

creating private IP for the azure from the vnet so that it will be effectively be brought into the vnet. but again it will use azure backbone network but connect to the new private ip instead of public ip. it will be used when internet/public access is restricted in your v-net.

Imagine, if Azure storage is a shop exist in some other country. we are opening a branch of it in our country(vnet) and assign a new private IP that is in the range of our vnet..

once it has the private ip within the vnet range.. it will be accessed as if any other resource in the same vnet.. more effective and secured. but it doesnt really exists physically within the vnet.. but it will act as if it is in the same vnet.. internally it will create a network interface(endpoint) that can be securely reached through azure backbone network route

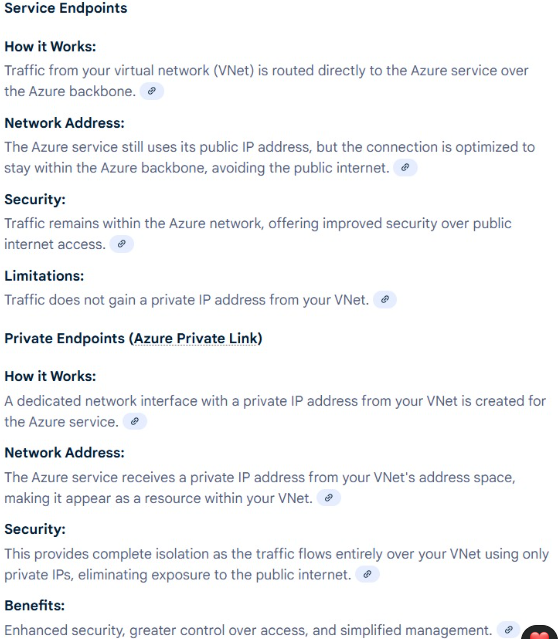
so using private endpoint we are bringing the service into our vnet "Effectively" not "physically"

with **service endpoint** we are not bringing the resource inside(not creating a private ip within the vnet IP range) but we are using again Azure backbone network for traffic, as we know the public ip address of the resource and internet(outbound rule NSG) is allowed to identify the resource.

Both Azure service endpoints and private endpoints send traffic over the Azure backbone network, but private endpoints provide greater isolation and security by bringing the Azure service into your virtual network using a private IP address. Service endpoints, on the other hand, route traffic from your VNet to the Azure service's public endpoint over the backbone network, still using the service's public IP.

Service Endpoint doesnt allow the traffic in public internet channel but azure backbone network which is secure and fast.. public ip is nothing but making the resource allowed to access publicly(if no firewall restrictions). Azure backbone network is Microsoft infra.. isolated from public internet. so safe!

vpn connection uses tunnels and tunnels are for encrypted data



**Public IP source from your laptop to Azure synapse:**

visit www.whatismyipaddress.com

at the top of homepage, you will see your public ip address. that is the source ip. you can give end ip as same. then whitelist your ip address in azure synapse firewall to allow you access to it.

but the network traffic is through public internet traffic not azure backbone network which is a secure flow of traffic

Vnet (Azure VM and Azure MI concept(Single Tenant) but not for Azure SQL)

Azure SQL(Multi-tenant) service has service and private endpoints to allow the connections from clients

Connectivity types

Proxy : connection through Gateway to the Azure SQL or MI. For the outside Azure clients(like on-prem). Gateway exposes its public IP to the clients and it listens on 1433 and forwards the traffic to the backend Azure SQL which actually listens on some dynamic port(11000-11999). Latency will be high comparatively as it involves hopping.

Redirect : First connection from client(inside Azure or On-prem that is connected through S2S, P2S) received by Gateway which listens on 1433 and authenticates the client and it is valid(inside Azure network or connected to Azure network through S2S or P2S) then it will forward the private IP of the backened Azure SQL/MI along with the dynamic port that backend SQL is listened to. Client has to allow outbount rules for both 1433 and the ports 11000-11999.

If the client(on-prem) connection is through public internet it will be considered as outsider and allow the traffic through proxy. But it is connected through S2S/P2S(private) then it will be considered as insider even it is on-prem and allow the traffice through redirect, which has less latency.

Within the Vnet machines can connect directly(using private IP). But it requires peering or vnet-vnet connection configuration if it is between the vnets.

Peering is within Azure only. Local peering is between the vnets that are in same region. If the vnets are in different regions, global peering to be configured.

**Scenario(A to B to C):**

1. **Connection between Azure Vnets using peering:**

Machine A can connect to Machine C, provided the peering and gateway transit are correctly configured following the hub-and-spoke model. This relies on VNet 1 acting as a "hub" for the on-premises connection, which then forwards traffic to the "spoke" VNet 2.

Here is a breakdown of how this configuration works:

* **Non-transitive peering**: By default, VNet peering is non-transitive. This means VNet 1 can communicate with VNet 2, but VNet 1 cannot automatically forward traffic to VNet 2 on behalf of an external network (like your on-premises network).
* **Gateway transit**: The "Allow gateway transit" setting on the VNet peering explicitly enables this transitive routing. This feature allows a peered VNet (VNet 2) to use the VPN gateway located in the hub VNet (VNet 1) to connect to the on-premises network.
* **Hub-and-spoke model**: This architecture is also known as a hub-and-spoke model, where VNet 1 is the hub that contains the VPN gateway, and VNet 2 is a spoke that relies on the hub for external connectivity.

**Necessary configuration steps**

For this setup to work, both sides of the VNet peering must be configured correctly:

* **On VNet 1 (the hub)**:
  + The VNet peering connection to VNet 2 must have the **"Allow gateway transit"** option enabled. This permits VNet 1's gateway to act as a transit point.
* **On VNet 2 (the spoke)**:
  + The VNet peering connection to VNet 1 must have the **"Use remote gateways"** option enabled. This tells VNet 2 to route traffic destined for the on-premises network through VNet 1's gateway.

When configured correctly, Azure automatically updates the routing tables in the spoke VNet (VNet 2) to include routes for the on-premises network. Traffic from Machine C will then be routed through the peering to VNet 1's VPN gateway and forwarded to Machine A on-premises.

**B. Connection between vnets using Vnet to Vnet(VPN Gateway) :**

If VNet 1 and VNet 2 are connected via a VNet-to-VNet VPN gateway connection instead of peering configuration, Machine A can connect to Machine C, but only if you enable **Border Gateway Protocol (BGP)** on both VPN gateways.

A standard VNet-to-VNet VPN connection is not transitive by default, similar to VNet peering. Traffic from your on-premises network will reach VNet 1 but will not automatically be forwarded to VNet 2.

**How BGP enables transit routing**

When you enable BGP, the two Azure VPN gateways become BGP peers and automatically exchange routes with each other. This allows for transitive routing in the following way:

1. The on-premises gateway uses BGP to advertise the routes for the on-premises network to the VPN gateway in VNet 1.
2. The VPN gateway in VNet 1 exchanges these routes with the VPN gateway in VNet 2 over their VNet-to-VNet connection.
3. VNet 2 now has the routes for your on-premises network.
4. When Machine C in VNet 2 sends traffic destined for Machine A on-premises, it routes the traffic through its VPN gateway, which uses the learned route to forward the traffic to VNet 1's gateway and then out to the on-premises network.

**Key differences between using BGP vs. peering for transit**

|  |  |  |
| --- | --- | --- |
| Feature | VNet Peering with Gateway Transit | VNet-to-VNet VPN with BGP |
| **Connectivity** | Traffic between VNets and to the on-premises network is routed through the high-speed Microsoft backbone. | All traffic is sent over an encrypted VPN tunnel, which relies on the public internet. |
| **Bandwidth** | Offers very high bandwidth with no restrictions. | The maximum bandwidth is dependent on the VPN Gateway SKU you choose. |
| **Cost** | You incur charges for data transfers (ingress/egress). | You pay an hourly cost for the VPN gateways, plus data transfer egress charges. |
| **Complexity** | Simple to configure by selecting "Allow gateway transit" on the hub VNet and "Use remote gateways" on the spoke VNet. | More complex to set up, requiring active-active gateways and manual BGP configuration. |
| **Use Case** | Preferred for high-bandwidth, low-latency connectivity, such as data replication and large backups. | Best for scenarios where encryption is a mandatory requirement, even if the traffic is not latency-sensitive. |

A Vnet Gateway is required to configure to talk to the machine in other vnet. Vnet Gateways is nothing but one or more VMs in a separte subnet(Gateway Subnet) within the Vnet that contains routing tables and other details regarding the networking.

**Virtual Network Name Listener(VNN):**

Machines in the same subnet of vnet cannot identify each other in Azure as the Azure blocks the broadcasting for security purposes that exposes the MAC(physical address of the machine that is mapped to IP address) that is used to identify the machines and maintained in ARP routing tables. So load balancer is required to identify the right machine for listener connectivity.

The Windows Server Failover Clustering (WSFC) service is responsible for listening to the probe port and responding to the load balancer. It is the underlying technology that manages the Always On Availability Group.

Here is a breakdown of how the process works:

* **Probe Port**: The load balancer is configured with a health probe that periodically checks for a response on a specific probe port on all the virtual machines (VMs) in the availability group.
* **WSFC Service**: On the current primary replica, the WSFC cluster service resource for the availability group is online and listening on that designated probe port. When it receives a request from the load balancer, it sends back a successful response.
* **Redirection**: Only the active primary replica will respond to the TCP probe from the load balancer. This allows the load balancer to correctly identify which VM is currently hosting the primary replica and redirect all client connections to it.
* **Failover**: In the event of a failover, the WSFC cluster service moves the cluster resource to a different node. The new primary replica then starts listening and responding on the probe port, while the old primary stops. The load balancer detects this change and begins routing new connections to the new primary.

**Why a load balancer probe is not needed for multi-subnet**

In a single-subnet configuration, a virtual IP (VIP) is assigned to the Availability Group Listener. Since Azure VMs in the same subnet might be on different hosts, a load balancer is necessary to hold this VIP and route traffic to the correct VM that is currently the primary replica. The load balancer's probe is the mechanism that determines which VM is currently active.

However, in a multi-subnet configuration, the replicas are in different subnets and cannot share a single VIP managed by a load balancer. Instead, the following happens:

* The Availability Group Listener is configured with a static IP address for each subnet it spans.
* The Windows Server Failover Cluster registers all these IP addresses in DNS for the listener name.
* The cluster is configured with an "OR" dependency for the listener's IP addresses, so it will come online if at least one of the IP addresses is active.

**How the primary replica is identified**

Client applications identify the primary replica using a special parameter in their connection string: MultiSubnetFailover=True. This setting instructs the client driver to behave differently during the connection process:

1. **DNS resolution**: The client connects to DNS to resolve the listener name. Because of the multi-subnet configuration, the DNS server returns all the IP addresses registered for the listener—one for each subnet.
2. **Parallel connection attempts**: Instead of trying each IP address sequentially, the client driver attempts to connect to all the returned IP addresses simultaneously.
3. **Connection success**: The client connects to the first IP address that successfully responds. This will be the IP address of the node that is currently the primary replica, since only its listener IP is online. This process eliminates the need for the load balancer's health probe and significantly reduces failover latency.