Fortinet FortiGate and Fabric Connectors for Microsoft Azure
# Table of Contents

Introduction .................................................. 3
Architectural Diagram ...................................... 3
Deploying the Demo Solution ARM Template .............. 4
Deploying the ARM Template ................................ 10
Configuring FortiGate ........................................ 12
Use Case: Blocking a URL .................................... 16
Use Case: Intrusion Prevention ............................... 22
Use Case: Botnet C&C IP Blocking .......................... 26
Cleaning Up the Demo VNet Resources ...................... 28
Clean Up All Resources ....................................... 29
Connecting to the Existing Spoke VNet Resources ....... 30
Introduction

The FortiGate and Fabric Connector solution for Microsoft Azure is comprised of an automated deployment template for a FortiGate solution on Azure, with a built-in demo environment and three use cases. Once the demo is completed, this solution allows the user to clean up the demo environment and optionally connect and protect their existing environment without the need to redeploy FortiGate. Not only does this deliver a hands-on demo but users also have the opportunity to keep the Fabric Connector solution.

This solution delivers the core capabilities of the Fortinet FortiGate next-generation firewall (NGFW), along with Fortinet Fabric Connectors. Fabric Connectors are software-defined network (SDN) connectors that provide integration and orchestration of Fortinet products with key SDN solutions.

The Fortinet Security Fabric provides visibility into your security posture across multiple cloud networks, spanning private, public, and Software-as-a-Service (SaaS) clouds.

Using the Fabric Connector with Microsoft Azure Infrastructure-as-a-Service (IaaS) enables changes to attributes in the Azure environment to be automatically updated in the Fortinet Security Fabric.

The complete solution is deployed using an Azure Resource Manager (ARM) template that contains the user input parameters and uses an Azure hub-spoke topology. The hub is a virtual network (VNet) in Azure that acts as a central point of connectivity to spoke VNets, and it also can be extended to your personal cloud environment.

The spokes are VNets that peer with the hub and can be used to isolate workloads.

Architecture Diagram

![Architecture Diagram](image)

Figure 1: Architecture diagram of FortiGate and VNets.

Terms to Know

Hub VNets—Azure VNet is used as the hub in the hub-spoke topology as the central point of connectivity to different workloads hosted in the spoke VNets.

Spoke VNets—One or more Azure VNets that are used as spokes in the hub-spoke topology. Spokes can be used to isolate workloads in their own VNets. Each spoke VNet can include multiple subnets with workloads deployed.

VNet peering—Two VNets tethered using a nontransitive and low-latency peering connection. In a hub-spoke network topology, VNet peering connects the hub to each spoke.

Resource groups—Supports hub-spoke VNets in the same resource group.

User-defined routes (UDRs)—Used to force traffic destined to a spoke to be sent to FortiGate acting as a router at the hub VNet, allowing the spokes to connect to each other.
Network flow diagram

Figure 2: Hub-spoke network flow diagram.

Users achieve the same network flow if they deploy the template with default values. If any network-related parameters are changed, users will see a different network flow based on the new values.

- All inbound traffic is from Port 1 to Port 2
- All outbound traffic is from Port 2 to Port 1
- A static route is established from FortiGate to the workload (172.1.0.0/16 to 10.0.1.1(Gateway))

### Deploying the Demo Solution ARM Template

This section describes the step-by-step process for deploying the FortiGate and Fabric Connector on the ARM template. Follow this section in chronological order to ensure you have everything from the pre-deployment phase to get started.

#### Pre-deployment: Collecting the details from Azure

Before you deploy the template, you need to have the following information:

- **Azure tenant ID.** A tenant is a representation of an organization. An Azure tenant ID is the Azure Active Directory (Azure AD) GUID associated with an Azure subscription.

- **Azure client ID** (also called an “application ID”). The unique identifier Azure AD issues to an application registration that identifies a specific application and the associated configurations. This application ID (client ID) is used while performing authentication requests.

- **Azure client secret.** The password for API access in the registered application settings.

- **Azure subscription ID.** The subscription ID is a unique alphanumeric string that identifies your Azure subscription.

#### Create a service principal for Azure AD using Azure Portal

An Azure AD application must be created to generate the Azure client ID and the corresponding Azure client secret, or key. This application must be a service principal. Otherwise, the Azure SDN connector cannot read the inventory.

Before creating the service principal, verify that the required permissions are enabled. Make sure the subscription has the following permissions:

- Application administrator
- A user with application registrations set to “yes”
In the Azure subscription, the account must have the owner or user access administrator role.

Once the settings are configured properly, follow the steps below to create a service principal:

1. Sign into the Azure account and select: **Azure Active Directory > App registrations > New registration.**

2. Provide a name and select **Supported account types.** Add a **redirect URL** for the application. Select **Web** for the type of application you want to create. After setting the values, click on **Register.**
3. Once you register the application, the following screen appears. You need to copy the application/client ID.

4. To create the password, you will need to click on Certificates & secrets then click on + new client secret. Provide the needed description, select the expiry duration and click on Add.

5. Once you click on Add, the client secret/password will be created and shown under the value column. Copy the value and keep it for future use.

Create a service principal for Azure AD using Azure CLI

To get the application ID (Azure client ID), tenant ID, and password (Azure client secret), run this command:

```bash
az ad sp create-for-rbac -n sdntest-ap --skip-assignment
```

If you do not have the required permissions, contact the Azure portal administrator to create a service principal with network contributor and virtual machine contributor roles.

Copy all of the details from the command output and save them to use in the ARM template input parameters while deploying the template.

Role assignment to service principal using Azure Portal

Dynamic address objects, or IP addresses, in Azure can be resolved by the Fortinet Fabric Connector, provided that the service principal is
granted **network contributor** and **virtual machine contributor** roles for the target subscription.

Follow the steps below to assign a role to the service principal using Azure Portal:

1. Go to Azure Portal, search for **subscriptions** and click on **Subscriptions**.

![Azure Portal search tool](image-url)
2. When clicked, it opens the subscriptions page. Click on the target **subscription** where you are going to deploy the template.

![Figure 10: Azure Portal subscription selection.](image)

3. On the subscription page, click on the **Access control (IAM) > +Add > Add role assignment.**

![Figure 11: How to navigate to role assignment.](image)

4. Set the **Role** as **Network Contributor** and then select the service principal by typing the service principal name under **Select**, then click on **Save**.
Repeat steps 3 and 4 to assign **Virtual Machine Contributor** to the service principal.
Role assignment to the service principal using Azure CLI

Run the following commands in **Azure CLI** to assign roles to the application on the subscription level:

**Ex:**
```
az role assignment create --assignee <App ID> --role <Role Name> --subscription <subscription name/ID>
```
```
az role assignment create --assignee “fbc3c19f-0ce7-4XX9-aXXd-4e75f29330a3” --role “Network Contributor” --subscription “demoSubscription”
```
```
az role assignment create --assignee “fbc3c19f-0ce7-4XX9-aXXd-4e75f29330a3” --role “Virtual Machine Contributor” --subscription “demoSubscription”
```

![Azure CLI](image)

Figure 14: Running a script on Azure CLI.

Deploying the ARM Template

1. Click on the **Deploy to Azure** button at: [github.com/fortinetsolutions/Azure-Templates](https://github.com/fortinetsolutions/Azure-Templates)

2. Enter your Azure Portal credentials to access the **custom deployment** page. Complete the details, accept the terms and conditions, and click on the **purchase** button.

3. The deployment will begin. It will take about 8-12 minutes to deploy the template.
## ARM template input parameters

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Description</th>
<th>Allowed Values</th>
<th>Default Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>hubFotigateVNetCIDR</td>
<td>CIDR address for FortiGate VNet</td>
<td></td>
<td>10.0.0.0/16</td>
</tr>
<tr>
<td>FortiGatePublicFacing-SubnetCIDR</td>
<td>CIDR address for FortiGate public-facing subnet</td>
<td></td>
<td>10.0.0.0/24</td>
</tr>
<tr>
<td>FortiGateInsideSubnet-CIDR</td>
<td>CIDR address for FortiGate internal (private-facing) subnet</td>
<td></td>
<td>10.0.1.0/24</td>
</tr>
<tr>
<td>FortiGatePublicFacing-SubnetAddress</td>
<td>Static address for public-facing subnet</td>
<td></td>
<td>10.0.0.4</td>
</tr>
<tr>
<td>FortiGateInsideSubnetAddress</td>
<td>Static address of internal (private-facing) subnet</td>
<td></td>
<td>10.0.1.4</td>
</tr>
<tr>
<td>adminUsername</td>
<td>Username for FortiGate VM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>adminPassword</td>
<td>Password for FortiGate VM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ClientID</td>
<td>Application ID of AD application</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clientSecret</td>
<td>Password for AD application</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fabricConnectorWorkloadAd-AddressName</td>
<td>Name of the Fabric Connector address for workload VM with tag as “web”</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fabricConnectorFirewallAd-AddressName</td>
<td>Name of the Fabric Connector address for FortiGate VM with tag as “firewall”</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ExistingspokeVnet</td>
<td>This parameter will allow a user to choose between an existing VNet or to create a new VNet based on the values yes or no.</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>clientSecret</td>
<td>You need to select the existing resource group while deploying the template if you choose yes for the existing VNet.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>existingVnetName</td>
<td>If you choose ExistingspokeVNet as yes, provide the name of the existing VNet. Otherwise keep the default value as it is.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>existingVnetPrefix</td>
<td>If you choose ExistingspokeVNet as yes, provide the address prefix of the existing VNet. Otherwise keep the default value as it is.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>spokeDemoVnetCIDR</td>
<td>CIDR address for demo VNet (spoke)</td>
<td></td>
<td>172.1.0.0/16</td>
</tr>
<tr>
<td>spokeDemoSubnetCIDR1</td>
<td>CIDR address for demo subnet1 (spoke)</td>
<td></td>
<td>172.1.0.0/24</td>
</tr>
<tr>
<td>spokeDemoSubnetCIDR2</td>
<td>CIDR address for demo subnet2 (spoke)</td>
<td></td>
<td>172.1.1.0/24</td>
</tr>
</tbody>
</table>

Figure 15: ARM input parameters.
Configuring FortiGate

1. After a successful deployment, log in to the FortiGate UI using the `fortigateWebURL` provided in the outputs of the deployments. You will see a page that says “Your connection is not private.” Simply select the advanced button, then click on the “Proceed to…” link.

2. Provide the username and password to log in to the FortiGate web interface. These values can be accessed from `fortiGateVMUserName` and `fortiGateVMPassword` output parameters.
3. After login, the dashboard will be displayed.

Creating Fabric Connectors

The ARM template automates the creation of the Fabric Connector for Microsoft Azure. If you use the Fabric Connector, the changes made to the Azure environment are automatically reflected to the Fabric Connector address objects without any manual changes.

The template creates two Fabric Connector addresses with filter values: `tags.displayName=web` and `tags.displayName=firewall`.

To demonstrate how to use a dynamic address, the template creates virtual machines with tag names listed as `displayName` with values `web` and `firewall`. The address objects contain IP addresses within the Azure instance that are running.

When changes occur to addresses in the Azure environment, the Fabric Connector populates and updates the changes automatically based on the specified filtering condition. Administrators do not need to manually reconfigure the address object’s content.

The following screenshot shows the Fabric Connector for Microsoft Azure and the addresses. Achieve the same view by navigating to the path Security Fabric > Fabric Connectors.
For Fabric Connector addresses, click on **Policy & Objects > Addresses**.

It will take a couple of minutes to resolve the Fabric Connector address. If it does not resolve and you see any error, then double-click on the address to change any of the values such as **Name** or **Color** to a different value and click **OK**. After a couple of minutes, hover over the name of the Fabric Connector address to see resolved IP addresses.

To work through the use cases in this document you will need two virtual IPs (VIPs)—one for SSH and one to access the sample web application from the workload VM.

The two VIPs allow internet users to connect to your server protected behind a FortiGate firewall, without knowing the server’s internal IP address and only through ports that you choose.

**Virtual IP Definition**

Mapping one specific IP address to another is referred to as a destination NAT. When the central NAT table is not being used, this is known as a **virtual IP (VIP)** in FortiOS.
A firewall policy is the axis around which most of the other features of the FortiGate firewall revolve. A majority of the settings in the firewall at some point will relate to the firewall policies and the traffic that they govern. You will need to create three policies to enable the firewall policy:

1. A policy that establishes SSH connection into the workload VM.
2. A policy for accessing the sample web application.
3. A policy for allowing internet inside the workload VM.

View all policies by navigating to **Policy & Objects > IPv4 Policy**.

TCP ports 8888 (HTTP) and 22 (SSH) are opened for remote users to communicate with a server behind the firewall. The external IP address used is 10.0.0.4 (FortiGate public-facing IP) and is mapped to 172.1.0.4 (workload IP/spoke VM IP) by the VIP. The following screenshots select the interface as port 1—a public-facing interface under the **network** section—which allows internet users to connect to the FortiGate firewall.

View the VIPs created via the ARM template by navigating to **Policy & Objects > Virtual IPs**.

A firewall policy is the axis around which most of the other features of the FortiGate firewall revolve. A majority of the settings in the firewall at some point will relate to the firewall policies and the traffic that they govern. You will need to create three policies to enable the firewall policy:

1. A policy that establishes SSH connection into the workload VM.
2. A policy for accessing the sample web application.
3. A policy for allowing internet inside the workload VM.

View all policies by navigating to **Policy & Objects > IPv4 Policy**.
This view displays four policies:

1. The Fabric Connector as an IPv4 policy from the FortiGate VM on Port 2 to Port 1.
2. SSH into the workload VM.
3. Accessing the sample website.
4. Blocking the URL.

To view the sample website, use the link as shown below and open the test website from a web browser on your local machine:
http://<DNS of the FortiGate VM>:8888

Use Case: Blocking a URL

Open PuTTY from your local machine and SSH into the workload VM through the FortiGate VM DNS. You can get the required details from the outputs of the template deployment. Refer to this section for viewing the output parameters of the template deployment.

You can get the FortiGate VM DNS from the output parameter named “fortigate-Dns” and the username from the output parameter “workloadVM-UserName.” The password is from the output parameter “workloadVM-Password.”

SSH instructions for Windows

1. Grab all of the required values and SSH into the workload VM as shown below:
2. Click on **Yes**.

![PuTTY security alert](image)

Figure 26: PuTTY security alert.

3. After successfully connecting to the VM, provide the **username** and **password** of the workload VM.

![Workload VM password](image)

Figure 27: Workload VM password.

4. After a successful login, you can view the following screen:

![View upon login](image)

Figure 28: View upon login.
SSH instruction for Mac OS

1. Launch the Terminal application. It can be launched from Spotlight by pressing command and the spacebar, then typing “Terminal.”

![Figure 29: Terminal application.](image)

2. In the command prompt, enter the following SSH command and hit enter:

   Syntax: `ssh <username>@<DNS(OR)IP Address> -p <port number>`

   `ssh demouser@fortigatepw2x4.westus2.cloudapp.azure.com -p 2222`

![Figure 30: Command prompt.](image)

3. Log in to the remote server.

   ![Figure 31: Remote server login.](image)

You can log in to the workload VM using the FortiGate DNS with port 2222, as you have created a policy in the earlier steps to route the SSH traffic to the workload VM through FortiGate VM.
4. Once you log in to the workload VM, run the following command to access a specific URL.

   The sample URL is: **www.facebook.com**

   Command: **wget www.facebook.com**

   After login to the workload VM, the commands are the same for the Windows OS and Mac OS.

   ![Screenshot of command output](image.png)

   **Figure 32: Accessing a URL.**

   Once you access the sample URL, you can download the index.html file by entering the “ls” command.

   Control web content by blocking access to webpages containing specific words or patterns. This can be done by using the web content filter. The **web content filter** feature scans the content of every webpage that is accepted by a security policy.

5. Navigate to **Security Profiles > Web Filter**, and click **Create New**.

   ![Screenshot of web filter creation](image.png)

   **Figure 33: Creating a new web filter.**
6. Provide the name of the web filter profile under **Name**.

   ![Creating a new name](image)

   **Figure 34: Creating a new name.**

7. Scroll down until you find the **static URL filter**, then enable **URL Filter** and click on **Create**. A New URL Filter pop-up will appear. Fill in the **URL** with *facebook.com*. Select **Type** as **Wildcard** and **Action** as **Block**, and then click **OK** and **Apply**.

   ![Static URL filter screen](image)

   **Figure 35: Static URL filter screen.**

8. You will need to update an **IPv4 Policy (blocking-url-policy)** to block the URL in the workload VM. Click on **Policy & Objects > IPv4 Policy** then double-click on **blocking-url-policy** and enable **Web Filter**. Next, select **certificate-inspection** in the **SSL Inspection** category and click **Enable this policy**.
9. Now, try to access Facebook from the workload VM again. You will not be able to access the blocked web URL.

Move the block Facebook policy to the top of the list so that it will take the highest precedence if you have any similar policies. Move the policy by holding the left mouse button and move it to the top of the policy list.

10. View the log details in the FortiGate UI by navigating to Log & Report > Web Filter and clicking on the log entry. Then click on Details for more information.
The use case in Figure 38 shows the FortiGate Fabric Connector for Azure. Using the FortiGate Fabric Connector for Azure means the configuration of FortiGate policies is not dependent on the IP addresses of the resources connected to it. In this use case, `testip` is the Fabric Connector address, created by using resource tag "firewall."

The entire environment can be moved to a new Azure location on a different continent with different public IP addresses, even for internal resources. After the move, no reconfiguration is necessary. Everything will work just as it did before the move.

Use Case: Intrusion Prevention

The FortiOS intrusion prevention system (IPS) combines signature detection and prevention with low latency and strong reliability.

With intrusion protection, you can create multiple IPS sensors, each containing a complete configuration based on signatures. Then, you can apply any IPS sensor to any security policy.

The following use case explains how to enable IPS on FortiGate by using the EICAR test file.

In this example, you will create a new IPS sensor and include a filter that detects the EICAR test file and saves a packet log when it is found.

Create an IPS sensor

1. Go to Security Profiles > Intrusion Prevention. Select Create New and provide the name for the new IPS sensor under Name as IPS-test.
2. Click on **+add signatures**. A pop-up will appear. Click on **+Add Filter**.

   **For Severity**, select all of the options. You will need to repeat this to select all the values.

   **For Target**, select **Server** only.

   **For OS**, select **Linux** only.

   Search for **Eicar.Virus.Test.File** under the **Name** column. Select it and click on **Use Selected Signature**.

3. Under the **IPS Signatures** table, click on **Action** and select **Block**.
4. Under the **IPS Signatures** table, click on **Packet Logging** and select **Enable**.

![Figure 42: Enabling an IPS signature.](image)

5. Under the **IPS Signatures** table, click on **Packet Logging**, then **Status**, and select **Enable**.

![Figure 43: Enabling an IPS signature status.](image)

6. Once the IPS signature is enabled, click **OK**. Then, click **Apply**.

![Figure 44: Approving an IPS signature.](image)
Add the IPS sensor to the security policy to allow internet access

1. Go to Policy & Objects > IPv4 Policy and then edit the internet-inbound-policy. Under Security Profiles, enable IPS by selecting IPS-test, and then click on OK.

![Figure 45: Selecting IPS testing.]

2. Make sure that the internet-inbound-policy is at the top of the list. Follow the instructions from here on how to move policies.

![Figure 46: Moving the IPS signature to the top of the list.]

Test the IPS sensor

When the IPS sensor is configured and selected in the security policy, FortiGate will block any attempt to download the EICAR test file.

1. Enter the SSH into the workload VM using FortiGate DNS with port 2222, and run the following command:

   ```
   wget http://2016.eicar.org/download/eicar.com
   ```

   [Jump to this section](#) for instructions on how to get the login details of the workload VM from outputs of the template deployments.

   ![Figure 47: Running the command in SSH.]

   ```
   demouser@workloadVM-ly6lah:~$ wget http://2016.eicar.org/download/eicar.com
   Resolving 2016.eicar.org (2016.eicar.org)... 213.211.198.58
   Connecting to 2016.eicar.org (2016.eicar.org)|213.211.198.58|:80... connected.
   HTTP request sent, awaiting response...
   ```
No file is downloaded, which means the custom signature successfully detected the EICAR test file and blocked the download.

View the log details by navigating to **Log & Report > Intrusion Prevention.**

---

**Use Case: Botnet C&C IP Blocking**

FortiGuard Security Services for Azure continually update the botnet C&C domain list. The botnet C&C domain blocking feature prevents botnet website access at the DNS name-resolving stage, providing additional network protection.

1. Go to **Security Profiles > Intrusion Prevention** and click on +create new. Provide a **Name** and enable **botnet C&C** by setting **Scan Outgoing Connections to Botnet Sites** to **Block**. Click on **OK** and then click on **Apply**.

---

**Figure 48: Log and report dashboard.**

**Figure 49: Enabling botnet C&C.**
2. Go to **Policy & Objects > IPv4 Policy** and then edit the **internet-inbound-policy**. Under **Security Profiles**, enable **IPS** by selecting **botnet-cnc-ips** and click **OK**.

![Policy & Objects > IPv4 Policy](image)

---

**Test the botnet C&C IP blocking**

1. Go to FortiGate CLI by clicking on the icon in the FortiGate UI. Type the command below to see the botnet C&C IP address database:

   ```
   diag sys botnet list 0 10
   
   For more IPs, enter 0 100.
   ```

   ![Figure 51: Entering the command in FortiGate CLI.](image)

2. Enter SSH in the workload VM and type the following command:

   ```
   curl -v 46.166.135.177 -m 5
   
   Take any IP with Port 80 from the list and run the command.
   ```

   ![Figure 52: Running the command with Port 80.](image)
3. View the logs by navigating to **Log & Report > Intrusion Prevention.**

![Log and report dashboard.](image)

**Cleaning Up the Demo VNet Resources**

Once you have completed these demos and use cases, you can delete them and connect to your existing spoke VNet. Follow the steps below to clean up the demo VNet resources:

1. Copy the URL present in the output parameter named `demoVnetCleanupUrl` from the deployment outputs section and paste it in the browser. Select **Enter** to clean up the demo VNet resources.

![Selecting the URL from the outputs section.](image)
Cleanup will take approximately three to five minutes depending on the resources deployed. To ensure that the resources have been deleted, go back to the resource group to verify that the VNet resources are deleted.

You can delete the end-user VNet and end-user VM if they are not required for further testing. Go to the resource group and delete the resources related to end user.

**Clean Up All Resources**

Once the use case testing is completed, you can remove all the resources created through an ARM template by deleting the resource group. This action deletes all resources contained within the resource group.

This is applicable only for the demo environment if you had selected `existingSpokeVnet` as NO while deploying the ARM template.

Go to the Azure Portal, click on the resource group you created through the template, and delete the resource group.

**Deleting the service principal**

Delete the service principal that was created as part of the demo environment. This can be done using either the Azure Portal or as a CLI command.

**Deleting the service principal with Azure Portal**

In Azure Portal, navigate to `Azure Active Directory > App registrations` and search for the service principal to be deleted.
Click on **App > Overview > Delete > Yes.**

You will see a notification confirming the deletion.

**Deleting the service principal using the Azure CLI**

Run the following command in Azure Cloud Shell to delete the service principal:

```
az ad sp delete --id < App ID/object ID of the service principal >
```

**Connecting to the Existing Spoke VNet Resources**

To connect the hub VNet to the existing spoke VNet resources, choose **Yes** in the input parameter **existing spoke VNet** while deploying the ARM template. Deploy the template in the same resource group as the existing spoke VNet. The ARM template automates the VNet peering between the hub VNet and the spoke VNet.

1. The ARM template that connects the existing spoke VNet assumes that the existing VNet contains only one subnet.
2. Update the static route in FortiGate with the existing spoke VNet CIDR by navigating to **Network > Static Routes** in FortiGate.
3. Update the mapped IP address with the private IP of the workload VM in both virtual IPs (**ssh-wl** and **web-ip**) by navigating to **Policy & Objects > Virtual IPs.**
Associating subnets to the route table using Azure Portal

Perform the following steps if you choose Yes for the input parameter existing spoke VNet.

1. Go to the resource group with the existing VNet and then click on spoke route.

   ![Figure 60: Locating the resource group.]

2. Click on Subnets, then click on +Associate.

   ![Figure 61: Selecting associate subnets.]
3. In the **Associate subnet** page, choose the existing VNet and subnet.

![Figure 62: Selecting associate subnets and VNet.](image)

4. Click on **OK**.

![Figure 63: Confirming the change.](image)

### Associating subnets to the route table using Azure CLI

1. Run the following command to associate the existing subnet to the workload route table as shown below:

   ```bash
   az network vnet subnet update -g <Resource group name> -n <subnet name> --vnet-name <VNet name to connect> --route-table <spoke route table name>
   ```

   Example: `az network vnet subnet update -g existing-vnet-demo -n default --vnet-name existingvnet --route-table WorkloadRoutedzzqn`

   ![Figure 64: Example of running a command in Azure CLI.](image)