micro
https://portal.azure.com/#@az700examrefoutlook.onmicrosof...
Microsoft Azure
Search resources, services, and docs (G+/)
Home > Load balancing > lb-eus-app1 >

Add load balancing rule

lb-eus-app1

A load balancing rule distributes incoming traffic that is sent to a selected IP address and port combination across a group of backend pool instances. Only backend instances that the health probe considers healthy receive new traffic.

Name *	app1-https
IP Version *	[✓ IPv4] IPv6
Frontend IP address *	fe-app1 (20.115.107.149)
Protocol *	[✓ TCP] UDP
Port *	80
Backend port *	8080
Backend pool *	bep-vmss-app1
Health probe *	hp-app1 (TCP:80) Create new
Session persistence	None
Idle timeout (minutes) *	4
TCP reset	[✓ Disabled] Enabled
Floating IP	[✓ Disabled] Enabled
Outbound source network address translation (SNAT)	[✓ (Recommended) Use outbound rules to provide backend pool members access to the internet. Learn more] Use implicit outbound rule. This is not recommended because it can cause SNAT port exhaustion. Learn more

FIGURE 3-14 Add load balancing rule

The configuration displayed in Figure 3-14 creates a rule that brings together the individual components of the load balancer:

- Frontend IP address
- Backend pool
- Health probe

Along with these components, the protocol, external port number, and internal port number are defined. This will effectively distribute traffic incoming on public IP address 20.115.107.149 on TCP port 80 to the internal virtual machine scale set named bep-vmss-app1, internally on port 8080.

Azure load balancers also offer an option named *session persistence*, not to be confused with *session affinity*. Session persistence has three configuration options:

- None
- Client IP
- Client IP and protocol

The default option, None, uses a hash-based algorithm that allows successive requests from the same end user to be forwarded to any virtual machine in the backend pool. Client IP tracks the end user's IP address to ensure that successive requests from that IP address are forwarded to the same virtual machine in the backend pool, regardless of protocol. Client IP and protocol takes both the end user's IP address and the protocol that is being used, and if they are the same, then successive requests will be sent to the same virtual machine in the backend pool.

The idle timeout option defines the time period before a connection is considered dropped. The default timeout period is 4 minutes, but it can be configured up to 30 minutes for load balancer rules. By default, when a connection times out, there is no communication to the backend pool. If the application has its own timeout period, it may or may not release the connection. If you would prefer that the load balancer specifically sends a TCP reset packet when a timeout occurs, you can enable this setting.

Floating IP addresses allow you to reuse the same port number on the backend pool with multiple load balancer rules. Some applications or specific architectures might require this, including:

- High availability clustering
- Network virtual appliances
- Multiple TLS endpoints without re-encryption

For the outbound source network address translation (SNAT), the default option is the recommended setting of using outbound rules. This enables you to configure additional IP addresses to use as SNAT ports. This avoids exhausting the available number of ports, which can happen with the *Use implicit outbound rule* option. Outbound rules are further discussed later in this skill section.

Create and configure inbound NAT rules

An inbound NAT rule is similar to the load balancer rules, as it has a lot of the same configuration properties:

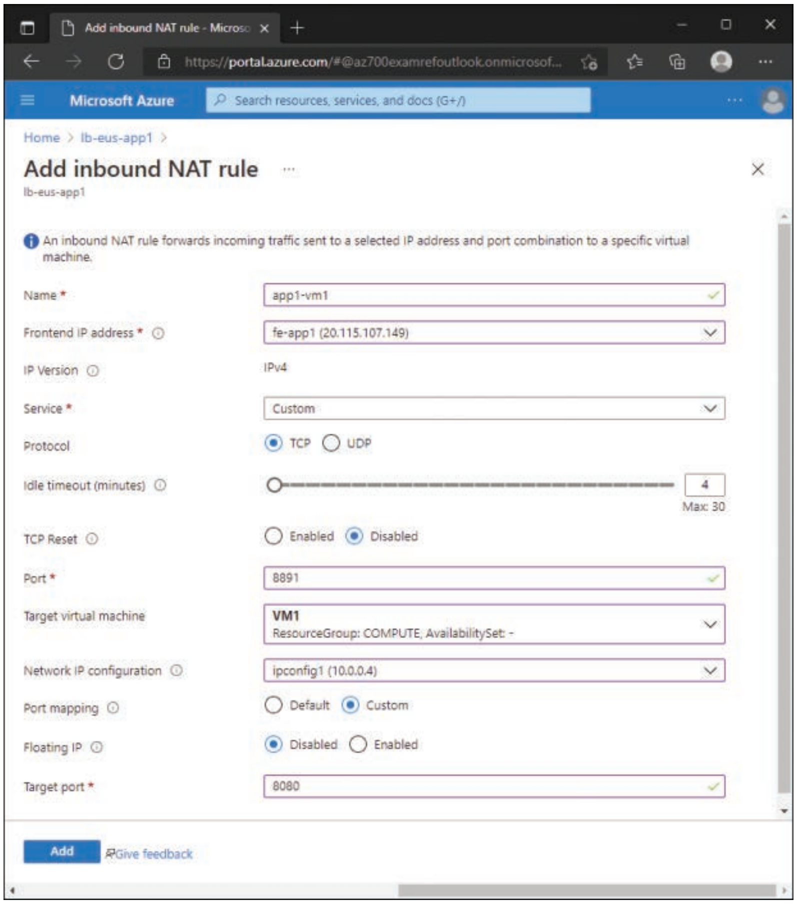
- Frontend IP address
- Protocol
- Port
- Idle timeout
- TCP reset

The difference is that instead of load balancing across multiple resources in a backend pool, an inbound NAT rule is to perform address translation for an individual virtual machine. This is useful if you have an administrative portal or custom port number for troubleshooting your application or service on a different port number. This gives you the ability to connect directly to a single VM in your backend pool through the load balancer.

To create an inbound NAT rule, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Load Balancers**.
3. On the Load Balancer page, select an existing load balancer, for example **lb-eus-app1**.
4. On the selected load balancer, click the **Inbound NAT rules** blade.
5. On the Inbound NAT rules blade, click **Add**.
6. In the Name field, specify a name such as **app1-vm1**.
7. In the Frontend IP address dropdown, select the desired IP address. For example, select **fe-app1**.
8. In the next four fields, leave the default settings:
   - Service: **Custom**
   - Protocol: **TCP**
   - Idle timeout: **4 minutes**
   - TCP Reset: **Enabled**
9. In the Port field, specify a port number to translate. For example, specify **8891**.
10. In the Target virtual machine dropdown menu, select a virtual machine that has been deployed. For example, select **VM1**. Note that the virtual machine cannot have a public IP address associated with it.
11. In the Network IP configuration dropdown menu, select the IP configuration of the NIC associated with the VM. For example, select **ipconfig1**.
12. In the Port mapping field, select **Custom**.

13. Leave the Floating IP option set to the default of **Disabled**.
14. In the Target port field, specify the internal port number on the virtual machine. For example, specify **8080**.
15. Click **Add**. Figure 3-15 displays the completed configuration.


Add inbound NAT rule - Microso
https://portal.azure.com/#@az700examrefoutlook.onmicrosof...
Microsoft Azure
Search resources, services, and docs (G+/)
Home > lb-eus-app1 >

Add inbound NAT rule
lb-eus-app1

An inbound NAT rule forwards incoming traffic sent to a selected IP address and port combination to a specific virtual machine.

Name *	app1-vm1
Frontend IP address *	fe-app1 (20.115.107.149)
IP Version	IPv4
Service *	Custom
Protocol	[✓ TCP] UDP
Idle timeout (minutes)	4 (Max: 30)
TCP Reset	Enabled [✓ Disabled]
Port *	8891
Target virtual machine	VM1 ResourceGroup: COMPUTE, AvailabilitySet: -
Network IP configuration	ipconfig1 (10.0.0.4)
Port mapping	Default [✓ Custom]
Floating IP	[✓ Disabled] Enabled
Target port *	8080

FIGURE 3-15 Add inbound NAT rule

As you can see in the configuration, instead of specifying a backend pool of multiple resources, you select a single target virtual machine. This is the primary difference between load balancing rules and inbound NAT rules. In the configuration displayed in Figure 3-15, any communication on port 8891 directed at the frontend IP address of 20.115.107.149 would be forwarded directly to VM1's private IP address of 10.0.0.4 but translated to port 8080.

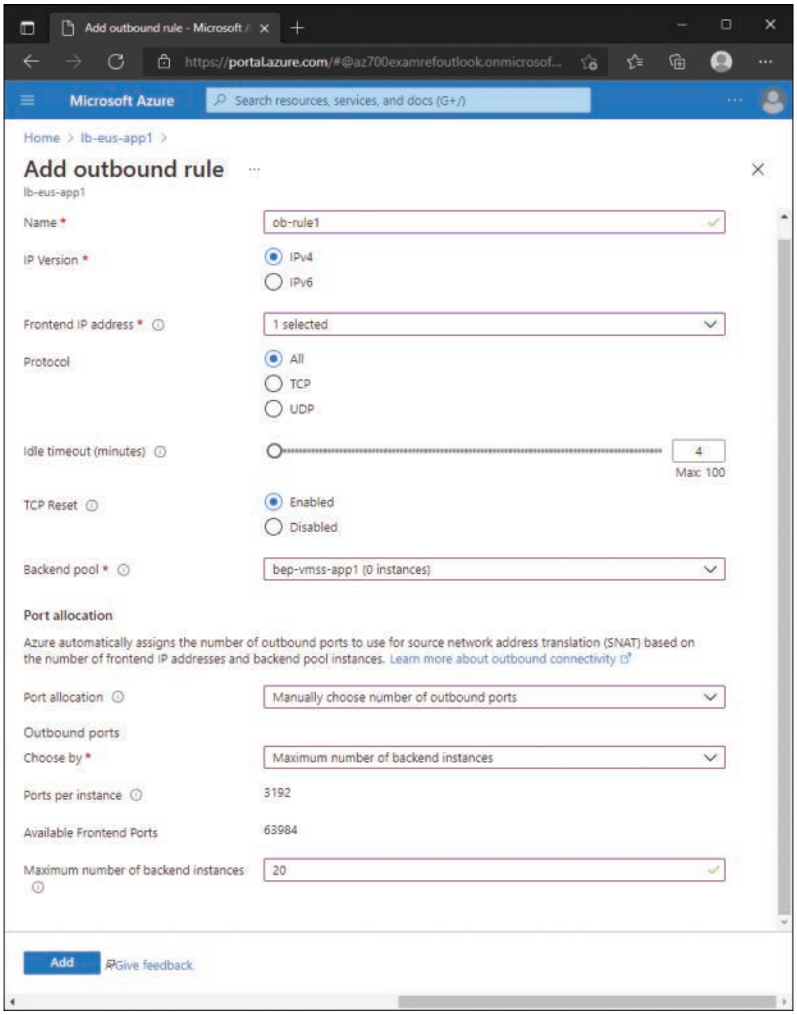
Create explicit outbound rules for a load balancer

Outbound rules allow you to configure network address translation (NAT) for all of the virtual machines that are defined in the backend pools. Outbound rules are available only with Standard SKU load balancers that have a public frontend IP address.

To configure an outbound rule, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Load Balancers**.
3. On the Load Balancer page, select an existing load balancer, for example **lb-eus-app1**.
4. On the selected load balancer, click the **Outbound rules** blade.
5. On the Outbound rules blade, click **Add**.
6. In the Name field, provide a name for the rule. For example, enter **ob-rule1**.
7. In the IP Version field, leave the default **IPv4** selected.
8. In the Frontend IP address dropdown, select the previously configured IP address. For example, select **fe-app1**.
9. In the Protocol field, leave the default of **All** selected.
10. In the Idle timeout field, leave the default setting of **4**.
11. In the TCP Reset field, leave the default setting of **Enabled**.
12. In the Backend pool dropdown, select the previously configured backend pool. For example, select **bep-vmss-app1**.
13. In the Port allocation dropdown menu, select **Manually choose number of outbound ports**.
14. In the Choose by field, select **Maximum number of backend instances**.
15. In the Maximum number of backend instances field, specify the maximum number of instances the VM scale set could increase to. For example, specify **20**.
16. Click **Add**. Figure 3-16 displays the completed configuration.

An outbound rule configures the source network address translation for the resources defined in the backend pool to communicate outbound through the load balancer. The available frontend ports are determined by the number of frontend IP addresses that you select. These available frontend ports are then distributed across the virtual machines in the backend pool. In the above configuration, the 63,984 possible frontend ports across a maximum of 20 virtual machines would result in approximately 3,192 ports per instance.


Add outbound rule - Microsoft A
https://portal.azure.com/#@az700examrefoutlook.onmicrosof...
Microsoft Azure
Search resources, services, and docs (G+/)
Home > lb-eus-app1 >

Add outbound rule

lb-eus-app1

Name *	ob-rule1
IP Version *	[✓ IPv4] IPv6
Frontend IP address *	1 selected
Protocol	[✓ All] TCP UDP
Idle timeout (minutes)	4 (Max: 100)
TCP Reset	[✓ Enabled] Disabled
Backend pool *	bep-vmss-app1 (0 instances)

Port allocation

Azure automatically assigns the number of outbound ports to use for source network address translation (SNAT) based on the number of frontend IP addresses and backend pool instances. Learn more about outbound connectivity

Port allocation	Manually choose number of outbound ports
Outbound ports Choose by *	Maximum number of backend instances
Ports per instance	3192
Available Frontend Ports	63984
Maximum number of backend instances	20

FIGURE 3-16 Add outbound rule

Skill 3.3: Design and implement Azure Application Gateway

Application gateways are the region-based load balancing option that focuses on layer 7 of the OSI model. By operating at layer 7, an application gateway provides many more features than a traditional load balancer. It can perform the basic functionality of distributing traffic across a backend pool of resources, but it also has the ability to redirect to certain backend pools based on the URL or headers of an incoming request. You can optionally enable a Web Application

Firewall (WAF), perform TLS termination, and enable end-to-end TLS encryption. This skill section introduces the application gateway and how to configure the various components.

This skill covers how to:

- Recommend Azure Application Gateway deployment options
- Choose between manual and autoscale
- Create a backend pool
- Configure HTTP settings
- Configure health probes
- Configure listeners
- Configure routing rules
- Configure Transport Layer Security (TLS)
- Configure rewrite policies

Recommend Azure Application Gateway deployment options

Azure application gateways operate at layer 7 of the OSI model, compared to layer 4 for traditional and Azure load balancers. By operating at layer 7, there are additional features that an application gateway has compared to a traditional load balancer:

- SSL termination
- Autoscaling
- Web Application Firewall (WAF)
- Ingress Controller for Azure Kubernetes Service (AKS)
- URL-based routing
- Multiple-site hosting
- Session affinity
- Connection draining
- Custom error pages
- HTTP header and URL rewrites

Application gateways are deployed into an Azure region and can be deployed to be zone-redundant. This is important when considering the deployment options that are available with Azure Front Door, which is discussed more in the next skill section. For applications that will be deployed into one or two regions and require SSL termination or any of the other features listed above, an application gateway is a good choice to recommend. If the application will reside in more than two regions or require some of the other features outlined with Azure Front Door, such as a content delivery network (CDN), then Azure Front Door might be a better option.

After you decide that an application gateway is the recommended ingress point for an application, there are a few SKUs to decide between. First, you can decide whether you want a v1 or a v2 SKU. The v1 SKUs have three preset configuration sizes:

- Small
- Medium
- Large

The v1 SKUs do not provide autoscaling and must be manually configured to one of the predefined sizes. The v2 SKUs include these features that v1 gateways do not:

- Autoscaling
- Zone redundancy
- Static virtual IPs
- AKS Ingress controller
- Azure Key Vault integration
- HTTPS header rewrites
- WAF custom rules

After you decide whether to deploy a v1 or a v2 SKU, both options have a separate SKU for enabling a WAF. For example, there is WAF v1, Medium size; or WAF v2 with autoscaling available.

Choose between manual and autoscale

The first factor in choosing between manual and autoscale is the SKU that you decide to deploy. If you choose to deploy a v1 SKU, then autoscaling is not available and you must choose from one of the predetermined sizes and set the instance count manually, up to 32 instances.

If you choose to deploy a v2 SKU, then you also have the choice of using manual instance counts or autoscaling. In either scale, you can scale up to 125 instances with a v2 SKU. When you choose to use autoscaling, you also set minimum and maximum guardrails. If you know for your application that you need at least two instances at any given time, you can set the minimum to two. If for budget and cost management, you don’t expect the peak traffic to go beyond the need for 10 instances, you can set the maximum to 10.

Whether you decide to use manual or autoscale, each scale instance represents approximately 10 *compute units*. Compute units define the amount of compute behind the application gateway when you decide to use a v2 SKU type instead of the Small, Medium, or Large preset sizes. Each compute unit can accept approximately 50 concurrent connections per second without the WAF enabled, or 10 concurrent connections per second when using the WAF.

Compute units are also a component of the overall *capacity unit*. A capacity unit defines the compute, throughput, and consistent connections that an instance supports. A v2 SKU can process approximately 2.2 Mbps of throughput with each capacity unit, and a total of 2,500 persistent connections. For additional compute, throughput, or connections, additional instances are required.

To create an application gateway, follow these steps:

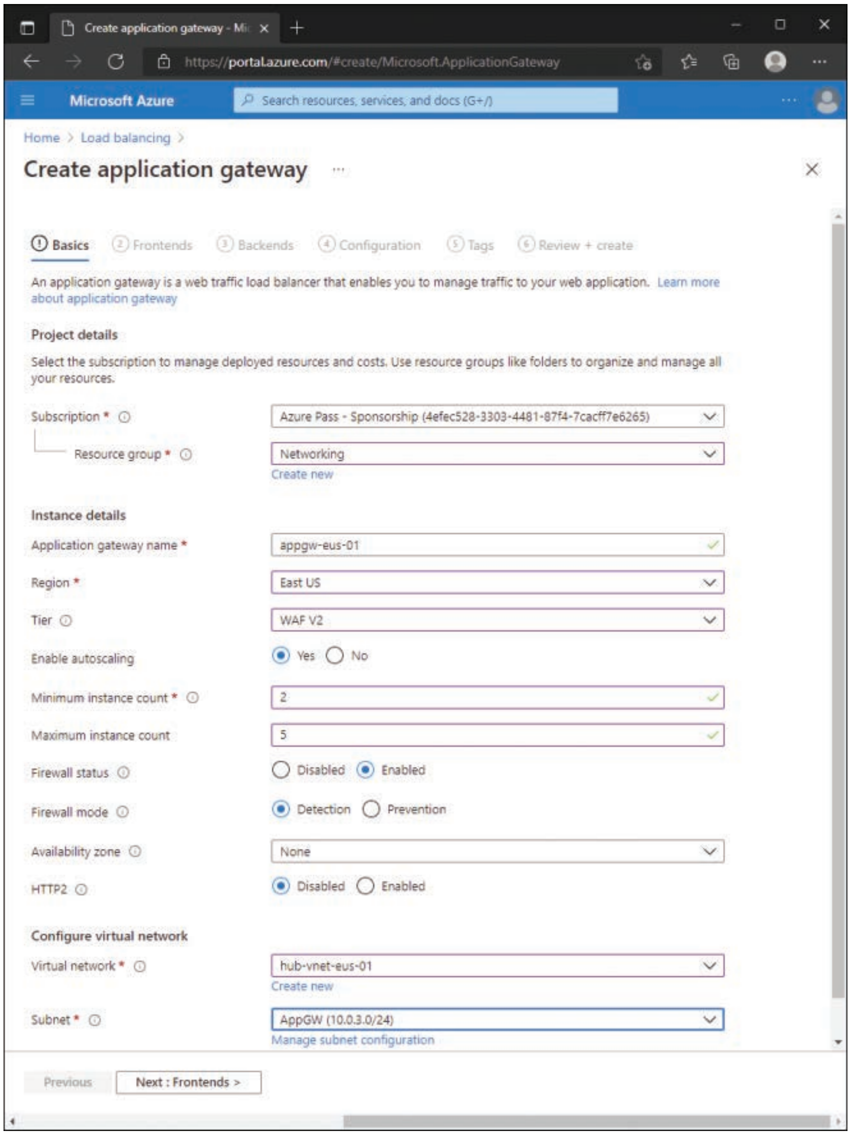
1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Application Gateways**.
3. On the Application Gateway page, click **Create**.
4. In the Subscription dropdown menu, select the subscription that has the resources you would like to put the application gateway in front of.
5. In the Resource group dropdown, select a resource group to put the application gateway in. For example, select **Networking**.
6. In the Application gateway name field, provide a name for the gateway. For example, enter **appgw-eus-01**.
7. In the Region dropdown menu, select the Azure region to deploy the gateway to. This must be the same as the resources you will add to the backend pool. For example, select **East US**.
8. In the Tier dropdown menu, select **WAF V2**.
9. In the Enable autoscaling field, set the value to **Yes**.
10. In the Minimum instance count field, set the value to **2**.
11. In the Maximum instance count field, set the value to **5**.
12. Leave the next fields at their default values:

    Firewall status: **Enabled**

    Firewall mode: **Detection**

    Availability zone: **None**

    HTTP2: **Disabled**
13. In the Virtual network dropdown menu, select the virtual network to associate the application gateway with. For example, select **hub-vnet-eus-01**.
14. In the Subnet field, select an available subnet that does not have resources connected. For example, select a dedicated subnet named **AppGW**. Figure 3-17 displays the completed configuration on the Basics tab.
15. Click **Next: Frontends**.
16. Leave the Frontend IP address type at the default, **Public**.
17. In the Public IP address field, click **Add new**.


Create application gateway - Mi

https://portal.azure.com/#create/Microsoft.ApplicationGateway

Microsoft Azure

Search resources, services, and docs (G+/)

Home > Load balancing >

Create application gateway

Basics | Frontends | Backends | Configuration | Tags | Review + create

An application gateway is a web traffic load balancer that enables you to manage traffic to your web application. Learn more about application gateway

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *	Azure Pass - Sponsorship (4efec528-3303-4481-87f4-7cacff7e6265)
Resource group *	Networking

Create new

Instance details

Application gateway name *	appgw-eus-01
Region *	East US
Tier	WAF V2
Enable autoscaling	[✓ Yes] No
Minimum instance count *	2
Maximum instance count	5
Firewall status	Disabled [✓ Enabled]
Firewall mode	[✓ Detection] Prevention
Availability zone	None
HTTP2	[✓ Disabled] Enabled

Configure virtual network

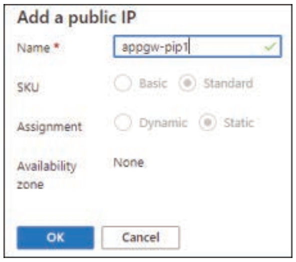
Virtual network *	hub-vnet-eus-01
Subnet *	AppGW (10.0.3.0/24)

Create new

Manage subnet configuration

FIGURE 3-17 Create application gateway basics

18. The Add a public IP window will pop up. In the Name field, name the public IP address object. For example, enter **appgw-pip1**. Because the IP address is being associated with an application gateway, the IP SKU must be Standard with a Static assignment. These values are not available to configure.
19. Click **OK**. Figure 3-18 displays the Add a public IP popup.


Add a public IP

Name * appgw-pip1

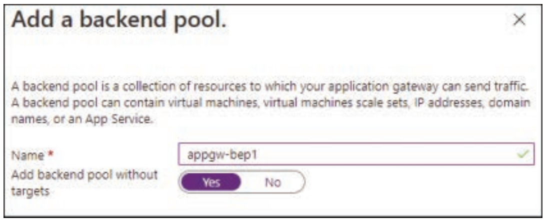
SKU ( ) Basic (●) Standard

Assignment ( ) Dynamic (●) Static

Availability zone None

FIGURE 3-18 Add a public IP

20. Click **Next: Backends**.
21. On the Backend pool tab, click **Add a backend pool**.
22. In the Add a backend pool window, provide a name for the backend pool. For example, enter **appgw-bep1**.
23. Set the Add backend pool without targets field to **Yes**. Backend pools are discussed more in the next section.
24. Click **Add**. Figure 3-19 displays the completed Add a backend pool page.


Add a backend pool.

A backend pool is a collection of resources to which your application gateway can send traffic. A backend pool can contain virtual machines, virtual machines scale sets, IP addresses, domain names, or an App Service.

Name * appgw-bep1

Add backend pool without targets [✓ Yes] No

FIGURE 3-19 Add a backend pool

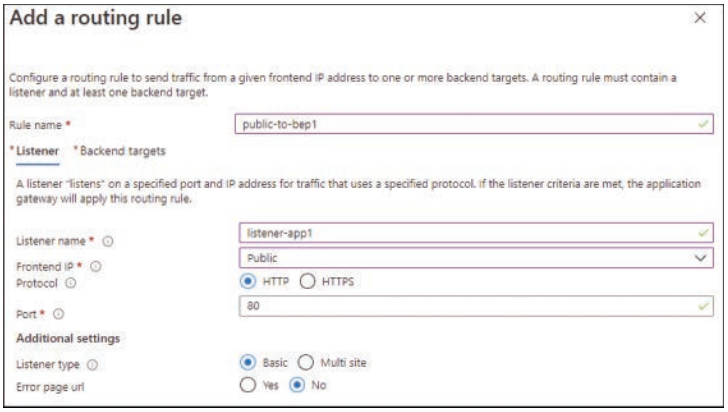
25. Click **Next: Configuration**.
26. On the Configuration tab, click **Add a routing rule**.
27. In the Rule name field, specify a name for the routing rule. For example, specify **public-to-bep1**.
28. In the Listener name field, enter a name for the listener. For example, enter **listener-app1**. Listeners are discussed in more detail later in this skill section.
29. In the Frontend IP dropdown, select **Public**.
30. Leave the remaining values set to their defaults:

    Protocol: **HTTP**

    Port: **80**

    Listener type: **Basic**

    Error page url: **No**
31. Figure 3-20 displays the completed Listener tab. Click the **Backend targets** tab.



Add a routing rule

Configure a routing rule to send traffic from a given frontend IP address to one or more backend targets. A routing rule must contain a listener and at least one backend target.

Rule name *: public-to-bep1

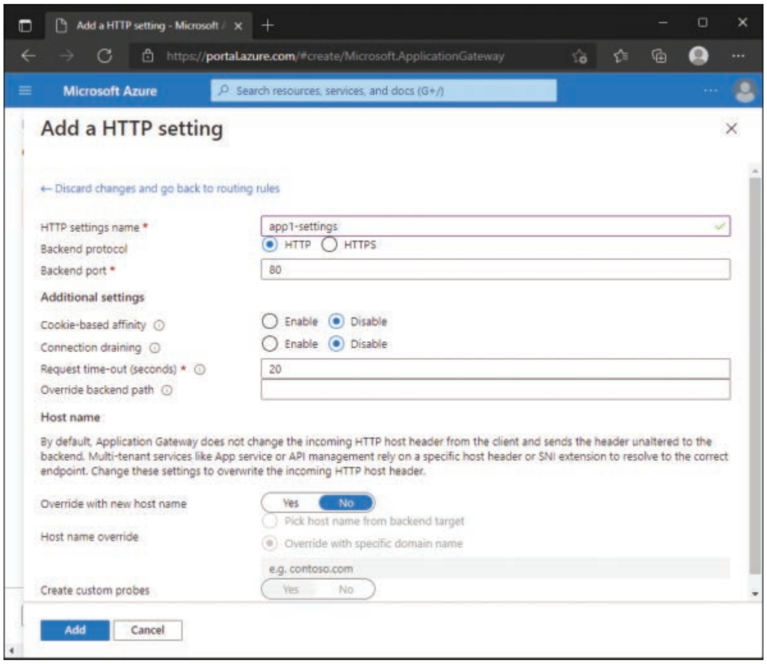
*Listener | *Backend targets

A listener "listens" on a specified port and IP address for traffic that uses a specified protocol. If the listener criteria are met, the application gateway will apply this routing rule.

Listener name *	listener-app1
Frontend IP *	Public
Protocol	[✓ HTTP] HTTPS
Port *	80
Additional settings	
Listener type	[✓ Basic] Multi site
Error page url	Yes [✓ No]

FIGURE 3-20 Add a routing rule – Listener tab

32. On the Backend targets tab, leave the default target type as **Backend pool**.
33. In the Backend target dropdown menu, select the previously configured backend pool. For example, select **appgw-bep1**.
34. In the HTTP settings field, click **Add new**.
35. In the Add a HTTP setting page, provide a name for the settings. For example, enter **app1-settings**. HTTP settings are discussed in more detail later in this skill section.
36. Leave all other fields set to default values and click **Add**. Figure 3-21 displays the configured HTTP settings.


Add a HTTP setting - Microsoft
https://portal.azure.com/#create/Microsoft.ApplicationGateway
Microsoft Azure
Search resources, services, and docs (G+/)

Add a HTTP setting

← Discard changes and go back to routing rules

HTTP settings name *	app1-settings
Backend protocol	[✓ HTTP] HTTPS
Backend port *	80
Additional settings	
Cookie-based affinity	Enable [✓ Disable]
Connection draining	Enable [✓ Disable]
Request time-out (seconds) *	20
Override backend path	

Host name

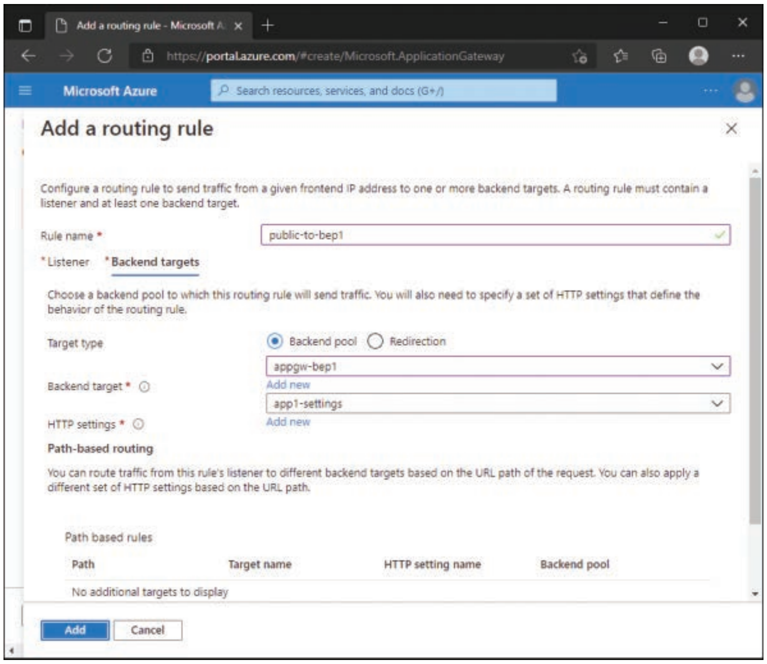
By default, Application Gateway does not change the incoming HTTP host header from the client and sends the header unaltered to the backend. Multi-tenant services like App service or API management rely on a specific host header or SNI extension to resolve to the correct endpoint. Change these settings to overwrite the incoming HTTP host header.

Override with new host name	Yes [✓ No]
Host name override	Pick host name from backend target / [✓ Override with specific domain name]
	e.g. contoso.com
Create custom probes	Yes No

Add | Cancel

FIGURE 3-21 Add a HTTP setting

37. On the Add a routing rule page, the HTTP settings field should display the new app1-settings. Click **Add**. Figure 3-22 displays the Backend targets tab.



Add a routing rule - Microsoft A
https://portal.azure.com/#create/Microsoft.ApplicationGateway
Microsoft Azure
Search resources, services, and docs (G+/)

Add a routing rule

Configure a routing rule to send traffic from a given frontend IP address to one or more backend targets. A routing rule must contain a listener and at least one backend target.

Rule name * public-to-bep1

*Listener *Backend targets

Choose a backend pool to which this routing rule will send traffic. You will also need to specify a set of HTTP settings that define the behavior of the routing rule.

Target type: (● Backend pool) ○ Redirection

Backend target *: appgw-bep1 (Add new)

HTTP settings *: app1-settings (Add new)

Path-based routing

You can route traffic from this rule's listener to different backend targets based on the URL path of the request. You can also apply a different set of HTTP settings based on the URL path.

Path based rules

Path	Target name	HTTP setting name	Backend pool
No additional targets to display			

FIGURE 3-22 Add a routing rule – Backend targets tab

38. Click **Next: Tags**.
39. Click **Next: Review + Create**.
40. Click **Create**.

These steps deploy a simple application gateway with the minimum requirements to get the first instance up and running. An application gateway supports multiple backend pools, health probes, listeners, routing rules, and more. All these are discussed in the next sections.

Create a backend pool

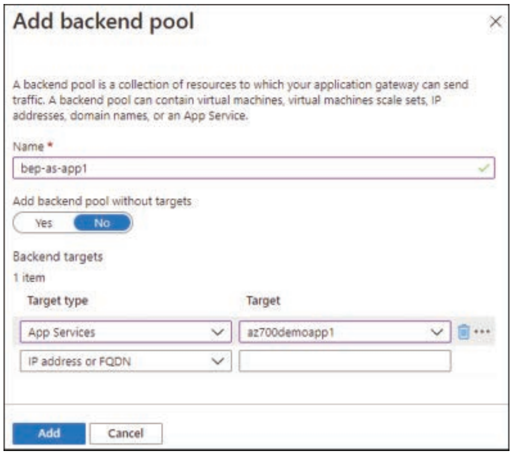
When you deploy an application gateway, there are a number of required components during deployment and more that you can configure after it is deployed, including backend pools. Backend pools are the resources “behind” the application gateway in a network topology or architecture design. Supported services for a backend pool in an application gateway include:

- IP addresses or fully qualified domain names
- Virtual machines
- Virtual machine scale sets
- App services

Essentially, as long as the application gateway can communicate with the resource, it can be added as a backend pool. This is true even if the resource is in another Azure region, on-premises, or in another cloud. After you have created a backend pool object that identifies these resources, you associate the pool with a listener by using a routing rule. Listeners and routing rules are discussed in a later section.

To create a new backend pool for an existing application gateway, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Application Gateways**.
3. On the Application Gateway page, select the application gateway that was previously created, for example **appgw-eus-01**.
4. On the Application Gateway page, click the **Backend pools** blade.
5. On the Backend pools blade, click **Add**. Note that one backend pool already exists from when you created the application gateway, but additional backend pools can be created.
6. On the Add backend pool screen, provide a name for the backend pool. For example, enter **bep-as-app1**.
7. In the Target type dropdown menu, select **App Services**.
8. In the Target dropdown menu, select an app service that you have deployed. For example, select **az700demoapp1**. Note that this requires an app service to be created in advance. App services are not covered on the AZ-700, and the steps to create one are not outlined here. If you would like to use a different target type, select and configure it here.
9. Click **Add**. Figure 3-23 shows the completed backend pool page.


Add backend pool

A backend pool is a collection of resources to which your application gateway can send traffic. A backend pool can contain virtual machines, virtual machines scale sets, IP addresses, domain names, or an App Service.

Name *
bep-as-app1

Add backend pool without targets
Yes [✓ No]

Backend targets
1 item

Target type	Target
App Services	az700demoapp1
IP address or FQDN	

FIGURE 3-23 Add backend pool

A backend pool is really just an object definition of the resource that the application gateway needs to direct traffic to. However, the backend pool is only the object definition. To be able to forward requests that the application gateway receives, then health probes, listeners, and routing rules must also be configured.

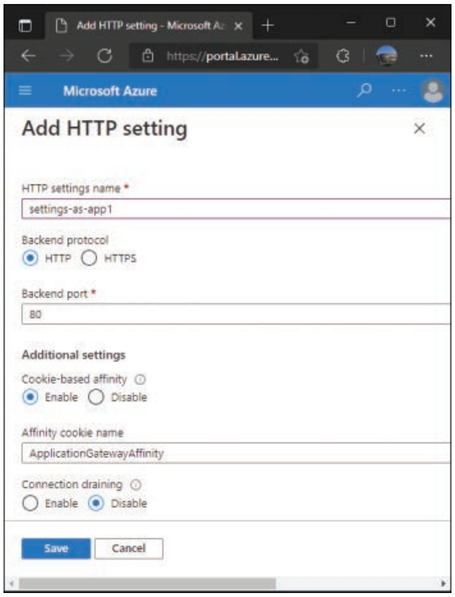
Configure HTTP settings

An application gateway relies on the HTTP settings that you configure to understand the specifics of forwarding traffic to the backend pool. HTTP settings are required to configure both health probes and routing rules.

HTTP settings are essentially a definition of the protocol and port number that should be used to communicate with a backend pool. The HTTP settings are also where you can configure cookie-based session affinity, as well as connection draining if you are taking a resource offline.

To add an HTTP setting to an application gateway, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Application Gateways**.
3. On the Application Gateway page, select the application gateway that was previously created, for example **appgw-eus-01**.
4. On the Application Gateway page, click the **HTTP settings** blade.
5. On the HTTP settings blade, click **Add**.
6. On the Add HTTP setting page, in the Name field provide a name for the collection of settings. For example, enter **settings-as-app1**.
7. In the Backend protocol and Port fields, leave the defaults of **HTTP** and **80**.
8. In the Cookie-based affinity field, select **Enable**, and then leave the cookie name the default value.
9. Leave the remaining settings at their default values and click **Save**. Figure 3-24 displays the completed configuration.



HTTP settings name *
settings-as-app1

Backend protocol
[✓ HTTP] HTTPS

Backend port *
80

Additional settings

Cookie-based affinity
[✓ Enable] Disable

Affinity cookie name
ApplicationGatewayAffinity

Connection draining
Enable [✓ Disable]

FIGURE 3-24 Add HTTP setting

Configure health probes

Application gateways automatically monitor the resources defined in the backend pool to ensure that they are healthy before sending connection requests and traffic to that resource. In the event that a resource becomes unavailable, the health probes are configured with a failure threshold to indicate to the application gateway that the resource is unavailable. If all resources become unavailable, then the application gateway will present HTTP 502 (Bad Gateway) errors to the client connections.

For example, if the resource does not respond to three consecutive requests at 30-second intervals, it would be deemed unhealthy. The default values of a health probe are to test the connection every 30 seconds, with a 30 second timeout, and to mark the resource as unhealthy at three consecutive requests.

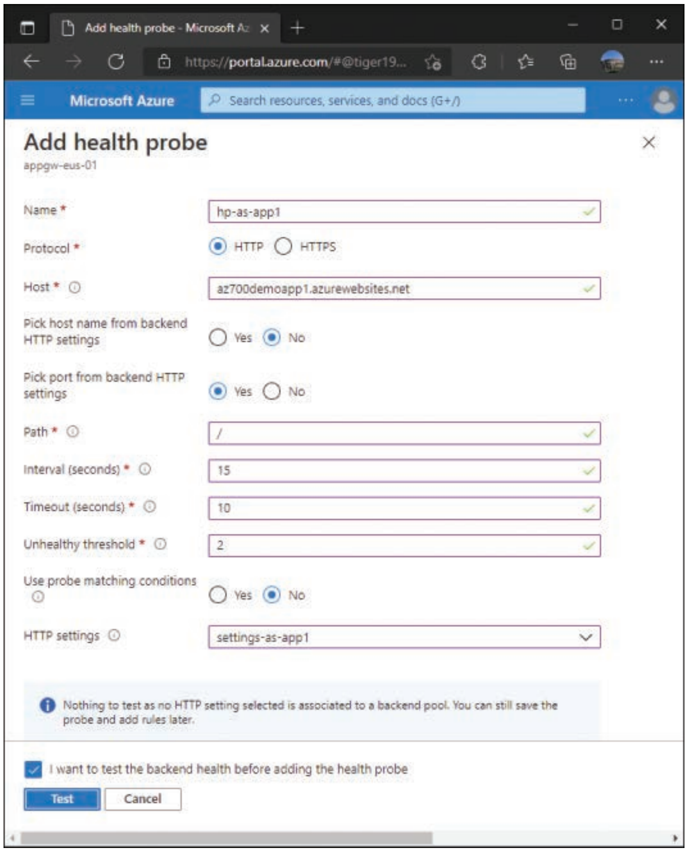
You can also configure custom health probes to be used with HTTP settings and rules so that you can customize the hostname, port number, interval, timeout period, or unhealthy threshold. Health probes require the following information to be configured:

- Name
- Backend host
- Virtual directory path
- Interval
- Timeout
- Unhealthy threshold

To configure a health probe on an existing application gateway, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Application Gateways**.
3. On the Application Gateway page, select the application gateway that was previously created, for example **appgw-eus-01**.
4. On the Application Gateway page, click the **Health probes** blade.
5. On the Health probes blade, click **Add**.
6. On the Add health probe page, in the Name field provide a name for the health probe. For example, enter **hp-as-app1**.
7. In the Protocol field, leave the default of **HTTP** selected.
8. In the Host field, enter the IP address or fully qualified domain name of the host to monitor. For example, enter **az700demoapp1.azurewebsites.net**.
9. Leave the Pick host name and Pick port from settings at their default values.
10. In the Path field, specify the path of the host to monitor. For example, to monitor the root of the path, enter **/**. You can specify the full path if there are certain virtual directories of your app to monitor.

11. In the Interval (seconds) field, set the desired interval to check with the resource. For example, set it to **15**.
12. In the Timeout (seconds) field, set the desired timeout value. For example, set it to **10**.
13. In the Unhealthy threshold field, set the desired consecutive failed attempts before marking the resource unhealthy. For example, set it to **2**. Overall, these settings would mark a resource unhealthy faster than would the default settings.
14. In the HTTP settings dropdown menu, select the settings that you previously created. For example, select **settings-as-app1**.
15. Click **Test**. Figure 3-25 shows the completed configuration. At this point, if you have followed the steps in the book, you will receive a message that the HTTP setting that has been selected is not associated with a backend pool. This is because the HTTP setting must also be configured with the routing rule, which then creates the backend pool association.



Add health probe - Microsoft Az
https://portal.azure.com/#@tiger19...
Microsoft Azure
Search resources, services, and docs (G+/)

Add health probe

appgw-eus-01

Name *	hp-as-app1
Protocol *	[✓ HTTP] HTTPS
Host *	az700demoapp1.azurewebsites.net
Pick host name from backend HTTP settings	Yes [✓ No]
Pick port from backend HTTP settings	[✓ Yes] No
Path *	/
Interval (seconds) *	15
Timeout (seconds) *	10
Unhealthy threshold *	2
Use probe matching conditions	Yes [✓ No]
HTTP settings	settings-as-app1

Nothing to test as no HTTP setting selected is associated to a backend pool. You can still save the probe and add rules later.

✓ I want to test the backend health before adding the health probe

FIGURE 3-25 Add health probe

16. Clear the checkbox next to I want to test the backend health before adding the health probe, and then click **Add**.

Configure listeners

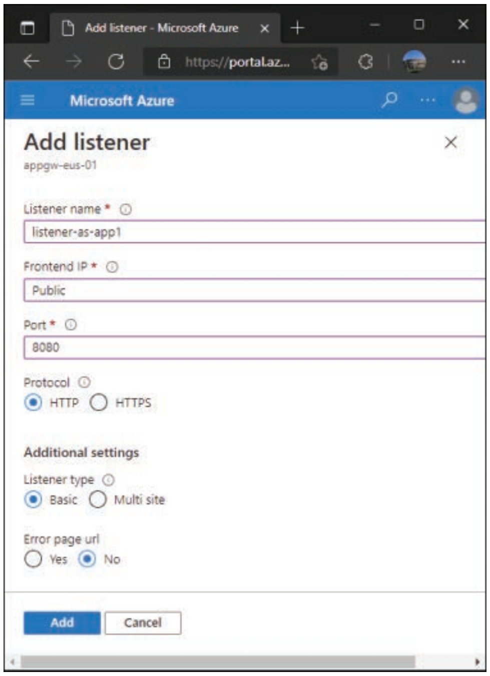
A listener is the component that "listens" to the frontend IP address of the application gateway for incoming connections. The listener defines the protocol, port number, hostname, and IP address that an incoming request is attempting to connect to and matches it with the configuration that you have defined. An application gateway can have multiple listeners for different hostnames, virtual directory paths, different backend pools, and many other scenarios.

As listeners identify the incoming traffic and connection requests, listeners support four protocols:

- HTTP
- HTTPS
- HTTP/2
- WebSocket

When you configure the listener, you can select either HTTP or HTTPS. WebSocket support is enabled by default and does not have a configurable setting. To create a listener on an existing application gateway, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Application Gateways**.
3. On the Application Gateway page, select the application gateway that was previously created, for example **appgw-eus-01**.
4. On the Application Gateway page, click the **Listeners** blade.
5. On the Health probes blade, click **Add listener**.
6. On the Add listener screen, provide a name for the listener. For example, enter **listener-as-app1**.
7. In the Frontend IP dropdown menu, select the frontend IP address that you configured when you deployed the application gateway. For example, select **Public**.
8. In the Port field, enter a port number that the listener should expect incoming traffic on. For example, enter **8080**.
9. In the remaining fields, leave the default values:
   - Protocol: **HTTP**
   - Listener type: **Basic**
   - Error page url: **No**
10. Click **Add**. Figure 3-26 displays the completed configuration.



Add listener
appgw-eus-01

Listener name * : listener-as-app1
Frontend IP * : Public
Port * : 8080
Protocol: [✓ HTTP] HTTPS

Additional settings
Listener type: [✓ Basic] Multi site
Error page url: Yes [✓ No]

FIGURE 3-26 Add listener

Configure routing rules

Routing rules are the glue that holds together the listener, backend pool, and HTTP settings that you have created. There are two types of routing rules:

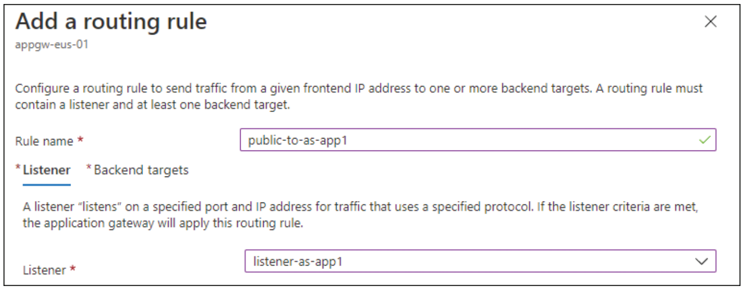
- Basic
- Multi-site

Basic routing rules map a single fully qualified domain name (FQDN) to one backend pool that has been configured. A multi-site listener allows you to define multiple FQDNs to a back-end pool. As of this writing, there is a preview feature to also allow wildcards to be used as part of domain names or as portions of sub-domain names.

To create a new routing rule, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Application Gateways**.
3. On the Application Gateway page, select the application gateway that was previously created, for example **appgw-eus-01**.
4. On the Application Gateway page, click the **Rules** blade.
5. On the Rules blade, click **Request routing rule**.
6. On the Add a routing rule page, in the Rule name field, provide a name for the rule. For example, enter **public-to-as-app1**.

7. In the Listener dropdown menu, select the listener that you previously created. For example, select **listener-as-app1**. Figure 3-27 shows the completed Listener tab.


Add a routing rule
appgw-eus-01

Configure a routing rule to send traffic from a given frontend IP address to one or more backend targets. A routing rule must contain a listener and at least one backend target.

Rule name * public-to-as-app1

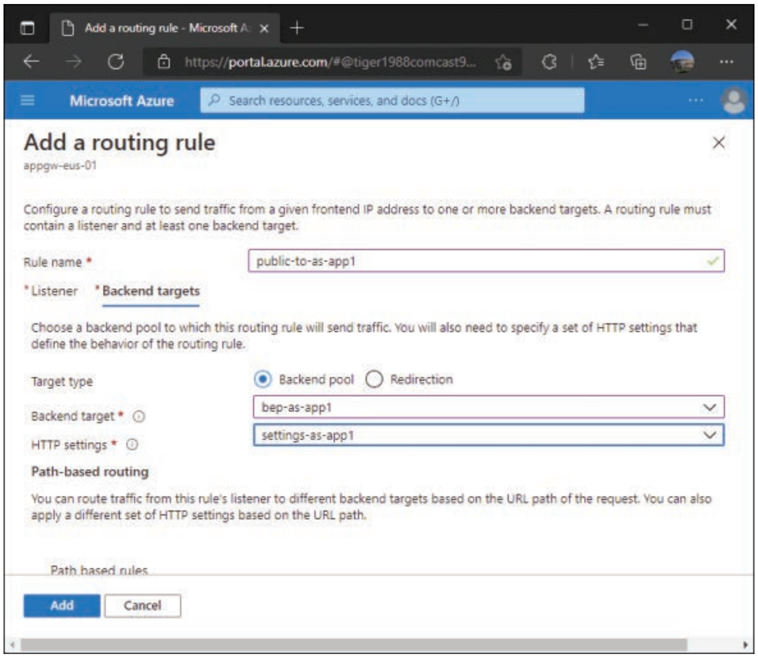
* Listener * Backend targets

A listener "listens" on a specified port and IP address for traffic that uses a specified protocol. If the listener criteria are met, the application gateway will apply this routing rule.

Listener * listener-as-app1

FIGURE 3-27 Add a routing rule – listener

8. Click the **Backend targets** tab.
9. In the Target type, leave the default value, **Backend pool**.
10. In the Backend target dropdown menu, select the desired backend target. For example, select **bep-as-app1**.
11. In the HTTP settings dropdown menu, select the HTTP settings that you previously created. For example, select **settings-as-app1**.
12. Click **Add**. Figure 3-28 shows the completed Backend targets tab.


Add a routing rule - Microsoft A
https://portal.azure.com/#@tiger1988comcast9...
Microsoft Azure
Search resources, services, and docs (G+/)
Add a routing rule
appgw-eus-01

Configure a routing rule to send traffic from a given frontend IP address to one or more backend targets. A routing rule must contain a listener and at least one backend target.

Rule name * public-to-as-app1

* Listener * Backend targets

Choose a backend pool to which this routing rule will send traffic. You will also need to specify a set of HTTP settings that define the behavior of the routing rule.

Target type [✓ Backend pool] Redirection

Backend target * bep-as-app1

HTTP settings * settings-as-app1

Path-based routing

You can route traffic from this rule's listener to different backend targets based on the URL path of the request. You can also apply a different set of HTTP settings based on the URL path.

Path based rules

Add Cancel

FIGURE 3-28 Add a routing rule – Backend targets

The above steps will create the rule that associates the frontend IP address of the application gateway, through the listener, with the backend pool and HTTP settings. Figure 3-29 outlines the various components of the application gateway that have been configured and how they are associated.

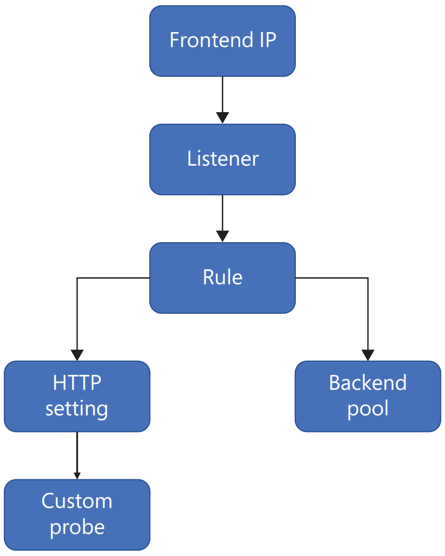


FIGURE 3-29 Application gateway diagram

Configure Transport Layer Security (TLS)

All the examples and steps that have been configured so far with the application gateway have used HTTP. Application gateways can also use TLS/SSL in two scenarios:

- TLS termination
- End-to-end TLS encryption

TLS termination is where the application gateway presents a certificate to the client through the listener. This enables the application gateway to decrypt the incoming traffic and provide an encrypted response back to the client. Therefore, when you configure the TLS certificate it must be a Personal Information Exchange (PFX) certificate file that has both public and private keys. For TLS termination, the certificate requires the full trust chain to be uploaded, including the root certificate from the CA, any intermediates, and the leaf certificate. Application gateway can use the following certificate types:

- Certificate Authority
- Extended Validation
- Wildcard
- Self-signed

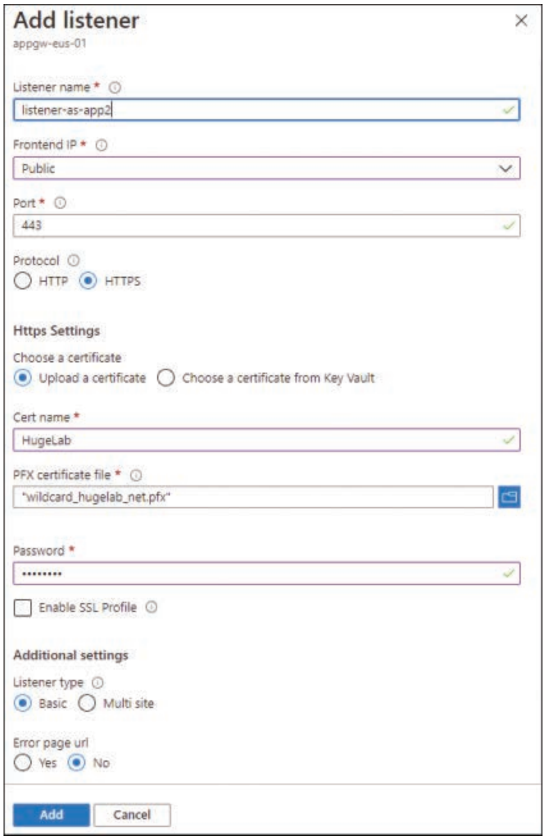
Allowing the application gateway to terminate the TLS connection typically provides better performance by the backend resources. This is especially true using larger key sizes for the certificates. By decrypting the traffic at the application gateway, it can view the request content and perform the URL or path-based routing to the various backend pools that you might define.

If you have a PFX certificate file to upload to an application gateway, follow these steps to add a new listener that uses HTTPS:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Application Gateways**.
3. On the Application Gateway page, select the application gateway that was previously created, for example **appgw-eus-01**.
4. On the Application Gateway page, click the **Listeners** blade.
5. On the Listeners blade, click **Add listener**.
6. On the Add listener screen, provide a name for the listener. For example, enter **listener-as-app2**.
7. In the Frontend IP dropdown menu, select the frontend IP address that you configured when you deployed the application gateway. For example, select **Public**.
8. In the Protocol field, select **HTTPS**. This will automatically set the port number to **443**.
9. Additional fields will appear for HTTPS settings. In the Choose a certificate field, leave the default value of **Upload a certificate**.
10. In the Cert name field, provide a name for the certificate. For example, enter **HugeLab**.
11. In the PFX certificate file field, browse and select your PFX certificate file.
12. In the Password field, provide the password to the PFX certificate file.
13. In the remaining fields, leave the default values:
    - Listener type: **Basic**
    - Error page url: **No**
14. Click **Add**. Figure 3-30 displays the completed configuration.

Some security requirements or compliance policies might require that all communication in the end-to-end process be encrypted. The process for this works similarly to TLS termination, except that the application gateway will perform an encrypted connection to the backend pool. The application gateway still decrypts the session when it arrives to identify the path and any other features that might be enabled, such as cookie-based session affinity, header rewrites, and more.

For an application gateway to be able to establish a new TLS session with a backend pool, the host settings must match the common name in the certificate that is being used. If the backend pool is using a self-signed certificate, then the certificate must be provided to the application gateway. The certificate must be defined in the HTTP settings that the routing rule is configured to use.



Add listener

appgw-eus-01

Listener name *: listener-as-app2

Frontend IP *: Public

Port *: 443

Protocol: HTTP [✓ HTTPS]

Https Settings

Choose a certificate: [✓ Upload a certificate] Choose a certificate from Key Vault

Cert name *: HugeLab

PFX certificate file *: "wildcard_hugelab_net.pfx"

Password *: ••••••••

☐ Enable SSL Profile

Additional settings

Listener type: [✓ Basic] Multi site

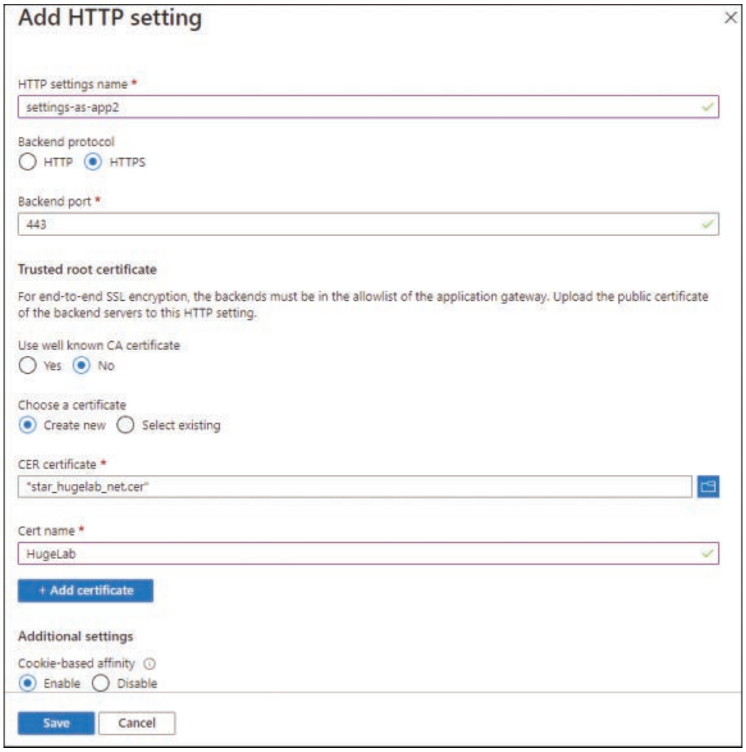
Error page url: Yes [✓ No]

FIGURE 3-30 Add listener

To configure settings that use HTTPS, follow these steps.

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Application Gateways**.
3. On the Application Gateway page, select the application gateway that was previously created, for example **appgw-eus-01**.
4. On the Application Gateway page, click the **HTTP settings** blade.
5. On the HTTP settings blade, click **Add**.
6. On the Add HTTP setting page, in the Name field provide a name for the collection of settings. For example, enter **settings-as-app2**.
7. In the Backend protocol and Port fields, select **HTTPS**. The backend port will automatically adjust to **443**.
8. In the CER certificate field, browse to and select your certificate file to communicate with the backend pool.
9. In the Cert Name field, provide a name for the certificate. For example, enter **HugeLab**.

10. In the Cookie-based affinity field, select **Enable**, and then leave the cookie name the default value.
11. Click **+Add certificate**.
12. Leave the remaining settings at their default values and click **Save**. Figure 3-31 displays the completed configuration.


Add HTTP setting

HTTP settings name *
settings-as-app2

Backend protocol
HTTP [✓ HTTPS]

Backend port *
443

Trusted root certificate

For end-to-end SSL encryption, the backends must be in the allowlist of the application gateway. Upload the public certificate of the backend servers to this HTTP setting.

Use well known CA certificate
Yes [✓ No]

Choose a certificate
[✓ Create new] Select existing

CER certificate *
"star_hugelab_net.cer"

Cert name *
HugeLab

Additional settings

Cookie-based affinity
[✓ Enable] Disable

FIGURE 3-31 Add HTTP setting

Configure rewrite policies

With an application gateway V2 SKU, you can create header rewrite sets to add, update, or remove various HTTP headers and server variables. When you create a rewrite set, you base the rewrites on rules, conditions, and actions based on actions, conditions, or variables. All headers in connection requests and responses can be modified, except for the *Connection* and *Upgrade* headers.

Each rule that you create has a rule sequence number that determines the order in which the rules are processed. Rules with the lower sequence number are processed first. If you configure two rules with the same sequence number, there is no predefined method of guaranteeing which rule is processed first.

Conditions in a rewrite set are simply if–then statements that identify the type of variable to check—either HTTP header or server variable—and then the value within that might need to be added or modified. Conditions can optionally be configured to be case-sensitive and to be looking for specific patterns to match in the header. Conditions are not required to be configured as part of a rewrite set if you want all traffic to be inspected or modified by the rule.

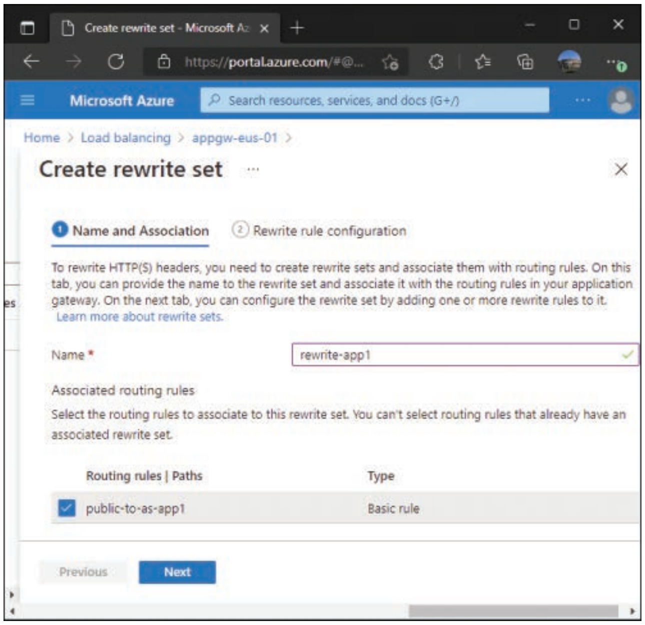
After identifying the component that needs to change, whether it is all packets or only the packets that meet the condition, the action defines what to change within the headers. The rewrite set can then set or delete a request header or response header, or modify the URL. These rewrite sets are associated with a routing rule when they are created.

Common scenarios for using header rewrites include:

- Removing port information from the X-Forwarded-For header
- Modifying a redirection URL
- Implementing security HTTP headers
- Deleting unwanted headers
- Parameter-based path selection

To create a rewrite set that sets a security HTTP header, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Application Gateways**.
3. On the Application Gateway page, select the application gateway that was previously created, for example **appgw-eus-01**.
4. On the Application Gateway page, click the **Rewrites** blade.
5. On the Rewrites blade, click **+ Rewrite set**.
6. On the Create rewrite set page, in the Name field provide a name for the set. For example, enter **rewrite-app1**.
7. In the Routing Rules | Paths section, select a previously created routing rule to associate the rewrite set with. For example, select **public-to-as-app1**.
8. Click **Next**. Figure 3-32 shows the completed Name and Association tab.
9. On the Rewrite rule configuration tab, click **Add rewrite rule**.
10. In the rewrite rule name field, provide a name for the new rule. For example, enter **SetTransportSecurity**. Leave the Rule sequence value at the default, **100**. Figure 3-33 displays the relevant portion of the Create rewrite set page.
11. In the Do section, select **Click to fix configuration this action**.
12. The Do section will expand. In the Rewrite type dropdown field, select **Response Header**.
13. In the Action type dropdown field, leave the default value of **Set**.


Create rewrite set - Microsoft Az
https://portal.azure.com/#@...
Microsoft Azure
Search resources, services, and docs (G+/)
Home > Load balancing > appgw-eus-01 >
Create rewrite set
Name and Association
Rewrite rule configuration

To rewrite HTTP(S) headers, you need to create rewrite sets and associate them with routing rules. On this tab, you can provide the name to the rewrite set and associate it with the routing rules in your application gateway. On the next tab, you can configure the rewrite set by adding one or more rewrite rules to it.
Learn more about rewrite sets.

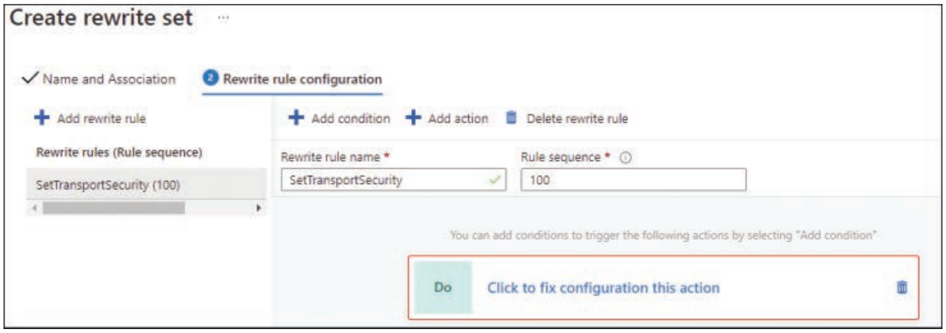
Name * rewrite-app1

Associated routing rules

Select the routing rules to associate to this rewrite set. You can't select routing rules that already have an associated rewrite set.

	Routing rules | Paths	Type
✓	public-to-as-app1	Basic rule

FIGURE 3-32 Create rewrite set



Rewrite rules (Rule sequence)
SetTransportSecurity (100)

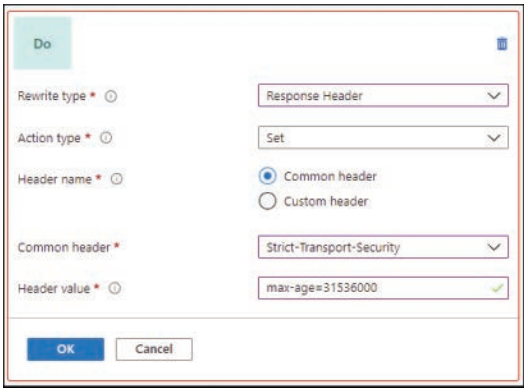
Rewrite rule name * SetTransportSecurity
Rule sequence * 100

You can add conditions to trigger the following actions by selecting "Add condition"

Do — Click to fix configuration this action

FIGURE 3-33 Rewrite rule configuration

14. In the Header name field, leave the default value of **Common header** selected.
15. In the Common header dropdown field, select **Strict-Transport-Security**.
16. In the Header value field, provide a value for the header. For example, enter **max-age=31536000**.
17. Click **OK**. Figure 3-34 displays the completed Do section.
18. Click **Create**.



Rewrite type *	Response Header
Action type *	Set
Header name *	[✓ Common header] Custom header
Common header *	Strict-Transport-Security
Header value *	max-age=31536000

FIGURE 3-34 Rewrite rule action

Skill 3.4: Implement Azure Front Door

Azure Front Door provides a combination of Azure services in a single solution, including layer 7 load balancing, web app firewall, security reporting, and content delivery and optimization. Unlike most other Azure services, Front Door is considered a global service that you do not deploy into a specific Azure region. Instead, when you create a Front Door endpoint, it is accessible from all of Azure's regions, edge locations, and points of presence (POPs) across the world.

As of this writing, *Azure Front Door* is the generally available product. A new deployment option named *Azure Front Door Standard/Premium* is currently in public preview and separates some of the features and functionality of the core product into the two versions. The Objectives list includes choosing an appropriate SKU, so although it is still in public preview, this skill section focuses on the preview versions.

This skill covers how to:

- Choose an Azure Front Door SKU
- Configure health probes
- Configure SSL termination and end-to-end SSL encryption
- Configure multisite listeners and configure back-end targets
- Configure routing rules

Choose an Azure Front Door SKU

There are two SKUs of Azure Front Door: Standard and Premium. The Standard SKU offers optimized content delivery across the world, not only in a single Azure region. Both SKUs of Azure Front Door are considered a global load balancer, which uses anycast IP addresses to locate the nearest Microsoft point of presence (POP) to access the Azure network backbone. Table 3-4 outlines the feature difference between the two SKU options.

TABLE 3-4 Azure Front Door SKU comparison

Feature	Standard	Premium
Custom domain	Yes	Yes
SSL Offloading	Yes	Yes
Caching	Yes	Yes
Compression	Yes	Yes
Global load balancing	Yes	Yes
Layer 7 routing	Yes	Yes
URL rewrites	Yes	Yes
Rules engine	Yes	Yes
Private Link	No	Yes
WAF	Custom rules only	Yes
Bot protection	No	Yes
Enhanced metrics and monitoring	Yes	Yes
Traffic report	Yes	Yes
Security report	No	Yes

As Table 3-4 outlines, the Premium SKU has all of the features of the Standard SKU, plus adds more rule options for the WAF, bot protection, integration with Microsoft Threat Intelligence and security analytics with reporting, and integration with Azure Private Link. To deploy an Azure front door with your choice of SKU, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Front Doors Standard/Premium**.
3. On the Front Doors Standard/Premium page, click **Create**.
4. In the Compare offerings screen, accept the default to deploy a Quick create Azure front door Standard/Premium, and click **Continue to create a front door**.
5. In the Subscription dropdown menu, select the desired subscription to deploy the front door in.

6. In the Resource Group dropdown, select the resource group. For example, select **Networking**.
7. In the Name field, provide a name for the front door. For example, enter **fd-app1**.
8. In the Tier field, select the desired SKU. For example, select **Standard**.
9. In the Endpoint name field, provide a globally unique name for the front door URL. For example, enter **az700fdapp**.
10. In the Origin type dropdown menu, select the origin component. For example, select **App Services**.
11. In the Origin host name dropdown menu, select the app service that you previously created. For example, select **az700demoapp1.azurewebsites.net**.
12. Leave the remaining fields at their default blank values, and click **Review + create**. Figure 3-35 displays the completed configuration.

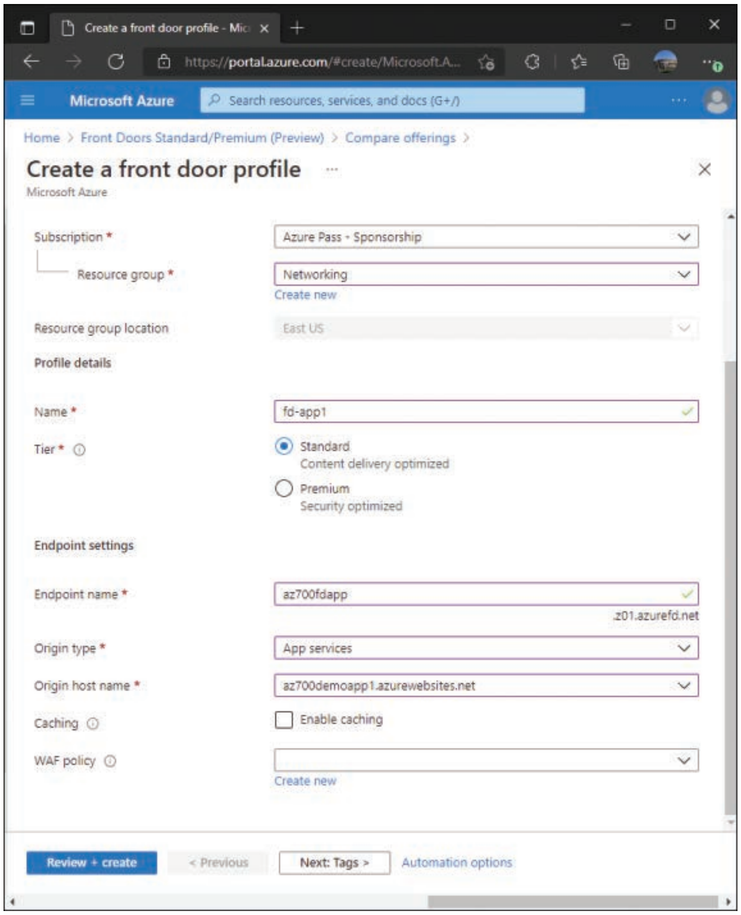

Create a front door profile - Mic
https://portal.azure.com/#create/Microsoft.A...
Microsoft Azure
Search resources, services, and docs (G+/)
Home > Front Doors Standard/Premium (Preview) > Compare offerings >
Create a front door profile
Microsoft Azure
Create new
Create new
Review + create
< Previous
Next: Tags >
Automation options
Subscription * Azure Pass - Sponsorship
Resource group * Networking
Resource group location East US
Profile details
Name * fd-app1
Tier * [✓ Standard] Content delivery optimized; Premium Security optimized
Endpoint settings
Endpoint name * az700fdapp .z01.azurefd.net
Origin type * App services
Origin host name * az700demoapp1.azurewebsites.net
Caching ☐ Enable caching
WAF policy

FIGURE 3-35 Create a front door profile

13. Click **Create**.

Configure health probes

Health probes are associated with the origin group that contains the resources (origins) that Front Door will send the client connections to. When you create the front door profile, it will automatically enable the health probe for the selected origin.

With Front Door, health probes perform two primary functions. First is to verify that the backend resource is online and healthy. Second, the health probe assists Front Door in determining the best backend resource to send the client requests to. Front Door also relies on POPs, and there can be many health probes that cause additional network traffic. The default health probe frequency is 30 seconds, which can result in approximately 200 health probe requests per minute, per backend source.

NEED MORE REVIEW? **AZURE POINTS OF PRESENCE**

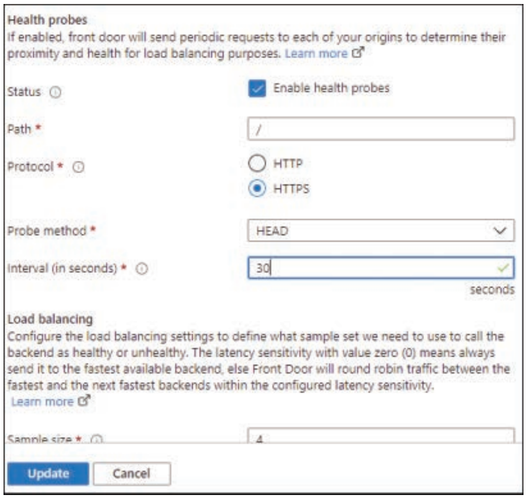
For more information on Azure regions, edge locations, and points of presence, visit *https://infrastructuremap.microsoft.com/*.

Health probes in Front Door have two possible probe methods: GET and HEAD. Using a GET probe retrieves information from the backend resource, including a message body. This can incur more costs and throughput with each health check. A HEAD request is the default value, and requires that the backend resource not include a message body in the response.

When a backend resource responds to a probe, both GET and HEAD requests should result in an HTTP 200 (OK) status code. Any other response type will count as a failed attempt. The response time in latency is also measured to assist the Front Door service in choosing the most responsive backend resource.

To modify the settings of a health probe associated with an origin group, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Front Doors Standard/Premium**.
3. On the Front Doors Standard/Premium page, select the Front Door instance that you previously created. For example, select **fd-app1**.
4. Click the **Origin groups** blade.
5. On the Origin groups blade, click the **default-origin-group**.
6. In the Update origin group screen, scroll down to the **Health probes** section.
7. Verify that the **Enable health probes** checkbox is already selected and that the path is set to **/**.
8. In the Protocol field, select **HTTPS**.
9. In the Interval field, set the time to **30** seconds.
10. Click **Update**. Figure 3-36 displays the updated Health probes section.


Health probes
If enabled, front door will send periodic requests to each of your origins to determine their proximity and health for load balancing purposes. Learn more

Status: [✓] Enable health probes
Path *: /
Protocol *: HTTP [✓ HTTPS]
Probe method *: HEAD
Interval (in seconds) *: 30 seconds

Load balancing
Configure the load balancing settings to define what sample set we need to use to call the backend as healthy or unhealthy. The latency sensitivity with value zero (0) means always send it to the fastest available backend, else Front Door will round robin traffic between the fastest and the next fastest backends within the configured latency sensitivity.
Learn more

Sample size *: 4

FIGURE 3-36 Origin group health probes

Configure SSL termination and end-to-end SSL encryption

By default and without any additional configuration, Azure Front Door supports using HTTPS on the default hostname that you configure when you first deploy the service. This is the *azurefd.net* domain that you customize when you create the service. There is no additional configuration required if you are using this domain to access the service.

If you plan on using a custom domain with your own certificate, then there is additional configuration required. As of this writing, Front Door only supports using a certificate that is in an Azure Key Vault in the same subscription. Azure Key Vault is a requirement, and it is not currently supported to be in a different subscription than the Front Door service. Additionally, the certificate must have a complete certificate chain, and the root CA listed must be on the Microsoft Trusted CA list. Finally, the certificate that you use cannot use an elliptic-curve (EC) cryptography algorithm.

The overall steps for allowing the Front Door service to access a Key Vault certificate are:

1. Register a service principal in Azure Active Directory associated with Azure Front Door.
2. Grant the Get permission to the service principal for Secrets and Certificates in Azure Key Vault.
3. Select the certificate from Front Door and associate it with a custom domain.

Register a service principal

The application ID for Azure Front Door is *ad0e1c7e-6d38-4ba4-9efd-0bc77ba9f037*. You can run this PowerShell command from Azure Cloud Shell to create a service principal that is associated with the Front Door application ID. The user identity that runs this command must be a Global Administrator in the Azure Active Directory environment.

```
New-AzADServicePrincipal -ApplicationId "205478c0-bd83-4e1b-a9d6-db63a3e1e1c8"
```

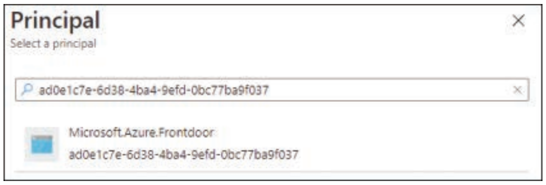
If you run the command from Azure Cloud Shell, you should receive information back about the newly created service principal. Note that the DisplayName is *Microsoft.Azure.FrontDoor-Cdn*.

```
ServicePrincipalNames : {205478c0-bd83-4e1b-a9d6-db63a3e1e1c8,
     https://microsoft.onmicrosoft.com/033ce1c9-f832-4658-b024-ef1cbea108b8}
ApplicationId         : 205478c0-bd83-4e1b-a9d6-db63a3e1e1c8
ObjectType            : ServicePrincipal
DisplayName           : Microsoft.AzureFrontDoor-Cdn
Id                    : a1d4a221-de6f-4b32-aa0d-d9873deeebd4
Type                  : ServicePrincipal
```

Azure Key Vault permissions

After you create the service principal, the object needs permission in Azure Key Vault to access the certificate that you upload. To use a Key Vault access policy to assign the permissions, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Key Vault**.
3. Select an existing key vault that you have deployed.
4. On the Key vault page, click the **Access policies** blade.
5. On the Access policies page, click **Add Access Policy**.
6. In the Key permissions dropdown menu, select **Get**.
7. In the Secret permission dropdown menu, select **Get**.
8. In the Select principal field, click **None selected**.
9. The Principal screen will appear. Search for the application ID, *205478c0-bd83-4e1b-a9d6-db63a3e1e1c8*, and select **Microsoft.Azure.FrontDoor-Cdn**.
10. Click **Select**. Figure 3-37 displays the search portion of the Principal screen.

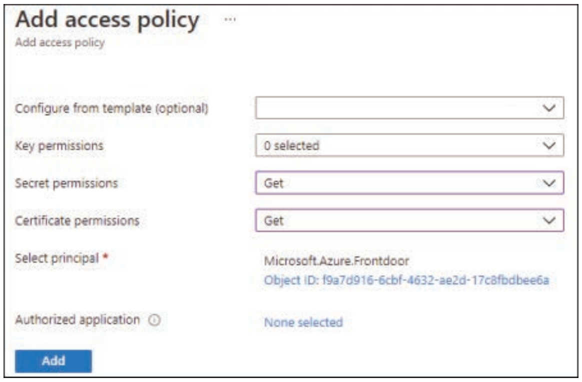


ad0e1c7e-6d38-4ba4-9efd-0bc77ba9f037

Microsoft.Azure.Frontdoor
ad0e1c7e-6d38-4ba4-9efd-0bc77ba9f037

FIGURE 3-37 Select a principal

11. On the Add access policy screen, click **Add**. Figure 3-38 displays the completed configuration.
12. On the Access policies page, click **Save**.


Add access policy ...
Add access policy

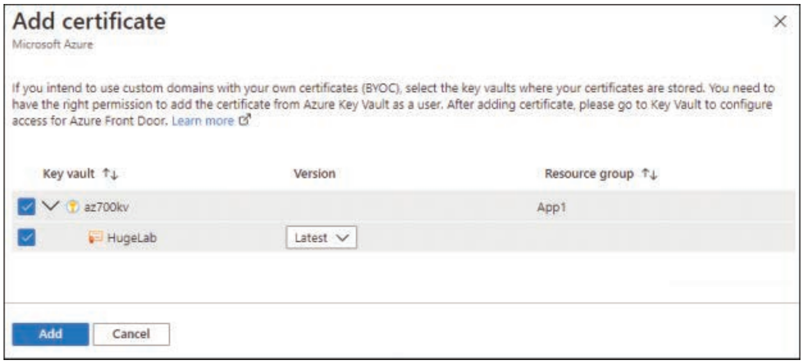
Configure from template (optional)	
Key permissions	0 selected
Secret permissions	Get
Certificate permissions	Get
Select principal *	Microsoft.Azure.Frontdoor Object ID: f9a7d916-6cbf-4632-ae2d-17c8fbdbee6a
Authorized application	None selected

FIGURE 3-38 Add access policy

Add custom domain to endpoint

After you have the certificate uploaded to Azure Key Vault and you have the Front Door service principal with the appropriate access policy, you can configure the custom domain on your Front Door endpoint. To add the custom domain to the endpoint with your own certificate, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Front Doors Standard/Premium**.
3. On the Front Doors Standard/Premium page, select the front door instance that you previously created. For example, select **fd-app1**.
4. Click the **Secrets** blade.
5. On the Secrets blade, click **Add certificate**.
6. On the Add certificate screen, expand the key vault with your certificate, then select the checkbox next to the certificate name.
7. Click **Add**. Figure 3-39 displays the selected certificate.

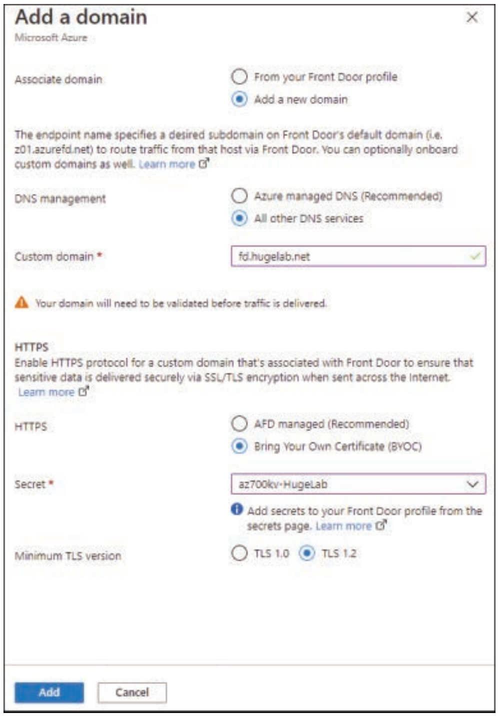

Add certificate
Microsoft Azure

If you intend to use custom domains with your own certificates (BYOC), select the key vaults where your certificates are stored. You need to have the right permission to add the certificate from Azure Key Vault as a user. After adding certificate, please go to Key Vault to configure access for Azure Front Door. Learn more

Key vault	Version	Resource group
✓ az700kv		App1
✓ HugeLab	Latest	

FIGURE 3-39 Add certificate

8. Click the **Endpoint manager** blade.
9. On the Endpoint manager screen, click **Edit endpoint** for the endpoint you want to configure the custom domain on.
10. In the Domains section, click **Add**.
11. In the Add a domain field, select **Add a new domain**.
12. In the DNS management field, select the appropriate DNS option for your domain. For example, select **All other DNS services**.
13. In the Custom domain field, enter the FQDN of your domain. For example, enter **fd.hugelab.net**.
14. In the HTTPS field, select **Bring Your Own Certificate (BYOC)**.
15. In the Secret dropdown menu, select the certificate that you added to Front Door from Key Vault. Figure 3-40 displays the completed Add a domain screen.



Add a domain

Microsoft Azure

Associate domain	From your Front Door profile [✓ Add a new domain]

The endpoint name specifies a desired subdomain on Front Door's default domain (i.e. z01.azurefd.net) to route traffic from that host via Front Door. You can optionally onboard custom domains as well. Learn more

DNS management	Azure managed DNS (Recommended) [✓ All other DNS services]
Custom domain *	fd.hugelab.net

Your domain will need to be validated before traffic is delivered.

HTTPS

Enable HTTPS protocol for a custom domain that's associated with Front Door to ensure that sensitive data is delivered securely via SSL/TLS encryption when sent across the Internet. Learn more

HTTPS	AFD managed (Recommended) [✓ Bring Your Own Certificate (BYOC)]
Secret *	az700kv-HugeLab

Add secrets to your Front Door profile from the secrets page. Learn more

Minimum TLS version	TLS 1.0 [✓ TLS 1.2]

FIGURE 3-40 Add a domain with a certificate

16. Click **Add**.

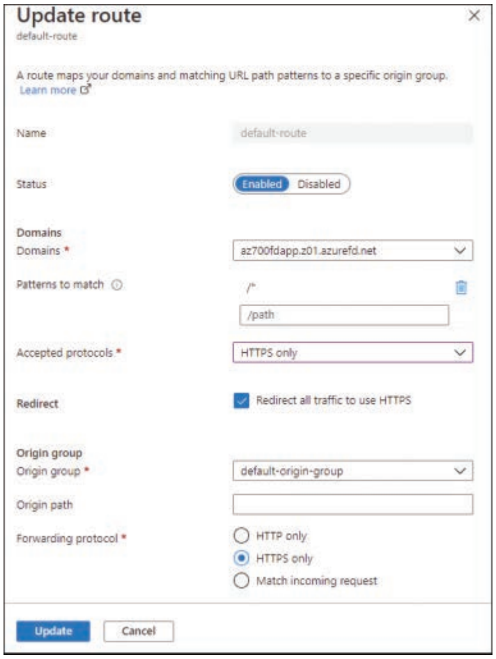
Completing these steps will add the certificate for the domain that you specified to the Front Door configuration. A CNAME record is also required at the DNS provider to forward requests for the custom domain that you added—for example, **fd.hugelab.net**. This would need to be redirected to the **azurefd.net** endpoint. This provides TLS termination at the Front Door endpoint.

End-to-end encryption

After configuring the HTTPS termination at the endpoint, the next step for your deployment might be to ensure end-to-end encryption. In an Azure Front Door Standard or Premium deployment, the forwarding protocol is configured in the routing rule. By default, the endpoint will accept both HTTP and HTTPS requests and then forward the request using the same incoming protocol.

To force end-to-end encryption and use only HTTPS, you must modify the routing rule. To modify the routing rule, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Front Doors Standard/Premium**.
3. On the Front Doors Standard/Premium page, select the front door instance that you previously created. For example, select **fd-app1**.
4. Click the **Endpoint manager** blade.
5. On the Endpoint manager screen, click **Edit endpoint** for the endpoint you want to configure the custom domain on.
6. In the Routes section, click the name of the route to modify. For example, click **default-route.**
7. On the Update route screen, locate the Accepted protocols dropdown menu and select **HTTPS only**.
8. In the Forwarding protocol field, select **HTTPS only**.
9. Click **Update**. Figure 3-41 displays the updated configuration.


Update route
default-route

A route maps your domains and matching URL path patterns to a specific origin group.
Learn more

Name: default-route

Status: [✓ Enabled] Disabled

Domains
Domains *: az700fdapp.z01.azurefd.net

Patterns to match: /*
/path

Accepted protocols *: HTTPS only

Redirect: [✓] Redirect all traffic to use HTTPS

Origin group
Origin group *: default-origin-group

Origin path:

Forwarding protocol *: HTTP only [✓ HTTPS only] Match incoming request

FIGURE 3-41 Update route

Configure multisite listeners and configure backend targets

With Azure Front Door Standard and Premium, the names of the components that are configured are different from application gateways and the original Front Door service. Figure 3-42 outlines how the named components connect.

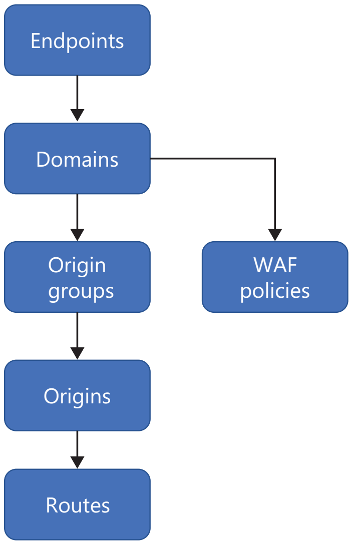


FIGURE 3-42 Azure Front Door Standard/Premium components

You can configure multiple endpoints per Azure Front Door instance. Each endpoint would have its own azurefd.net FQDN. Each endpoint can then have one or more domains associated with it. This would be the equivalent of a multi-site listener on an application gateway.

Each domain that you add must be verified if the DNS zone is not managed in your Azure subscription. The verification process is to add a TXT record with the confirmation string. Domains are then associated with origin groups, as well as WAF policies.

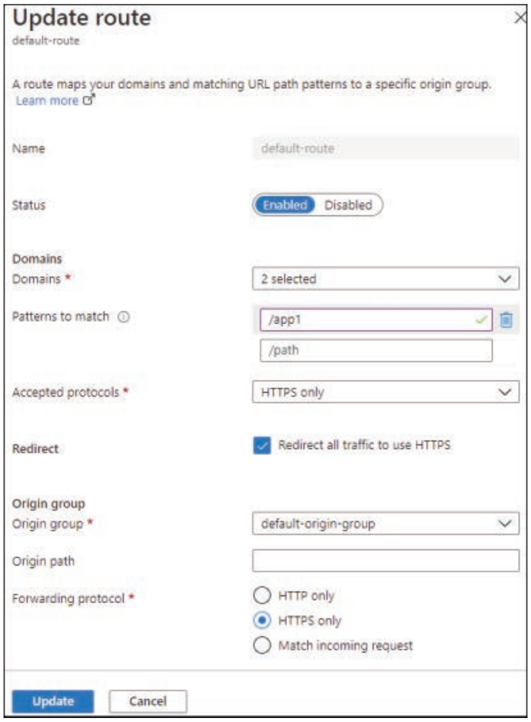
Origin groups are the equivalent of backend pool for application gateways. In an origin group, you create individual origins, which are the resources that you want to be able to communicate with through Front Door. These could be Azure resources such as App Services or Traffic Manager, or public IP addresses for on-premises or other applications hosted in other cloud providers.

Configure routing rules

Routes specify how the domains are associated with the origin groups. You can specify specific patterns to match in the incoming request URL to restrict which URLs Front Door responds to. By default, the first route will accept requests on **/***, which would be any directory on the endpoint. Routes are also where you can set whether to use HTTP, HTTPS, or both protocols on the incoming connection and the forward to the origin.

To configure the routing rules for an existing front door instance, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Front Doors Standard/Premium**.
3. On the Front Doors Standard/Premium page, select the front door instance that you previously created. For example, select **fd-app1**.
4. Click the **Endpoint manager** blade.
5. On the Endpoint manager screen, click **Edit endpoint** for the endpoint you want to configure the custom domain on.
6. In the Routes section, click the name of the route to modify. For example, click **default-route**.
7. In the Update route screen, locate the "Patterns to match" field and click the trashcan icon to delete the entry for **/***.
8. In the Patterns to match field, enter **/app1**.
9. Click **Update**. Figure 3-43 displays the updated configuration.


Update route
default-route

A route maps your domains and matching URL path patterns to a specific origin group.
Learn more

Name	default-route
Status	[✓ Enabled] Disabled
Domains	
Domains *	2 selected
Patterns to match	/app1
	/path
Accepted protocols *	HTTPS only
Redirect	✓ Redirect all traffic to use HTTPS
Origin group	
Origin group *	default-origin-group
Origin path	
Forwarding protocol *	HTTP only [✓ HTTPS only] Match incoming request

FIGURE 3-43 Update route

Making the change to the Patterns to match field will require that incoming traffic include / app1 on the incoming request to the endpoint. If the pattern matches, then the traffic will be forwarded to the origin group using HTTPS.

Skill 3.5: Implement an Azure Traffic Manager profile

Traffic Manager falls into the load balancer discussion because it provides a way to distribute traffic across multiple backend resources, whether they are Azure resources in a single region, in multiple regions, or even hosted outside of Azure. However, instead of load balancing at either layer 4 or layer 7, Traffic Manager load balances by using DNS.

When a client initiates a connection that uses Traffic Manager, the routing method that you configure determines which backend resource the client should connect to. But instead of facilitating the connection as an application gateway or Front Door does, Traffic Manager provides the client with the DNS name of the resource to connect to, and the client connects directly to the resource. Therefore, Traffic Manager is simply a DNS-based load balancer.

This skill covers how to:

- Configure a routing method
- Configure endpoints
- Create HTTP settings

Configure a routing method

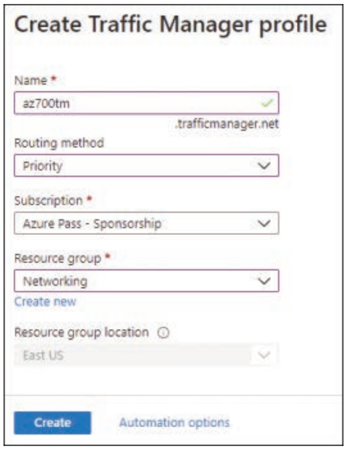
A routing method is required when you configure a Traffic Manager profile and determines how a backend resource, called an endpoint in Traffic Manager, is selected for the client connection. There are six routing methods that are available, and you must select one for the Traffic Manager profile. The routing methods are:

- **Performance.** Performance relies on the latency to the endpoint to provide the client connection with the lowest latency connection possible at that time. Traffic Manager tracks the latency for each endpoint by using an internal Internet Latency Table to track round-trip time.
- **Weighted.** Weighted enables you to distribute traffic across multiple endpoints at the ratio that you specify. If you provide the same weight to all endpoints, then the traffic would be distributed evenly.
- **Priority.** Priority enables you to set a priority for each endpoint to allow for active/passive types of configurations. The endpoint that has the lowest integer is considered to be the highest priority.
- **Geographic.** Geographic relies on the client connection's geography to direct users to specific endpoints. This is useful in compliance or data sovereignty scenarios where the end-user location can determine which endpoint to connect to.
- **MultiValue.** MultiValue enables the service to respond with multiple endpoints with one DNS query. MultiValue can be used only with External endpoints that are IPv4 or IPv6 addresses, not FQDNs.

- **Subnet.** Subnet routing allows you to map IP ranges to certain endpoints. Then, if a connection request is received from an IP in that range, Traffic Manager responds with the endpoint that you have configured. If you need one endpoint for certain IP ranges and a different endpoint for other ranges, this routing method is useful.

To configure a Traffic Manager profile in an active/passive configuration, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Traffic Manager profiles**.
3. On the Traffic Manager page, click **Create**.
4. In the Name field, provide a name for the Traffic Manager profile. The name must be globally unique, as it creates a **.trafficmanager.net** domain name. For example, name the profile **az700tm**.
5. In the Routing method dropdown menu, select **Priority**.
6. In the Subscription dropdown menu, select the desired subscription for the Traffic Manager resource. This does not have to be in the same subscription as the endpoints.
7. In the Resource group field, select an appropriate resource group. For example, select **Networking**.
8. Click **Create**. Figure 3-44 displays the completed configuration.


Create Traffic Manager profile

Name * az700tm .trafficmanager.net
Routing method Priority
Subscription * Azure Pass - Sponsorship
Resource group * Networking
Resource group location East US

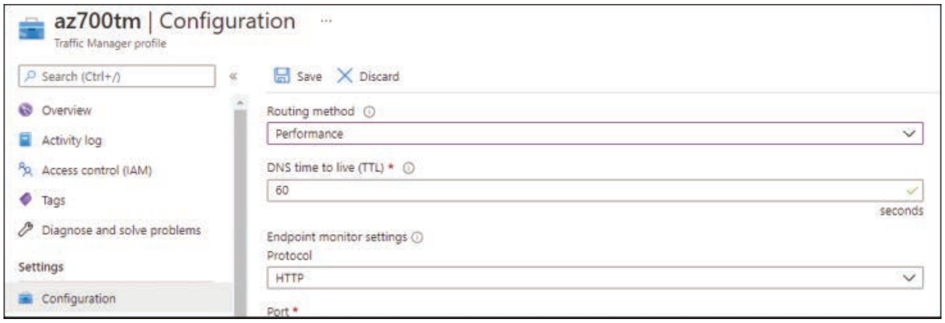
FIGURE 3-44 Create Traffic Manager profile

These steps create the Traffic Manager profile with the Priority routing method. This would then require at least two endpoints to be defined in the profile with priority values added for each endpoint. The endpoint with the lowest integer value would receive all of the traffic, as long as the health probe shows the endpoint as healthy.

To change the routing method of an existing Traffic Manager profile, follow these steps.

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Traffic Manager profiles**.

3. On the Traffic Manager page, select the previously created profile, **az700tm**.
4. On the Traffic Manager profile page, click the **Configuration** blade.
5. In the Routing method dropdown menu, select the desired routing method. For example, select **Performance**.
6. Click **Save**. Figure 3-45 displays a portion of the Configuration blade.


az700tm | Configuration
Traffic Manager profile

Routing method: Performance
DNS time to live (TTL) *: 60 seconds
Endpoint monitor settings
Protocol: HTTP
Port *

FIGURE 3-45 Traffic Manager configuration

Configure endpoints

Traffic Manager profiles require endpoints as the backend resources that clients connect directly to. Traffic Manager acts as a recursive DNS service that responds with an endpoint based on the routing method. The overall steps for this process are:

1. A client sends a request to connect to an app, such as a website at *app1.contoso.com*.
2. The client's IP configuration asks the configured DNS server the IP address of app1. contoso.com.
3. This FQDN is a CNAME that points to a Traffic Manager profile. The profile handles the DNS request and responds to the DNS server with an endpoint, based on the routing method.
4. The DNS server provides the endpoint information to the client as the DNS response.
5. The client connects directly to the endpoint and caches the DNS query for the TTL configured in Traffic Manager.

Figure 3-46 diagrams the process for each of these steps.

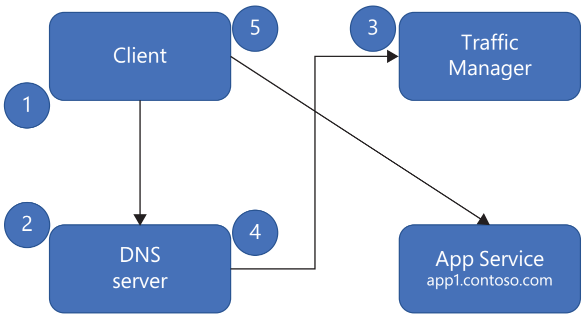


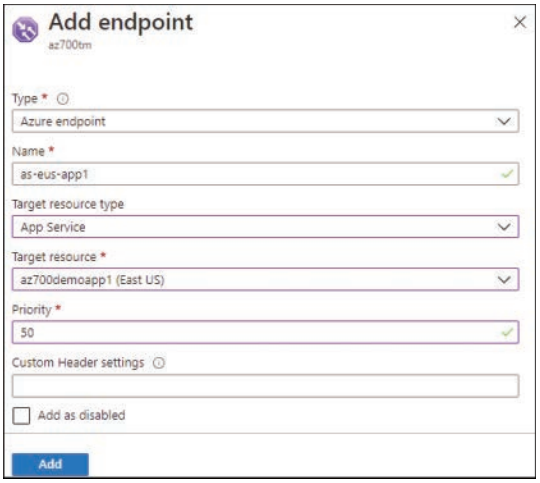
FIGURE 3-46 Traffic Manager DNS diagram

There are three types of endpoints that you can configure in a Traffic Manager profile:

- **Azure endpoint.** These are resources that you have deployed in your Azure environment.
- **External endpoint.** This is a public IP address or FQDN outside of your Azure environment.
- **Nested endpoint.** This is for high-availability deployments where one endpoint is another Traffic Manager profile in a different region for failover.

To create an app service as an Azure endpoint, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Traffic Manager profiles**.
3. On the Traffic Manager page, select the previously created profile, **az700tm**.
4. On the Traffic Manager profile page, click the **Endpoints** blade.
5. On the Endpoints blade, click **Add**.
6. Ensure that the Type dropdown menu is set to **Azure endpoint**.
7. In the Name field, provide a name for the endpoint. For example, enter **as-eus-app1**.
8. In the Target resource type dropdown field, select **App Service**.
9. In the Target resource dropdown field, select an app service that you previously deployed. For example, select **az700demoapp1**.
10. In the Priority field, set a priority for the target resource. The lower the integer, the higher in priority the resource will be. For example, set this to **50**.
11. Leave the remaining fields set to default and click **Add**. Figure 3-47 displays the completed configuration.



Add endpoint
az700tm

Type * Azure endpoint
Name * as-eus-app1
Target resource type App Service
Target resource * az700demoapp1 (East US)
Priority * 50
Custom Header settings
Add as disabled

FIGURE 3-47 Add endpoint

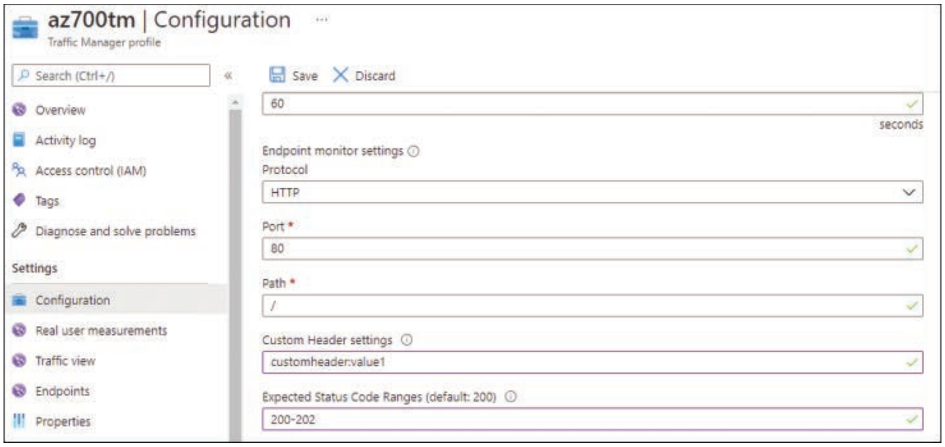
If you navigate to the Traffic Manager profile URL, you will now be redirected to the app service because it is the only endpoint that has been added. Repeat these steps as many times as necessary to add the backend resources to the Traffic Manager profile. Note that you cannot mix Azure endpoints and External endpoints in the same profile configuration.

Create HTTP settings

The objective "Create HTTP settings" is interesting because as we discussed in this skill section, Traffic Manager is a recursive DNS service that does not handle application (HTTP/HTTPS) traffic. The only settings that you can configure in a Traffic Manager profile that contain HTTP are the health probe and custom headers that can be added. You can add up to eight pairs of custom headers and values to the Custom Header settings field.

To customize the health probe settings, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Traffic Manager profiles**.
3. On the Traffic Manager page, select the previously created profile, **az700tm**.
4. On the Traffic Manager profile page, click the **Configuration** blade.
5. In the Custom Header settings field, enter any headers to add to the health probe in the **header:value** format.
6. In the Expected Status Code Ranges field, specify additional HTTP status codes for the health probe to accept. For example, enter **200-202**.
7. Click **Save**. Figure 3-48 displays the updated configuration.



60 seconds

Endpoint monitor settings

Protocol: HTTP

Port *: 80

Path *: /

Custom Header settings: customheader:value1

Expected Status Code Ranges (default: 200): 200-202

FIGURE 3-48 Traffic Manager configuration

Skill 3.6: Design and implement an Azure Virtual Network NAT

When you deploy virtual machines, you associate the network interface card of that VM to a subnet. The subnet is then part of a broader virtual network. If you've read the chapter that includes Network Security Groups, then you might remember that all outbound traffic is allowed by default from a virtual machine. By default, if the virtual machine does not have a public IP address assigned to the NIC, it connects to the internet using a random IP address that is allocated to that Azure region.

This skill covers how to:

- Choose when to use a Virtual Network NAT
- Allocate public IP or public IP prefixes for a NAT gateway
- Associate a Virtual Network NAT with a subnet

Choose when to use a Virtual Network NAT

Having a random outgoing IP address could be problematic if the VM needs to connect to specific external services that require a firewall rule or to know the source IP of your VM. You can achieve this by using a NAT gateway. A NAT gateway is a resource that you configure with a public IPv4 address, or range of public IPv4 addresses, and associate with a virtual network. Then any VM within the virtual network will use an IP address that you configure when communicating with the internet. Figure 3-49 displays the basic design of using a NAT gateway.

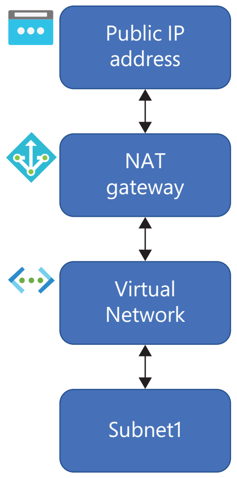


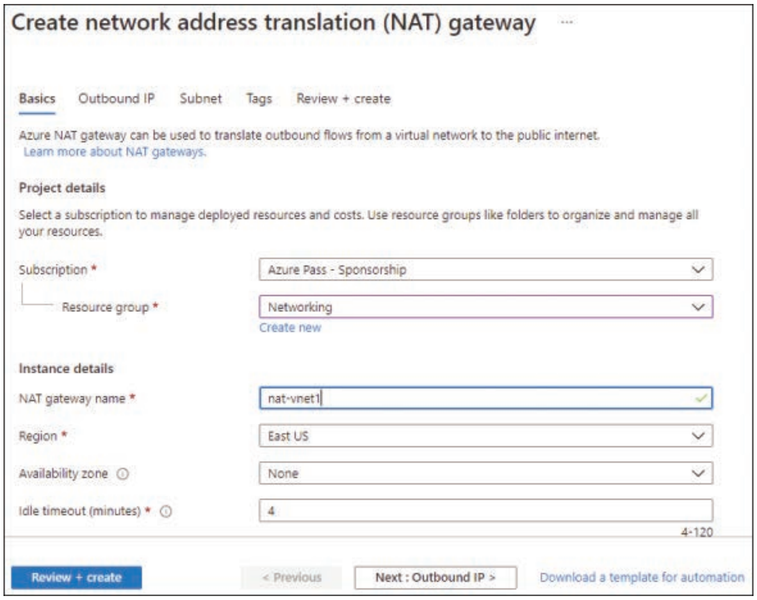
FIGURE 3-49 NAT gateway design

Allocate public IP addresses for a NAT gateway

When you create a NAT gateway, you need at least one public IP address to the gateway. This is the IP address that will be put in the header of the outbound packet from any VMs in the virtual network.

To create a NAT gateway, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **NAT gateways**.
3. On the NAT gateways page, click **Create**.
4. In the Subscription dropdown, select the desired subscription for the NAT gateway. This must be the same as the virtual network that you plan to associate the NAT gateway with.
5. In the Resource group dropdown, select the desired resource group. For example, select **Networking**.
6. In the NAT gateway name field, provide a name for the gateway. For example, enter **nat-vnet1**.
7. In the Region dropdown, select the desired Azure region. This must be the same region as your virtual network. For example, select **East US**.
8. Leave the remaining fields set to their default values, and click **Next: Outbound IP**. Figure 3-50 displays the completed Basics tab.


Create network address translation (NAT) gateway

Azure NAT gateway can be used to translate outbound flows from a virtual network to the public internet. Learn more about NAT gateways.

Project details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

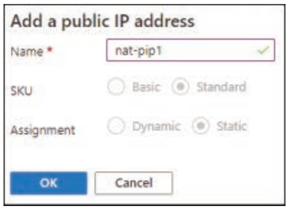
Subscription *	Azure Pass - Sponsorship
Resource group *	Networking

Instance details

NAT gateway name *	nat-vnet1
Region *	East US
Availability zone	None
Idle timeout (minutes) *	4

FIGURE 3-50 Create NAT gateway basics

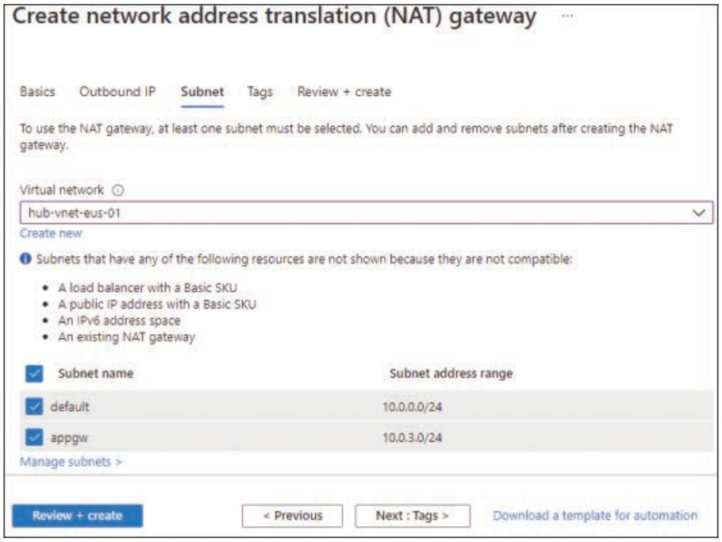
9. On the Outbound IP tab, click **Create a new public IP address**.
10. In the Add a public IP address field, name the new public IP address **nat-pip1**.
11. Click **OK**. Figure 3-51 displays the Add a public IP address screen.


Add a public IP address

Name *	nat-pip1
SKU	Basic [✓ Standard]
Assignment	Dynamic [✓ Static]

FIGURE 3-51 Add a public IP address

12. Click **Next: Subnet**.
13. In the Virtual network dropdown menu, select the desired virtual network to associate the NAT gateway with. For example, select **hub-vnet-eus-01**.
14. The available subnets will display below the selection. Select the checkboxes next to the desired subnets to include with the NAT gateway.
15. Click **Review + create**. Figure 3-52 displays the completed Subnet tab.
16. Click **Create**.


Create network address translation (NAT) gateway

To use the NAT gateway, at least one subnet must be selected. You can add and remove subnets after creating the NAT gateway.

Virtual network: hub-vnet-eus-01

Subnets that have any of the following resources are not shown because they are not compatible:

- A load balancer with a Basic SKU
- A public IP address with a Basic SKU
- An IPv6 address space
- An existing NAT gateway

Subnet name	Subnet address range
✓ default	10.0.0.0/24
✓ appgw	10.0.3.0/24

FIGURE 3-52 Create NAT gateway - Subnets

Associate a virtual network NAT with a subnet

When you create the NAT gateway, you can select the subnets that exist at that time to associate the NAT gateway with. However, if you add subnets to the same virtual network, they are *not* associated with the NAT gateway by default.

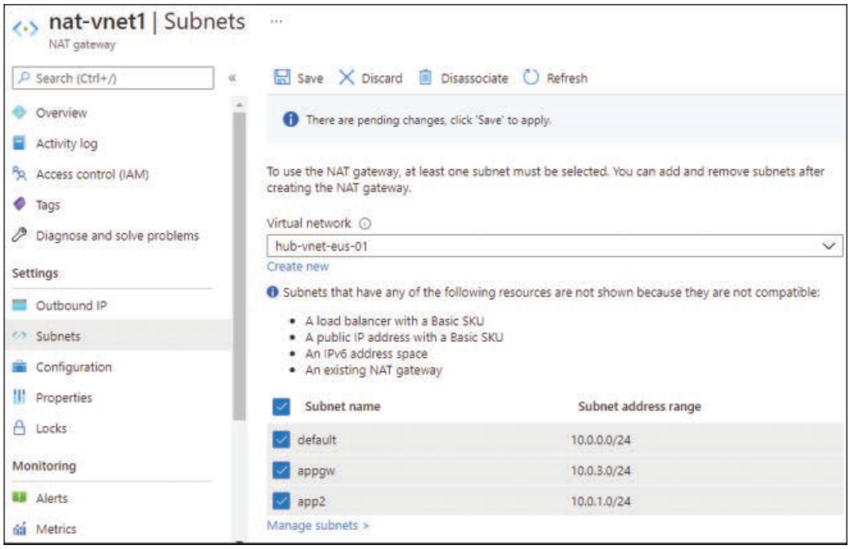
You can make the association either from the subnets page within the virtual network or from the NAT gateway. To associate the subnet with the NAT gateway from the subnet page, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **Virtual networks**.
3. On the virtual network page, select the previously associated virtual network. For example, select **hub-vnet-eus-01**.
4. On the selected virtual network blade, select the **Subnets** blade.
5. On the Subnets page, click the newly created subnet. For example, click **app2**.
6. In the NAT gateway dropdown menu, select the NAT gateway. For example, select **nat-vnet1**.
7. Click **Save**. **Note:** As of this writing, there is a UI bug that prevents you from saving the configuration using this method.

If you want to make the association from the NAT gateway, follow these steps:

1. Sign in to the Azure portal at *https://portal.azure.com*.
2. In the search bar, search for and select **NAT gateways**.

3. On the NAT gateways page, select the existing NAT gateway. For example, select **nat-vnet1**.
4. On the NAT gateway, click the **Subnets** blade.
5. Select the checkbox next to the newly created subnet. For example, select **app2**.
6. Click **Save**. Figure 3-53 displays the updated subnet association.



There are pending changes, click 'Save' to apply.

To use the NAT gateway, at least one subnet must be selected. You can add and remove subnets after creating the NAT gateway.

Virtual network: hub-vnet-eus-01

Subnets that have any of the following resources are not shown because they are not compatible:

- A load balancer with a Basic SKU
- A public IP address with a Basic SKU
- An IPv6 address space
- An existing NAT gateway

	Subnet name	Subnet address range
✓	default	10.0.0.0/24
✓	appgw	10.0.3.0/24
✓	app2	10.0.1.0/24

FIGURE 3-53 NAT gateway subnets

Chapter summary

- The built-in system routes allow outbound traffic from VMs in a virtual network to the internet.
- Subnets within a virtual network can communicate with each other without any additional routing requirements.
- User-defined routes can manipulate the traffic flow to deviate from the default routing.
- When you create user-defined routes to forward traffic through an on-premises firewall, the concept is named forced tunneling.
- Azure load balancers are available in Basic and Standard SKUs.
- Load balancers provide layer 4 load balancing across the backend pool that you specify.
- Load balancers can be either internal, with a private IP address, or external, with a public IP address.
- Load Balancer Basic SKU is a free load balancer service but has limited backend pool options and is not highly available.

- Load balancers have a frontend IP address, backend pool, health probes, and rules that define how traffic is forwarded.
- Load balancers can include NAT rules to translate incoming addresses and port numbers.
- Azure application gateways provide layer 7 load balancing across the backend pool that you specify.
- Application gateways can be scaled to multiple instances either manually or with auto-scaling rules.
- Application gateways provide domain-based and path-based routing to specific back-end resources.
- Application gateways include frontend IP addresses, backend pools, listeners, health probes, and rules that define how traffic is forwarded.
- Application gateways can rewrite headers before forwarding to a backend pool.
- Application gateways can perform both TLS termination and end-to-end TLS encryption.
- Azure Front Door combines layer 7 load balancing, security, content delivery and con-tent optimization into one service.
- Front Door is a global service that is not deployed into a single region.
- Front Door is deployed as an endpoint that can have custom domains associated with it.
- Front Door has origin groups that contain origins, or backend resources, that clients connect to.
- Azure Traffic Manager is a DNS-based routing service that uses recursive DNS to point client connections to a backend resource.
- Traffic Manager uses routing methods to determine how to select an endpoint, or back-end resource, to the client.
- When using Traffic Manager, clients connect directly to the endpoint and only use Traffic Manager for DNS.
- NAT gateways provide a way for VMs to share an outbound IP address.
- NAT gateways are associated with one virtual network and selectable subnets within that virtual network.
- NAT gateways only support IPv4 public IP addresses.

Thought experiment

In this thought experiment, demonstrate your skills and knowledge of the topics covered in this chapter. You can find the answers in the section that follows.

An organization is planning to deploy a new Azure subscription for a three-tier application. The first tier, the web tier, will be the frontend of a web app that is hosted in Azure App Services. The app service communicates with a custom API in a logic tier that will be hosted on a virtual machine scale set (VMSS). The third tier is the data tier, which will be hosted on Azure SQL Managed Instances.

The VMSS will communicate with a third-party reporting software. The third-party vendor has a requirement that all incoming traffic must be from the same IP address. All outgoing traffic from the VMs must use the same public IP address

The organization plans to deploy this application into four Azure regions across North America and Europe. As users connect to the web app, they should be directed to the nearest instance of the application based on their geographic location. The ingress solution must provide the ability to do end-to-end TLS encryption and provide security reporting.

1. What should the organization use as an ingress solution?
2. What should the organization deploy for the app services to communicate with the VMSS?
3. What should the organization deploy for the VMSS to communicate with the third-party vendor?

Thought experiment answers

This section contains the solution to the thought experiment. Each answer explains why the answer choice is correct.

1. What should the organization use as an ingress solution?

   Based on the overall scenario, it might seem that either an application gateway or Azure Front Door would work as an ingress solution. However, the requirement for security reporting points the solution toward Azure Front Door. Additionally, because the organization is using at least four Azure regions, Front Door would most likely be a better cost option than deploying application gateways in every region, which would most likely cost more.
2. What should the organization deploy for the app services to communicate with the VMSS?

   As an inbound solution to the VMSS, an Azure load balancer would be sufficient to distribute traffic from the app services to the VMSS.
3. What should the organization deploy for the VMSS to communicate with the third-party vendor?

The virtual network that the VMSS is associated with should be configured with a NAT gateway. The NAT gateway can have a single public IP address associated with it, which will represent the VMSS as it communicates outbound. This IP address can be given to the third-party vendor to be allow-listed in their firewall.

NEED MORE REVIEW? **SCALABLE WEB APP**

For more information on a reference architecture for a scalable web app, visit *https://docs.microsoft.com/en-us/azure/architecture/reference-architectures/app-service-web-app/scalable-web-app*.

Index