



Creating hybrid connections

Hybrid connections allow us to create secure connections with **Azure virtual networks (VNets)**. These connections can either be from on-premises or from other Azure VNets. Establishing connections to Azure VNets enables the exchange of secure network traffic with other services that are located in different Azure VNets, different subscriptions, or outside Azure (in different clouds or on-premises). Using secure connections removes the need for publicly exposed endpoints that present a potential security risk. This is especially important when we consider management, where opening public endpoints creates a security risk and presents a major issue. For example, if we consider managing virtual machines, it's a common practice to use either **Remote Desktop Protocol (RDP)** or PowerShell for management. Exposing these ports to public access presents a great risk. A best practice is to disable any kind of public access to such ports and use only access from an internal network for management. In this case, we use either a Site-to-Site or a Point-to-Site connection to enable secure management.

In another scenario, we might need the ability to access a service or a database on another network, either on-premises or via another Azure VNet. Again, exposing these services might present a risk, and we use either Site-to-Site, VNet-to-VNet, or VNet peering to enable such a connection in a secure way.

We will cover the following recipes in this chapter:

- Creating a Site-to-Site connection
- Downloading the VPN device configuration from Azure
- Creating a Point-to-Site connection
- Creating a VNet-to-VNet connection
- Connecting VNets using network peering

Technical requirements

For this chapter, the following are required:

- An Azure subscription
- Windows PowerShell

The code samples can be found at <https://github.com/PacktPublishing/Azure-Networking-Cookbook-Second-Edition/tree/master/Chapter08>.

Creating a Site-to-Site connection

A Site-to-Site connection is used to create a secure connection between an on-premises network and an Azure VNet. This connection is used to perform a number of different tasks, such as enabling hybrid connections or secure management. In a hybrid connection, we allow a service in one environment to connect to a service in another environment. For example, we could have an application in Azure that uses a database located in an on-premises environment. Secure management lets us limit management operations to being allowed only when coming from a secure and controlled environment, such as our local network.

Getting ready

Before you start, open your browser and go to the Azure portal at <https://portal.azure.com>.

How to do it...

To create a new Site-to-Site connection, we must follow these steps:

1. Locate the virtual network gateway (the one we created in *Chapter 5, Local and virtual network gateways*) and select **Connections**.
2. In **Connections**, select the **Add** option to add a new connection:

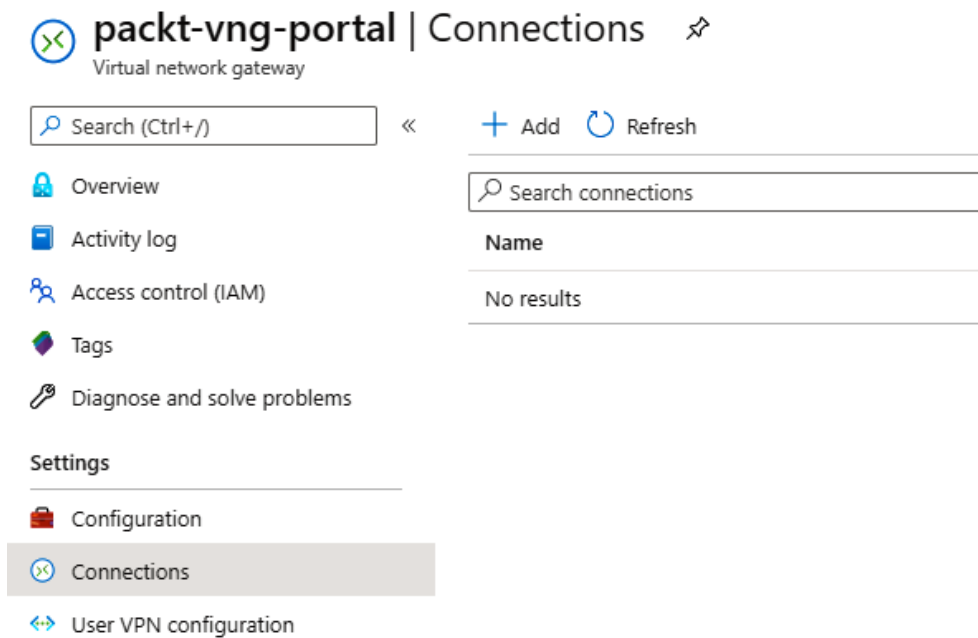
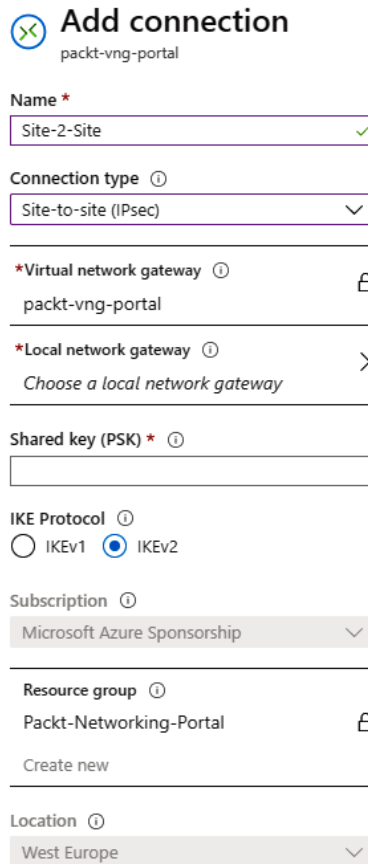


Figure 8.1: The Connections pane in the Azure portal

3. In the new pane, we need to enter the connection name and select **Site-to-site (IPsec)** for **Connection type**:



Add connection
packt-vng-portal

Name *
Site-2-Site ✓

Connection type ⓘ
Site-to-site (IPsec) ▼

*Virtual network gateway ⓘ
packt-vng-portal 🔒

*Local network gateway ⓘ
Choose a local network gateway >

Shared key (PSK) * ⓘ

IKE Protocol ⓘ
☐ IKEv1 ☒ IKEv2

Subscription ⓘ
Microsoft Azure Sponsorship ▼

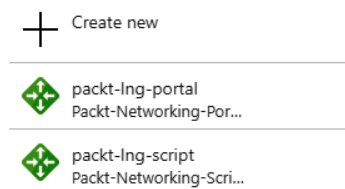
Resource group ⓘ
Packt-Networking-Portal 🔒
Create new

Location ⓘ
West Europe ▼

Figure 8.2: Adding connection attributes

4. Under **Local network gateway**, we need to select a local network gateway from the list (we created a local network gateway in *Chapter 5, Local and virtual network gateways*):

Choose local network gateway



+ Create new

packt-lng-portal
Packt-Networking-Por...

packt-lng-script
Packt-Networking-Scri...

Figure 8.3: Selecting a local network gateway

5. We need to provide a shared key in the **Shared key (PSK)** field that will be used for the IPsec connection. We also need to define the **IKE protocol** that will be used for security association. We can choose between **IKEv1** and **IKEv2**. Note that the options for **Subscription**, **Resource group**, and **Location** are locked and will be the same as they are for the virtual network gateway:

Add connection
packt-vng-portal

Name *
Site-2-Site ✓

Connection type ⓘ
Site-to-site (IPsec) ✓

***Virtual network gateway** ⓘ
packt-vng-portal 🔒

***Local network gateway** ⓘ
packt-lng-portal >

Shared key (PSK) * ⓘ
supersecretkey2020! ✓

IKE Protocol ⓘ
☐ IKEv1 ☒ IKEv2

Subscription ⓘ
Microsoft Azure Sponsorship ✓

Resource group ⓘ
Packt-Networking-Portal 🔒
Create new

Location ⓘ
West Europe ✓

Figure 8.4: Adding a new connection

6. Finally, we select **Create** and the deployment will start.

How it works...

Using the virtual network gateway, we set up the Azure side of the IPsec tunnel. The local network gateway provides information on the local network, defining the local side of the tunnel with the public IP address and local subnet information. This way, Azure's side of the tunnel has all the relevant information that's needed to form a successful connection with an on-premises network. However, this completes only half of the work, as the opposite side of the connection must be configured as well. This part of the work really depends on the VPN device that's used locally, and each device has unique configuration steps. After both sides of the tunnel are configured, the result is a secure and encrypted VPN connection between networks.

Let's take a look at how to configure our local VPN device.

Downloading the VPN device configuration from Azure

After creating the Azure side of the Site-to-Site connection, we still need to configure the local VPN device. The configuration depends upon the vendor and the device type. You can see all the supported devices at <https://docs.microsoft.com/azure/vpn-gateway/vpn-gateway-about-vpn-devices>. In some cases, there is an option to download configuration for a VPN device directly from the Azure portal.

Getting ready

Before you start, open the browser and go to the Azure portal at <https://portal.azure.com>.

How to do it...

To download the VPN device configuration, we must follow these steps:

1. Locate the **Site-2-Site** connection in the Azure portal. The **Overview** pane will be opened by default.
2. Select the **Download configuration** option from the top of the pane:

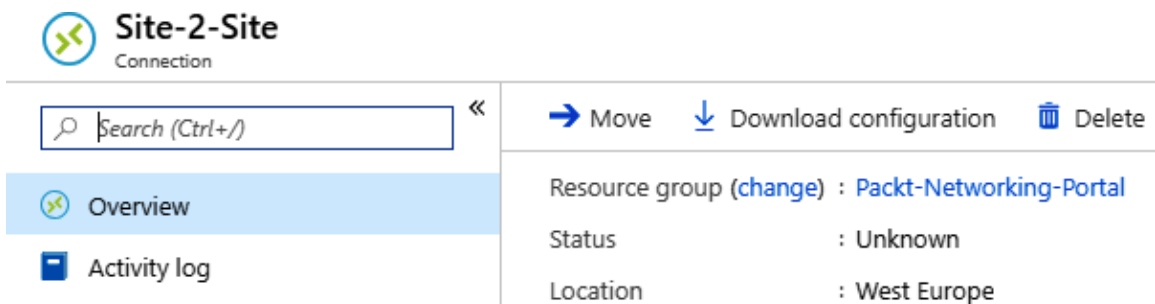
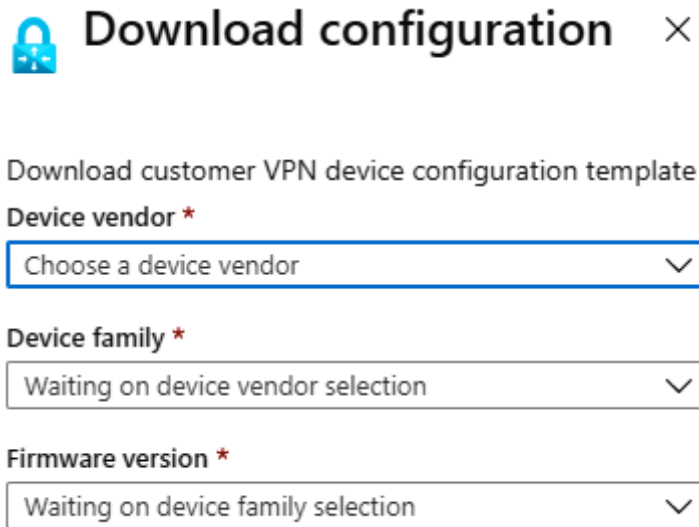


Figure 8.5: Site-2-Site connection overview in the Azure portal

3. A new pane will open, and you will see that all the options in the pane are predefined:



Download configuration ✕

Download customer VPN device configuration template

Device vendor *

Choose a device vendor ▼

Device family *

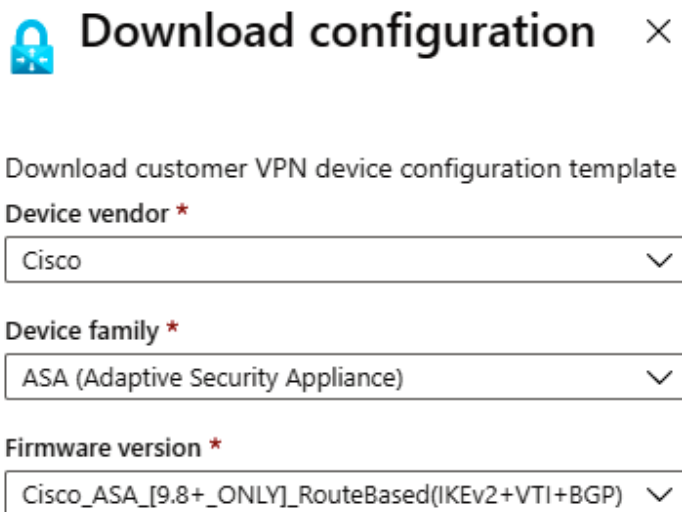
Waiting on device vendor selection ▼

Firmware version *

Waiting on device family selection ▼

Figure 8.6: Choosing VPN device configuration

4. Select the relevant options for the **Device vendor**, **Device family**, and **Firmware version** fields. Note that only some options are available, and not all the supported devices have these options. After all of these options have been selected, download the configuration file. The sample file (**Site-2-Site.txt** in the **Chapter 8** folder) can be found in the GitHub repository associated with this book:



Download configuration ✕

Download customer VPN device configuration template

Device vendor *

Cisco ▼

Device family *

ASA (Adaptive Security Appliance) ▼

Firmware version *

Cisco_ASA_[9.8+_ONLY]_RouteBased(IKEv2+VTI+BGP) ▼

Figure 8.7: Downloading the configuration file

- After using the configuration file for the local VPN device, both sides of the IPsec tunnel are configured. The **Status** value under the **Site-2-Site** connection will change to **Connected**:

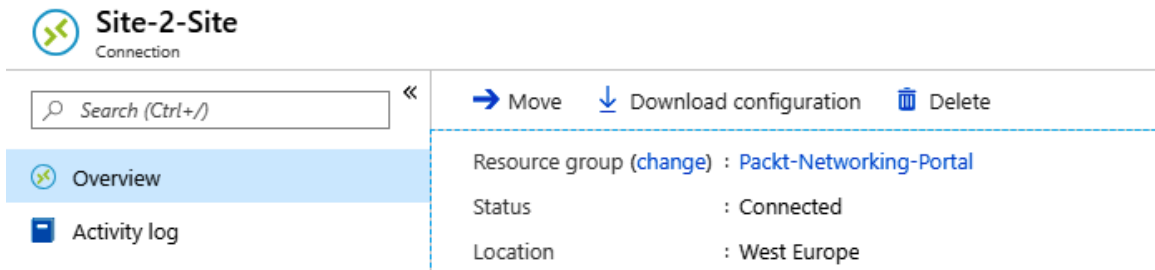


Figure 8.8: Checking the status of Site-2-Site connection

Now, let's have a look at how this connection functions in detail.

How it works...

After we set up the Azure side of the IPsec tunnel, we need to configure the other side, as well as the local VPN device. The steps and configuration are different for each device. In some cases, we can download the configuration file directly from the Azure portal. After the VPN device has been configured, everything is set up, and we can use the tunnel for secure communication between the local network and the VNet.

Creating a Point-to-Site connection

Accessing resources in a secure way is important, and this must be performed securely. It's not always possible to perform this using a Site-to-Site connection, especially when we have to perform something out of work hours. In this case, we can use Point-to-Site to create a secure connection that can be established from anywhere.

Getting ready

To create a Point-to-Site connection, we'll need to generate a certificate that will be used for the connection. To create a certificate, we must follow these steps:

- Execute the following PowerShell script to generate a certificate:

```
$cert = New-SelfSignedCertificate -Type Custom '
-KeySpec Signature '
-Subject "CN=P2SRootCert" '
-KeyExportPolicy Exportable '
-HashAlgorithm sha256 -KeyLength 2048 '
-CertStoreLocation "Cert:\CurrentUser\My" '
```



```

-KeyUsageProperty Sign '
-KeyUsage CertSign

New-SelfSignedCertificate -Type Custom '
-DnsName P2SChildCert '
-KeySpec Signature '
-Subject "CN=P2SChildCert" '
-KeyExportPolicy Exportable '
-HashAlgorithm sha256 -KeyLength 2048 '
-CertStoreLocation "Cert:\CurrentUser\My" '
-Signer $cert '
-TextExtension @"(2.5.29.37={text}1.3.6.1.5.5.7.3.2)"

```

2. Next, we need to export the certificate. Open **certmgr**, go to **Personal>Certificates**, select **P2SRootCert**, and then choose the **Export...** option:

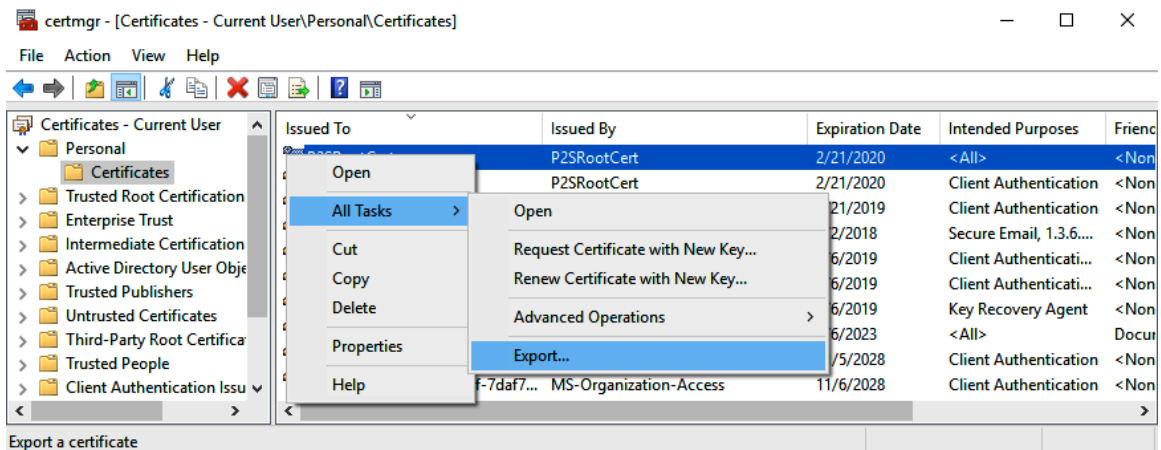


Figure 8.9: Exporting the certificate using certmgr

3. This will start the **Certificate Export Wizard**. Click **Next**.

4. Select the **No, do not export the private key** option and click **Next**:



Figure 8.10: Certificate Export Wizard

5. Select the **Base-64 encoded X.509 (.CER)** format and click **Next**:

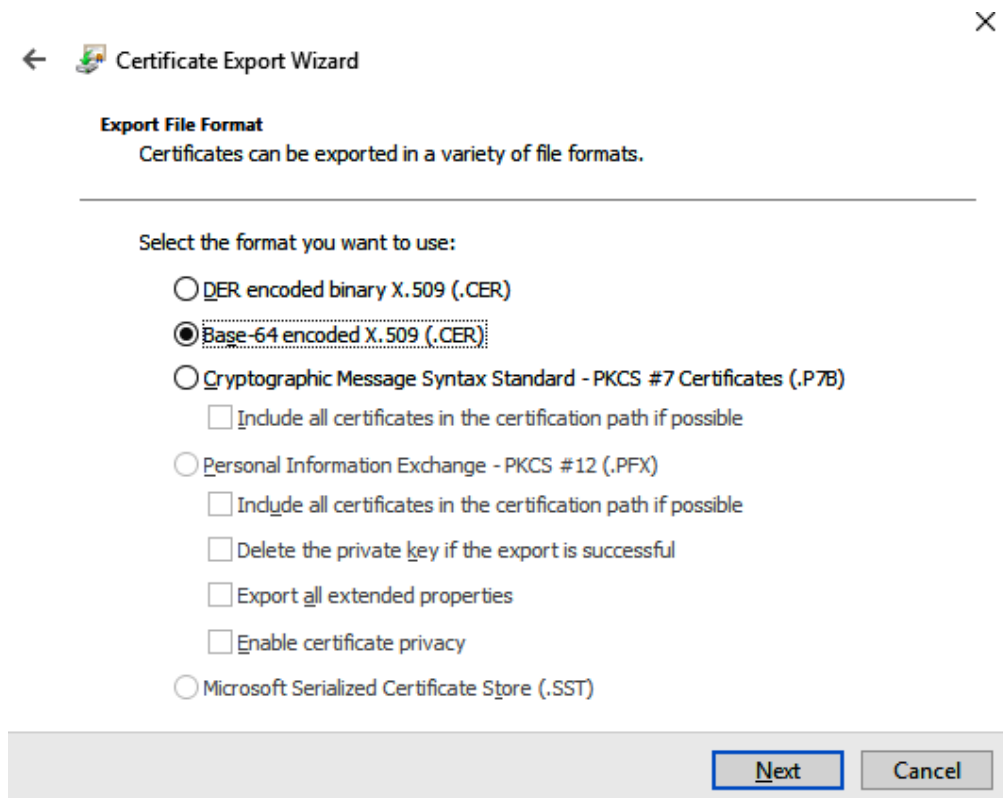


Figure 8.11: Selecting the export format

6. Select the location where you want to save the certificate and click **Next**.
7. Finally, we have the option to review all the information. After clicking **Finish**, the export will be complete:

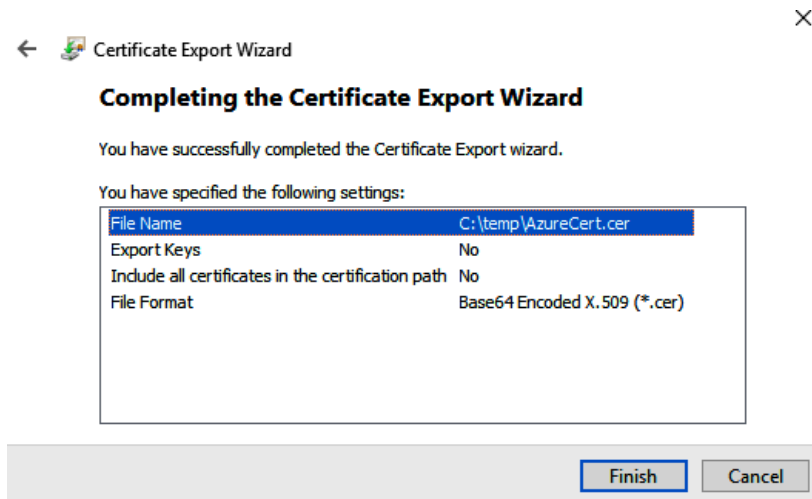


Figure 8.12: Completing the Certificate Export Wizard

Now, let's look at the steps to create a Point-to-Site connection.

How to do it...

To create a Point-to-Site connection, we need to do the following:

1. In the Azure portal, locate the virtual network gateway and **User VPN configuration**. Select **Configure now**:

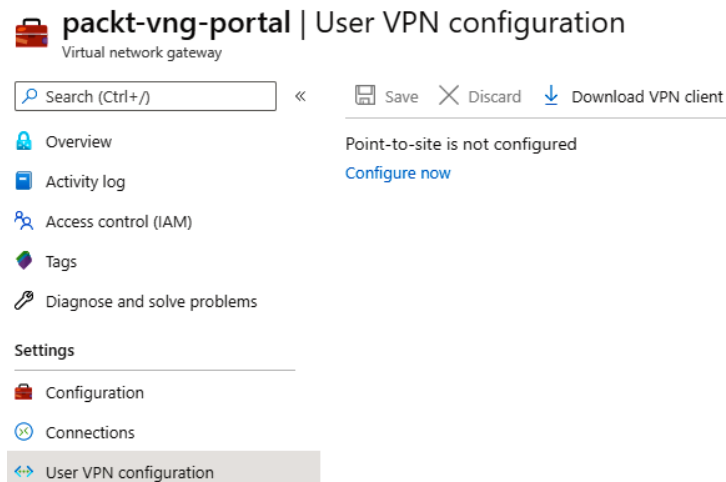


Figure 8.13: Configuring Point-to-Site connection

2. We need to define the **Address pool**. The address pool here cannot overlap with the address pool of the VNet associated with the virtual network gateway:

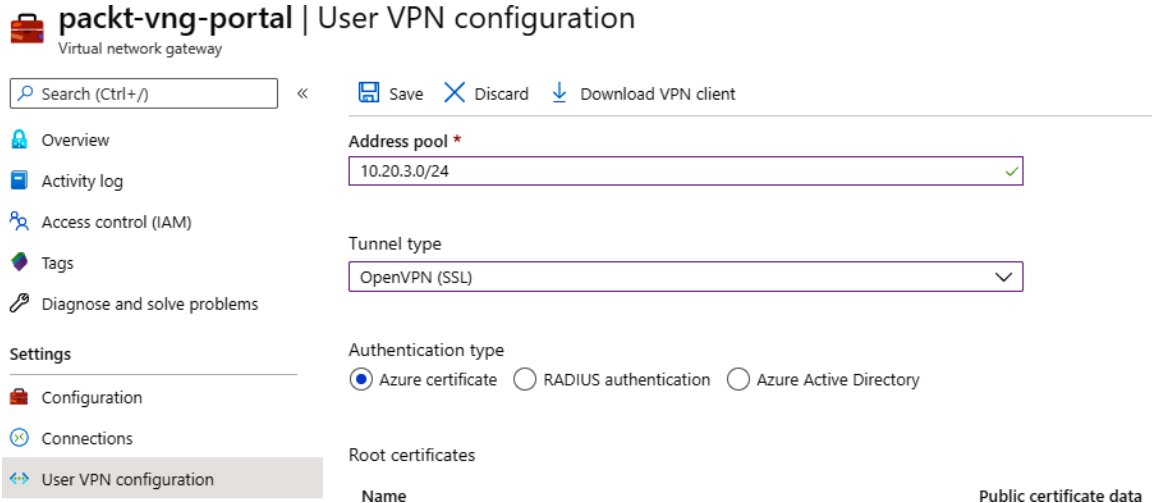


Figure 8.14: Adding the address pool

3. Next, we need to select a **Tunnel type** option from the list of predefined options. In this recipe, we'll select **OpenVPN (SSL)**, but any option is valid:

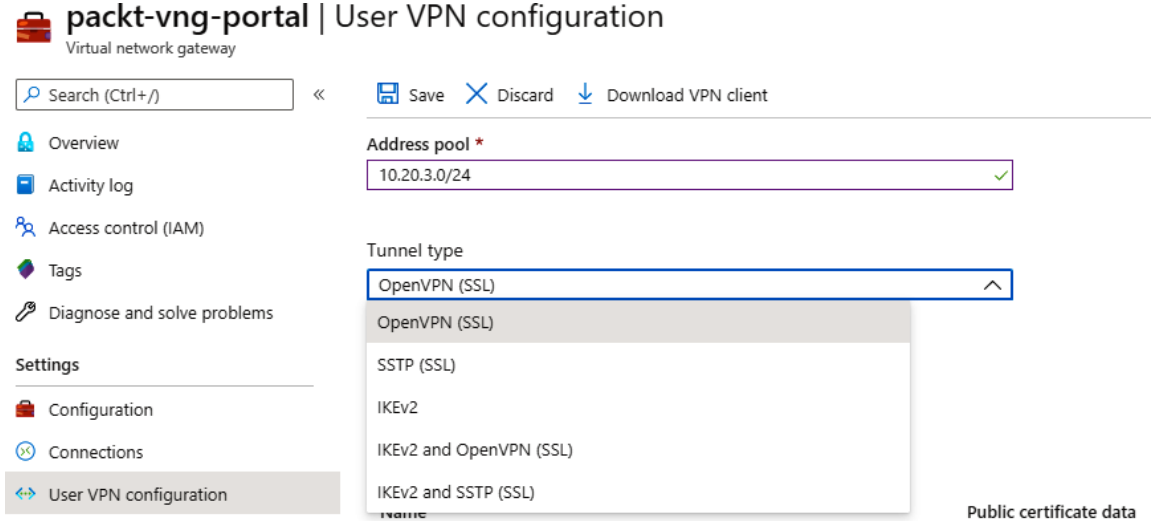


Figure 8.15: Selecting Tunnel type from the drop-down menu

4. Locate the exported certificate (from the *Getting ready* section) and open it in Notepad (or any other text editor). Select the value of the certificate and copy this value as follows:

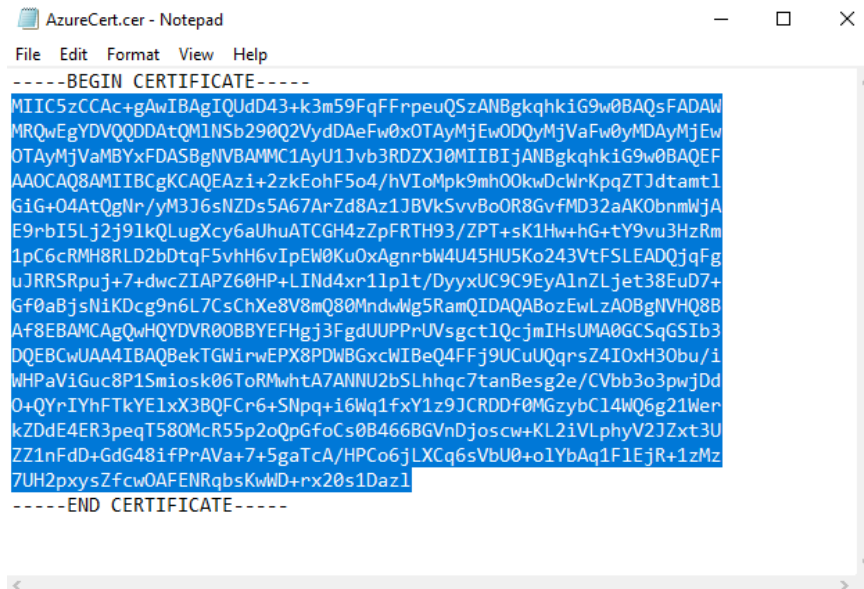


Figure 8.16: Opening the certificate in Notepad

5. In the Azure portal, we need to define the root certificate. Enter the name of the certificate and then paste the value of the certificate (from the previous step) into the **Public certificate data** field:

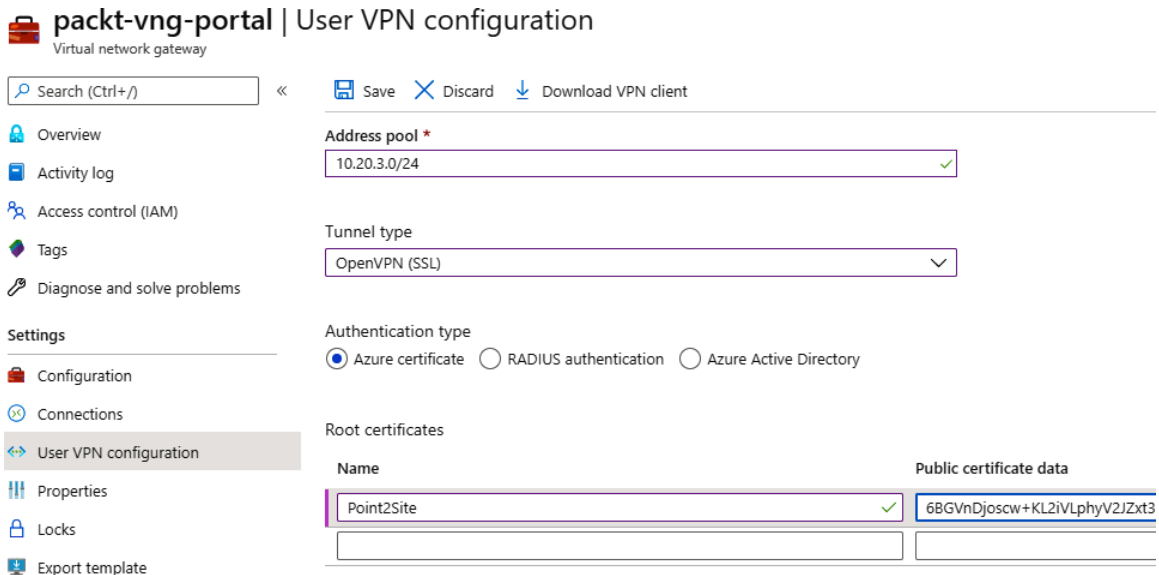


Figure 8.17: Defining the root certificate

- After clicking **Save** for the Point-to-Site configuration, a new option will become available: **Download VPN client**. We can download the configuration and start using this connection:

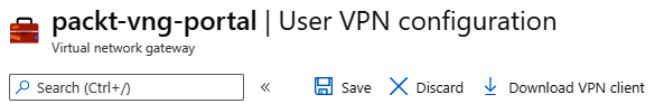


Figure 8.18: Download the configuration

Now, let's have a look at how it works.

How it works...

Point-to-Site allows us to access Azure VNets in a secure way. Access to a Site-to-Site connection is restricted to our local network, but Point-to-Site allows us to connect from anywhere. Certificate-based authentication is used, which uses the same certificate on both the server (Azure) and the client (the VPN client) to verify the connection and permit access. This allows us to access Azure VNets from anywhere and at any time. This type of connection is usually used for management and maintenance tasks, as it's an on-demand connection. If a constant connection is needed, you need to consider a Site-to-Site connection.

Creating a VNet-to-VNet connection

Similar to the need to connect Azure VNets to resources on a local network, we may have the need to connect to resources in another Azure VNet. In such cases, we can create a VNet-to-VNet connection that will allow us to use services and endpoints in another VNet. This process is very similar to creating a Site-to-Site connection; the difference is that we don't require a local network gateway. Instead, we use two virtual network gateways, one for each VNet.

Getting ready

Before you start, open your browser and go to the Azure portal at <https://portal.azure.com>.

How to do it...

To create a VNet-to-VNet connection, we must follow these steps:

- In the Azure portal, locate one of the virtual network gateways (associated with one of the VNets you are trying to connect to).
- In the **Virtual network gateway** pane, select **Connections** and select **Add** to add a new connection:

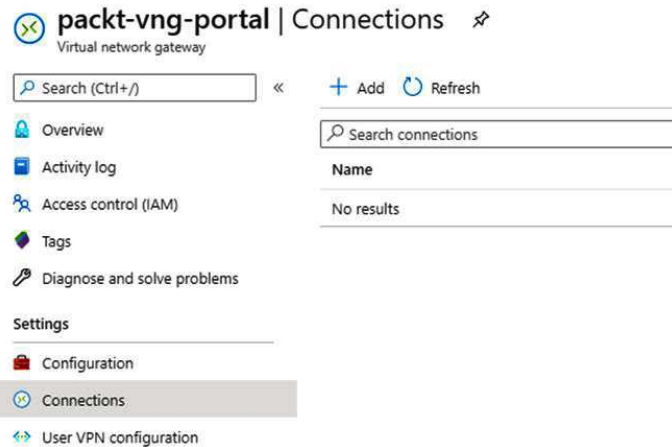


Figure 8.19: Adding a new connection

3. In the new pane, enter a **Name** value for the new connection and select **VNet-to-VNet** under **Connection type**:

The 'Add connection' form in the 'packt-vng-portal' contains the following fields and options:

- Name ***: A text input field containing 'VNet-2-VNet'.
- Connection type** ⓘ: A dropdown menu with 'VNet-to-VNet' selected.
- *First virtual network gateway** ⓘ: A text input field containing 'packt-vng-portal'.
- *Second virtual network gateway** ⓘ: A text input field containing 'Choose another virtual network gateway'.
- Shared key (PSK) *** ⓘ: An empty text input field.
- IKE Protocol** ⓘ: Radio buttons for 'IKEv1' and 'IKEv2', with 'IKEv2' selected.
- Subscription** ⓘ: A dropdown menu with 'Microsoft Azure Sponsorship' selected.
- Resource group** ⓘ: A dropdown menu with 'Packt-Networking-Portal' selected, and a 'Create new' link below it.
- Location** ⓘ: A dropdown menu with 'West Europe' selected.

Figure 8.20: Configuring the new connection

- The first virtual network gateway will be automatically highlighted. We need to select the second virtual network gateway:

Choose virtual network gateway

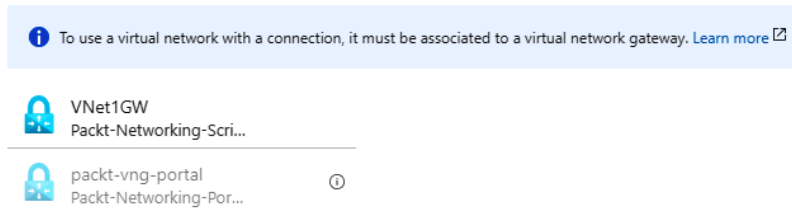


Figure 8.21: Choosing the virtual network gateway

- We need to provide a shared key for our connection before we select **Create** and start the deployment. Note that **Subscription**, **Resource group**, and **Location** are locked and that the values for the first virtual network gateway are used here:

Add connection
packt-vng-portal

Name *
VNet-2-VNet

Connection type ⓘ
VNet-to-VNet

***First virtual network gateway** ⓘ
packt-vng-portal

***Second virtual network gateway** ⓘ
VNet1GW

Shared key (PSK) * ⓘ
supersecretkey2020!

IKE Protocol ⓘ
☐ IKEv1 ☒ IKEv2

Subscription ⓘ
Microsoft Azure Sponsorship

Resource group ⓘ
Packt-Networking-Portal
Create new

Location ⓘ
West Europe

Figure 8.22: Providing a shared key for the connection

6. The deployment of VNet-to-VNet doesn't take long and should be done in a few minutes. However, it takes some time to establish connections, so you may see the status **Unknown** for up to 15 minutes before it changes to **Connected**:

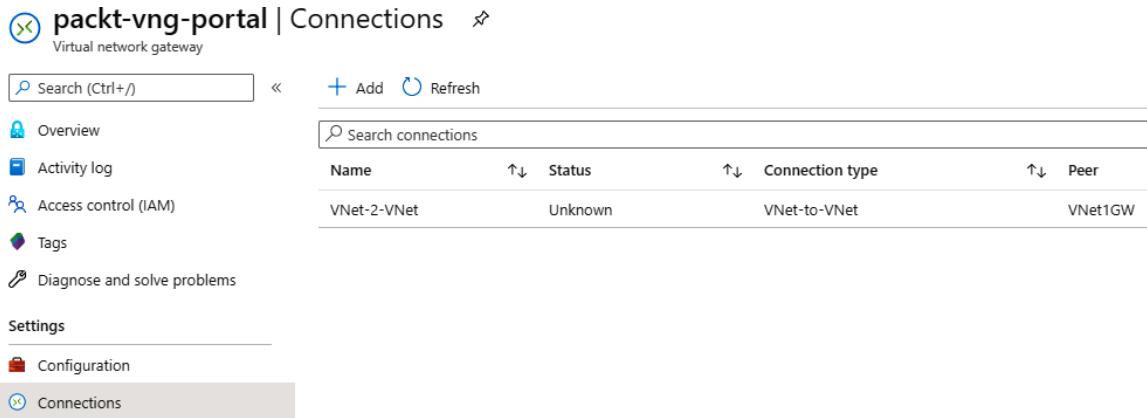


Figure 8.23: Deployment status of VNet-to-VNet

Now, let's have a look at its functioning in detail.

How it works...

A VNet-to-VNet connection works very similarly to a Site-to-Site connection. The difference is that Azure uses a local network gateway for information on the local network. In this case, we don't need this information; we use two virtual network gateways to connect. Each virtual network gateway provides network information for the VNet that it's associated with. This results in secure, encrypted VPN connections between two Azure VNets that can be used to establish connections between Azure resources on both VNets.

Now, let's learn about using network peering to connect VNets.

Connecting VNets using network peering

Another way to connect two Azure VNets is to use **network peering**. This approach doesn't require the use of a virtual network gateway, so it's more economical to use it if the only requirement is to establish a connection between Azure VNets. Network peering uses the Microsoft backbone infrastructure to establish a connection between two VNets, and traffic is routed through private IP addresses only. However, this traffic is not encrypted; it's private traffic that stays on the Microsoft network, similar to what happens to traffic on the same Azure VNet.

Getting ready

Before you start, open your browser and go to the Azure portal at <https://portal.azure.com>.

How to do it...

To create network peering, we must take the following steps:

1. In the Azure portal, locate one of the VNets that you want to connect to.
2. In the **Virtual network** pane, select the **Peerings** option, and select **Add** to add a new connection:

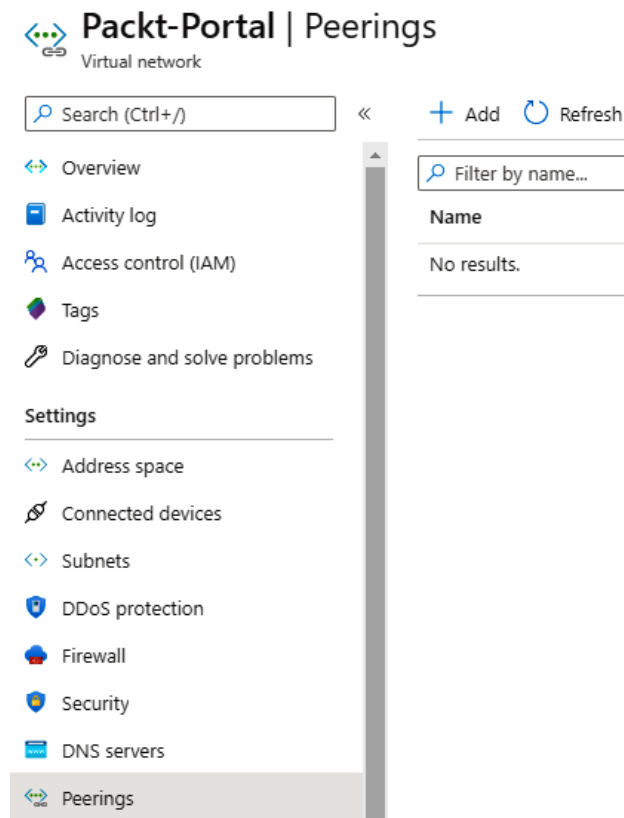


Figure 8.24: Adding a new network peering connection

3. In the new pane, we must enter the name of the connection, select a **Virtual network deployment model** option (**Resource manager** or **Classic**), and select the VNet we are connecting to. This information can be provided either by providing a resource ID or by selecting **Subscription** and **Virtual network** options from the drop-down menu. There are some additional configurations that are optional but provide us with better traffic control:

Add peering

Packt-Portal

i For peering to work, a peering link must be created from Packt-Portal to Packt-Script as well as from Packt-Script to Packt-Portal.

Name of the peering from Packt-Portal to Packt-Script *

Peering



Peer details

Virtual network deployment model ⓘ

☒ Resource manager ☐ Classic

☐ I know my resource ID ⓘ

Subscription * ⓘ

Microsoft Azure Sponsorship



Virtual network *

Packt-Script (Packt-Networking-Script)



Name of the peering from Packt-Script to Packt-Portal *

Peering



Configuration

Configure virtual network access settings

Allow virtual network access from Packt-Portal to Packt-Script ⓘ

Disabled **Enabled**

Allow virtual network access from Packt-Script to Packt-Portal ⓘ

Disabled **Enabled**

Configure forwarded traffic settings

Allow forwarded traffic from Packt-Script to Packt-Portal ⓘ

Disabled Enabled

Allow forwarded traffic from Packt-Portal to Packt-Script ⓘ

Disabled Enabled

Configure gateway transit settings

☐ Allow gateway transit ⓘ

Figure 8.25: Configuring peer details for a new connection

4. After a connection is created, we can see the information and the status for peering. We can also change the **Configuration** options at any time:

Peering

Packt-Portal

 Save  Discard  Delete

Name of the peering from Packt-Portal to Packt-Script

Peering

Peering status

Connected

Provisioning state


Succeeded

Peer details

Address space

10.11.0.0/16

Remote Vnet Id

/subscriptions/cb638267-a366-463c-bfe5-7a49311c27a8/resourceGroups/Packt-Networking-Scri... 

Virtual network

[Packt-Script](#)

Configuration

Configure virtual network access settings

Allow virtual network access from Packt-Portal to Packt-Script ⓘ

☐ Disabled ☒ Enabled

Configure forwarded traffic settings

Allow forwarded traffic from Packt-Script to Packt-Portal ⓘ

☒ Disabled ☐ Enabled

Configure gateway transit settings

☐ Allow gateway transit ⓘ

Configure Remote Gateways settings

☐ Use remote gateways ⓘ

Figure 8.26: Reviewing the peering information and status for a new connection

Now, let's have a look at its inner workings in detail.

How it works...

Network peering allows us to establish a connection between two Azure VNets in the same Azure tenant. Peering uses a Microsoft backbone network to route private traffic between resources on the same network, using private IP addresses only. There is no need for virtual network gateways (which create additional cost), as a virtual "remote gateway" is created to establish a connection. The downside of this approach is that the same VNet can't use peering and a virtual network gateway at the same time. If there is a need to connect VNet to both the local network and another VNet, we must take a different approach and use a virtual network gateway, which will allow us to create a Site-to-Site connection with a local network and a VNet-to-VNet connection with another VNet.

When it comes to network access settings, we have multiple options to control network traffic flow. For example, we can say that traffic is allowed from VNet A to VNet B, but denied from VNet B to VNet A. Of course, we can set it the other way around or make it bidirectional.

We can also control transit traffic when an additional network is in the mix. If VNet A is connected to VNet B, and additionally VNet A is connected to VNet C, we can control whether traffic is allowed between VNet B and VNet C as transit traffic through VNet A.

However, this only works if transit is not made via peering. If all networks are Azure VNets, and VNet A connected to VNet B via peering, and VNet B connected to VNet C via peering, the connection between VNet A and VNet C would not be possible via transit between VNets. This is because peering is a non-transitive relationship between two VNets. If VNet B is connected to VNet C via VNet-to-VNet (or to an on-premises network via Site-to-Site), transit would be possible between VNet A and VNet C over VNet B.

