9400 XP

Architect network infrastructure in Azure

8 hr 52 min remaining • Learning Path • 0 of 12 modules completed

Intermediate Solutions Architect Azure Virtual Network ExpressRoute VPN Gateway Application Gateway Traffic Manager

Learn how to architect a secure, flexible, network infrastructure in Azure and connect on-premises networks to your Azure resources.

Prerequisites

- Familiarity with network concepts, such as IP addressing and routing
- Familiarity with network connectivity methods, such as VPN

Modules in this learning path

Connect your on-premises network to Azure with VPN Gateway

500 XP

32 min remaining • Module • 2 of 5 units completed

4.6 (1,030)

VPN Gateway in Azure provides secure connectivity between your on-premises networks and clients.

Continue

Overview

- Introduction — 2 min ✓
- Connect on-premises networks to Azure by using site-to-site VPN gateways — 5 min ✓
- Exercise - Prepare Azure and on-premises virtual networks by using Azure CLI commands — 6 min
- Exercise - Create a site-to-site VPN gateway by using Azure CLI commands — 25 min
- Summary — 1 min

Connect your on-premises network to the Microsoft global network by using ExpressRoute

800 XP

40 min remaining • Module • 0 of 5 units completed

4.7 (807)

Connect your on-premises systems and users to Azure and Office 365 by using ExpressRoute for private, dedicated, and guaranteed throughput connectivity.

Overview

Secure and isolate access to Azure resources by using network security groups and service endpoints

600 XP

43 min remaining • Module • 0 of 6 units completed

4.6 (1,072)

Network security groups and service endpoints help you secure your virtual machines and Azure services from unauthorized network access.

Overview

Distribute your services across Azure virtual networks and integrate them by using virtual network peering

600 XP

42 min remaining • Module • 0 of 6 units completed

★★★★★ 4.7 (666)

Use virtual network peering to enable communication across virtual networks in a way that's secure and minimally complex.

Overview

Enhance your service availability and data locality by using Azure Traffic Manager

600 XP

29 min remaining • Module • 0 of 6 units completed

★★★★½ 4.5 (538)

Azure Traffic Manager provides DNS load balancing to your application, so you improve your ability to distribute your application around the world. Use Traffic Manager to improve the performance and availability of your application.

Overview

Improve application scalability and resiliency by using Azure Load Balancer

800 XP

47 min remaining • Module • 0 of 6 units completed

★★★★★ 4.7 (462)

Discuss the different load balancers in Azure and how to choose the right Azure load balancer solution to meet your requirements.

Overview

Load balance your web service traffic with Application Gateway

800 XP

1 hr 32 min remaining • Module • 0 of 7 units completed

★★★★½ 4.6 (583)

Improve application resilience by distributing load across multiple servers and use path-based routing to direct web traffic.

Overview

Manage and control traffic flow in your Azure deployment with routes

900 XP

50 min remaining • Module • 0 of 7 units completed

★★★★½ 4.6 (402)

Learn how to control Azure virtual network traffic by implementing custom routes.

Overview

Design an IP addressing schema for your Azure deployment

1300 XP

37 min remaining • Module • 0 of 6 units completed

★★★★★ 4.7 (216)

A good Azure IP addressing schema provides flexibility, room for growth, and integration with on-premises networks. The schema ensures that communication works for deployed resources, minimizes public exposure of systems, and gives the organization flexibility in its

network. If not properly designed, systems might not be able to communicate, and additional work will be required to remediate.

Overview S

Design a hybrid network architecture on Azure

1000 XP

47 min remaining • Module • 0 of 6 units completed

4.6 (175)

You have a traditional on-premises infrastructure that you need to connect to resources in Azure. In this module, you learn how to select a connectivity method for your use cases that balances functionality, cost, and security.

Overview

Centralize your core services by using hub and spoke Azure virtual network architecture

800 XP

36 min remaining • Module • 0 of 6 units completed

4.7 (303)

Design a network architecture in Azure that allows for growth and flexibility, secure isolation of critical resources, low administrative overhead, and communication with on-premises network resources.

Overview

Monitor and troubleshoot your end-to-end Azure network infrastructure by using network monitoring too..

700 XP

37 min remaining • Module • 0 of 6 units completed

4.6 (596)

Use Network Watcher tools, diagnostics, and logs to help find and fix networking issues in your Azure infrastructure.

Overview

R Previous

Unit 3 of 5 S

Next T

100 XP

Exercise - Prepare Azure and on-premises virtual networks by using Azure CLI commands

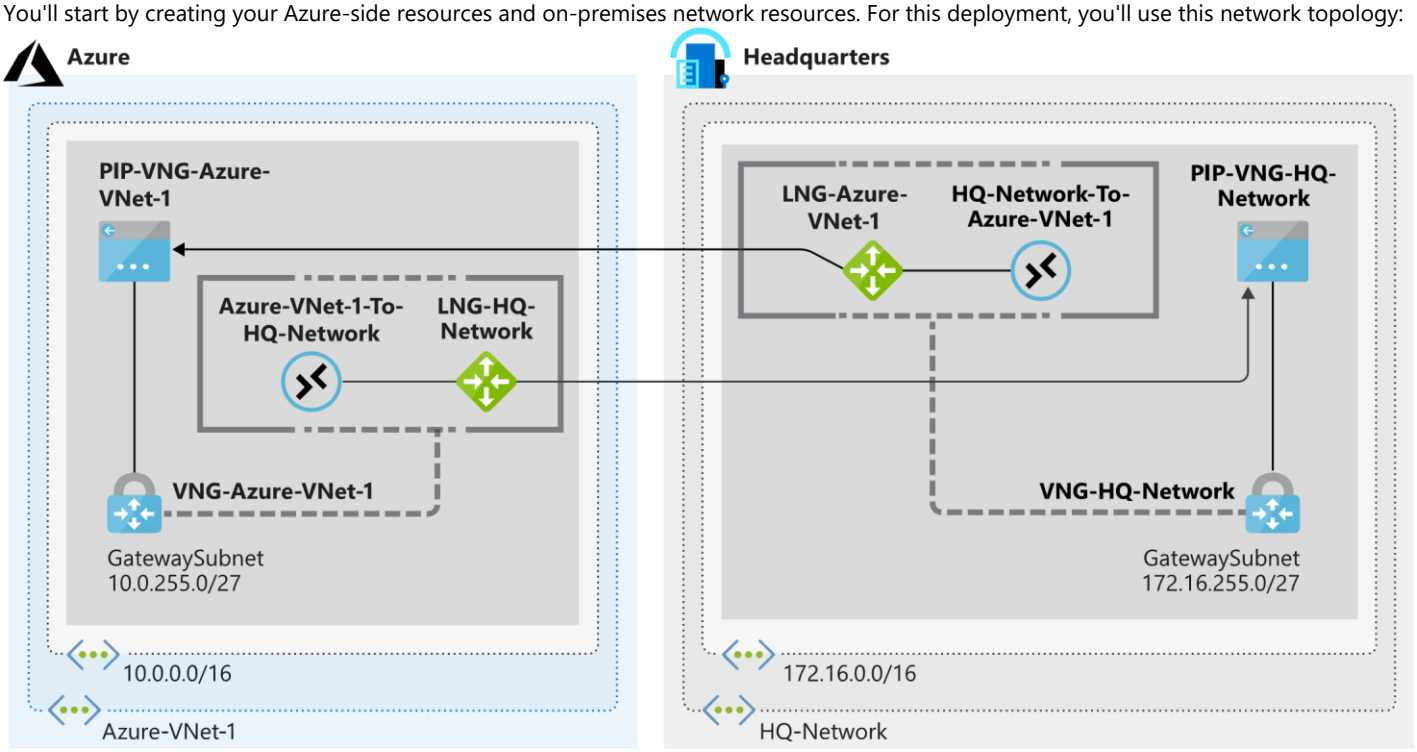
6 minutes

This module requires a sandbox to complete. A **sandbox** gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

Activate sandbox

Your company is now ready to deploy a site-to-site VPN. This VPN will allow your on-premises servers to connect to resources in Azure. You can then start to securely share data from each of your sites, and you can use resources hosted in Azure across your organization.

You'll start by creating your Azure-side resources and on-premises network resources. For this deployment, you'll use this network topology:



In this exercise, you'll simulate an on-premises datacenter (HQ-Network) by using an additional Azure virtual network. There are many makes and models of on-premises VPN devices, and it won't be possible to describe their configuration in this unit. The logical method of configuration is the same for a VPN device. You just need to replace the steps for HQ-Network with steps tailored to your on-premises device.

In the diagram, notice that the local network gateway names in each location reflect the *target* networks rather than the *source* network. This is a good practice. It clarifies that the local network gateway refers to the *other* network that you're connecting to.

In this unit, you'll configure the virtual networks with a subnet, add a gateway subnet, and then create the local network gateway by using th Azure CLI.

Create the Azure-side resources

1. Run this command in Azure Cloud Shell to create the **Azure-VNet-1** virtual network and the **Services** subnet.

Azure CLI = Copy

```
az network vnet create \
  --resource-group [sandbox resource group name] \
  --name Azure-VNet-1 \
  --address-prefix 10.0.0.0/16 \
  --subnet-name Services \
  --subnet-prefix 10.0.0.0/24
```

2. Run this command in Cloud Shell to add the **GatewaySubnet** subnet to **Azure-VNet-1**.

Azure CLI = Copy

```
az network vnet subnet create \
  --resource-group [sandbox resource group name] \
  --vnet-name Azure-VNet-1 \
  --address-prefix 10.0.255.0/27 \
  --name GatewaySubnet
```

3. Run this command in Cloud Shell to create the **LNG-HQ-Network** local network gateway.

Azure CLI = Copy

```
az network local-gateway create \
  --resource-group [sandbox resource group name] \
  --gateway-ip-address 94.0.252.160 \
  --name LNG-HQ-Network \
  --local-address-prefixes 172.16.0.0/16
```

This gateway represents the on-premises network that you're connecting to. The IP address specified as the remote gateway (which is the simulated on-premises network) will need to be updated later because it doesn't exist yet in our scenario.

·ate the simulated on-premises network and supporting resources

1. Run this command in Cloud Shell to create the **HQ-Network** virtual network and the **Applications** subnet.

Azure CLI = Copy

```
az network vnet create \
  --resource-group [sandbox resource group name] \
  --name HQ-Network \
  --address-prefix 172.16.0.0/16 \
  --subnet-name Applications \    --subnet-prefix 172.16.0.0/24
```

2. Run this command in Cloud Shell to add **GatewaySubnet** to **HQ-Network**.

Azure CLI = Copy

```
az network vnet subnet create \
  --resource-group [sandbox resource group name] \
  --address-prefix 172.16.255.0/27 \
  --name GatewaySubnet \    --vnet-name HQ-
Network
```

3. Run this command in Cloud Shell to create the **LNG-Azure-VNet-1** local network gateway.

Azure CLI = Copy

```
az network local-gateway create \
  --resource-group [sandbox resource group name] \
  --gateway-ip-address 94.0.252.160 \    --name LNG-Azure-
VNet-1 \
  --local-address-prefixes 10.0.0.0/16
```

This gateway describes the Azure network that you're connecting to. You'll update the IP address specified as the remote gateway (which is in Azure) later.

Verify the topology

1. Run this command in Cloud Shell to verify that the virtual networks have been successfully created.

Azure CLI Copy

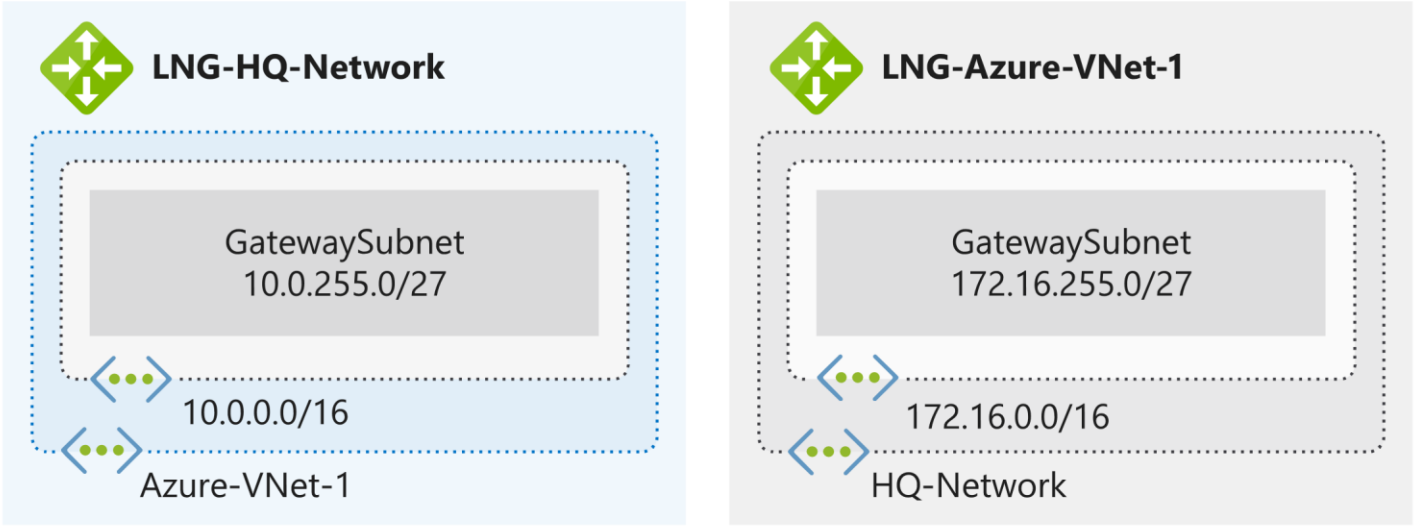
```
az network vnet list --output table
```

2. Run this command in Cloud Shell to verify that the local network gateways have been successfully created.

Azure CLI Copy

```
az network local-gateway list \
    --resource-group [sandbox resource group name] \
    --output table
```

This diagram shows the resources that you've deployed:



Next unit: Exercise - Create a site-to-site VPN gateway by using Azure CLI commands

Continue >

English (United States)

Azure Cloud Shell

This module requires a sandbox to complete. A sandbox gives you access to Azure resources. Your Azure subscription

will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

R Previous Unit 4 of 5 S Next T

100 XP

Exercise - Create a site-to-site VPN gateway by using Azure CLI commands

25 minutes

This module requires a sandbox to complete. A **sandbox** gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

Activate sandbox

You're now ready to complete your site-to-site VPN gateway by creating the public IP addresses, virtual network gateways, and connections. Remember that you used placeholders for the public IP address references when you created your local network gateways. So one of your tasks now is to update these gateways with the actual public IP addresses assigned to your virtual network gateways.

Ideally, the public IP addresses and virtual network gateways should be created *before* the local network gateways. In this exercise, you'll see how to update the local network gateways. You can use the same commands to update any configuration elements in the local network gateways, like remote network address spaces.

Create the Azure-side VPN gateway

First, you'll create the VPN gateway for the Azure end of the connection. It can take up to 45 minutes to create a virtual network gateway. To save time, you'll use Azure CLI commands with the `--no-wait` parameter. This parameter lets you create both virtual network gateways simultaneously to minimize the overall time required to create these resources.

1. Run this command in Cloud Shell to create the **PIP-VNG-Azure-VNet-1** public IP address.

Azure CLI — Copy

```
az network public-ip create \
  --resource-group [sandbox resource group name] \
  --name PIP-VNG-Azure-VNet-1 \
  --allocation-method Dynamic
```

2. Run this command in Cloud Shell to create the **VNG-Azure-VNet-1** virtual network gateway.

Azure CLI — Copy

```
az network vnet-gateway create \
  --resource-group [sandbox resource group name] \
  --name VNG-Azure-VNet-1 \
  --public-ip-address PIP-VNG-Azure-VNet-1 \
  --vnet Azure-VNet-1 \
  --gateway-type Vpn \
  --vpn-type RouteBased \
  --sku VpnGw1 \
  --no-wait
```

Create the on-premises VPN gateway

Next, you'll create a VPN gateway to simulate an on-premises VPN device.

1. Run this command in Cloud Shell to create the **PIP-VNG-HQ-Network** public IP address.

Azure CLI — Copy

```
az network public-ip create \
  --resource-group [sandbox resource group name] \
  --name PIP-VNG-HQ-Network \
  --allocation-method Dynamic
```

2.

Run these commands in Cloud Shell to create the **VNG-HQ-Network** virtual network gateway.

Azure CLI = Copy

```
az network vnet-gateway create \
  --resource-group [sandbox resource group name] \
  --name VNG-HQ-Network \
  --public-ip-address PIP-VNG-HQ-Network \
  --vnet HQ-Network \
  --gateway-type Vpn \
  --vpn-type RouteBased \
  --sku VpnGw1 \
  --no-wait
```

3.

To monitor the progress of the gateway creation, run the following command. We're using the Linux watch command to run the az network vn gateway list command periodically, which allows you to monitor the progress.

bash = Copy

```
watch -d -n 5 az network vnet-gateway list \    --resource-group [sandbox resource
group name] \
  --output table
```

After each VPN gateway shows a **ProvisioningState** of **Succeeded**, you're ready to continue. Press Ctrl+C to halt the command after the gateway is created.

output = Copy

```
ActiveActive    EnableBgp    GatewayType    Location      Name           ProvisioningState    ResourceGroup           ResourceGuid                          VpnType
--------------  -----------  -------------  ------------  -------------  -------------------  ----------------------  ------------------------------------  ----------
False           False        Vpn            southcentralus  VNG-Azure-VNet-1  Succeeded         [sandbox resource group name]  48dc714e-a700-42ad-810f-a8163ee8e001  RouteBased
False           False        Vpn            southcentralus  VNG-HQ-Network    Succeeded         [sandbox resource group name]  49b3041d-e878-40d9-a135-58e0ecb7e48b  RouteBased
```

date the local network gateway IP references

) **Important**

ur virtual network gateways must be successfully deployed before you start the next exercise.

s section, you'll update the remote gateway IP address references that are defined in the local network gateways. You can't update the loc ork gateways until you've created the VPN gateways and an IPv4 address is assigned to and associated with them. You can use this Azure nand to check whether both virtual network gateways have been created:

Azure CLI = Co

```
az network vnet-gateway list \
 --resource-group [sandbox resource group name] \
 --query "[?provisioningState=='Succeeded']" \
 --output table
```

ember to wait until the lists of gateways are successfully returned. Also, remember that the local network gateway resources define the gs of the *remote* gateway and network that they're named after. For example, the **LNG-Azure-VNet-1** local network gateway contains

information like the IP address and networks for **Azure-VNet-1**.

1. Run this command in Cloud Shell to retrieve the IPv4 address assigned to **PIP-VNG-Azure-VNet-1**.

```
bash
```

```
PIPVNGAZUREVNET1="$(az network public-ip show \
  --resource-group [sandbox resource group name] \
  --name PIP-VNG-Azure-VNet-1 \
  --query "[ipAddress]" \
  --output tsv)"
```

2. Run this command in Cloud Shell to update the **LNG-Azure-VNet-1** local network gateway so that it points to the public IP address attached to the **VNG-Azure-VNet-1** virtual network gateway.

```
Azure CLI
```

```
az network local-gateway update \
  --resource-group [sandbox resource group name] \
  --name LNG-Azure-VNet-1 \
  --gateway-ip-address $PIPVNGAZUREVNET1
```

3. Run this command in Cloud Shell to retrieve the IPv4 address assigned to **PIP-VNG-HQ-Network**.

```
bash
```

```
PIPVNGHQNETWORK="$(az network public-ip show \
  --resource-group [sandbox resource group name] \
  --name PIP-VNG-HQ-Network \
  --query "[ipAddress]" \
  --output tsv)"
```

4. Run this command in Cloud Shell to update the **LNG-HQ-Network** local network gateway so that it points to the public IP address attached to the **VNG-HQ-Network** virtual network gateway.

```
Azure CLI
```

```
az network local-gateway update \
  --resource-group [sandbox resource group name] \
  --name LNG-HQ-Network \
  --gateway-ip-address $PIPVNGHQNETWORK
```

Create the connections

You'll now complete the configuration by creating the connections from each VPN gateway to the local network gateway that contains the public IP address references for that gateway's remote network.

1. Create the shared key to use for the connections. In the following command, replace <shared key> with a text string to use for the IPSec pre-shared key. The pre-shared key is a string of printable ASCII characters no longer than 128 characters. You'll use this pre-shared key on both connections.

```
bash
```

```
SHAREDKEY=<shared key>
```

2. Remember that **LNG-HQ-Network** contains a reference to the IP address on your simulated on-premises VPN device. Run this command in Cloud Shell to create a connection from **VNG-Azure-VNet-1** to **LNG-HQ-Network**.

```
Azure CLI
```

```
az network vpn-connection create \
  --resource-group [sandbox resource group name] \
  --name Azure-VNet-1-To-HQ-Network \
  --vnet-gateway1 VNG-Azure-VNet-1 \
  --shared-key $SHAREDKEY \
  --local-gateway2 LNG-HQ-Network
```

3.

Remember that **LNG-Azure-VNet-1** contains a reference to the public IP address associated with the **VNG-Azure-VNet-1** VPN gateway. This connection would normally be created from your on-premises device. Run this command in Cloud Shell to create a connection from **VNG-HQ-Network** to **LNG-Azure-VNet-1**.

Azure CLI = Copy

```
az network vpn-connection create \
  --resource-group [sandbox resource group name] \
  --name HQ-Network-To-Azure-VNet-1  \
  --vnet-gateway1 VNG-HQ-Network \    --shared-key $SHAREDKEY
\
  --local-gateway2 LNG-Azure-VNet-1
```

'e now finished the configuration of the site-to-site connection. This may take a few minutes, but the tunnels should automatically ect and become active.

rification steps

1. confirm that the VPN tunnels are connected.

Run the following command to confirm that **Azure-VNet-1-To-HQ-Network** is connected.

Azure CLI = Copy

```
az network vpn-connection show \
  --resource-group [sandbox resource group name] \
  --name Azure-VNet-1-To-HQ-Network  \
  --output table \
  --query '{Name:name,ConnectionStatus:connectionStatus}'
```

You should see output like below indicating the connection is successful. If the `ConnectionStatus` shows as `Connecting`, wait a minute or two and rerun the command. The connections can take a few minutes to fully connect.

output = Copy

```
Name                    ConnectionStatus
----------------------  ------------------
Azure-VNet-1-To-HQ-Network  Connected
```

2. Now lets confirm the corresponding **HQ-Network-To-Azure-VNet-1** connection is also established.

Azure CLI = Copy

```
az network vpn-connection show \
  --resource-group [sandbox resource group name] \
  --name HQ-Network-To-Azure-VNet-1  \
  --output table \
  --query '{Name:name,ConnectionStatus:connectionStatus}'
```

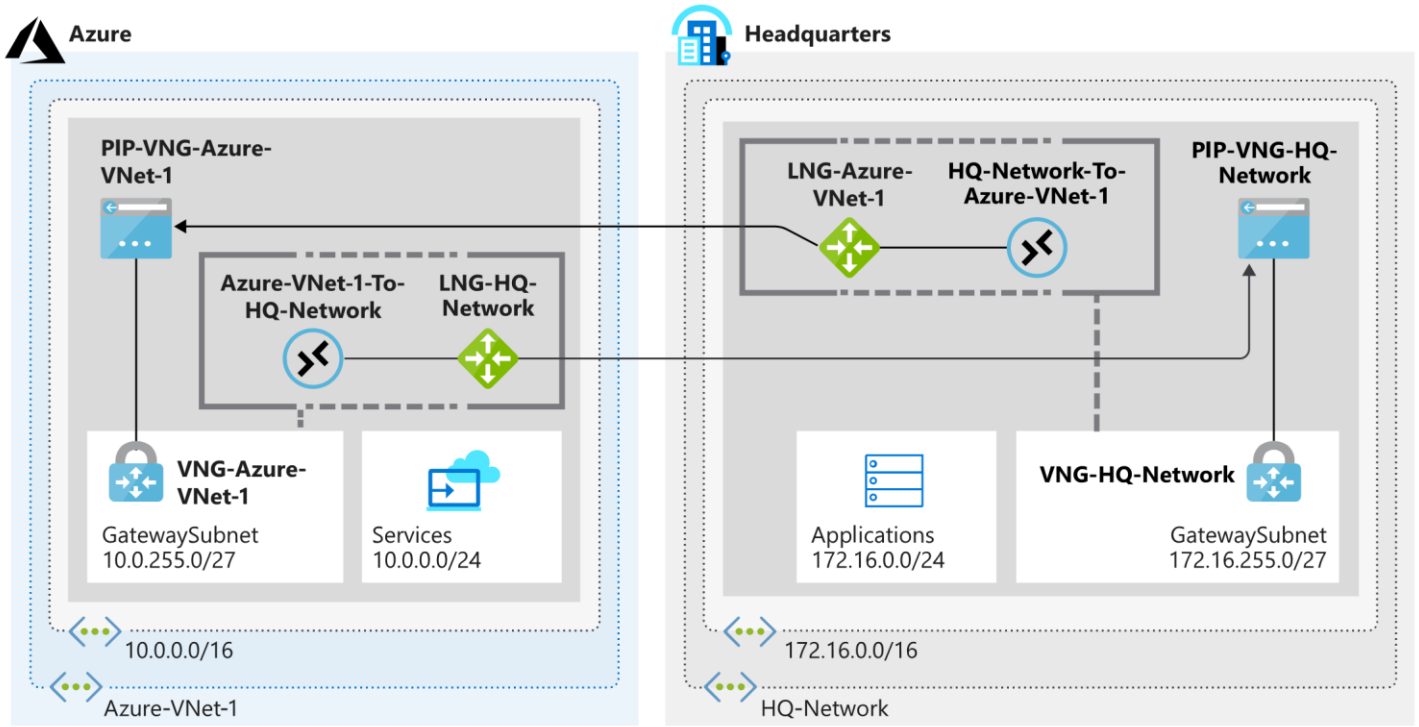
You should see the following output indicating this connection is also successful.

output = Copy

```
Name                    ConnectionStatus
----------------------  ------------------
HQ-Network-To-Azure-VNet-1  Connected
```

ite-to-site configuration is now complete. Your final topology, including the subnets, and connections, with logical connection points, is n in this diagram. Virtual machines deployed in the **Services** and **Applications** subnets can now communicate with each other, now

the VPN connections have been successfully established.



Next unit: Summary

Continue >

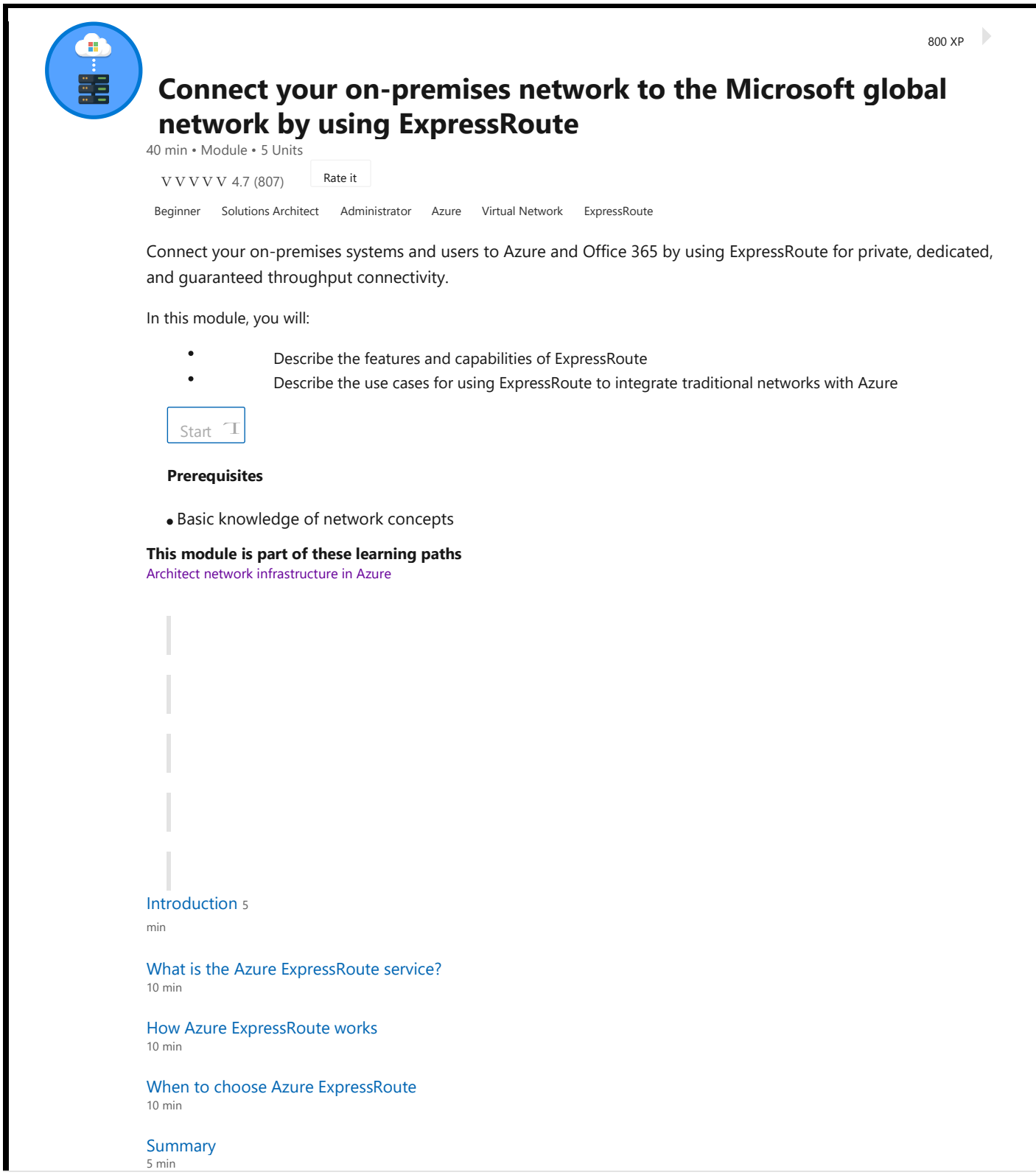
English (United States)

Previous Version Docs • Blog • Contribute • Privacy & Cookies • Terms of Use • Trademarks

Azure Cloud Shell

This module requires a sandbox to complete. A sandbox gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on

2/21/2020 Connect your on-premises network to the Microsoft global network by using ExpressRoute - Learn | Microsoft Docs



https://docs.microsoft.com/en-us/learn/modules/connect-on-premises-network-with-expressroute/ 1/1

2/21/2020 Introduction - Learn | Microsoft Docs

Unit 1 of 5 Next

100 XP

Introduction

5 minutes

You can use the Azure ExpressRoute service to extend your on-premises networks into the Microsoft cloud. Connections are made over a private high-bandwidth connection. The ExpressRoute service provides a secure and reliable way to connect your on-premises network directly to Azure.

Imagine you're the solution architect for a financial organization that has begun migrating resources to Azure. The organization has systems that need to communicate between an on-premises network and Azure, and it doesn't want this traffic traversing the internet. These applications have higher bandwidth requirements and need to have consistent network performance.

The organization also uses Office 365. It wants to reduce traffic over the internet and send this traffic over a dedicated connection to Azure. The organization believes ExpressRoute meets its needs, but it wants to understand more about the service and whether it should include the service in the infrastructure.

Your goal is to identify whether ExpressRoute is the correct service to use to allow connectivity from on-premises networks to the Microsoft cloud.

Learning objectives

In this module, you will:

- Describe the features and capabilities of ExpressRoute
- Describe the use cases for using ExpressRoute to integrate traditional networks with Azure

Prerequisites

- Basic knowledge of network concepts

Next unit: What is the Azure ExpressRoute service?

Continue

200 XP

What is the Azure ExpressRoute service?

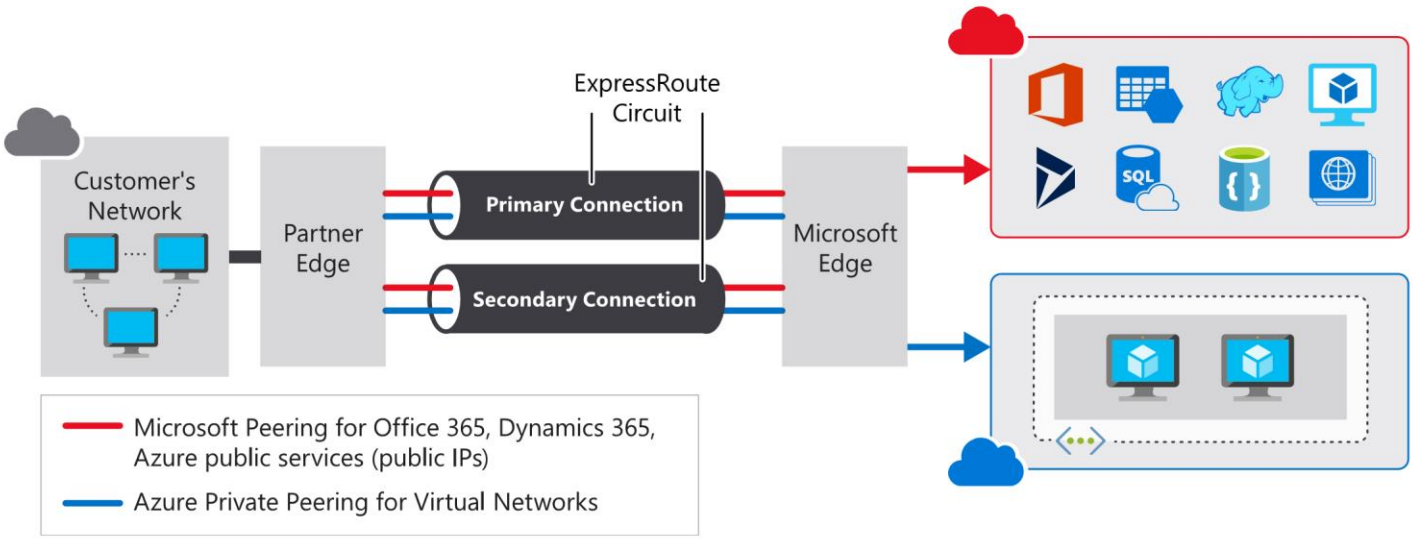
10 minutes

As part of the project for the finance company, you need to understand:

- What Azure ExpressRoute is.
- How ExpressRoute integrates with on-premises and Azure networks.
- The benefits that ExpressRoute provides compared to other site-to-site connectivity options.
- Whether ExpressRoute can provide the finance company with the best possible network performance.

ExpressRoute overview

Azure ExpressRoute lets you seamlessly extend your on-premises networks into the Microsoft cloud. This connection between your organization and Azure is dedicated and private. Establishing an ExpressRoute connection enables you to connect to Microsoft cloud services like Azure, Office 365, and Dynamics 365. Security is enhanced, connections are more reliable, latency is minimal, and throughput is greatly increased.



Features and benefits of ExpressRoute

There are several benefits to using ExpressRoute as the connection service between Azure and on-premises networks.

Layer 3 connectivity

ExpressRoute provides Layer 3 (address-level) connectivity between your on-premises network and the Microsoft cloud through connectivity partners. These connections can be from a point-to-point, any-to-any network, or they can be virtual cross-connections through an exchange.

Built-in redundancy

Each connectivity provider uses redundant devices to ensure that connections established with Microsoft are highly available. You can configure multiple circuits to complement this feature. All redundant connections are configured with Layer 3 connectivity to meet SLAs.

Connectivity to Microsoft cloud services

ExpressRoute enables direct access to the following services in all regions:

- Microsoft Office 365
- Microsoft Dynamics 365

- Azure compute services, such as Azure Virtual Machines
- Azure cloud services, such as Azure Cosmos DB and Azure Storage

Office 365 was created to be accessed securely and reliably via the internet. Because of this, we recommend ExpressRoute for specific scenarios. The "Learn more" section at the end of this module includes a link about using ExpressRoute to access Office 365.

Across on-premises connectivity with ExpressRoute Global Reach

You can enable ExpressRoute Global Reach to exchange data across your on-premises sites by connecting your ExpressRoute circuits. For example, assume that you have a private datacenter in California connected to ExpressRoute in Silicon Valley. You have another private datacenter in Texas connected to ExpressRoute in Dallas. With ExpressRoute Global Reach, you can connect your private datacenters through two ExpressRoute circuits. Your cross-datacenter traffic will travel through the Microsoft network.

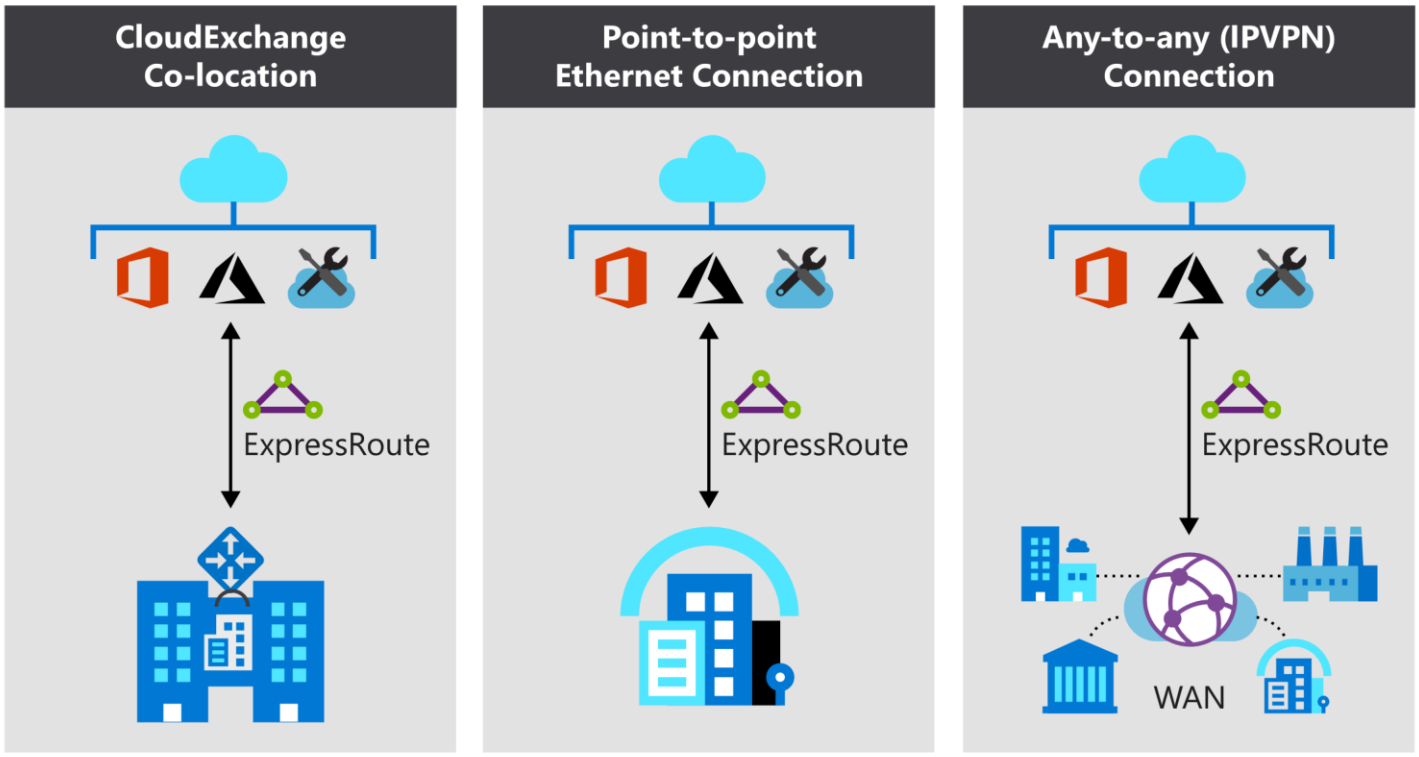
Dynamic routing

ExpressRoute uses the Border Gateway Protocol (BGP) routing protocol. BGP is used to exchange routes between on-premises networks and resources running in Azure. This protocol enables dynamic routing between your on-premises network and services running in the Microsoft cloud.

ExpressRoute connectivity models

ExpressRoute supports three models that you can use to connect your on-premises network to the Microsoft cloud:

- CloudExchange co-location
- Point-to-point Ethernet connection
- Any-to-any connection



Co-location at a cloud exchange

Co-located providers can normally offer both Layer 2 and Layer 3 connections between your infrastructure, which might be located in the co-location facility, and the Microsoft cloud. For example, if your datacenter is co-located at a cloud exchange such as an internet service provider (ISP), you can request a virtual cross-connection to the Microsoft cloud.

Point-point Ethernet connection

Point-to-point connections provide Layer 2 and Layer 3 connectivity between your on-premises site and Microsoft Azure. You can connect your offices or datacenters to Azure by using the point-to-point links. For example, if you have an on-premises datacenter, you can use a point-topoint Ethernet link to connect to Microsoft.

Any-to-any networks

With point-to-point connectivity, you can integrate your wide area network (WAN) with Microsoft Azure by providing connections to your offices and datacenters. Azure will integrate with your WAN connection to provide a seamless connection, just like you would have between your datacenter and any branch offices.

With point-to-point connections, all WAN providers offer Layer 3 connectivity. For example, if you already use Multiprotocol Label Switching (MPLS) to connect to your branch offices or other sites in your organization, an ExpressRoute connection to Microsoft will behave just like another location on your private WAN.

Security considerations

With ExpressRoute, your data doesn't travel over the public internet, so it's not exposed to the potential risks associated with internet communications. ExpressRoute is a private connection from your on-premises infrastructure to your Azure infrastructure. Even if you have an ExpressRoute connection, DNS queries, certificate revocation list checking, and Azure Content Delivery Network requests are still sent over the public internet.

Check your knowledge

1. What is the Azure ExpressRoute service?

- It's a service that provides a VPN connection between on-premises and the Microsoft cloud.
- It's a service that encrypts your data in transit.
- It's a service that provides a direct connection from your on-premises datacenter to the Microsoft cloud. **This answer is correct.**
- It's a service that provides a site-to-site VPN connection between your on-premises network and the Microsoft cloud.

2. Which of the following is not a benefit of ExpressRoute?

- Redundant connectivity
- Consistent network throughput
- Encrypted network communication

  Correct. ExpressRoute does provide private connectivity, but it isn't encrypted.
- Access to Microsoft cloud services

Next unit: How Azure ExpressRoute works

Continue

How Azure ExpressRoute works

10 minutes

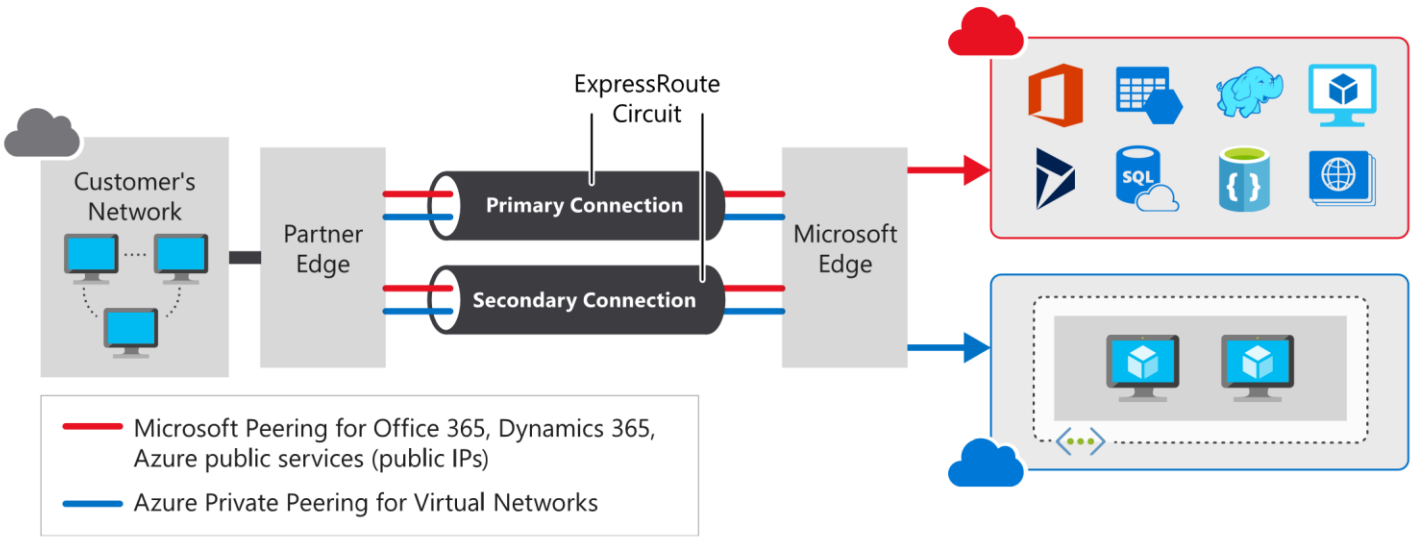
You've learned about the purpose of the Azure ExpressRoute service, and the services for which you can use it. Now you're ready to start learning about how the service works. You'll see how it interacts with Azure and on-premises networks to help create a secure and reliable connection between your on-premises datacenter and the Microsoft cloud.

In this unit, you'll learn how to create and use Azure circuits to connect your on-premises networks to the cloud. You'll see the steps that you need to take to create a circuit. You'll also learn about the other components of an ExpressRoute connection, which work together to form a connection from your on-premises datacenter to the Microsoft cloud.

Architecture of ExpressRoute

ExpressRoute is supported across all regions and locations. To implement ExpressRoute, you need to work with an ExpressRoute partner. The partner provides the *edge service*: an authorized and authenticated connection that operates through a partner-controlled router. The edge service is responsible for extending your network to the Microsoft cloud.

The partner sets up connections to an endpoint in an ExpressRoute location (implemented by a Microsoft edge router). These connections enable you to peer your on-premises networks with the virtual networks available through the endpoint. These connections are called *circuits*.



A circuit provides a physical connection for transmitting data through the ExpressRoute provider's edge routers to the Microsoft edge routers. A circuit is established across a private wire rather than the public internet. Your on-premises network is connected to the ExpressRoute provider's edge routers. The Microsoft edge routers provide the entry point to the Microsoft cloud.

Prerequisites for ExpressRoute

Before you can connect to Microsoft cloud services by using ExpressRoute, you need to have:

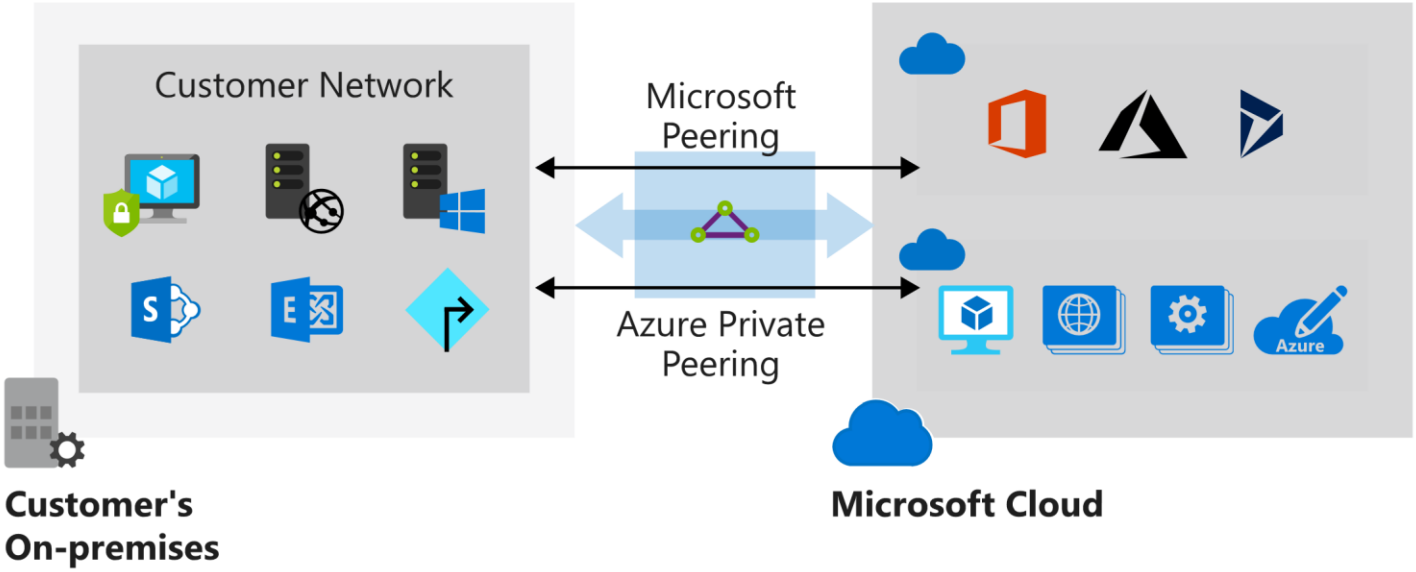
- An ExpressRoute connectivity partner or cloud exchange provider that can set up a connection from your on-premises networks to the Microsoft cloud.
- An Azure subscription that is registered with your chosen ExpressRoute connectivity partner.
- An active Microsoft Azure account that can be used to request an ExpressRoute circuit.
- An active Office 365 subscription, if you want to connect to the Microsoft cloud and access Office 365 services.

ExpressRoute works by peering your on-premises networks with networks running in the Microsoft cloud. Resources on your networks can communicate directly with resources hosted by Microsoft. To support these peerings, ExpressRoute has a number of network and routing requirements:

- Ensure that BGP sessions for routing domains have been configured. Depending on your partner, this might be their or your responsibility. Additionally, for each ExpressRoute circuit, Microsoft requires redundant BGP sessions between Microsoft's routers and your peering routers.
- You or your providers need to translate the private IP addresses used on-premises to public IP addresses by using a NAT service. Microsoft will reject anything except public IP addresses through Microsoft peering.
- Reserve several blocks of IP addresses in your network for routing traffic to the Microsoft cloud. You configure these blocks as either a /29 subnet or two /30 subnets in your IP address space. One of these subnets is used to configure the primary circuit to the Microsoft cloud, and the other implements a secondary circuit. You use the first address in these subnets to communicate with services in the Microsoft cloud. Microsoft uses the second address to establish a BGP session.

ExpressRoute supports two peering schemes:

- Use private peering to connect to Azure IaaS and PaaS services deployed inside Azure virtual networks. The resources that you access must all be located in one or more Azure virtual networks with private IP addresses. You can't access resources through their public IP address over a private peering.
- Use Microsoft peering to connect to Azure PaaS services, Office 365 services, and Dynamics 365.



> **Note**
>
> You can also use the Azure portal to configure public peering. This form of peering enables you to connect to the public addresses exposed by Azure services. However, this peering is deprecated and is not available for new circuits. This module does not describe public peering.

Create an ExpressRoute circuit and peering

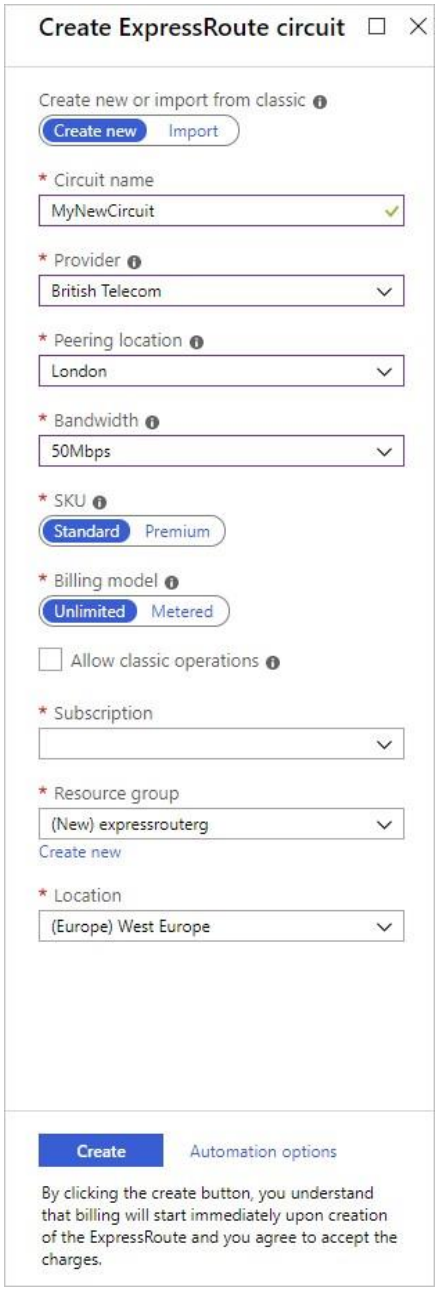
Establishing a connection to Azure through ExpressRoute is a multistep process. You can perform many of these steps either by using the Azure portal, or from the command line by using PowerShell or the Azure CLI. This section describes the process of using the Azure portal. For PowerShell and CLI instructions, see the "Learn more" section at the end of this module.

Create a circuit

When you're using the Azure portal, select **Create a resource** > **Networking** > **ExpressRoute**. The **Create ExpressRoute circuit** page requires you to complete the following fields:

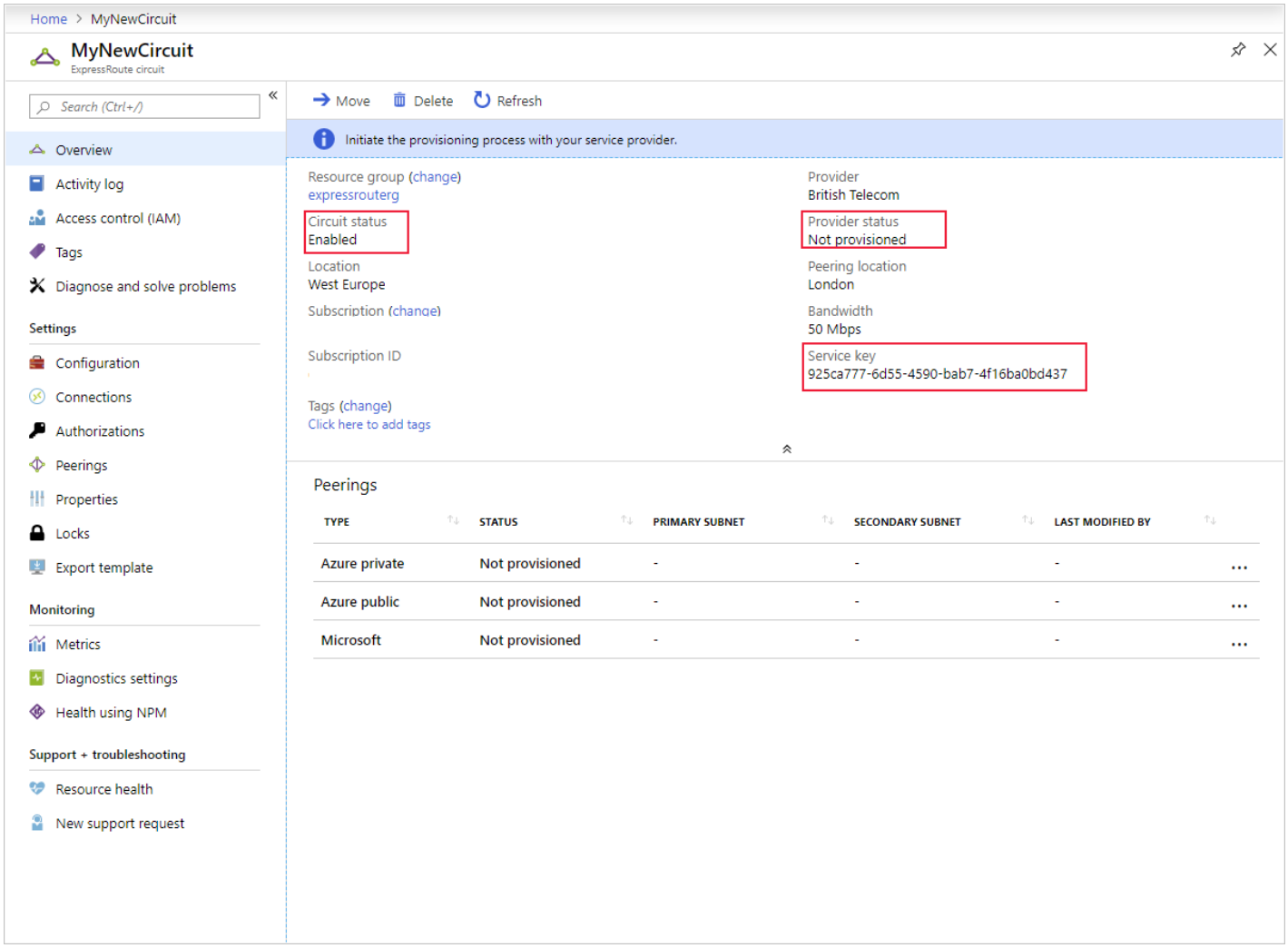
Property	Value
Circuit name	A meaningful name for your circuit, without any white space or special characters.
Provider	The ExpressRoute provider with which you've registered your subscription.

Property	Value
Peering location	A location enabled by the ExpressRoute provider in which to create your circuit.
Bandwidth	Select your bandwidth, from 50 Mbps up to 10 Gbps. Start with a low value. You can increase it later with no interruption to service. However, you can't reduce the bandwidth if you set it too high initially.
SKU	Select **Standard** if you have up to 10 virtual networks and only need to connect to resources in the same geopolitical region. Otherwise, select **Premium**.
Billing model	Select **Unlimited** to pay a flat fee regardless of usage. Or select **Metered** to pay according to the volume of traffic that enters and exits the circuit.
Subscription	The subscription you've registered with your ExpressRoute provider.
Resource group	The Azure resource group in which to create the circuit.
Location	The Azure location in which to create the circuit.



Circuit creation can take several minutes. After the circuit has been provisioned, you can use the Azure portal to view the properties. You'll see that **Circuit status** is enabled, meaning that the Microsoft side of the circuit is ready to accept connections. **Provider status** will be **Not provisioned** initially. This is because the provider hasn't configured their side of the circuit for connecting to your network.

You send the provider the value in the **Service key** field to enable them to configure the connection. This can take several days. You can revisit this page to check the provider status.



Create a peering configuration

After the provider status is reported as**Provisioned**, you can configure the routing for the peerings. These steps apply only to circuits that are created with service providers who offer Layer 2 connectivity. For any circuits that operate at Layer 3, the provider might be able to configure the routing for you.

The **ExpressRoute circuit**page (shown earlier) lists each peering and its properties. You can select a peering to configure these properties.

Configure private peering

You use private peering to connect your network to your virtual networks running in Azure. To configure private peering, you must provide the following information:

- **Peer ASN**. The autonomous system number for your side of the peering. This ASN can be public or private, and 16 bits or 32 bits.
- **Primary subnet** This is the address range of the primary /30 subnet that you created in your network. You'll use the first IP address in this subnet for your router. Microsoft uses the second for its router.
- **Secondary subnet** This is the address range of your secondary /30 subnet. This subnet provides a secondary link to Microsoft. The first two addresses are used to hold the IP address of your router and the Microsoft router.
- **VLAN ID**. This is the VLAN on which to establish the peering. The primary and secondary links will both use this VLAN ID.
- **Shared key**. This is an optional MD5 hash that's used to encode messages passing over the circuit.

Configure Microsoft peering

You use Microsoft peering to connect to Office 365 and its associated services. To configure Microsoft peering, you provide a peer ASN, a primary subnet address range, a secondary subnet address range, a VLAN ID, and an optional shared key as described for a private peering. You must also provide the following information:

- **Advertised public prefixes** This is a list of the address prefixes that you use over the BGP session. These prefixes must be registered to you, and must be prefixes for public address ranges.
- **Customer ASN** This is optional. It's the client-side autonomous system number to use if you are advertising prefixes that aren't registered to the peer ASN.
- **Routing registry name** This name identifies the registry in which the customer ASN and public prefixes are registered.

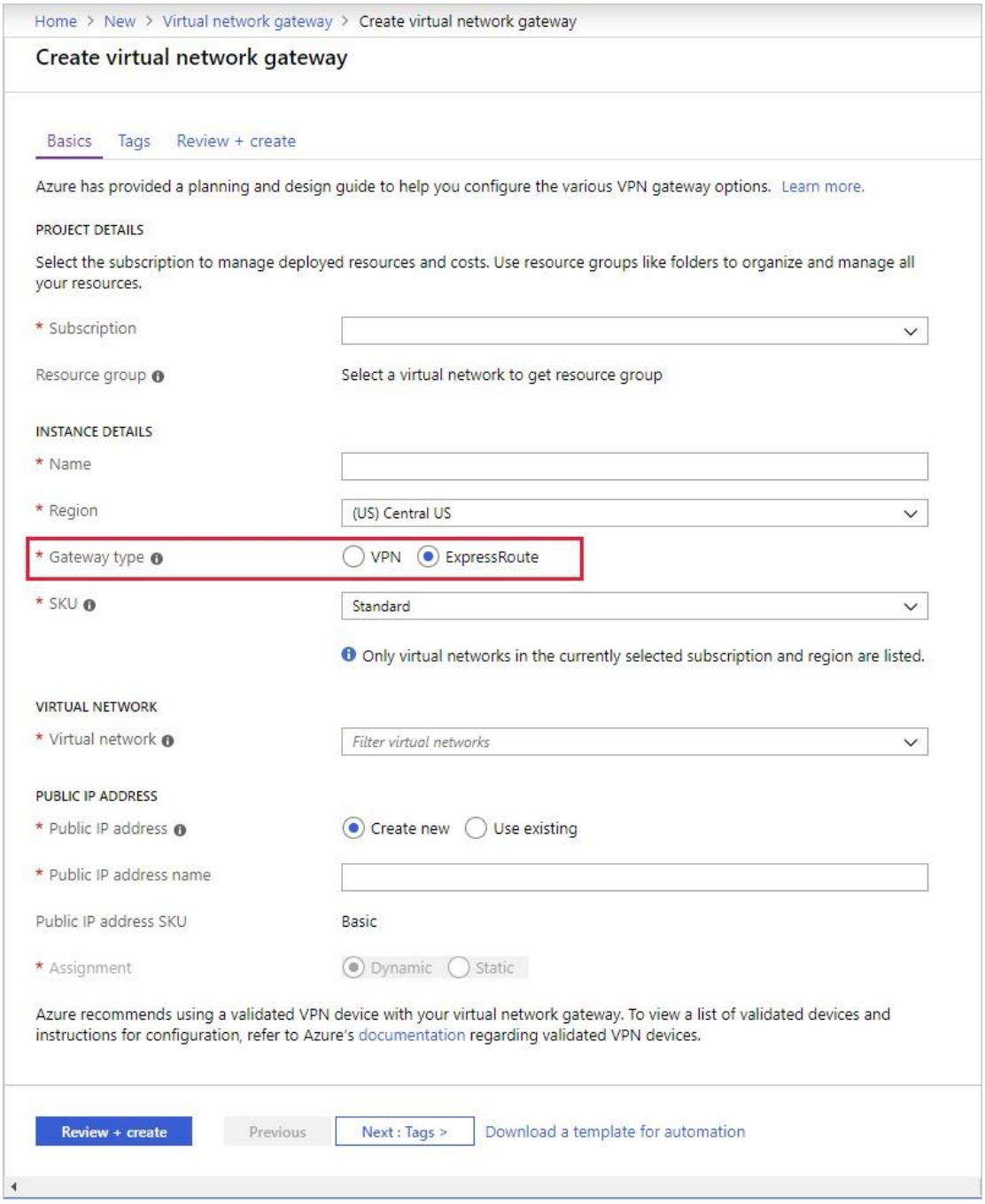
Connect a virtual network to an ExpressRoute circuit

After the ExpressRoute circuit has been established, Azure private peering is configured for your circuit, and the BGP session between your network and Microsoft is active, you can enable connectivity from your on-premises network to Azure.

Before you can connect to a private circuit, you must create an Azure virtual network gateway by using a subnet on one of your Azure virtual networks. The virtual network gateway provides the entry point to network traffic that enters from your on-premises network. It directs incoming traffic through the virtual network to your Azure resources.

You can configure network security groups and firewall rules to control the traffic that's routed from your on-premises network. You can also block requests from unauthorized addresses in your on-premises network.

> **Note**
>
> You must create the virtual network gateway by using the typ**ExpressRoute** and not **VPN**.



Up to 10 virtual networks can be linked to an ExpressRoute circuit, but these virtual networks must be in the same geopolitical region as the ExpressRoute circuit. You can link a single virtual network to four ExpressRoute circuits if necessary. The ExpressRoute circuit can be in the same subscription to the virtual network, or in a different one.

If you're using the Azure portal, you connect a peering to a virtual network gateway as follows:

1. On the **ExpressRoute circuit** page for your circuit, select **Connections**.
2. On the **Connections** page, select **Add**.
3. On the **Add connection** page, give your connection a name, and then select your virtual network gateway. When the operation has finished, your on-premises network will be connected through the virtual network gateway to your virtual network in Azure. The connection will be made across the ExpressRoute connection.

High availability and failover with ExpressRoute

In each ExpressRoute circuit, there are two connections from the connectivity provider to two different Microsoft edge routers. This configuration occurs automatically. It provides a degree of availability within a single location.

Consider setting up ExpressRoute circuits in different peering locations to provide high availability and help protect against a regional outage. For example, you might create circuits in the US East and US Central regions and connect these circuits to your virtual network. This way, if one ExpressRoute circuit goes down, you won't lose connectivity to your resource and you can fail over the connection to another ExpressRoute circuit.

You can also have multiple circuits across different providers to ensure that your network stays available even if an outage affects all circuits from a single approved provider. You can set the **Connection Weight** property to prefer one circuit to another.

ExpressRoute Direct and FastPath

Microsoft also provides an ultra-high-speed option called ExpressRoute Direct. This service enables dual 100-Gbps connectivity. It's suitable for scenarios that involve massive and frequent data ingestion. It's also suitable for solutions that require extreme scalability, such as banking, government, and retail.

You enroll your subscription with Microsoft to activate ExpressRoute Direct. For more information, visit the ExpressRoute article in the "Learn more" section at the end of this module.

ExpressRoute Direct supports FastPath. When FastPath is enabled, it sends network traffic directly to a virtual machine that's the intended destination. The traffic bypasses the virtual network gateway, improving the performance between Azure virtual networks and on-premises networks.

FastPath doesn't support virtual network peering (where you have virtual networks connected together). It also doesn't support user-defined routes on the gateway subnet.

Check your knowledge

1. What is Microsoft peering?

- It provides a direct connection from your on-premises network to an Azure datacenter.
- It enables you to connect your on-premises network to Office 365 services and Dynamics 365.

  This is the correct answer.
- It provides a connection between your on-premises network and an ExpressRoute provider.
- It provides a point-to-site connection for computers in your on-premises location to Office 365 services.

2. What is an ExpressRoute circuit?

- An ExpressRoute circuit implements a site-to-site connection across a VPN connection to an Azure datacenter.
- An ExpressRoute circuit is a direct hard-wired connection between your on-premises datacenter and an Azure datacenter.
- A backup service that provides connectivity to the Microsoft cloud if your VPN connection fails.
- A circuit provides a physical connection for transmitting data through the ExpressRoute provider's edge routers to the Microsoft edge routers.

  This is the correct answer.

3. What security benefits does Azure ExpressRoute provide?

- An ExpressRoute connection is automatically encrypted, to help protect traffic that passes across the internet to the Microsoft cloud.
- The speed at which data traverses an ExpressRoute connection makes it impossible to intercept by network monitors and packet sniffers.
- ExpressRoute uses a dedicated, private network to connect to the Microsoft cloud. Traffic doesn't traverse the public internet, so it's difficult to intercept.

This is the correct answer.

ExpressRoute uses a proprietary transmission protocol that constantly varies, so it's difficult to intercept traffic.

Next unit: When to choose Azure ExpressRoute

Continue T

200 XP

When to choose Azure ExpressRoute

10 minutes

You've learned how the Azure ExpressRoute service works, and how to connect your on-premises networks to the Microsoft cloud by using an ExpressRoute circuit. You've also learned about the different peering options available, and how to use a virtual network gateway to route requests between your on-premises network and the Microsoft cloud.

In this unit, you'll learn about the most common use cases for deploying ExpressRoute. You'll compare ExpressRoute to other connection options available for Azure, like site-to-site and point-to-site through a virtual network gateway. This information will to help you determine whether ExpressRoute is the most appropriate solution for your organization.

When to use Azure ExpressRoute

Consider using the Azure ExpressRoute service in the following scenarios:

- Low-latency connectivity to services in the cloud. In these situations, eliminating or reducing the network overhead will have a significant impact on the performance of your applications.
- Accessing high-volume systems in the cloud that consume or produce massive volumes of data quickly. ExpressRoute can move data around rapidly, with high reliability.
- Consuming Microsoft Cloud Services, such as Office 365 and Dynamics 365. ExpressRoute is especially useful if your organization has a large number of users who need to access these services concurrently.

Organizations that have migrated large-scale on-premises systems to Azure. Using ExpressRoute helps ensure that the results of the migrations are seamless for on-premises clients. They should notice no drop in performance. They might even experience some improvement if the previous on-premises systems were restricted by network bandwidth.

- Situations where data should not traverse the public internet for security reasons.
- Large datacenters, with a high number of users and systems accessing SaaS offerings.

Benefits of using ExpressRoute

ExpressRoute offers several advantages for building highly scalable, cloud-based solutions.

Predictable performance

Having a dedicated connection to the Microsoft cloud guarantees performance. There are no concerns over internet provider outages or spikes in internet traffic. With ExpressRoute, your providers are accountable to provide the necessary throughput and latency SLA.

Data privacy for your traffic

Traffic that's sent over ExpressRoute connection is as secure as using MPLS WAN links. There's no risk of internet monitoring or packet capture by malicious users.

High-throughput, low-latency connections

With ExpressRoute, you can obtain speeds of up to 10 Gbps when connecting to the Microsoft cloud. If you're using ExpressRoute Direct, you can achieve up to 100 Gbps. Latency is minimal, so your systems are highly responsive.

Availability and connectivity

Microsoft guarantees a minimum of 99.95 percent availability for an ExpressRoute dedicated circuit.

With ExpressRoute enabled, you can connect to Microsoft through one of several peering connections and have access to regions within the same geopolitical region. For example, if you connect to Microsoft through ExpressRoute in France, you'll have access to all Microsoft services hosted in Western Europe.

You can also enable ExpressRoute Premium, which provides cross-region accessibility. For example, if you access Micros in Germany, you'll have access to all Microsoft cloud services in all regions globally.

You can also take advantage of a feature called ExpressRoute Global Reach. It allows you to exchange data across all of datacenters by connecting all of your ExpressRoute circuits.

Alternatives to ExpressRoute

ExpressRoute is one of three solutions that you can use to connect your on-premises network to Azure. The others are site connection and a virtual network point-to-site connection.

Site-to-site VPN

An Azure site-to-site VPN connection enables you to connect your on-premises network to Azure over an IPsec tunnel t solution. You configure an on-premises VPN device with a public IP address. You connect this device to an Azure virtual Azure virtual network gateway.

Point-to-site VPN

With point-to-site VPN, you can establish a secure connection to a network from individual computers located on-prem useful for someone who wants to connect to Azure from remote locations such as a home or customer site. Point-to-sit only a few clients that need to connect to a virtual network.

Azure ExpressRoute vs. site-to-site and point-to-site VPN connections

The following table shows a comparison between ExpressRoute, point-to-site, and site-to-site networks with Azure.

Connection	Azure services supported	Bandwidth	Protocols	Typical use case
Virtual network, point-to-site	Azure IaaS services, Azure Virtual Machines	Based on the gateway SKU	Active/passive	Dev, test, and lab environments for cloud services and virtual machines.
Virtual network, site-to-site	Azure IaaS services, Azure Virtual Machines	Typically < 1 Gbps aggregate	Active/passive	Dev, test, and lab environments. Small-scale production workloads and virtual machines.
ExpressRoute	Azure IaaS and PaaS services, Microsoft Office 365 services	50 Mbps up to 10 Gbps (100 Gbps for ExpressRoute Direct)	Active/active	Enterprise-class and mission-critical workloads. Big data solutions.

Check your knowledge

1 When should you use Azure ExpressRoute instead of Azure site-to-site connectivity?

- [x] For handling enterprise-class and mission-critical workloads.

  This answer is

- [ ] For connecting mobile users directly to your virtual network running in Azure.
- [ ] For handling small-scale production workloads running on Azure virtual machines.
- [ ] To save connection costs for occasionally connected users to the Microsoft cloud.

2 Which connection type is best to use for Office 365 users?

- [ ] Point-to-site over a VPN connection through an Azure network gateway.
- [ ] Site-to-site over a VPN connection through an Azure network gateway.
- [x] An ExpressRoute connection.

  This is the correct

Next unit: Summary

Continue

Previous Unit 5 of 5

100 XP

Summary

5 minutes

In this module, you've learned about the Azure ExpressRoute service and how it can be used to connect your on-premises networks to the Microsoft cloud infrastructure. You've also learned how the service works and that you need to work with an approved connectivity provider establish the connections via an ExpressRoute circuit.

In particular, you saw the benefits and use cases for the ExpressRoute service. You saw how the service can provide an organization with a sa reliable, and fast connection to the Microsoft cloud.

Learn more

For more information on ExpressRoute, see the following articles on Microsoft Docs:

- Azure ExpressRoute for Office 365
- ExpressRoute routing requirements
- ExpressRoute partners and peering locations
- Create and modify an ExpressRoute circuit using PowerShell
- Create and modify an ExpressRoute circuit using CLI
- About ExpressRoute Direct

Module complete:

Unlock achievement

2/21/2020 Secure and isolate access to Azure resources by using network security groups and service endpoints - Learn | Microsoft Docs

600 XP

Secure and isolate access to Azure resources by using network security groups and service endpoints

43 min • Module • 6 Units

V V V V W 4.6 (1,072) Rate it

Beginner Solutions Architect Administrator Security Engineer Azure Virtual Network

Network security groups and service endpoints help you secure your virtual machines and Azure services from unauthorized network access.

In this module, you will:

Identify the capabilities and features of network security groups.
Identify the capabilities and features of virtual network service endpoints.
Use network security groups to restrict network connectivity.
Use virtual network service endpoints to control network traffic to and from Azure services.

- Knowledge of basic networking concepts, including subnets and IP addressing
- Basic familiarity with Azure services, specifically Azure SQL Database and Azure Storage
- Familiarity with Azure virtual machines and virtual networking

This module is part of these learning paths

Implement network security in Azure

Architect network infrastructure in Azure

Introduction
3 min

Use network security groups to control network access 8 min

Exercise - Create and manage network security groups 15 min

Secure network access to PaaS services with virtual network service endpoints 5 min

Exercise - Restrict access to Azure Storage by using service endpoints 10 min

Summary
2 min

https://docs.microsoft.com/en-us/learn/modules/secure-and-isolate-with-nsg-and-service-endpoints/ 1/1

2/21/2020 Introduction - Learn | Microsoft Docs

Unit 1 of 6 S Next T

100 XP

Introduction

3 minutes

Imagine you are the solution architect for a manufacturing company. Your company has several sites, and users throughout the company wil need to use an enterprise resource planning (ERP) application to migrate to Azure. The company will only consider moving key systems onto the platform if stringent security requirements can be met, including tight control over which computers have network access to the servers running the application. You want to secure both virtual machine networking and Azure services networking as part of your company's network security strategy. Your goal is to prevent unwanted or unsecured network traffic from being able to reach key systems.

You'll use network security groups to secure network traffic for virtual machines running on Azure. You'll learn to use virtual network service endpoints to control network traffic to and from Azure services, such as storage or database services.

Learning objectives

In this module, you will:

- Identify the capabilities and features of network security groups.
- Identify the capabilities and features of virtual network service endpoints.
- Use network security groups to restrict network connectivity.
- Use virtual network service endpoints to control network traffic to and from Azure services.

Prerequisites

- Knowledge of basic networking concepts, including subnets and IP addressing.
- Basic familiarity with Azure services, specifically Azure SQL Database and Azure Storage.
- Familiarity with Azure virtual machines and virtual networking.

Next unit: Use network security groups to control network access

Continue

100 XP

Use network security groups to control network access

8 minutes

As part of the project to move your ERP system to Azure, you need to ensure that servers have proper isolation, so that only allowed systems can make network connections. For example, you have database servers that store data for your ERP application. You want to block prohibited systems from communicating with the servers over the network, while allowing application servers to communicate with the database servers.

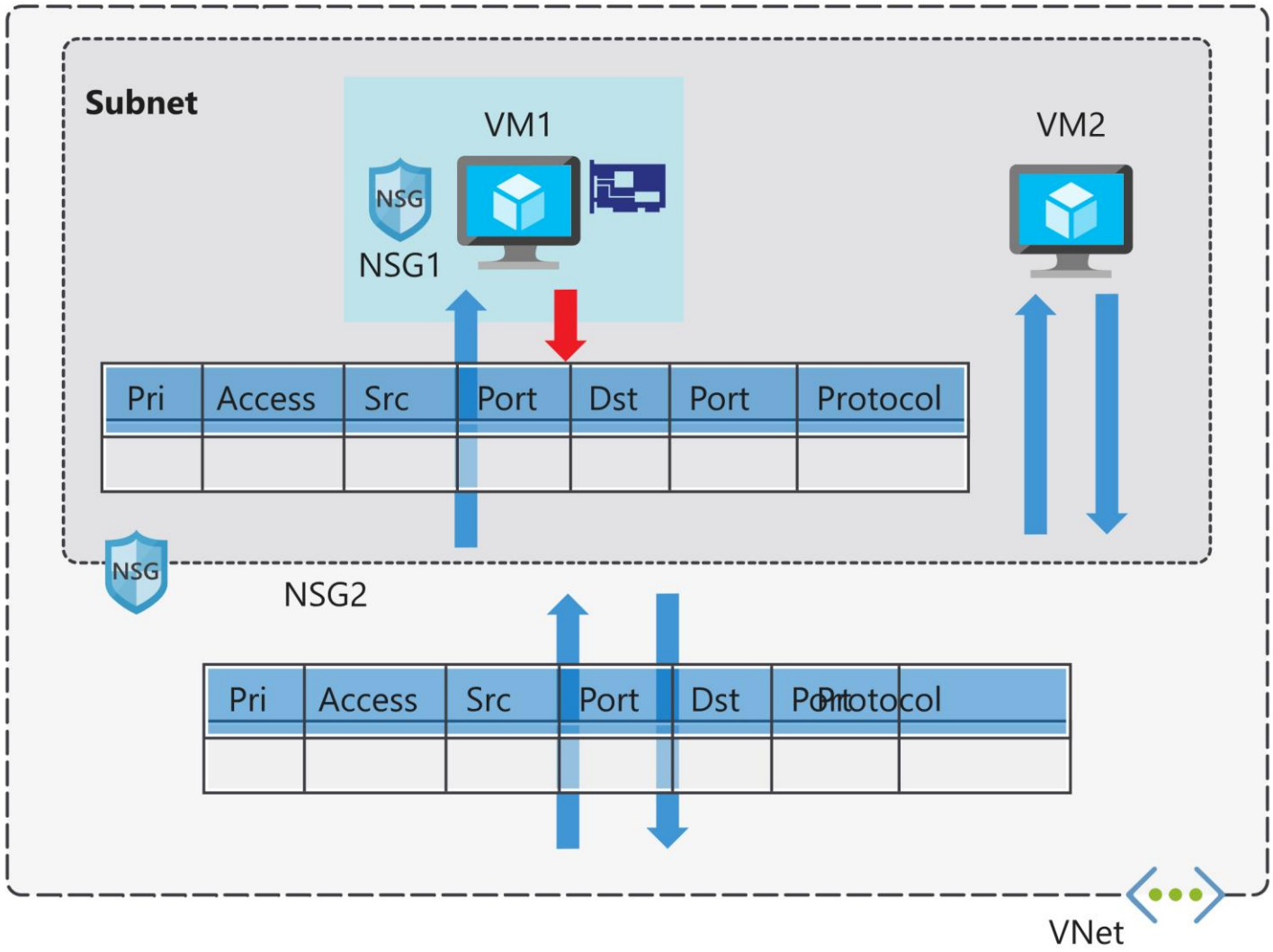
Network security groups

Network security groups filter network traffic to and from Azure resources. Network security groups contain security rules that you configure to allow or deny inbound and outbound traffic. You can use network security groups to filter traffic between virtual machines or subnets, both within a virtual network and from the internet.

Network security group assignment and evaluation

Network security groups are assigned to a network interface or a subnet. When you assign a network security group to a subnet, the rules apply to all network interfaces in that subnet. You can restrict traffic further by associating a network security group to the network interface of a virtual machine.

When you apply network security groups to both a subnet and a network interface, each network security group is evaluated independently. Inbound traffic is first evaluated by the network security group applied to the subnet, and then by the network security group applied to the network interface. Conversely, outbound traffic from a virtual machine is first evaluated by the network security group applied to the network interface, and then by the network security group applied to the subnet.



Applying a network security group to a subnet instead of individual network interfaces can reduce administration and management efforts. This approach also ensures that all virtual machines within the specified subnet are secured with the same set of rules.

Each subnet and network interface can have one network security group applied to it. Network security groups support TCP, UDP, and ICMP, and operate at Layer 4 of the OSI model.

In our manufacturing company scenario, network security groups can help you secure the network. You can control which computers can connect to your application servers. You configure the network security group so that only a specific range of IP addresses can connect to the servers. You can lock this down even more by only allowing access to or from specific ports, or from individual IP addresses. These rules can be applied to devices that are connecting remotely from on-premises networks, or between resources within Azure.

Security rules

A network security group contains one or more security rules. Configure security rules to either allow or deny traffic.

Rules have several properties:

Property	Explanation
Name	A unique name within the network security group.
Priority	A number between 100 and 4096.
Source or destination	Any, or an individual IP address, classless inter-domain routing (CIDR) block (10.0.0.0/24, for example), service tag, or application security group.
Protocol	TCP, UDP, or Any.

Property	Explanation
Direction	Whether the rule applies to inbound, or outbound traffic.
Port range	An individual port or range of ports.
Action	Allow or deny the traffic.

Network security group security rules are evaluated by priority, using the 5-tuple information (source, source port, dest and protocol) to allow or deny the traffic. When the conditions for a rule match the device configuration, rule processin

For example, suppose your company has created a security rule to allow inbound traffic on port 3389 (RDP) to your web 200 Then suppose that another administrator has created a rule to deny inbound traffic on port 3389, with a priority of precedence, because it's processed first. The rule with priority 150 is processed before the rule with priority 200.

With network security groups, the connections are stateful. Return traffic is automatically allowed for the same TCP/UD inbound rule allowing traffic on port 80 also allows the virtual machine to respond to the request (typically on an epher need a corresponding outbound rule.

With regard to the ERP system, the web servers for the ERP application connect to database servers that are in their ow security rules to state that the only allowed communication from the web servers to the database servers is port 1433 f communications. All other traffic to the database servers will be denied.

Default security rules

When you create a network security group, Azure creates several default rules. These default rules can't be changed, bu your own rules. These default rules allow connectivity within a virtual network and from Azure load balancers. They also communication to the internet, and deny inbound traffic from the internet.

The default rules for inbound traffic are:

Priority	Rule name	Description
65000	AllowVnetInbound	Allow inbound coming from any VM to any VM within the subnet.
65001	AllowAzureLoadBalancerInbound	Allow traffic from the default load balancer to any VM within the subnet.
65500	DenyAllInBound	Deny traffic from any external source to any of the VMs.

The default rules for outbound traffic are:

Priority	Rule name	Description
65000	AllowVnetOutbound	Allow outbound going from any VM to any VM within the subnet.
65001	AllowInternetOutbound	Allow outbound traffic going to the internet from any VM.
65500	DenyAllOutBound	Deny traffic from any internal VM to a system outside the virtual network.

Augmented security rules

You use augmented security rules for network security groups to simplify the management of large numbers of rules. A also help when you need to implement more complex network sets of rules. Augmented rules let you add the following security rule:

- multiple IP addresses
- multiple ports
- service tags
- application security groups

Suppose your company wants to restrict access to resources in your datacenter, spread across several network address rules, you can add all these ranges into a single rule, reducing the administrative overhead and complexity in your netw

Service tags

You use service tags to simplify network security group security even further. You can allow or deny traffic to a specific globally or per region.

Service tags simplify security for virtual machines and Azure virtual networks, by allowing you to restrict access by resou tags represent a group of IP addresses, and help simplify the configuration of your security rules. For resources that you tag, you don't need to know the IP address or port details.

You can restrict access to many services. Microsoft manages the service tags (you can't create your own). Some exampl

- **VirtualNetwork** This tag represents all virtual network addresses anywhere in Azure, and in your on-premises netwo hybrid connectivity.
- **AzureLoadBalancer** This tag denotes Azure's infrastructure load balancer. The tag translates to the virtual IP address (168.63.129.16) where Azure health probes originate.
- **Internet** This tag represents anything outside the virtual network address that is publicly reachable, including resou IP addresses. One such resource is the Web Apps feature of Azure App Service.
- **AzureTrafficManager** This tag represents the IP address for Azure Traffic Manager.
- **Storage** This tag represents the IP address space for Azure Storage. You can specify whether traffic is allowed or de specify if access is allowed only to a specific region, but you can't select individual storage accounts.
- **SQL** This tag represents the address for Azure SQL Database, Azure Database for MySQL, Azure Database for Postg Data Warehouse services. You can specify whether traffic is allowed or denied, and you can limit to a specific regio
- **AppService** This tag represents address prefixes for Azure App Service.

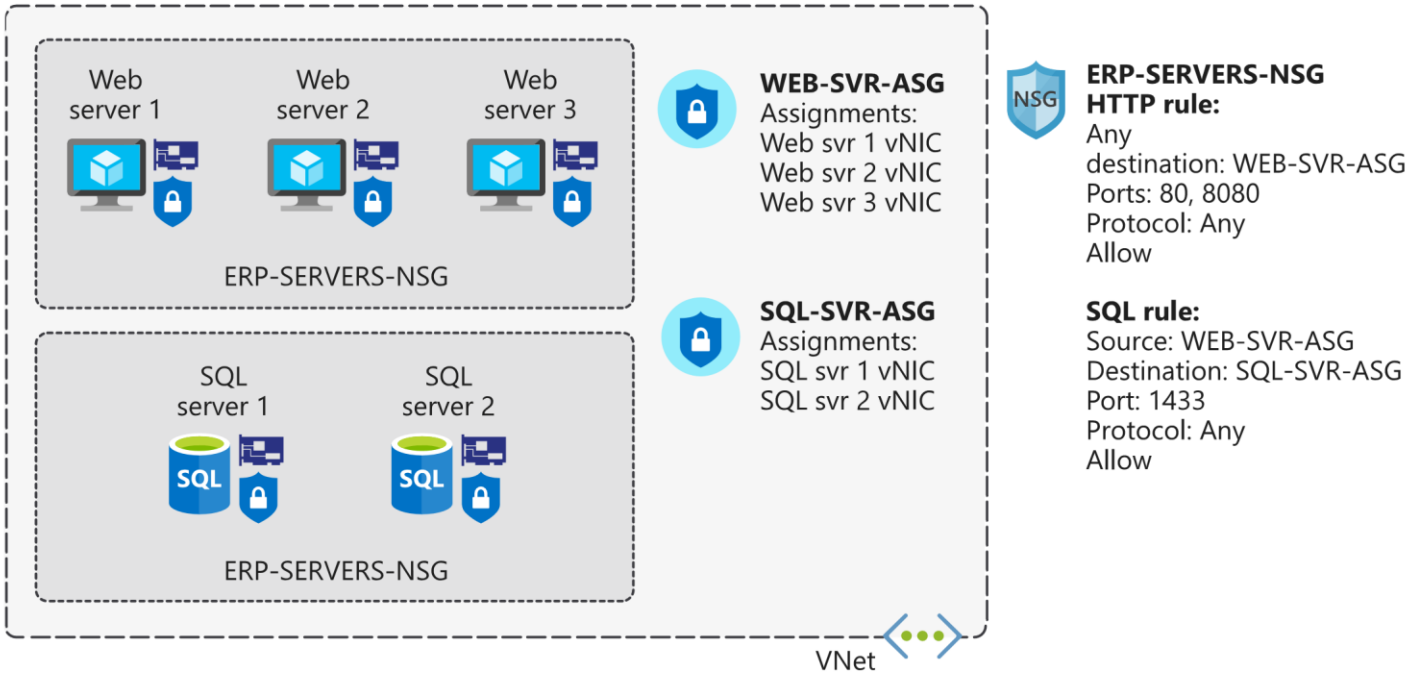
Application security groups

Application security groups let you configure network security for resources used by specific applications. You can grou logically, no matter what their IP address or subnet assignment.

Use application security groups within a network security group to apply a security rule to a group of resources. It's easi specific application workloads. You just add a new virtual machine deployment to one or more application security grou machine automatically picks up your security rules for that workload.

An application security group allows you to group network interfaces together. You can then use that application securi destination rule within a network security group.

For example, your company has a number of front-end servers in a virtual network. The web servers must be accessible Database servers must be accessible over port 1433. You assign the network interfaces for the web servers to one appli the network interfaces for the database servers to another application security group. You then create two inbound rul group. One rule allows HTTP traffic to all servers in the web server application security group. The other rule allows SQL database server application security group.



Without application security groups, you'd need to create a separate rule for each virtual machine.

The key benefit of application security groups is that it makes administration easier. You can easily add and remove network interfaces to an application security group as you deploy or redeploy application servers. You can also dynamically apply new rules to an application security group, which are then automatically applied to all the virtual machines in that application security group.

When to use network security groups

As a best practice, you should always use network security groups to help protect your networked assets against unwanted traffic. Network security groups give you granular control access over the network layer, without the potential complexity of setting security rules for every virtual machine or virtual network.

Next unit: Exercise - Create and manage network security groups

Continue >

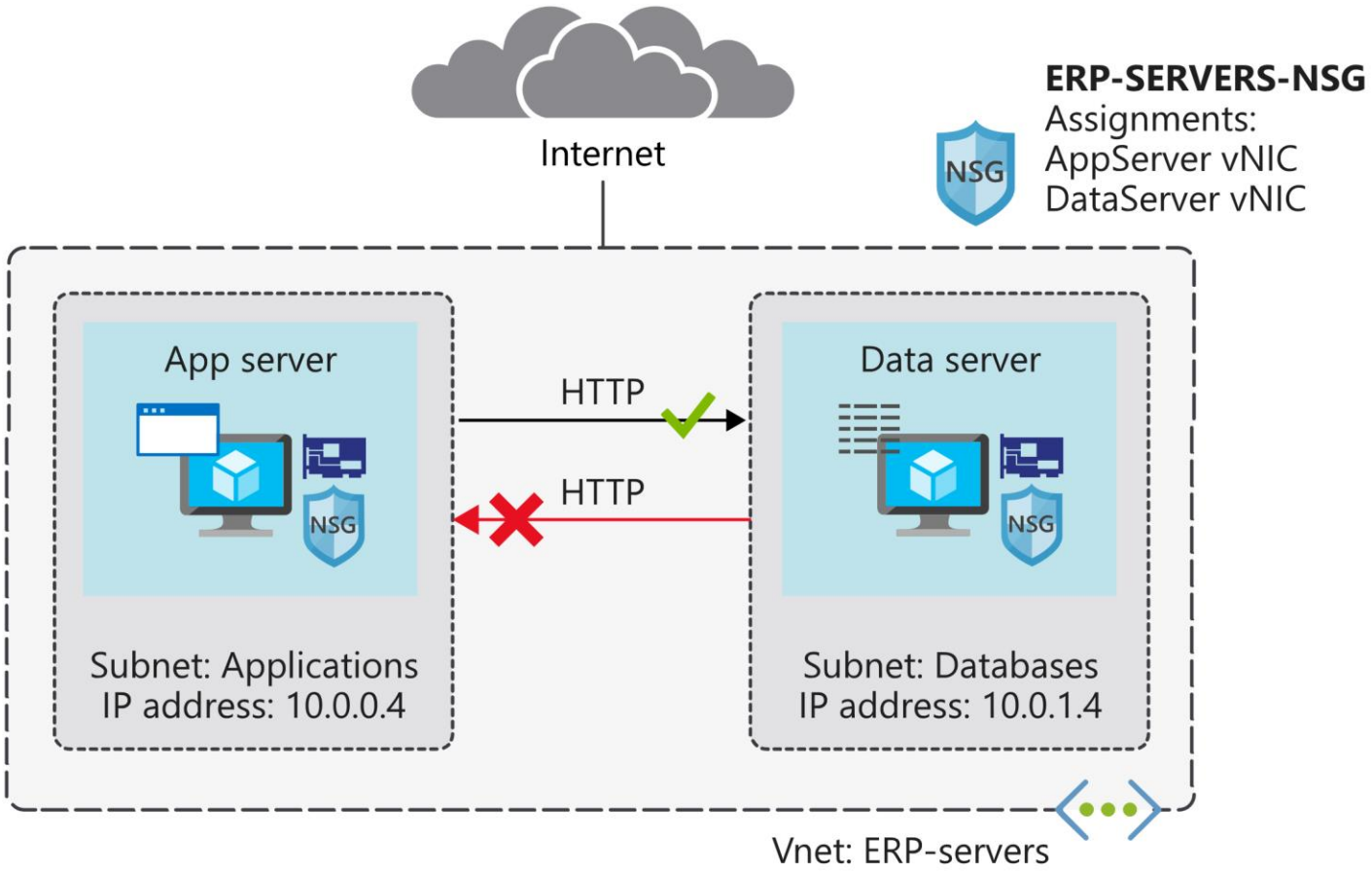
100 XP

Exercise - Create and manage network security groups

15 minutes

As the solution architect for the manufacturing company, you now want to start moving the ERP application and database servers to Azure. As a first step, you're going to test out your network security plan, using two of your servers.

In this unit, you'll configure a network security group and security rules to restrict network traffic to specific servers. You want your application server to be able to connect to your database server over HTTP. You don't want the database server to be able to use HTTP to connect to the application server.



Important

You need your own Azure subscription to run this exercise and you may incur charges. If you don't already have an Azure subscription, create a **free account** before you begin.

Create a virtual network and network security group

First, you'll create a resource group, the virtual network, and subnets for your server resources. You'll then create a network security group.

1. Open the Azure Cloud Shell in your browser, and log in to the directory with access to the subscription you want to create resources in. Use the Bash version of Cloud Shell.

2. Run the following command in the Cloud Shell to create a variable to store your resource group name, and a resource group for your resources. Replace `<resource group name>` with a name for your resource group, and `<location>` with the Azure region you'd like to deploy your resources in.

Azure CLI = Copy

```
rg=<resource group name>
 az group create --name $rg --location <location>
```

3. Run the following command in Azure Cloud Shell to create the **ERP-servers** virtual network and the **Applications** subnet.

```
Azure CLI
az network vnet create \     --resource-
group $rg \
    --name ERP-servers \
    --address-prefix 10.0.0.0/16 \
    --subnet-name Applications \
    --subnet-prefix 10.0.0.0/24
```

4. Run the following command in Cloud Shell to create the **Databases** subnet.

```
Azure CLI
az network vnet subnet create \     --resource-
group $rg \
    --vnet-name ERP-servers \
    --address-prefix 10.0.1.0/24 \
    --name Databases
```

5. Run the following command in Cloud Shell to create the **ERP-SERVERS-NSG** network security group.

```
Azure CLI
az network nsg create \     --
resource-group $rg \     --name
ERP-SERVERS-NSG
```

eate virtual machines running Ubuntu

Next, you create two virtual machines called **AppServer** and **DataServer**. You deploy **AppServer** to the **Applications** subnet, and **DataServer** to the **atabases** subnet. Add the virtual machine network interfaces to the **ERP-SERVERS-NSG** network security group. Then use these virtual ines to test your network security group.

1. Run the following command in Cloud Shell to build the **AppServer** virtual machine. Define a `<password>` for the admin account.

```
Azure CLI
wget -N https://raw.githubusercontent.com/MicrosoftDocs/mslearn-secure-and-isolate-with-nsg-and-
serviceendpoints/master/cloud-init.yml && \ az vm create \
    --resource-group $rg \
    --name AppServer \
    --vnet-name ERP-servers \
    --subnet Applications \
    --nsg ERP-SERVERS-NSG \
    --image UbuntuLTS \
    --size Standard_B1ls \
    --admin-username azureuser \
    --custom-data cloud-init.yml \
    --no-wait \
    --admin-password <password>
```

2. Run the following command in Cloud Shell to build the **DataServer** virtual machine. Define a `<password>` for the admin account.

```
Azure CLI
```

3.

```
az vm create \
    --resource-group $rg \
    --name DataServer \
    --vnet-name ERP-servers \
    --subnet Databases \
    --nsg ERP-SERVERS-NSG \
    --size Standard_B1ls \
    --image UbuntuLTS \
    --admin-username azureuser \
    --custom-data cloud-init.yml \
    --admin-password <password>
```

It can take several minutes for the virtual machines to be in a running state. To confirm that the virtual machines are running, run the following command in Cloud Shell.

Azure CLI = Copy

```
az vm list \
    --resource-group $rg \
    --show-details \
    --query "[*].{Name:name, Provisioned:provisioningState, Power:powerState}" \     --
output table
```

Ch

Now,
grou

When virtual machine creation is complete, you should see the following output.

1. output = Copy

```
Name        Provisioned    Power
----------  -------------  ----------
AppServer   Succeeded      VM running
DataServer  Succeeded      VM running
```

eck default connectivity

you'll try to open a Secure Shell (SSH) session to each of your virtual machines. Remember, so far you've deployed a network security
with default rules.

2. To connect to your virtual machines, use SSH directly from Cloud Shell. To do this, you need the public IP addresses that have been assigned to your virtual machines. Run the following command in Cloud Shell to list the IP addresses that you'll use to connect to the virtual machines.

Azure CLI = Copy

```
az vm list \
    --resource-group $rg \
    --show-details \
    --query "[*].{Name:name, PrivateIP:privateIps, PublicIP:publicIps}" \
    --output table
```

To make it easier to connect to your virtual machines during the rest of this exercise, assign the public IP addresses to variables. Run the following command in Cloud Shell to save the public IP address of **AppServer** and **DataServer** to a variable.

bash = Copy

```
APPSERVERIP="$(az vm list-ip-addresses \
                --resource-group $rg \
                --name AppServer \
                --query "[].virtualMachine.network.publicIpAddresses[*].ipAddress" \
                --output tsv)"

DATASERVERIP="$(az vm list-ip-addresses \
                --resource-group $rg \
                --name DataServer \
```

```
                --query "[].virtualMachine.network.publicIpAddresses[*].ipAddress" \
                --output tsv)"
```

3. Run the following command in Cloud Shell to check whether you can connect to your **AppServer** virtual machine.

```
bash
ssh azureuser@$APPSERVERIP -o ConnectTimeout=5
```

4. You'll get a `Connection timed out` message.

Run the following command in Cloud Shell to check whether you can connect to your **DataServer** virtual machine.

```
bash
ssh azureuser@$DATASERVERIP -o ConnectTimeout=5
```

You'll get the same connection failure message.

Inboun mber that the default rules deny all inbound traffic into a virtual network, unless this traffic is coming from another virtual network. The **Deny bound** rule blocked the inbound SSH connections you just attempted.

Name	Priority	Source IP	Destination IP	Access
Allow VNet Inbound	65000	VIRTUAL_NETWORK	VIRTUAL_NETWORK	Allow
y All Inbound	65500	*	*	Deny

eate a security rule for SSH

u've now experienced, the default rules in your **ERP-SERVERS-NSG** network security group include a **Deny All Inbound** rule. You'll now add so that you can use SSH to connect to **AppServer** and **DataServer**.

1. Run the following command in Cloud Shell to create a new inbound security rule to enable SSH access.

```
Azure CLI
az network nsg rule create \
    --resource-group $rg \
    --nsg-name ERP-SERVERS-NSG \
    --name AllowSSHRule \
    --direction Inbound \
    --priority 100 \
    --source-address-prefixes '*' \
    --source-port-ranges '*' \
    --destination-address-prefixes '*' \
    --destination-port-ranges 22 \
    --access Allow \     --
protocol Tcp \
    --description "Allow inbound SSH"
```

2. Run the following command in Cloud Shell to check whether you can now connect to your **AppServer** virtual machine.

```
bash
ssh azureuser@$APPSERVERIP -o ConnectTimeout=5
```

The network security group rule might take a minute or two to take effect. If you receive a connection failure message, try again.

3. You should now be able to connect. After the `Are you sure you want to continue connecting (yes/no)?` message, type `yes`.

4. Enter the password you used when you created the virtual machine.
5. Type `exit` to close the **AppServer** session.
6. Run the following command in Cloud Shell to check whether you can now connect to your **DataServer** virtual machine.

```bash
ssh azureuser@$DATASERVERIP -o ConnectTimeout=5
```

7. You should now be able to connect. After the `Are you sure you want to continue connecting (yes/no)?` message, type `yes`.
8. Enter the password you used when you created the virtual machine.
9. Type `exit` to close the **DataServer** session.

Create a security rule to prevent web access

Now add a rule so that **AppServer** can communicate with **DataServer** over HTTP, but **DataServer** can't communicate with **AppServer** over HTTP. These are the internal IP addresses for these servers:

Server name	IP address
AppServer	10.0.0.4
DataServer	10.0.1.4

1. Run the following command in Cloud Shell to create a new inbound security rule to deny HTTP access over port 80.

```azure-cli
az network nsg rule create \
    --resource-group $rg \
    --nsg-name ERP-SERVERS-NSG \
    --name httpRule \
    --direction Inbound \
    --priority 150 \
    --source-address-prefixes 10.0.1.4 \
    --source-port-ranges '*' \
    --destination-address-prefixes 10.0.0.4 \
    --destination-port-ranges 80 \
    --access Deny \
    --protocol Tcp \
    --description "Deny from DataServer to AppServer on port 80"
```

Test HTTP connectivity between virtual machines

Here, you check if your new rule works **AppServer** should be able to communicate with **DataServer** over HTTP. **DataServer** shouldn't be able to communicate with **AppServer** over HTTP.

1. Run the following command in Cloud Shell to connect to your **AppServer** virtual machine, and check if **AppServer** can communicate with **DataServer** over HTTP.

```bash
ssh -t azureuser@$APPSERVERIP 'wget http://10.0.1.4; exit; bash'
```

2. Enter the password you used when you created the virtual machine.
3. The response should include a `200 OK` message.

4. Run the following command in Cloud Shell to connect to your **DataServer** virtual machine, and check if **DataServer** can communicate with **AppServer** over HTTP.

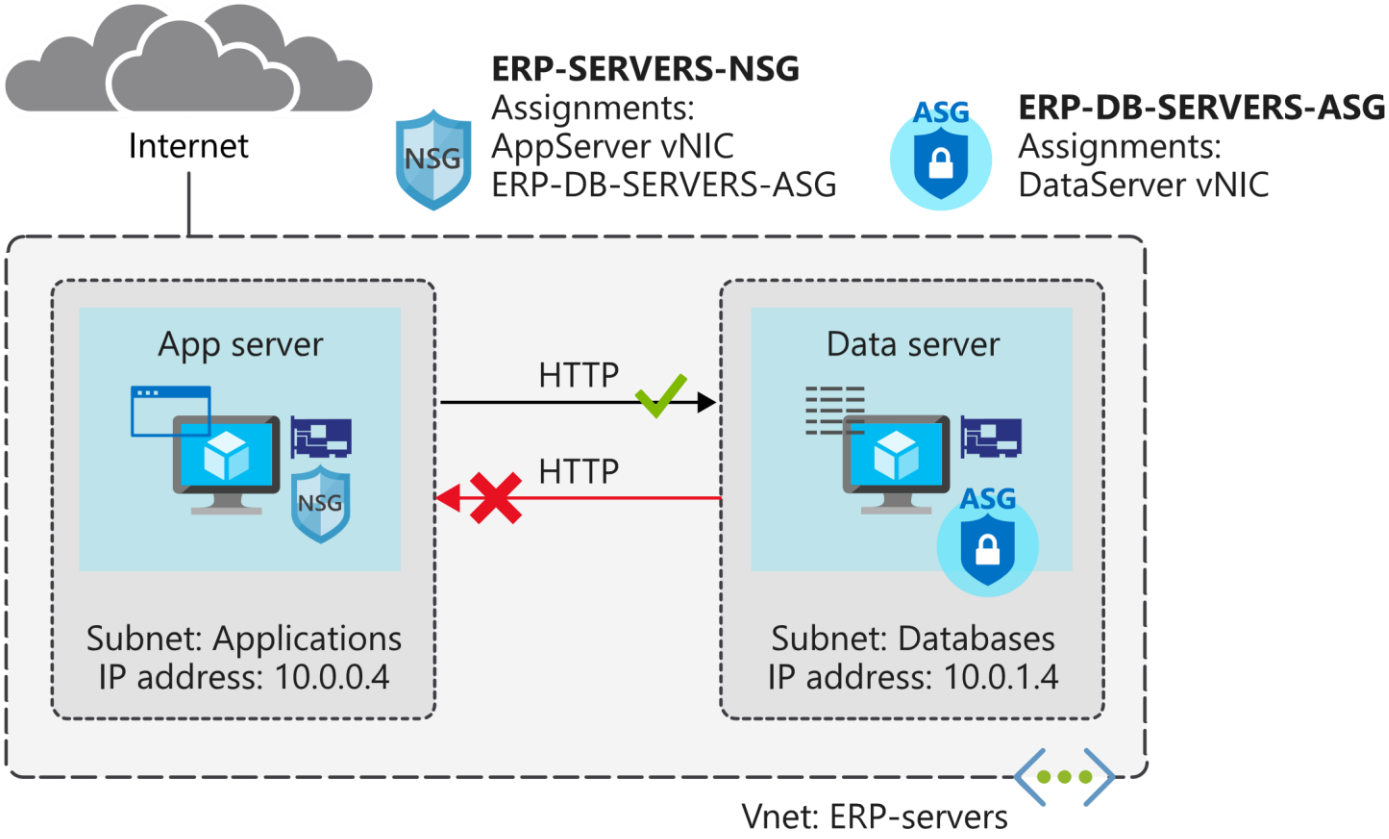
```bash
ssh -t azureuser@$DATASERVERIP 'wget http://10.0.0.4; exit; bash'
```

5. Enter the password you used when you created the virtual machine.

6. This shouldn't succeed, because you've blocked access over port 80. After several minutes, you should get a `Connection timed out` message. Press Ctrl+C to stop the command prior to the timeout.

Deploy an application security group

Next, create an application security group for database servers, so that all servers in this group can be assigned the same settings. You're planning to deploy more database servers, and want to prevent these servers from accessing application servers over HTTP. By assigning sources in the application security group, you don't need to manually maintain a list of IP addresses in the network security group. Instead, you assign the network interfaces of the virtual machines you want to manage to the application security group.



1. Run the following command in Cloud Shell to create a new application security group called **ERP-DB-SERVERS-ASG**.

```Azure CLI
az network asg create \     --resource-
group $rg \
    --name ERP-DB-SERVERS-ASG
```

2. Run the following command in Cloud Shell to associate **DataServer** with the application security group.

```Azure CLI
```

```
az network nic ip-config update \
    --resource-group $rg \
    --application-security-groups ERP-DB-SERVERS-ASG \
    --name ipconfigDataServer \
    --nic-name DataServerVMNic \
    --vnet-name ERP-servers \
    --subnet Databases
```

3. Run the following command in Cloud Shell to update the HTTP rule in the **ERP-SERVERS-NSG** network security group. It should reference the **ERP-DB-Servers** application security group.

Azure CLI = Copy

```
az network nsg rule update \
    --resource-group $rg \
    --nsg-name ERP-SERVERS-NSG \
    --name httpRule \
    --direction Inbound \
    --priority 150 \
    --source-address-prefixes "" \
    --source-port-ranges '*' \
    --source-asgs ERP-DB-SERVERS-ASG \     --destination-address-
prefixes 10.0.0.4 \
    --destination-port-ranges 80 \
    --access Deny \
    --protocol Tcp \
    --description "Deny from DataServer to AppServer on port 80 using application security group"
```

st the updated HTTP security rule

Run the following command in Cloud Shell to connect to your **AppServer** virtual machine, and check if **AppServer** can communicate with **DataServer** over HTTP.

1. bash = Copy

```
ssh -t azureuser@$APPSERVERIP 'wget http://10.0.1.4; exit; bash'
```

Enter the password you used when you created the virtual machine.

2. As before, the response should include a `200 OK` message. The application security group settings can take a minute or two to take effect. If you don't initially receive the `200 OK` message, wait a minute and try again.

3. Run the following command in Cloud Shell to connect to your **DataServer** virtual machine, and check if **DataServer** can communicate with **AppServer** over HTTP.

4. bash = Copy

```
ssh -t azureuser@$DATASERVERIP 'wget http://10.0.0.4; exit; bash'
```

Enter the password you used when you created the virtual machine.

5. As before, this shouldn't succeed, because you've blocked access over port 80. After several minutes, you should get a `Connection timed out` message. Press Ctrl+C to stop the command prior to the timeout.

6. 'e now confirmed that your network security group rule works using an application security group, in the same way as when you used a e IP address. If we were to add additional data servers, we can easily ensure they have the proper network security by adding the new rs to the **ERP-DB-SERVERS-ASG**.

---

t unit: Secure network access to PaaS services with virtual network service endpoints

ntinue T

100 XP

Secure network access to PaaS services with virtual network service endpoints

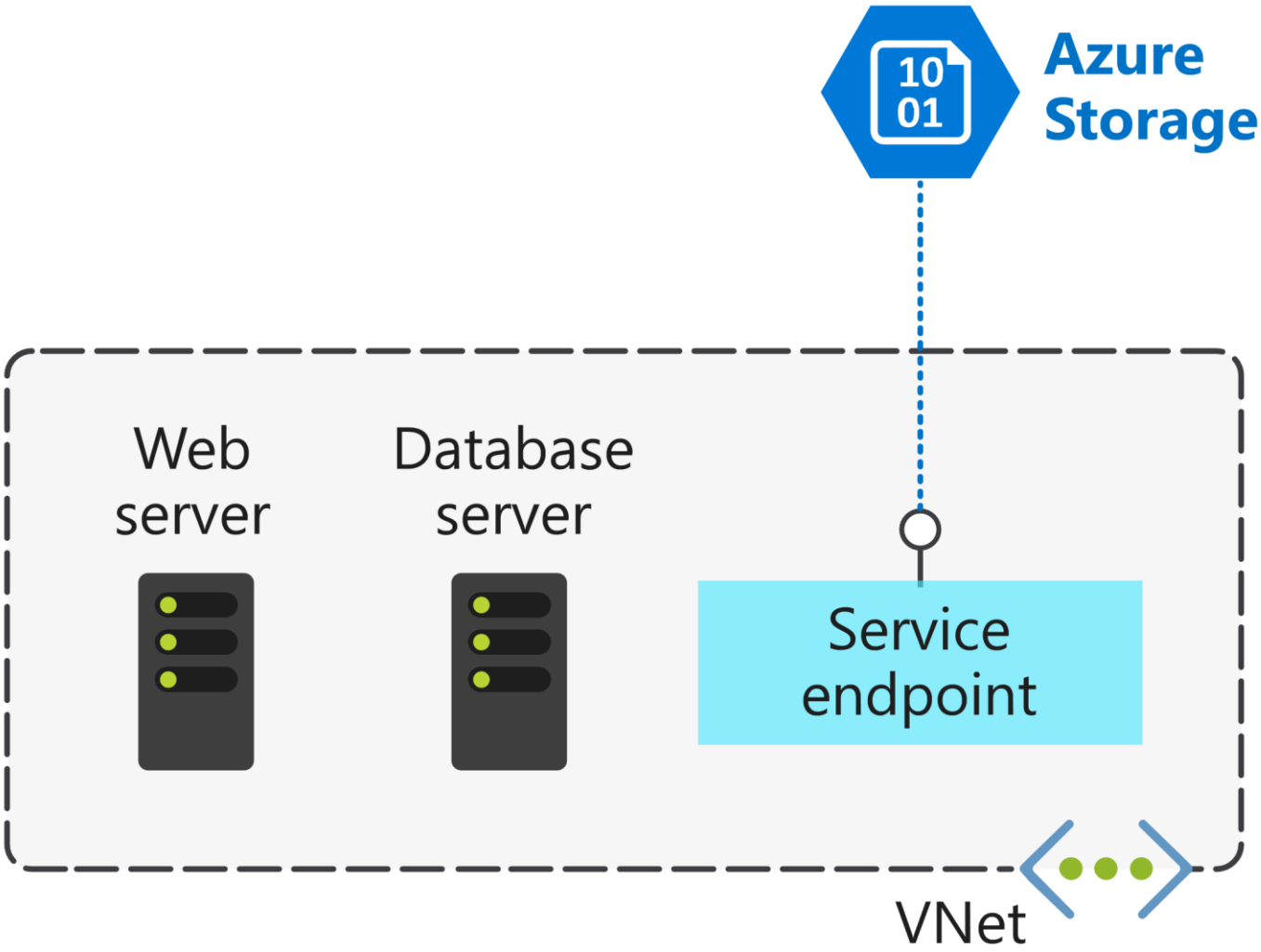
5 minutes

You've migrated your existing application and database servers for your ERP system to Azure as virtual machines. Now you're considering using some Azure platform as a service (PaaS) services to reduce your costs and administrative requirements. Storage services will hold certain large file assets, such as engineering diagrams. These engineering diagrams have proprietary information in them, and must remain secure from unauthorized access. These files must only be accessible from specific systems.

In this unit, you'll look at how to use virtual network service endpoints for securing supported Azure services.

Virtual network service endpoints

Use virtual network service endpoints to extend your private address space in Azure by providing a direct connection to your Azure services. Service endpoints let you secure your Azure resources to only your virtual network. Service traffic will remain on the Azure backbone, and doesn't go out to the internet.



By default, Azure services are all designed for direct internet access. All Azure resources have public IP addresses, including PaaS services such as Azure SQL Database and Azure Storage. Because these services are exposed to the internet, anyone can potentially access your Azure services.

Service endpoints can connect certain PaaS Services directly to your private address space in Azure, so they act like they're on the same virtual network. You use your private address space to access the PaaS services directly. Adding service endpoints doesn't remove the public endpoint. It simply provides a redirection of traffic.

Azure service endpoints are available for many services, such as:

- Azure Storage
- Azure SQL Database
- Azure Cosmos DB
- Azure Key Vault
- Azure Service Bus
- Azure Data Lake

For a service like SQL Database, which can't be accessed until you add IP addresses to its firewall, service endpoints should still be considered. Using a service endpoint for SQL Database restricts access to specific virtual networks, providing greater isolation and reducing the attack surface.

How service endpoints work

To enable a service endpoint, you must do two things:

1. Turn off public access to the service.
2. Add the service endpoint to a virtual network.

When you enable a service endpoint, you restrict the flow of traffic, and allow your Azure virtual machines to access the service directly from your private address space. Devices cannot access the service from a public network. On a deployed virtual machine vNIC, if you look at **Effective routes**, you'll notice the service endpoint as the **Next Hop Type**.

This is an example route table, before enabling a service endpoint:

SOURCE	STATE	ADDRESS PREFIXES	NEXT HOP TYPE
Default	Active	10.1.1.0/24	VNet
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	192.168.0.0/16	None

And here's an example route table after you've added two service endpoints to the virtual network:

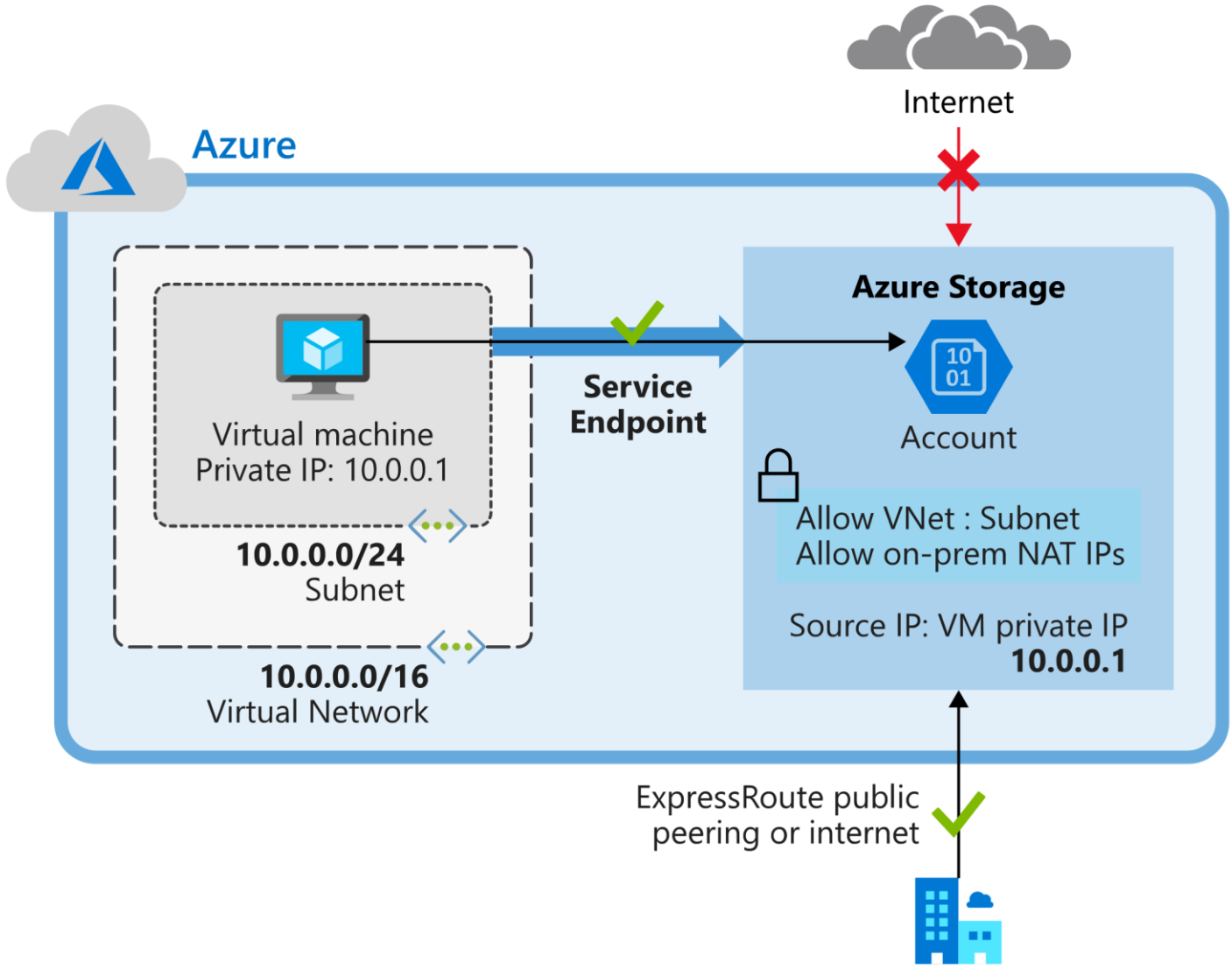
SOURCE	STATE	ADDRESS PREFIXES	NEXT HOP TYPE
Default	Active	10.1.1.0/24	VNet
Default	Active	0.0.0.0./0	Internet
Default	Active	10.0.0.0/8	None
Default	Active	100.64.0.0./10	None
Default	Active	192.168.0.0/16	None
Default	Active	20.38.106.0/23, 10 more	VirtualNetworkServiceEndpoint
Default	Active	20.150.2.0/23, 9 more	VirtualNetworkServiceEndpoint

All traffic for the service now is routed to the **VirtualNetworkServiceEndpoint**, and remains internal to Azure.

Service endpoints and hybrid networks

Service resources that you've secured by using virtual network service endpoints are not, by default, accessible from on-premises networks. To access resources from an on-premises network, use NAT IPs. If you use ExpressRoute for connectivity from on-premises to Azure, you have to identify the NAT IP addresses that are used by ExpressRoute. By default, each circuit uses two NAT IP addresses to connect to the Azure backbone network. You then need to add these IP addresses into the IP firewall configuration of the Azure service resource (for example, Azure Storage).

This diagram shows how you can use a service endpoint and firewall configuration to enable on-premises devices to access Azure Storage resources.



Next unit: Exercise - Restrict access to Azure Storage by using service endpoints

Continue >

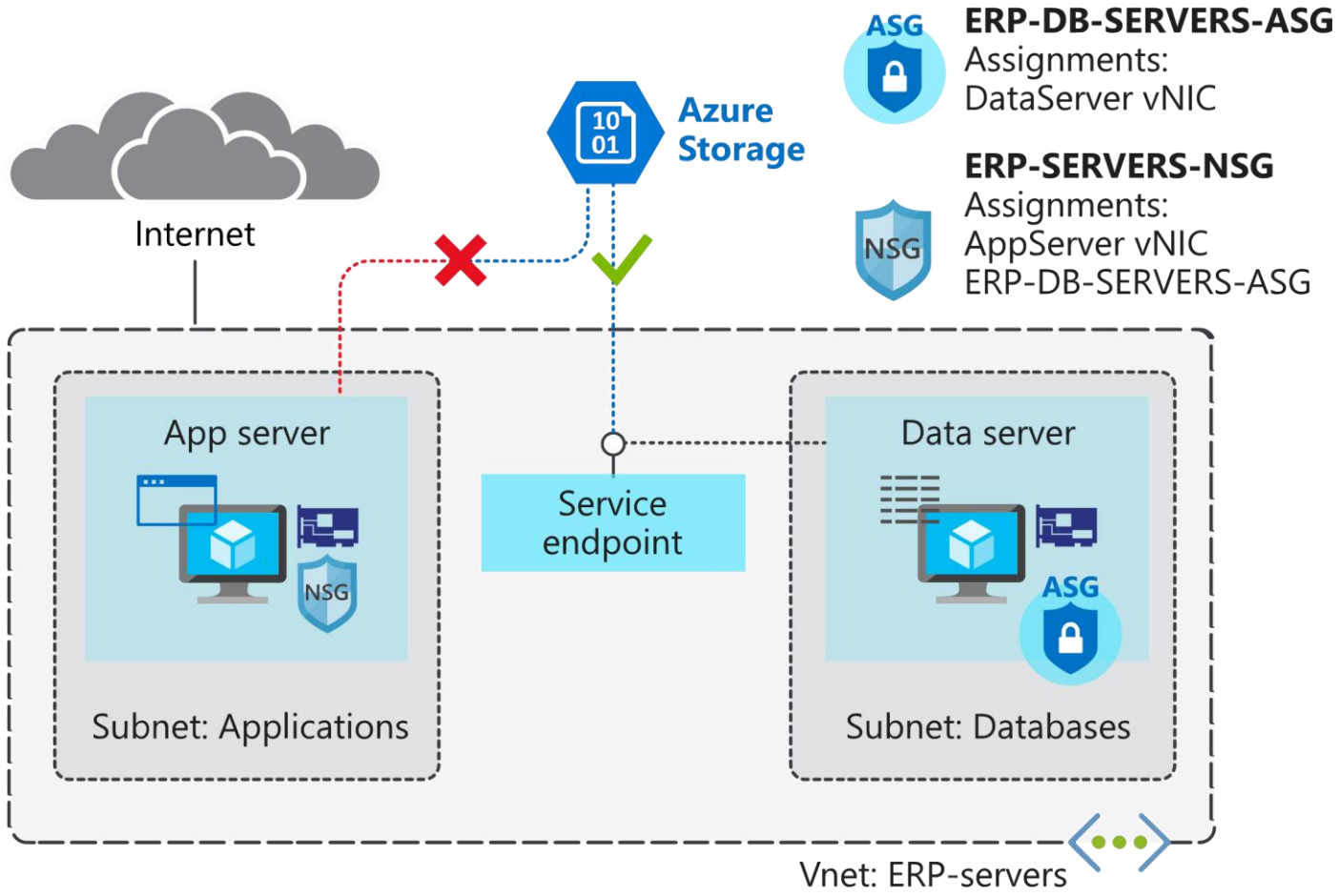
100 XP

Exercise - Restrict access to Azure Storage by using service endpoints

10 minutes

As the solution architect, you're planning to move sensitive engineering diagram files into Azure Storage. The files must only be accessible from computers inside the corporate network. You want to create a virtual network service endpoint for Azure Storage to secure the connectivity to your storage accounts.

In this unit, you'll create a service endpoint, and use network rules to restrict access to Azure Storage. You'll create a virtual network service endpoint for Azure Storage on the **Databases** subnet. You'll then verify that your **DataServer** virtual machine can access Azure Storage. Finally, you'll check that the **AppServer** virtual machine, which is on a different subnet, can't access storage.



Add rules to the network security group

Ensure that communications with Azure Storage pass through the service endpoint. Add outbound rules to allow access to the Storage service, but deny all other internet traffic.

1. Run the following command in Azure Cloud Shell to create an outbound rule to allow access to Storage.

Azure CLI | Copy

```
az network nsg rule create \
    --resource-group $rg \
    --nsg-name ERP-SERVERS-NSG \
    --name Allow_Storage \
```

```
    --priority 190 \
    --direction Outbound \
    --source-address-prefixes "VirtualNetwork" \
    --source-port-ranges '*' \
    --destination-address-prefixes "Storage" \
    --destination-port-ranges '*' \
    --access Allow \     --protocol
'*' \
    --description "Allow access to Azure Storage"
```

2. Run the following command in Cloud Shell to create an outbound rule to deny all internet access.

Azure CLI = Copy

```
az network nsg rule create \
    --resource-group $rg \
    --nsg-name ERP-SERVERS-NSG \
    --name Deny_Internet \     --
priority 200 \
    --direction Outbound \
    --source-address-prefixes "VirtualNetwork" \
    --source-port-ranges '*' \
    --destination-address-prefixes "Internet" \
    --destination-port-ranges '*' \
    --access Deny \
    --protocol '*' \
    --description "Deny access to Internet."
```

You should now have the following rules in ERP-SERVERS-NSG:

Rule name	Direction	Priority	Purpose
AllowSSHRule	Inbound	100	Allow inbound SSH
httpRule	Inbound	150	Deny from DataServer to AppServer on 80
Allow_Storage	Outbound	190	Allow access to Azure Storage
Deny_Internet	Outbound	200	Deny access to Internet from VNet

At this point, both **AppServer** and **DataServer** have access to the Azure Storage service.

Configure storage account and file share

In this step, you'll create a new storage account, and then add an Azure file share to this account. This share is where you store your engineering diagrams.

1. Run the following command in Cloud Shell to create a storage account for engineering documents.

bash = Copy

```
STORAGEACCT=$(az storage account create \
                --resource-group $rg \
                --name engineeringdocs$RANDOM \
                --sku Standard_LRS \
                --query "name" | tr -d '"')
```

2. Run the following command in Cloud Shell to store the primary key for your storage in a variable.

bash = Co

```
STORAGEKEY=$(az storage account keys list \
                --resource-group $rg \
                --account-name $STORAGEACCT \
                --query "[0].value" | tr -d '"')
```

3. Run the following command in Cloud Shell to create an Azure file share called **erp-data-share**.

Azure CLI = Copy

```
az storage share create \
    --account-name $STORAGEACCT \
    --account-key $STORAGEKEY \
    --name "erp-data-share"
```

Enable the service endpoint

You now need to configure the storage account to be accessible only from database servers, by assigning the storage endpoint to the **Databases** subnet. You then add a security rule to the storage account.

1. Run the following command in Cloud Shell to assign the **Microsoft.Storage** endpoint to the subnet.

Azure CLI = Copy

```
az network vnet subnet update \
    --vnet-name ERP-servers \
    --resource-group $rg \
    --name Databases \
    --service-endpoints Microsoft.Storage
```

2. Run the following command to deny all access to change the default action to `Deny`. After network access is denied, the storage account is not accessible from any network.

Azure CLI = Copy

```
az storage account update \     --resource-
group $rg \
    --name $STORAGEACCT \     --default-
action Deny
```

3. Run the following command in Cloud Shell to restrict access to the storage account. By default, storage accounts are open to accept all traffic. You want only traffic from the **Databases** subnet to be able to access the storage.

Azure CLI = Copy

```
az storage account network-rule add \
    --resource-group $rg \
    --account-name $STORAGEACCT \
    --vnet ERP-servers \
    --subnet Databases
```

Test access to storage resources

In this step, you'll connect to both of your servers, and verify that only **DataServer** has access to the Azure file share on the storage account.

1. Run the following command in Cloud Shell to save the public IP addresses of **AppServer** and **DataServer** to variables.

bash = Copy

```
APPSERVERIP="$(az vm list-ip-addresses \
                    --resource-group $rg \
                    --name AppServer \
                    --query "[].virtualMachine.network.publicIpAddresses[*].ipAddress" \
                    --output tsv)"

DATASERVERIP="$(az vm list-ip-addresses \
                    --resource-group $rg \
                    --name DataServer \
```

2.

```
                    --query "[].virtualMachine.network.publicIpAddresses[*].ipAddress" \
                    --output tsv)"
```

Run the following command in Cloud Shell to connect to your **AppServer** virtual machine, and attempt to mo the Azure file share.

```
bash
```

3.

```
ssh -t azureuser@$APPSERVERIP \     "mkdir azureshare; \     sudo mount -t cifs
//$STORAGEACCT.file.core.windows.net/erp-data-share azureshare \
    -o vers=3.0,username=$STORAGEACCT,password=$STORAGEKEY,dir_mode=0777,file_mode=0777,sec=ntlmssp; findmnt \     -
t cifs; exit; bash"
```

4. Enter the password you used when you created the virtual machine.

5. The response should include a `mount error` message. This connection isn't allowed, because there is no service endpoint for the storage account on the **Applications** subnet.

Run the following command in Cloud Shell to connect to your **DataServer** virtual machine, and attempt to mount the Azure file share.

```
bash
```

6.

```
ssh -t azureuser@$DATASERVERIP \
    "mkdir azureshare; \     sudo mount -t cifs //$STORAGEACCT.file.core.windows.net/erp-
data-share azureshare \
    -o vers=3.0,username=$STORAGEACCT,password=$STORAGEKEY,dir_mode=0777,file_mode=0777,sec=ntlmssp;findmnt \     -
t cifs; exit; bash"
```

7. Enter the password you used when you created the virtual machine.

that endpoint. The mount should be successful, and the response should include details of the mount point. This is allowed because you created the service endpoint for the storage account on the **Databases** subnet.

ve now verified that **DataServer** can access storage, by using the storage service endpoint on the **Database** subnet. You've also verified

AppServer can't access storage. This is because this server is on a different subnet, and doesn't have access t the virtual network service

t unit: Summary

ntinue

100 XP

Summary

2 minutes

You've learned about isolating and securing network resources in Azure. You now know how to use network security groups to secure virtua networks and virtual machines by creating rules to control network traffic. You've also learned how to use network service endpoints to control traffic to services such as Azure Storage and Azure SQL Database.

You can now use network security groups and service endpoints to ensure that network access to your Azure virtual machines and services is properly secured.

Clean up

The sandbox automatically cleans up your resources when you're finished with this module.

When you're working in your own subscription, it's a good idea at the end of a project to identify whether you still need the resources you created. Resources left running can cost you money. You can delete resources individually or delete the resource group to delete the entire set of resources.

Learn more

For more information on network security groups, see the security groups overview. For more information on virtual network service endpoints, see the virtual network service endpoint overview.

Module complete:

Unlock achievement

Introduction

2 minutes

Imagine you're the solution architect for an engineering company that has been migrating services into Azure. The company has deployed services into separate virtual networks. It hasn't configured private connectivity between the virtual networks.

Several business units have identified services in these virtual networks that need to communicate with each other. You need to enable this connectivity, but you don't want to expose these services to the internet. You also want to keep the integration as simple as possible.

In this module, you'll learn about virtual network connection options and why virtual network peering is suited to this scenario. You'll create virtual networks and configure virtual network peering between them. You'll then test your configuration to make sure it meets your connect goals.

Learning objectives

In this module, you'll:

- Identify use cases for virtual network peering.
- Identify the features and limitations of virtual network peering.
- Configure peering connections between virtual networks.

Next unit: Connect services by using virtual network peering

Continue

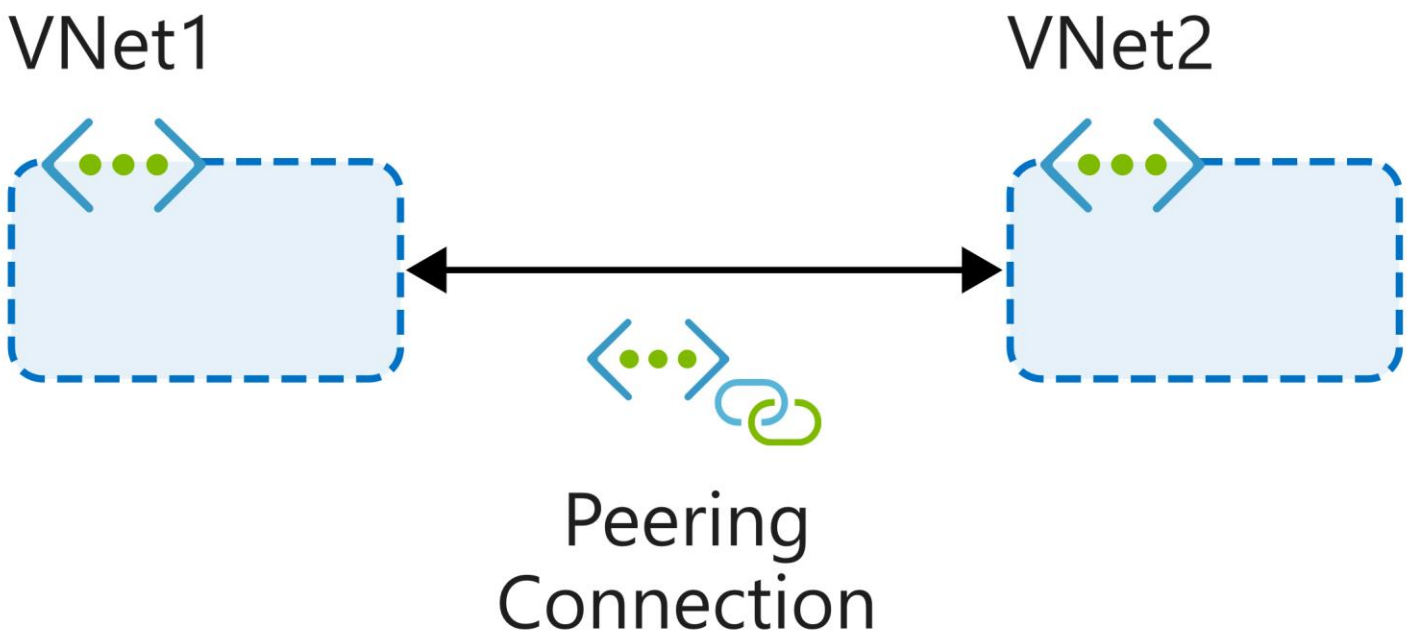
100 XP

Connect services by using virtual network peering

8 minutes

You can use virtual network peering to directly connect Azure virtual networks. When you use peering to connect virtual networks, virtual machines (VMs) in these networks can communicate with each other as if they were in the same network.

In peered virtual networks, traffic between virtual machines is routed through the Azure network. The traffic uses only private IP addresses. It doesn't rely on internet connectivity, gateways, or encrypted connections. The traffic is always private, and it takes advantage of the high bandwidth and low latency of the Azure backbone network.



The two types of peering connections are created in the same way:

- **Virtual network peering** connects virtual networks in the same Azure region, such as two virtual networks in North Europe.
- **Global virtual network peering** connects virtual networks that are in different Azure regions, such as a virtual network in North Europe and a virtual network in West Europe.

Virtual network peering doesn't affect or disrupt any resources that you've already deployed to the virtual networks. But when you use virtual network peering, consider the key features that the following sections define.

Reciprocal connections

When you create a virtual network peering connection in only one virtual network to connect to a peer in another network, you're not connecting the networks together. To connect the networks by using virtual network peering, you have to create connections in each virtual network.

Think of how you connect two network switches together. You connect a cable to each switch and maybe configure some settings so that the switches can communicate. Virtual network peering requires similar connections in each virtual network. Reciprocal connections provide this functionality.

Cross-subscription virtual network peering

You can use virtual network peering even when both virtual networks are in different subscriptions. This might be necessary for mergers and acquisitions or to connect virtual networks in subscriptions that different departments manage. Virtual networks can be in different subscriptions, and the subscriptions can use the same or different Azure Active Directory tenants.

When you use virtual network peering across subscriptions, you might find that an administrator of one subscription doesn't administer the peer network's subscription. The administrator might not be able to configure both ends of the connection. To peer the virtual networks when both subscriptions are in different Azure Active Directory tenants, the administrators of each subscription must grant the peer subscription's administrator the `Network Contributor` role on their virtual network.

Transitivity

Virtual network peering is nontransitive. Only virtual networks that are directly peered can communicate with each other. The virtual networks can't communicate with the peers of their peers.

Suppose, for example, that your three virtual networks (A, B, C) are peered like this: A <-> B <-> C. Resources in A can't communicate with resources in C because that traffic can't transit through virtual network B. If you need communication between virtual network A and virtual network C, you must explicitly peer these two virtual networks.

Gateway transit

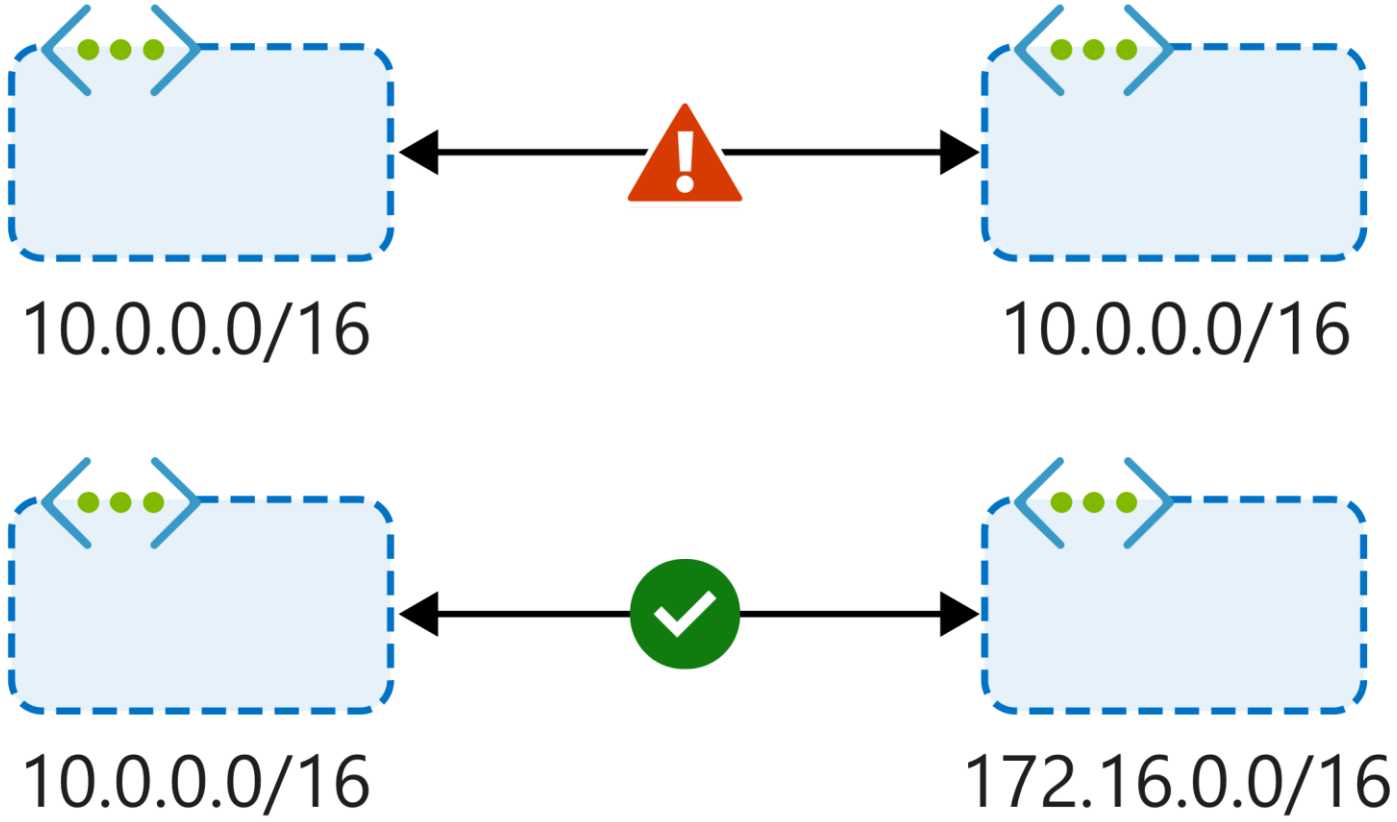
You can configure transitive connections on-premises if you use virtual network gateways as transit points. Using gateway transit, you can enable on-premises connectivity without deploying virtual network gateways to all your virtual networks. This method might reduce cost and complexity. By using gateway peering, you can configure a single virtual network as a hub network. Connect this hub network to your on-premises datacenter and share its virtual network gateway with peers.

To enable gateway transit, configure the **Allow gateway transit** option in the hub virtual network where you deployed the gateway connection to your on-premises network. Also configure the **Use remote gateways** option in any spoke virtual networks.

> **Note**
>
> If you want to enable the **Use remote gateways** option in a spoke network peering, you can't deploy a virtual network gateway in the spoke virtual network. Additionally, gateway transit currently isn't supported with global virtual network peering.

Overlapping address spaces

IP address spaces of connected networks within Azure and between Azure and your on-premises system can't overlap. This is also true for peered virtual networks. Keep this rule in mind when you're planning your network design. In any networks you connect through virtual network peering, VPN, or ExpressRoute, assign different address spaces that don't overlap.



Alternative connectivity methods

Virtual network peering is the least complex way to connect virtual networks. Other methods focus primarily on connectivity between on-premises and Azure networks rather than connections between virtual networks.

You can also connect virtual networks together through the ExpressRoute circuit. ExpressRoute is a dedicated, private connection between an on-premises datacenter and the Azure backbone network. The virtual networks that connect to an ExpressRoute circuit are part of the same routing domain and can communicate with each other. ExpressRoute connections don't go over the public internet, so your communications with Azure services are as secure as possible.

VPNs use the internet to connect your on-premises datacenter to the Azure backbone through an encrypted tunnel. You can use a site-to-site configuration to connect virtual networks together through VPN gateways. VPN gateways have higher latency than virtual network peering setups. They're more complex to manage, and they can cost more.

When virtual networks are connected through both a gateway and virtual network peering, traffic flows through the peering configuration.

When to choose virtual network peering

Virtual network peering can be a great way to enable network connectivity between services that are in different virtual networks. Because it's easy to implement and deploy, and it works well across regions and subscriptions, virtual network peering should be your first choice when you need to integrate Azure virtual networks.

Peering might not be your best option if you have existing VPN or ExpressRoute connections or services behind Azure Basic Load Balancers that would be accessed from a peered virtual network. In these cases, you should research alternatives.

Next unit: Exercise - Prepare virtual networks for peering by using Azure CLI commands

Continue >

R Previous Unit 3 of 6 S Next T

100 XP

Exercise - Prepare virtual networks for peering by using Azure CLI commands

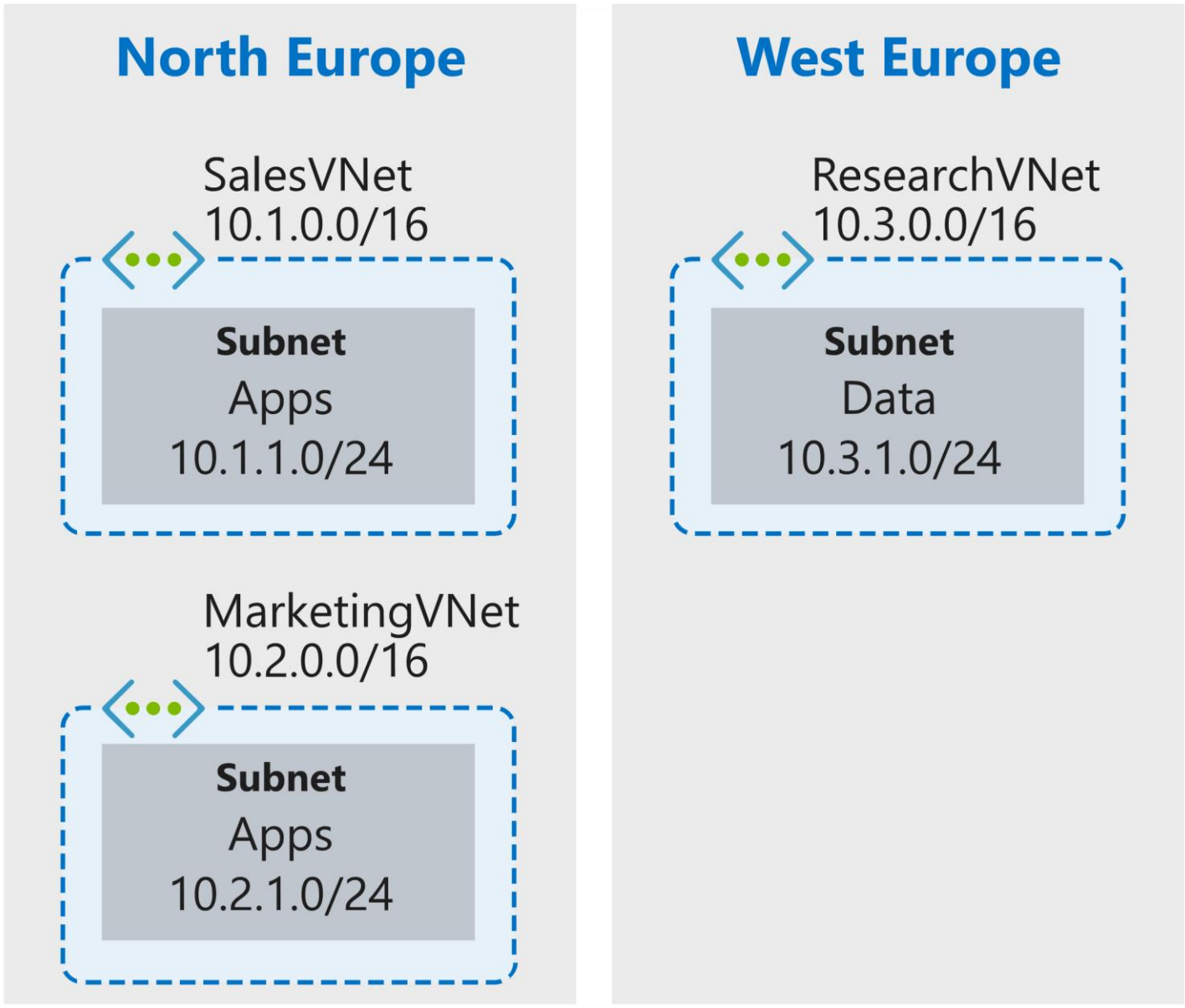
10 minutes

This module requires a sandbox to complete. A **sandbox** gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

Activate sandbox

Let's say your company is now ready to implement virtual network peering. You want to connect systems that are deployed in different virtual networks. To test out this plan, you'll start by creating virtual networks to support the services your company is already running in Azure. You need three virtual networks:

- The **Sales** virtual network is deployed in **North Europe**. Sales systems use this virtual network to process data that's added after a customer is engaged. The Sales team wants access to Marketing data.
- The **Marketing** virtual network is deployed in **North Europe**. Marketing systems use this virtual network. Members of the Marketing team regularly chat with the Sales team. To share their data with the Sales team, they must download it because the Sales and Marketing systems aren't connected.
- The **Research** virtual network is deployed in **West Europe**. Research systems use this virtual network. Members of the Research team have a logical working relationship with Marketing, but they don't want the Sales team to have direct access to their data.



You'll create the following resources:

Virtual network	Region	Virtual network address space	Subnet	Subnet address space
SalesVNet	North Europe	10.1.0.0/16	Apps	10.1.1.0/24
MarketingVNet	North Europe	10.2.0.0/16	Apps	10.2.1.0/24
ResearchVNet	West Europe	10.3.0.0/16	Data	10.3.1.0/24

Create the virtual networks

1. In Cloud Shell, run the following command to create the virtual network and subnet for the **Sales** systems.

Azure CLI Copy

```
az network vnet create \
    --resource-group [sandbox resource group name] \
    --name SalesVNet \
    --address-prefix 10.1.0.0/16 \
    --subnet-name Apps \
    --subnet-prefix 10.1.1.0/24 \
    --location northeurope
```

2. Run the following command to create the virtual network and subnet for the **Marketing** systems.

Azure CLI = Copy

```
az network vnet create \
    --resource-group [sandbox resource group name] \
    --name MarketingVNet \
    --address-prefix 10.2.0.0/16 \
    --subnet-name Apps \
    --subnet-prefix 10.2.1.0/24 \
    --location northeurope
```

3. Run the following command to create the virtual network and subnet for the **Research** systems.

Azure CLI = Copy

```
az network vnet create \
    --resource-group [sandbox resource group name] \
    --name ResearchVNet \
    --address-prefix 10.3.0.0/16 \
    --subnet-name Data \
    --subnet-prefix 10.3.1.0/24 \
    --location westeurope
```

Confirm the virtual network configuration

Let's take a quick look at what you created.

1. In Cloud Shell, run the following command to view the virtual networks.

Azure CLI = Copy

```
az network vnet list --output table
```

2. You should see an output like this:

output = Copy

```
Name           ResourceGroup                              Location    NumSubnets   Prefixes    DnsServers   DDOSProtection
VMProtection
-------------- ------------------------------------------ ----------- ------------ ----------- ------------ ---------------
- --------------
MarketingVNet [sandbox resource group name] northeurope 1            10.2.0.0/16              False            False
SalesVNet     [sandbox resource group name] northeurope 1            10.1.0.0/16              False            False
ResearchVNet  [sandbox resource group name] westeurope  1            10.3.0.0/16              False            False
```

Create virtual machines in each virtual network

Now you'll deploy some Ubuntu virtual machines (VMs) in each of the virtual networks. These VMs simulate the services in each virtual network.

In the final unit of this module, you'll use these VMs to test connectivity between the virtual networks.

1. In Cloud Shell, run the following command to create an Ubuntu VM in the **Apps** subnet of **SalesVNet**. In the command, replace `<password>` with a password that meets the requirements for Linux VMs. Note this password for later use.

Azure CLI = Copy

```
az vm create \
    --resource-group [sandbox resource group name] \
    --no-wait \
    --name SalesVM \
    --location northeurope \
    --vnet-name SalesVNet \
    --subnet Apps \
    --image UbuntuLTS \
    --admin-username azureuser \
    --admin-password <password>
```

2.

> **Note**
>
> The `--no-wait` parameter in this command lets you continue working in Cloud Shell while the VM is building.

Run the following command to create another Ubuntu VM in the **Apps** subnet of **MarketingVNet**. Replace `<password>` with a password that meets the requirements for Linux VMs. Note this password for later use.

Azure CLI = Copy

3.

```
az vm create \
    --resource-group [sandbox resource group name] \
    --no-wait \
    --name MarketingVM \
    --location northeurope \
    --vnet-name MarketingVNet \
    --subnet Apps \
    --image UbuntuLTS \
    --admin-username azureuser \
    --admin-password <password>
```

Run the following command to create an Ubuntu VM in the **Data** subnet of **ResearchVNet**. Replace `<password>` with a password that meets the requirements for Linux VMs. Note this password for later use.

Azure CLI = Copy

4.

```
az vm create \
    --resource-group [sandbox resource group name] \
    --no-wait \
    --name ResearchVM \
    --location westeurope \
    --vnet-name ResearchVNet \
    --subnet Data \
    --image UbuntuLTS \
    --admin-username azureuser \
    --admin-password <password>
```

The VMs might take several minutes to reach a running state.

To confirm that the VMs are running, run the following command. This uses the Linux `watch` command which will refresh every five seconds.

bash = Copy

```
watch -d -n 5 "az vm list \
    --resource-group [sandbox resource group name] \
    --show-details \
    --query '[*].{Name:name, ProvisioningState:provisioningState, PowerState:powerState}' \     --
output table"
```

A **ProvisioningState** of **Succeeded** and a **PowerState** of **VM running** indicates a successful deployment. When your VMs are running, you're ready to move on. Press `Ctrl-c` to stop the command and continue on with the exercise.

t unit: Exercise - Configure virtual network peering connections by using Azure CLI commands

ıtinue T

Azure Cloud Shell

This module requires a sandbox to complete. A sandbox gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

R Previous Unit 4 of 6 S Next T

100 XP

Exercise - Configure virtual network peering connections by using Azure CLI commands

10 minutes

This module requires a sandbox to complete. A **sandbox** gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

Activate sandbox

Now you have created virtual networks and have run virtual machines (VMs) within them. But the virtual networks have no connectivity, and none of these systems can communicate with each other.

To enable communication, you need to create peering connections for the virtual networks. To satisfy your company's requirements, configure a hub and spoke topology, and permit virtual network access when you create the peering connections.

Create virtual network peering connections

Follow these steps to create connections between the virtual networks and to configure the behavior of each connection.

1. In Cloud Shell, run the following command to create the peering connection between the **SalesVNet** and **MarketingVNet** virtual networks.
   This command also permits virtual network access across this peering connection.

   Azure CLI = Copy

   ```
   az network vnet peering create \
     --name SalesVNet-To-MarketingVNet \
     --remote-vnet MarketingVNet \
     --resource-group [sandbox resource group name] \
     --vnet-name SalesVNet \
     --allow-vnet-access
   ```

2. Run the following command to create a reciprocal connection from **MarketingVNet** to **SalesVNet**. This step completes the connection between these virtual networks.

   Azure CLI = Copy

   ```
   az network vnet peering create \
     --name MarketingVNet-To-SalesVNet \
     --remote-vnet SalesVNet \
     --resource-group [sandbox resource group name] \
     --vnet-name MarketingVNet \
     --allow-vnet-access
   ```

Now that you have connections between Sales and Marketing, create connections between Marketing and Research.

1. In Cloud Shell, run the following command to create the peering connection between the **MarketingVNet** and **ResearchVNet** virtual networks.

   Azure CLI = Copy

```
az network vnet peering create \
  --name MarketingVNet-To-ResearchVNet \
  --remote-vnet ResearchVNet \
  --resource-group [sandbox resource group name] \
```

```
  --vnet-name MarketingVNet \
  --allow-vnet-access
```

2. Run the following command to create the reciprocal connection between **ResearchVNet** and **MarketingVNet**.

```
Azure CLI
```

```
az network vnet peering create \
  --name ResearchVNet-To-MarketingVNet \
  --remote-vnet MarketingVNet \    --resource-group [sandbox resource
group name] \
  --vnet-name ResearchVNet \
  --allow-vnet-access
```

eck the virtual network peering connections

that you've created the peering connections between the virtual networks, make sure the connections work.

In Cloud Shell, run the following command to check the connection between **SalesVNet** and **MarketingVNet**.

1.

```
Azure CLI
```

```
az network vnet peering list \
  --resource-group [sandbox resource group name] \
  --vnet-name SalesVNet \
  --output table
```

You've created only one connection from **SalesVNet**, so you see only one result. In the **PeeringState** column, make sure the status is **Connected**.

2. Run the following command to check the peering connection between the **ResearchVNet** and **MarketingVNet** virtual networks.

3.

```
Azure CLI
```

```
az network vnet peering list \
  --resource-group [sandbox resource group name] \
  --vnet-name ResearchVNet \
  --output table
```

Again, you've created only one connection from **ResearchVNet**, so you see only one result. In the **PeeringState** column, make sure the status is **Connected**.

4. Run the following command to check the peering connections for the **MarketingVNet** virtual network.

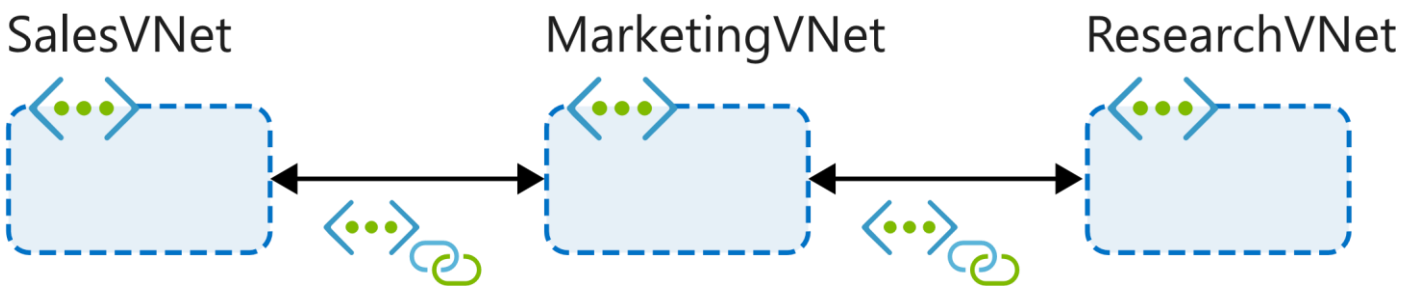
5.

```
Azure CLI
```

```
az network vnet peering list \
  --resource-group [sandbox resource group name] \
  --vnet-name MarketingVNet \
  --output table
```

Remember that you created connections from Marketing to Sales and from Marketing to Research, so you should see two connections. In the **PeeringState** column, make sure the status of both connections is **Connected**.

peering connections between the virtual networks should now look like this:



Check effective routes

You can further check the peering connection by looking at the routes that apply to the network interfaces of the VMs.

1. Run the following command to look at the routes that apply to the **SalesVM** network interface.

```
Azure CLI                                                    = Copy

az network nic show-effective-route-table \
  --resource-group [sandbox resource group name] \
  --name SalesVMVMNic \
  --output table
```

The output table shows the effective routes for the VM's network interface. For **SalesVMVMNic**, you should see a route to **10.2.0.0/16** with a next hop type of **VNetPeering**. This is the network route for the peering connection from **SalesVNet** to **MarketingVNet**.

```
output                                                       = Copy

Source    State   Address Prefix   Next Hop Type    Next Hop IP
--------  ------- ---------------- ---------------- -------------
Default   Active  10.1.0.0/16      VnetLocal
Default   Active  10.2.0.0/16      VNetPeering
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  192.168.0.0/16   None
```

2. Look at the routes for **MarketingVM**.

```
Azure CLI                                                    = Copy

az network nic show-effective-route-table \
  --resource-group [sandbox resource group name] \
  --name MarketingVMVMNic \
  --output table
```

The output table shows the effective routes for the VM's network interface. For **MarketingVMVMNic**, you should see a route to **10.1.0.0/16** and **10.3.0.0/16** with a next hop type of **VNetGlobalPeering**. This is the network route for the peering connection from **MarketingVNet** to **SalesVNet** and from **SalesVNet** to **ResearchVNet**.

```
output                                                       = Copy

Source    State   Address Prefix   Next Hop Type     Next Hop IP
--------  ------- ---------------- ----------------- -------------
Default   Active  10.2.0.0/16      VnetLocal
Default   Active  10.1.0.0/16      VNetPeering
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  192.168.0.0/16   None
Default   Active  10.3.0.0/16      VNetGlobalPeering
```

3. Look at the routes for **ResearchVM**.

Azure CLI = Copy

```
az network nic show-effective-route-table \
  --resource-group [sandbox resource group name] \
  --name ResearchVMVMNic \
  --output table
```

The output table shows the effective routes for the VM's network interface. For **ResearchVMVMNic**, you should see a route to **10.2.0.0/16** with a next hop type of **VNetPeering**. This is the network route for the peering connection from **ResearchVNet** to **MarketingVNet**.

output = Copy

```
Source    State    Address Prefix    Next Hop Type     Next Hop IP
--------  -------  ----------------  ----------------  -------------
Default   Active   10.3.0.0/16       VnetLocal
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   192.168.0.0/16    None
Default   Active   10.2.0.0/16       VNetGlobalPeering
```

that your peering connections are configured, let's take a look at how this affects the communication between VMs.

t unit: Exercise - Verify virtual network peering by using SSH between Azure virtual machines

ıtinue T

English (United States)

Azure Cloud Shell

This module requires a sandbox to complete. A sandbox gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

100 XP

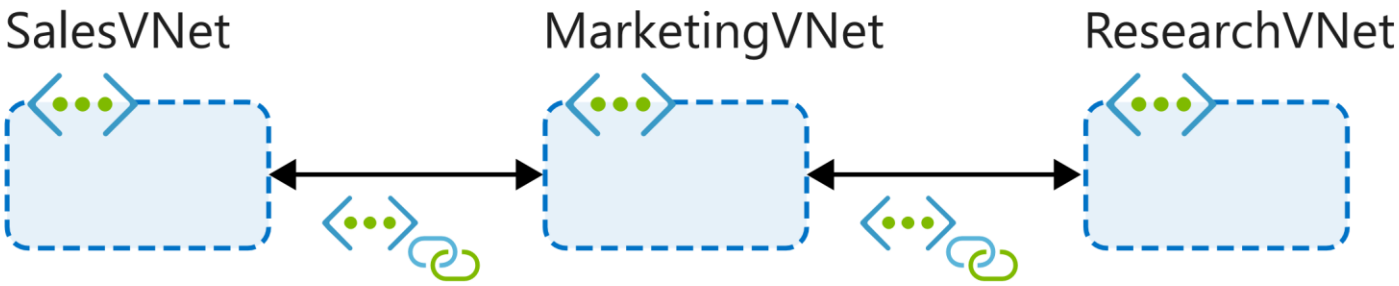
Exercise - Verify virtual network peering by using SSH between Azure virtual machines

10 minutes

This module requires a sandbox to complete. A **sandbox** gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

Activate sandbox

In the previous unit, you configured peering connections between the virtual networks to enable resources to communicate with each other. Your configuration used a hub and spoke topology. MarketingVNet was the hub, and SalesVNet and ResearchVNet were spokes.



Remember, peering connections are nontransitive. Intermediate virtual networks don't allow connectivity to flow through them to connected virtual networks. **SalesVNet** can communicate with **MarketingVNet**. **ResearchVNet** can communicate with **MarketingVNet**. **MarketingVNet** can communicate with both **SalesVNet** and **ResearchVNet**. The only communication that's not permitted is between **SalesVNet** and **ResearchVNet**. Even though **SalesVNet** and **ResearchVNet** are both connected to **MarketingVNet**, they can't communicate with each other because they're not directly peered to each other.

Let's confirm the connectivity across the peering connections. To do this, you'll first create a connection from Azure Cloud Shell to a target VM's *public* IP address. Then you'll connect from the target VM to the destination VM by using the destination VM's *private* IP address.

> **Important**
>
> To test the virtual network peering connection, connect to the private IP address assigned to each VM.

1. To connect to your VMs, you'll use SSH (Secure Shell) directly from Cloud Shell. When using SSH, you first find the public IP addresses that are assigned to your test VMs.
2. In Cloud Shell, run the following command to list the IP addresses you'll use to connect to the VMs:

   Azure CLI | Copy

   ```
   az vm list \
       --resource-group [sandbox resource group name] \
       --query "[*].{Name:name, PrivateIP:privateIps, PublicIP:publicIps}" \
       --show-details \
       --output table
   ```

3. Record the output. You'll need the IP addresses for the exercises in this unit.

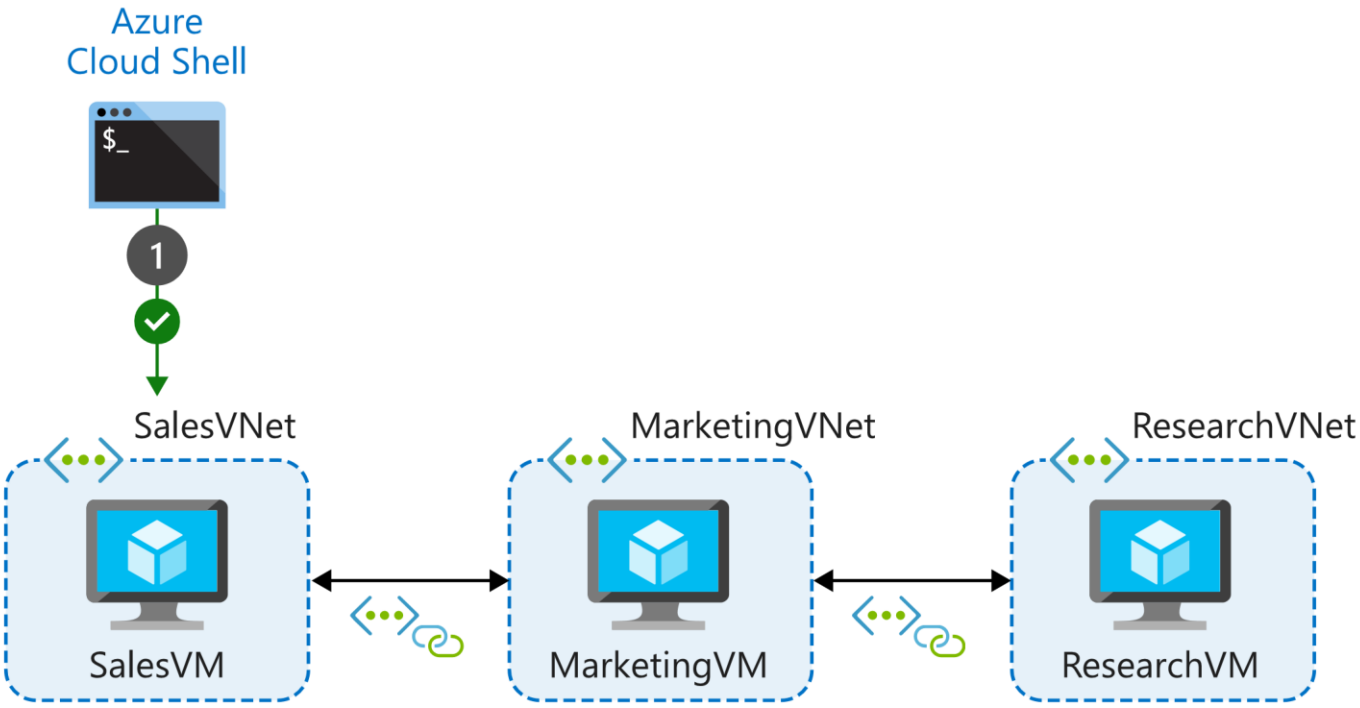
Before you start the tests, think about what you've learned in this module. What results do you expect? Which VMs will and will not be able to communicate with each other?

Test connections from SalesVM

In the first test, in Cloud Shell you'll use SSH to connect to the public IP address of **SalesVM**. You'll then attempt to connect from **SalesVM** to **MarketingVM** and **ResearchVM**.

1. In Cloud Shell, run the following command, using SSH to connect to the public IP address of **SalesVM**. In the command, replace `<SalesVM public IP>` with the VM's *public* IP address.

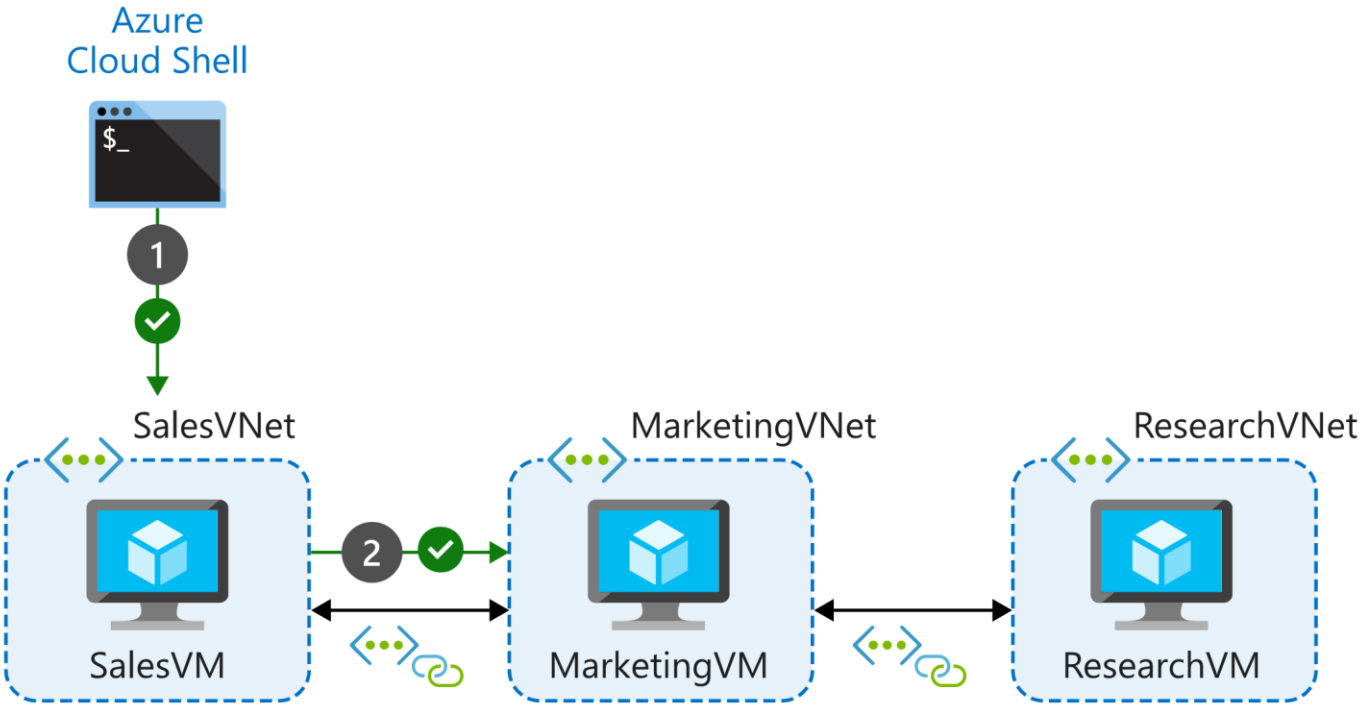
```
bash
ssh -o StrictHostKeyChecking=no azureuser@<SalesVM public IP>
```



2. Sign in with the password that you used to create the VM. The prompt now shows that you're signed in to **SalesVM**.

3. In Cloud Shell, run the following command, using SSH to connect to the private IP address of **MarketingVM**. In the command, replace `<MarketingVM private IP>` with this VM's *private* IP address.

```
bash
ssh -o StrictHostKeyChecking=no azureuser@<MarketingVM private IP>
```



The connection attempt should succeed because of the peering connection between the **SalesVNet** and **MarketingVNet** virtual networks.

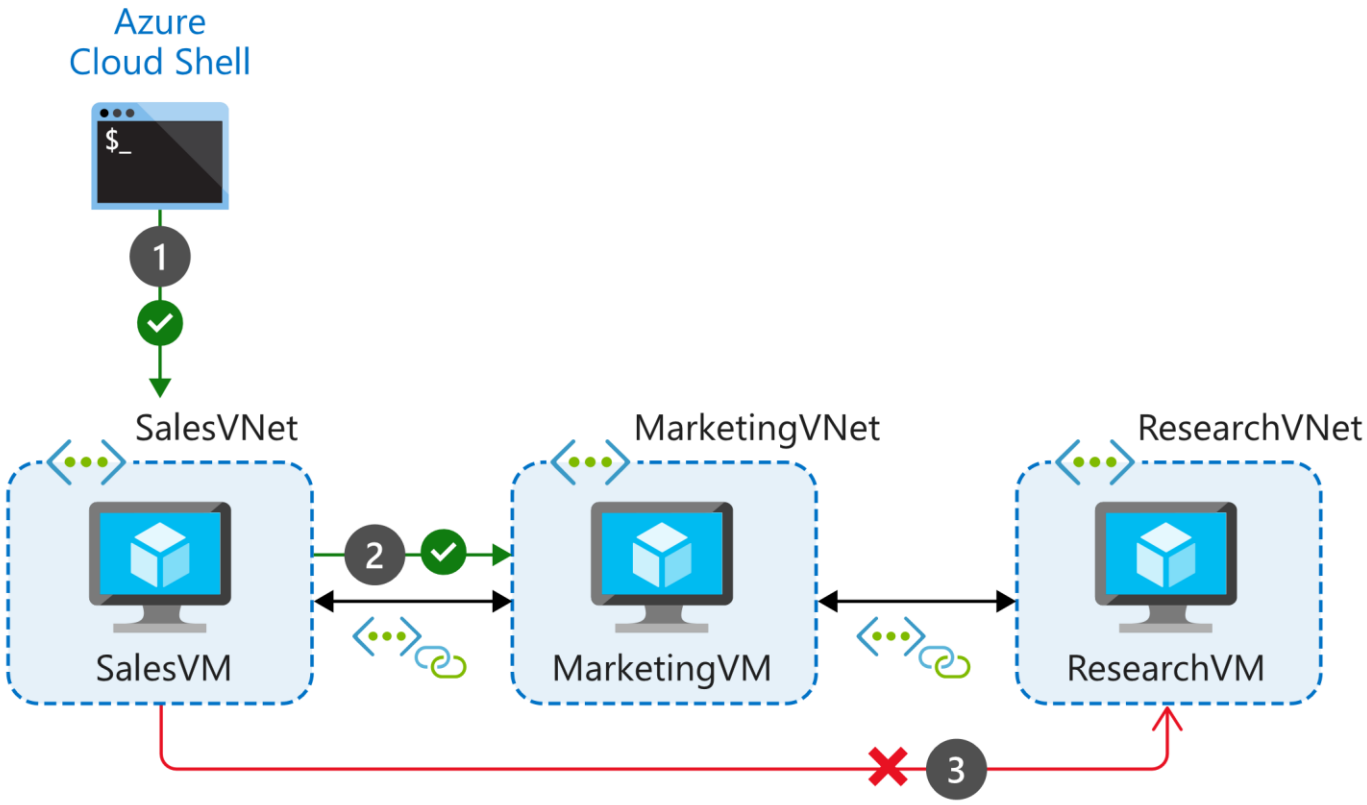
4. Sign in by using the password you used to create the VM.

5. Enter `exit` to close this SSH session and return to the **SalesVM** prompt.

6. In Cloud Shell, run the following command, using SSH to connect to the private IP address of **ResearchVM**. In the command, replace `<ResearchVM private IP>` with this VM's *private* IP address.

```bash
ssh -o StrictHostKeyChecking=no azureuser@<ResearchVM private IP>
```

7. The connection attempt should fail because there's no peering connection between the **SalesVNet** and **ResearchVNet** virtual networks. Up to 60 seconds might pass before the connection attempt times out. To force the attempt to stop, use Ctrl+C.



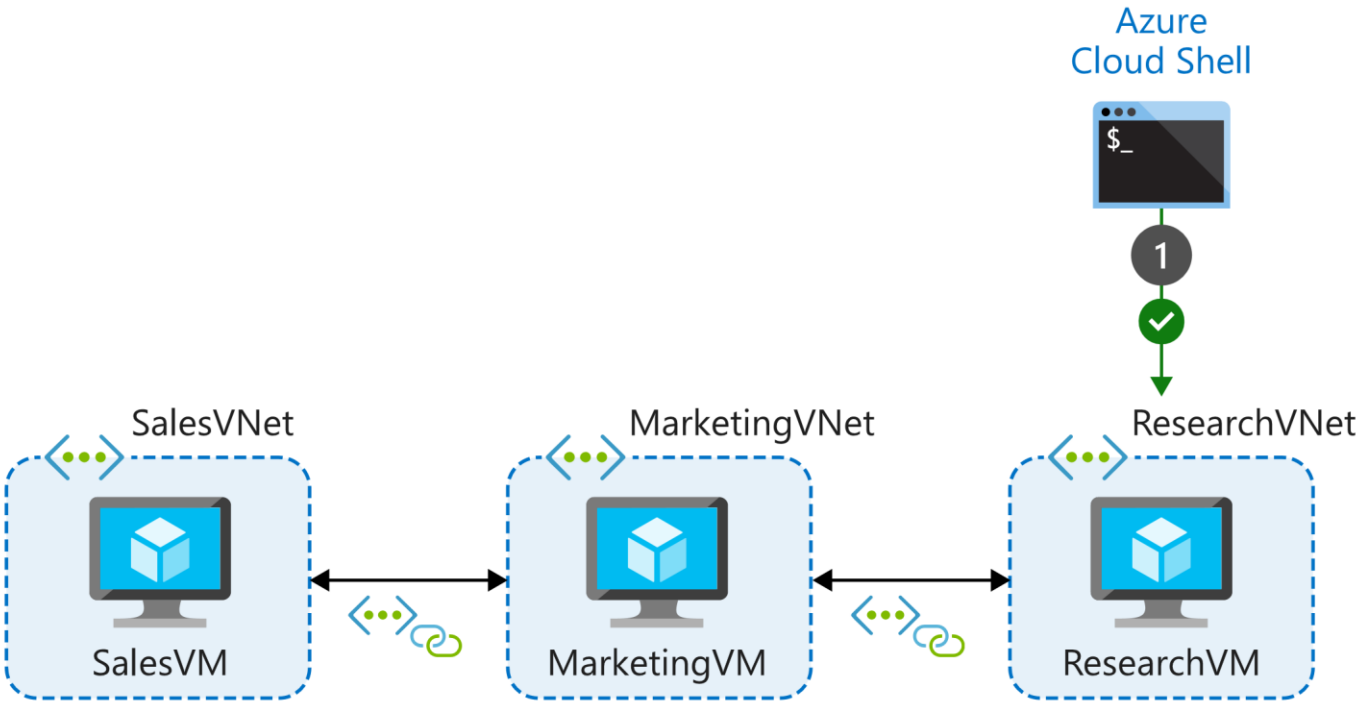
8. Enter `exit` to close the SSH session and return to Cloud Shell.

Test connections from ResearchVM

In the second test, in Cloud Shell you'll use SSH to connect to the public IP address of **ResearchVM**. You'll then attempt to connect from **ResearchVM** to **MarketingVM** and **SalesVM**.

1. In Cloud Shell, run the following command, using SSH to connect to the public IP address of **ResearchVM**. In the command, replace `<ResearchVM public IP>` with this VM's *public* IP address.

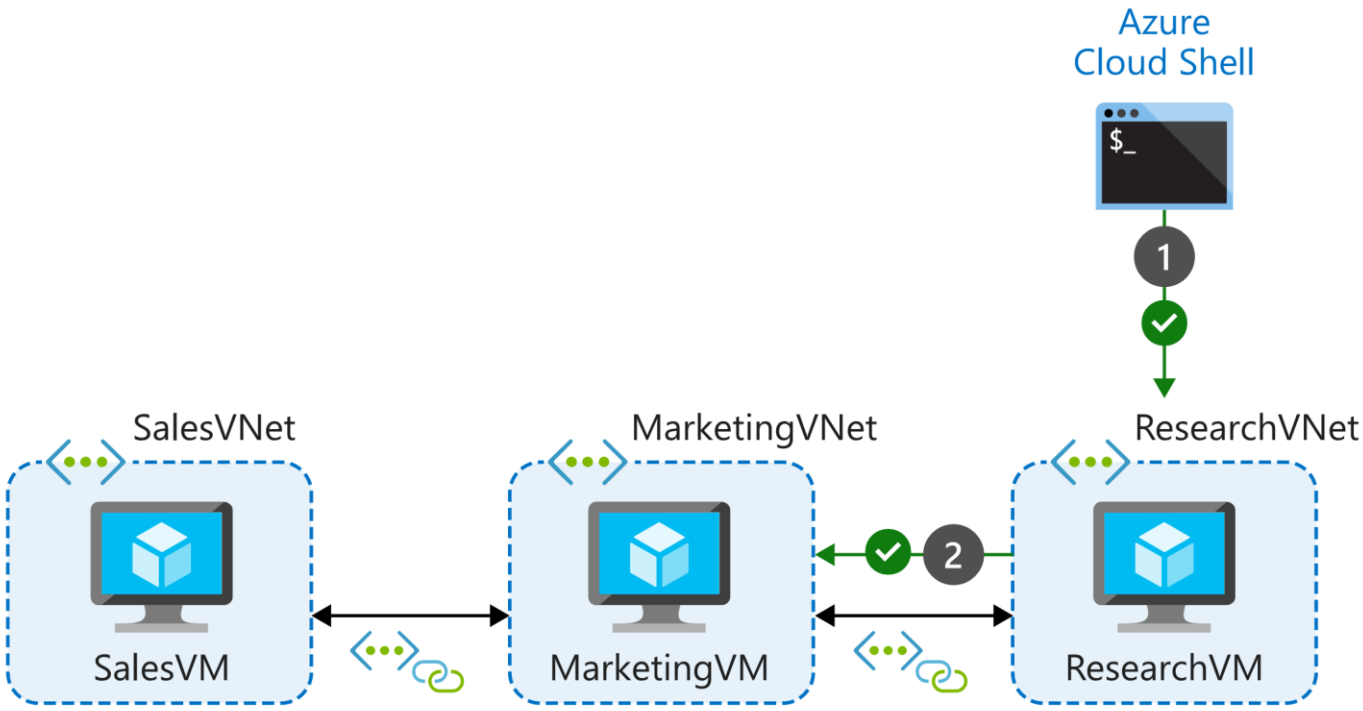
```bash
ssh -o StrictHostKeyChecking=no azureuser@<ResearchVM public IP>
```



2. Sign in by using the password that you used to create the VM. The prompt now shows that you're signed in to **ResearchVM**.

3. In Cloud Shell, run the following command, using SSH to connect to the private IP address of **MarketingVM**. In the command, replace `<MarketingVM private IP>` with this VM's *private* IP address.

```
bash
ssh -o StrictHostKeyChecking=no azureuser@<MarketingVM private IP>
```



The connection attempt should succeed because of the peering connection between the **ResearchVNet** and **MarketingVNet** virtual networks.

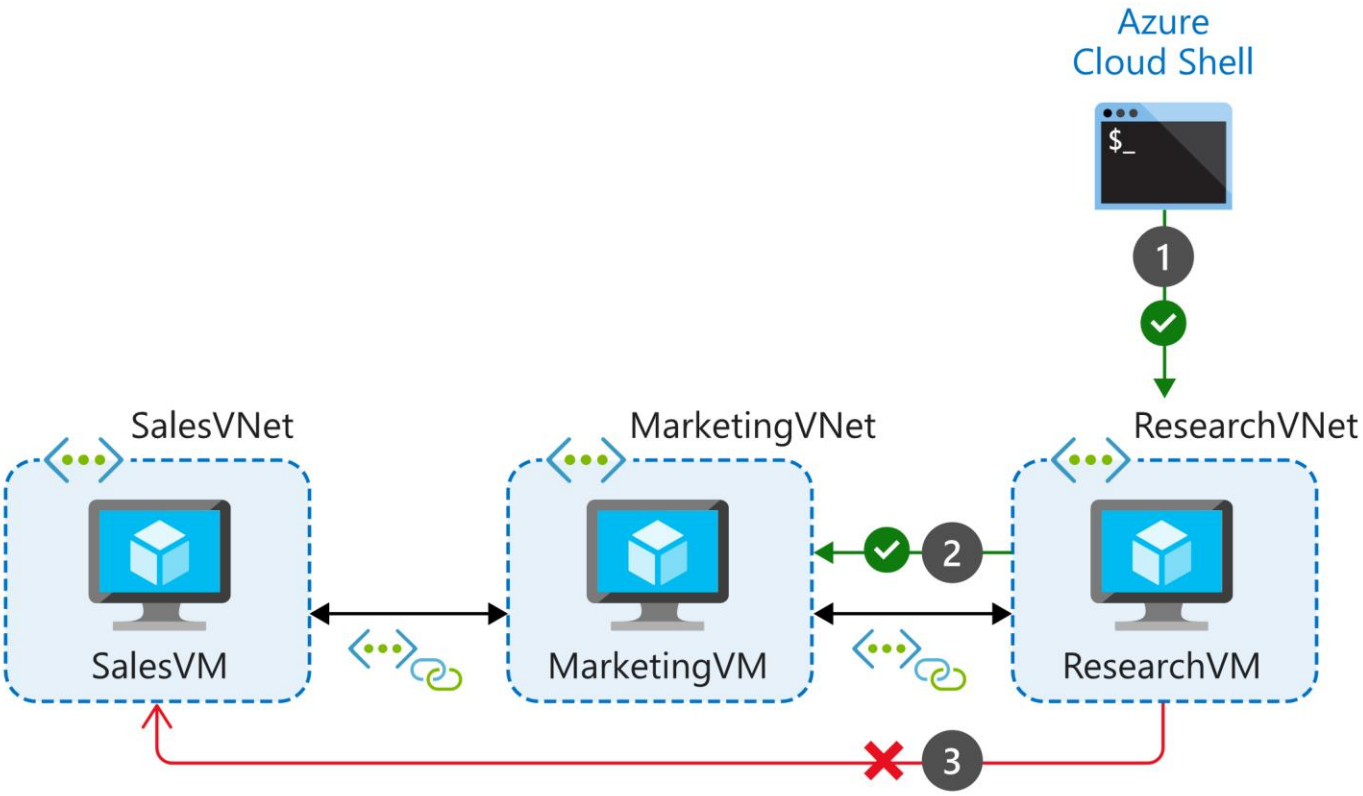
4. Sign in by using the password you used to create the VM.

5. Enter `exit` to close this SSH session and return to the **ResearchVM** prompt.

6. In Cloud Shell, run the following command, using SSH to connect to the private IP address of **SalesVM**. In the command, replace `<SalesVM private IP>` with this VM's *private* IP address.

```
bash
ssh -o StrictHostKeyChecking=no azureuser@<SalesVM private IP>
```

7. The connection attempt should fail because there's no peering connection between the **ResearchVNet** and **SalesVNet** virtual networks. Up to 60 seconds might pass before the connection attempt times out. To force the attempt to stop, use Ctrl+C.



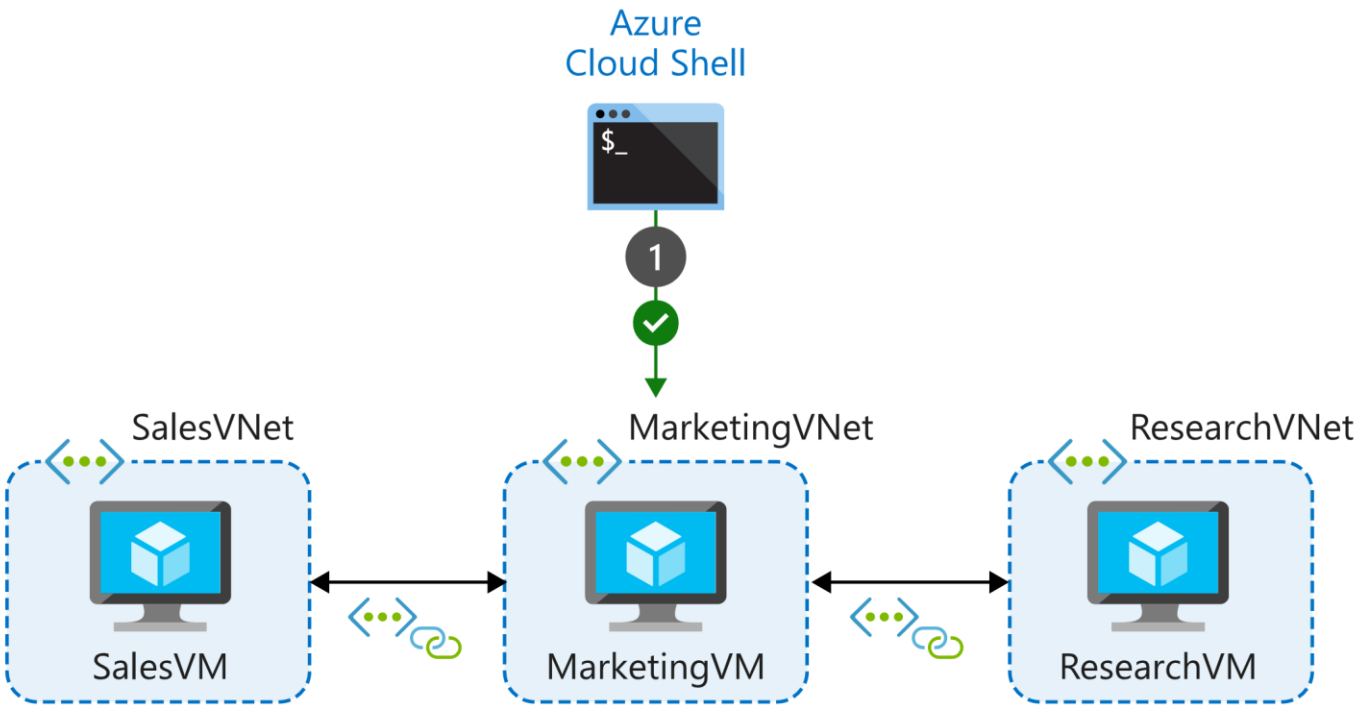
8. Enter `exit` to close the SSH session and return to Cloud Shell.

Test connections from Marketing VM

In the final test, in Cloud Shell you'll use SSH to connect to the public IP address of **MarketingVM**. You'll then attempt to connect from **MarketingVM** to **ResearchVM** and **SalesVM**.

1. In Cloud Shell, run the following command, using SSH to connect to the public IP address of **MarketingVM**. In the command, replace `<MarketingVM public IP>` with this VM's *public* IP address.

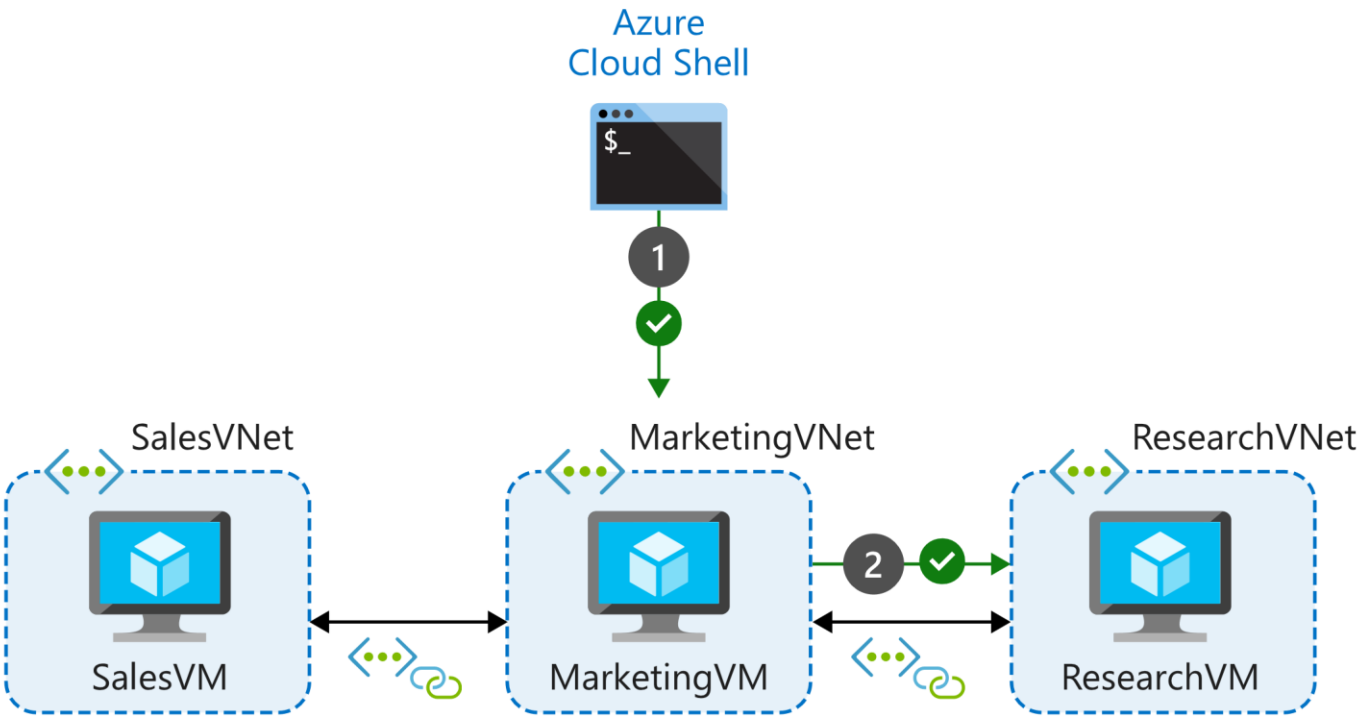
```
bash
ssh -o StrictHostKeyChecking=no azureuser@<MarketingVM public IP>
```



2. Sign in by using the password that you used to create the VM. The prompt shows that you're signed in to **MarketingVM**.

3. In Cloud Shell, run the following command, using SSH to connect to the private IP address of **ResearchVM**. In the command, replace `<ResearchVM private IP>` with this VM's *private* IP address.

```
bash
ssh -o StrictHostKeyChecking=no azureuser@<ResearchVM private IP>
```



The connection attempt should succeed because of the peering connection between the **MarketingVNet** and **ResearchVNet** virtual networks.

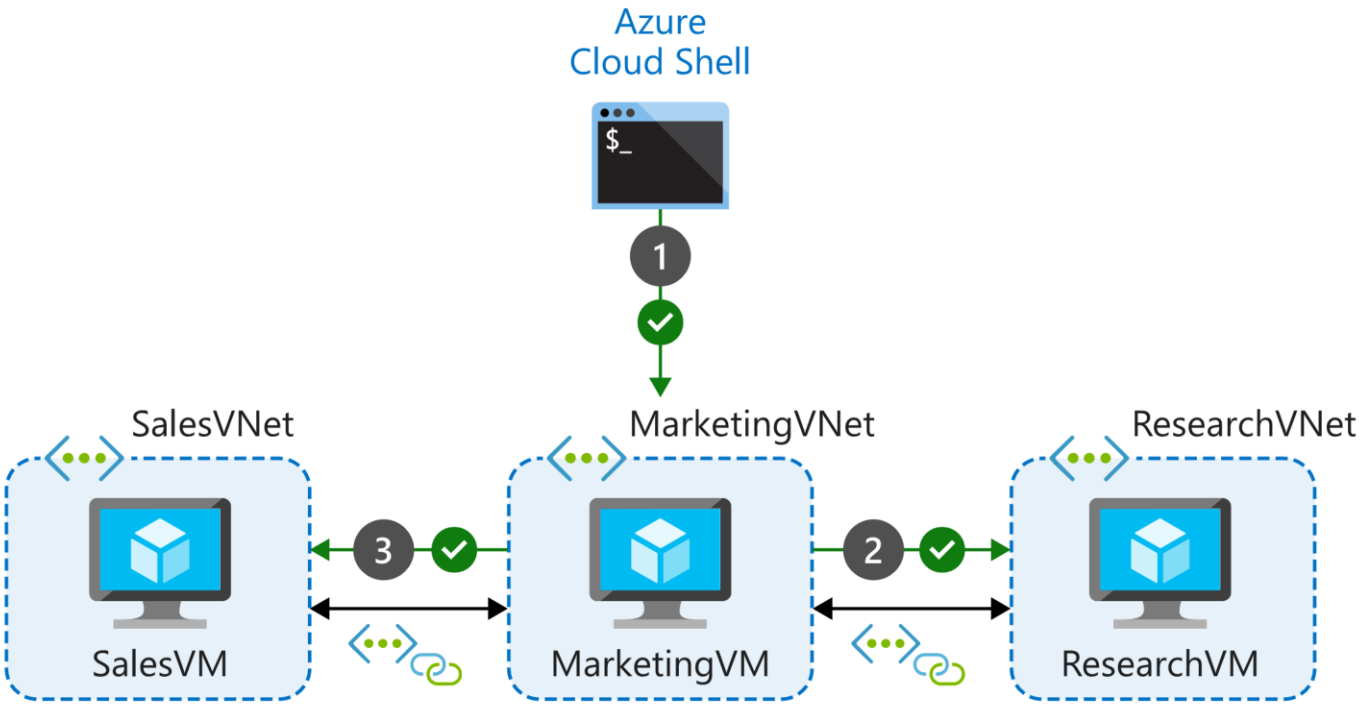
4. Sign in by using the password you used to create the VM.

5. Enter `exit` to close this SSH session and return to the **MarketingVM** prompt.

6. In Cloud Shell, run the following command, using SSH to connect to the private IP address of **SalesVM**. In the command, replace `<SalesVM private IP>` with this VM's *private* IP address.

```
bash
ssh -o StrictHostKeyChecking=no azureuser@<SalesVM private IP>
```

The connection attempt should also succeed because there *is* a peering connection between the **MarketingVNet** and **SalesVNet** virtual networks.



7. Sign in by using the password you used to create the VM.
8. Enter `exit` to close this SSH session and return to the **MarketingVM** prompt.
9. Enter `exit` to close the SSH session and return to Cloud Shell.

This is a simple test using SSH. It demonstrates network connectivity between peered virtual networks. It also demonstrates lack of network connectivity for transitive connections.

If these servers were running application services, the server connectivity would allow communication between the services running on the VMs. The connectivity would allow the business to share data across departments as required.

Next unit: Summary

Continue >

Azure Cloud Shell

This module requires a sandbox to complete. A sandbox gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

2/21/2020 Summary - Learn | Microsoft Docs

R Previous Unit 6 of 6 S

100 XP

Summary

2 minutes

In this module, you learned how to use peering to connect virtual networks in a hub and spoke topology. You used VMs and SSH to verify connectivity between virtual networks. The peering connections will enable communication for services that run on the VMs.

Now that you understand how to peer virtual networks together, you can use this cost-effective and minimally complex method in your Azure network infrastructure. The method enables low-latency communication between resources in virtual networks. It supports scenarios where resources are in different regions or subscriptions. Virtual network peering should be your first choice when you need to connect virtual networks.

Clean up

The sandbox automatically cleans up your resources when you're finished with this module.

When you're working in your own subscription, it's a good idea at the end of a project to identify whether you still need the resources you created. Resources left running can cost you money. You can delete resources individually or delete the resource group to delete the entire set of resources.

Module complete:

Unlock achievement

https://docs.microsoft.com/en-us/learn/modules/integrate-vnets-with-vnet-peering/6-summary 1/2

2/21/2020 Enhance your service availability and data locality by using Azure Traffic Manager - Learn | Microsoft Docs

600 XP

Enhance your service availability and data locality by using Azure Traffic Manager

29 min • Module • 6 Units

V V V V W 4.5 (538) Rate it

Beginner Solutions Architect Administrator Azure Traffic Manager Virtual Network

Azure Traffic Manager provides DNS load balancing to your application, so you improve your ability to distribute your application around the world. Use Traffic Manager to improve the performance and availability of your application.

In this module, you will:

Learn how to use Traffic Manager to dynamically distribute network traffic
Set up Traffic Manager for automatic failover to a secondary region
Set up Traffic Manager to redirect client requests to the nearest endpoint

- Familiarity with Azure App Service
- Basic knowledge of networking and DNS

This module is part of these learning paths
Architect network infrastructure in Azure

Introduction
2 min

Route network traffic by using Traffic Manager
6 min

Exercise - Enable automatic failover by using priority routing
10 min

Optimize applications across regions by using performance routing 2 min

Exercise - Optimize applications across regions by using performance routing 7 min

Summary
2 min

https://docs.microsoft.com/en-us/learn/modules/distribute-load-with-traffic-manager/ 1/1

2/21/2020 Introduction - Learn | Microsoft Docs

Unit 1 of 6 S Next T

100 XP

Introduction

2 minutes

Suppose you're the lead architect for a company that provides a global music streaming web application. You want your customers, wherever they are in the world, to experience near-zero downtime. The application needs to be responsive. You know that poor performance might drive your customers to your competitors. You'd also like to have customized experiences for customers who are in specific regions for user interface, legal, and operational reasons.

Azure Traffic Manager is a DNS-based traffic load balancer that you can use to distribute traffic optimally to services across Azure regions globally. You can use Traffic Manager to distribute traffic to different regions while providing high availability, resilience, and responsiveness in your app.

Learning objectives

By the end of this module, the learner will be able to:

- Describe how to use Traffic Manager to dynamically distribute network traffic
- Set up Traffic Manager for automatic failover to a secondary region
- Set up Traffic Manager to redirect client requests to the nearest endpoint

Prerequisites

- Familiarity with Azure App Service
- Basic knowledge of networking and DNS

Next unit: Route network traffic by using Traffic Manager

Continue

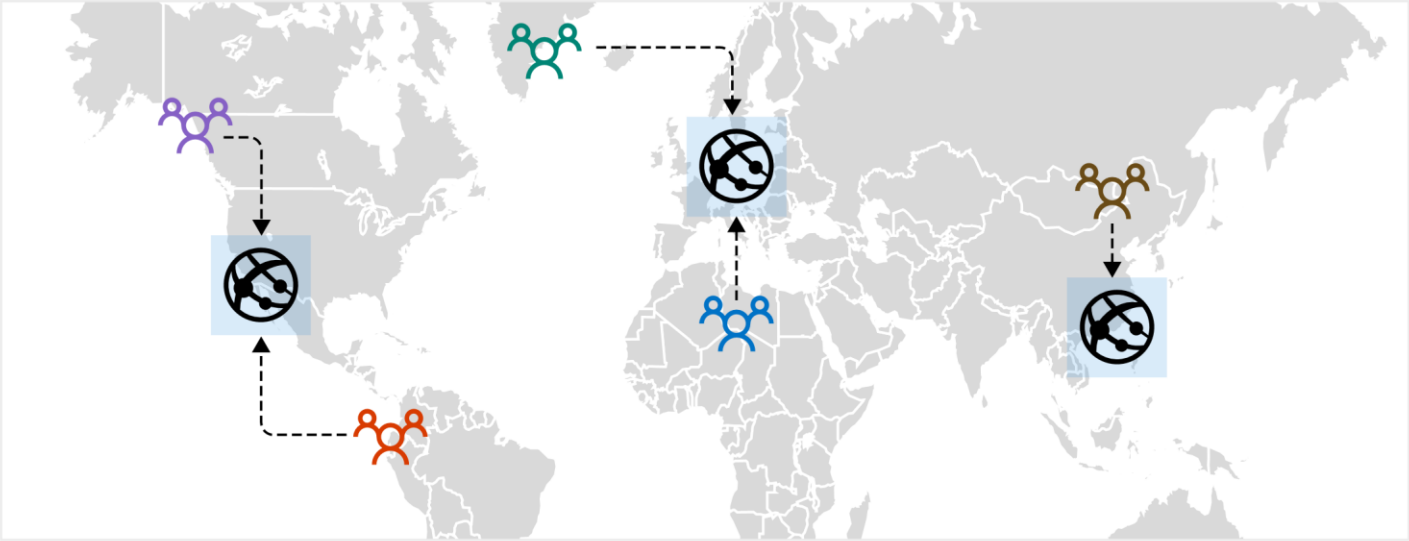
https://docs.microsoft.com/en-us/learn/modules/distribute-load-with-traffic-manager/1-introduction 1/2

100 XP

Route network traffic by using Traffic Manager

6 minutes

Your customers require 24x7 availability of your company's streaming music application. Cloud services in one region might become unavailable because of technical issues, such as planned maintenance or scheduled security updates. In these scenarios, your company wants to have a failover endpoint so your customers can continue to access its services. To manage routing traffic and to handle these situations, you've decided to implement Azure Traffic Manager.



How Traffic Manager works

When a client attempts to connect to a service, first it resolves the DNS name of the service as an IP address. The client then connects to that IP address to access the service.

Traffic Manager uses DNS to direct clients to a specific service endpoint IP address based on the rules of the traffic routing method that's used. Clients connect directly to the selected endpoint. Traffic Manager isn't a proxy or gateway. Traffic Manager doesn't see the traffic that passes between the clients and the service; it just gives clients the IP address of where they need to go.

Traffic Manager endpoints

Endpoints are the destination location that is returned to the client. You configure each application deployment as an 'endpoint' in Traffic Manager. When Traffic Manager receives a DNS request, it chooses an available endpoint to return in the DNS response. There are three types of endpoint supported by Traffic Manager:

- **Azure endpoints** are used for services hosted in Azure. These can be services like Azure App Service, as well as public IP resources that are associated with load balancers or virtual machines.
- **External endpoints** are used for IPv4/IPv6 addresses, FQDNs, or for services hosted outside Azure that can either be on-premises or with a different hosting provider.
- **Nested endpoints** are used to combine Traffic Manager profiles to create more flexible traffic-routing schemes to support the needs of larger, more complex deployments.

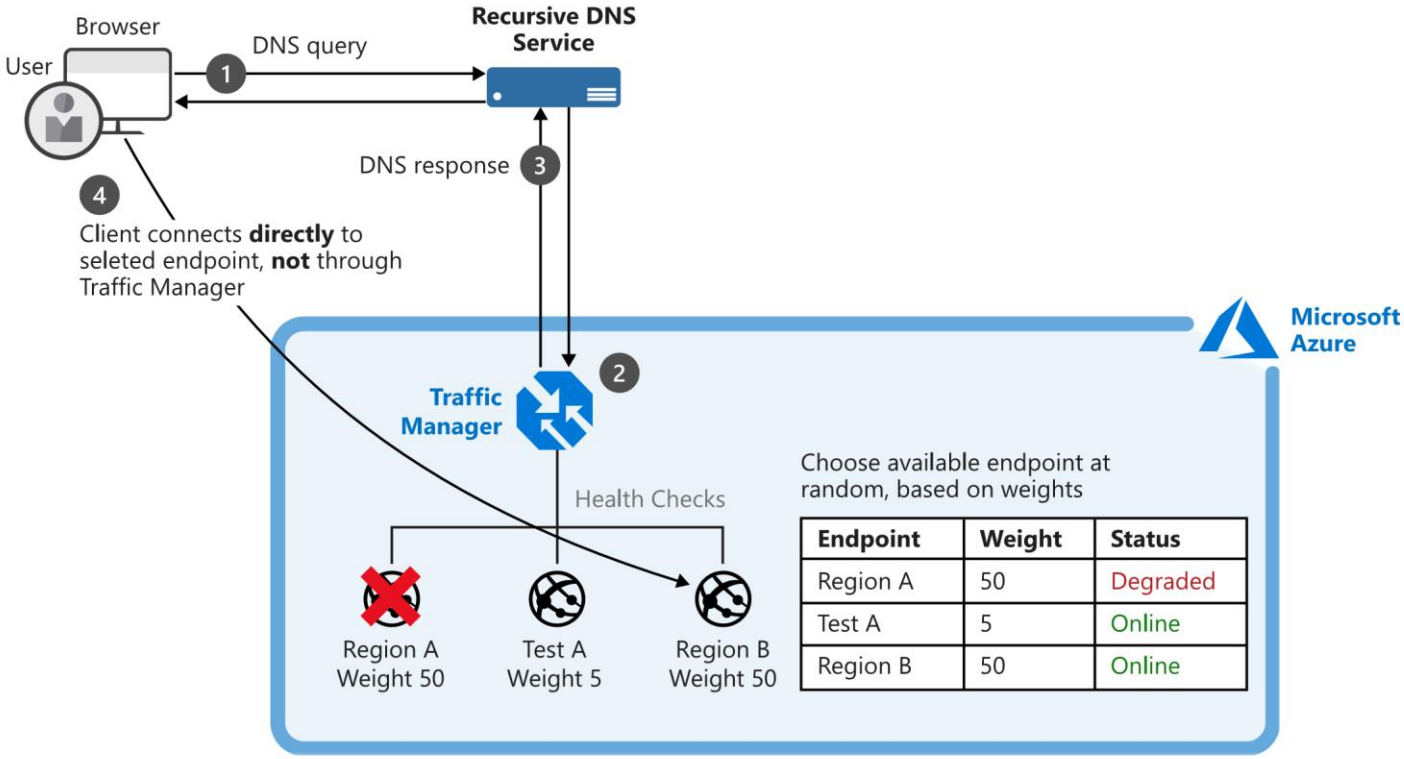
There is no restriction on how endpoints of different types are combined in a single Traffic Manager profile. Each profile can contain any mix of endpoint types.

Traffic Manager routing methods

Traffic Manager supports different methods for choosing how traffic is routed to multiple endpoints. Traffic Manager applies a traffic routing method to each DNS query it receives and determines which endpoint is returned in the response. You can choose from six traffic routing methods.

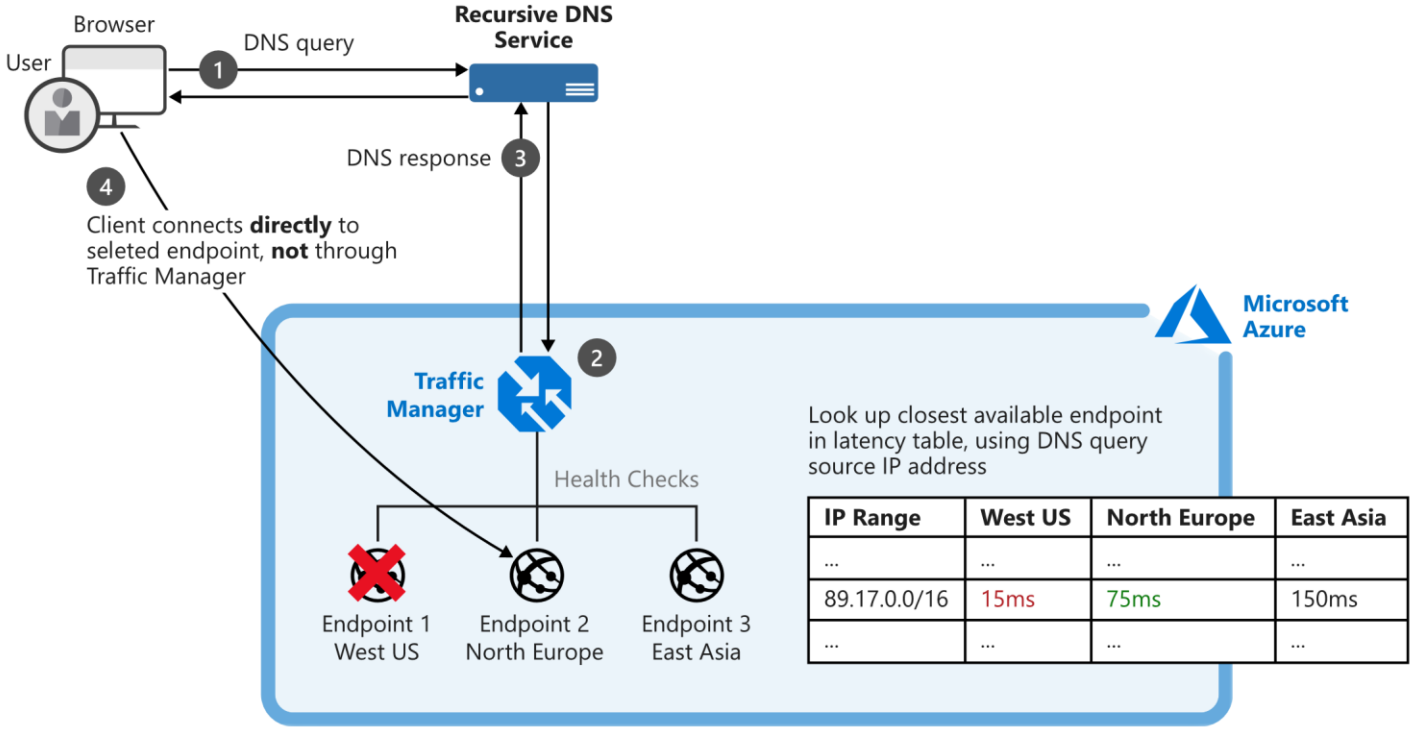
Weighted routing

Choose weighted when you want to distribute traffic across a set of endpoints, either evenly or based on different weights. The weight is an integer from 1 to 1,000. For each DNS query received, Traffic Manager randomly chooses an available endpoint. The probability of choosing an endpoint is based on the weights assigned to all available endpoints.



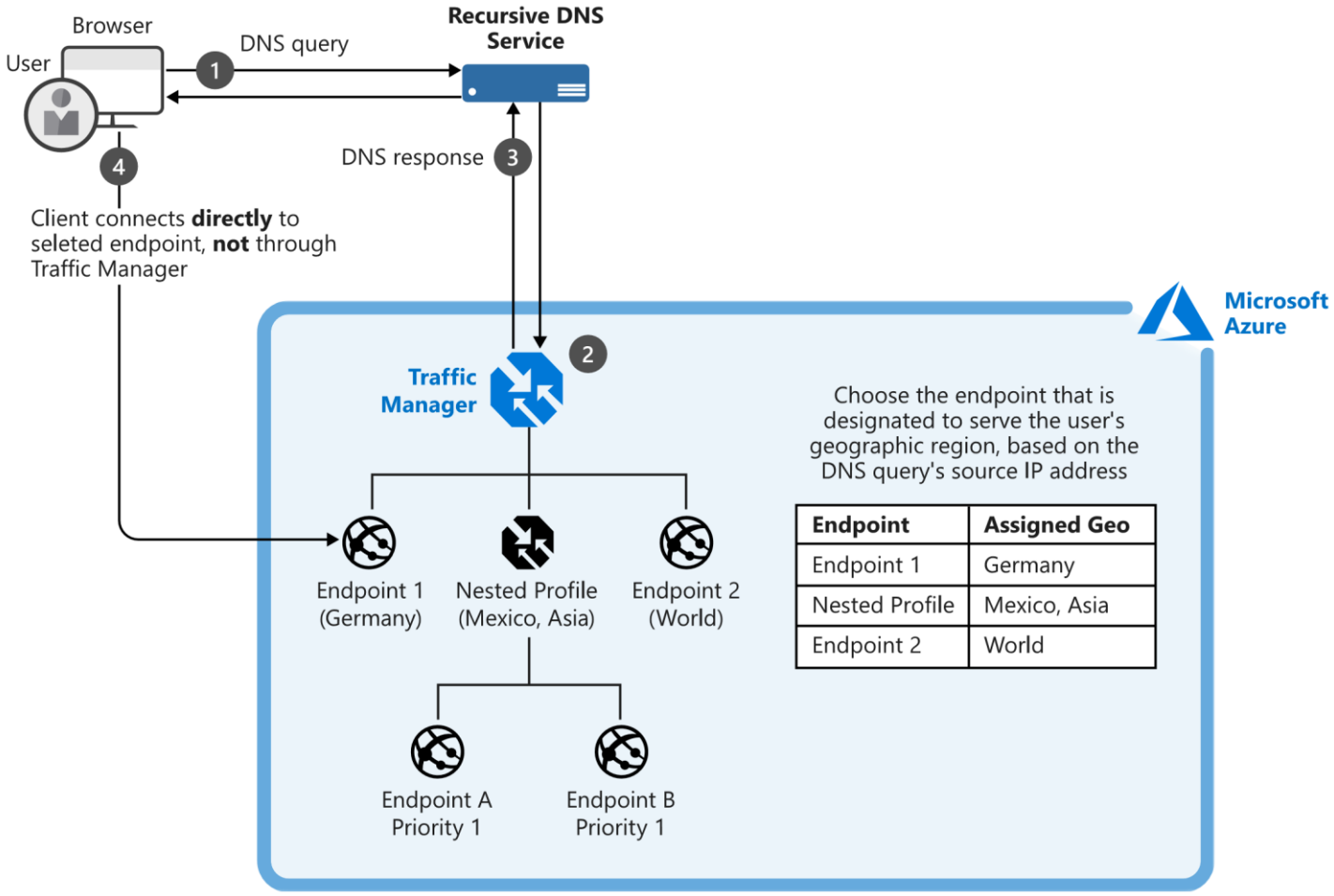
Performance routing

If you have endpoints in different geographic locations, you can use performance routing to send users to the endpoint that has the best performance for the user. To choose the best endpoint to use, this routing method uses an internet latency table, which actively tracks network latencies to the endpoints from locations around the globe. When a user makes a request, Traffic Manager returns the best performing endpoint based on the location of the request.



Geographic routing

With the geographic routing method, users are directed to specific endpoints based on where their DNS query originates. Using this method allows you to geo-fence content to specific user regions. For example, European users can be directed to an endpoint in Europe that has specific terms and conditions for regional compliance. Users in China can be directed to an endpoint that has been localized in Mandarin.



Multivalue routing

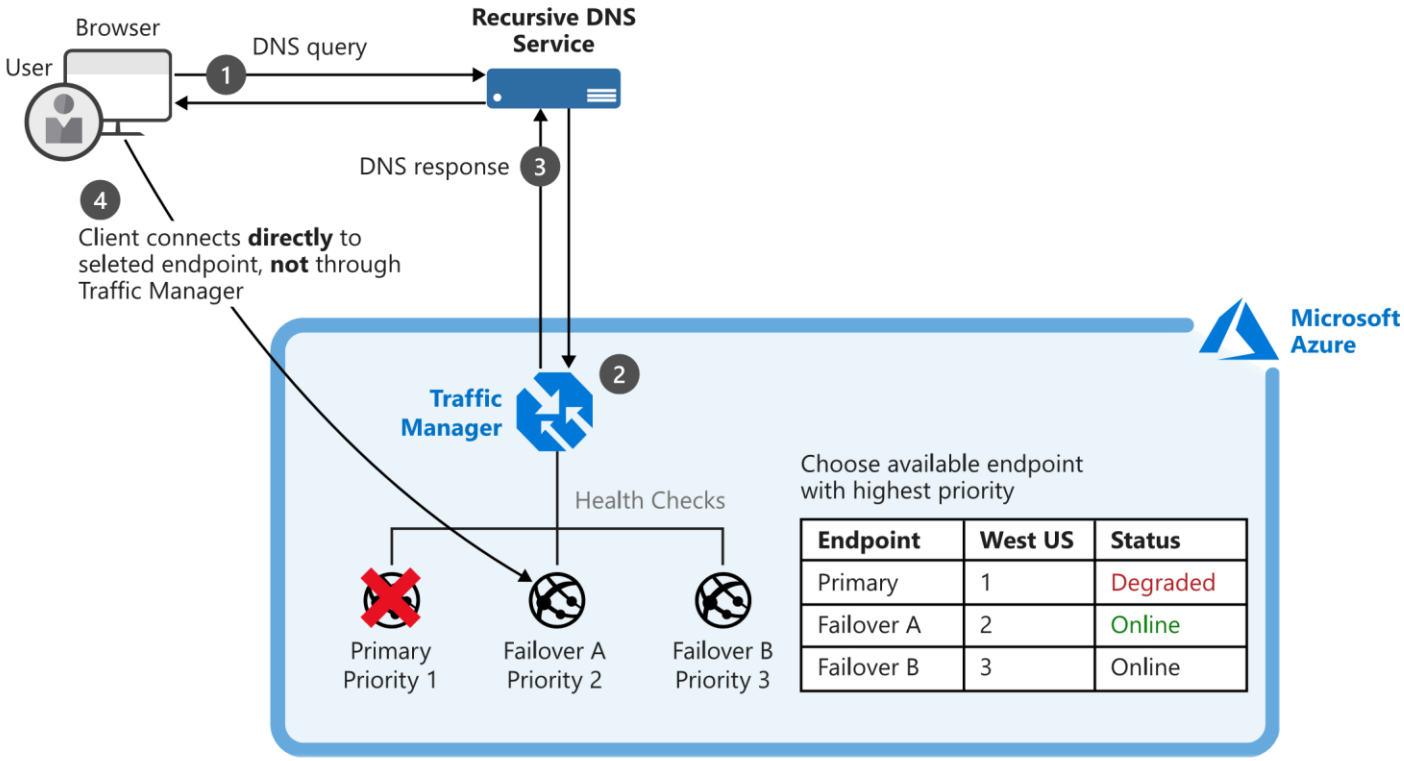
You can use the multivalue routing method to get multiple healthy endpoints in a single DNS query response. The caller can make client-side retries with other endpoints if an endpoint is unresponsive. This pattern can increase the availability of a service and reduce the latency associated with a new DNS query to obtain a healthy endpoint.

Subnet routing

This method maps the set of user IP address ranges to specific endpoints within a Traffic Manager profile. When a request is received, the endpoint returned will be the one mapped for that request's source IP address. For example, using subnet routing, a customer can route all requests from their corporate office to a different endpoint, where they might be testing an internal-only version of the app. Another scenario is if you want to provide a different experience to users who connect from a specific ISP (for example, to block users from a specific ISP).

Priority routing

The Traffic Manager profile contains a prioritized list of service endpoints. By default, Traffic Manager sends all traffic to the primary (highest-priority) endpoint. If the primary endpoint isn't available, Traffic Manager routes the traffic to the second endpoint. If both the primary and secondary endpoints are not available, the traffic goes to the third endpoint, and so on. Availability of the endpoint is based on the configured status (enabled or disabled) and the ongoing endpoint monitoring that is set up.



Next unit: Exercise - Enable automatic failover by using priority routing

Continue >

100 XP

Exercise - Enable automatic failover by using priority routing

10 minutes

This module requires a sandbox to complete. A **sandbox** gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

Activate sandbox

Let's assume that your music streaming application has an equal distribution of users in the western United States and eastern Asia. You'd like to have a failover version of the app in one region.

The sample application we use for this exercise displays the region it's running in. One of the two instances has higher priority and is the primary endpoint. The other instance has a lower priority and is the failover endpoint. Taking the primary endpoint offline automatically routes all traffic to the failover endpoint.

In this exercise, you set up Traffic Manager to use the United States endpoint as the primary, failing over to the Asian endpoint if any errors occur.

Create a new Traffic Manager profile

1. Run this command in the Cloud Shell to create a new Traffic Manager profile.

Azure CLI = Copy

```
az network traffic-manager profile create \
  --resource-group Sandbox resource group  \
  --name TM-MusicStream-Priority \
  --routing-method Priority \
  --unique-dns-name TM-MusicStream-Priority-$RANDOM
```

You use these parameters in the command:

- **-routing-method Priority**: Creates the Traffic Manager profile by using the priority routing method.
- **-unique-dns-name**: Creates the globally unique domain name <unique-dns-name>.trafficmanager.net. We use the $RANDOM Bash function to return a random whole number to ensure that the name is unique.

Deploy the web applications

1. Run this command to deploy a Resource Manager template. The template creates two servers, one in the East Asia region, and one in the West US 2 region.

Azure CLI = Copy

```
az group deployment create \
  --resource-group Sandbox resource group  \
  --template-uri  https://raw.githubusercontent.com/MicrosoftDocs/mslearn-distribute-load-with-trafficmanager/master/azuredeploy.json \
  --parameters password="$(head /dev/urandom | tr -dc A-Za-z0-9 | head -c 32)"
```

Add the endpoints to Traffic Manager

1. The web applications are now running on virtual machines. Run these commands to add the public IP address resources of the virtual machines as endpoints to the Traffic Manager profile.

Azure CLI = Copy

```
WestId=$(az network public-ip show \
  --resource-group Sandbox resource group  \
  --name westus2-vm-nic-pip \
  --query id \
  --out tsv)

az network traffic-manager endpoint create \    --resource-group Sandbox resource
group  \
  --profile-name TM-MusicStream-Priority \
  --name "Primary-WestUS" \
  --type azureEndpoints \
  --priority 1 \
  --target-resource-id $WestId

EastId=$(az network public-ip show \
  --resource-group Sandbox resource group  \
  --name eastasia-vm-nic-pip \
  --query id \
  --out tsv)

az network traffic-manager endpoint create \    --resource-group Sandbox resource
group  \
  --profile-name TM-MusicStream-Priority \
  --name "Failover-EastAsia" \
  --type azureEndpoints \
  --priority 2 \
  --target-resource-id $EastId
```

The code gets the resource IDs from both virtual machines. Then, the code uses the IDs to add them as endpoints to the Traffic Manager profile. The code uses the `--priority` flag to set the West US app to the highest priority.

2. Let's take a quick look at the endpoints we configured.

Azure CLI = Copy

```
az network traffic-manager endpoint list \    --resource-group Sandbox resource
group  \
  --profile-name TM-MusicStream-Priority \
  --output table
```

Test the app

1. Let's take a look at what DNS shows for the web apps and for our Traffic Manager profile. The following commands display the IP addresses for each of the resources we've created.

bash = Copy

```
# Retrieve the address for the West US 2 web app nslookup $(az network
public-ip show \
        --resource-group Sandbox resource group  \
        --name eastasia-vm-nic-pip \
        --query dnsSettings.fqdn \
        --output tsv)
# Retrieve the address for the East Asia web app nslookup $(az network public-
ip show \
        --resource-group Sandbox resource group  \
        --name westus2-vm-nic-pip \
        --query dnsSettings.fqdn \
        --output tsv)
# Retrieve the address for the Traffic Manager profile nslookup $(az network traffic-
manager profile show \
        --resource-group Sandbox resource group  \
        --name TM-MusicStream-Priority \
```

```
    --query dnsConfig.fqdn \
    --out tsv)
```

2. The address for the Traffic Manager profile should match the IP address for the **westus2-vm-nic-pip** public IP assigned to the **westus2-vm** virtual machine.

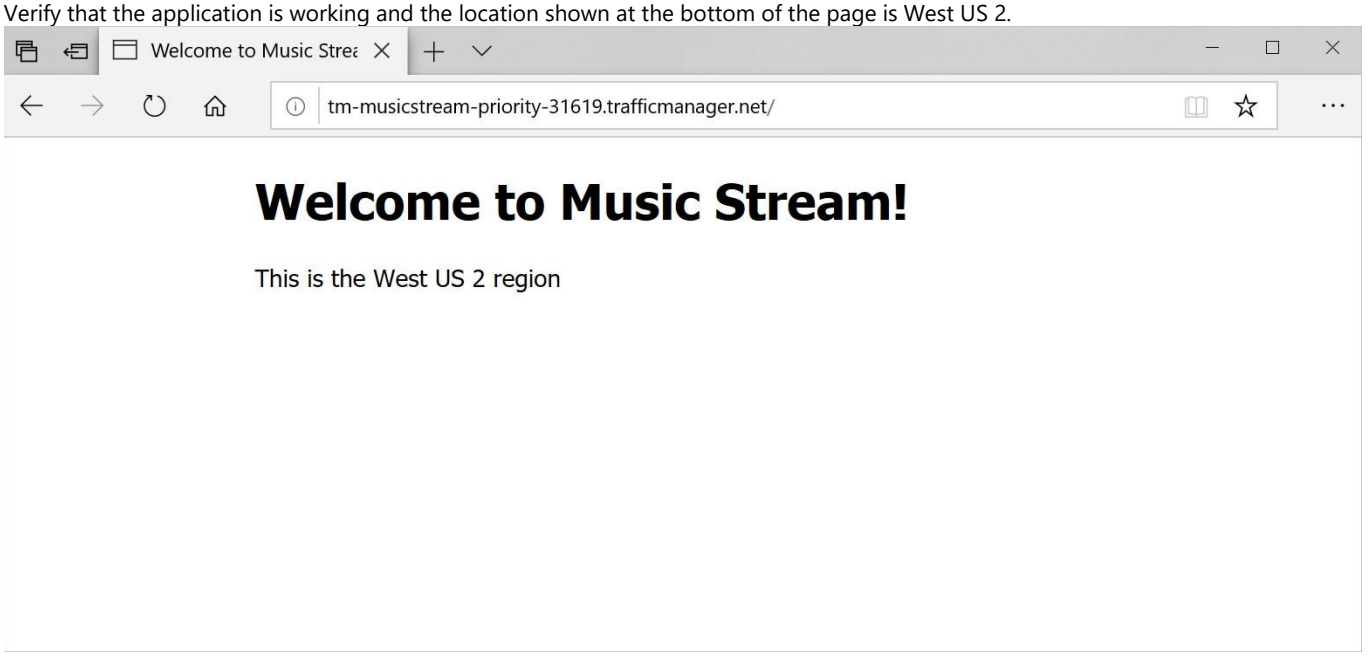
   Go to the Traffic Manager profile's fully qualified domain name (FQDN). Your request is routed to the endpoint that responds with the highest priority.

3. bash — Copy

```
echo http://$(az network traffic-manager profile show \
  --resource-group Sandbox resource group  \
  --name TM-MusicStream-Priority \
  --query dnsConfig.fqdn \
  --out tsv)
```

   The code prints out the FQDN in Cloud Shell. You can select the FQDN to open a new browser window or tab.

   Verify that the application is working and the location shown at the bottom of the page is West US 2.

4. 

   Disable the primary endpoint.

5. bash — Copy

```
az network traffic-manager endpoint update \
  --resource-group Sandbox resource group  \
  --name "Primary-WestUS" \
  --profile-name TM-MusicStream-Priority \
  --type azureEndpoints \
  --endpoint-status Disabled
```

   Let's look again at what DNS shows for the web apps and for our Traffic Manager profile.

bash — Copy

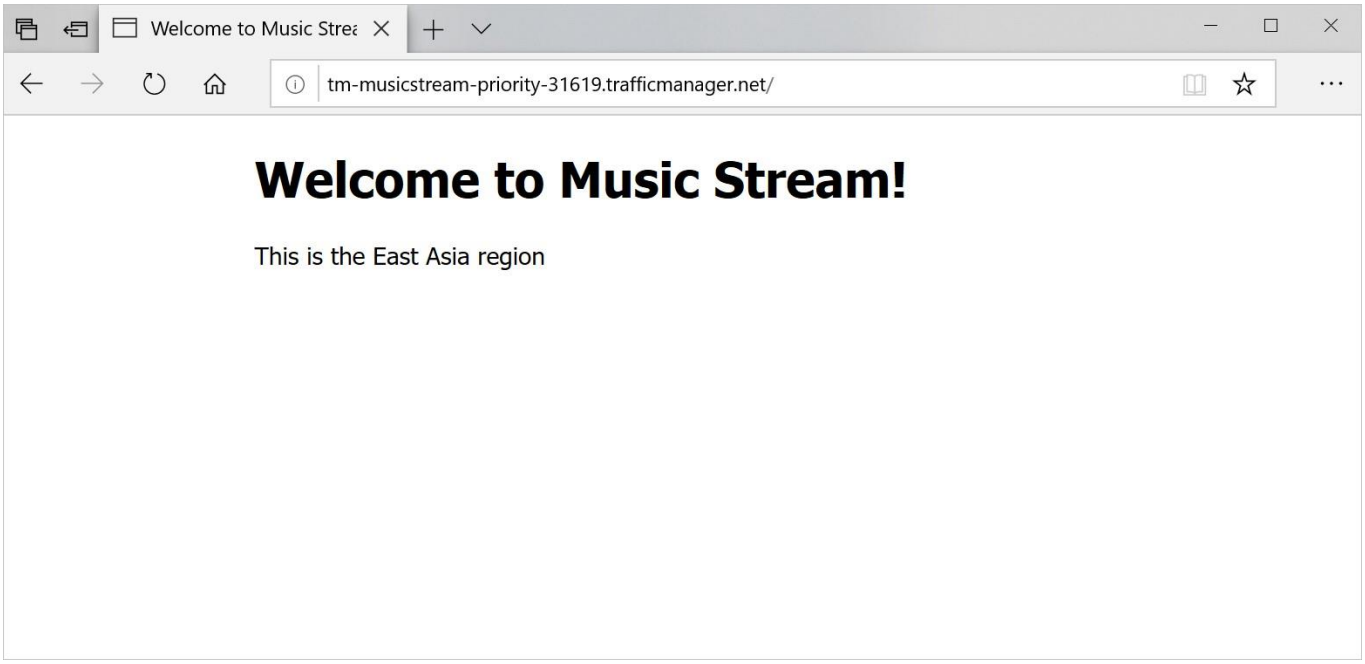
```
# Retrieve the address for the West US 2 web app nslookup $(az network public-ip
show \
        --resource-group Sandbox resource group  \
        --name eastasia-vm-nic-pip \
        --query dnsSettings.fqdn \
        --output tsv)
# Retrieve the address for the East Asia web app nslookup $(az network public-ip
show \
```

6.

```
        --resource-group Sandbox resource group  \
        --name westus2-vm-nic-pip \
        --query dnsSettings.fqdn \
        --output tsv)
# Retrieve the address for the Traffic Manager profile nslookup $(az network traffic-
manager profile show \
        --resource-group Sandbox resource group  \
        --name TM-MusicStream-Priority \
        --query dnsConfig.fqdn \
        --out tsv)
```

The address for the Traffic Manager profile should now match the East Asia web app.

Test the application again from your browser by refreshing the web page. Traffic Manager should automatically redirect the traffic to the East Asia endpoint. Depending on your browser, it might take a few minutes for the locally cached address to expire. Opening the site in a private window should bypass the cache, so you can see the change immediately.



t unit: Optimize applications across regions by using performance routing

ıtinue T

English (United States)

Previous Version Doc s • Blog • Contribut e • Privacy & Cookie s • Terms of Use • Trademarks

Azure Cloud Shell

This module requires a sandbox to complete. A sandbox gives you access to Azure resources. Your Azure subscription

will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

2/21/2020 Optimize applications across regions by using performance routing - Learn | Microsoft Docs

100 XP

Optimize applications across regions by using performance routing

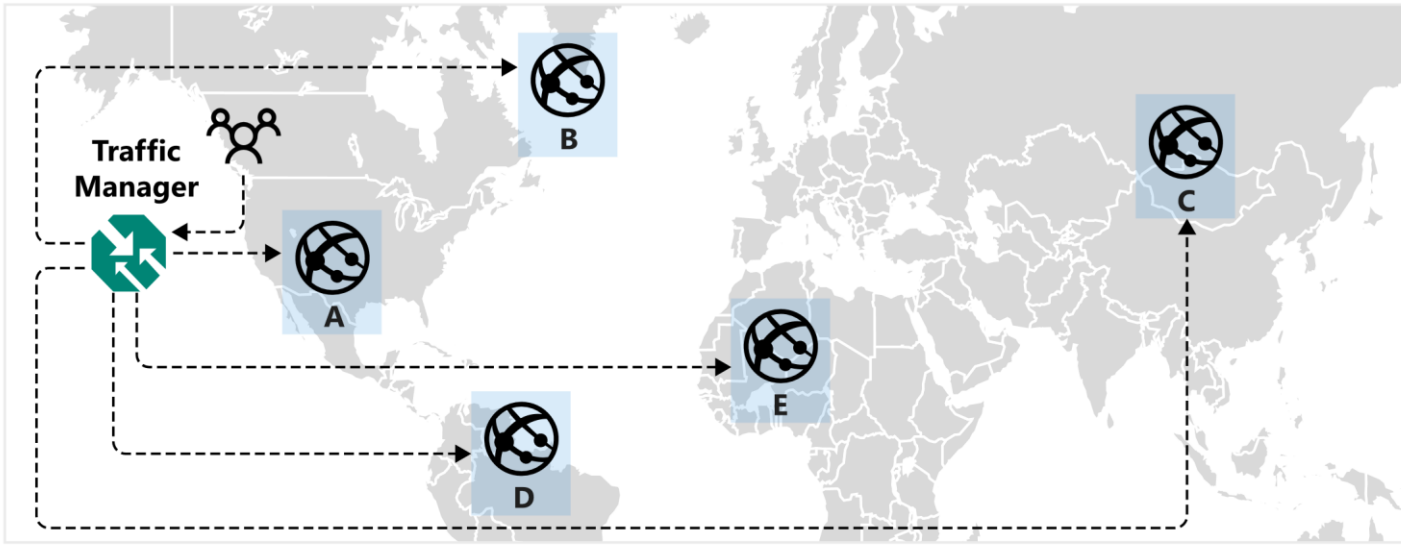
2 minutes

The music streaming app has users in different regions around the world. Some users are located far from where the application is deployed. Streaming performance is partially dependent on the distance between a user's device and the host. To offer the best possible performance to your users, you deploy your application in multiple regions. As you add regions, you'd like Traffic Manager to automatically direct the traffic to the best performing endpoint.

The *performance* traffic routing method connects users with the server that performs best for the user. It might be better performing because it's physically closer to the user, but it might also be due to congestion on internet network connectivity. Azure stores historical DNS query latency for connecting clients in an internet latency table. Azure can use this information to direct traffic to the fastest responding server, which is the server with the lowest latency. Traffic Manager maintains the internet latency table by tracking the roundtrip time between IP address ranges and each Azure datacenter. If an endpoint becomes unavailable, Traffic Manager doesn't include it in DNS query responses.

You don't have to do anything more than configure a Traffic Manager profile and select **performance** as the routing method. Endpoints don't need to be prioritized, Traffic Manager will route all the traffic automatically to the fastest responding server.

In the following example, if endpoint A stopped performing as efficiently as endpoint B, customer traffic is automatically routed to endpoint B.



Client traffic is routed consistently. A client will be directed to the same endpoint for each request it makes if nothing changes in the underlying servers and networking. If you need more granular control, for example, to choose a preferred failover within a region, you can use Traffic Manager in a nested configuration.

Next unit: Exercise - Optimize applications across regions by using performance routing

Continue

https://docs.microsoft.com/en-us/learn/modules/distribute-load-with-traffic-manager/4-performance-routing 1/2 2/21/2020 Optimize applications across regions by using performance routing - Learn | R Previous Unit 5 of 6 S Next T

100 XP

Exercise - Optimize applications across regions by using performance routing

7 minutes

This module requires a sandbox to complete. A **sandbox** gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

Activate sandbox

Your network architect would like to ensure customers have the best performance. By using the performance routing method in Traffic Manager, you can ensure that users access the location closest to them. Let's configure an instance of Traffic Manager to use performance routing.

Create a Traffic Manager profile using performance routing

1. Create a new Traffic Manager profile that is set up with performance routing.

Azure CLI = Copy

```
az network traffic-manager profile create \
--resource-group Sandbox resource group  \
--name TM-MusicStream-Performance \
--routing-method Performance \
--unique-dns-name TM-MusicStream-Performance-$RANDOM \
--output table
```

2. Create two new endpoints that that point to the public IP addresses of the virutal machines.

Azure CLI = Copy

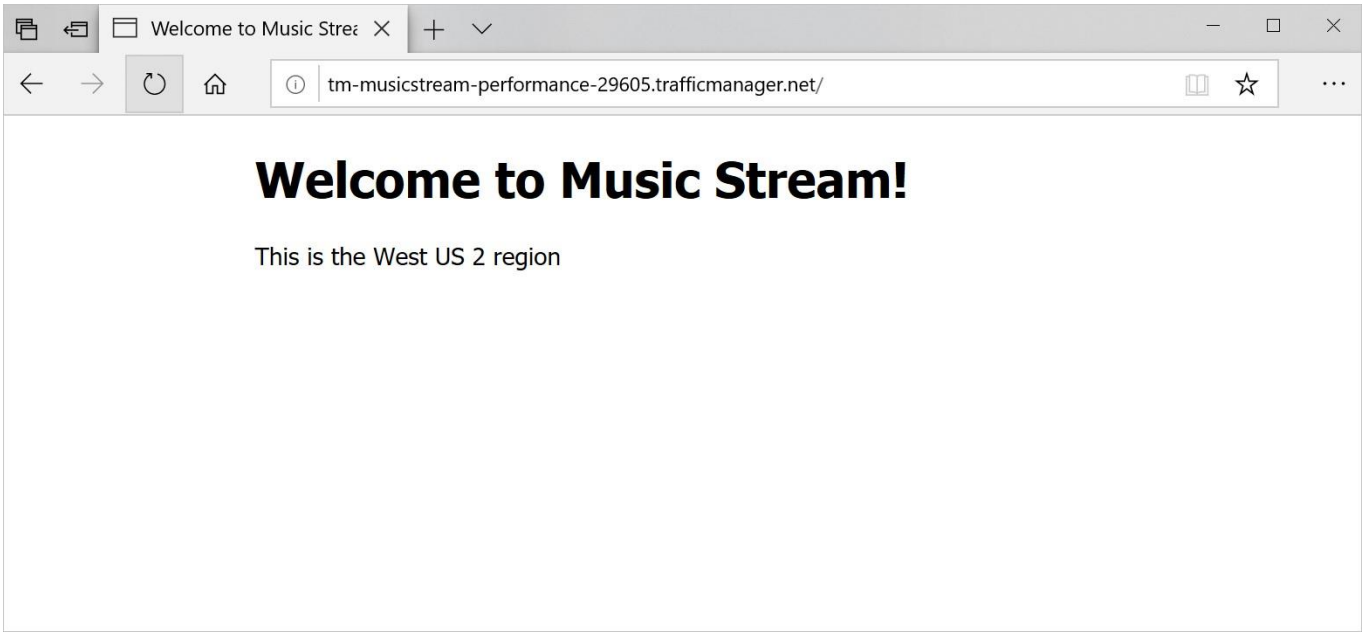
```
WestId=$(az network public-ip show \
  --resource-group Sandbox resource group  \
  --name westus2-vm-nic-pip \
  --query id \
  --out tsv)

az network traffic-manager endpoint create \    --resource-group Sandbox resource
group  \
  --profile-name TM-MusicStream-Performance \
  --name "WestUS" \
  --type azureEndpoints \
  --target-resource-id $WestId

EastId=$(az network public-ip show \
  --resource-group Sandbox resource group  \
  --name eastasia-vm-nic-pip \
  --query id \
  --out tsv)
 az network traffic-manager endpoint create \    --resource-
group Sandbox resource group  \
  --profile-name TM-MusicStream-Performance \
  --name "EastAsia" \
  --type azureEndpoints \
  --target-resource-id $EastId
```

Test the new configuration

1. Go to the Traffic Manager profiles fully qualified domain name (FQDN). Your request is routed to the endpoint that responds with the lowest latency.

```bash
echo http://$(az network traffic-manager profile show \
  --resource-group Sandbox resource group \
  --name TM-MusicStream-Performance \
  --query dnsConfig.fqdn \
  --output tsv)
```

2. Depending on where you're located, you'll be directed to the best performing endpoint.



3. Use `nslookup` to resolve the Traffic Manager profile domain name.

```bash
nslookup $(az network traffic-manager profile show \
    --resource-group Sandbox resource group \
    --name TM-MusicStream-Performance \
    --query dnsConfig.fqdn \
    --output tsv)
```

The `nslookup` command returns where the domain name resolves. For example, if you're closest to Europe, it returns the following.

```output
Non-authoritative answer:
tm-musicstream-performance-29605.trafficmanager.net  canonical name = westus2-vm-
rmzkcs3kmupuq.westus2.cloudapp.azure.com.
Name:   westus2-vm-rmzkcs3kmupuq.westus2.cloudapp.azure.com
Address: 13.66.168.61
```

If your customers have two endpoints that have equal network latency, they might be routed to either endpoint. Refresh the web page to see i you are served the same endpoint.

Next unit: Summary

Continue

Azure Cloud Shell

This module requires a sandbox to complete. A sandbox gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

2/21/2020 Summary - Learn | Microsoft Docs

R Previous

Unit 6 of 6 S

100 XP

Summary

2 minutes

In this module, you learned how to use Traffic Manager as a DNS load balancer to distribute traffic to different web apps deployed in differer regions to:

- Improve performance by serving requests from the closest deployment.
- Direct traffic to specific endpoints through priority routing.
- Improve application resilience by detecting failures and routing customers to available endpoints.
- Perform maintenance or updates without any downtime.
- Serve requests from specific geographic locations from specific deployments.

You can now use Traffic manager to improve your availability and global distribution of your application, which improves customer experience.

Learn more

Visit the following links to learn more about some of the subjects we explored in this module:

- What is Traffic Manager?

Module complete:

Unlock achievement

800 XP

Improve application scalability and resiliency by using Azure Load Balancer

47 min • Module • 6 Units

V V V V V 4.7 (462) Rate it

Beginner Solutions Architect Administrator Azure Load Balancer Virtual Network

Discuss the different load balancers in Azure and how to choose the right Azure load balancer solution to meet your requirements.

In this module, you will:

- Identify the features and capabilities of Azure Load Balancer
- Deploy and configure an Azure load balancer

Start

Prerequisites

- Basic knowledge of networking concepts
- Basic knowledge of Azure virtual machines

This module is part of these learning paths
Architect network infrastructure in Azure

Introduction
5 min

Azure Load Balancer features and capabilities
10 min

Configure a public load balancer
10 min

Exercise - Configure a public load balancer
10 min

Internal load balancer
10 min

Summary
2 min

https://docs.microsoft.com/en-us/learn/modules/improve-app-scalability-resiliency-with-load-balancer/ 1/1

2/21/2020 Introduction - Learn | Microsoft Docs

Unit 1 of 6 Next

100 XP

Introduction

5 minutes

Many apps need to be resilient to failure and scale easily when demand increases. You can address those needs by using Azure Load Balancer.

Suppose you work for a healthcare organization that's launching a new portal application in which patients can schedule appointments. The application has a patient portal and web application front end and a business tier database. The database is used by the front end to retrieve and save patient information.

The new portal needs to be available around the clock to handle failures. The portal must adjust to fluctuations in load by adding and removing resources to match the load. The organization needs a solution that distributes work to virtual machines across the system as virtual machines are added. The solution should detect failures and reroute jobs to virtual machines as needed. Improved resiliency and scalability helps ensure that patients can schedule appointments from any location.

By the end of this module, you will be able to use Azure Load Balancer to build a resilient and scalable app architecture.

Learning objectives

In this module, you will:

- Identify the features and capabilities of Azure Load Balancer.
- Deploy and configure an instance of Azure Load Balancer.

Prerequisites

- Basic knowledge of networking concepts.
- Basic knowledge of Azure virtual machines.
- Familiarity with the Azure portal.

https://docs.microsoft.com/en-us/learn/modules/improve-app-scalability-resiliency-with-load-balancer/1-introduction1/2 2/21/2020 Introduction - Learn | Microsoft Doc 2/2

R Previous Unit 2 of 6 S Next T

200 XP

Azure Load Balancer features and capabilities

10 minutes

With Azure Load Balancer, you can spread user requests across multiple virtual machines or other services. That way, you can scale the app to larger sizes than a single virtual machine can support, and you ensure that users get service, even when a virtual machine fails.

In your healthcare organization, you can expect large user demand. It's of vital importance to each user that they can book an appointment, even at times of peak demand or when virtual machines fail. If you use multiple virtual servers for your front end and a load balancer to distribute traffic between them, you achieve a high capacity because all the virtual servers collaborate to satisfy requests. You also improve resilience because the load balancer can automatically route traffic away when a virtual server fails.

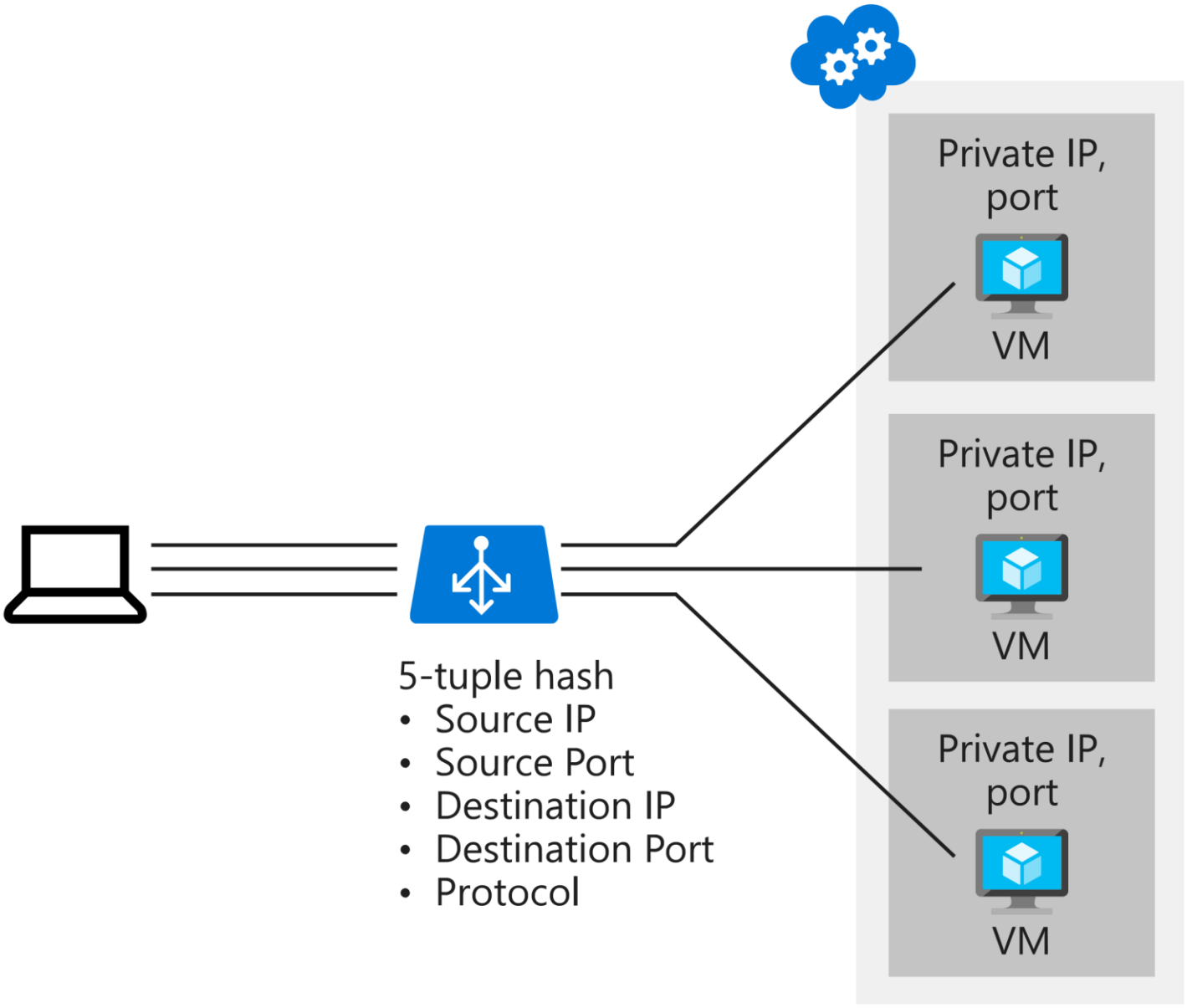
Here, you will learn how the features of Load Balancer help you create robust app architectures.

Distribute traffic with Azure Load Balancer

Azure Load Balancer is a service you can use to distribute traffic across multiple virtual machines. Use Load Balancer to scale applications and create high availability for your virtual machines and services. Load balancers use a hash-based distribution algorithm. By default, a five-tuple hash is used to map traffic to available servers. The hash is made from the following elements:

- **Source IP**: The IP address of the requesting client.
- **Source port**: The port of the requesting client.
- **Destination IP**: The destination IP of the request.
- **Destination port**: The destination port of the request.

Protocol type: The specified protocol type. For example, HTTP, HTTPS, HTTP/2.



Load Balancer supports inbound and outbound scenarios, provides low latency and high throughput, and scales up to millions of flows for all TCP and UDP applications.

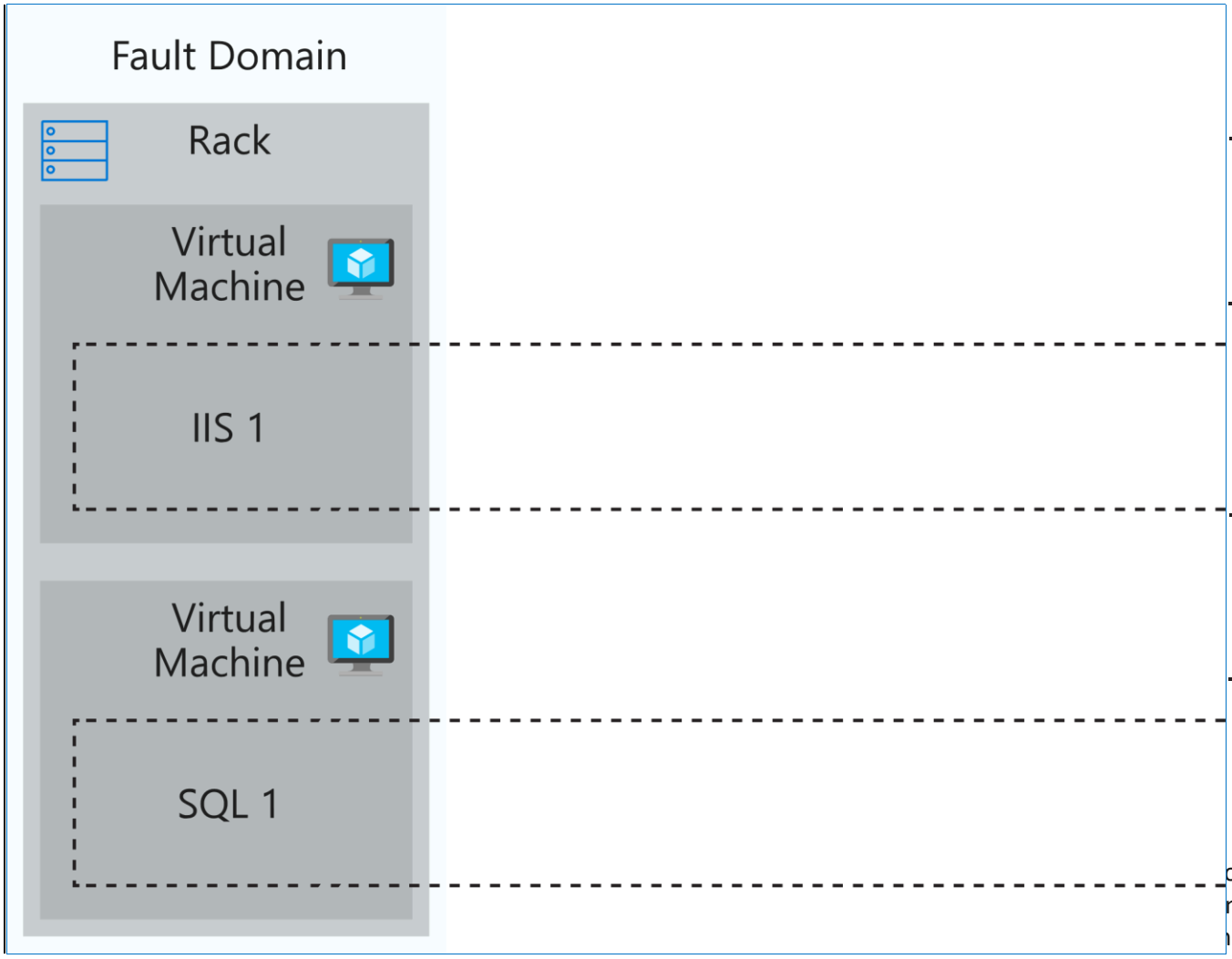
Load balancers aren't physical instances. Load balancer objects are used to express how Azure configures its infrastructure to meet your requirements.

To achieve high availability with Load Balancer, you can choose to use availability sets and availability zones to ensure that virtual machines are always available:

Configuration	Service level agreement (SLA)	Information
Availability set	99.95%	Protection from hardware failures within datacenters
Availability zone	99.99%	Protection from entire datacenter failure

Availability sets

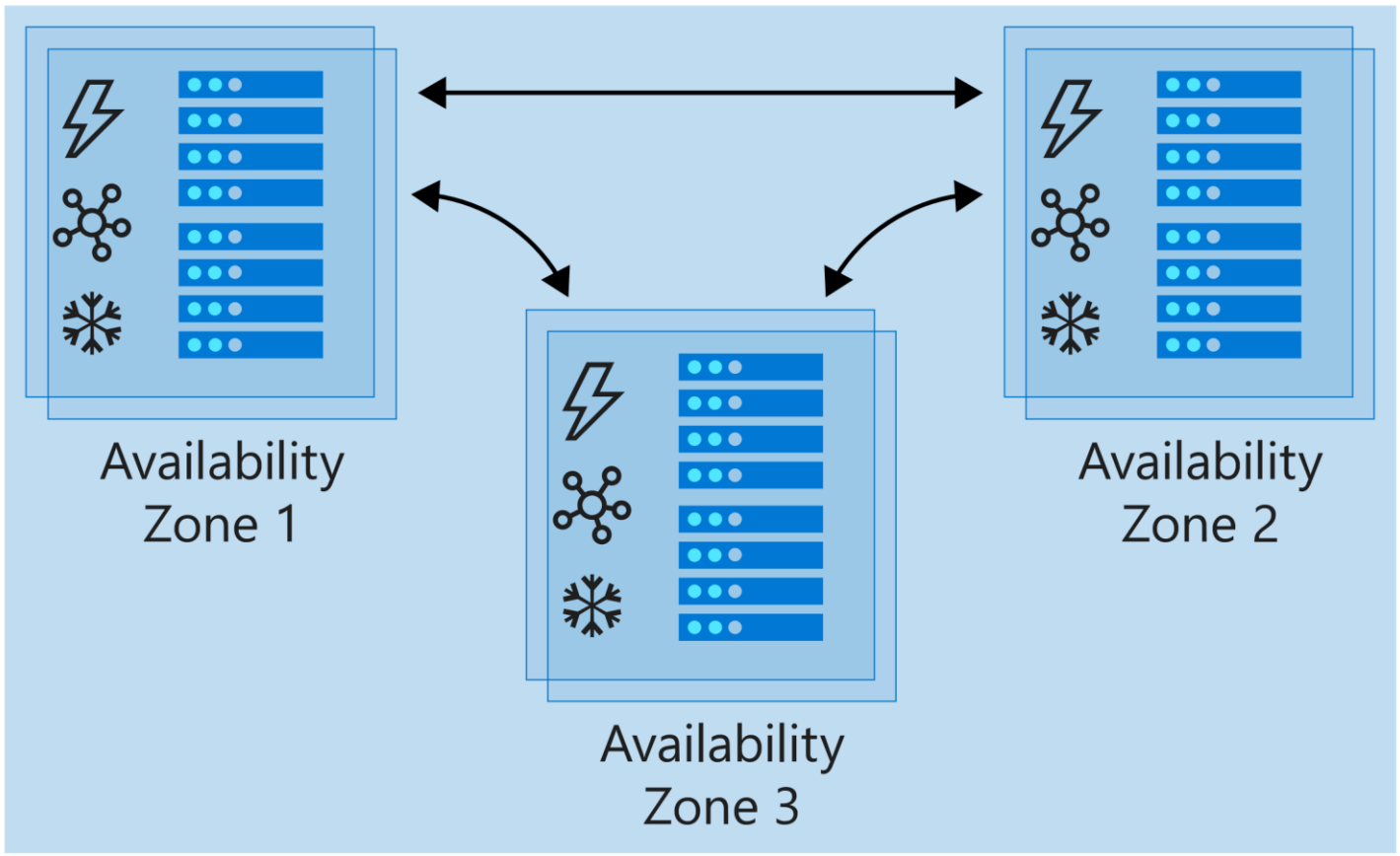
An availability set is a logical grouping that you use to isolate virtual machine resources from each other when they're deployed. Azure ensures that the virtual machines you put in an availability set run across multiple physical servers, compute racks, storage units, and network switches. If there's a hardware or software failure, only a subset of your virtual machines is affected. Your overall solution stays operational. Availability sets are essential for building reliable cloud solutions.



or more d
ne are pla
at, when

Availability zones

entire datacenter fails, you can continue to serve users.



Availability zones don't support all virtual machine sizes and aren't available in all Azure regions. Check that they are supported in your region before you use them in your architecture.

Select the right Load Balancer product

Two products are available when you create a load balancer in Azure: basic load balancers and standard load balancers.

Basic load balancers allow:

- Port forwarding
- Automatic reconfiguration
- Health probes
- Outbound connections through source network address translation (SNAT)
- Diagnostics through Azure Log Analytics for public-facing load balancers

Basic load balancers can be used only with availability sets.

Standard load balancers support all of the basic features. They also allow:

- HTTPS health probes
- Availability zones
- Diagnostics through Azure Monitor, for multidimensional metrics
- High availability (HA) ports
- Outbound rules
- A guaranteed SLA (99.99% for two or more virtual machines)

Internal and external load balancers

An external load balancer operates by distributing client traffic across multiple virtual machines. An external load balancer permits traffic from the internet. The traffic might come from browsers, module apps, or other sources. In a healthcare organization, the balancer distributes the load

of all the browsers that run the client healthcare application.

An internal load balancer distributes a load from internal Azure resources to other Azure resources. For example, if you have front-end web servers that need to call business logic that's hosted on multiple middle-tier servers, you can distribute that load evenly by using an internal load balancer. No traffic is allowed from internet sources. In a healthcare organization, the load balancer distributes a load across the internal application tier.

Check your knowledge

1. What is the default distribution type for traffic through a load balancer?

- Source IP affinity
- Five-tuple hash

  Five-tuple hash is the default.
- Three-tuple hash

2. What is the main advantage of an availability set?

- It allows virtual machines to be available across datacenter failures.
- It allows virtual machines to be available across physical server failures.

  Availabilty sets allow virtual machines to remain available when a physical server fails.
- It allows virtual machines to be grouped into logical categories.

Next unit: Configure a public load balancer

Continue

Configure a public load balancer

10 minutes

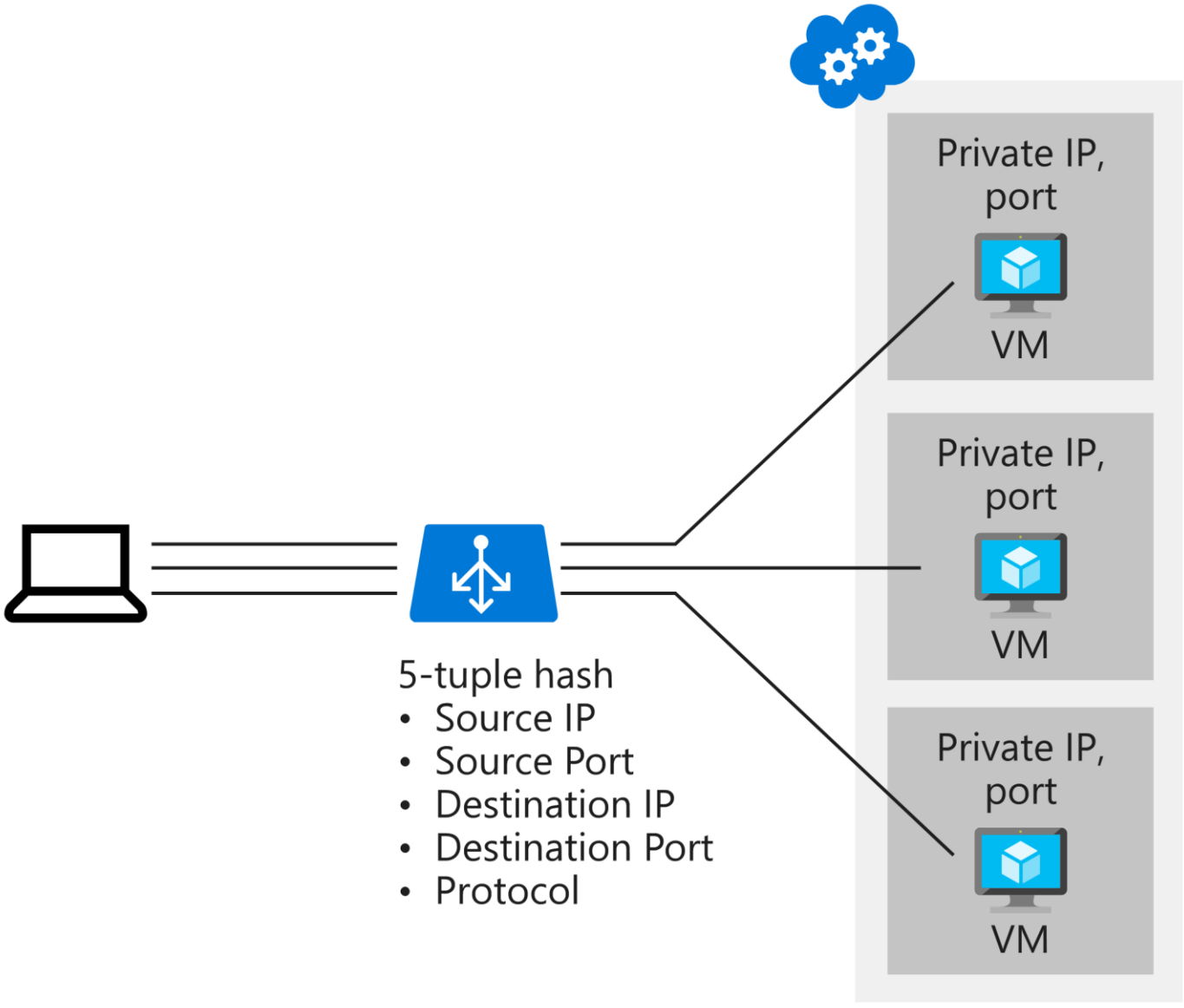
As the solution architect for the healthcare portal, you need to distribute the load from the client browsers over the virtual machines in your web farm. You need to set up a load balancer and configure the virtual machines to be balanced.

A public load balancer maps the public IP address and port number of incoming traffic to the private IP address and port number of a virtual machine in the back-end pool. The responses are then returned to the client. By applying load-balancing rules, you distribute specific types of traffic across multiple virtual machines or services.

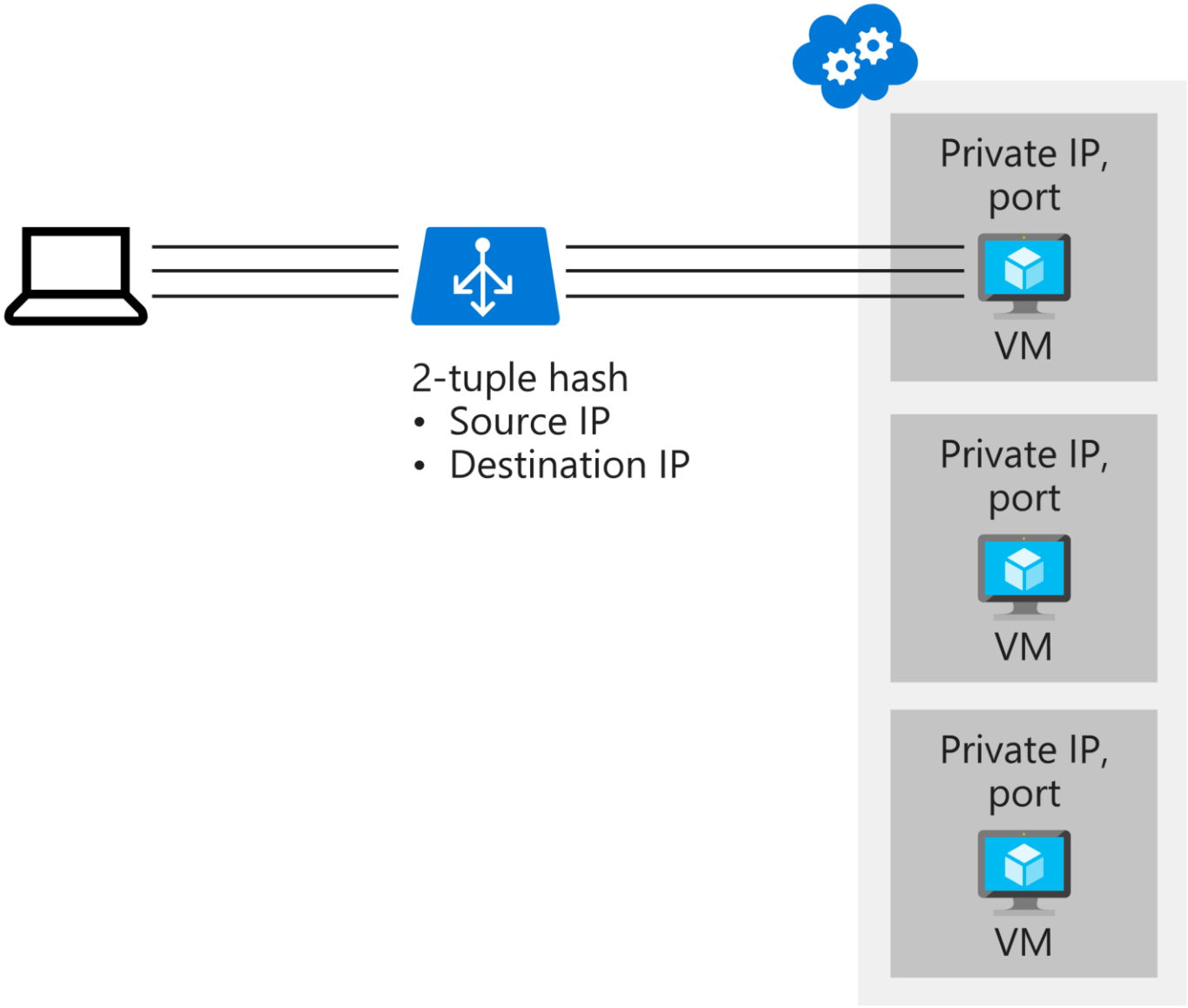
Distribution modes

By default, Azure Load Balancer distributes network traffic equally among virtual machine instances. The following distribution modes are also possible if a different behavior is required:

- **Five-tuple hash**. The default distribution mode for Load Balancer is a five-tuple hash. The tuple is composed of the source IP, source port, destination IP, destination port, and protocol type. Because the source port is included in the hash and the source port changes for each session, clients might be directed to a different virtual machine for each session.



- **Source IP affinity**. This distribution mode is also known as *session affinity* or *client IP affinity*. To map traffic to the available servers, the mode uses a two-tuple hash (from the source IP address and destination IP address) or three-tuple hash (from the source IP address, destination IP address, and protocol type). The hash ensures that requests from a specific client are always sent to the same virtual machine behind the load balancer.



Choose a distribution mode

In the healthcare portal example, imagine that a developer requirement of the presentation tier is to use in-memory sessions to store the logged user's profile as the user interacts with the portal.

In this scenario, the load balancer must provide source IP affinity to maintain a user's session. The profile is stored only on the virtual machine that the client first connects to because that IP address is directed to the same server. When you create the load balancer endpoint, you must specify the distribution mode by using the following PowerShell example:

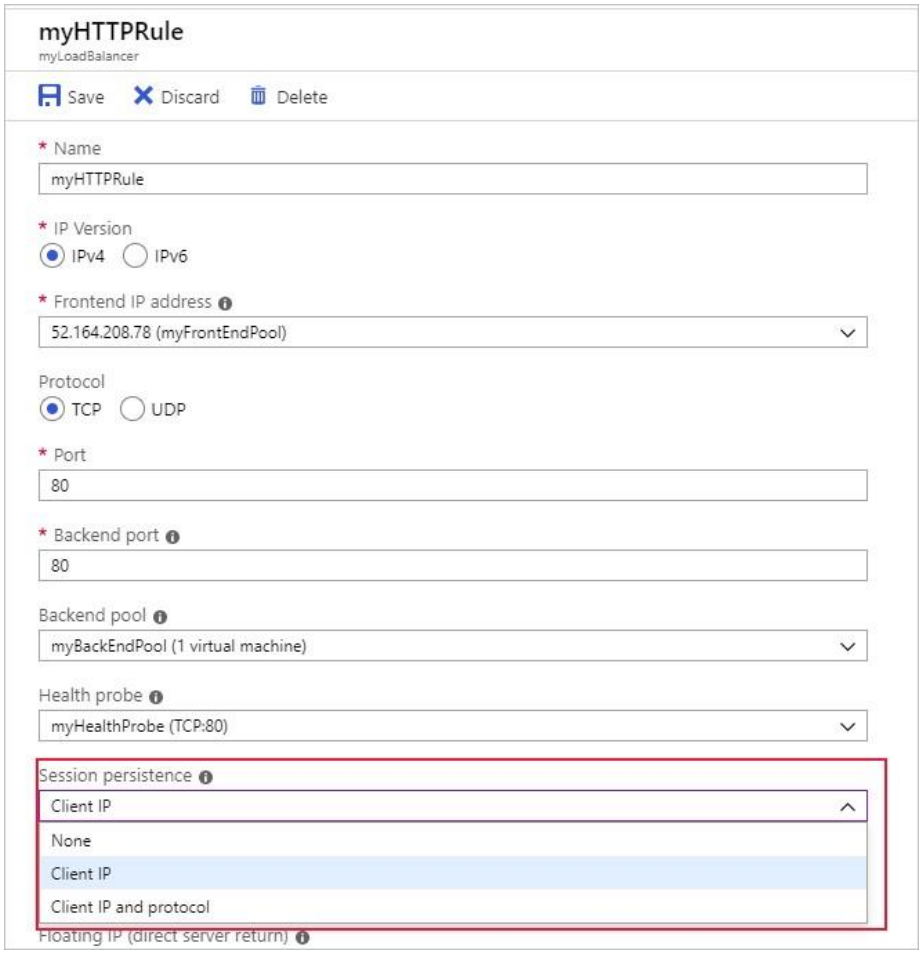
PowerShell Copy

```
$lb = Get-AzLoadBalancer -Name MyLb -ResourceGroupName MyResourceGroup
$lb.LoadBalancingRules[0].LoadDistribution = 'sourceIp'
Set-AzLoadBalancer -LoadBalancer $lb
```

To add session persistence through the Azure portal:

1. In the Azure portal, open the load balancer resource.

2. Edit the relevant line of the**Load-balancing rules**
3. Change the value for**Session persistence**to **Client IP**.



Load Balancer and Remote Desktop Gateway

Remote Desktop Gateway is a Windows service that you can use to enable clients on the internet to make Remote Desktop Protocol (RDP) connections through firewalls to Remote Desktop servers on your private network. The default five-tuple hash in Load Balancer is incompatible with this service. If you want to use Load Balancer with your Remote Desktop servers, use source IP affinity.

Load Balancer and media upload

Another use case for source IP affinity is media upload. In many implementations, a client initiates a session through a TCP protocol and connects to a destination IP address. This connection remains open throughout the upload to monitor progress, but the file is uploaded through a separate UDP protocol.

With the five-tuple hash, the load balancer likely will send the TCP and UDP connections to different destination IP addresses and the upload won't finish successfully. Use source IP affinity to resolve this issue.

Next unit: Exercise - Configure a public load balancer

Continue

Exercise - Configure a public load balancer

10 minutes

This module requires a sandbox to completeA **sandbox** gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

Activate sandbox

Choose your shell

Bash | PowerShell | Azure portal

You can configure Azure Load Balancer by using the Azure portal, PowerShell, or the Azure CLI.

In your healthcare organization, you want to load-balance client traffic to provide a consistent response based on the health of the patient portal web servers. You have two virtual machines in an availability set to act as your healthcare portal web application.

Here, you will create a load balancer resource and use it to distribute a load across the virtual machines.

Deploy the patient portal web application

First, deploy your patient portal application across two virtual machines in a single availability set. To save time, let's start by running a script to create this application. The script will:

- Create a virtual network and network infrastructure for the virtual machines.
- Create two virtual machines in this virtual network.

To deploy the patient portal web application:

1. Run the following `git clone` command in Azure Cloud Shell. The command clones the repo that contains the source for the app and runs the setup script from GitHub. You then change to the directory of the cloned repo.

```
bash
```

Copy

```
git clone https://github.com/MicrosoftDocs/mslearn-improve-app-scalability-resiliency-with-load-balancer.git
cd mslearn-improve-app-scalability-resiliency-with-load-balancer
```

2. As its name suggests, this script generates two virtual machines in a single availability set. The script takes about two minutes to run.

```
bash
```

Copy

```
bash create-high-availability-vm-with-sets.sh [ sandbox resource group name ]
```

3. When the script finishes, on the Azure portal menu or from the **Home** page, select **Resource groups**, then select the **[sandbox resource group name]** resource group. Review the resources that were created by the script.

Create a load balancer

Let's use the Azure CLI to create the load balancer and its associated resources.

1. Create a new public IP address.

Azure = Copy

```
az network public-ip create \
  --resource-group [sandbox resource group name] \
  --allocation-method Static \
  --name myPublicIP
```

2. Create the load balancer.

Azure CLI = Copy

```
az network lb create \
  --resource-group [sandbox resource group name] \
  --name myLoadBalancer \
  --public-ip-address myPublicIP \
  --frontend-ip-name myFrontEndPool \
  --backend-pool-name myBackEndPool
```

3. To allow the load balancer to monitor the status of the healthcare portal, create a health probe. The health probe dynamically adds or removes virtual machines from the load balancer rotation based on their response to health checks.

Azure CLI = Copy

```
az network lb probe create \
  --resource-group [sandbox resource group name] \
  --lb-name myLoadBalancer \
  --name myHealthProbe \
  --protocol tcp \
  --port 80
```

4. Now you need a load balancer rule that's used to define how traffic is distributed to the virtual machines. You define the front-end IP configuration for the incoming traffic and the back-end IP pool to receive the traffic, along with the required source and destination port. To make sure only healthy virtual machines receive traffic, you also define the health probe to use.

Azure CLI = Copy

```
az network lb rule create \
  --resource-group [sandbox resource group name] \
  --lb-name myLoadBalancer \
  --name myHTTPRule \
  --protocol tcp \
  --frontend-port 80 \
  --backend-port 80 \
  --frontend-ip-name myFrontEndPool \   --backend-pool-name myBackEndPool
\
  --probe-name myHealthProbe
```

5. Connect the virtual machines to the back-end pool by updating the network interfaces you created in the script to use the back-end pool information.

Azure CLI = Copy

```
az network nic ip-config update \
  --resource-group [sandbox resource group name] \
  --nic-name webNic1 \
  --name ipconfig1 \
  --lb-name myLoadBalancer \
  --lb-address-pools myBackEndPool
 az network nic ip-config update \
  --resource-group [sandbox resource group name] \
  --nic-name webNic2 \
  --name ipconfig1 \
  --lb-name myLoadBalancer \
  --lb-address-pools myBackEndPool
```

6. Run the following command to get the public IP address of the load balancer and the URL for your website.

Azure CLI = Copy

```
echo http://$(az network public-ip show \
        --resource-group [sandbox resource group name] \
        --name myPublicIP \
        --query ipAddress \
        --output tsv)
```

Test the load balancer configuration

Let's test the load balancer setup to show how it can handle availability and health issues dynamically.

1. In a new browser tab, go to the public IP address that you noted. You'll see that the response is returned from one of the virtual machines.
2. Try a "force refresh" by pressing Ctrl+F5 a few times to see that the response is returned randomly from both virtual machines.
3. On the Azure portal M menu or from the **Home** page, select **All resources**. Then select **webVM1** > **Stop**.
4. Return to the tab that shows the website and force a refresh of the webpage. All requests are returned from **webVM2**.

Next unit: Internal load balancer

Continue T

English (United States)

Azure Cloud Shell

This module requires a sandbox to complete. A sandbox gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent R Previous

200 XP

Internal load balancer

10 minutes

In addition to balancing requests from users to front-end servers, you can use Azure Load Balancer to distribute traffic from front-end servers evenly among back-end servers.

In your healthcare organization, front-end servers call business logic that's hosted on a middle tier. You want to ensure that the middle tier is as scalable and resilient as the front end. You want to use a load balancer to distribute requests from the front-end servers evenly among the middle-tier servers. This way, you'll scale out the middle-tier servers to achieve the highest capacity possible. You'll also ensure that the middle tier is resilient to failure. When a server fails, the load balancer automatically reroutes traffic.

Here, you'll learn how to use load balancers to distribute internal traffic.

Configure an internal load balancer

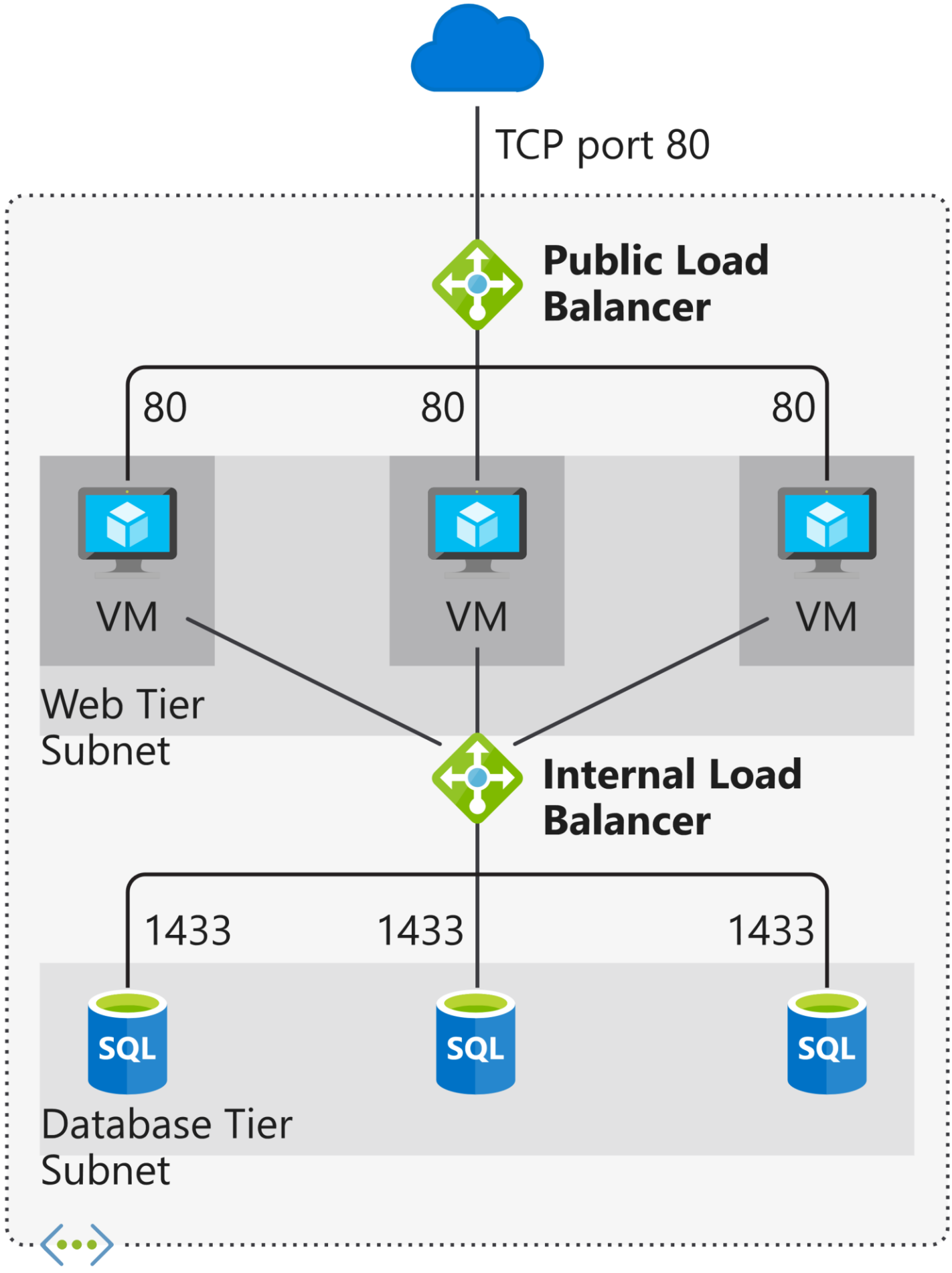
In the healthcare portal scenario, a web tier handles requests from users. The web tier connects to databases to retrieve data for users. The database tier is also deployed on two virtual machines. To allow the front-end web portal to continue to serve client requests if a database server fails, you can set up an internal load balancer to distribute traffic to the database servers.

You can configure an internal load balancer in almost the same way as an external load balancer, but with these differences:

- When you create the load balancer, for the **Type** value, select **Internal**. When you select this setting, the front-end IP address of the load balancer isn't exposed to the internet.
- Assign a private IP address instead of a public IP address for the front end of the load balancer.

Place the load balancer in the protected virtual network that contains the virtual machines you want to handle the requests.

The internal load balancer should be visible only to the web tier. All the virtual machines that host the databases are in one subnet. You can use an internal load balancer to distribute traffic to those virtual machines.


TCP port 80
Public Load Balancer
80
80
80
VM
VM
VM
Web Tier Subnet
Internal Load Balancer
1433
1433
1433
SQL
SQL
SQL
Database Tier Subnet

Choose the distribution mode

In the healthcare portal, the application tier is stateless, so you don't need to use source IP affinity. You can use the default distribution mode of a five-tuple hash. This mode offers the greatest scalability and resilience. The load balancer routes traffic to any healthy server.

Check your knowledge

1. Which configuration is required to configure an internal load balancer?

- [x] Virtual machines should be in the same virtual network.

The virtual machines that you use a load balancer to distribute a load to must be in the same virtual network.

- [ ] Virtual machines must be publicly accessible.
- [ ] Virtual machines must be in an availability set.

2. Which of the following statement about external load balancers is correct?

- [ ] They have a private, front-facing IP address.
- [ ] They don't have a listener IP address.
- [x] They have a public IP address.

External load balancers have public IP addresses.

Next unit: Summary

Continue

100 XP

Summary

2 minutes

In this module, you learned about Azure Load Balancer and how you can use Load Balancer to minimize the effect of failures and increase resilience and stability. You used this knowledge to create a resilient healthcare portal that's capable of adapting to meet the application requirements of session affinity. You learned how to group virtual machines behind a load balancer to increase availability. By implementing load balancer in the healthcare portal scenario, you learned about the differences between an internal and an external load balancer. You als discovered how a load balancer can be configured to provide availability across datacenters by using availability zones.

Clean up

The sandbox automatically cleans up your resources when you're finished with this module.

When you're working in your own subscription, it's a good idea at the end of a project to identify whether you still need the resources you created. Resources left running can cost you money. You can delete resources individually or delete the resource group to delete the entire s resources.

Learn more

- Azure Load Balancer
- What is Azure Load Balancer?
- Tutorial: Load-balance internet traffic to VMs by using the Azure porta
- Tutorial: Balance internal traffic load with a basic load balancer in the Azure porta
- What are availability zones in Azure?

Module complete:

Unlock achievement

https://docs.microsoft.com/en-us/learn/modules/improve-app-scalability-resiliency-with-load-balancer/6-summary 1/2

2/21/2020 Load balance your web service traffic with Application Gateway - Learn | Microsoft Docs

800 XP

Load balance your web service traffic with Application Gateway

1 hr 32 min • Module • 7 Units

V V V V W 4.6 (583) Rate it

Beginner Solutions Architect Administrator Azure Virtual Network Application Gateway

Improve application resilience by distributing load across multiple servers and use path-based routing to direct web traffic.

In this module, you will:

Identify the load balancing capabilities of Application Gateway
Create an Application Gateway and configure load balancing
Configure an Application Gateway to use URL path-based routing

- Knowledge of basic networking concepts
- Familiarity with Azure virtual machines and Azure App Service
- Familiarity with Azure virtual networking

This module is part of these learning paths
Architect network infrastructure in Azure

Introduction
5 min

Route traffic with Application Gateway
10 min

Exercise - Create web sites
10 min

Application Gateway creation and configuration
10 min

Exercise - Create and configure an Application Gateway
45 min

Exercise - Test your Application Gateway
10 min

Summary
2 min

https://docs.microsoft.com/en-us/learn/modules/load-balance-web-traffic-with-application-gateway/ 1/1

2/21/2020 Introduction - Learn | Microsoft Docs

Introduction

5 minutes

Imagine that you work for the motor vehicle department of a governmental organization. The department runs several public web sites that enable drivers to register their vehicles and renew their drivers license online. The vehicle registration web site has been running on a single server and has suffered multiple outages because of server failures. This has resulted in frustrated drivers who are trying to register their vehi at the end of the month before their registrations expire. The department would like to improve resiliency by adding multiple web servers to their site and distribute the load across them. They would also like to centralize their site on a single load-balancing service to simplify the U for site visitors.

Learning objectives

In this module, you will:

- Identify the load-balancing capabilities of Application Gateway
- Create an Application Gateway and configure load balancing
- Configure an Application Gateway to use URL path-based routing

Prerequisites

- Knowledge of basic networking concepts
- Familiarity with Azure virtual machines and Azure App Service
- Familiarity with Azure virtual networking

Next unit: Route traffic with Application Gateway

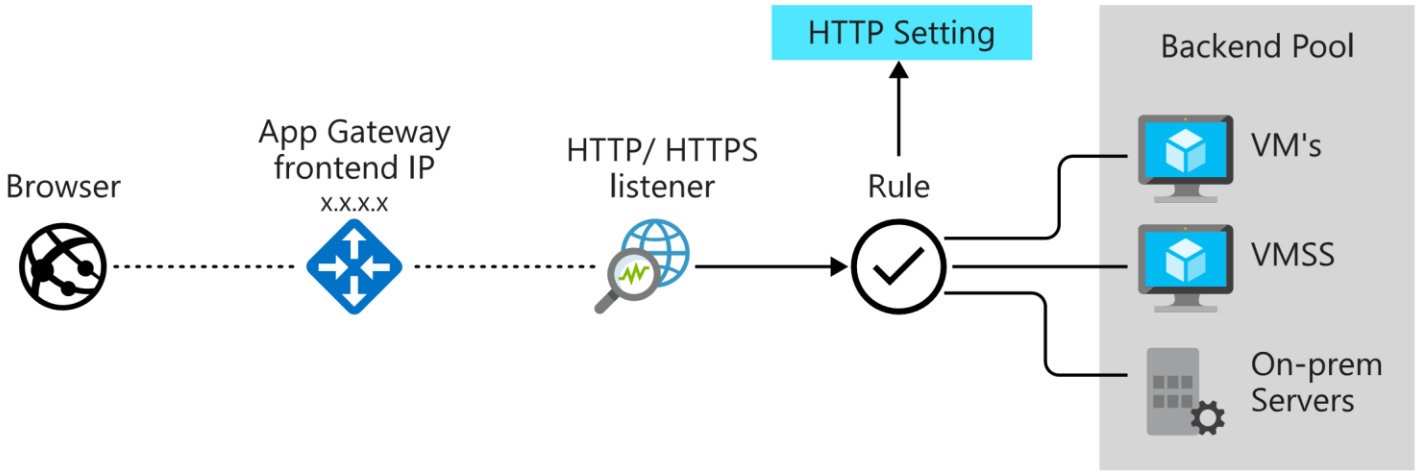
Continue

200 XP

Route traffic with Application Gateway

10 minutes

Application Gateway manages the requests that client applications can send to a web app. Application Gateway routes traffic to a pool of web servers based on the URL of a request. This is known as *application layer routing*. The pool of web servers can be Azure virtual machines, Azure virtual machine scale sets, Azure App Service, and even on-premises servers.



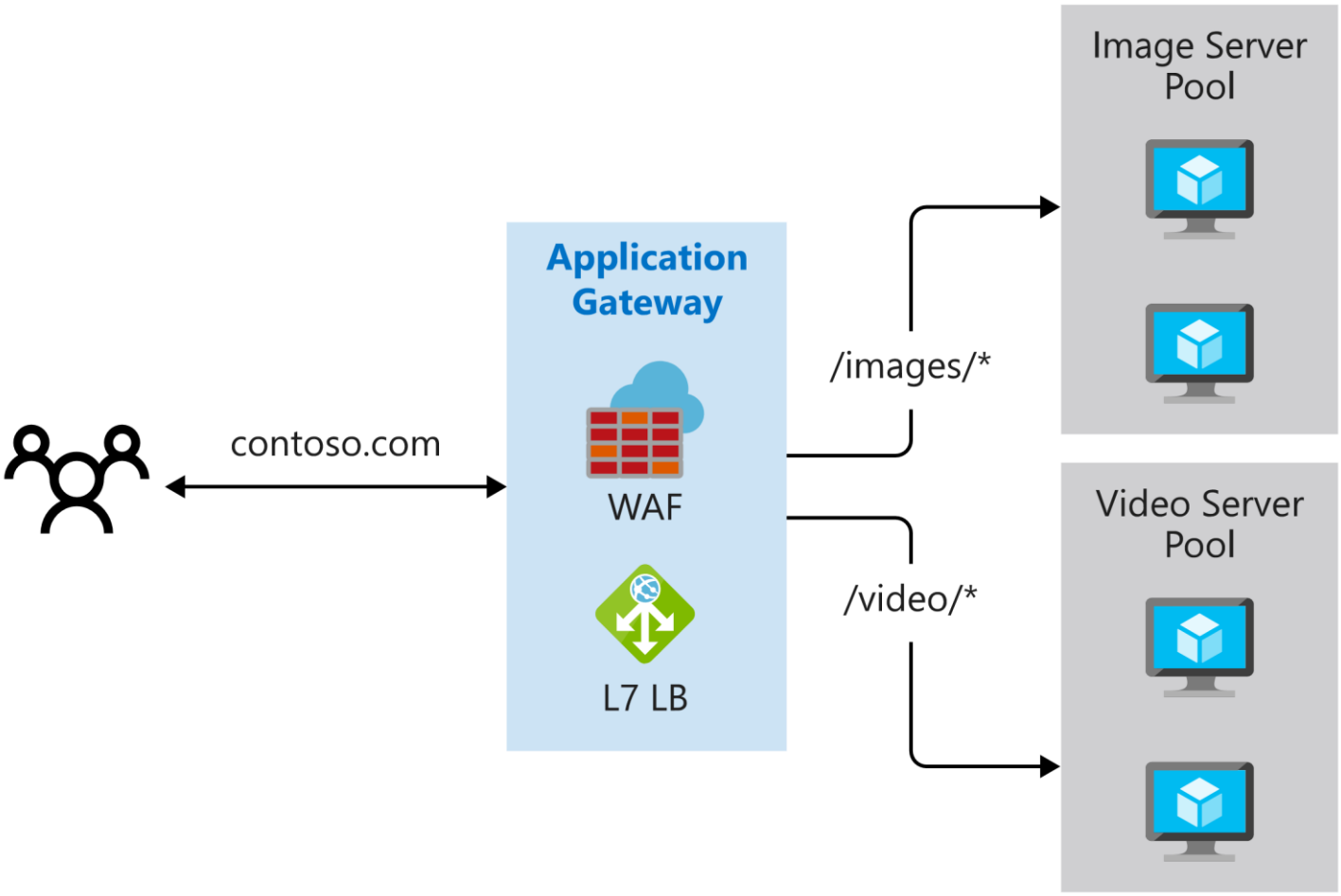
How Application Gateway routes requests

Clients send requests to your web apps to the IP address or DNS name of the gateway. The gateway routes requests to a selected web server in the back-end pool, using a set of rules configured for the gateway to determine where the request should go.

There are two primary methods of routing traffic, path-based routing and multiple site hosting. Let's take a look at the capabilities for each.

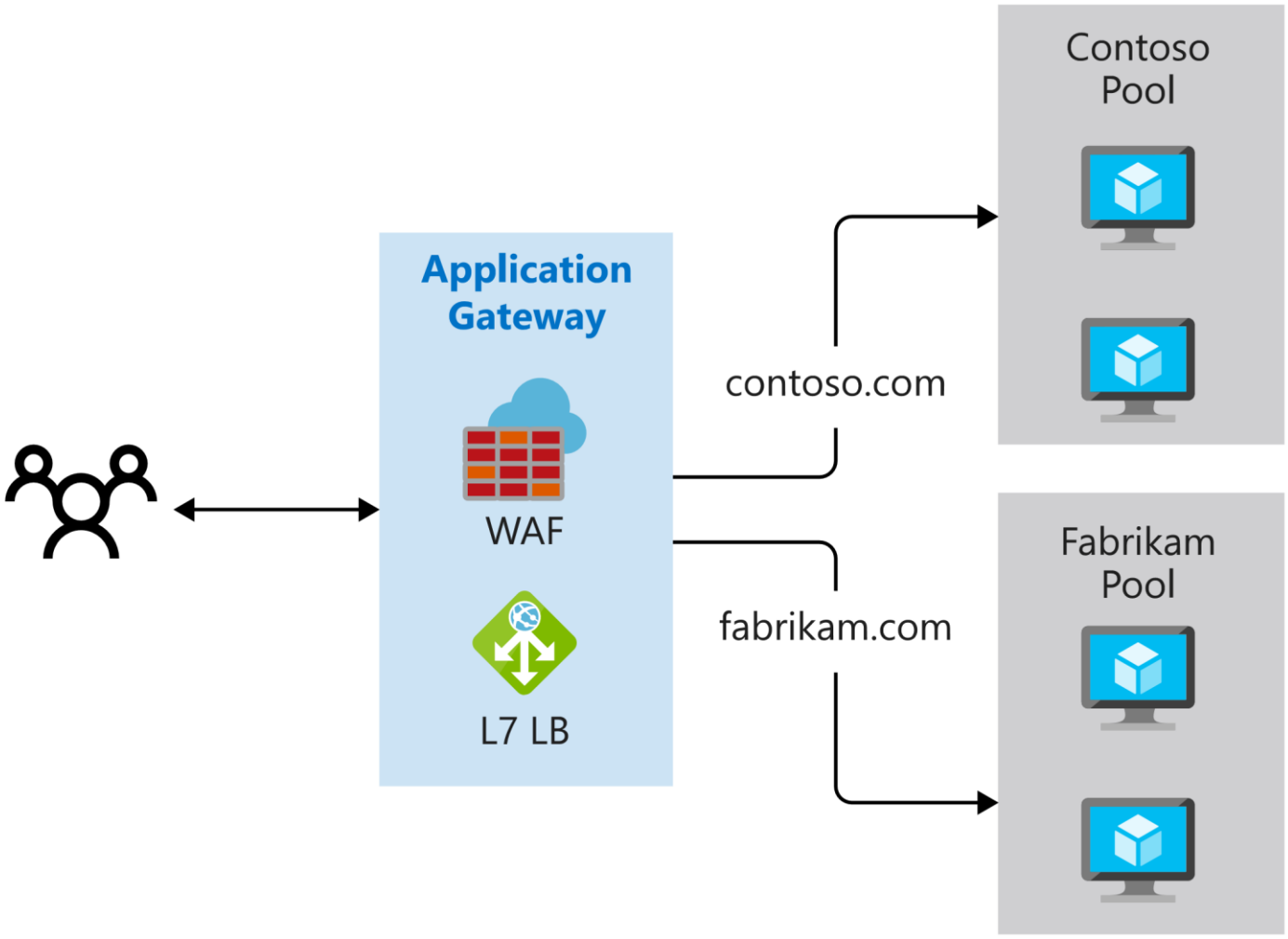
Path-based routing

Path-based routing enables you to send requests with different paths in the URL to a different pool of back-end servers. For example, you could direct requests with the path `/video/*` to a back-end pool containing servers that are optimized to handle video streaming, and direct `/images/*` requests to a pool of servers that handle image retrieval.



Multiple site hosting

Multiple site hosting enables you to configure more than one web application on the same application gateway instance. In a multi-site configuration, you register multiple DNS names (CNAMEs) for the IP address of the Application Gateway, specifying the name of each site. Application Gateway uses separate listeners to wait for requests for each site. Each listener passes the request to a different rule, which can route the requests to servers in a different back-end pool. For example, you could configure Application Gateway to direct all requests for `http://contoso.com` to servers in one back-end pool, and requests for `http://fabrikam.com` to another back-end pool. The following diagram shows this configuration.



Multi-site configurations are useful for supporting multi-tenant applications, where each tenant has its own set of virtual machines or other resources hosting a web application.

Other routing capabilities

Along with path-based routing and multiple site hosting, there are a few additional capabilities when routing with Application Gateway.

- **Redirection** - Redirection can be used to another site, or from HTTP to HTTPS.
- **Rewrite HTTP headers** - HTTP headers allow the client and server to pass additional information with the request or the response.
- **Custom error pages** - Application Gateway allows you to create custom error pages instead of displaying default error pages. You can use your own branding and layout using a custom error page.

Load balancing in Application Gateway

Application Gateway will automatically load balance requests sent to the servers in each back-end pool using a round-robin mechanism. However, you can configure session stickiness, if you need to ensure that all requests for a client in the same session are routed to the same server in a back-end pool.

Load-balancing works with the OSI Layer 7 routing implemented by Application Gateway routing, which means that it load balances requests based on the routing parameters (host names and paths) used by the Application Gateway rules. In comparison, other load balancers, such as Azure Load Balancer, function at the OSI Layer 4 level, and distribute traffic based on the IP address of the target of a request.

Operating at OSI Layer 7 enables load balancing to take advantage of the other features that Application Gateway provides. These features include:

- Support for the HTTP, HTTPS, HTTP/2 and WebSocket protocols.
- A web application firewall to protect against web application vulnerabilities.
- End-to-end request encryption.

- Autoscaling, to dynamically adjust capacity as your web traffic load changes.

Routing for the motor vehicle department

Revisiting our scenario at the motor vehicle department, Application Gateway can be used to address both issues. We can use the load balancing and health probe capabilities to ensure that failures are handled without user impact. We can also use path-based routing to provide a single endpoint for users to access sites hosted across disparate web services.

Let's take a closer look at how we can do this.

Check your knowledge

1. Which criteria does Application Gateway use to route requests to a web server?

- The IP address of the web server that is the target of the request
- The region in which the servers hosting the web application are located.
- The hostname, port, and path in the URL of the request

Correct!

2. Which load balancing strategy does Application Gateway implement?

- Application Gateway selects the server in the backend pool that currently has the lightest load.
- Application Gateway polls each server in the backend pool in turn, and sends the request to the first server that responds.
- Application Gateway follows a round-robin approach, distributing requests to each available server in a backend pool in turn.

Correct!

Next unit: Exercise - Create web sites

Continue

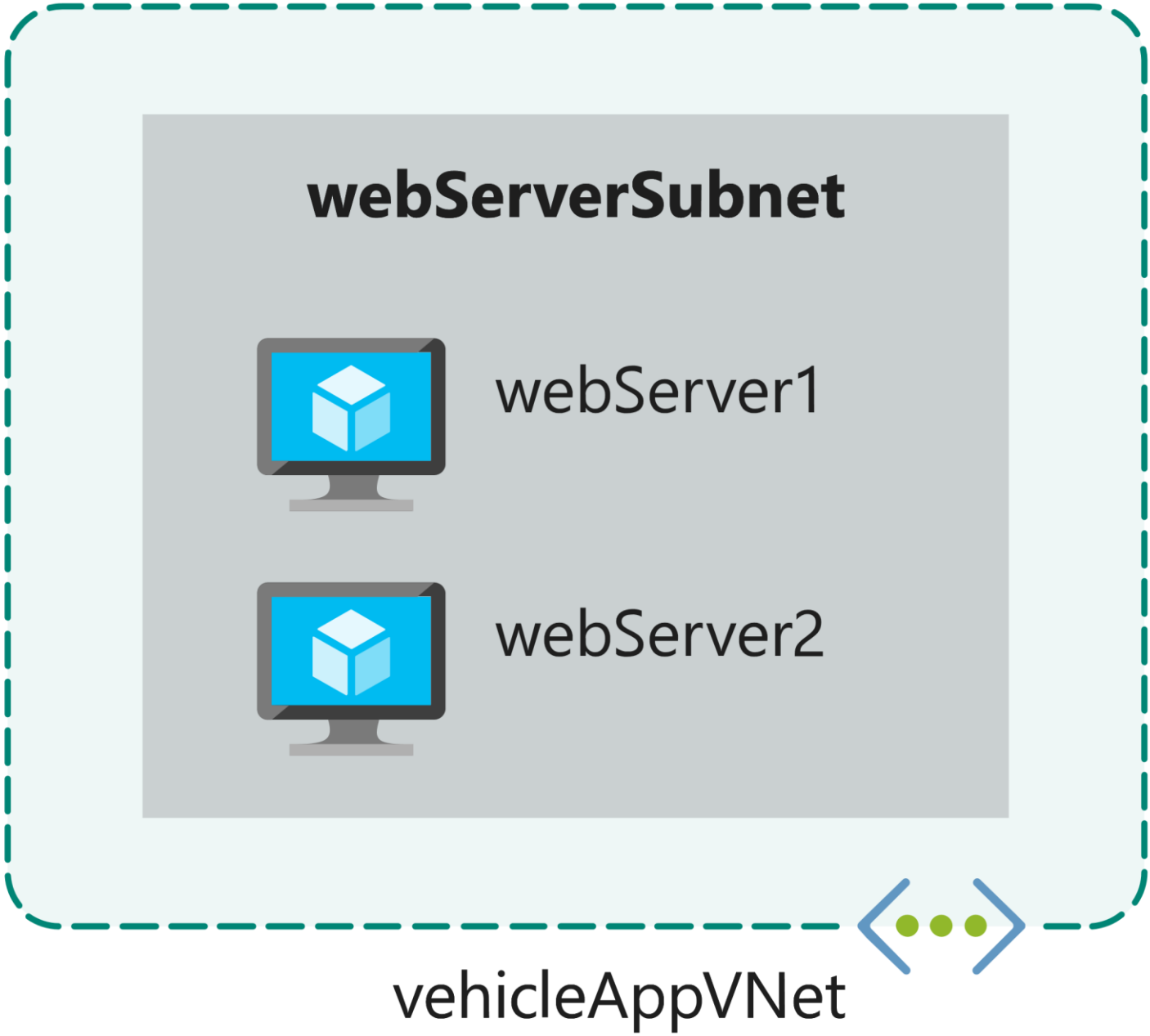
100 XP

Exercise - Create web sites

10 minutes

In the motor vehicle department system, you decide to run the web app on two servers. You'll implement each server using a virtual machine.

In this exercise, you'll create a pair of virtual machines and install the vehicle registration web app. You'll also configure a virtual network that Application Gateway can use to connect to the virtual machines. Finally, you'll deploy the license renewal web site to an instance of Azure App Service.



licenserenewal
App Service

Create virtual machines and deploy the vehicle registration site

1. Open the Azure Cloud Shell in your browser, and log in to the directory with access to the subscription you want to create resources in.

2. Run the following command in the Cloud Shell to create a variable to store your resource group name, and a resource group for your resources. Replace `<resource group name>` with a name for your resource group, and `<location>` with the Azure region you'd like to deploy your resources in.

Azure CLI Copy

```
rg=<resource group name>
 az group create --name $rg --location <location>
```

3. In the Cloud Shell window on the right, run the following command. This command uses the Azure command-line interface to create a virtual network named `vehicleappvnet`. It's a private network that provides addresses in the range 10.0.0.0 to 10.0.255.255. The command also creates a subnet called `webServerSubnet`, with the address range 10.0.1.0 to 10.0.1.255. This subnet will contain the virtual machines.

```
Azure CLI
```

```
az network vnet create \   --resource-
group $rg \
  --name vehicleAppVnet \
  --address-prefix 10.0.0.0/16 \
  --subnet-name webServerSubnet \
  --subnet-prefix 10.0.1.0/24
```

4. Download the script that creates the virtual machines with the following command:

```
bash
```

```
    git clone https://github.com/MicrosoftDocs/mslearn-load-balance-web-traffic-with-application-gateway/ module-files
```

5. Run the following commands to create and configure the virtual machines for the web servers. The virtual machines are called `webServer1` and `webServer2`. Each virtual machine runs Ubuntu Server. An administrative user account is created for each virtual machine, with the login name `azureuser`. Each virtual machine has the vehicle registration web app installed.

   The first command runs asynchronously to allow both virtual machines to be created simultaneously.

```
Azure CLI
```

```
az vm create \
  --resource-group $rg \
  --name webServer1 \
  --image UbuntuLTS \
  --admin-username azureuser \
  --generate-ssh-keys \
  --vnet-name vehicleAppVnet \
  --subnet webServerSubnet \
  --public-ip-address "" \
  --nsg "" \
  --custom-data module-files/scripts/vmconfig.sh \
  --no-wait

az vm create \
  --resource-group $rg \
  --name webServer2 \
  --image UbuntuLTS \
  --admin-username azureuser \
  --generate-ssh-keys \
  --vnet-name vehicleAppVnet \
  --subnet webServerSubnet \
  --public-ip-address "" \
  --nsg "" \
  --custom-data module-files/scripts/vmconfig.sh
```

6. Run the following command to confirm both virtual machines were created successfully.

```
Azure CLI
```

```
az vm list \
  --resource-group $rg \
  --show-details \
  --output table
```

You should see output similar to the following. Ensure the **PowerState** is **VM running** for both virtual machines before continuing.

```
output
```

```
Name          ResourceGroup      PowerState    PublicIps    Fqdns    Location        Zones ------
------  -----------------  ------------  -----------  -------  --------------  ------- webServer1
MyResourceGroup    VM running                          southcentralus webServer2
MyResourceGroup    VM running                          southcentralus
```

You've now created the virtual machines running the vehicle registration web app. Both virtual machines are identical, and are part of the same virtual network.

Create App Service and deploy the license renewal site

1. To start, run the following command to generate a unique name for the web site.

```
bash
```

```
APPSERVICE="licenserenewal$RANDOM"
```

2. Next, run the following command to create the app service plan the web app will use.

```
Azure CLI
```

```
az appservice plan create \
    --resource-group $rg \
    --name vehicleAppServicePlan \
    --sku S1
```

3. Lastly, create the web app and deploy the license renewal site.

```
Azure CLI
```

```
az webapp create \
    --resource-group $rg \
    --name $APPSERVICE \
    --plan vehicleAppServicePlan \
    --deployment-source-url https://github.com/MicrosoftDocs/mslearn-load-balance-web-traffic-with-application-gateway \     --
deployment-source-branch appService
```

Now let's take a closer look at configuring Application Gateway.

Next unit: Application Gateway creation and configuration

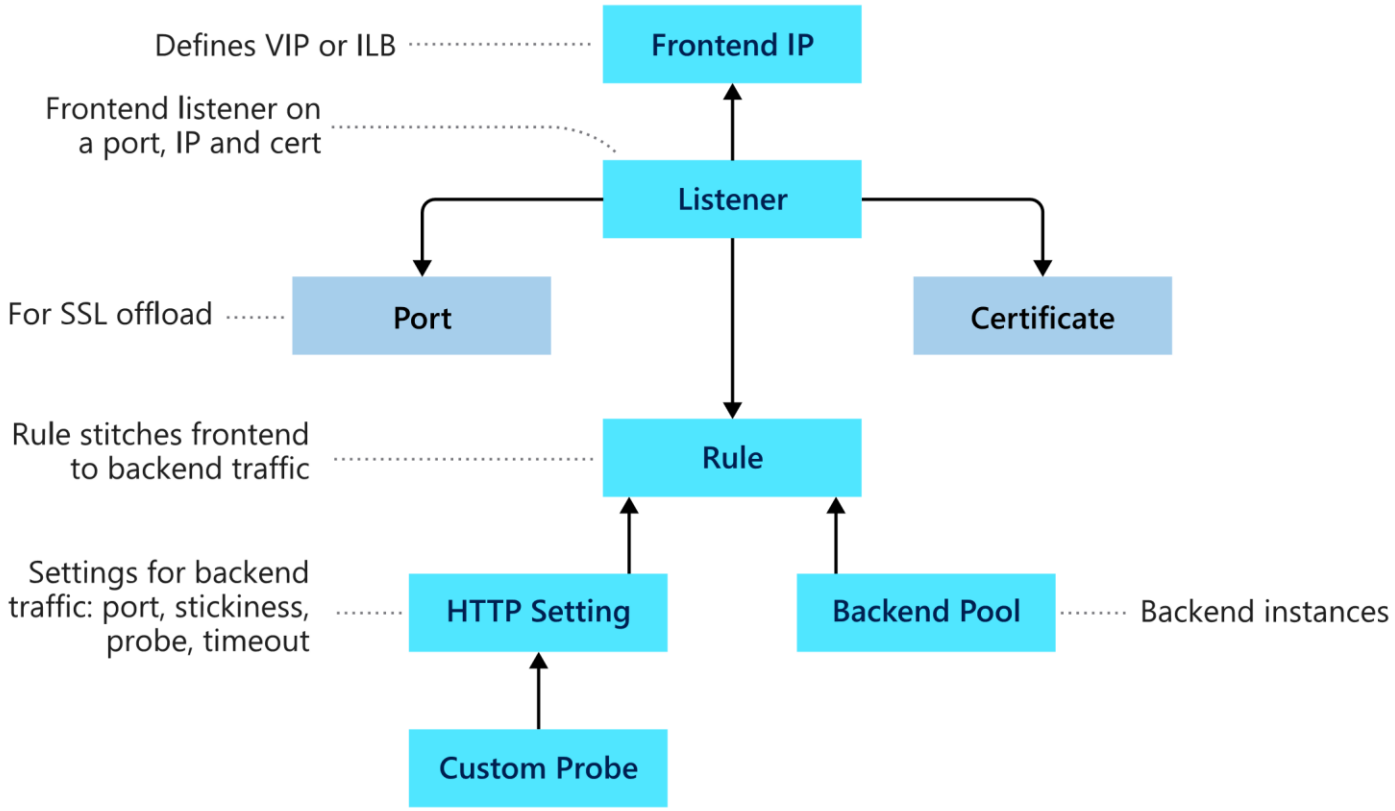
Continue

100 XP

Application Gateway creation and configuration

10 minutes

Application Gateway comprises a series of components that combine to route requests to a pool of web servers and to check the health of these web servers. Let's take a look at how these components are related and what role they play in an Application Gateway.



Front-end IP address

Client requests are received through a *front-end IP address*. You can configure Application Gateway to have a public IP address, a private IP address, or both. Application Gateway can't have more than one public and one private IP address.

Listeners

Application Gateway uses one or more *listeners* to receive incoming requests. A listener accepts traffic arriving on a specified combination of protocol, port, host, and IP address. Each listener routes requests to a back-end pool of servers following routing rules that you specify. A listener can be *Basic* or *Multi-site*. A Basic listener only routes a request based on the path in the URL. A Multi-site listener can also route requests using the hostname element of the URL.

Listeners also handle SSL certificates for securing your application between the user and Application Gateway.

Routing rules

A *routing rule* binds a listener to the back-end pools. A rule specifies how to interpret the hostname and path elements in the URL of a request, and direct the request to the appropriate back-end pool. A routing rule also has an associated set of HTTP settings. These settings indicate whether (and how) traffic is encrypted between Application Gateway and the back-end servers, and other configuration information such as:

- Protocol (HTTP or HTTPS).
- Session stickiness, to pass all requests in a client session to the same web server rather than distributing them across servers with load balancing.

- Connection draining, to enable the graceful removal of servers from a back-end pool.
- Request timeout period, in seconds.
- Health probes, specifying a probe URL, time out periods, and other parameters used to determine whether a server in the back-end pool is available.

Back-end pools

A *back-end pool* references a collection of web servers. You provide the IP address of each web server and the port on which it listens for requests when configuring the pool. Each pool can specify a fixed set of virtual machines, a virtual machine scale-set, an app hosted by Azure App Services, or a collection of on-premises servers. Each back-end pool has an associated load balancer that distributes work across the pool

Web application firewall

The *web application firewall* (WAF) is an optional component that handles incoming requests before they reach a listener. The web application firewall checks each request for many common threats, based on the *Open Web Application Security Project* (OWASP). These include:

- SQL-injection
- Cross-site scripting
- Command injection
- HTTP request smuggling
- HTTP response splitting
- Remote file inclusion
- Bots, crawlers, and scanners
- HTTP protocol violations and anomalies

OWASP has defined a set of generic rules for detecting attacks. These rules are referred to as the Core Rule Set (CRS). The rule sets are under continuous review as attacks evolve in sophistication. WAF supports two rule sets, CRS 2.2.9 and CRS 3.0. CRS 3.0 is the default and more recent of these rule sets. If necessary, you can opt to select only specific rules in a rule set, targeting certain threats. Additionally, you can customize the firewall to specify which elements in a request to examine, and limit the size of messages to prevent massive uploads from overwhelming your servers.

WAF is enabled on your Application Gateway by selecting the `WAF` tier when you create a gateway.

Health probes

Health probes are an important part in assisting the load balancer to determine which servers are available for load balancing in a back-end pool. Application Gateway uses a health probe to send a request to a server. If the server returns an HTTP response with a status code between 200 and 399, the server is deemed healthy.

If you don't configure a health probe, Application Gateway creates a default probe that waits for 30 seconds before deciding that a server is unavailable.

Application Gateway network requirements

Application Gateway requires a virtual network in which to operate. You must create this virtual network and a dedicated subnet before setting up Application Gateway. Application Gateway uses a number of private addresses for internal use and for communicating with each instance if the gateway scales out. For example, If you plan on scaling out to four instances, create a /28 size subnet. If you're likely to scale to more instances, then create a bigger subnet.

You can expose the Application Gateway through a public IP address, or you can or keep it private by only giving it a private IP inside virtual network. This is useful if you have internal sites that you would like to use Application Gateway to provide load balancing.

Application Gateway options

You can create an Application Gateway on the *Standard* tier or the *WAF* tier. You also have a choice of three sizes with varying performance, pricing, and scalability: Small, Medium, and Large.

The *Standard* and *WAF* tiers are available in two versions, V1 and V2. V2 supports Azure availability zones, but is currently in preview.

Application Gateway supports manual scaling and autoscaling. If you select autoscaling, Application Gateway will scale out and in automatically according to the application traffic. You can limit the maximum and minimum number of instances of Application Gateway.

Create and configure a gateway

You can create and configure Application Gateway using the Azure portal, Azure PowerShell, or the Azure CLI. For Azure CLI, use the `az network application-gateway create` command to create a new gateway. If you prefer PowerShell, you can use the `New-AzApplicationGateway` cmdlet. You can also use the Azure portal to perform most operations.

You can examine and modify the configuration of the components in a gateway using the `az network application-gateway http-listener`, `az network application-gateway rule`, `az network application-gateway address-pool`, `az network application-gateway http-settings`, and `az network application-gateway front-end-port` commands from the Azure CLI. The `Get-AzApplicationGateway*` and `Set-AzApplicationGateway*` series of cmdlets provide the same operations for PowerShell.

Let's create and configure an Application gateway for the motor vehicle department web sites we deployed.

Next unit: Exercise - Create and configure an Application Gateway

Continue T

100 XP

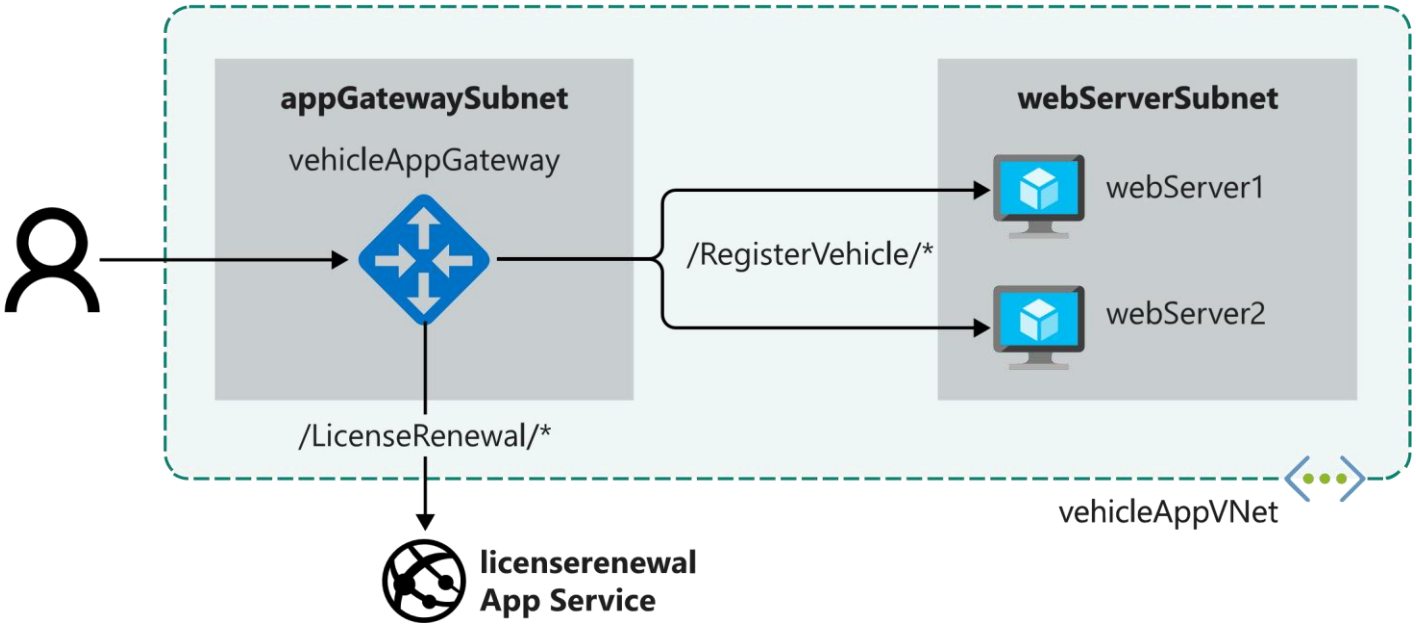
Exercise - Create and configure an Application Gateway

45 minutes

Application Gateway listens on an endpoint for incoming requests, and forwards these requests to one of the web servers in its back-end pool. You provide the configuration that describes how Application Gateway directs traffic, and how to load balance requests across web servers.

In the motor vehicle department system, you need to configure Application Gateway to load balance incoming requests across the web servers hosting the vehicle registration web app. You also need to configure Application Gateway to detect when either of the web servers has failed, so it can redirect traffic to a working server. Additionally, you need to configure path-based routing to send requests for the vehicle registration and license renewal sites to the proper back-end web services.

In this exercise, you'll create an instance of Application Gateway with a back-end pool of web servers. You'll verify that Application Gateway is configured with the correct listener to handle incoming HTTP requests, and routes these requests to a functioning web server.



Configure the network for Application Gateway

1. Run the following command to create the private subnet required by Application Gateway. The subnet is named `appGatewaySubnet`, in the `vehicleAppVnet` virtual network that you created in the previous exercise.

Azure CLI = Copy

```
az network vnet subnet create \   --resource-
group $rg \
  --vnet-name vehicleAppVnet  \   --name
appGatewaySubnet \
  --address-prefixes 10.0.0.0/24
```

2. Run the following command to create a public IP address and DNS label for Application Gateway. The DNS label must be globally unique.

   The code below uses the `$RANDOM` function to generate a label.

Azure CLI = Copy

```
az network public-ip create \   --resource-
group $rg \
  --name appGatewayPublicIp \
```

```
  --sku Standard \
  --dns-name vehicleapp${RANDOM}
```

1. ## ɛate an application gateway

   Create an application gateway named `vehicleAppGateway` with the following configuration:

   - A back-end pool containing the IP addresses of the web server virtual machines
   - A firewall that blocks malicious requests, such as those used by SQL Injection and Cross-Site Scripting attacks
   - A temporary listener that listens to port 8080, this will be replaced in a later step but is required for Application Gateway creation A rule that routes (and load balances) these requests to the web servers in the back-end pool

   Azure CLI = Copy

   ```
   az network application-gateway create \
   --resource-group $rg \
   --name vehicleAppGateway \
   --sku WAF_v2 \ --capacity
   2 \
   --vnet-name vehicleAppVnet \
   --subnet appGatewaySubnet \
   --public-ip-address appGatewayPublicIp \
   --http-settings-protocol Http \
   --http-settings-port 8080 \
   --frontend-port 8080
   ```

2. Not

   This command can take several minutes to complete.

   Run the following commands to find the private IP addresses of `webServer1` and `webServer2`. We will save these to variables to use in the next command.

   Azure CLI = Copy

3. ```
   WEBSERVER1IP="$(az vm list-ip-addresses \
     --resource-group $rg \
     --name webServer1 \
     --query [0].virtualMachine.network.privateIpAddresses[0] \
     --output tsv)"

   WEBSERVER2IP="$(az vm list-ip-addresses \
     --resource-group $rg \
     --name webserver2 \
     --query [0].virtualMachine.network.privateIpAddresses[0] \
     --output tsv)"
   ```

   Next, we'll add the back-end pools for each web site. First, create the back-end pool for the vehicle registration site running on virtual machines. We'll use the variables with the IP addresses for each VM from the previous command.

   Azure CLI = Copy

4. ```
   az network application-gateway address-pool create \
     --gateway-name vehicleAppGateway \
     --resource-group $rg \
     --name vmPool \
     --servers $WEBSERVER1IP $WEBSERVER2IP
   ```

   Now run the following command to create a back-end pool for the license renewal site running on App Service.

   Azure CLI =Copy

```
az network application-gateway address-pool create \
    --resource-group $rg \
    --gateway-name vehicleAppGateway \
    --name appServicePool \
    --servers $APPSERVICE.azurewebsites.net
```

5. We will now create a front-end port for port 80.

Azure CLI = Copy

```
az network application-gateway frontend-port create \
    --resource-group $rg \
    --gateway-name vehicleAppGateway \
    --name port80 \
    --port 80
```

6. Now we will create the listener to handle requests on port 80.

Azure CLI = Copy

```
az network application-gateway http-listener create \
    --resource-group $rg \
    --name vehicleListener \
    --frontend-port port80 \
    --gateway-name vehicleAppGateway
```

d a health probe

1. Create a health probe that tests the availability of a web server. The health probe runs every 15 seconds (`--interval 15`) and sends an HTTP GET request to the root path of the web app. If the web app doesn't respond within 10 seconds (`--timeout 10`), the probe times out.

   The web server is marked as unhealthy if the probe fails three times in succession (`--threshold 3`).

   Since we're using App Service as one of our back-ends, we will set the host header to the name of the App Service. Without this setting, the App Service won't respond and will not show as healthy.

Azure CLI = Copy

```
az network application-gateway probe create \
    --resource-group $rg \
    --gateway-name vehicleAppGateway \
    --name customProbe \
    --path / \
    --interval 15 \
    --threshold 3 \
    --timeout 10 \
    --protocol Http \
    --host-name-from-http-settings true
```

2. Next, create the HTTP Settings for the gateway to use the health probe we created.

Azure CLI = Copy

```
az network application-gateway http-settings update \
    --resource-group $rg \
    --gateway-name vehicleAppGateway \
    --name appGatewayBackendHttpSettings \
    --host-name-from-backend-pool true \
    --port 80 \
    --probe customProbe
```

nfigure path-based routing

Now we need to configure path-based routing for our Application gateway. We'll route requests to **/VehicleRegistration/** to the **vmPool** and requests to **/LicenseRenewal/** to the **appServicePool**. Any requests without any URL context will be routed to the **vmPool** as a default.

1. Run the following command to create the path map for the **vmPool**.

Azure CLI = Copy

```
az network application-gateway url-path-map create \
    --resource-group $rg \
    --gateway-name vehicleAppGateway \
    --name urlPathMap \
    --paths /VehicleRegistration/* \
    --http-settings appGatewayBackendHttpSettings \
    --address-pool vmPool
```

2. Run the following command to create the path map rule for the **appServicePool**.

Azure CLI = Copy

```
az network application-gateway url-path-map rule create \
    --resource-group $rg \
    --gateway-name vehicleAppGateway \
    --name appServiceUrlPathMap \
    --paths /LicenseRenewal/* \
    --http-settings appGatewayBackendHttpSettings \
    --address-pool appServicePool \
    --path-map-name urlPathMap
```

3. Now, create a new routing rule using the path map we created.

Azure CLI = Copy

```
az network application-gateway rule create \
    --resource-group $rg \
    --gateway-name vehicleAppGateway \
    --name appServiceRule \
    --http-listener vehicleListener \
    --rule-type PathBasedRouting \
    --address-pool appServicePool \
    --url-path-map urlPathMap
```

4. The last piece of configuration is to delete the rule that was created when we initially deployed the Application Gateway. With our custom rule in place, we no longer need it.

Azure CLI = Copy

```
az network application-gateway rule delete \
    --resource-group $rg \
    --gateway-name vehicleAppGateway \
    --name rule1
```

With everything set up it's time to test it out.

Next unit: Exercise - Test your Application Gateway

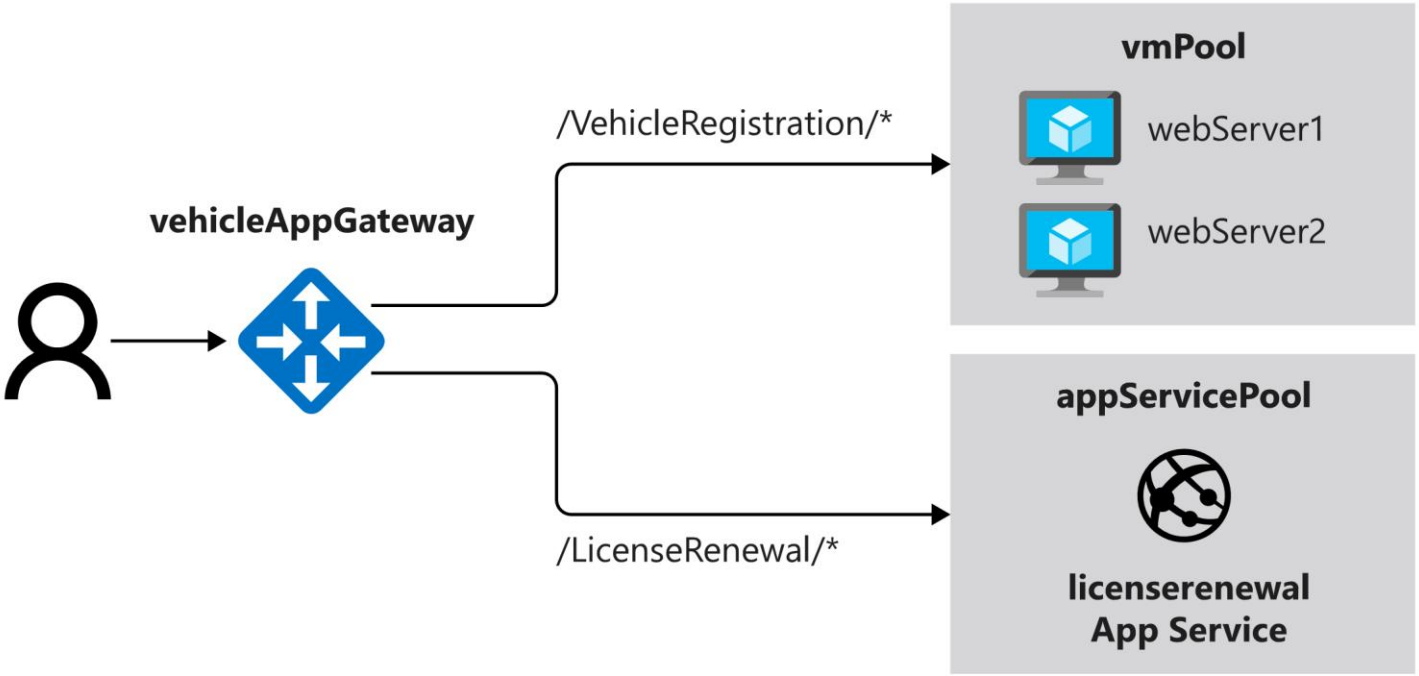
Continue

100 XP

Exercise - Test your Application Gateway

10 minutes

The final step is to test the application gateway and verify that it implements load balancing, and won't attempt to direct traffic to a web server that is unavailable. We also want to ensure that our path-based routing is working correctly.



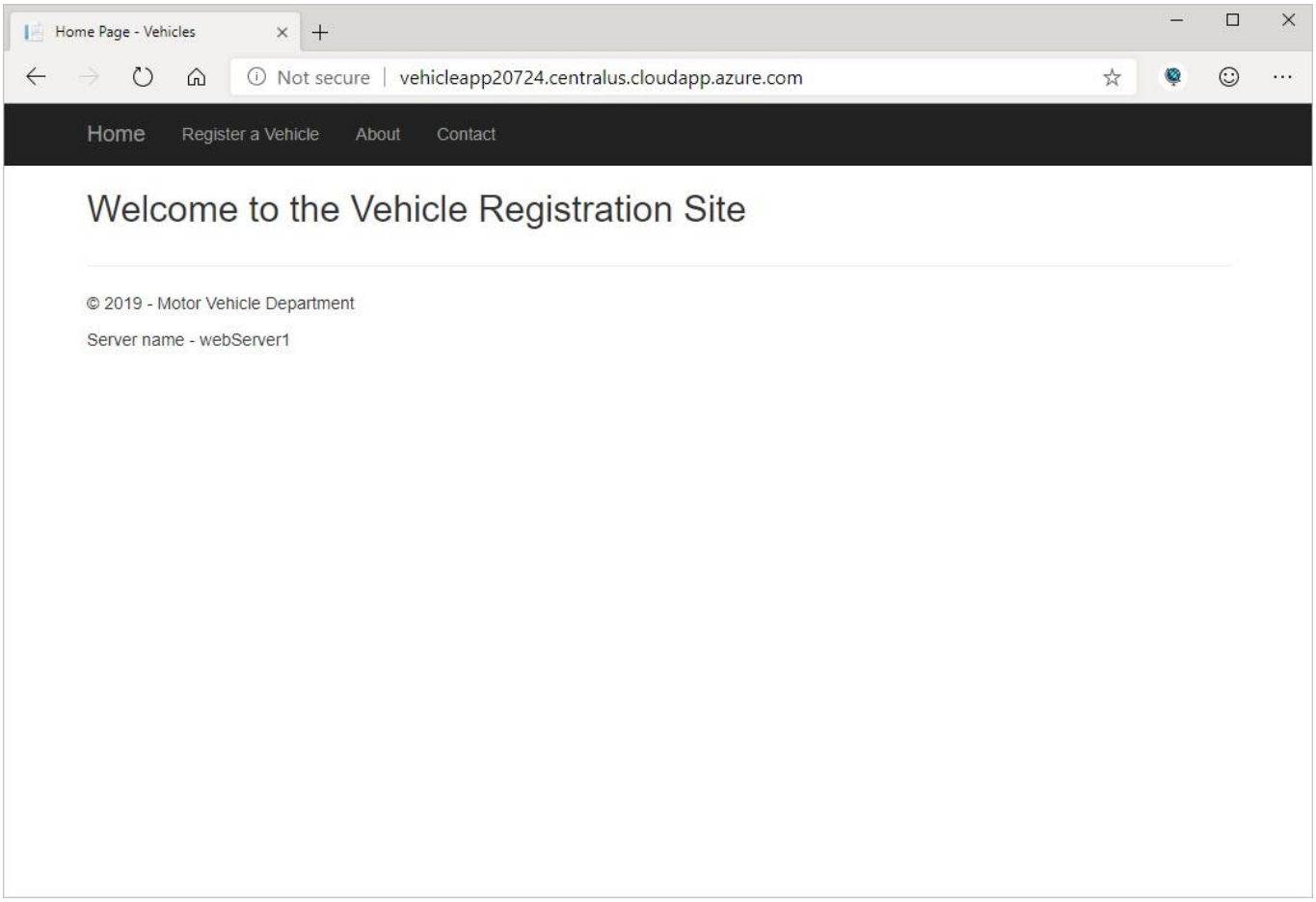
Test load balancing for the vehicle registration web app

1. In the Cloud Shell, run the following command to generate the root URL your Application Gateway.

Azure CLI Copy

```
echo http://$(az network public-ip show \
  --resource-group $rg \
  --name appGatewayPublicIp \
  --query dnsSettings.fqdn \
  --output tsv)
```

2. Using a web browser, navigate to the web site at the URL returned by the previous command. This is the address of your application gateway. Verify that the home page of the vehicle registration web app appears. Note the name of the web server that you're using as shown in the footer (**webServer1** or **webServer2**).



3. Click **Register a Vehicle**, enter the details of a vehicle, and then click **Register**.
4. Click **Refresh** in the address bar of the web browser. Notice that your session should now be connected to a different web server. In this configuration, Application Gateway uses round-robin load balancing.
5. Click **Refresh** a few more times. The requests should oscillate between servers.

Test the resilience of Application Gateway to a failed server

1. In the Cloud Shell, run the following command to stop and deallocate the virtual machine for **webServer1**:

Azure CLI Copy

```
az vm deallocate \
  --resource-group $rg \
  --name webServer1
```

2. Return to the application in the web browser and click **Refresh** several times. Notice that the web browser now only connects to **webServer2**.
3. In the Cloud Shell window on the right, restart the **webServer1** instance:

Azure CLI Copy

```
az vm start \
  --resource-group $rg \
  --name webServer1
```

4. Return to the web application in the web browser and click **Refresh** several times. You should see that the requests are now distributed across both web servers again.

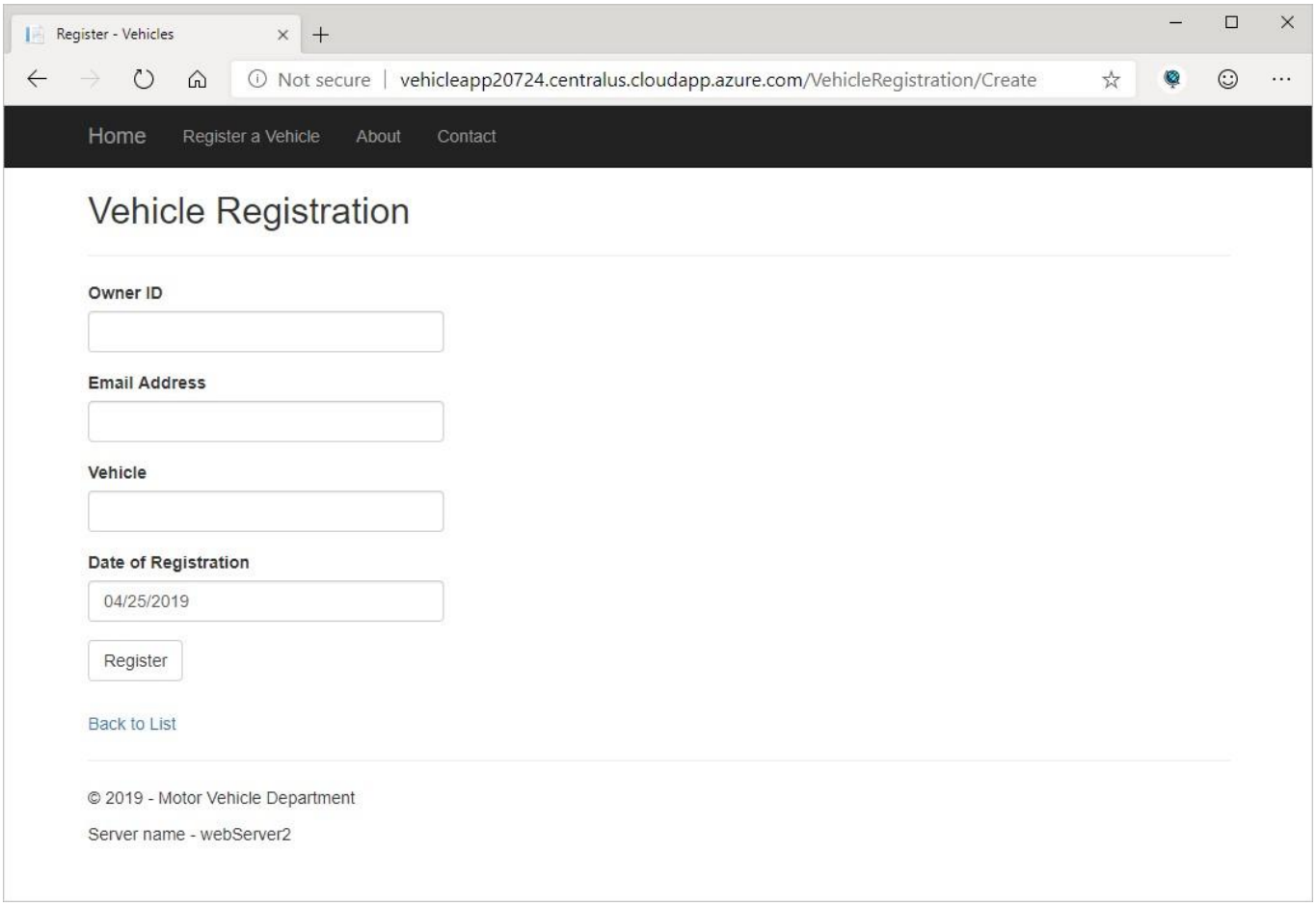
You've verified that the web application is running. Application Gateway uses load balancing to distribute requests across servers. Application Gateway detects whether a server has failed, and won't route requests to a server that is unavailable.

Test path-based routing

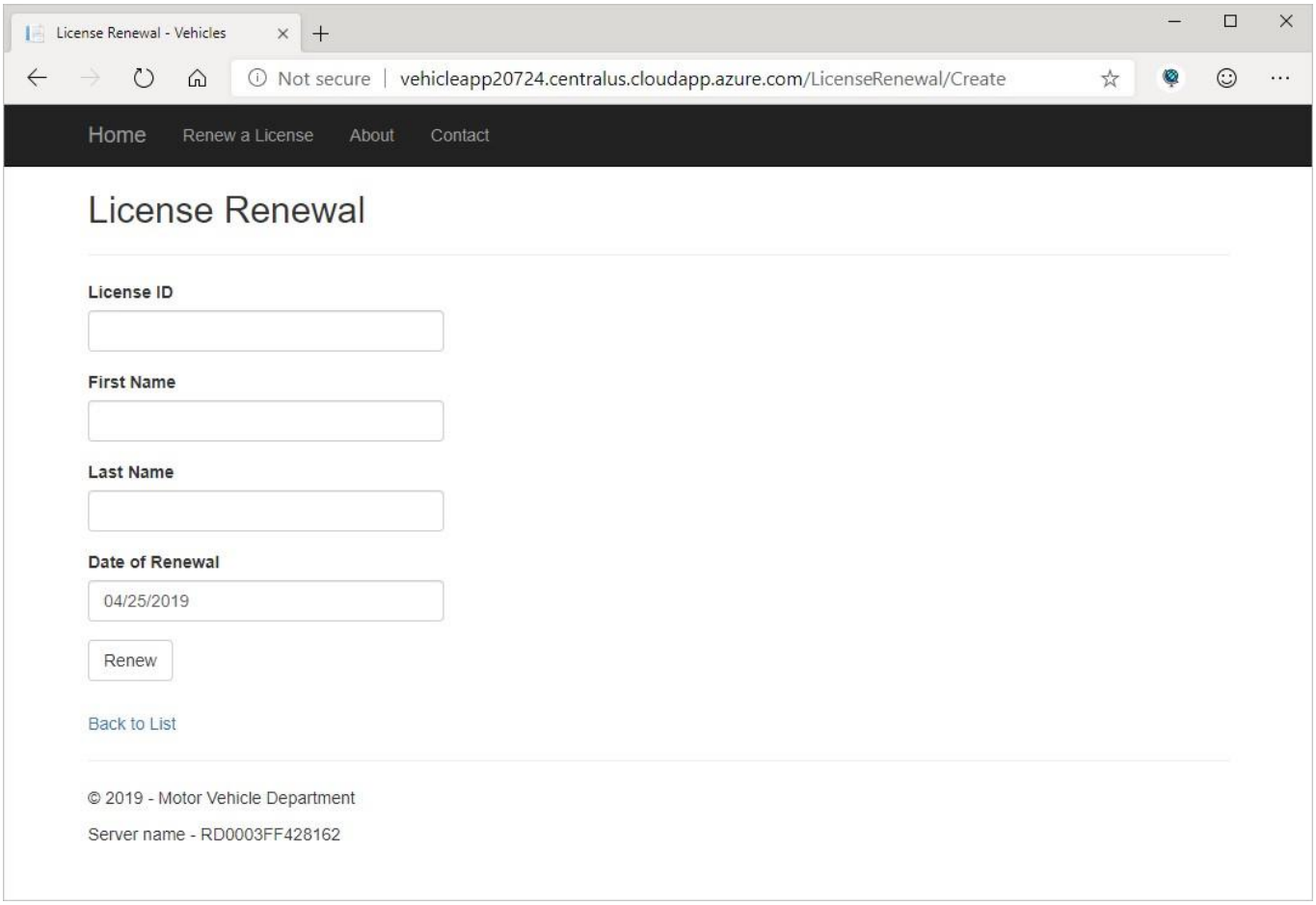
Now let's test out the path-based routing. Recall that URLs to the root of the site and with **/VehicleRegistration/** will be routed to the **vmPool** containing our VMs, and requests to **/LicenseRenewal/** will be routed to the **appServicePool** containing our App Service.

You just confirmed that routing to the root page works, as you were able to pull up the vehicle registration page. Let's try the other routes to confirm they work.

1. Now click **Register a Vehicle** in the web browser of the application gateway page. This should bring up the Vehicle Registration page for the vehicle registration site. With **/VehicleRegistration/** in the URL, this routes to the **vmPool** where our vehicle registration site is running.



2. Now visit `http://<vehicleAppGateway>/LicenseRenewal/Create`. This should take you to the license renewal page running on App Service. With **/LicenseRenewal/** in the URL, this routes to the **appServicePool** where our license renewal site is running.



With this configuration, we can direct all users for both sites through our Application Gateway, giving them one root URL to remember. We can add additional sites as we expand our web presence.

Web application firewall

We've also enabled the WAF on our Application Gateway. By doing this, we've automatically added security protection to both web sites. This provides a solid layer of protection from common vulnerabilities and helps protect our infrastructure and data.

Next unit: Summary

Continue

100 XP

Summary

2 minutes

You've learned about the load balancing and application routing capabilities of Application Gateway. You've deployed multiple web sites across different services, and used Application Gateway to ensure availability of those services. You've also used path-based routing to route requests based on the URL the user is requesting. Lastly, you deployed Application Gateway with the WAF enabled to take advantage of the built-in security that this feature provides.

You can now take advantage of the advanced routing and load-balancing capabilities that Application Gateway provides in your own environment.

Clean up

The sandbox automatically cleans up your resources when you're finished with this module.

When you're working in your own subscription, it's a good idea at the end of a project to identify whether you still need the resources you created. Resources left running can cost you money. You can delete resources individually or delete the resource group to delete the entire set of resources.

Learn more

Visit the following documentation to learn more about Application Gateway.

- How Application Gateway Works
- Application gateway components

Module complete:

Unlock achievement

900 XP

Manage and control traffic flow in your Azure deployment with routes

50 min • Module • 7 Units

V V V V W 4.6 (402) Rate it

Beginner Solutions Architect Administrator Azure Virtual Machines Virtual Network

Learn how to control Azure virtual network traffic by implementing custom routes.

In this module, you will:

Identify the routing capabilities of an Azure virtual network
Configure routing within a virtual network
Deploy a basic network virtual appliance
Configure routing to send traffic through a network virtual appliance

- Knowledge of basic networking concepts
- Familiarity with Azure virtual networking

This module is part of these learning paths
Architect network infrastructure in Azure

Introduction
2 min

Identify routing capabilities of an Azure virtual network
10 min

Exercise - Create custom routes
10 min

What is an NVA?
7 min

Exercise - Create an NVA and virtual machines
10 min

Exercise - Route traffic through the NVA
10 min

Summary
1 min

https://docs.microsoft.com/en-us/learn/modules/control-network-traffic-flow-with-routes/ 1/1

2/21/2020 Introduction - Learn | Microsoft Docs

Unit 1 of 7 S Next T

100 XP

Introduction

2 minutes

A virtual network lets you implement a security perimeter around your resources in the cloud. You can control the information that flows in and out of a virtual network. You can also restrict access to only the traffic that originates from trusted sources.

Suppose you're the solution architect for a retail organization. Also suppose your organization recently suffered a security incident that exposed customer information such as names, addresses, and credit cards. Malicious actors infiltrated vulnerabilities in the retailer's network infrastructure, which resulted in the loss of customers' confidential information.

As part of a remediation plan, the security team recommends adding network protections in the form of network virtual appliances. The cloud infrastructure team must ensure traffic is properly routed through the virtual appliances and is inspected for malicious activity.

You'll learn about Azure routing and you'll create custom routes to control the traffic flow. You'll also learn to redirect the traffic through the network virtual appliance so you can inspect the traffic before it's allowed through.

Learning objectives

In this module, you'll:

- Identify the routing capabilities of an Azure virtual network.
- Configure routing within a virtual network.
- Deploy a basic network virtual appliance.
- Configure routing to send traffic through a network virtual appliance.

Prerequisites

- Knowledge of basic networking concepts, including subnets and IP addressing
- Familiarity with Azure virtual networking

Next unit: Identify routing capabilities of an Azure virtual network

Continue

200 XP

Identify routing capabilities of an Azure virtual network

10 minutes

To control traffic flow within your virtual network, you must learn the purpose and benefits of custom routes. You must also learn how to configure the routes to direct traffic flow through a network virtual appliance (NVA).

Azure routing

Network traffic in Azure is automatically routed across Azure subnets, virtual networks, and on-premises networks. This routing is controlled by system routes, which are assigned by default to each subnet in a virtual network. With these system routes, any Azure virtual machine that is deployed to a virtual network can communicate with all other Azure virtual machines in subnets in that network. These virtual machines are also potentially accessible from on-premises through a hybrid network or the internet.

You can't create or delete system routes. But you can override the system routes by adding custom routes to control traffic flow to the next hop.

Every subnet has the following default system routes:

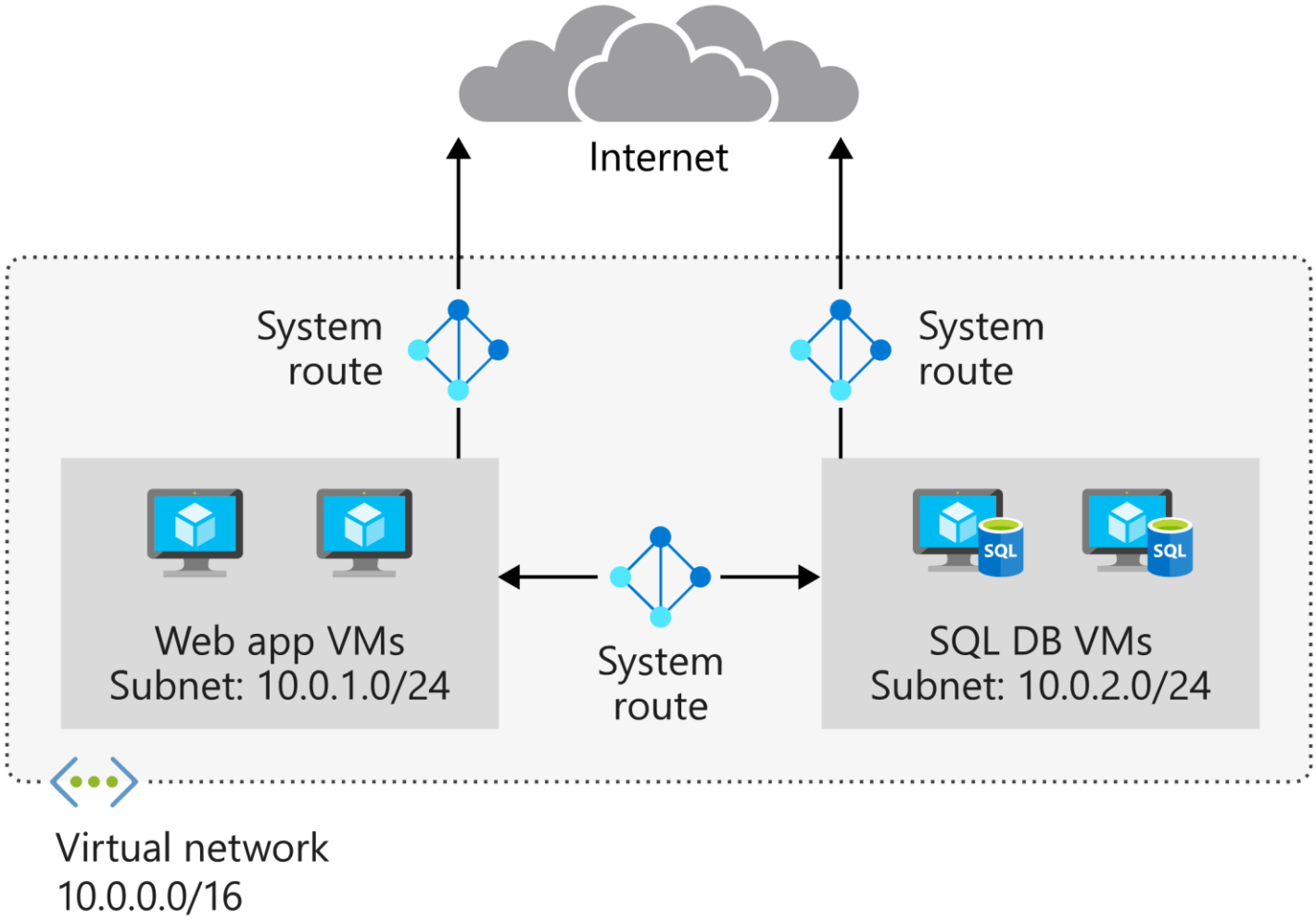
Address prefix	Next hop type
Unique to the virtual network	Virtual network
0.0.0.0/0	Internet
10.0.0.0/8	None
172.16.0.0/12	None
192.168.0.0/16	None
100.64.0.0/10	None

The **Next hop type** column shows the network path taken by traffic sent to each address prefix. The path can be one of the following hop types:

- **Virtual network**: A route is created in the address prefix. The prefix represents each address range created at the virtual-network level. If multiple address ranges are specified, multiple routes are created for each address range.
- **Internet**: The default system route 0.0.0.0/0 routes any address range to the internet, unless you override Azure's default route with a custom route.

None: Any traffic routed to this hop type is dropped and doesn't get routed outside the subnet. By default, the following IPv4 privateaddress prefixes are created: 10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16. The prefix 100.64.0.0/10 for a shared address space is also added. None of these address ranges are globally routable.

The following diagram shows an overview of system routes and shows how traffic flows among subnets and the internet by default. You can see from the diagram that traffic flows freely among the two subnets and the internet.



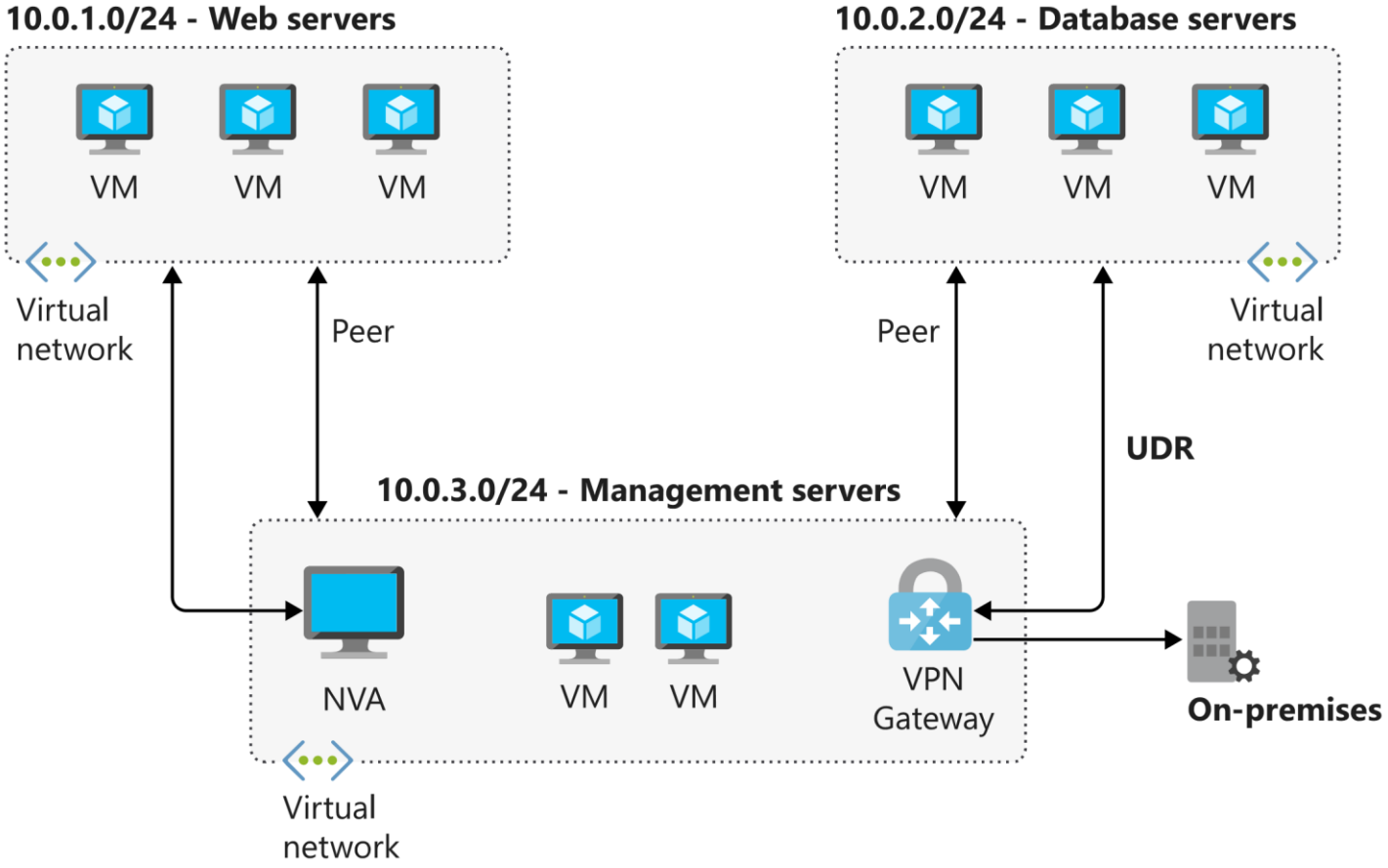
Within Azure, there are additional system routes. Azure will create these routes if the following capabilities are enabled:

- Virtual network peering
- Service chaining
- Virtual network gateway
- Virtual network service endpoint

Virtual network peering and service chaining

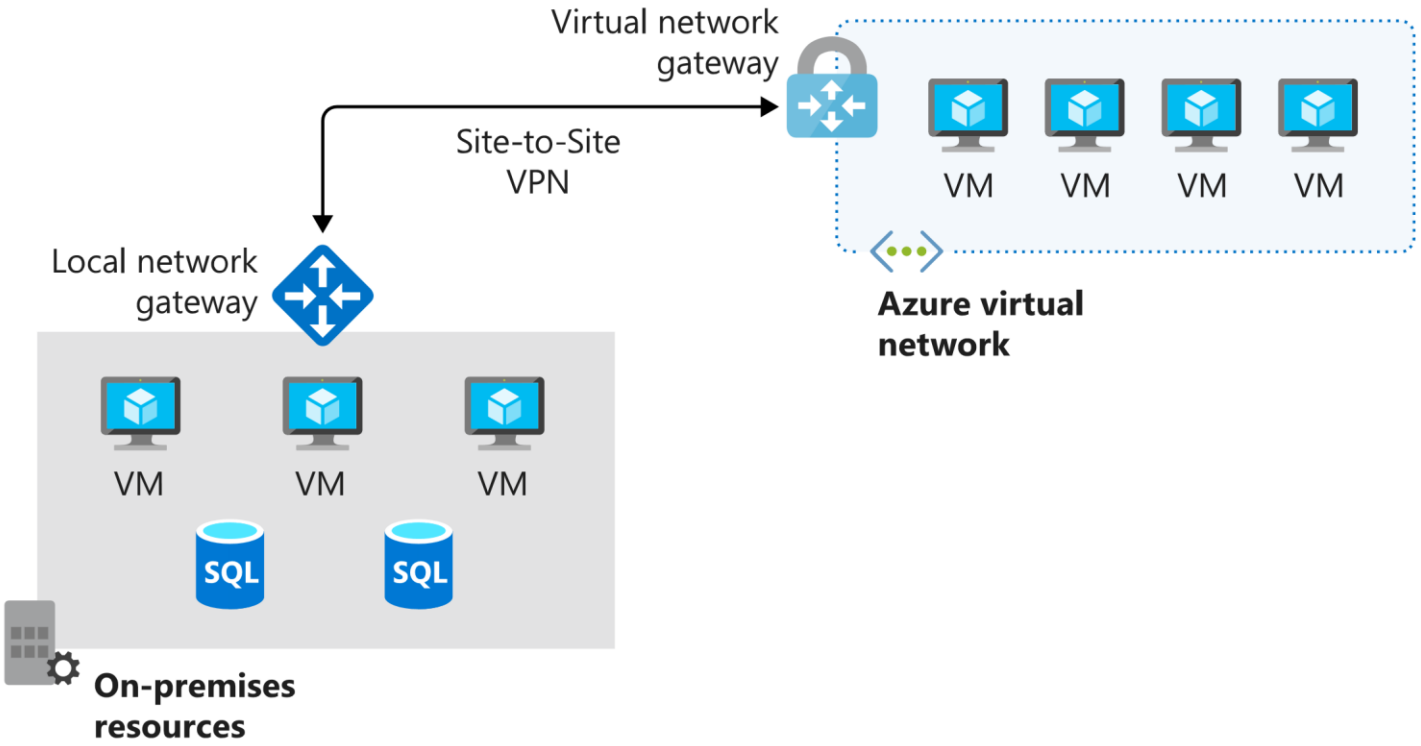
Virtual network peering and service chaining let virtual networks within Azure be connected to one another. With this connection, virtual machines can communicate with each other within the same region or across regions. This communication in turn creates additional routes within the default route table. Service chaining lets you override these routes by creating user-defined routes between peered networks.

The following diagram shows two virtual networks with peering configured. The user-defined routes are configured to route traffic through an NVA or an Azure VPN gateway.



Virtual network gateway

Use a virtual network gateway to send encrypted traffic between Azure and on-premises over the internet and to send encrypted traffic between Azure networks. A virtual network gateway contains routing tables and gateway services.



Virtual network service endpoint

Virtual network endpoints extend your private address space in Azure by providing a direct connection to your Azure resources. This connection restricts the flow of traffic: your Azure virtual machines can access your storage account directly from the private address space and deny access from a public virtual machine. As you enable service endpoints, Azure creates routes in the route table to direct this traffic.

Custom routes

System routes might make it easy for you to quickly get your environment up and running. But there are many scenarios in which you'll want to more closely control the traffic flow within your network. For example, you might want to route traffic through an NVA or through a firewall from partners and others. This control is possible with custom routes.

You have two options for implementing custom routes: create a user-defined route or use Border Gateway Protocol (BGP) to exchange routes between Azure and on-premises networks.

User-defined routes

You use a user-defined route to override the default system routes so that traffic can be routed through firewalls or NVAs.

For example, you might have a network with two subnets and want to add a virtual machine in the perimeter network to be used as a firewall. You create a user-defined route so that traffic passes through the firewall and doesn't go directly between the subnets.

When creating user-defined routes, you can specify these next hop types:

- **Virtual appliance**: A virtual appliance is typically a firewall device used to analyze or filter traffic that is entering or leaving your network. You can specify the private IP address of a NIC attached to a virtual machine so that IP forwarding can be enabled. Or you can provide the private IP address of an internal load balancer.
- **Virtual network gateway**: Use to indicate when you want routes for a specific address to be routed to a virtual network gateway. The virtual network gateway is specified as a VPN for the next hop type.
- **Virtual network**: Use to override the default system route within a virtual network.
- **Internet**: Use to route traffic to a specified address prefix that is routed to the internet.
- **None**: Use to drop traffic sent to a specified address prefix.

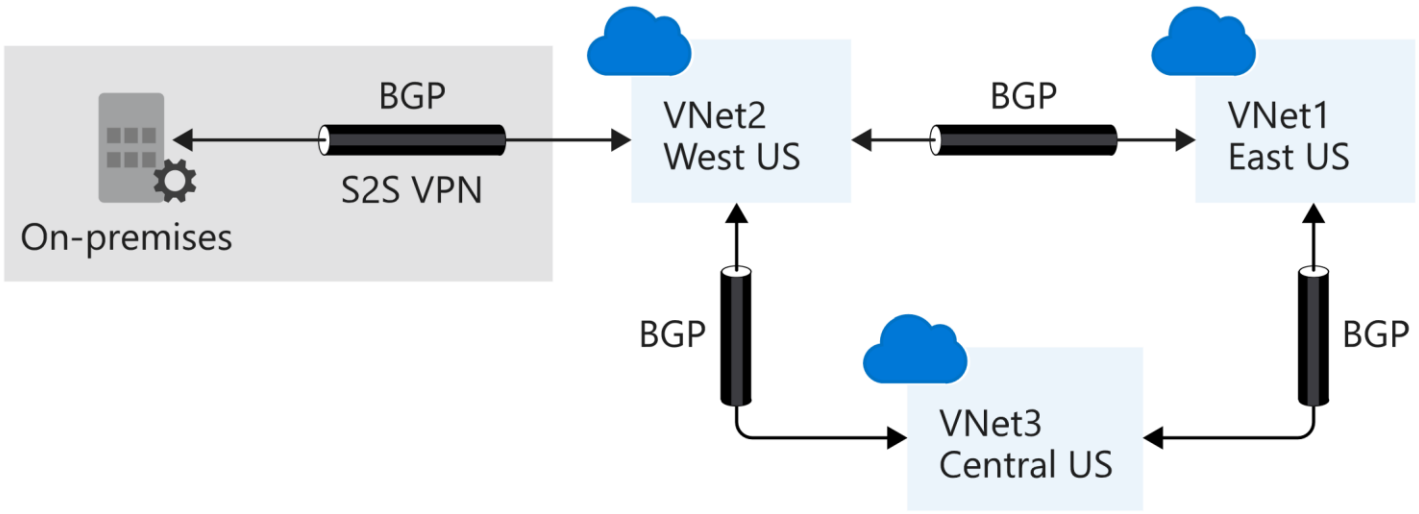
With user-defined routes, you can't specify the next hop type **VirtualNetworkServiceEndpoint**, which indicates virtual network peering.

Border gateway protocol

A network gateway in your on-premises network can exchange routes with a virtual network gateway in Azure by using BGP. BGP is the standard routing protocol that is normally used to exchange routing and information among two or more networks. BGP is used to transfer data and information between different host gateways like on the internet or between autonomous systems.

You typically use BGP to advertise on-premises routes to Azure when you're connected to an Azure datacenter through Azure ExpressRoute. You can also configure BGP if you connect to an Azure virtual network by using a VPN site-to-site connection.

The following diagram shows a topology with paths that can pass data between Azure VPN Gateway and on-premises networks:



BGP offers network stability because routers can quickly change connections to send packets if a connection path goes down.

Route selection and priority

If multiple routes are available in a route table, Azure uses the route with the longest prefix match. For example, if a message is sent to the IP address 10.0.0.2, but two routes are available with the 10.0.0.0/16 and 10.0.0.0/24 prefixes, Azure selects the route with the 10.0.0.0/24 prefix because it's more specific.

The longer the route prefix, the shorter the list of IP addresses available through that prefix. By using longer prefixes, the routing algorithm can select the intended address more quickly.

You can't configure multiple user-defined routes with the same address prefix.

If multiple routes share the same address prefix, Azure selects the route based on its type in the following order of priority:

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

Check your knowledge

1. Why would you use a custom route in a virtual network?

- To load balance the traffic in your virtual network.
- To connect to your Azure virtual machines using RDP or SSH.
- To control the flow of traffic in your Azure virtual network.

  This is the correct answer. Custom routes are used to override the default Azure routing so that you can route traffic through a network virtual appliance (NVA).
- To connect to resources in another virtual network hosted in Azure.

2. Why might you use virtual network peering?

- To connect virtual networks together in the same region.

  This is the correct answer. Virtual network peering is used to connect multiple virtual networks together. Once peered, the networks become one network, and resources across virtual networks can communicate with one another.
- To assign public IP addresses to all of your resources across multiple virtual networks.
- So that load balancers can control traffic flow across your virtual networks.
- To run custom reports that scan and identify what resources are running across all of your virtual networks, as opposed to running reports on each virtual network.

Next unit: Exercise - Create custom routes

Continue

100 XP

Exercise - Create custom routes

10 minutes

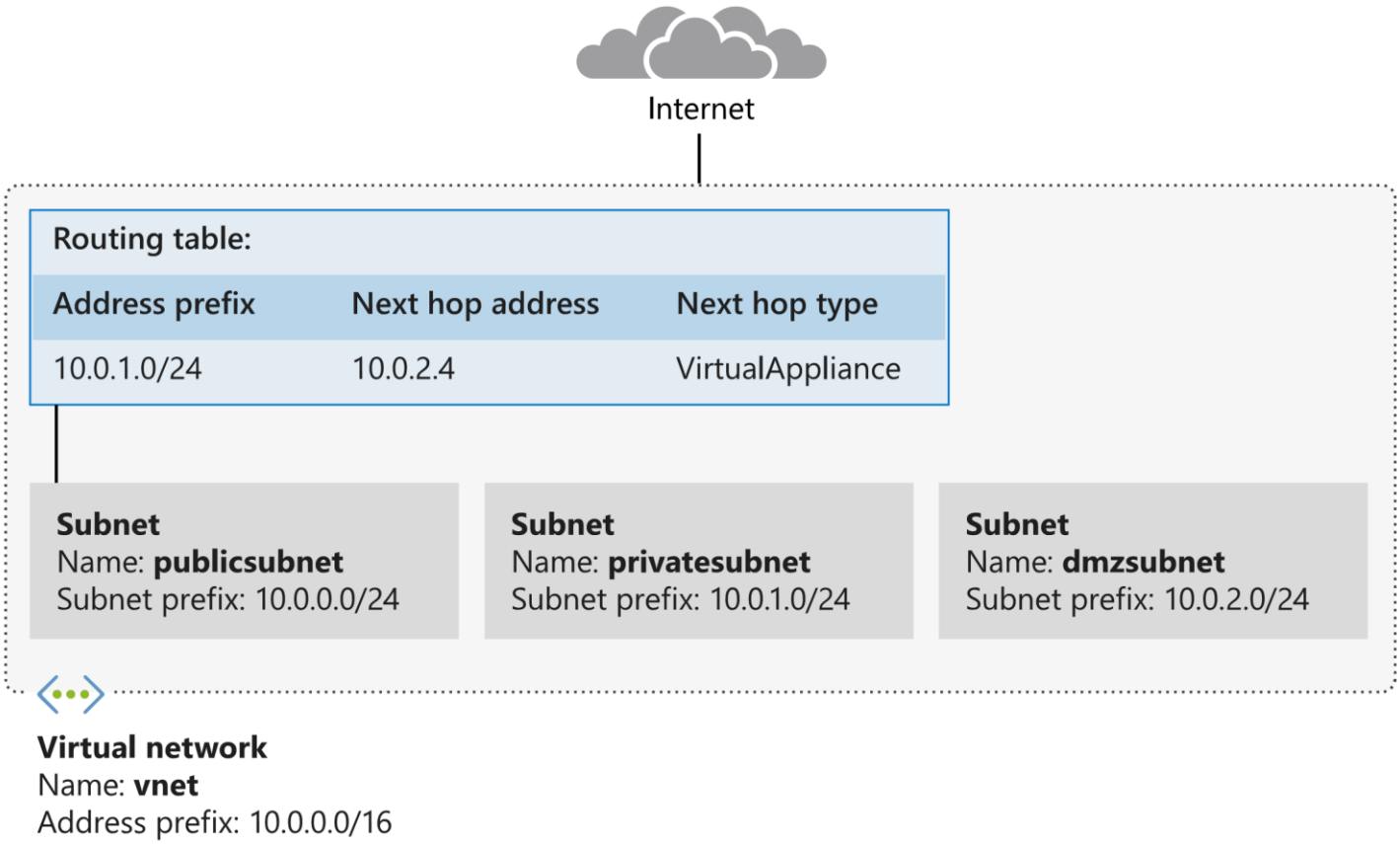
This module requires a sandbox to complete. A **sandbox** gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

Activate sandbox

As you implement your security strategy, you want to control how network traffic is routed across your Azure infrastructure.

In the following exercise, you'll use a network virtual appliance (NVA) to help secure and monitor traffic. You'll want to ensure communication between front-end public servers and internal private servers is always routed through the appliance.

You'll configure the network so that all traffic flowing from a public subnet to a private subnet will be routed through the NVA. To make this flow happen, you'll create a custom route for the public subnet to route this traffic to a perimeter-network subnet. Later, you'll deploy an NVA to the perimeter-network subnet.



In this exercise, you'll create the route table, custom route, and subnets. You'll then associate the route table with a subnet.

Create a route table and custom route

The first task is to create a new routing table and then add a custom route for all traffic intended for the private subnet.

1. On the right side of the Azure Cloud Shell window, run the following command to create a route table.

Azure CLI Copy

```
az network route-table create \
  --name publictable \
  --resource-group [sandbox resource group name] \
  --disable-bgp-route-propagation false
```

2. Run the following command in Cloud Shell to create a custom route.

Azure CLI — Copy

```
az network route-table route create \
  --route-table-name publictable \
  --resource-group [sandbox resource group name] \
  --name productionsubnet \
  --address-prefix 10.0.1.0/24 \
  --next-hop-type VirtualAppliance \
  --next-hop-ip-address 10.0.2.4
```

eate a virtual network and subnets

1. ext task is to create the **vnet** virtual network and the three subnets that you need: **publicsubnet**, **privatesubnet**, and **dmzsubnet**.

Run the following command to create the **vnet** virtual network and the **publicsubnet** subnet.

Azure CLI — Copy

```
az network vnet create \
  --name vnet \
  --resource-group [sandbox resource group name] \
  --address-prefix 10.0.0.0/16 \
  --subnet-name publicsubnet \
  --subnet-prefix 10.0.0.0/24
```

2. Run the following command in Cloud Shell to create the **privatesubnet** subnet.

Azure CLI — Copy

```
az network vnet subnet create \
  --name privatesubnet \
  --vnet-name vnet \
  --resource-group [sandbox resource group name] \
  --address-prefix 10.0.1.0/24
```

3. Run the following command to create the **dmzsubnet** subnet.

Azure CLI — Copy

```
az network vnet subnet create \
  --name dmzsubnet \    --vnet-name vnet \
  --resource-group [sandbox resource group name] \
  --address-prefix 10.0.2.0/24
```

4. You should now have three subnets. Run the following command to show all of the subnets in the **vnet** virtual network.

Azure CLI — Copy

```
az network vnet subnet list \
  --resource-group [sandbox resource group name] \
  --vnet-name vnet \
  --output table
```

sociate the route table with the public subnet

The final task in this exercise is to associate the route table with the **publicsubnet** subnet.

Run the following command to associate the route table with the public subnet.

Azure CLI = Copy

```
az network vnet subnet update \
  --name publicsubnet \
  --vnet-name vnet \
  --resource-group [sandbox resource group name] \
  --route-table publictable
```

Next unit: What is an NVA?

Continue T

Azure Cloud Shell

This module requires a sandbox to complete. A sandbox gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

200 XP

What is an NVA?

7 minutes

A network virtual appliance (NVA) is a virtual appliance that consists of various layers like:

- a firewall
- a WAN optimizer
- application-delivery
- controllers routers
- load balancers
- IDS/IPS proxies

You can deploy NVAs chosen from providers in Azure Marketplace. Such providers include Check Point, Barracuda, Sophos, WatchGuard, and SonicWall. You can use an NVA to filter traffic inbound to a virtual network, to block malicious requests, and to block requests made from unexpected resources.

In the retail-organization example scenario, you must work with the security and network teams. You want to implement a secure environment that scrutinizes all incoming traffic and blocks unauthorized traffic from passing on to the internal network. You also want to secure both virtualmachine networking and Azure-services networking as part of your company's network-security strategy.

Your goal is to prevent unwanted or unsecured network traffic from reaching key systems.

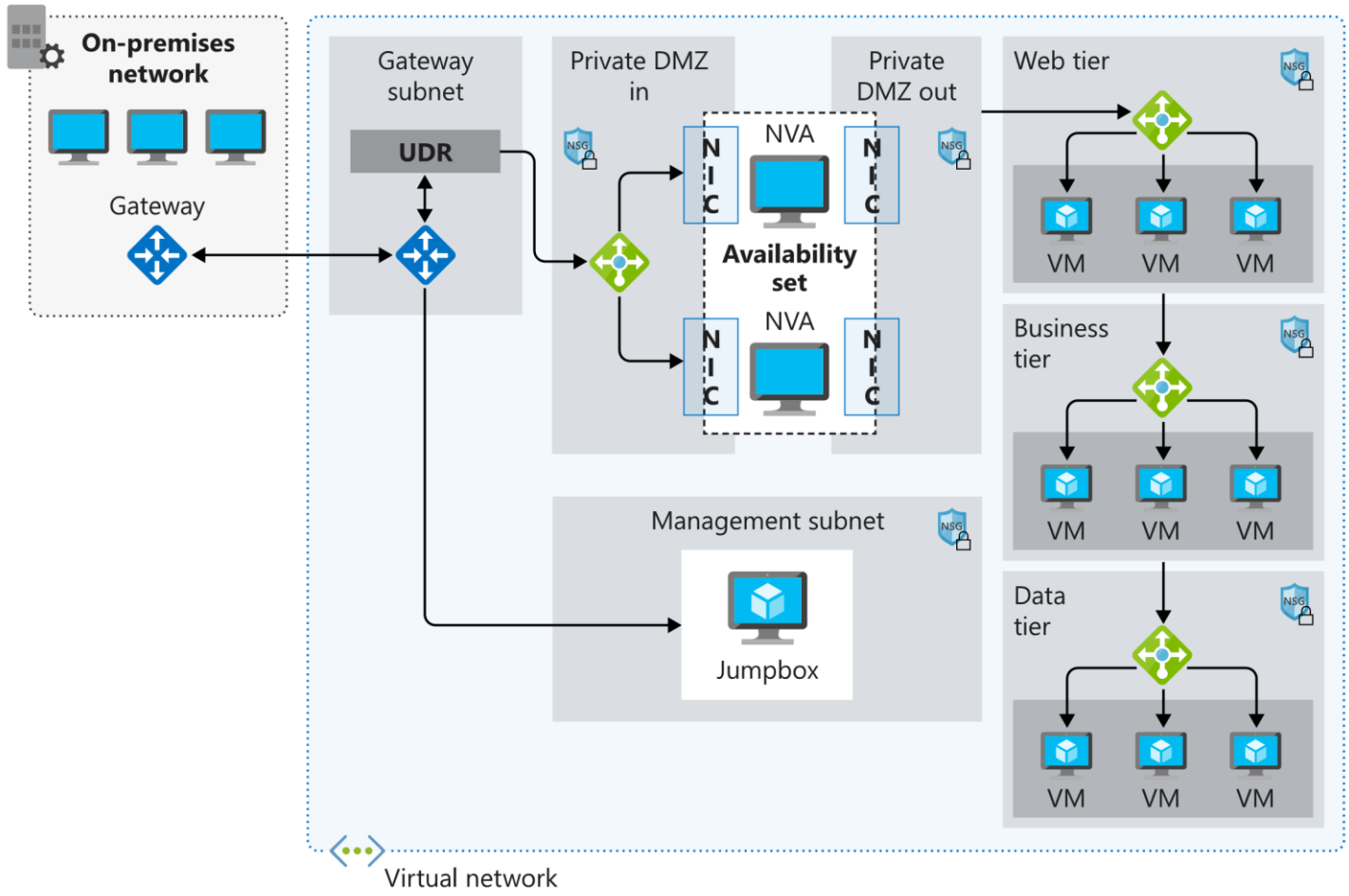
As part of the network-security strategy, you must control the flow of traffic within your virtual network. You also must learn the role of an NVA and the benefit of using an NVA to control traffic flow through an Azure network.

Network virtual appliance

Network virtual appliances or NVAs are virtual machines that control the flow of network traffic by controlling routing. You typically use them to manage traffic flowing from a perimeter-network environment to other networks or subnets.

An NVA often includes various protection layers like:

- a firewall
- a WAN optimizer
- application-delivery
- controllers routers
- load balancers
- proxies an SD-WAN edge



You can deploy firewall appliances into a virtual network in different configurations. You can put a firewall appliance in a perimeter-network subnet in the virtual network. Or if you want more control of security, implement a microsegmentation approach.

With the microsegmentation approach, you can create dedicated subnets for the firewall and then deploy web applications and other services in other subnets. All traffic is routed through the firewall and inspected by the NVAs. You enable forwarding on the virtual-appliance network interfaces to pass traffic that is accepted by the appropriate subnet.

Microsegmentation lets the firewall inspect all packets at OSI Layer 4 and, for application-aware appliances, Layer 7. When you deploy an NVA to Azure, it acts as a router that forwards requests between subnets on the virtual network.

Some NVAs require multiple network interfaces. One network interface is usually dedicated to the management network for the appliance. Additional network interfaces manage and control the traffic processing. After you've deployed the NVA, you can then configure the appliance to route the traffic through the proper interface.

User-defined routes

For most environments, the default system routes already defined by Azure are enough to get the environments up and running. But in certain cases you should create a routing table and add custom routes. Examples include:

- Access to the internet via on-premises network using forced tunneling.
- Using virtual appliances to control traffic flow.

You can define multiple routing tables in Azure. Each routing table is associated with one or more subnets. But each subnet is associated with only one routing table.

Network virtual appliances in a highly available architecture

If traffic is routed through an NVA, the NVA becomes a critical piece of your infrastructure. Any NVA failures will directly affect the ability of your services to communicate. It's important to include a highly available architecture in your NVA deployment.

There are several methods of achieving high availability when using NVAs. At the end of this module, you can find more information about using NVAs in highly available scenarios.

Check your knowledge

1. What is the main benefit of using a network virtual appliance?

- To control outbound access to the internet.
- To load balance incoming traffic from the internet across multiple Azure virtual machines and across two regions for DR purposes.
- To control incoming traffic from the perimeter network and allow only traffic that meets security requirements to pass through.

  This is the correct answer. A network virtual appliance acts like a firewall. It checks all inbound and outbound traffic, and it secures your environment by allowing or denying the traffic.
- To control who can access Azure resources from the perimeter network.

2. How might you deploy a network virtual appliance?

- You can configure a Windows virtual machine and enable IP forwarding after routing tables, user-defined routes, and subnets have been updated. Or you can use a partner image from Azure Marketplace.

  This is the correct answer. Customers often create network virtual appliances. And you can download many appliances from Azure Marketplace.
- Using Azure CLI, deploy a Linux virtual machine in Azure, connect this virtual machine to your production virtual network, and assign a public IP address.
- Using the Azure portal, deploy a Windows 2016 Server instance. Next, using Azure Application Gateway, add the Windows 2016 Server instance as a target endpoint.
- Download a virtual appliance from Azure Marketplace and configure the appliance to connect to the production and perimeter networks.

Next unit: Exercise - Create an NVA and virtual machines

Continue

100 XP

Exercise - Create an NVA and virtual machines

10 minutes

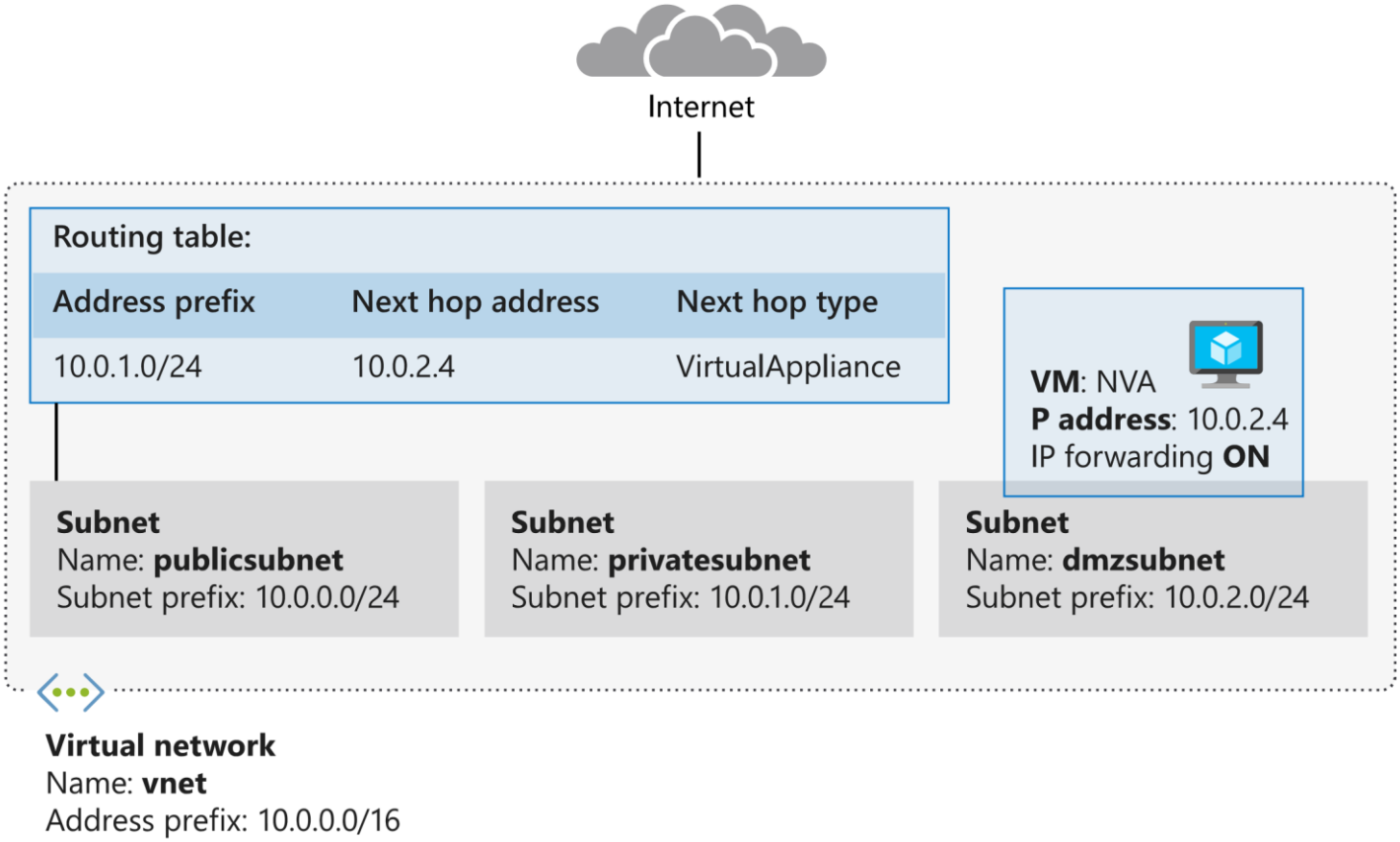
This module requires a sandbox to complete. A **sandbox** gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

Activate sandbox

In the next stage of your security implementation, you'll deploy a network virtual appliance (NVA) to secure and monitor traffic between your front-end public servers and internal private servers.

You'll configure the appliance to forward IP traffic. If IP forwarding isn't enabled, traffic that is routed through your appliance will never be received by its intended destination servers.

In this exercise, you'll deploy the **nva** network appliance to the **dmzsubnet** subnet. You'll then enable IP forwarding so that traffic from **publicsubnet** and traffic that uses the custom route is sent to the **privatesubnet** subnet.



In the following steps, you'll deploy an NVA. You'll then update the Azure virtual NIC and the network settings within the appliance to enable IP forwarding.

Deploy the network virtual appliance

To build the NVA, deploy an Ubuntu LTS instance.

1. In Cloud Shell, run the following command to deploy the appliance. Replace `<password>` with a suitable password of your choice for the **azureuser** admin account.

Azure CLI = Copy

```
az vm create \
  --resource-group [sandbox resource group name] \
  --name nva \
  --vnet-name vnet \
  --subnet dmzsubnet \
  --image UbuntuLTS \
  --admin-username azureuser \
  --admin-password <password>
```

2.

Run the following commands to retrieve the public IP address of the appliance virtual machine. Save the address to the variable named NVAIP.

Azure CLI = Copy

```
NVAIP="$(az vm list-ip-addresses \
  --resource-group [sandbox resource group name] \
  --name nva \    --query "[].virtualMachine.network.publicIpAddresses[*].ipAddress" \
  --output tsv)"
echo $NVAIP
```

1. ## able IP forwarding for the Azure network interface

e next steps, IP forwarding for the **nva** network appliance is enabled. When traffic flows to the NVA but is meant for another target, the will route that traffic to its correct destination.

Run the following command to get the ID of the NVA network interface.

Azure CLI = Copy

2.
```
NICID=$(az vm nic list \
  --resource-group [sandbox resource group name] \
  --vm-name nva \
  --query "[].{id:id}" --output tsv)
echo $NICID
```

Run the following command to get the name of the NVA network interface.

Azure CLI = Copy

3.
```
NICNAME=$(az vm nic show \
  --resource-group [sandbox resource group name] \
  --vm-name nva \
  --nic $NICID \
  --query "{name:name}" --output tsv)
echo $NICNAME
```

Run the following command to enable IP forwarding for the network interface.

Azure CLI = Copy

```
az network nic update --name $NICNAME \
  --resource-group [sandbox resource group name] \
  --ip-forwarding true
```

1. ## able IP forwarding in the appliance

Run the following command to save the public IP address of the NVA virtual machine to the variable NVAIP.

2. Azure CLI = Copy

```
NVAIP="$(az vm list-ip-addresses \
  --resource-group [sandbox resource group name] \
  --name nva \    --query "[].virtualMachine.network.publicIpAddresses[*].ipAddress" \
  --output tsv)"
echo $NVAIP
```

Run the following command to enable IP forwarding within the NVA.

bash = Copy

```
ssh -t -o StrictHostKeyChecking=no azureuser@$NVAIP 'sudo sysctl -w net.ipv4.ip_forward=1; exit;'
```

When prompted, enter the password you used when you created the virtual machine.

t unit: Exercise - Route traffic through the NVA

ıtinue T

English (United States)

Azure Cloud Shell

This module requires a sandbox to complete. A sandbox gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

100 XP

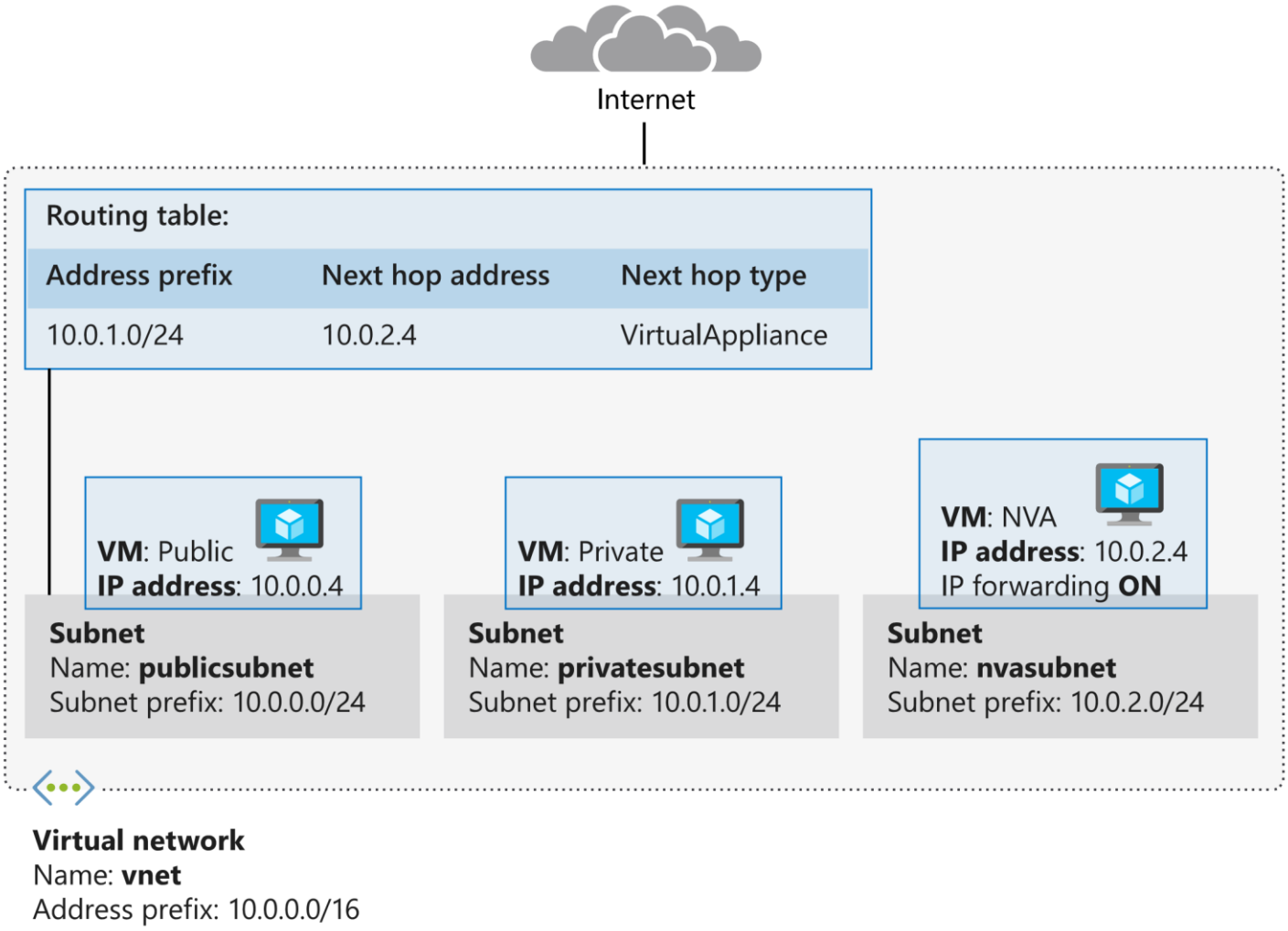
Exercise - Route traffic through the NVA

10 minutes

This module requires a sandbox to complete. A **sandbox** gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

Activate sandbox

Now that you've created the network virtual appliance (NVA) and virtual machines, you'll route the traffic through the NVA.



Create public and private virtual machines

The next steps deploy a virtual machine into the public and private subnets:

1. Open the Visual Studio Code editor and create a file named cloud-init.txt.

```bash
code cloud-init.txt
```

2. Add the following configuration information to the file. With this configuration, the inetutils-traceroute package is installed when you create a new virtual machine. This package contains the traceroute utility that you'll use later in this exercise.

```
Text
#cloud-config package_upgrade: true
packages:
  - inetutils-traceroute
```

3. Select Ctrl-S to save the file, and then select Ctrl-Q to close the editor.

4. Run the following command in Cloud Shell to create the **public** virtual machine. Replace <password> with a suitable password for the **azureuser** account.

```
Azure CLI
az vm create \
  --resource-group [sandbox resource group name] \
  --name public \
  --vnet-name vnet \
  --subnet publicsubnet \
  --image UbuntuLTS \
  --admin-username azureuser \
  --no-wait \
  --custom-data cloud-init.txt \
  --admin-password <password>
```

5. Run the following command to create the **private** virtual machine. Replace <password> with a suitable password.

```
Azure CLI
az vm create \
  --resource-group [sandbox resource group name] \
  --name private \
  --vnet-name vnet \
  --subnet privatesubnet \
  --image UbuntuLTS \
  --admin-username azureuser \
  --no-wait \
  --custom-data cloud-init.txt \
  --admin-password <password>
```

6. Run the following Linux watch command to check that the virtual machines are running. The watch command periodically runs the az vm list command so that you can monitor the progress of the virtual machines.

```
bash
watch -d -n 5 "az vm list \
  --resource-group [sandbox resource group name] \
  --show-details \
  --query '[*].{Name:name, ProvisioningState:provisioningState, PowerState:powerState}' \
  --output table"
```

A **ProvisioningState** value of "Succeeded" and a **PowerState** value of "VM running" indicate a successful deployment. When all three virtual machines are running, you're ready to move on. Select Ctrl-C to stop the command and continue with the exercise.

7. Run the following command to save the public IP address of the **public** virtual machine to a variable named PUBLICIP.

```
Azure CLI
```

```
PUBLICIP="$(az vm list-ip-addresses \
  --resource-group [sandbox resource group name] \
  --name public \
  --query "[].virtualMachine.network.publicIpAddresses[*].ipAddress" \
  --output tsv)"
```

```
echo $PUBLICIP
```

8. Run the following command to save the public IP address of the **private** virtual machine to a variable named PRIVATEIP.

Azure CLI = Copy

```
PRIVATEIP="$(az vm list-ip-addresses \
  --resource-group [sandbox resource group name] \
  --name private \
  --query "[].virtualMachine.network.publicIpAddresses[*].ipAddress" \
  --output tsv)"
echo $PRIVATEIP
```

st traffic routing through the network virtual appliance

nal steps use the Linux traceroute utility to show how traffic is routed. You'll use the ssh command to run traceroute on each virtual machine. rst test will show the route taken by ICMP packets sent from the **public** virtual machine to the **private** virtual machine. The second test now the route taken by ICMP packets sent from the **private** virtual machine to the **public** virtual machine.

1. Run the following command to trace the route from **public** to **private**. When prompted, enter the password for the **azureuser** account that you specified earlier.

bash = Copy

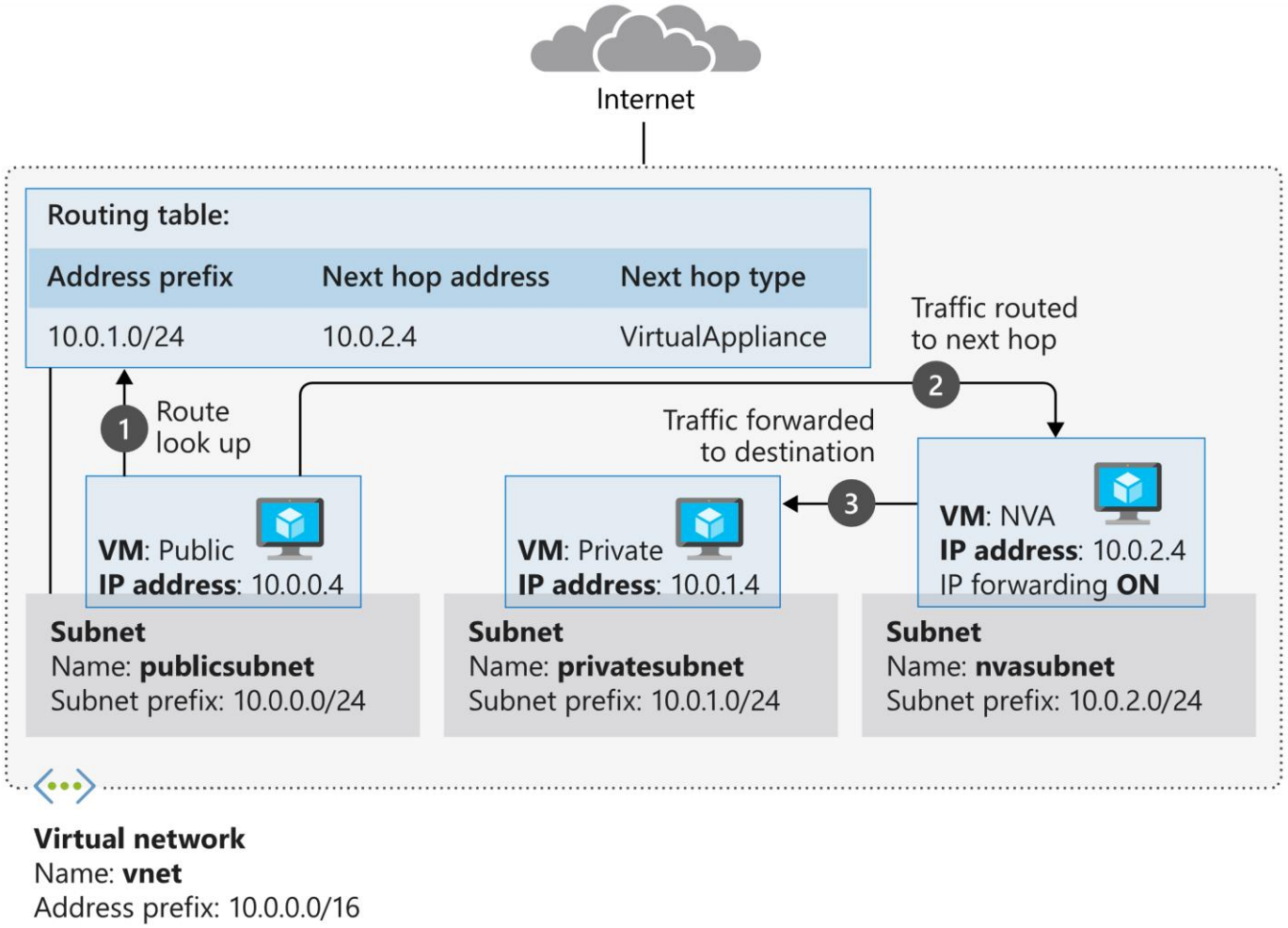
```
ssh -t -o StrictHostKeyChecking=no azureuser@$PUBLICIP 'traceroute private --type=icmp; exit'
```

If you receive the error message bash: traceroute: command not found, wait a minute and retry the command. The automated installation of traceroute can take a minute or two after virtual machine deployment. After the command succeeds, the output should look similar to the following example:

Text = Copy

```
traceroute to private.kzffavtrkpeulburui2lgywxwg.gx.internal.cloudapp.net (10.0.1.4), 64 hops max
1      10.0.2.4  0.710ms  0.410ms  0.536ms
2      10.0.1.4  0.966ms  0.981ms  1.268ms
Connection to 52.165.151.216 closed.
```

Notice that the first hop is to 10.0.2.4. This address is the private IP address of **nva**. The second hop is to 10.0.1.4, the address of **private**. In the first exercise, you added this route to the route table and linked the table to the **publicsubnet** subnet. So now all traffic from **public** to **private** is routed through the network virtual appliance.



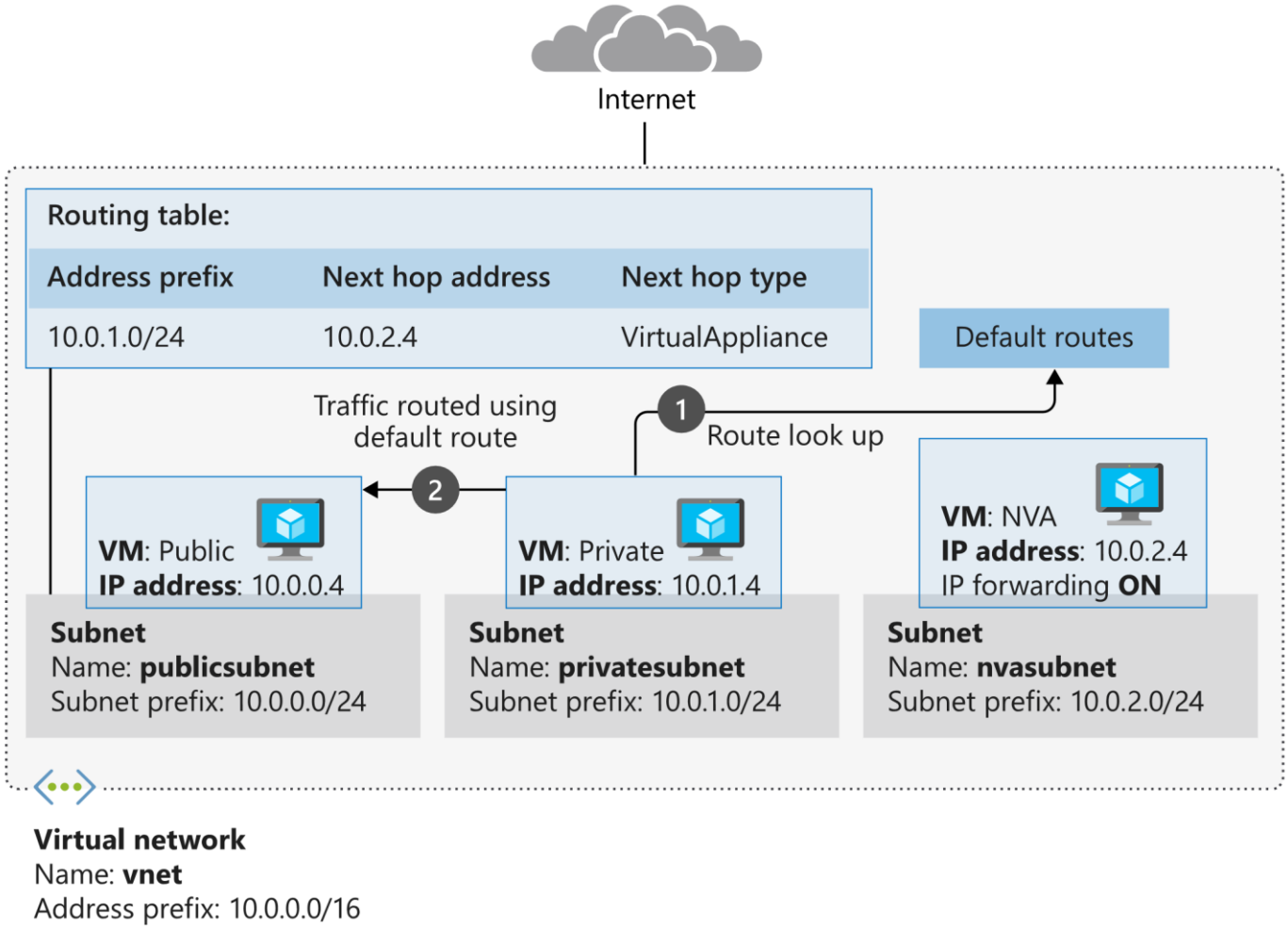
2. Run the following command to trace the route from **private** to **public**. When prompted, enter the password for the **azureuser** account.

```bash
ssh -t -o StrictHostKeyChecking=no azureuser@$PRIVATEIP 'traceroute public --type=icmp; exit'
```

You should see the traffic go directly to **public** (10.0.0.4) and not through the NVA, as shown in the following command output.

```Text
traceroute to public.kzffavtrkpeulburui2lgywxwg.gx.internal.cloudapp.net (10.0.0.4), 64 hops max
1   10.0.0.4  1.095ms  1.610ms  0.812ms
Connection to 52.173.21.188 closed.
```

The **private** virtual machine is using default routes, and traffic is routed directly between the subnets.



You've now configured routing between subnets to direct traffic from the public internet through the **dmzsubnet** subnet before it reaches the private subnet. In the **dmzsubnet** subnet, you added a virtual machine that acts as an NVA. You can configure this NVA to detect potentially malicious requests and block them before they reach their intended targets.

Next unit: Summary

Continue >

English (United States)

Azure Cloud Shell

This module requires a sandbox to complete. A sandbox gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any

other reason is prohibited, and may result in permanent

loss of access to the sandbox.

2/21/2020 Summary - Learn | Microsoft Docs

100 XP

Summary

1 minute

In this module, you learned how to customize routes in an Azure virtual network and how to redirect the traffic flow through a network virtua appliance. You also learned how to create your own custom network virtual appliance by deploying an Azure virtual machine.

Clean up

The sandbox automatically cleans up your resources when you're finished with this module.

When you're working in your own subscription, it's a good idea at the end of a project to identify whether you still need the resources you created. Resources left running can cost you money. You can delete resources individually or delete the resource group to delete the entire s resources.

Learn more

For more information on using routes in your network infrastructure, see the following articles:

- Virtual network traffic routing
- Tutorial: Route network traffic with a route table using the Azure porta
- Deploy highly available network virtual appliances
- Implement a DMZ between Azure and the Internet

Module complete:

Unlock achievement

1300 XP

Design an IP addressing schema for your Azure deployment

37 min • Module • 6 Units

V V V V V 4.7 (216) Rate it

Beginner Solutions Architect Azure Virtual Network

A good Azure IP addressing schema provides flexibility, room for growth, and integration with on-premises networks. The schema ensures that communication works for deployed resources, minimizes public exposure of systems, and gives the organization flexibility in its network. If not properly designed, systems might not be able to communicate, and additional work will be required to remediate.

In this module, you will:

Identify the private IP addressing capabilities of Azure virtual networks.
Identify the public IP addressing capabilities of Azure.
Identify the requirements for IP addressing when integrating with on-premises networks.

- Knowledge of basic networking concepts, network subnets, and IP addressing
- Familiarity with Azure virtual networking

This module is part of these learning paths
Architect network infrastructure in Azure

Introduction
2 min

Network IP addressing and integration
8 min

Public and private IP addressing in Azure
8 min

Plan IP addressing for your networks
7 min

Exercise - Design and implement IP addressing for Azure virtual networks
10 min

Summary
2 min

Unit 1 of 6 S Next T

100 XP

Introduction

2 minutes

Imagine you're the solution architect for a manufacturing company. Your company is beginning a project to move many services out of its existing datacenter and into the Azure cloud. The company wants to integrate the existing network with Azure. You need to plan the public and private IP addresses for the network carefully, so you don't run out of addresses and will have capacity for future growth. A good IP addressing scheme provides flexibility, room for growth, and integration with on-premises networks.

In this module, you'll learn about the public and private IP addressing capabilities of Azure virtual networks. You'll learn how to gather the necessary requirements for planning an IP address scheme. This module covers the on-premises integration methods of point-to-site and siteto-site, and also virtual network-to-virtual network peering. You'll also design and implement virtual networks, and configure and verify virtual network peering. By the end of this module, you'll understand how to plan IP addressing for an Azure network and how to integrate Azure with an on-premises network.

Learning objectives

In this module, you will:

- Identify the private IP addressing capabilities of Azure virtual networks.
- Identify the public IP addressing capabilities of Azure.
- Identify the requirements for IP addressing when integrating with on-premises networks.

Prerequisites

- Knowledge of basic networking concepts, network subnets, and IP addressing
- Familiarity with Azure virtual networking

Next unit: Network IP addressing and integration

Continue

200 XP

Network IP addressing and integration

8 minutes

To integrate resources in an Azure virtual network with resources in your on-premises network, you must understand how you can connect those resources and how to configure IP addresses.

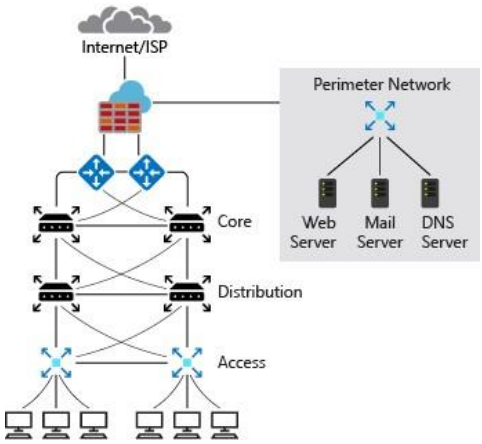
Your manufacturing company wants to migrate a business-critical database to Azure. Client applications on desktop computers, laptops, and mobile devices need constant access to the database as if the database remained in the on-premises network. You want to move the database server without affecting users.

In this unit, you will look at a typical on-premises network design and compare it to a typical Azure network design. You also will learn about requirements for IP addressing when integrating an Azure network with on-premises networks.

On-premises IP addressing

A typical on-premises network design includes these components:

- Routers
- Firewalls
- Switches
- Network segmentation



The diagram shows a simplified version of a typical on-premises network. On the routers facing the internet service provider (ISP), you have public IP addresses that are used by your outbound internet traffic as their source. These addresses also are used for inbound traffic across the internet. The ISP might assign you a block of IP addresses to assign to your devices, or you might have your own block of public IP addresses that your organization owns and controls. These addresses can be assigned to systems that you would like to make accessible from the internet, such as web servers.

The perimeter network and internal zone have private IP addresses. In the perimeter network and internal zone, the IP addresses that are assigned to these devices aren't accessible over the internet. The administrator has full control over the IP address assignment, name resolution, security settings, and security rules. There are three ranges of non-routable IP addresses that are designed for internal networks that won't be sent over internet routers:

- 10.0.0.0 to 10.255.255.255
- 172.16.0.0 to 172.31.255.255

192.168.0.1 to 192.168.255.255

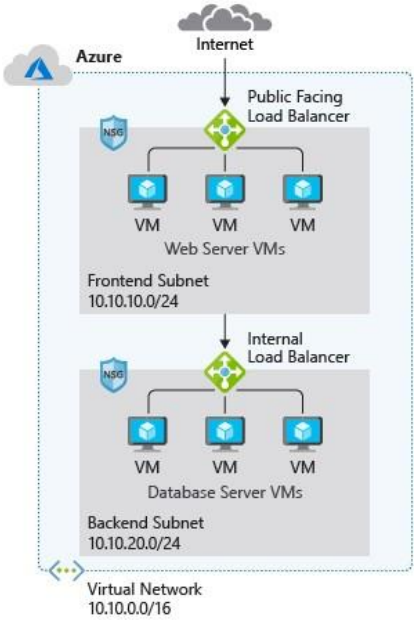
The administrator can add or remove on-premises subnets to accommodate network devices and services. The number of subnets and IP addresses you can have in your on-premises network depends on the Classless Inter-Domain Routing (CIDR) for the IP address block.

Azure IP addressing

Azure virtual networks use private IP addresses. The ranges of private IP addresses are the same as for on-premises IP a
premises networks, the administrator has full control over the IP address assignment, name resolution, security setting
Azure virtual network. The administrator can add or remove subnets depending on the CIDR for the IP address block.

In a typical Azure network design, we usually have these components:

- Virtual networks
- Subnets
- Network security groups
- Firewalls
- Load balancers



In Azure, the network design has features and functions that are similar to an on-premises network, but the structure o
The Azure network does not follow the typical on-premises hierarchical network design. The Azure network provides th
scale down infrastructure based on demand. Provisioning in the Azure network happens in a matter of seconds. There a
like routers or switches. The entire infrastructure is virtual and you slice it into chunks that suit your requirements.

In Azure, you typically would implement a network security group and a firewall. You use subnets to isolate front-end s
servers and DNS, and back-end services like databases and storage systems. Network security groups filter internal and
network layer. A firewall has more extensive capabilities for network layer filtering and application layer filtering. By de
security groups and a firewall, you get improved isolation of resources for a secure network architecture.

Basic properties of Azure virtual networks

A virtual network is your network in the cloud. You can divide your virtual network into multiple subnets. Each subnet h
address space that is assigned to your virtual network. You can add, remove, expand, or shrink a subnet if there are no
in it.

By default, all subnets in an Azure virtual network can communicate with each other. However, you can use a network s
communication between subnets. The smallest subnet that is supported uses a /29 subnet mask. The largest supported
mask.

Integrate Azure with on-premises networks

Before you start integrating Azure with on-premises networks, it's important to identify the current private IP address s
premises network. There can be no IP address overlap for interconnected networks.

For example, you can't use 192.168.0.0/16 on your on-premises network and use 192.168.10.0/24 on your Azure virtual network. These ranges both contain the same IP addresses and won't be able to route traffic between each other.

You can, however, have the same class range for multiple networks. For example, you can use the 10.10.0.0/16 address space for your onpremises network and the 10.20.0.0/16 address space for your Azure network because they don't overlap.

It is vital to check for overlaps when you're planning an IP address scheme. If there's an overlap of IP addresses, you can't integrate your onpremises network with your Azure network.

Check your knowledge

1. Which objects are required when you connect a virtual network to an on-premises network by using an Azure VPN gateway?

- A dedicated leased line
- A dedicated subnet for the gateway

  A dedicated subnet assigns an IP address to devices connected through the tunnel. You must have a dedicated subnet named GatewaySubnet for the VPN gateway.
- A dedicated network security group

2. Which of the following IP address ranges is routable over the internet?

- 10.0.0.0 to 10.255.255.255
- 215.11.0.0 to 215.11.255.255

  Correct, this is address range is routable over the internet.
- 172.16.0.0 to 172.31.255.255
- 192.168.0.1 to 192.168.255.255

Next unit: Public and private IP addressing in Azure

Continue

200 XP

Public and private IP addressing in Azure

8 minutes

In your manufacturing company, you are moving resources into Azure, starting with a database server. You want to ensure that the database server is accessible for clients in your on-premises network. Public resources like web servers must be accessible from the internet. You want to ensure that you plan IP addresses that support these requirements.

In this unit, you'll explore the constraints and limitations for public and private IP addresses in Azure. You also will look at the capabilities that are available in Azure to reassign IP addresses in your network.

IP address types

There are two types of IP addresses that you can use in Azure:

- **Public IP addresses**
- **Private IP addresses**

Both types of IP addresses can be allocated in one of two ways:

- **Dynamic**
- **Static**

Let's take a closer look at how the IP address types work together.

Public IP addresses

Use a public IP address for public-facing services. A public address can be either static or dynamic. A public IP address can be assigned to a VM, an internet-facing load balancer, a VPN gateway, or an application gateway.

Dynamic public IP addresses are assigned addresses that can change over the lifespan of the Azure resource. The dynamic IP address is allocated when you create or start a VM. The IP address is released when you stop or delete the VM. In each Azure region, public IP addresses are assigned from a unique pool of addresses. The default allocation method is dynamic.

Static public IP addresses are assigned addresses that will not change over the lifespan of the Azure resource. To ensure that the IP address for the resource remains the same, you can set the allocation method explicitly to static. In this case, an IP address is assigned immediately. It is released only when you delete the resource or change the IP allocation method to dynamic.

Basic and Standard SKUs

For public IP addresses, there are two types of SKUs to choose from: **Basic** and **Standard**. All public IP addresses created before the introduction of SKUs are Basic SKU public IP addresses. With the introduction of SKUs, you have the option to specify which SKU you would like the public IP address to be.

Basic

Basic public IPs can be assigned by using static or dynamic allocation methods. Basic IPs have an adjustable inbound originated flow idle timeout of 4-30 minutes, with a default of 4 minutes, and a fixed outbound originated flow idle timeout of 4 minutes. Basic IPs are open by default. We recommend that you use network security groups to restrict inbound or outbound traffic. Network security groups are recommended but optional for restricting inbound or outbound traffic.

Basic public IPs can be assigned to any Azure resource that can be assigned a public IP address, such as network interfaces, VPN gateways, application gateways, and internet-facing load balancers. They do not support availability zone scenarios. You must use a Standard SKU public IP for an availability zone scenario.

Standard

Standard SKU public IP addresses always use the static allocation method. They have an adjustable inbound originated flow idle timeout of 4-30 minutes, with a default of 4 minutes, and a fixed outbound originated flow idle timeout of 4 minutes.

Standard IPs are secure by default and closed to inbound traffic. You must explicitly allow inbound traffic by using a network security group.

Standard IPs can be assigned to network interfaces, Standard public load balancers, application gateways, or VPN gateways. For more information about Standard load balancers, see Azure Standard Load Balancer overview. Standard IPs are zone-redundant by default and optionally zonal (they can be created zonal and guaranteed in a specific availability zone).

Public IP address prefix

You can't bring your own public IP addresses from on-premises networks into Azure. Based on the location of the resource, an IP address is assigned from a pool of available addresses. Public IP addresses are allocated from a range that's unique to each region in each Azure cloud. Public IP addresses can't be moved between regions; all IP addresses are region-specific. If your business needs to have datacenters in different regions, you would have a different public IP address range for each region. You can use technology like Azure Traffic Manager to balance between region-specific instances.

To ensure a static range of public IP addresses, you can create a public IP address prefix. You can't specify the addresses when you create the prefix, but after the prefix is created, the addresses will be fixed. The IP addresses will be a contiguous range. The advantage of a public IP address prefix is that you can specify firewall rules for these IP addresses with the knowledge that they will not change. You can assign the addresses from a public IP address prefix to any resource in Azure that supports public IP addresses.

Private IP addresses

Private IP addresses are used for communication within a virtual network. Private IP addresses are used within Azure Virtual Network virtual networks and your on-premises networks. They can be set to dynamic (DHCP lease) or static (DHCP reservation).

Dynamic private IP addresses are assigned through a DHCP lease and can change over the lifespan of the Azure resource.

Static private IP addresses are assigned through a DHCP reservation and do not change throughout the lifespan of the Azure resource. Static private IP addresses persist if a resource is stopped or deallocated.

-

IP addressing for Azure virtual networks

A virtual network is a fundamental component that acts as an organization's network in Azure. In the virtual network, the administrator has full control over the IP address assignment, security settings, and security rules. When you create a virtual network, you define a scope of IP addresses. Private IP addressing in Azure works the same way as it does in the on-premises network. You choose the private IP addresses that are reserved by the Internet Assigned Numbers Authority (IANA) based on your network requirements:

10.0.0.0/8
172.16.0.0/12

-
-

192.168.0.0/16

A subnet is a range of IP address within the virtual network. You can divide the virtual network into multiple subnets. Each subnet must have a unique address range, which is specified in classless inter-domain routing (CIDR) format. CIDR is a way to represent a block of network IP addresses. An IPv4 CIDR, specified as part of the IP address, shows the length of the network prefix.

Consider, for example, the CIDR 192.168.10.0/24. "192.168.10.0" is the network address. The "24" indicates that the first 24 bits are part of the network address, leaving the last 8 bits for specific host addresses. The address range can't overlap with other subnets in the virtual network or with the on-premises network.

The first three IP addresses are reserved for all subnets by default in Azure. For protocol conformance, the first and last IP addresses of all subnets also are reserved. An internal DHCP service within Azure assigns and maintains the lease of IP addresses. The .1, .2, .3, and last IP addresses are not visible or configurable by the Azure customer. These addresses are reserved and used by internal Azure services.

In Azure virtual networks, IP addresses can be allocated to the following types of resources:

Virtual machine network interfaces
Load balancers
Application gateways

Check your knowledge

1 Which of the following resources can you assign a public IP address to?

- [x] A virtual machine

Correct. You can assign public IP addresses to virtual

- [ ] Azure Data Lake
- [ ] Azure Key Vault

2 What must a virtual machine have to communicate with the other resources in the same virtual network?

- [ ] Load balancer
- [ ] Network security group
- [x] Network interface

Correct. An IP address is assigned to a network interface, which is assigned to a virtual

Next unit: Plan IP addressing for your networks

Continue

100 XP

Plan IP addressing for your networks

7 minutes

In your manufacturing company, you have asked the operations and engineering teams about their requirements for the number of virtual machines in Azure. You've also asked them about their plans for expansion. Based on the results of this survey, you want to plan an IP addressing scheme that you won't have to change in the foreseeable future.

In this unit, you'll explore the requirements for a network IP address scheme. You'll learn about classless inter-domain routing (CIDR) and how you use it to slice an IP block to meet your addressing needs. At the end of the module, there's an exercise that shows how to plan IP addressing for Azure virtual networks.

Gather your requirements

Before planning your network IP address scheme, you must gather the requirements for your infrastructure. These requirements also will help you prepare for future growth by reserving extra IP addresses and subnets.

Here are two of the questions you might ask to discover the requirements:

- How many devices do you have on the network?
- How many devices are you planning to add to the network in the future?

When your network expands, you don't want to redesign the IP address scheme. Here are some other questions you could ask:

- Based on the services running on the infrastructure, what devices do you need to separate?
- How many subnets do you need?
- How many devices per subnet will you have?
- How many devices are you planning to add to the subnets in future?
- Are all subnets going to be the same size?
- How many subnets do you want or plan to add in future?

You'll need to isolate some services. Isolation of services provides an additional layer of security, but also requires good planning. For example, your front-end servers can be accessed by public devices, but the back-end servers need to be isolated. Subnets help isolate the network in Azure. However, by default, all subnets within a virtual network can communicate with each other in Azure. To provide further isolation, you can use a network security group. You might isolate services depending on the data and its security requirements. For example you might choose to isolate HR data and the company's financial data from customer databases.

When you know the requirements, you'll have a greater understanding of the total number of devices on the network per subnet and how many subnets you'll need. CIDR allows more flexible allocation of IP addresses than was possible with the original system of IP address classes.
Depending on your requirements, you'll slice the IP block into the required subnets and hosts.

Remember that Azure uses the first three addresses on each subnet. The first and last IP addresses of the subnets also are reserved for protocol conformance. Therefore, the number of possible addresses on an Azure subnet is **2^n-5**, where **n** represents the number of host bits.

Next unit: Exercise - Design and implement IP addressing for Azure virtual networks

Continue

https://docs.microsoft.com/en-us/learn/modules/design-ip-addressing-for-azure/4-plan-design-ip-addressing 2/2

R Previous Unit 5 of 6 S Next T

600 XP

Exercise - Design and implement IP addressing for Azure virtual networks

10 minutes

This module requires a sandbox to complete. A **sandbox** gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox. Activate sandbox

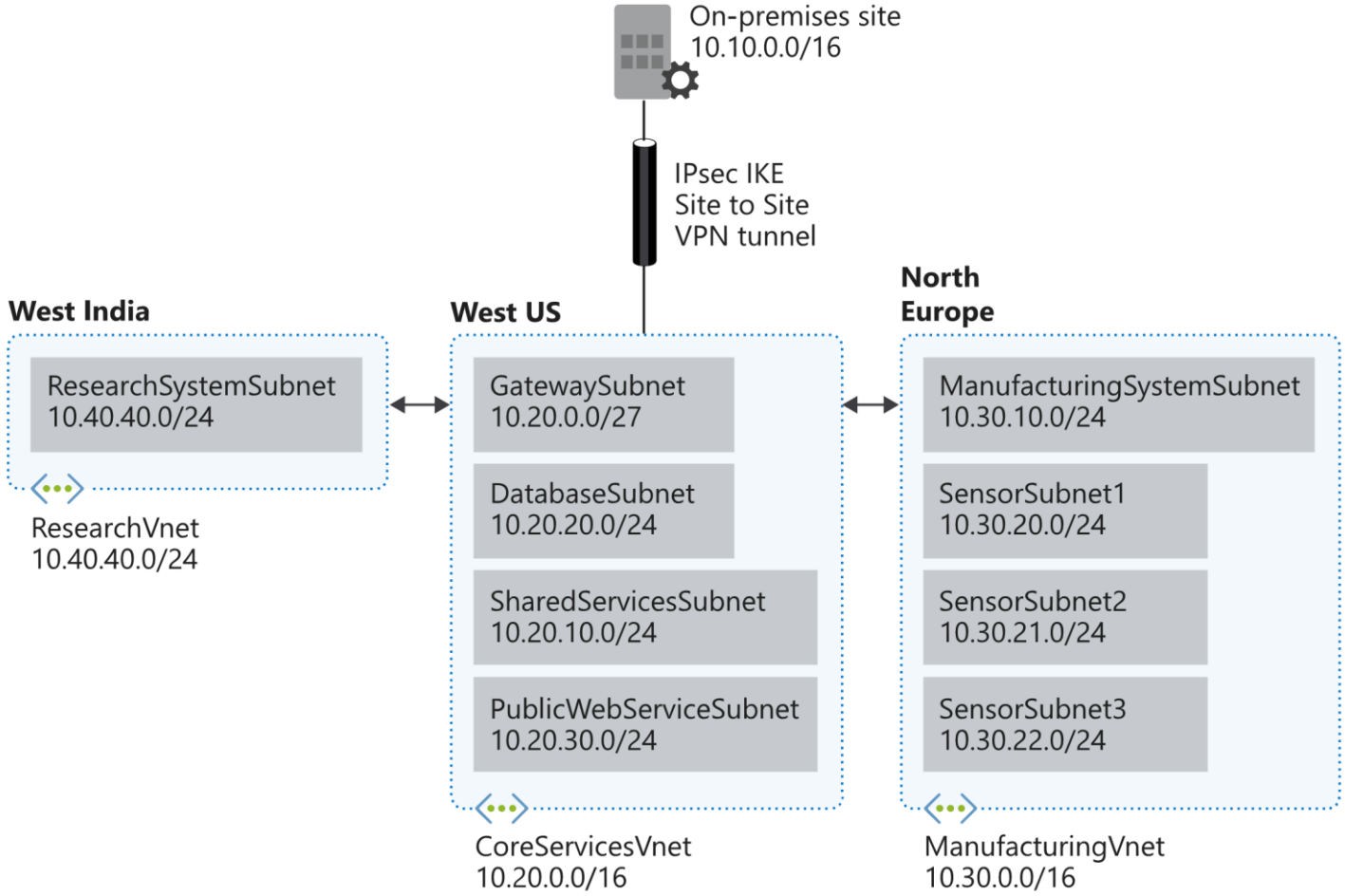
Now, you're ready to create and deploy some virtual networks with the IP addresses that you planned.

In this unit, you will implement three virtual networks and subnets to support resources in those virtual networks.

The **CoreServicesVnet** virtual network is deployed in the **US West** region. This virtual network will have the largest number of resources. It will have connectivity to on-premises networks through a VPN connection. This network will have web services, databases, and other systems that are key to the operations of the business. Shared services, such as Azure Active Directory (Azure AD) domain controllers and DNS also will be located here. A large amount of growth is anticipated, so a large address space is necessary for this virtual network.

The **ManufacturingVnet** virtual network is deployed in the **North Europe** region, near the location of your organization's manufacturing facilities. This virtual network will contain systems for the operations of the manufacturing facilities. The organization is anticipating a large number of internal connected devices for their systems to retrieve data from, such as temperature, and will need an IP address space that it can expand into.

The **ResearchVnet** virtual network is deployed in the **West India** region, near the location of the organization's research and development team. The research and development team uses this virtual network. The team has a small, stable set of resources that is not expected to grow. The team needs a small number of IP addresses for a few virtual machines for their work.



You will create the following resources:

Virtual network	Region	Virtual network address space	Subnet	Subnet address space
CoreServicesVnet	West US	10.20.0.0/16	-	-
			GatewaySubnet	10.20.0.0/27
			SharedServicesSubnet	10.20.10.0/24
			DatabaseSubnet	10.20.20.0/24
			PublicWebServiceSubnet	10.20.30.0/24
ManufacturingVnet	North Europe	10.30.0.0/16	-	-
			ManufacturingSystemSubnet	10.30.10.0/24
			SensorSubnet1	10.30.20.0/24
			SensorSubnet2	10.30.21.0/24
			SensorSubnet3	10.30.22.0/24
ResearchVnet	West India	10.40.40.0/24	-	-
			ResearchSystemSubnet	10.40.40.0/24

These virtual networks and subnets are structured in a way that accommodates existing resources yet allows for projected growth. Let's create these virtual networks and subnets to lay the foundation for our networking infrastructure.

Create the *CoreServicesVnet* virtual network

1. In Azure Cloud Shell, run the following command to create the **CoreServicesVnet** virtual network:

```
Azure CLI
```

```
az network vnet create \
    --resource-group [sandbox resource group name] \
    --name CoreServicesVnet \
    --address-prefix 10.20.0.0/16 \
    --location westus
```

2. Now, let's create the subnets that we need for the planned resources in the virtual network:

```
Azure CLI
```

```
az network vnet subnet create \
    --resource-group [sandbox resource group name] \
    --vnet-name CoreServicesVnet \
    --name GatewaySubnet \
    --address-prefixes 10.20.0.0/27

az network vnet subnet create \
    --resource-group [sandbox resource group name] \
    --vnet-name CoreServicesVnet \
    --name SharedServicesSubnet \
    --address-prefixes 10.20.10.0/24

az network vnet subnet create \
    --resource-group [sandbox resource group name] \
    --vnet-name CoreServicesVnet \
    --name DatabaseSubnet \
    --address-prefixes 10.20.20.0/24

az network vnet subnet create \
    --resource-group [sandbox resource group name] \
    --vnet-name CoreServicesVnet \
    --name PublicWebServiceSubnet \
    --address-prefixes 10.20.30.0/24
```

3. Let's take a look at what we have created. Run this command to show all the subnets that we configured:

```
Azure CLI
```

```
az network vnet subnet list \
    --resource-group [sandbox resource group name] \
    --vnet-name CoreServicesVnet \
    --output table
```

You should see the following subnets listed:

```
output
```

```
AddressPrefix    Name                   ProvisioningState    ResourceGroup
---------------  ---------------------  -------------------  ----------------------------------------
10.20.0.0/27     GatewaySubnet          Succeeded            [sandbox resource group name]
10.20.10.0/24    SharedServicesSubnet   Succeeded            [sandbox resource group name]
10.20.20.0/24    DatabaseSubnet         Succeeded            [sandbox resource group name]
10.20.30.0/24    PublicWebServiceSubnet Succeeded            [sandbox resource group name]
```

Create the *ManufacturingVnet* virtual network

1. In Cloud Shell, run the following command to create the **ManufacturingVnet** virtual network:

```
Azure CLI
```

```
az network vnet create \
  --resource-group [sandbox resource group name] \
  --name ManufacturingVnet \
  --address-prefix 10.30.0.0/16 \
  --location northeurope
```

2. Now, let's create the subnets that we need for the planned resources in the virtual network:

```
Azure CLI
```

```
az network vnet subnet create \
  --resource-group [sandbox resource group name] \
  --vnet-name ManufacturingVnet \
  --name ManufacturingSystemSubnet \
  --address-prefixes 10.30.10.0/24
 az network vnet subnet create \
  --resource-group [sandbox resource group name] \
  --vnet-name ManufacturingVnet \
  --name SensorSubnet1 \
  --address-prefixes 10.30.20.0/24

az network vnet subnet create \
  --resource-group [sandbox resource group name] \
  --vnet-name ManufacturingVnet \
  --name SensorSubnet2 \
  --address-prefixes 10.30.21.0/24
 az network vnet subnet create \
  --resource-group [sandbox resource group name] \
  --vnet-name ManufacturingVnet \    --name
SensorSubnet3 \
  --address-prefixes 10.30.22.0/24
```

3. Let's take a look at what we have created. Run this command to show all the subnets that we configured:

```
Azure CLI
```

```
az network vnet subnet list \
  --resource-group [sandbox resource group name] \
  --vnet-name ManufacturingVnet \
  --output table
```

You should see the following subnets listed:

```
Azure CLI
```

```
AddressPrefix    Name                 ProvisioningState    ResourceGroup
---------------  -------------------  -----------------  ----------------------------------------
10.30.10.0/24    ManufacturingSystemSubnet  Succeeded          [sandbox resource group name] 10.30.20.0/24    SensorSubnet1
Succeeded          [sandbox resource group name] 10.30.21.0/24   SensorSubnet2          Succeeded          [sandbox resource group
name]
10.30.22.0/24   SensorSubnet3          Succeeded          [sandbox resource group name]
```

Create the *ResearchVnet* virtual network

1. In Cloud Shell, run the following command to create the **ResearchVnet** virtual network:

```
Azure CLI
```

```
az network vnet create \
  --resource-group [sandbox resource group name] \
```

2.

```
    --name ResearchVnet \
    --address-prefix 10.40.40.0/24 \
    --location westindia
```

Now, let's create the subnets that we need for the planned resources in the virtual network:

Azure CLI = Copy

3.

```
az network vnet subnet create \
  --resource-group [sandbox resource group name] \
  --vnet-name ResearchVnet \
  --name ResearchSystemSubnet \
  --address-prefixes 10.40.40.0/24
```

Let's take a look at the final virtual network. Run this command to show all the subnets that we configured:

Azure CLI = Copy

```
az network vnet subnet list \
  --resource-group [sandbox resource group name] \
  --vnet-name ResearchVnet \
  --output table
```

You should see the following subnets listed:

Azure CLI = Copy

```
AddressPrefix   Name              ProvisioningState   ResourceGroup
--------------  ----------------  ------------------  ----------------------------------------
10.40.40.0/24   ResearchSystemSubnet  Succeeded         [sandbox resource group name]
```

that you have created the virtual networks and subnets, you have the infrastructure on which you can deploy resources.

These networks can be further integrated through virtual network peering and through Azure VPN Gateway to connect to on-premises networks.

:an use network security groups to filter traffic and control access within and between virtual networks.

Azure Cloud Shell

This module requires a sandbox to complete. A sandbox gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

Previous Unit 6 of 6

100 XP

Summary

2 minutes

In this module, you have:

- Identified the private and public IP addressing capabilities of Azure virtual networks.
- Identified how to integrate on-premises networks with Azure.
- Planned an IP address scheme for your Azure infrastructure and created virtual networks.

Now that you understand how to plan IP addressing for Azure networks, you understand the private and public IP addressing capabilities of Azure virtual networks. You can use this information to plan out the IP addressing for your own Azure infrastructure.

Learn more

For more information about IP addressing in Azure, see the following articles:

- IP address types and allocation methods in Azure
- Public IP address prefix

Module incomplete:

Go back to finish

2/21/2020 Design a hybrid network architecture on Azure - Learn | Microsoft Docs

1000 XP

Design a hybrid network architecture on Azure

47 min • Module • 6 Units

V V V V W 4.6 (175)

Beginner Solutions Architect Azure ExpressRoute Virtual Network VPN Gateway

You have a traditional on-premises infrastructure that you need to connect to resources in Azure. In this module, you learn how to select a connectivity method for your use cases that balances functionality, cost, and security.

In this module, you will:

Identify various methods for integrating on-premises networks with Azure.
Identify the features and limitations of each method of network integration.
Select the appropriate hybrid network solution for specified use cases.

- Knowledge of networking fundamentals
- Familiarity with Azure virtual networking

This module is part of these learning paths
Architect network infrastructure in Azure

Introduction
5 min

On-premises network integration on Azure
10 min

Use Azure ExpressRoute for hybrid networks
10 min

Work with hybrid networks on Azure
10 min

Choose a hybrid network architecture on Azure
10 min

Summary
2 min

2/21/2020 Introduction - Learn | Microsoft Docs

Unit 1 of 6 S

Next T

100 XP

Introduction

5 minutes

You work for a global enterprise that is planning a migration to Azure. You need to integrate resources in Azure with your on-premises networks, but you're unsure which technology is best suited for your needs.

Your company has a central datacenter and several remote offices that need connectivity to Azure. After completing your research, you'll understand the pros and cons of your connectivity options, and you'll know how and when to select the various technologies for hybrid connectivity.

In this module, you explore the hybrid networking capabilities in Azure. The module looks at services for on-premises integration, using tools such as Azure ExpressRoute. It covers the scenarios that apply to the design of your company's hybrid architecture.

Learning objectives

In this module, you will:

- Identify the methods of integrating on-premises networks with Azure.
- Identify the features and limitations of each method of network integration.
- Select the appropriate hybrid network solution for specific use cases.

Prerequisites

- Knowledge of basic networking concepts (for example, subnets and IP addressing)
- Familiarity with Azure virtual networking

Next unit: On-premises network integration on Azure

Continue

200 XP

On-premises network integration on Azure

10 minutes

Your company plans to migrate most of its on-premises resources to Azure. However, a small datacenter must remain on-premises to be integrated into the Azure network. The architectural model needs to consider using Azure network connectivity for several satellite offices. You want to use a hybrid network architecture that grants access to both your on-premises and cloud-based resources.

To handle the migration, you'll produce a network integration plan for Azure that includes a selection of the best hybrid network options available in Azure. The options must meet the organization's requirements for hybrid connectivity.

In this unit, you'll explore on-premises connectivity on the Azure platform. You'll also get an overview of Azure Virtual Network, and see how to use Azure VPN Gateway to secure traffic to an on-premises network.

About Azure Virtual Network

The Azure Virtual Network service has a specific set of tools and resources for building a cloud-based network architecture for your organization.
Azure virtual networks provide a secure virtual communication channel for all permitted Azure resources within your subscription.

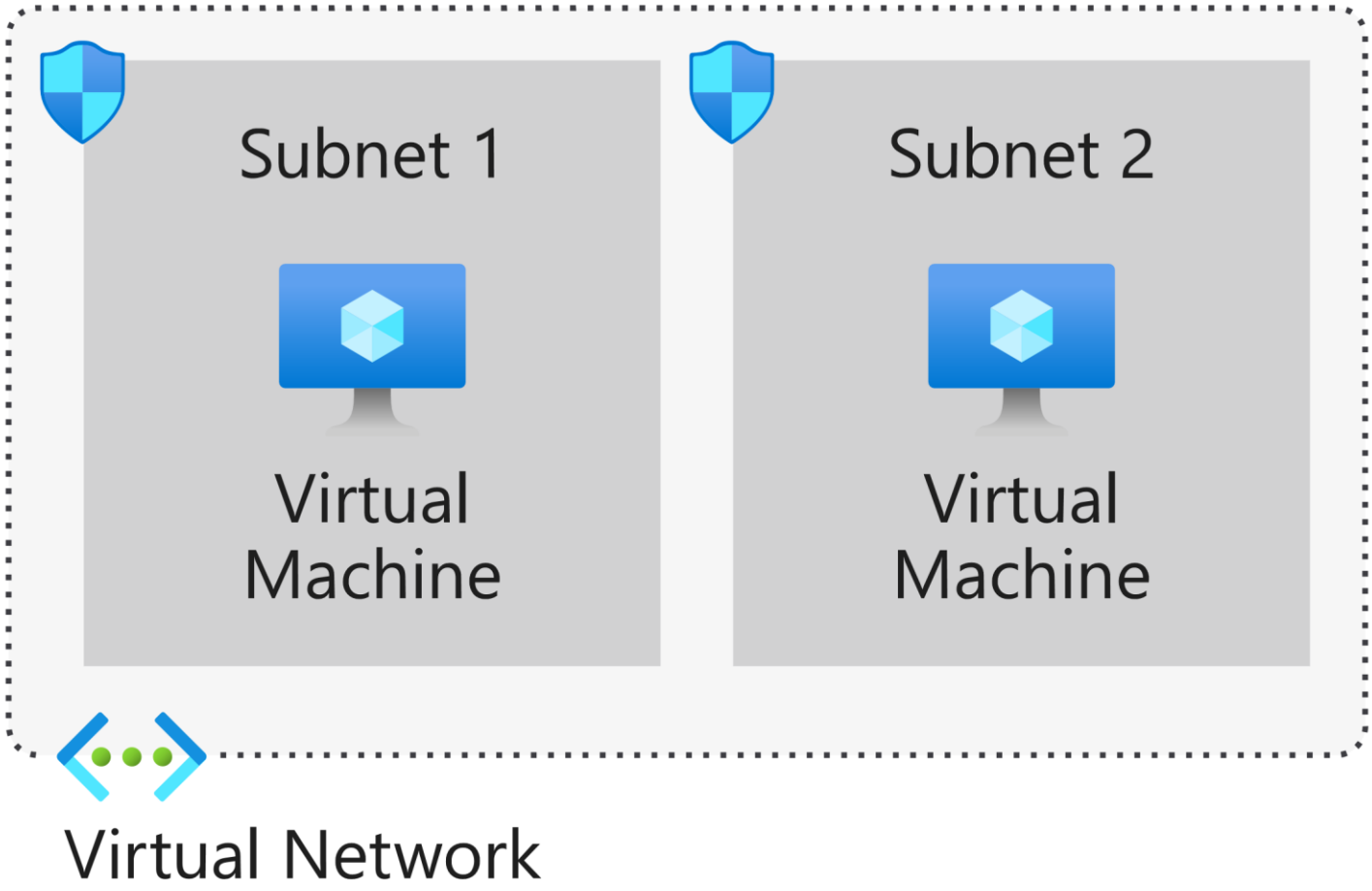
With an Azure virtual network, you can:

- Connect virtual machines to the internet.
- Provide secure communications between Azure resources that are hosted in various datacenters and regions.
- Isolate and manage Azure resources.
- Connect to on-premises computers.

Manage network traffic.

By default, all Azure resources within a virtual network have outbound connectivity to the internet. External inbound communication must come through a public-facing endpoint. All internal resources use a private endpoint to access the virtual network.

A virtual network is composed of many elements including, but not limited to, network interfaces, load balancers, subnets, network security groups, and public IP addresses. These elements work together and enable secure, reliable network communication between your Azure resources, the internet, and on-premises networks.



Traffic routing on an Azure virtual network

Outbound traffic from a subnet is routed based on the destination IP address. A routing table defines how the traffic routes and what happens next. A destination IP address can exist across multiple routing table prefix definitions (for example, 10.0.0.0/16 and 10.0.0.0/24). The router uses a sophisticated algorithm to find the longest prefix match. Traffic that's heading for a 10.0.0.6 address would resolve to the 10.0.0.0/24 prefix and be routed accordingly.

There are two principal routing tables: system and custom.

System routing tables

Azure automatically creates a set of default routing tables for the virtual network and each subnet mask within the virtual network. These system routes are fixed and can't be overridden or deleted. However, you can override the default settings by using a custom routing table.

A typical default routing table might look like this:

Source	Address prefixes	Next hop type
Default	Unique to the virtual network	Virtual network
Default	0.0.0.0/0	Internet
Default	10.0.0.0/8	None
Default	172.16.0.0/12	None
Default	192.168.0.0/16	None
Default	100.64.0.0/10	None

A routing table is made up of a source, an address prefix, and a next hop. All traffic that leaves the subnet uses the routing table where it should go next. In effect, the traffic looks for the next hop in its journey.

A next hop defines what happens to the traffic flow next, based on the prefix. There are three types of next hop:

- **Virtual network**: The traffic is routed according to the IP address within the virtual network.
- **Internet**: The traffic is routed to the internet.
- **None**: The traffic is dropped.

Custom routing tables

Apart from system-defined routing tables, you can also create custom routing tables. These user-defined routing tables system table. There are limitations on the number of routing items you can have in a custom table.

A few of the many limitations that apply to virtual networks are listed in the following table:

Resource	Default or maximum number
Virtual networks	1,000
Subnets per virtual network	3,000
Virtual network peerings per virtual network	500
Private IP addresses per virtual network	65,536

Much like the system routing table, custom routing tables also have a next hop type. But the custom routing tables offe

- **Virtual appliance**: This option is usually a virtual machine that runs a specific network application, such as a firewall.
- **Virtual network gateway**: Use this option when you want to send traffic to a virtual network gateway. A virtual networ be VPN. The type can't be Azure ExpressRoute, which requires setting a Border Gateway Protocol (BGP) routing pro
- **None**: This option drops the traffic rather than forwarding it.
- **Virtual network**: This option lets you override a default system routing.
- **Internet**: This option lets you specify that any prefix forwards traffic to the internet.

Connect Azure virtual networks

You can connect your virtual networks in any of several ways. You can use Azure VPN Gateway or ExpressRoute, or you can use the method directly.

Azure VPN Gateway

When you're working toward integrating your on-premises network with Azure, there needs to be a bridge between th Azure service that provides this functionality. A VPN gateway can send encrypted traffic between the two networks. VP multiple connections, which enable them to route VPN tunnels that use any available bandwidth. A virtual network can assigned. VPN gateways can also be used for connections between virtual networks in Azure.

Implementing a VPN gateway requires two or more virtual machines to be deployed to the subnet that you create whe network. In this instance, the subnet is also known as the gateway subnet. Each virtual machine is assigned a default co and gateway services, explicit to the provisioned gateway. You can configure these virtual machines directly.

When you create a gateway, several topologies are available. These topologies, also known as gateway types, determin the expected connection type.

Site-to-

You use a site-to-site connection for cross-premises and hybrid-network configurations. This connection topology requi device to have a publicly accessible IP address, and must not be behind a NAT. The connection uses a secret ASCII string to authenticate between the gateway and the VPN device.

Multisite

A multisite connection is similar to a site-to-site connection, but with a slight variation. Multisite supports multiple VPN connections to your on-premises VPN devices. This connection topology requires a RouteBased VPN known as a dynamic gateway. It's important to note that, with a multisite configuration, all connections route through and share all available bandwidth.

Point-to-site

A point-to-site connection is suited to a remote individual client device that connects to your network. You must authenticate the client device either through Azure Active Directory or by using Azure certificate authentication. This model suits home working scenarios.

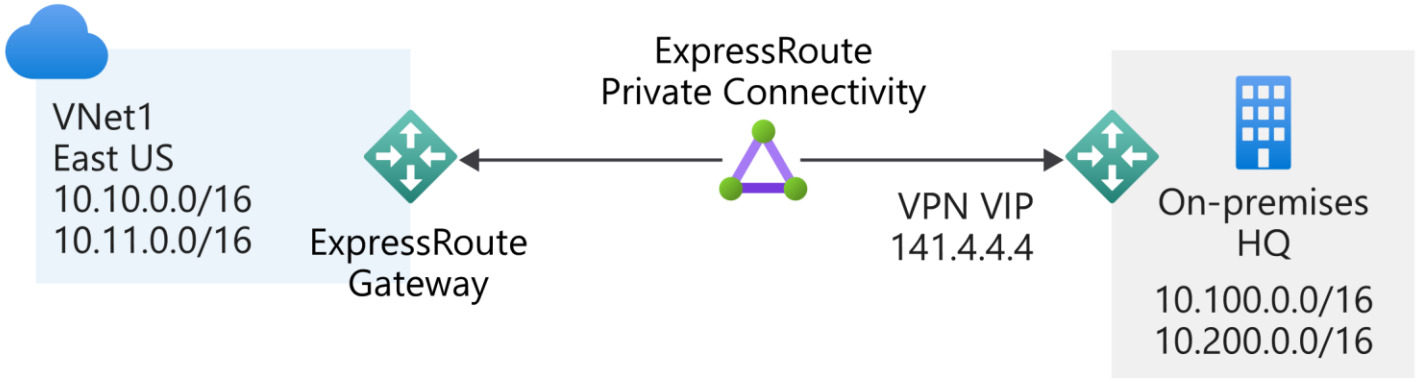
Network-to-network

You use a network-to-network connection to create connections between multiple Azure virtual networks. This connection topology, unlike the others, doesn't require a public IP or VPN device. It can also be used in a multisite configuration to establish combined cross-premises connections with inter-virtual network connectivity.

ExpressRoute

ExpressRoute creates a direct connection between your on-premises network and the Azure virtual network that doesn't use the internet. You use ExpressRoute to seamlessly extend your local network across to the Azure virtual network space. The ExpressRoute service is offered by many third-party connectivity providers. There are three different ExpressRoute connection types:

- CloudExchange colocation
- Point-to-point Ethernet connection
- Any-to-any (IPVPN) connection

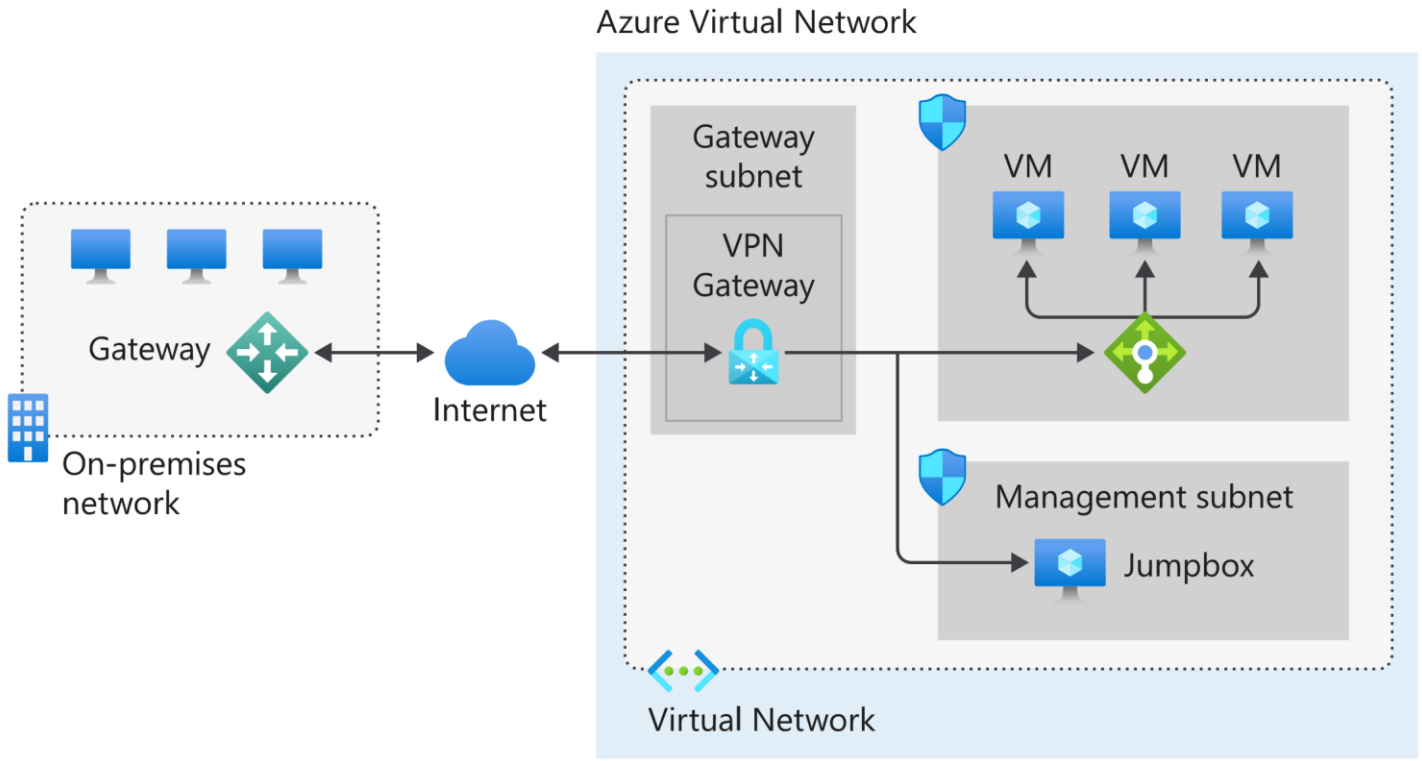


Peering

Virtual networks can peer across subscriptions and Azure regions. After the virtual networks are peered, resources in these networks communicate with each other as if they're in the same network. The traffic is routed between resources in a peered virtual network by using only private IP addresses. Routing is achieved by routing traffic through the Azure network and keeping the connection private as part of the Azure backbone network. The backbone network provides low latency and high-bandwidth network connections.

Site-to-site VPN gateway reference architecture

Although many reference architectures are available when you design a hybrid network, one popular architecture is the site-to-site configuration. The simplified reference architecture shown in the following diagram illustrates how you would connect an on-premises network to the Azure platform. The internet connection uses an IPsec VPN tunnel.



The architecture features several components:

- The **on-premises network** represents your on-premises Active Directory and any data or resources.
- The **gateway** is responsible for sending encrypted traffic to a virtual IP address when it uses a public connection.
- The **Azure virtual network** holds all your cloud applications and any Azure VPN gateway components.
- An **Azure VPN gateway** provides the encrypted link between the Azure virtual network and your on-premises network. An Azure VPN gateway is made up of these elements:
  - Virtual network gateway
  - Local network gateway
  - Connection
  - Gateway subnet
- **Cloud applications** are the ones you've made available through Azure.
- An **internal load balancer**, located in the front end, routes cloud traffic to the correct cloud-based application or resource.

Using this architecture offers several benefits, including:

- Configuration and maintenance are simplified.
- Having a VPN gateway helps ensure that all data and traffic are encrypted between the on-premises gateway and the Azure gateway.
- The architecture can be scaled and extended to meet your organization's networking needs.

This architecture isn't applicable in all situations, because it uses an existing internet connection as the link between the two gateway points. Bandwidth constraints can cause latency issues that result from reuse of the existing infrastructure.

Check your knowledge

1. Where is a point-to-site VPN connection established?

- ○ The VPN gateway initializes the connection
- ○ From the host machine
- ◉ From a client computer ✓

  Point-to-site connections get established from a client machine, such as a desktop computer connecting to the VPN.

- ○ In an Azure virtual network

2. How is a site-to-site VPN authenticated?

- [ ] No authentication is needed
- [ ] By using an administrator account
- [ ] Through a service principal in Azure Active Directory
- [x] By using an ASCII string secret

An ASCII string secret is used to authenticate between the VPN device and the VPN

3. When you connect to Azure by using a VPN, which connection method would you use?

- [x] A VPN gateway

A VPN gateway must be set up to bridge the connection between on-premises and

- [ ] An encrypted connection
- [ ] An Azure virtual machine
- [ ] A VPN tunnel

Next unit: Use Azure ExpressRoute for hybrid networks

Continue

Use Azure ExpressRoute for hybrid networks

10 minutes

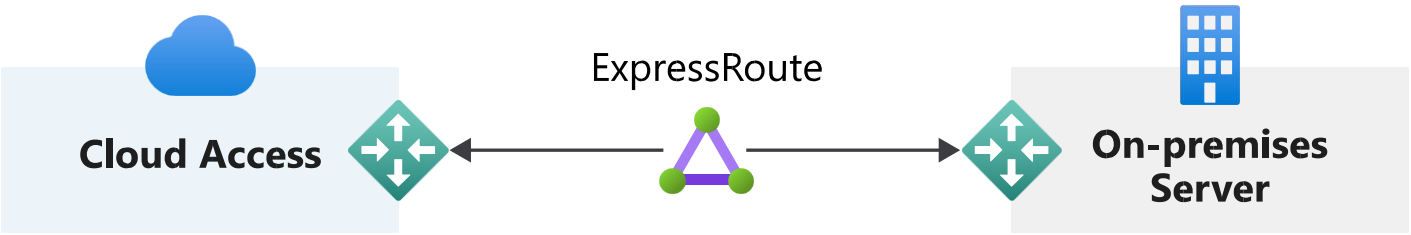
Your company is migrating some of its on-premises resources to Azure. As part of this migration, the central datacenter must remain on-premises with a connection to Azure. The architectural model also needs to consider Azure network connectivity for several satellite offices.

So far, you've identified a need for a resilient high-bandwidth connection from the on-premises network to Azure. In your initial investigations, you found that Azure ExpressRoute could suit your organization's hybrid network needs.

In this unit, you'll explore on-premises hybrid connectivity by using ExpressRoute, get an overview of the components that available in ExpressRoute, and walk through a reference architecture that supports this topology.

What is ExpressRoute?

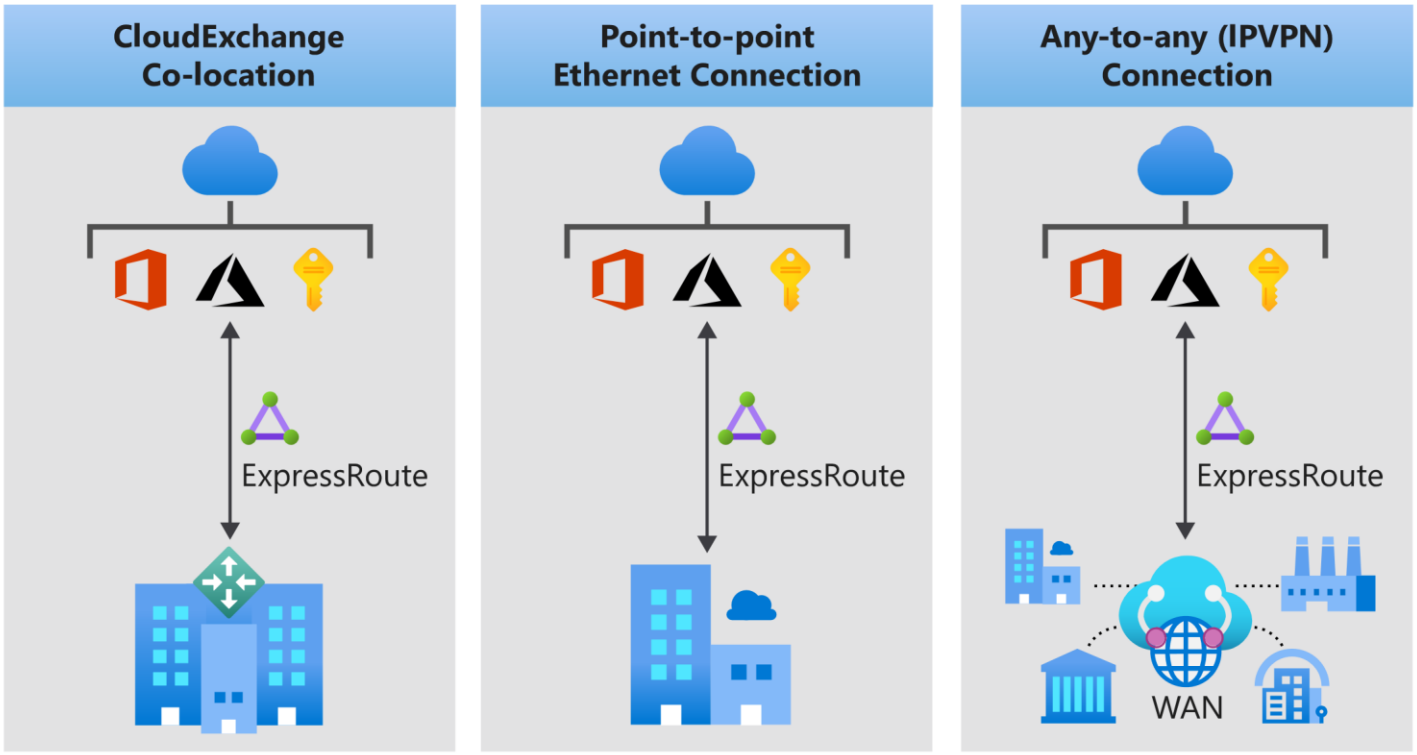
Azure ExpressRoute is an Azure service that allows you to extend on-premises networks over a private connection. A connectivity provider helps make this connection. ExpressRoute extends beyond Azure and lets you establish connections to other Microsoft cloud services, such as Office 365.



ExpressRoute connections don't use the public internet. By using a dedicated connection between your on-premises network and Azure, you achieve greater resilience, faster speeds, higher security, and lower latency.

ExpressRoute connectivity types

There are three ExpressRoute connectivity types, each serving a different need, as shown in the following diagram:



- **CloudExchange**: With the CloudExchange method, you cross-connect to Azure by using the Ethernet exchange that's provided by your colocation facility.
- **Any-to-any**: With the any-to-any network method, you integrate your WAN with Azure by using an IP virtual private network (VPN) provider. This connection type offers links between branch offices and datacenters. When it's enabled, the connection to Azure is similar to any other branch office that's connected via the WAN.
- **Point-to-point**: With the point-to-point Ethernet network method, you connect on-premises datacenters and offices to Azure through a point-to-point Ethernet link.

ExpressRoute circuits

With ExpressRoute, the logical connection between your on-premises network and your Azure network is called a circuit. You configure traffic management and routing in ExpressRoute by using circuits. You can have multiple circuits, which exist across various regions. ExpressRoute circuits also support connections through many connectivity providers.

Each circuit has multiple routing domains and peerings associated with it. Examples include Azure public peering, Azure private peering, and Microsoft peering. Each type has identical properties. Each circuit uses a pair of routers in either an active-active or load-sharing configuration, which creates a high availability environment. An ExpressRoute circuit doesn't map to anything physical.

Azure private peering

Private peering is a trusted extension of your core network in Azure with bidirectional connectivity. By using this peering model, you can connect to virtual machines and cloud services directly on their private IP addresses.

Microsoft peering

Microsoft peering provides connectivity to all Microsoft online services: Office 365, Dynamics 365, and Azure platform as a service (PaaS). This model requires a public IP address, owned by you or your connectivity provider, which adheres to a set of predefined rules.

Each circuit is assigned a globally unique identifier (GUID), or service key. This key is the only information exchanged between the three parties and is a one-to-one mapping for each circuit.

Circuit bandwidth

You can have as many circuits as you need, each matching the bandwidth you require. For example, you might want a higher bandwidth between your datacenter and the cloud, but a lower bandwidth for your satellite offices. Bandwidth speeds come in fixed tiers:

- 50 Mbps
- 100 Mbps
- 200 Mbps
- 500 Mbps
- 1 Gbps
- 10 Gbps
- 100 Gbps

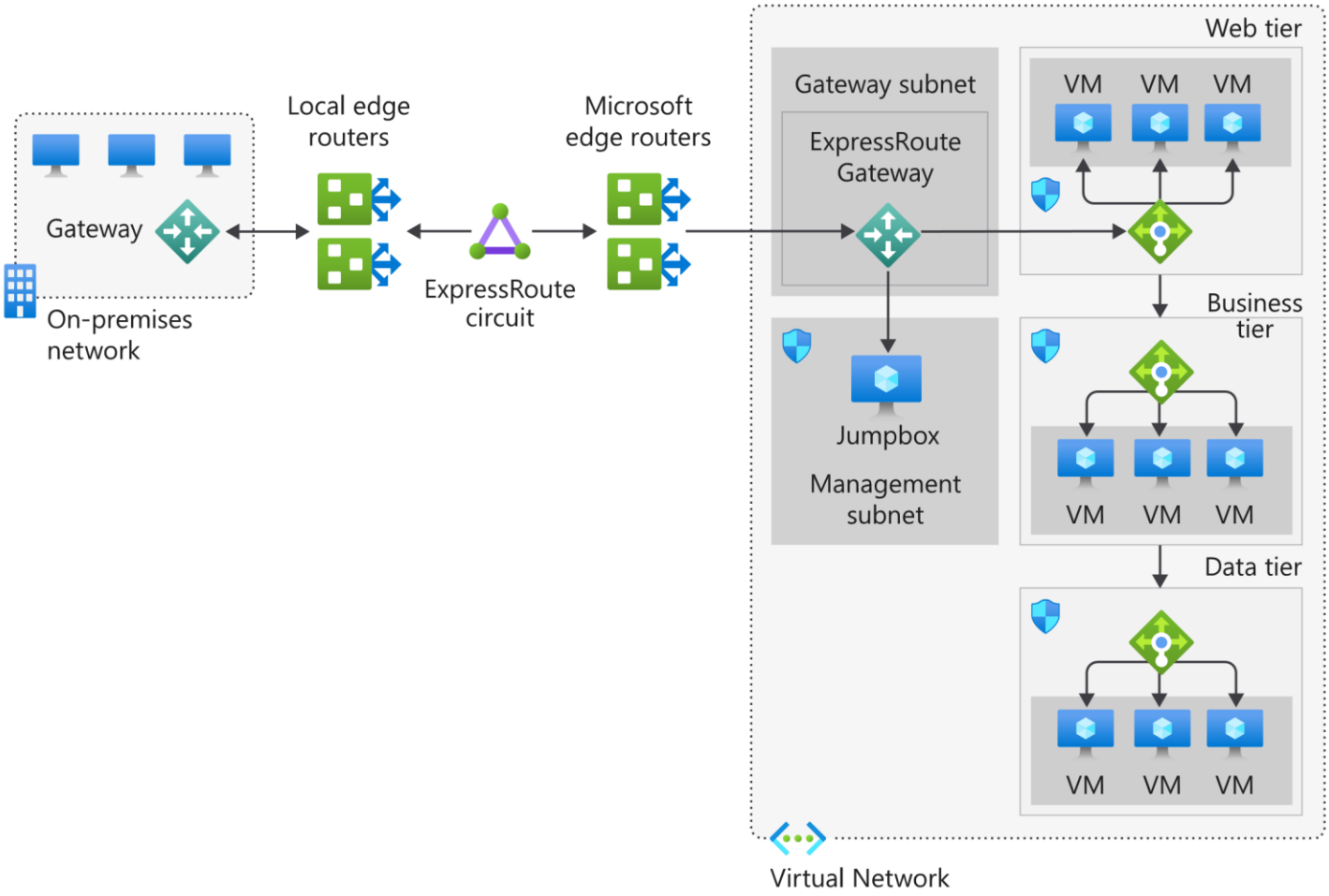
The bandwidth gets shared across any peering in the circuit and is mapped to the connectivity provider and peering location.

Coexisting connections and ExpressRoute

To use ExpressRoute, you must have a private connection, which is provided by a connectivity partner. However, ExpressRoute can exist alongside any of your current site-to-site, point-to-site, or VPN-to-VPN connections.

ExpressRoute reference architecture

The reference architecture that's illustrated in the following diagram shows how to connect your on-premises network to your Azure virtual networks.



The architecture model includes several components:

- The **on-premises network** is your local Active Directory-managed network.
- **Local edge routers** connect your on-premises network to the connectivity provider's circuit.
- An **ExpressRoute circuit**, supplied by your connectivity provider, operates as a layer 3 circuit. It provides the link between the Azure edge routers and your on-premises edge router.

- The **Microsoft edge routers** are the cloud-side connection between your on-premises network and the cloud. There are always two edge routers, which provides a highly available active-active connection.
- The **Azure virtual network** is where you'll segment your network and assets into tiers. Each application tier, or subnet, can manage specific business operations (for example, web, business, and data).

Is ExpressRoute right for you?

When you're evaluating whether to switch to ExpressRoute, consider the points in the following sections.

Benefits

Implementing ExpressRoute in your organization helps produce the following benefits:

- ExpressRoute is better suited to high-speed and critical business operations.
- ExpressRoute circuits support a maximum bandwidth of 100 Gbps.
- ExpressRoute provides dynamic scalability to help meet your organizational needs.
- ExpressRoute uses layer 3 connectivity and security standards.

Considerations

The following list identifies a few key considerations:

- The setup and configuration for ExpressRoute is more complex, and will require collaboration with the connectivity provider.
- ExpressRoute requires the on-premises installation of high-bandwidth routers.
- The ExpressRoute circuit is handled and managed by the connectivity provider.
- ExpressRoute doesn't support the Hot Standby Router Protocol (HSRP). You'll need to enable a Border Gateway Protocol (BGP) configuration.
- ExpressRoute operates on a layer 3 circuit and requires a network security appliance to manage threats.
- Monitoring the connectivity between your on-premises network and Azure must use the Azure Connectivity Toolkit.
- To improve network security, ExpressRoute requires network security appliances between the provider's edge routers and your on-premises network.

Check your knowledge

1. Which of the following is a valid connectivity model for Azure ExpressRoute?

- A cross-network connection
- A point-to-point Ethernet network

  A point-to-point Ethernet connection is used to connect on-premises datacenters and offices to Azure through a point-to-point Ethernet link.
- Azure network-to-network connectivity
- A point-to-site network

2. What is the maximum available bandwidth for an ExpressRoute circuit?

- 50 Mbps
- 1 Gbps
- 200 Gbps
- 100 Gbps

  A 100 Gbps bandwidth circuit is currently the maximum available with ExpressRoute.

3. What would you use a VPN gateway for?

- To connect to a CloudExchange provider
- For ExpressRoute to connect to on-premises networks

As the cloud connector for the ExpressRoute circuit

As a coexisting connection for failover

A VPN gateway can be configured as a failover route if there's a loss of connectivity to an ExpressRoute circuit.

Next unit: Work with hybrid networks on Azure

Continue

Previous Unit 4 of 6 Next

200 XP

Work with hybrid networks on Azure

10 minutes

Your organization is eager to continue with migration to the cloud. You've explored the merits of using Azure ExpressRoute to provide a dedicated, high-speed connection between your on-premises network and Azure.

Due diligence requires you to explore the other available hybrid architecture options for connecting your on-premises network to Azure.

In this unit, you will:

- Gain an understanding of virtual private network connections.
- Look at a resiliency option for ExpressRoute.
- Consider the merits of hub-spoke network topology.

What is a hybrid network architecture?

Hybrid network is a term that's used when two different network topologies combine to form a single cohesive network. With Azure, a hybrid network represents the merging or combining of an on-premises network with an Azure virtual network. It allows the continued use of your existing infrastructure while giving you all the benefits of cloud-based computing and access.

There are several reasons why you might want to adopt a hybrid network solution. The two most common are:

- To migrate from a pure on-premises network to a pure cloud-based network.
- To extend your on-premises network and resources to support the cloud services.

Whatever your motivations for adding cloud services to your infrastructure, there are several architectures to consider. We covered ExpressRoute in the preceding unit. The other architectures are:

- Azure VPN Gateway
- ExpressRoute with VPN failover
- Hub-spoke network topology

Azure VPN Gateway

Azure VPN Gateway, a virtual network gateway service, allows site-to-site and point-to-site VPN connectivity between your on-premises network and Azure.

A VPN or virtual private network is a well-established, well-understood network architecture.

VPN Gateway uses your existing connection to the internet. However, all communication is encrypted using the Internet Key Exchange (IKE) and Internet Protocol Security (IPsec) protocols. You can have only one virtual network gateway per virtual network.

When you set up a virtual network gateway, you must specify whether it's a VPN gateway or an ExpressRoute gateway.

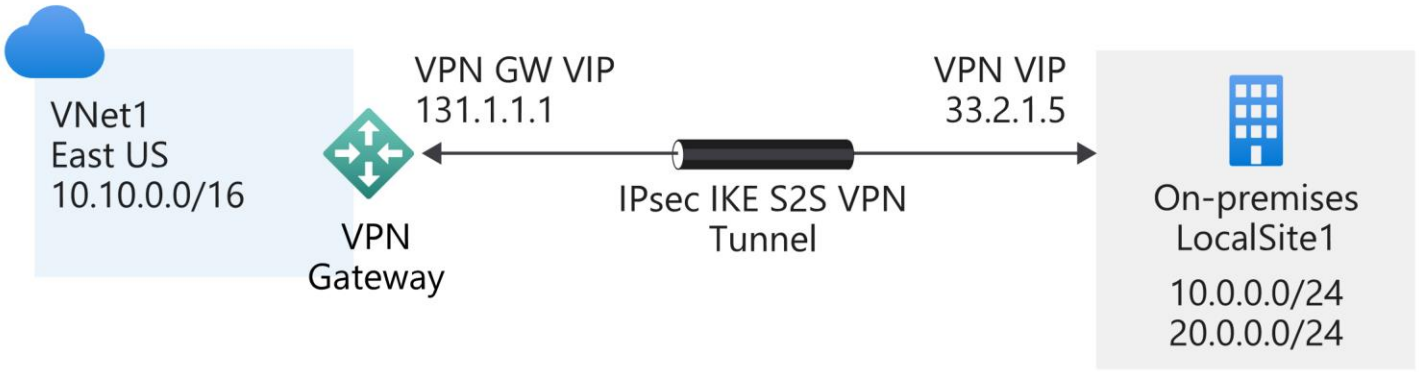
The VPN type depends on the type of connection topology you need. For example, if you want to create a point-to-site (P2S) or a point-to-point (P2P) gateway, you use a *RouteBased* type. There are two VPN types:

- **PolicyBased**: Uses an IPsec tunnel to encrypt data packets. Configuration of the policy uses address prefixes that are drawn from your Azure virtual network and your on-premises network.
- **RouteBased**: Uses the routing or IP forwarding tables to route data packets to the correct tunnel. Each tunnel encrypts and decrypts all packets.

After you've specified the *VPN type* for the virtual network gateway, it can't be altered. If you have to make a change, you need to delete the virtual network gateway and create it again.

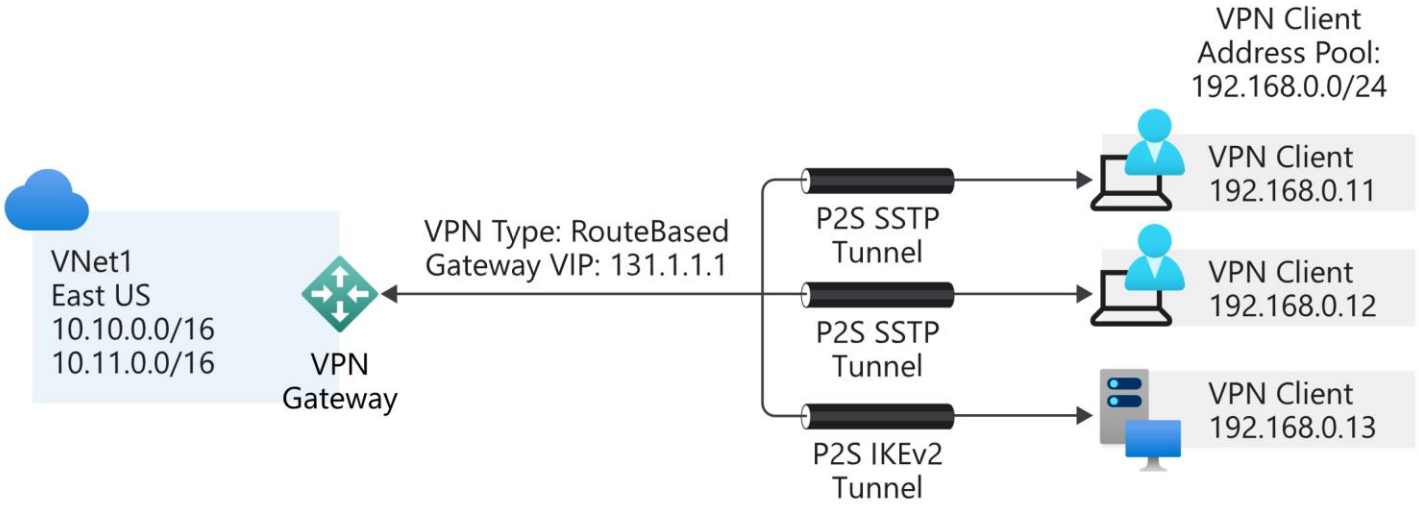
Site-to-site

All site-to-site gateway connections use an IPsec/IKE VPN tunnel to create a connection between Azure and your on-premises network. For a site-to-site connection to work, you'll need an on-premises VPN device with a publicly accessible IP address.



Point-to-site

A point-to-site gateway connection creates a secured connection between an individual device and your Azure virtual network. This gateway type is suited to remote workers; for example, users attending a conference or working from home. A point-to-site connection doesn't require a dedicated on-premises VPN device.



Benefits

Here are some of the benefits of using a VPN connection:

- It's a well-known technology, easy to configure and maintain.
- All data traffic is encrypted.
- It's better suited to lighter data-traffic loads.

Considerations

When you're evaluating the use of this hybrid architecture, consider the following points:

- A VPN connection uses the internet.
- Potential latency issues might exist, depending on bandwidth size and usage.
- Azure supports a maximum bandwidth of 1.25 Gbps.
- For site-to-site connections, you need a local VPN device.

ExpressRoute with VPN failover

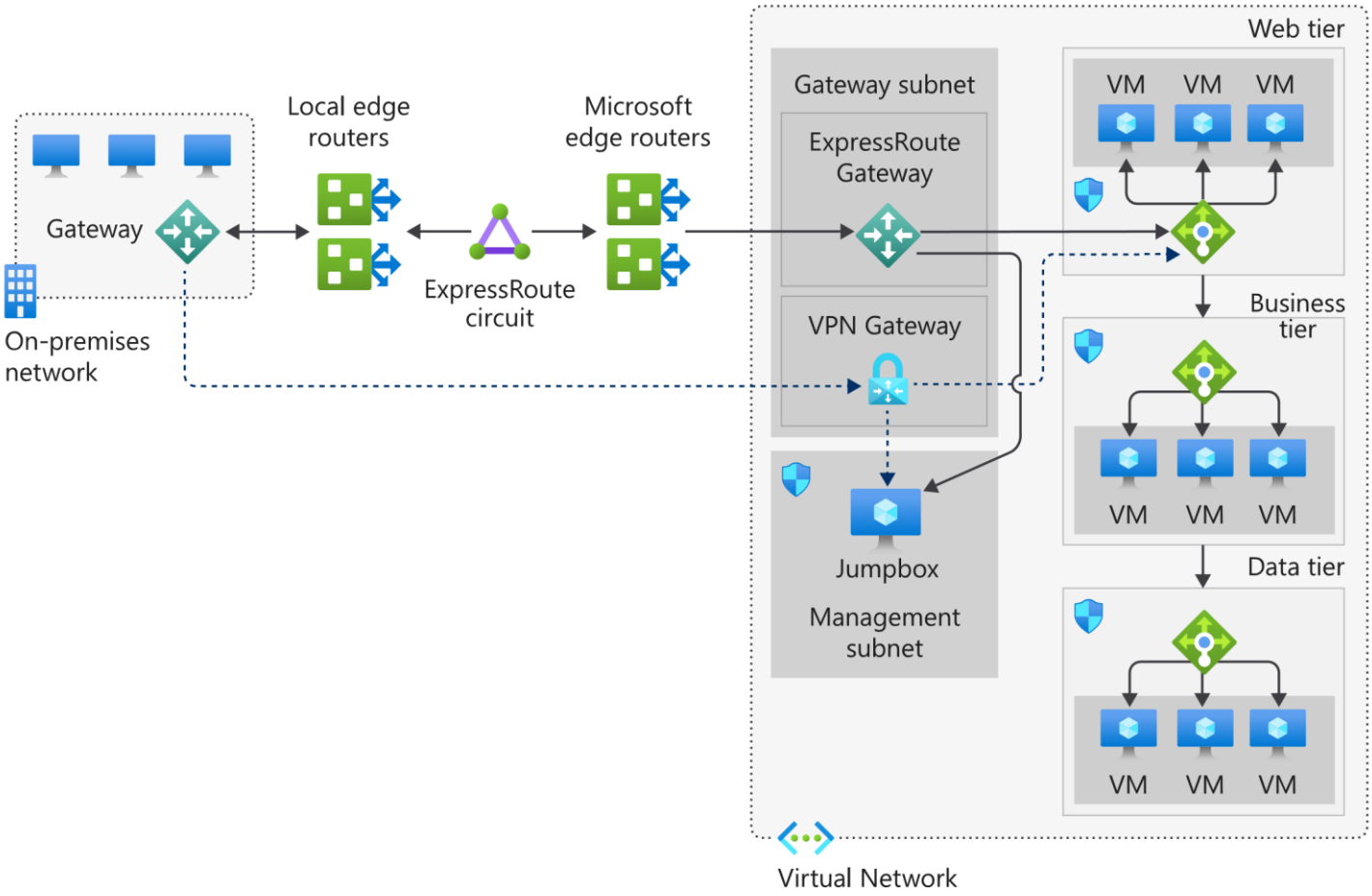
One of the guarantees of using ExpressRoute is that it provides a high level of availability. Each ExpressRoute circuit comes with dual ExpressRoute gateways. However, even with this level of resiliency built into the Azure side of the network, connectivity might be interrupted.

One way to remedy this situation, and maintain connectivity, is to provide a VPN failover service.

The merging of the VPN connection and ExpressRoute improves the resiliency of your network connection. When it operates under normal conditions, ExpressRoute behaves precisely like a regular ExpressRoute architecture, with the VPN connection remaining dormant. If the ExpressRoute circuit fails or goes offline, the VPN connection takes over. This action ensures network availability under all circumstances. When the ExpressRoute circuit is restored, all traffic reverts to using the ExpressRoute connection.

Reference architecture for ExpressRoute with VPN failover

The following diagram illustrates how to connect your on-premises network to Azure by using ExpressRoute with a VPN failover. The chosen topology in this solution is a VPN-based, site-to-site connection with high traffic flow.



In this model, all network traffic routes through the ExpressRoute private connection. When connectivity is lost on the ExpressRoute circuit, the gateway subnet automatically fails over to the site-to-site VPN gateway circuit. This scenario is indicated by the dotted line from the gateway to the VPN gateway in the Azure virtual network.

When the ExpressRoute circuit is restored, traffic automatically switches back from the VPN gateway.

Benefits

The following benefit is available when you implement ExpressRoute with a VPN failover:

- It creates a resilient, high availability network.

Considerations

When you implement an ExpressRoute with VPN failover architecture, consider the following points:

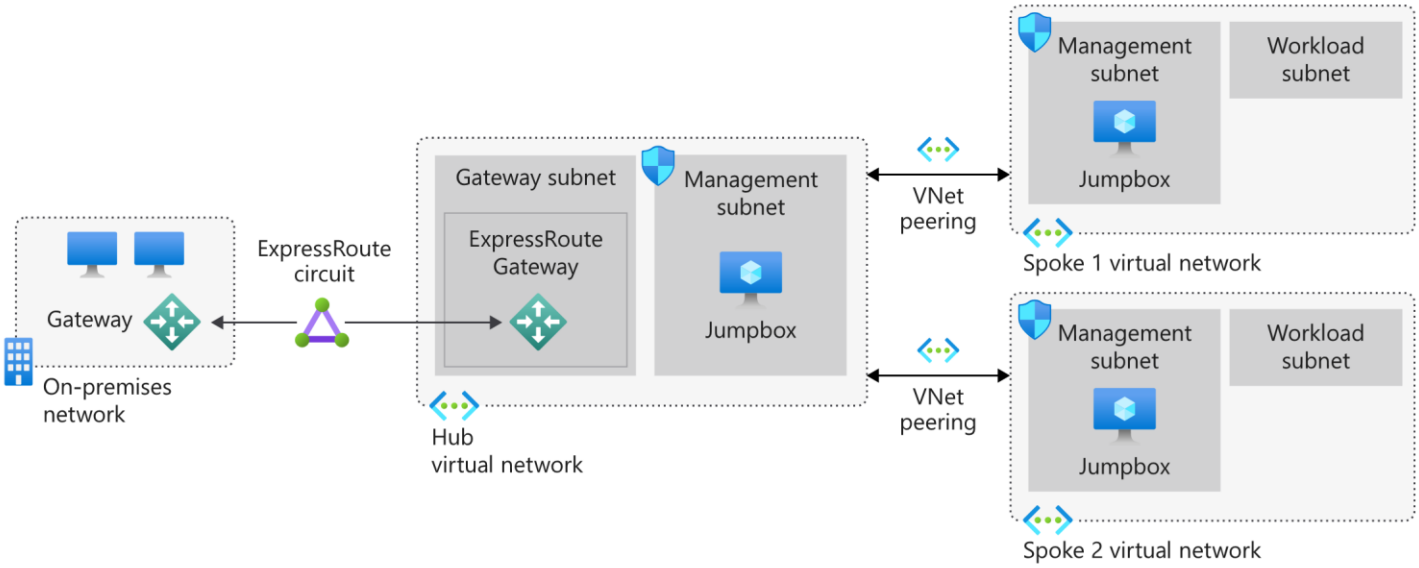
- When a failover occurs, bandwidth is reduced to VPN connection speeds.
- The ExpressRoute and VPN gateway resources must be in the same virtual network.

- There's a highly complex configuration.
- Implementation requires both an ExpressRoute connection and a VPN connection.
- Implementation requires a redundant VPN gateway and local VPN hardware.

> **Note**
>
> A redundant VPN gateway incurs payment charges, even when it's not being used.

Hub-spoke network topology

The hub-spoke network topology allows you to structure the workloads that are carried out by your servers. It uses a single virtual network as the hub, which is also connected to your on-premises network through either VPN or ExpressRoute. The spokes are other virtual networks that are peered with the hub. Each spoke can be assigned specific workloads, and the hub is used for shared services.



You can implement the hub and each spoke in separate subscriptions or resource groups and then peer them together.

This model uses one of the three previously discussed approaches: VPN, ExpressRoute, and ExpressRoute with VPN failover. The associated benefits and challenges are discussed in the following sections.

Benefits

Implementing a hub-spoke architecture has the following benefits:

- The use of sharing and centralized services on the hub might reduce the need for duplication on the spokes, which can reduce costs.
- Subscription limits are overcome by peering virtual networks.
- The hub-spoke model allows for the separation of organizational work areas into dedicated spokes, such as SecOps, InfraOps, and DevOps.

Considerations

When you're evaluating the use of this hybrid architecture, consider the following point:

- Look at the services that are shared on the hub and what remains on the spokes.

Check your knowledge

1. Why would you implement a VPN gateway in your Azure virtual network?

   ○ To improve the performance of the ExpressRoute circuit

- To allow the ExpressRoute circuit to connect to your Azure virtual network
- To allow the ExpressRoute circuit to connect to your on-premises network
- To provide a redundant failover connection

The VPN gateway provides a direct connection between your on-premises gateway and the VPN gateway can be used in the event of an ExpressRoute circuit

2. How are VPN gateway connections to an on-premises network routed?

- Through a private VPN connection
- Over the public internet

A VPN gateway sends encrypted traffic between an on-premises network and the Azure net internet

- Through the Azure datacenter
- By provisioning a traffic manager

3. What is the underlying reason for implementing a hub-spoke architecture?

- Greater cost visibility
- Greater security
- Greater business visibility

The hub becomes the core of the business and provides the foundations for much deeper bu

- Faster network communications

Next unit: Choose a hybrid network architecture on Azure

Continue

Previous Unit 5 of 6 Next

200 XP

Choose a hybrid network architecture on Azure

10 minutes

Your organization is almost ready to migrate resources to Azure. This migration uses a hub-spoke architecture, but there remains an open question about the best method to use to connect the on-premises datacenter with the Azure network.

You've already provided a high-level design for this migration. However, your project manager has requested extra information about the choice between Azure ExpressRoute and Azure VPN Gateway. You also need to put together a capability matrix, so that non-technical staff members understand the features of each service.

In this unit, you'll assess two hybrid networking scenarios: Azure ExpressRoute and Azure VPN Gateway.

Azure ExpressRoute

As the solution architect, you understand the need to provide a robust and reliable network connection between your on-premises datacenters and Azure. The requirements demand low latency and a high level of availability. From your investigations, you know there are two possibilities:
VPN or ExpressRoute.

Both solutions provide a secure connection between the on-premises network and the Azure virtual network.

The VPN solution uses known and established technologies. However, a VPN isn't designed to handle high data volumes, and the core infrastructure backbone still uses the internet. Using a VPN might be a cost-effective solution in the short term. You know that in the long term a VPN is unlikely to provide the performance, scalability, and resilience your organization requires.

The ExpressRoute option uses a dedicated private connectivity partner to provide a direct connection between your on-premises datacenters and Azure. Microsoft requires the connectivity provider to maintain two discrete connections to the Azure network for each ExpressRoute circuit, which helps ensure a high level of resilience. ExpressRoute also provides a connection bandwidth that's almost 10 times faster than a VPN.

ExpressRoute requires a close working partnership with the connectivity provider, it's a more complex configuration, and it can be more expensive than a regular VPN network.

However, based on what you've learned so far, you decide that ExpressRoute is better suited to meet your organization's primary needs.

Azure VPN Gateway

Your organization's presence covers a wide geographical location. Although your datacenter and HQ are all in the same building, many satellite branches also need a connection to the Azure network.

You've established ExpressRoute as your preferred connection to Azure. Now you'll consider your second requirement: connecting regional satellite offices to the Azure network.

Unlike your HQ, which has thousands of employees and an on-premises datacenter, each satellite office has between 20 and 50 employees. There's no requirement for a fast, low-latency connection.

This secondary requirement has different needs. ExpressRoute, although ideal for high speeds and resilience, doesn't suit smaller satellite offices that have a lower connectivity requirement. Also, each office would need a dedicated private connection to be maintained and run by the connectivity provider.

In this instance, using a VPN would provide a better overall solution. A VPN is based on existing and understood technology, which means it can be managed in-house. The lower-speed bandwidth is within an acceptable tolerance for day-to-day usage.

Hybrid networking capability matrix

To show what you've learned in this unit and to help you choose between the two connectivity types, we've built a capability matrix. The following table lists the key features for each method, the supported bandwidths, the resiliency model, typical use cases, and associated SLAs.

https://docs.microsoft.com/en-us/learn/modules/design-a-hybrid-network-architecture/5-choosing-hybrid-architecture 1/2 2/21/2020 Choose a hybrid network architecture on Azure - Learn | Microsoft Docs

Capability	VPN Gateway	ExpressRoute
Azure services support	Azure Cloud Services and Azure Virtual Machines	Microsoft Cloud Platform
Bandwidth	< 1.25 Gbps	< 10 Gbps or 100 Gbps (direct)
Protocol	SSTP or IPsec	Direct over VLAN or MPLS
Routing	Static or dynamic	Border Gateway Protocol (BGP)
Connection resiliency	Active-passive or active-active	Active-passive or active-active
Use case	Prototyping, dev, test, labs, RDC, and small production workloads	Access to all Azure services, enterprise-grade, supporting critical largescale workloads
SLA	99.95-99.99%	99.95%

Check your knowledge

1. What is a valid use case for Azure ExpressRoute?

- [x] Mission-critical workloads

ExpressRoute is the recommended service for mission-critical, large-scale workloads.

- [ ] Prototyping
- [ ] Dev/test workloads
- [ ] Small production workloads

2. Why would an organization choose VPN Gateway over Azure ExpressRoute?

- [ ] Bandwidth limitations
- [x] Cost and traffic volume

VPN Gateway is less expensive than ExpressRoute and is better suited to smaller traffic volumes.

- [ ] Redundancy support
- [ ] Lack of ExpressRoute partners

3. What is the preferred connection resiliency for ExpressRoute?

- [ ] passive-passive
- [ ] passive-active
- [ ] active-passive
- [x] active-active

Express route uses an active-active connection resiliency.

Next unit: Summary

Continue

Previous Unit 6 of 6

100 XP

Summary

2 minutes

In this module, you explored some of the available technologies for connecting your on-premises network with Azure. You now understand an Azure virtual network operates. You learned about the capabilities of Azure VPN Gateway and its associated topologies, such as site-to-si and point-to-site connections. Additionally, you know the benefits and features of several hybrid network services and architectures, includin Azure ExpressRoute, VPN, ExpressRoute with VPN, and the hub-spoke topology.

You can apply this knowledge to your organization's migration. By using a Microsoft connectivity provider to set up an ExpressRoute connec you can help give your organization a fast, secure connection to the cloud. The ExpressRoute circuit can incorporate a VPN failover for additi resilience. Should the ExpressRoute circuit become temporarily unavailable, the VPN link maintains the connection.

By using a hybrid network architecture, such as ExpressRoute, you establish a high-performance user experience that provides a feature-rich, cost-effective, and private networking connection.

Learn more

To learn more about networking services on Azure, see the following articles:

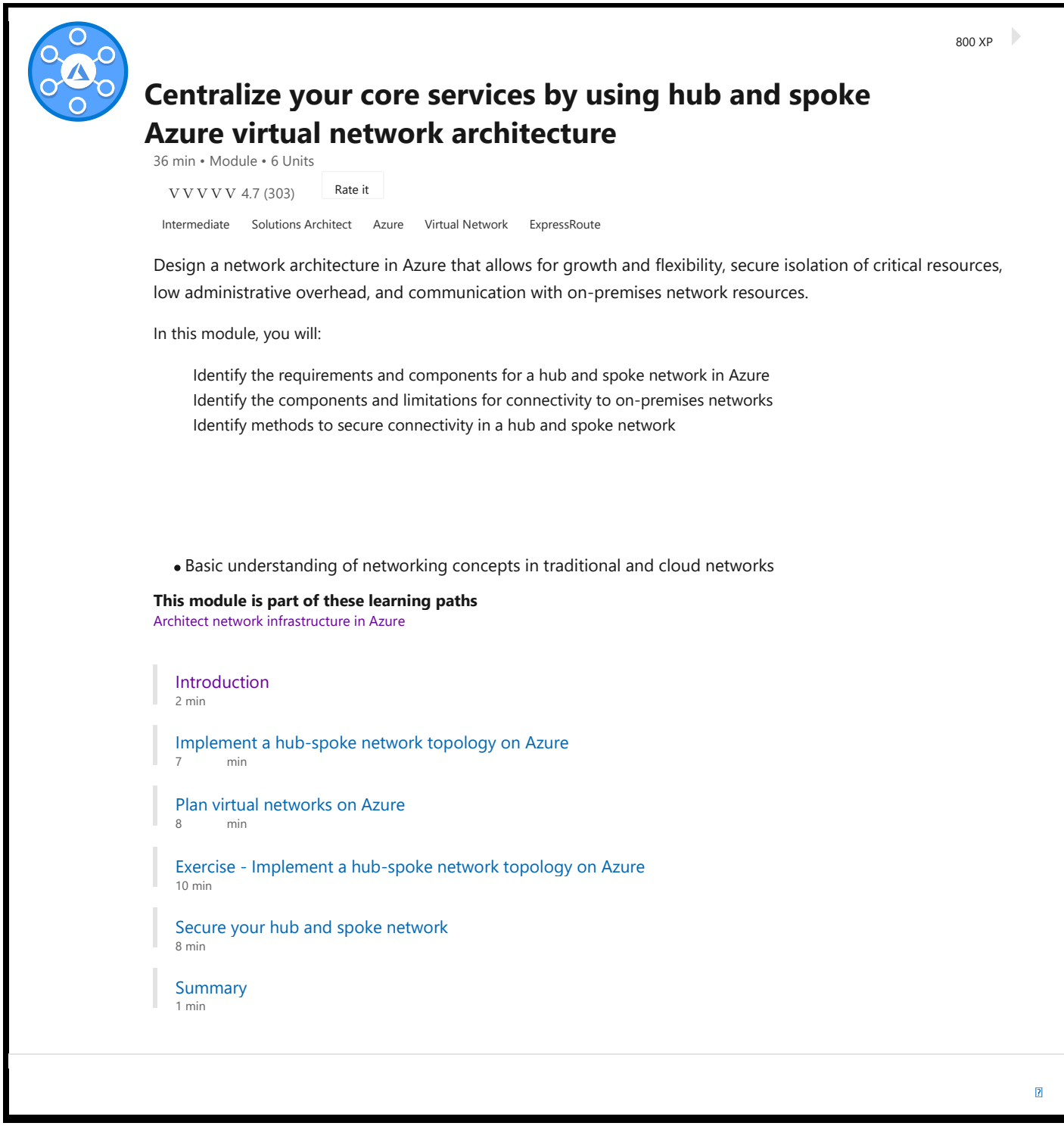
- Choose a solution for connecting an on-premises network to Azure
- What is VPN Gateway?
- What is ExpressRoute?
- Connect an on-premises network to Azure by using ExpressRoute with VPN failover

Module complete:

Unlock achievement

https://docs.microsoft.com/en-us/learn/modules/design-a-hybrid-network-architecture/6-summary 1/2

2/21/2020 Centralize your core services by using hub and spoke Azure virtual network architecture - Learn | Microsoft Docs



800 XP

Centralize your core services by using hub and spoke Azure virtual network architecture

36 min • Module • 6 Units

V V V V V 4.7 (303) Rate it

Intermediate Solutions Architect Azure Virtual Network ExpressRoute

Design a network architecture in Azure that allows for growth and flexibility, secure isolation of critical resources, low administrative overhead, and communication with on-premises network resources.

In this module, you will:

Identify the requirements and components for a hub and spoke network in Azure
Identify the components and limitations for connectivity to on-premises networks
Identify methods to secure connectivity in a hub and spoke network

- Basic understanding of networking concepts in traditional and cloud networks

This module is part of these learning paths
Architect network infrastructure in Azure

Introduction
2 min

Implement a hub-spoke network topology on Azure
7 min

Plan virtual networks on Azure
8 min

Exercise - Implement a hub-spoke network topology on Azure
10 min

Secure your hub and spoke network
8 min

Summary
1 min

https://docs.microsoft.com/en-us/learn/modules/hub-and-spoke-network-architecture/index 1/1

2/21/2020 Introduction - Learn | Microsoft Docs

Unit 1 of 6 S Next T

100 XP

Introduction

2 minutes

You work for a large insurance organization that's planning a migration to Azure. Your head office currently hosts all the company's public facing websites and its insurance quote app. Your company headquarters and multiple satellite offices around the country will still need to access internal resources that were migrated to Azure, and the back-office area of the quote app. Your virtual machine infrastructure has central dependencies, such as Active Directory and DNS, that should be accessible.

In this module, you'll look at a hub and spoke topology in Azure that allows for growth and flexibility, planning and designing virtual networks, and helping to secure your architectural design. The final design will provide for isolation of resources, have a low administrative overhead, and allow communication from on-premises resources.

Learning objectives

In this module, you'll:

- Identify the requirements and components for a hub and spoke network in Azure.
- Identify the components and limitations for connectivity to on-premises networks.
- Identify methods to secure connectivity in a hub and spoke network.

Prerequisites

- Basic understanding of networking concepts in traditional and cloud networks.

Next unit: Implement a hub-spoke network topology on Azure

Continue

https://docs.microsoft.com/en-us/learn/modules/hub-and-spoke-network-architecture/1-introduction 1/2

Previous Unit 2 of 6 Next

100 XP

Implement a hub-spoke network topology on Azure

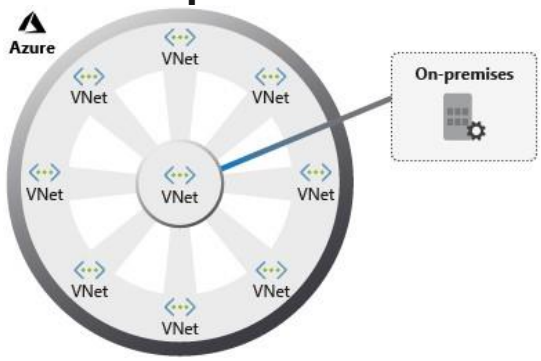
7 minutes

By using virtual networks, network security groups, virtual network peering, and Azure ExpressRoute, you can create different network topologies hosted in the cloud.

Your company is planning a migration of your on-premises resources to Azure. Your central datacenter is currently located at your headquarters, and will eventually migrate completely to Azure. Initially, you've been asked to move some of your satellite offices into the cloud, while also maintaining the connectivity to your headquarters. The ultimate goal for the migration is to host all your computing resources on Azure.

In this unit, you'll explore the hub and spoke architecture, the topology, the components needed in Azure, and how to plan implementing infrastructure by using this model.

Hub and spoke architecture foundations



A hub and spoke consists of a centralized architecture (a hub) connecting to multiple points (spokes). When drawn, it looks similar to a wheel, with a hub at the center and spokes connected to it. This model in Azure organizes your network infrastructure into multiple connected virtual networks. This architecture provides an efficient way to manage common communication, security requirements, and potential subscription limitations.

Implementing a hub and spoke architecture can have the following benefits:

- A centrally managed connection to your on-premises environment.
- Integration of separate working environments into a central location for shared services.
- Traffic routing through the central hub, so workloads can be managed centrally.

Introduction to the hub-spoke topology

Hub-spoke networks are commonly used for hybrid cloud architectures, and can be simpler to implement and maintain in the long term. The hub is a virtual network that acts as a central location for managing external connectivity, and hosting services used by multiple workloads. The hub coordinates all communications to and from the spokes. IT rules or processes like security can inspect, route, and centrally manage traffic.
The spokes are virtual networks that host workloads, and connect to the central hub through virtual network peering.

Hub and spoke topologies offer several business benefits:

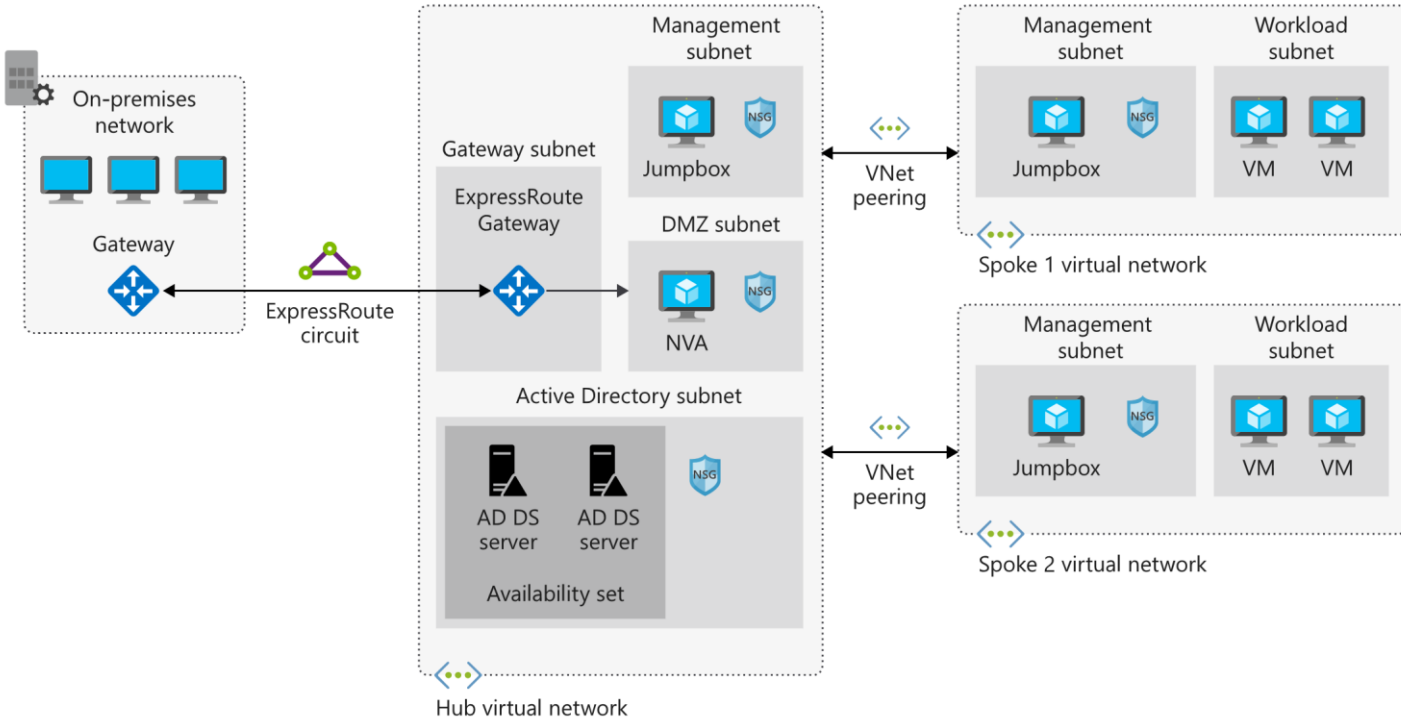
- Increased business agility by standardizing on network connections. Organizations can adapt to changing markets, adding a new branch in a different geopolitical region, or a new business channel, as spokes.
- Liability reduction by maintaining a consistent architecture. As the business grows, or traffic volumes increase, it's simple to add more systems.
- Greater visibility into the business, with data flowing through the same place. The hub is the core of the business and provides the foundations for deeper business insights, as it processes every piece of information belonging to the organization.
-

A single location in which to share centralized services by multiple workloads. This enables you to minimize redundant resources and the effort required to manage them.

https://docs.microsoft.com/en-us/learn/modules/hub-and-spoke-network-architecture/2-implement-hub-spoke 1/2 2/21/2020 Implement a hub-spoke network topology on Azure - Learn | Microsoft Docs

Architectural components

Let's take a look at a reference architecture for a hub-spoke topology. The following image shows the proposed architecture of a pattern to extend your on-premises environment to Azure.



The hub is a virtual network in Azure that's the center point for your business connectivity. Shared services are hosted in their own subnets for sharing with the spokes, and a perimeter subnet acts as a security appliance.

The spokes are also virtual networks in Azure, used to isolate individual workloads. The traffic flow between the on-premises headquarters and Azure is connected through ExpressRoute, connected to the hub virtual network. The virtual networks from the spokes to the hub are peered, and enable communication to on-premises resources. You can implement the hub and each spoke in separate subscriptions or resource groups.

The components included in this architecture are:

- **Azure Virtual Network**: Azure virtual networks are a representation of your own IT network. They're contained within the cloud, logically isolating dedicated organizational resources in your subscriptions.
- **Azure VPN Gateway**: VPN Gateway is the bridge between your on-premises network and Azure. VPN gateways are a special type of Virtua Network gateway that sends encrypted traffic between the two networks over the internet.
- **Azure ExpressRoute**: ExpressRoute is a service in Azure that allows you to extend on-premises networks over a private connection.

Next unit: Plan virtual networks on Azure

Continue >

https://docs.microsoft.com/en-us/learn/modules/hub-and-spoke-network-architecture/2-implement-hub-spoke 2/2

Previous Unit 3 of 6 Next

200 XP

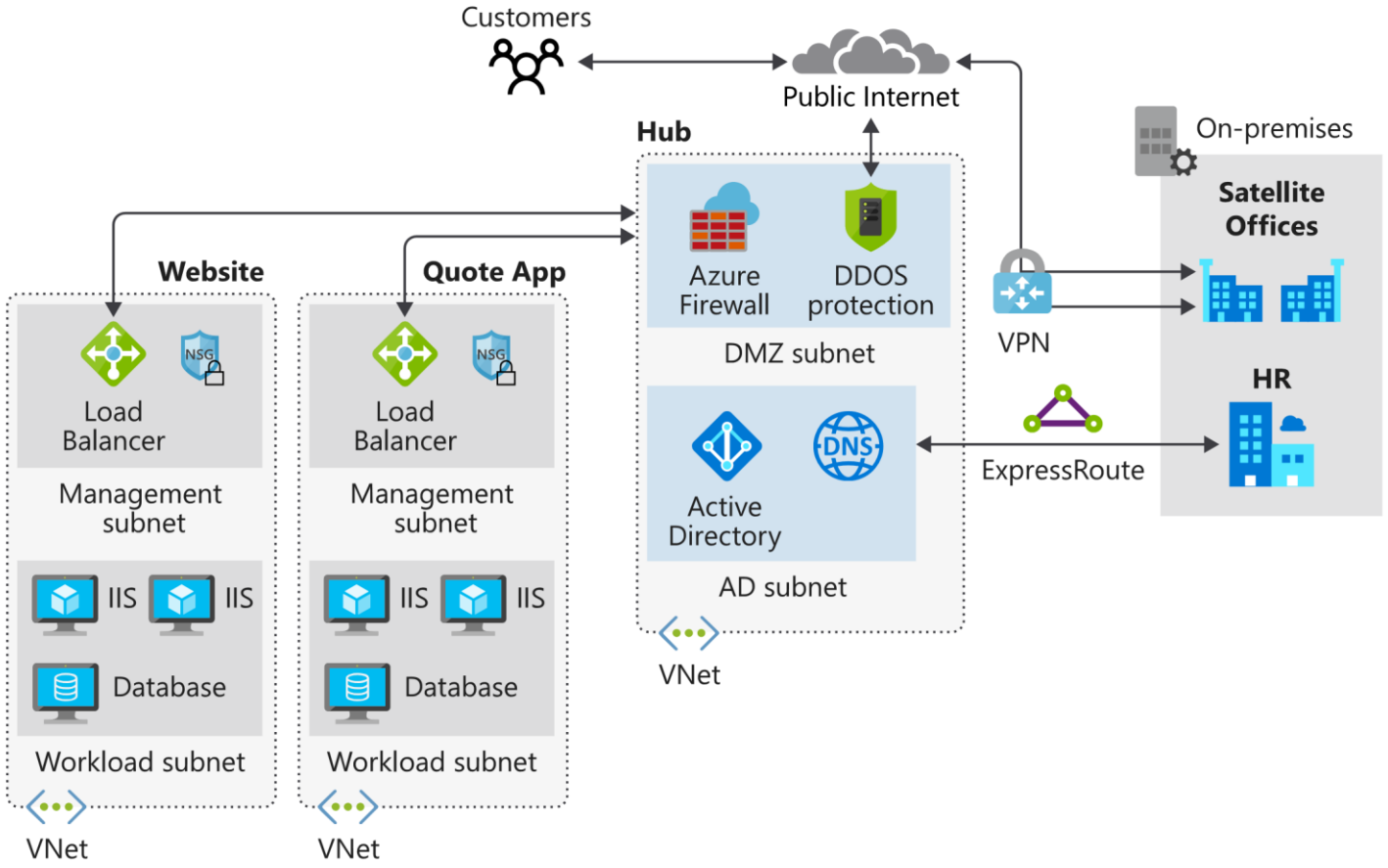
Plan virtual networks on Azure

8 minutes

You've chosen to support your company's move to Azure by implementing a hub and spoke architecture.

As a lead architect on the project, you're managing the production of a virtual networking design by using Azure ExpressRoute for your headquarters' connectivity. You also must decide how to connect your company's satellite offices to the new hub and spoke network.

In this unit, you'll explore virtual networking in the Azure platform, design considerations, and how to implement ExpressRoute for connectivity to on-premises networks.



Introduction to Azure virtual networking

Virtual networks provide networking services in Azure and enable you to extend your existing on-premises infrastructure. An Azure virtual network can represent your private IT infrastructure within the cloud, logically isolating dedicated resources in your subscriptions. Virtual networks enable:

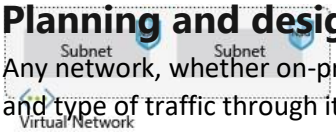
- External connections to the internet.
- Communication between different internal Azure resources.
- Isolation of those resources.
- Connections to on-premises computers.
- Network traffic management.

Two important elements of virtual networks are subnets and network security groups.

Subnets: Each virtual network can include a number of subnets. Each subnet has its own unique properties.

Network security groups: These allow you to filter the inbound and outbound traffic through your virtual network or subnet. You can also use network security groups to filter traffic by source and destination IP address, port, or protocol.

Planning and design considerations for virtual networks



Any network, whether on-premises or in the cloud, requires a method for managing the flow, direction, and type of traffic through it. There are several considerations for virtual networks:

Segmentation: It's important to consider potential isolation of traffic into different subnets or virtual networks, or into separate subscriptions.

Security: Use network security groups and network virtual appliances to filter network traffic to and from resources in a virtual network. **Connectivity**: You can connect a virtual network to other virtual networks by using virtual network peering, or to your on-premises networks by using ExpressRoute or a VPN gateway.

Routing: Azure virtual networks automatically create routing tables within each subnet, and add default system routes to the tables. Custom routes allow you to override these default system routes. With custom routes, you can direct traffic through network virtual appliances to provide enhanced security and filtering capabilities.

Connect your on-premises network

When working towards integrating your on-premises network with Azure, you need to bridge between the two networks. Azure VPN Gateway provides this functionality. A VPN gateway sends encrypted traffic between the two networks over the internet. Gateways support multiple connections that route the VPN tunnels through the available bandwidth, although a virtual network can only have one gateway assigned. You can also use a VPN gateway for network-to-network connections in Azure.

Azure ExpressRoute is another option to consider for bridging. ExpressRoute allows you to extend your on-premises networks over a private connection to Azure. This connection is facilitated by a connectivity or cloud exchange provider. ExpressRoute extends wider than just Azure resources, and allows you to establish connections to other Microsoft cloud services like Office 365.

Implementation of ExpressRoute does take some time. You have to work through a connectivity provider, and might require a physical network device implementation. To provide connectivity while this implementation is ongoing, you can use site-to-site VPN to add a connection between your on-premises resources and your Azure virtual networks. You then migrate to your new ExpressRoute connection when the service provider confirms that the setup is complete.

Use ExpressRoute in a hub-spoke topology

Using ExpressRoute in a hub and spoke topology is no different than other architectural patterns. ExpressRoute, which underpins the connectivity between the hub and the on-premises network, works best when there's high data ingress and egress.

You use circuits to manage and route traffic. Link ExpressRoute into a virtual network in Azure. The circuits to be connected to the virtual network might be in different regions or subscriptions. There are limits to the number of virtual networks per ExpressRoute circuit. For the standard tier, the limit is currently 10 networks. If you use the premium add-on, the limit is increased based on the circuit size.

The lowest number is 20 virtual networks on a 50-Mbps circuit, up to 100 for circuits that are 10-Gbps or bigger.

Check your knowledge

1. Which one of these is a way resources can communicate with each other inside Azure?

- Have the resources communicate over an ExpressRoute connection.
- Having the resources inside a single virtual network.

  Correct, all resources within a virtual network can communicate in the private address space.
- Having each resource specify a network endpoint.

2 What's the limit for the number of virtual networks you can connect to ExpressRoute?

- 10
- 100
- It depends on the size of the circuit.

  The limit is based on the SKU and size of the ExpressRoute circuit. At present, 100 is the maxi

Next unit: Exercise - Implement a hub-spoke network topology on Azure

Continue

100 XP

Exercise - Implement a hub-spoke network topology on Azure

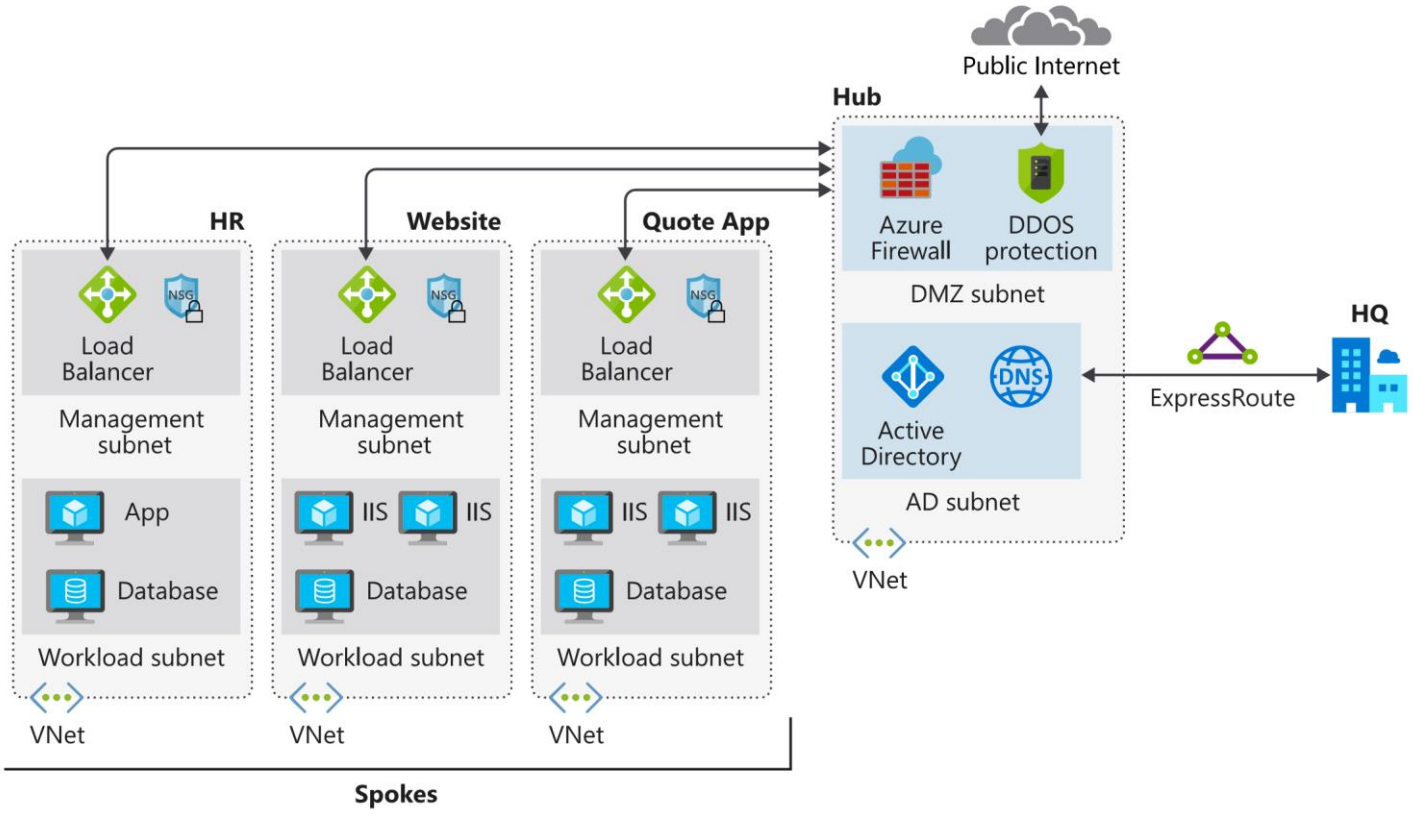
10 minutes

This module requires a sandbox to complete. A **sandbox** gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

Activate sandbox

You need to deploy your network infrastructure in a hub and spoke configuration for your resources. Additionally, your internal HR department wants to host a new internal HR system that shouldn't be accessible from the internet. The system should be accessible to everyone in the company, whether they work at headquarters or in a satellite office.

In this exercise, you'll deploy your network infrastructure, and then create a new virtual network to host the servers for your company's new HR system.



Environment setup

This deployment will create the Azure network resources matching the preceding diagram. With these resources in place, you can add the new HR virtual network.

First, create the virtual networks and subnets for your server resources. Run the following command:

Azure CLI Copy

```
az group deployment create \
    --resource-group [ sandbox resource group name ] \
    --template-uri https://raw.githubusercontent.com/MicrosoftDocs/mslearn-hub-and-spoke-network-architecture/master/azuredeploy.json
```

Create a new spoke in your virtual network

You can create a virtual network with the Azure portal, Azure CLI, or Azure PowerShell. Let's do the rest of this exercise through the Azure porta

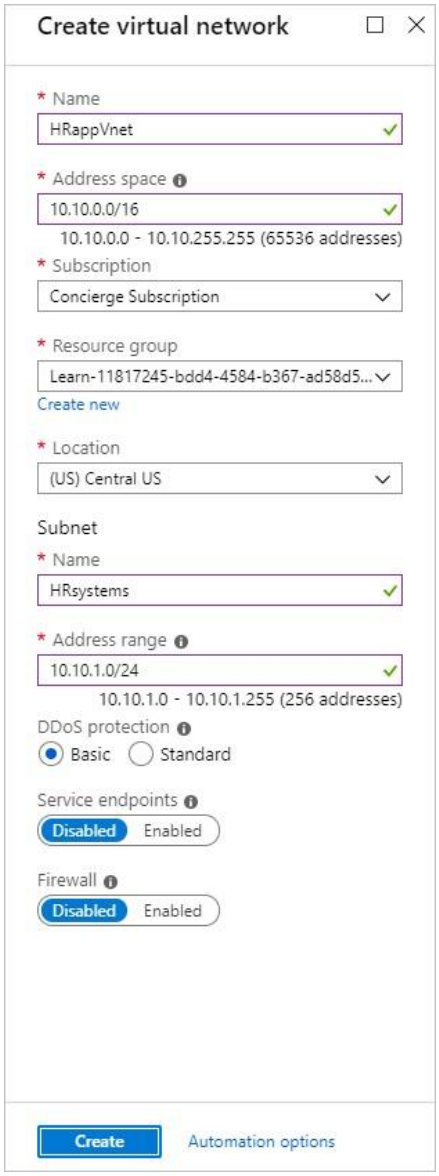
1. Sign in to the Azure portal by using the same account you used to activate the sandbox.
2. Select **Create a Resource** in the upper left of the Azure portal.
3. In the search box, enter **Virtual Network**, and then select the link with the same title in the list.
4. Select **Create** to start configuring the virtual network.

Configure the virtual network settings

The resource creation experience on the portal is a wizard that walks you through the initial configuration for the virtual network.

1. Use the values below to create the virtual network:

Property name	Field property
Name	**HRappVnet**
Address Space	**10.10.0.0/16**
Subscription	**Concierge subscription**
Resource Group	**[sandbox resource group name]**
Subnet – Name	**HRsystems**
Subnet – Address Range	**10.10.1.0/24**
DDoS Protection	**Basic**
Service Endpoints	**Disabled**
Firewall	**Disabled**



2. Select **Create** to start provisioning the virtual network.

Configure the hub virtual network peering

Now that you've created the third spoke, you need to configure the virtual network peering between the hub and spokes.

1. In the resources menu on the left, select **Virtual Networks** You should see the **HubVNet**, **WebVNet**, **QuoteVNet**, and **HRappVnet** virtual networks.
2. Select **HubVNet**.
3. Select the **Peerings** pane in the settings menu on the left.
4. On the **Peerings** pane, select **Add**, and fill in the fields as follows:

Property name	Field property
Name of the peering from hubVnet to HRappVnet	**gwPeering_hubVNet_HRappVnet**
Peer Details	**Resource Manager**
Subscription	**Concierge subscription**
Virtual Network	**HRappVnet**

Property name	Field property
Name of the peering from HRappVnet to hubVnet	**gwPeering_HRappVnet_hubVNet**
Allow virtual network access from hubVnet to HRappVnet	**Enabled**
Allow virtual network access from HRappVnet to hubVnet	**Enabled**
Allow forwarded traffic from HRappVnet to hubVnet	**Disabled**
Allow forwarded traffic from hubVnet to HRappVnet	**Disabled**
Configure gateways transit settings	**Unchecked**

5. Select **OK** to create the peering.
6. Close the **hubVNet** pane.

You've now peered the hub virtual network to the spoke virtual network. You've allowed traffic to be forwarded from the hub to the spoke, by using a VPN gateway in the configuration.

Create a network security group for the virtual network

To configure traffic flow, create a network security group.

1. Select **Create a Resource** in the upper left corner of the Azure portal.
2. In the search box, enter **network security group**, and then select the link with the same title in the list.
3. Select **Create** to start configuring the virtual network.
4. Enter **HRNsg** for the name. Then select the **[sandbox resource group name]** resource group and the same location as **HRappVnet**.
5. Select **Create** to provision the network security group.

You've created a network security group that can be assigned to each of the virtual networks.

Associate the network security group to the new HR virtual network

Now you associate the network security group to the virtual network.

1. Select **All services** in the upper left corner of the Azure portal.
2. In the search box, type **Network security group**, and then select **Network security groups**.
3. In the **Network security groups** pane, you should see the network security groups you created.
4. Select the network security group you created for the spoke, **HRNsg**.
5. Select the **Subnets** menu.
6. On the **Subnets** pane, select **Associate**.
7. Select the **HRappVnet** virtual network.
8. Select the **HRsystems** subnet.
9. Select **OK** to associate the network security group.

Configure the network security group rule to stop inbound HTTP traffic

You have a security requirement to meet for the HR application to be hosted on HRappVnet. There shouldn't be any inbound HTTP traffic from the spoke, because only internal employees need access. Configure the network security group rule to meet this requirement.

1. On the **HRNsg** pane, select **Inbound security rules** in the menu.
2. On the **Inbound security rules** pane, select **Add**.

Property name	Field property
Source	Any
Source port ranges	*
Destination	VirtualNetwork
Destination port ranges	80,443
Protocol	Any
Action	Deny
Priority	100
Name	Block-Inbound-HTTP-HTTPS

3. Select **Add** to add the rule.

You've now blocked inbound HTTP access from the spoke on port 80 and 443.

In this scenario, you created a spoke Azure virtual network, and then peered it with an existing hub virtual network. You then secured the traffic from this spoke by blocking inbound access on port 80 and 443, while ensuring it can connect via the hub.

Next unit: Secure your hub and spoke network

Continue

English (United States)

Azure Cloud Shell

This module requires a sandbox to complete. A sandbox gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft

Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

R Previous Unit 5 of 6 S Next T

200 XP

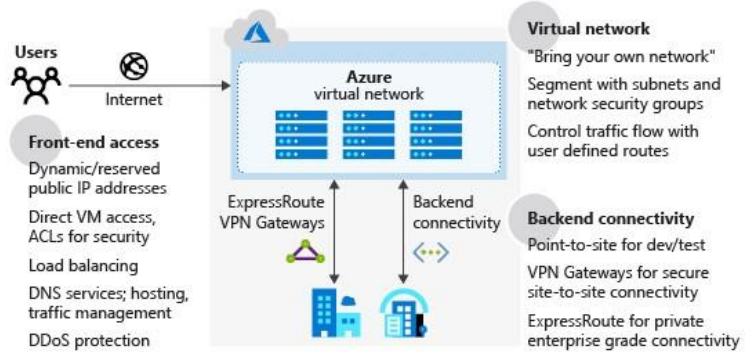
Secure your hub and spoke network

8 minutes

Azure provides a number of services to help enable an organization to secure and protect its cloud infrastructure. Your organization needs to understand how to secure its new network, and what other Azure services are available.

In this unit, you'll explore secure networking in the Azure platform, see an overview of Azure Firewall, and learn how to secure virtual networks by using network security groups.

Secure network design on Azure



The preceding diagram shows the Azure network infrastructure, and the methods that enable you to connect more securely your on-premises environment, Azure hosted resources, and the public internet.

There are several features to consider as part of securing a network design:

- **Azure Virtual Network**: Provides a base layer of security by logically isolating your environments in Azure, to prevent unauthorized or unwanted access.
- **Azure DNS**: A hosting service for your domain names. Azure DNS is a secure service that manages and resolves domain names in your virtual network.
- **Azure Application Gateway**: A dedicated virtual appliance that provides an application delivery controller as a service, including a web application firewall (WAF).
- **Azure Traffic Manager**: A service to control the distribution of user traffic in Azure.

Azure Load Balancer: Provides high availability and network performance to your Azure applications.
Perimeter network: Segments assets between your Azure virtual network and the internet.

Additionally, consider incorporating some of the following into your network architecture to improve network security:

- Network access controls, to make sure that your Azure services are accessible to only the users and devices you want.
- Network security groups as a packet filtering firewall, to control virtual network traffic.
- Route control, and forced tunneling, to define custom routes through your infrastructure, and ensure services can't connect to an internet device.
- Enabling a virtual network security appliance through Azure Marketplace.
- Using Azure ExpressRoute for a dedicated WAN link, to securely extend your on-premises networks to Azure.

Azure Security Center to prevent, detect, and respond to threats against your Azure services.
Azure Firewall as a network security service.

There's a wide variety of security solutions for your organization, many of which complement each other to provide additional layers of security. Microsoft has recommended best practices that you should align with overall. You then implement any features needed to meet your organization's internal security requirements.

Base components of Azure security for hub-spoke topologies

You want to ensure that your resources are protected from unauthorized access, or attack, by controlling your network spoke model, there are several components you need to implement:

Network security group

Each subnet within the topology has a network security group configured. The network security groups implement secu network traffic to and from each resource in the topology.

Perimeter

Configure a perimeter network in its own subnet in the hub virtual network for routing external traffic. The perimeter n network virtual appliances to provide security functionality, such as firewalls and packet inspection. You can route the o perimeter network through virtual appliances, so the traffic is monitored, secured, and audited.

Network virtual appliance

Network virtual appliances (NVAs) provide a secure network boundary by checking all inbound and outbound network t passes only the traffic that meets network security rules, essentially acting as a firewall.

Azure Firewall can replace some components discussed in this article, to control access to Azure network resources. For following section, "Azure Firewall."

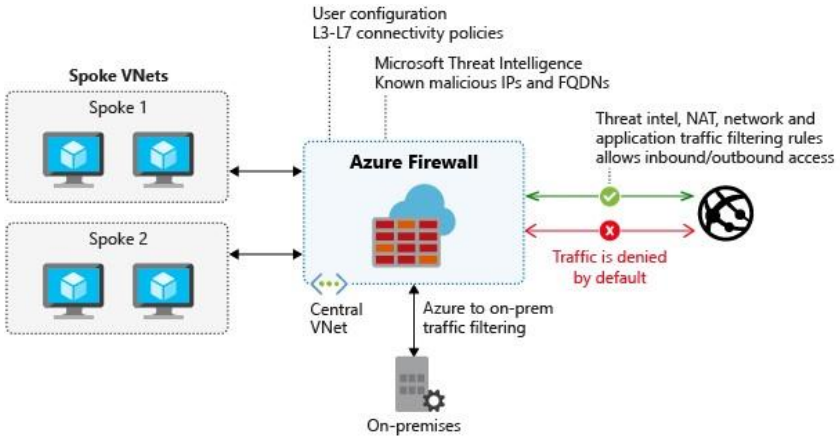
Azure

ExpressRoute creates a dedicated private WAN link between on-premises resources and an Azure gateway subnet in th add a network security appliance between the on-premises network and the ExpressRoute provider edge routers. This n unauthorized traffic from the virtual network.

Azure Firewall

Microsoft manages this network security service. It protects Azure virtual networks and their resources by letting you m connectivity policies centrally. Azure Firewall uses a static, public IP address for virtual network resources, allowing outs your virtual network traffic.

Azure Firewall is a fully stateful network firewall that tracks the operating state, and the characteristics of network conn Firewall enables central control of all network communications through policy enforcement. These policies can be enfo networks, regions, and Azure subscriptions. In a hub and spoke topology, Azure Firewall is typically provisioned in the h of traffic through the network.



The monitoring of Azure Firewall consists of reviewing the firewall and activity logs. Because Azure Firewall is integrated Logs, you can view the full logs there. Some logs are also available to view in the Azure portal.

The logs can be stored in an Azure Storage Account, streamed to an Azure Event Hub, or sent to Azure Monitor Logs.

Network security with network security groups

Network security groups (NSGs) enforce and control network traffic rules. Access is controlled by permitting or denying workloads in a virtual network. NSGs are rules-based, and evaluate traffic using a 5-tuple method. NSGs evaluate traffic port, destination IP, destination port, and protocol, to determine if traffic is allowed or denied.

Defining security rules

Security rules in an NSG provide the mechanism that defines the control of traffic flow. An NSG has a set of rules by def deleted, but you can override them with your own custom rules. The default rules are:

- Traffic originating from, and ending in, a virtual network is allowed.
- Outbound traffic to the internet is allowed, but inbound traffic is blocked.
- Azure Load Balancer is allowed to probe the health of virtual machines, or role instances.

Additional security considerations

The ability to control how traffic is routed through your resources is an important security measure to take. Azure helps of your overall infrastructure by offering other services:

- **Application security groups**: Provides central policy and security management for your applications. Use application se define detailed network security policies by using a moniker. You can then use a zero-trust approach, where only s permitted.
- **Azure Network** : Enables insights into your network logging and diagnostics. Network Watcher allows you to u health and performance of your Azure networks.
- **Virtual network service endpoints**: Extends your virtual network private address space to make it available to Azure serv endpoints allow you to restrict access to Azure resources.
- **Azure DDoS Protection**: Allows you to mitigate volumetric, protocol, and resource layer attacks.

Check your knowledge

1 How do network security groups help you control the flow of traffic in your virtual networks?

- They have strict rules defined to block inbound traffic from the internet, and can't be modified.
- They act like a stateful firewall, monitoring network traffic and enforcing policies you define.
- They allow you to create traffic rules, which can allow or deny traffic over specified ports and protocols.

This is the correct definition of a network security

2 How does using Azure ExpressRoute help improve your security?

- It provides a dedicated, private connection between your on-premises resources and Azure. Extra security is by adding network security appliances between edge routers.

  This is the correct definition of an ExpressRoute
- It provides a base layer of security by allowing you to segment and isolate your Azure resources.
- It routes traffic from your network to Azure over the internet.

Next unit: Summary

Continue

Previous Unit 6 of 6

100 XP

Summary

1 minute

In this module, you've explored migrating your on-premises infrastructure to Azure by using the hub and spoke model. You covered the components that you need to create hub and spoke networks, including Azure ExpressRoute, and how to secure them in Azure.

A hub and spoke architecture in Azure allows your business to quickly and easily adapt to new requirements. You can add spokes to segrega workloads with network security groups and Azure Firewall.

Clean up

The sandbox automatically cleans up your resources when you're finished with this module.

When you're working in your own subscription, it's a good idea at the end of a project to identify whether you still need the resources you created. Resources left running can cost you money. You can delete resources individually or delete the resource group to delete the entire s resources.

More information

- Hub and spoke topology
- Connect to Azure by using ExpressRoute
- Azure DDoS protection

Module complete:

Unlock achievement