Hybrid connectivity between Azure Stack and Azure using ExpressRoute

Overview

Hybrid cloud opens new scenarios for customers as it allows them to deploy their workloads based on some of the following business needs. 1. The need for their applications to span both on-premises datacenters and in the cloud, in order to take advantage of the benefits such as dynamic scaling and global reach. 2. The business need to comply with local policies and regulations like data sovereignty constraints that require that data must remain within the country. This brings a new challenge in terms of connectivity; because when application components span both, the cloud and on-premises datacenters, customer require a hybrid connectivity model that allows them to connect the different application components across locations, on-premises and the cloud.

With Microsoft Azure Stack, customers can deploy applications on-premises in their datacenters, or if they prefer, they could deploy the same application in the cloud in Microsoft Azure. Customers can also, if they choose to, deploy some application components on-premises in Azure Stack (for example, a VM running SQL Server 2016) and some components in Microsoft Azure (for example, some VMs running a web application). In this way, customers can take advantage of the best of both platforms, while keeping the same application development model across clouds.

To help you evaluate, hybrid scenarios between Azure and Azure Stack, this article will explain how to create a hybrid connection between a virtual network (VNet) in Azure and a virtual network in Azure Stack using Microsoft ExpressRoute, and specifically, we will be using ExpressRoute Private peering only. If you need to know more about ExpressRoute you can refer to the online documentation using this link. While you configure the connections, you will learn the different options for the hybrid connectivity between Azure and Azure stack.

Note: This document applies for Microsoft Azure Stack Development Kit (also known as Single Node) and Azure Stack multimode systems.

Hybrid connectivity options

Before we go into the details of how to connect the Azure Stack to Azure, let us see the various hybrid connectivity options available to connect to Azure.

1. Internet Connectivity
2. Secure Point-to-site connectivity
3. Secure Site-to-site VPN connectivity
4. Private ExpressRoute connectivity
Option 1 and 2 discuss how to connect from a single client to Azure. The Option 3 discussed here helps to connect an entire virtual network to a remote virtual network using Site-to-Site VPN (S2S VPN). It shows how to establish S2S VPN connectivity using two Azure stack deployments. You can use the model to connect Azure Stack to Azure in the same way. This document covers the details for the option 4 to connect Azure stack VNet to Azure VNet using ExpressRoute.

![Diagram 1: Hybrid connectivity models to connect to Azure](image)

**Hybrid connectivity concept**

**Requirements**

The connectivity between Azure Stack and Azure using ExpressRoute has some specific requirements; hence, it is important to understand the requirements first.

- Azure subscription to create ExpressRoute circuit and VNets in Azure
- A provisioned ExpressRoute circuit through a [connectivity provider](#)
- A router that has the ExpressRoute circuit connected to its WAN ports
- The LAN side of the router is linked to the Azure Stack Multitenant Gateway
- The Router must support Site to Site VPN connection between its LAN interface and Azure Stack Multitenant Gateway
- Ability to create multiple VRFs (Virtual Routing and Forwarding) if more than one tenant is added in your Azure Stack deployment

The following picture depicts the conceptual diagram on how the connectivity between Azure Stack and Azure through ExpressRoute would look like when it is completed. The process for setting up this connectivity involves each Azure Stack tenant to create their own ExpressRoute circuit using their...
Azure subscription and use the service key generated while creating the ExpressRoute circuit to link it to the virtual circuit (VC) in their connectivity provider’s portal. In addition, ExpressRoute router must allow a Site-to-Site (S2S) VPN connection from the Azure Stack multitenant gateway to the LAN interface of the router. By linking, each tenant’s S2S VPN and the respective ExpressRoute circuit in a separate Virtual Routing and Forwarding (VRF), we can ensure the end-to-end traffic isolation for each tenant.
Diagram 2: Conceptual diagram for connecting Azure Stack to Azure through an ER circuit
Azure Stack Network Architecture
The picture below depicts the network architecture of how the Azure Stack multi-tenant gateway connects to Azure through a S2S VPN connection.

Diagram 3: Network Architecture diagram for Azure Stack single node deployment
We will be using the same architecture to connect to Azure via ExpressRoute private peering. This will be achieved by using a S2S VPN connection from the virtual network gateway in the Azure Stack to an ExpressRoute Router. The steps below in this document explain how to create an end-to-end connectivity between two VNets from two different tenants in Azure Stack to their respective VNets in Azure. You could choose to add as many tenant VNets and replicate the steps for each tenant or use this document to deploy just a single tenant VNet.
Diagram 4: Architecture diagram for connecting Azure Stack to Azure through an ER circuit

**Configuration steps**

Part 1- Setup S2S VPN between Azure Stack VNet and on-premises ExpressRoute Router.

**Before you begin**

Please make sure you have the following requirements before you start this configuration:

- An ExpressRoute circuit provisioned and connected to an ExpressRoute Router
- Private Peering configured on the ExpressRoute circuit
- Azure Stack multinode or Development Kit successfully deployed with Azure Stack Technical Preview 3
- Configure Quotas for Compute, Network and Storage in the Azure Stack environment
Create a Plan and Offer and subscribe to an offer

Create the Network Resources in Azure Stack
Now you are going to create the resources you need to set up your Azure Stack environment as a tenant. The following steps illustrate what you will be doing. These instructions show how to create resources using the Azure Stack Portal, but you can also use PowerShell to accomplish this.

Log in as a tenant
A service administrator can log in as a tenant to test the plans, offers, and subscriptions that their tenants might use. If you do not already have one, create a tenant account before you log in and subscribe to the offer created by the service administrator that allows to create compute, network and storage resources.

Create the virtual network & VM subnet

1. Log in using a tenant account.
2. In the Azure portal, click New.
3. Select Networking from the Marketplace menu.
4. Click the Virtual network item on the menu.
5. Click Create near the bottom of the resource description blade. Enter the following values into the appropriate fields per this table.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>TenantAVNet1</td>
</tr>
<tr>
<td>Address space</td>
<td>10.1.0.0/16</td>
</tr>
<tr>
<td>Subnet name</td>
<td>TenantA-Sub1</td>
</tr>
<tr>
<td>Subnet address range</td>
<td>10.1.1.0/24</td>
</tr>
</tbody>
</table>
6. You should see the Subscription you created earlier populated in the Subscription field.
7. For Resource Group, either you can create a new Resource Group or if you already have one, select Use existing.
8. Verify the default location.
9. Click Create.

Create the Gateway Subnet

1. Open the Virtual network resource you just created (TenantA-VNet01) from the Dashboard.
2. On the Settings blade, select Subnets.
3. Click the Gateway Subnet button to add a gateway subnet to the virtual network.

4. The name of the subnet is set to GatewaySubnet by default. Gateway subnets are special and must have this specific name to function properly.
5. In the Address range field, type 10.1.0.0/24.
6. Click Create to create the gateway subnet.

Create the Virtual Network Gateway

1. In the Azure portal, click New.

2. Select Networking from the Marketplace menu.
3. Select Virtual network gateway from the list of network resources.
4. Review the description and click Create.
5. In the Name field, type GW1.
6. Click the Virtual network item to choose a virtual network. Select TenantAVNet1 from the list.
7. Click the Public IP address menu item. When the Choose public IP address blade opens click Create new.
8. In the Name field, type GW1-PiP and click OK.
9. The Gateway type should have VPN selected by default. Keep this setting.
10. The VPN type should have Route-based selected by default. Keep this setting.
11. Verify that Subscription and Location are correct. You can pin the resource to the Dashboard if you like. Click Create.

Local Network Gateway
The implementation of a local network gateway in this Azure Stack evaluation deployment is a bit different from an actual Azure deployment.

The purpose of the Local Network Gateway resource is to indicate the remote gateway at the other end of the VPN connection. For the hybrid connectivity scenario described on this article, the remote side is the LAN sub-interface of the ExpressRoute Router. For the scenario described on this article, for Tenant A, the remote address is 10.60.3.255 as shown in Diagram 4 above.

Create the Local Network Gateway Resource

1. Log in to the Azure Stack physical machine for POC1.
2. In the Computer field, type the name MAS-CON01 and click Connect. Sign in to the tenant portal.
3. In the Azure Stack portal, click New.

   ![Microsoft Azure Stack](image)

4. Select Networking from the Marketplace menu.
5. Select local network gateway from the list of resources.
6. In the Name field, type ER Router GW.
7. From our diagram 4 it is the IP address of ER router’s LAN sub-interface assigned to Tenant 1. From diagram 4 it is the 10.60.3.255 (you will replace it with the IP of the LAN Sub interface of your ExpressRoute router)
8. In the Address Space field, type the address space of the VNets you want to connect to in Azure. From diagram 2 and sample diagram 5 it is 192.168.1.0/24 and 10.100.0.0/16. (You will replace this addresses with your address of the VNets in Azure)
9. **Note:** This is steps assumes you will be using static routes for the S2S VPN connection between Azure stack gateway and the ER router.
10. Verify that your Subscription, Resource Group and location are all correct and click Create.

Create the Connection

1. In the Azure portal, click New.

   ![Microsoft Azure Stack](image)

2. Select Networking from the Marketplace menu.
3. Select Connection from the list of resources.
4. In the Basic settings blade, choose Site-to-site (IPSec) as the Connection type.
5. Select the Subscription, Resource Group and Location and click Ok.
6. In the Settings blade, choose the Virtual Network Gateway (GW1) you created previously.
7. Choose the local Network Gateway (ER Router GW) you created previously.
8. In the Connection Name field, type ConnectToAzure
9. In the Shared Key (PSK) field type abc123 and click OK.

Once the connection is created, you will be able to see the public IP address used by the Virtual Network Gateway. For this, in the Azure Stack portal, browse to your Virtual Network Gateway, In the properties, you’ll find the public IP address. Take note of this address, as we will use it in the next section.
Create a VM
To validate data traveling through the VPN Connection, you need VMs to send and receive data in the Azure Stack VNet. Create a VM in your VNet now and put it on your VM subnet in your virtual network.

1. In the Azure portal, click **New**.

2. Select **Virtual Machines** from the Marketplace menu.
3. In the list of virtual machine images, select the Windows Server 2016 Datacenter image.
4. On the Basics blade, in the Name field type VM01.
5. Type a valid user name and password. You will use this account to log in to the VM after it has been created.
6. Provide a Subscription, Resource Group and Location and then click OK.
7. On the Size blade, choose a VM size for this instance and then click Select.
8. On the Settings blade, you can accept the defaults; just ensure that the Virtual network selected is TenantAVNet1 and select the Subnet 10.1.0.24. Click OK.
9. Review the settings on the Summary blade and click OK.

**Note:** Repeat the above steps from “Login as Tenant” until “Create VM” section for each tenant VNet you want to connect to their respective VNets in Azure using their dedicated ExpressRoute circuit.

**Configure the NAT VM for gateway traversal (Azure Stack Development Kit only)**

Azure Stack Development Kit is designed to be self-contained and isolated from the network on which the physical host is deployed, the “External” VIP network that the gateways are connected to is not actually external, but instead is hidden behind a router doing Network Address Translation (NAT). The router is actually a Windows Server VM (MAS-BGPNAT01) running the Routing and Remote Access Services (RRAS) role in the POC infrastructure. You must configure NAT on the MAS-BGPNAT01 VM to allow the Site-to-Site VPN Connection to connect on both ends.

**NOTE** This configuration is required for Azure Stack Development Kit environments only. The multi node deployment does not have a NATVM

**Configure NAT**

Execute these steps in the Azure Stack POC environment:

1. Log in to the Azure Stack physical machine.
2. Press and hold [Windows Key] + R to open the Run menu and type mstsc and press Enter.
3. In the Computer field type the name MAS-BGPNAT01 and click Connect. Use Azurestack\Azurestackadmin credentials to login.
4. Click on the Start menu and right-click Windows PowerShell and select Run As Administrator.
5. Type ipconfig /all.
6. Find the Ethernet Adapter that is connected to your external side and note the IPv4 address bound to that adapter. In the example environment shown in diagram 6 below, it is 10.10.0.62 but yours will be different.
7. Type the following PowerShell command to designate the external NAT address for the ports that the IKE authentication uses. Remember to change the IP address to the one that matches your environment.

   ```powershell
   Add-NetNatExternalAddress -NatName BGPNAT -IPAddress 10.10.0.62 PortStart 499 -PortEnd 501
   ```
9. Next, you create a static NAT mapping to map the external address to the Gateway Public IP Address to map the ISAKMP port 500 for PHASE 1 of the IPSEC tunnel.

   Add-NetNatStaticMapping -NatName BGPNAT -Protocol UDP -ExternalIPAddress 10.10.0.62 -InternalIPAddress 192.168.102.1 -ExternalPort 500 -InternalPort 500

10. Finally, you must configure NAT traversal, which uses port 4500 to successfully establish and complete IPEC tunnel over NAT devices.

   Add-NetNatExternalAddress -NatName BGPNAT -IPAddress 10.10.0.62 PortStart 4499 -PortEnd 4501

   Add-NetNatStaticMapping -NatName BGPNAT -Protocol UDP -ExternalIPAddress 10.10.0.62 -InternalIPAddress 192.168.102.1 -ExternalPort 4500 -InternalPort 4500

   **Note:** The -InternalAddress parameter here is the Public IP Address of the Virtual Network Gateway you created earlier. To find this IP address, look at the properties of the Virtual Network Gateway blade, and find the value for the Public IP Address.

Part 2- Create the Network Resources in Azure.

Now that we have completed the configuration on the Azure Stack side, we need to deploy some resources in Azure. The diagram 5 below provides a sample architecture to walk through the steps to create your virtual network in Azure. You can use any name and addressing scheme for your VNet in Azure. However, the address range of the VNets in Azure and Azure stack must be unique and non-overlapping.
Diagram 5: Sample Tenant Network in Azure

In this architecture, we will use a standard hub (192.168.1.x/24) and spoke (10.100.x.x/16) VNet model to deploy the workloads in the spoke and connect the ExpressRoute circuit in the hub VNet and link the two VNets using VNet peering feature.

- Log in to your Azure portal with your Azure credentials
- Follow the steps here to create VNets as per your need. But in this document we will assume you will follow the steps to create a Hub VNet with 192.168.1.X/24 address space and a Spoke VNet with a 10.100.x.x/16 address space
- Peer the Hub and Spoke VNets using the steps here. When configuring VNet peering, make sure to select the following options:
  - From hub to spoke: Allow gateway transit
  - From spoke to hub: Use remote gateway
- Refer the steps here for the ExpressRoute Prerequisites before creating your ExpressRoute circuit
- Follow the steps here to create an Express Route circuit using your Azure subscription
- Get the service key from the previous step and share it with your hoster /Provider to provision your ER circuit at their end
- Configure Private Peering on the ExpressRoute circuit
- Create a virtual network gateway for ExpressRoute in the hub VNet using the steps here
- Link the ER circuit to the hub VNet using the steps here
- Deploy your workload VMs in the spoke VNet.
Repeat the same steps above for any additional tenant VNets you want to connect in Azure through their respective ExpressRoute circuit

Part 3- Configure the Router and connect to your ExpressRoute

We will use the following diagram to show all the components of the end-to-end infrastructure for guiding our configuration of the ExpressRoute Router. This diagram shows how two tenants A & B with their respective ER circuit is linked to their own VRF (Virtual Routing and Forwarding) in the LAN and WAN side of the ExpressRoute router to ensure end to end isolation between the two tenants all the way. Please use the IP addresses used in the router interfaces to follow the configuration sample provided.
Diagram 6: End-to-End connectivity between Azure Stack Tenant VNet and Azure VNets
You can use any router that supports IKEv2 VPN and BGP to terminate the S2S VPN connection from Azure Stack and use the same router to connect to Azure using ExpressRoute circuit. Here is a sample configuration from a Cisco ASR 1000 to support the above Network diagram.

**Note:**

1. In the section that shows steps to support routing between the ER router and Azure Stack multitenant gateway, the configuration assumes Azure stack is using static routing.
2. The below configuration highlights a two tenant (Tenant A and B) deployment with a dedicated VRF for each tenant. You could use it to add as many tenants/VRFs you need. Tenants must have their own ER circuit. (i.e. a logical connection or a VC (virtual circuit)) and linked to the same physical connection to your provider using a separate VLAN ID (VLAN 101, VLAN 102 as shown in the diagram 6 above). The configuration below uses an ER circuit through Layer 2 network provider with 802.1Q VLAN tagging

```
ip vrf Tenant A
  description Routing Domain for PRIVATE peering to Azure for Tenant A
  rd 1:1
  !
ip vrf Tenant B
  description Routing Domain for PRIVATE peering to Azure for Tenant B
  rd 1:5
  !
crypto ikev2 proposal V2-PROPOSAL2
description IKEv2 proposal for Tenant A
  encryption aes-cbc-256
  integrity sha384
  group 2
crypto ikev2 proposal V4-PROPOSAL2
description IKEv2 proposal for Tenant B
  encryption aes-cbc-256
  integrity sha384
  group 2
```
crypto ikev2 policy V2-POLICY2
description IKEv2 Policy for Tenant A
match fvrf Tenant A
match address local 10.60.3.255
proposal V2-PROPOSAL2
description IKEv2 Policy for Tenant B
crypto ikev2 policy V4-POLICY2
match fvrf Tenant B
match address local 10.60.3.251
proposal V4-PROPOSAL2
!
crypto ikev2 profile V2-PROFILE
description IKEv2 profile for Tenant A
match fvrf Tenant A
match address local 10.60.3.255
match identity remote any
authentication remote pre-share key abc123
authentication local pre-share key abc123
ivrf Tenant A
!
crypto ikev2 profile V4-PROFILE
description IKEv2 profile for Tenant B
match fvrf Tenant B
match address local 10.60.3.251
match identity remote any
authentication remote pre-share key abc123
authentication local pre-share key abc123
ivrf Tenant B
!
crypto ipsec transform-set V2-TRANSFORM2 esp-aes esp-sha-hmac
mode tunnel
crypto ipsec transform-set V4-TRANSFORM2 esp-aes esp-sha-hmac
mode tunnel
!
crypto ipsec profile V2-PROFILE
set transform-set V2-TRANSFORM2
set ikev2-profile V2-PROFILE
!
crypto ipsec profile V4-PROFILE
set transform-set V4-TRANSFORM2
set ikev2-profile V4-PROFILE
!
interface Tunnel10
description **S2S VPN Tunnel for Tenant A**
ip vrf forwarding Tenant A
ip address 11.0.0.2 255.255.255.252
ip tcp adjust-mss 1350
tunnel source TenGigabitEthernet0/1/0.211
tunnel mode ipsec ipv4
tunnel destination 10.10.0.62
tunnel vrf Tenant A
tunnel protection ipsec profile V2-PROFILE
!
interface Tunnel20
description **S2S VPN Tunnel for Tenant B**
ip vrf forwarding Tenant B
ip address 11.0.0.2 255.255.255.252
ip tcp adjust-mss 1350
tunnel source TenGigabitEthernet0/1/0.213
tunnel mode ipsec ipv4
tunnel destination 10.10.0.62
tunnel vrf VNET3
tunnel protection ipsec profile V4-PROFILE

interface GigabitEthernet0/0/1
description PRIMARY Express Route Link to AZURE over Equinix
no ip address
negotiation auto

interface GigabitEthernet0/0/1.100
description Primary WAN interface of Tenant A
description PRIMARY ER link supporting Tenant A to Azure
en encapsulation dot1Q 101
ip vrf forwarding Tenant A
ip address 192.168.1.1 255.255.255.252

interface GigabitEthernet0/0/2
description BACKUP Express Route Link to AZURE over Equinix
no ip address
negotiation auto

interface GigabitEthernet0/0/2.100
description Secondary WAN interface of Tenant A
description BACKUP ER link supporting Tenant A to Azure
en encapsulation dot1Q 101
ip vrf forwarding Tenant A
ip address 192.168.1.5 255.255.255.252
!
interface GigabitEthernet0/0/2.102
description Secondary WAN interface of Tenant B
description BACKUP ER link supporting Tenant B to Azure
encapsulation dot1Q 102
ip vrf forwarding Tenant B
ip address 192.168.1.21 255.255.255.252
!
interface TenGigabitEthernet0/1/0
description Downlink to --- Port 1/47
no ip address
!
interface TenGigabitEthernet0/1/0.211
description LAN interface of Tenant A
description Downlink to --- Port 1/47.211
encapsulation dot1Q 211
ip vrf forwarding Tenant A
ip address 10.60.3.255 255.255.255.254
!
interface TenGigabitEthernet0/1/0.213
description LAN interface of Tenant B
description Downlink to --- Port 1/47.213
encapsulation dot1Q 213
ip vrf forwarding Tenant B
ip address 10.60.3.251 255.255.255.254
!
router bgp 65530
  bgp router-id <removed>
  bgp log-neighbor-changes
description **BGP neighbor config and route advertisement for Tenant A VRF**

```plaintext
address-family ipv4 vrf Tenant A
  network 10.1.0.0 mask 255.255.0.0
  network 10.60.3.254 mask 255.255.255.254
  network 192.168.1.0 mask 255.255.255.252
  network 192.168.1.4 mask 255.255.255.252
  neighbor 10.10.0.62 remote-as 65100
  neighbor 10.10.0.62 description VPN-BGP-PEER-for-Tenant A
  neighbor 10.10.0.62 ebgp-multihop 5
  neighbor 10.10.0.62 activate
  neighbor 10.60.3.254 remote-as 4232570301
  neighbor 10.60.3.254 description LAN peer for CPEC:INET:2112 VRF
  neighbor 10.60.3.254 activate
  neighbor 10.60.3.254 route-map BLOCK-ALL out
  neighbor 192.168.1.2 remote-as 12076
  neighbor 192.168.1.2 description PRIMARY ER peer for Tenant A to Azure
  neighbor 192.168.1.2 ebgp-multihop 5
  neighbor 192.168.1.2 activate
  neighbor 192.168.1.2 soft-reconfiguration inbound
  neighbor 192.168.1.2 route-map Tenant A-ONLY out
  neighbor 192.168.1.6 remote-as 12076
  neighbor 192.168.1.6 description BACKUP ER peer for Tenant A to Azure
  neighbor 192.168.1.6 ebgp-multihop 5
  neighbor 192.168.1.6 activate
  neighbor 192.168.1.6 soft-reconfiguration inbound
  neighbor 192.168.1.6 route-map Tenant A-ONLY out
  maximum-paths 8
```

exit-address-family

```

!
```

description **BGP neighbor config and route advertisement for Tenant B VRF**

```plaintext
address-family ipv4 vrf Tenant B
```
network 10.1.0.0 mask 255.255.0.0
network 10.60.3.250 mask 255.255.255.254
network 192.168.1.16 mask 255.255.255.252
network 192.168.1.20 mask 255.255.255.252
neighbor 10.10.0.62 remote-as 65300
neighbor 10.10.0.62 description VPN-BGP-PEER-for-Tenant B
neighbor 10.10.0.62 ebgp-multihop 5
neighbor 10.10.0.62 activate
neighbor 10.60.3.250 remote-as 4232570301
neighbor 10.60.3.250 description LAN peer for CPEC:INET:2112 VRF
neighbor 10.60.3.250 activate
neighbor 10.60.3.250 route-map BLOCK-ALL out
neighbor 192.168.1.18 remote-as 12076
neighbor 192.168.1.18 description PRIMARY ER peer for Tenant B to Azure
neighbor 192.168.1.18 ebgp-multihop 5
neighbor 192.168.1.18 activate
neighbor 192.168.1.18 soft-reconfiguration inbound
neighbor 192.168.1.18 route-map VNET-ONLY out
neighbor 192.168.1.22 remote-as 12076
neighbor 192.168.1.22 description BACKUP ER peer for Tenant B to Azure
neighbor 192.168.1.22 ebgp-multihop 5
neighbor 192.168.1.22 activate
neighbor 192.168.1.22 soft-reconfiguration inbound
neighbor 192.168.1.22 route-map VNET-ONLY out
maximum-paths 8
exit-address-family
!
ip forward-protocol nd
!
ip as-path access-list 1 permit ^$
ip route vrf Tenant A 10.1.0.0 255.255.0.0 Tunnel10
ip route vrf Tenant B 10.1.0.0 255.255.0.0 Tunnel20
!
ip prefix-list BLOCK-ALL seq 5 deny 0.0.0.0/0 le 32
!
route-map BLOCK-ALL permit 10
   match ip address prefix-list BLOCK-ALL
!
route-map VNET-ONLY permit 10
   match as-path 1
!

**Test the connection**

After establishing the Site-to-Site, connection and the ER circuit, you should validate that you can get traffic flowing through it. This task is simple as it just involves logging in to one of the VMs you created in your Azure VNet and pinging the VM you created in the Azure stack POC environment or vice versa. To ensure that you are putting the traffic through the Site-to-Site connection and ER connection, you want to make sure that you ping the dedicated IP (DIP) address of the VM at both ends and not the VIP address of the VM. To do this, you need to find and note the address on the other end of the connection.

**Log in to the tenant VM in Azure Stack**

1. Log in to the Azure Stack Portal using a tenant account.
2. Click **Virtual Machines** in the left navigation bar.
3. Find **VM** that you created previously in the list of VMs and click it.
4. On the blade for the virtual machine, click **Connect**.

5. Open a Command prompt from inside the VM and type `ipconfig /all`.
6. Find the **IPv4 Address** in the output and note it. You will ping this address from the VM in Azure VNet. In the example environment, the address will be from the **10.1.1.x/24 subnet** but in your environment, it might be different. It should however fall within the **subnet you created for the Tenant VNet** subnet. As long as the Firewall settings within the source and destination VM allows ICMP, the ping should succeed.
Viewing data transfer statistics through the gateway connection

If you want to know how much data is passing through this connection, this information is available in the Connection blade in the Azure Stack portal. This is also another good way to verify that the ping you just sent actually went through the VPN and ER connection.

1. Log in to the **Microsoft Azure Stack Portal** using your tenant account.
2. Navigate to the resource group where your VPN Gateway is created and select the object type **Connections**
3. Click the **ConnectToAzure** connection in the list.
4. On the Connection blade, you can see statistics for **Data in** and **Data out**. You should see some non-zero values there.

![Connection Blade Screenshot](image-url)