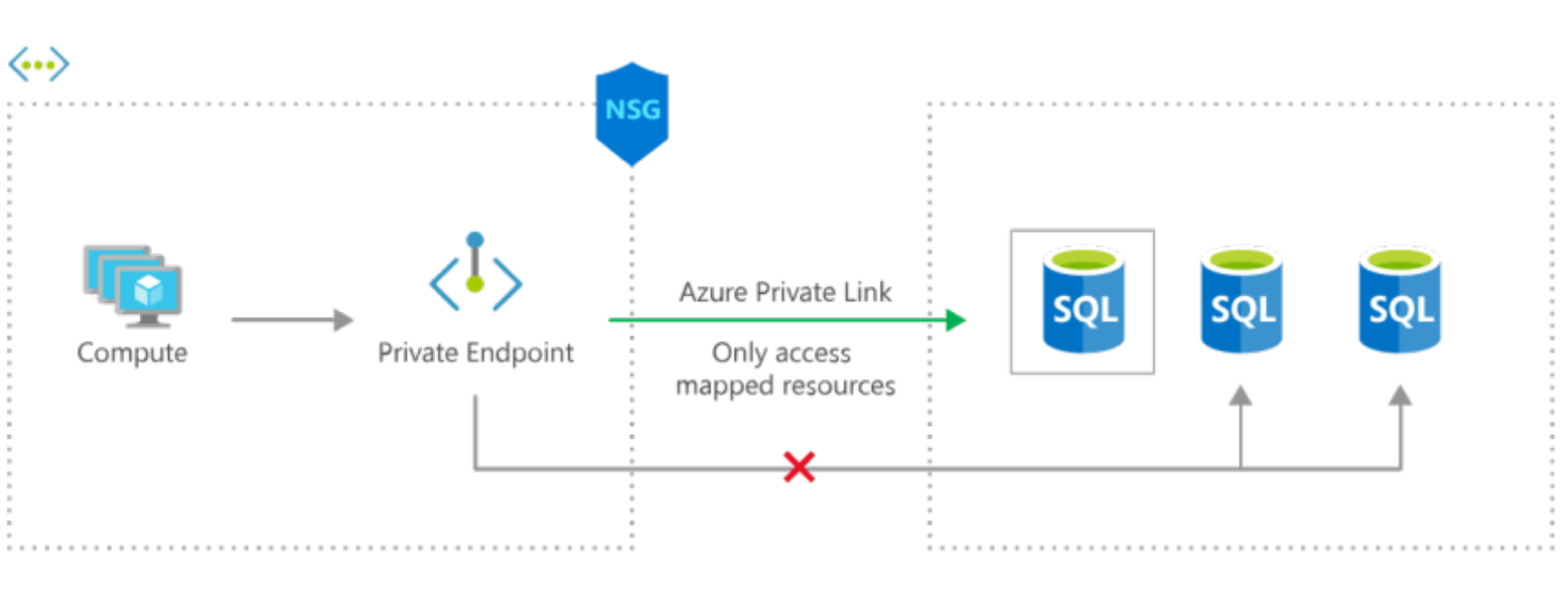
Cloud Network Security 101: Azure Private Link & Private Endpoints

Becki Lee | September 25, 2020

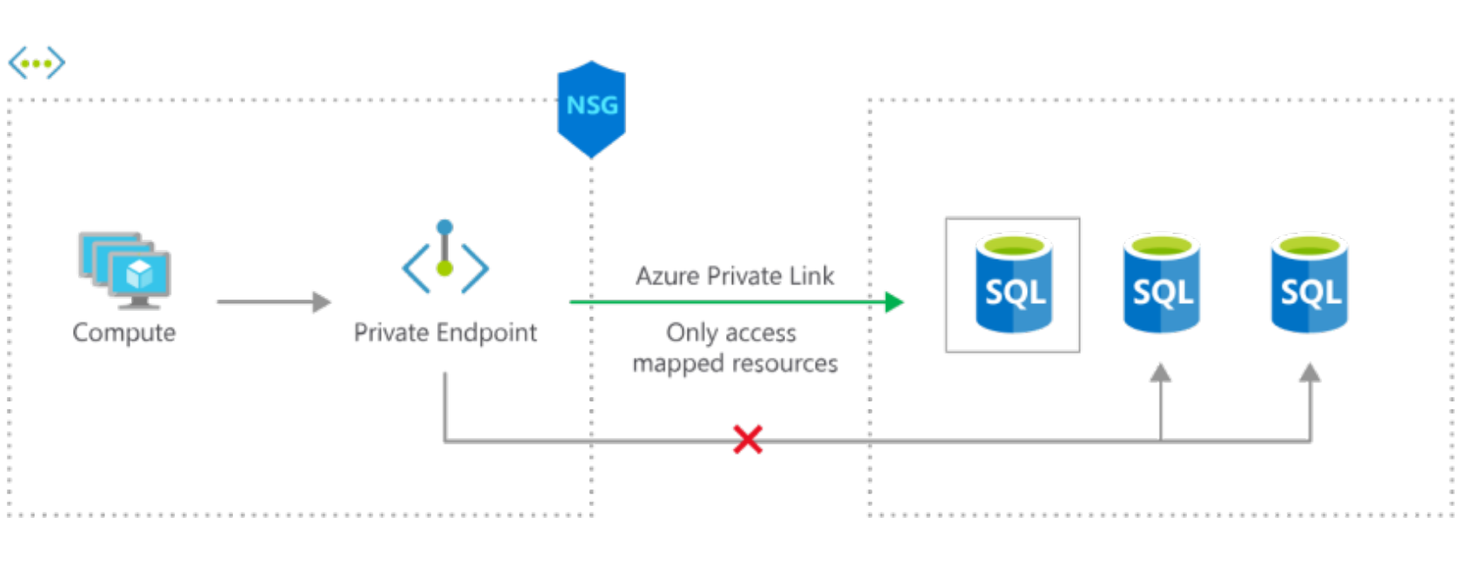


Azure offers two similar but distinct services to allow virtual network (VNet) resources to privately connect to other Azure services. Azure VNet Service Endpoints and Azure Private Endpoints (powered by Azure Private Link) both promote network security by allowing VNet traffic to communicate with service resources without going over the internet, but there are some differences. This three-part blog series goes into detail about both services.

- In part 1 of this series, we explored virtual network service endpoints.
- In part 2 **(this part!)**, we'll go over Private Link and private endpoints.
- In part 3, we'll compare and contrast the two and explain when to use which.

Ready to learn about Private Link and private endpoints? Let's go!

What is Private Link? What is a private endpoint?



Above, virtual machines in a VNet can use an Azure Private Link private endpoint to connect to a specific SQL database as if it were inside the VNet, even with an NSG denying outbound traffic. The private endpoint makes it possible for traffic to flow from a private IP address to another private IP in the same VNet -- no internet traversal necessary. Image from https://azure.microsoft.com/en-us/services/private-link/#how-it-works

Azure Private Link is a service that allows virtual network resources to privately connect to other resources as if they were part of the same network, carrying traffic across the Microsoft Azure backbone instead of the internet.

To take advantage of this service, you create a Private Link private endpoint. A private endpoint is a network interface that provides a private IP address to a service that would normally only be accessible to a VNet via public IP address.

For instance, every storage account has a public endpoint that by default is open to clients on any network. With a private endpoint, you can assign the storage account a private IP address from a VNet, and a virtual machine (VM) in that VNet can access the storage account without going over the internet. This is powerful because it means *you don't need to use public IP addresses*, either at the traffic source *or* destination. It's as if you're bringing the storage account inside the VNet.

But that's not all. The storage account still has a public endpoint, of course -- it doesn't go away just because you're not using it. So if desired, you can block *all* traffic to its public endpoint, further shielding it from network vulnerability.

Private endpoints can be enabled for two different categories of service:

1. **Azure PaaS services such as Azure Storage, Azure SQL Database, Azure Key Vault, and more.** See the full list here. For example, you can create an endpoint to securely connect a VM in a private subnet to a storage account. After creating a private IP address for the storage account, you can choose to block access to its public endpoint, so the only traffic that can reach it comes from the sanctioned subnet via the private endpoint.

Without a private endpoint, the VM would need to be assigned a public IP address, exposing it to the internet and all the threats that go along with it; the subnet would need a NAT or gateway device, requiring an extra step of configuration and potentially slowing traffic; the storage account would need to be open to clients on any network, so if credentials are leaked, anyone on the internet can access it. Not good!

2. **Your own service, if it's running behind a standard load balancer.** This is called a private link service, and you'll want to create one if you have customers who need to privately connect to your service from within their own VNet. After you've enabled your private link service, consumers create a private endpoint in their virtual network and send a request to connect to your service.

Without a private endpoint, your consumers would have to go over the internet to access your service. Once again, their VMs would need to be assigned public IP addresses, the associated subnets would need a NAT or gateway device, and your service resources would have to be accessible from the internet, too.

(Private link services are beyond the scope of this article, so we're going to focus on using private endpoints with Azure PaaS services.)

Benefits of private endpoints

Now, let's look at some of the benefits that private endpoints introduce. If you've read part 1 of this series, where we discussed service endpoints, this will sound familiar:

- **Enhanced security:** Since private endpoints promote private connectivity to target resources (which Azure calls *private link resources*), there's no need to assign a public IP address on the VNet resource end. Without a public IP address, malicious actors can't scan a VM's open ports for vulnerabilities and bring down your application or steal data. Additionally, you can map a private endpoint to a specific resource or even sub-resource (e.g., storage account or blob), so there's less risk of data exfiltration. (You can find a list of available private link resource and sub-resource types in the Azure documentation.)
- **Optimized routing:** The private endpoint provides a direct route over the Azure backbone network from the VNet to the private link resource, so there are no extra hops to slow down traffic.
- **Simpler network architecture:** Since traffic flows from the VNet resource to the private link resource over the Azure backbone network, you don't need to assign a public IP address or configure a NAT or gateway device.
- **On-premises support:** Private endpoints enable traffic from on-premises to access private link resources without public peering or traversing the internet. VPN tunnels, ExpressRoute private peering, and peered VNets all work with private endpoints.

Similarity to AWS VPC endpoints

If you're an AWS user and all of this sounds familiar to you, you might be thinking of VPC interface endpoints, which are also network interfaces that enable traffic from a subnet in a virtual network to access AWS services or endpoint services (services hosted by other AWS customers) without requiring traffic to go over the internet. Interface endpoints connect you to services powered by AWS PrivateLink and are assigned private IP addresses from the associated subnet, so the traffic source *and* destination are private IPs. Much like the way Azure Private Link works!

How to create a private endpoint for an Azure PaaS resource

Let's return to our VM and storage account example. Say you want traffic to flow from the former to the latter without having to access the storage account's public endpoint. We'll show you an abbreviated version of this excellent Azure tutorial.

Here's how you'd create a private endpoint:

1. Navigate to Private Link Center and select "Create private endpoint."
2. Enter the subscription, resource group, and a name and region for the private endpoint.
3. Select "Connect to an Azure resource in my directory," then select the subscription and "Microsoft.Storage/storageAccounts" as the resource type.
4. Select the name of the target resource, and then specify the target sub-resource. For storage, this can be `blob`, `blob_secondary`, `table`, `table_secondary`, `dfs`, and `dfs_secondary`.
5. Select the VNet and subnet the endpoint should be deployed to.
6. You can then opt to integrate with a private DNS zone, which is recommended if you're using the default Azure-provided DNS since Azure handles all of the hard work for you. If you're using a custom DNS setup, you'll need to use your own DNS servers or create DNS records using the VM's hosts file. (More on this in a bit!)

Now that you've created a private endpoint, you can test it out. SSH or RDP into the instance and run `nslookup mystorageaccount.blob.core.windows.net`, replacing `mystorageaccount` with your storage account name. You'll see something like this:

```
Server:     127.0.0.53
Address:    127.0.0.53#53

Non-authoritative answer:
mystorageaccount.blob.core.windows.net    canonical name = mystorageacc
Name:    mystorageaccount.privatelink.blob.core.windows.net
Address: 10.1.0.5
```

Note how the IP address above is a *private* IP address. If you run the same command from your local terminal, you'll see something like this:

```
Server:     75.75.75.75
Address:    75.75.75.75#53

Non-authoritative answer:
mystorageaccount.blob.core.windows.net    canonical name = mystorageacc
mystorageaccount.privatelink.blob.core.windows.net    canonical name = b
Name:    blob.blz81prdstr02z.store.core.windows.net
Address: 52.238.154.132
```

For bonus points: Block all traffic to the storage account's public endpoint. You can do this by navigating to the storage account and selecting "Firewalls and virtual networks" in the sidebar. Under "Allow access from," select "Selected networks." Then save your changes. Because you haven't whitelisted any networks, no network can access the storage account via its public endpoint.

You can test this out by using the Azure CLI or PowerShell on your VM to upload a file to your storage account. Try to retrieve the file from your local terminal (or from a VM in the same VNet but a different subnet), and you'll get an error message that you can't connect to the storage account. As expected, you can connect to the storage account from the VM via the private endpoint, but you cannot connect to it outside of the subnet.

Extra credit: You can create an NSG to lock down the VNet even further by blocking outbound traffic from the subnet hosting the VM. The VM will *still* be able to access the storage account via the private endpoint, and you can be assured that no other traffic can leave the subnet.

Once again, for the full tutorial, see the Azure documentation.

Things to know before you use private endpoints

- Private endpoints cost money. You pay for private endpoint resource hours as well as inbound and outbound data processed. However, if you're running your own service powered by Private Link, there's no added charge for the private link service -- just the private endpoint.
- It's critical to correctly configure your DNS settings when using private endpoints, particularly when using the fully qualified domain name (FQDN) to connect to the private endpoint resource, since the FQDN of Azure services resolves to their public IP addresses. (For example, our example's FQDN `mystorageaccount.blob.core.windows.net` would resolve to `52.238.154.132`.) If your services are configured to connect to a private endpoint resource using its public endpoint, and you have a custom DNS setup, you'll need to override the DNS resolution to use the private IP address instead. (If you are using the Azure-provided DNS and opted to integrate with a private DNS zone when you created the private endpoint, you're all set -- Azure takes care of the details.)
- Private endpoints must be deployed in the same region as the virtual network, but the private link resource can be in a different region and/or AD tenant.
- Private endpoints don't support network policies such as Network Security Groups (NSGs), so security rules won't apply to them. (This is the reason the extra credit assignment above works!) But don't worry -- other resources in the subnet are still governed by NSG security rules as usual. Be aware that if you're not using the Azure Portal to create the private endpoint, you'll need to manually disable the `PrivateEndpointNetworkPolicies` setting for the subnet. (If you use the portal, this is handled for you automatically.)

Conclusion

As you can see, private endpoints are an excellent way to secure your VNet and private endpoint resources by essentially bringing the private endpoint resource into your VNet. The traffic source is a private IP address and the destination is a private IP address in the same subnet.

"Sounds great," you might say, "but what's the difference between private endpoints and service endpoints?

To find out, stay tuned for part 3 of this blog series, where we compare and contrast private endpoints and service endpoints!

CATEGORIZED UNDER

Security & Compliance | Azure | network security

SEARCH

Search

RELATED POSTS

Fugue Now Supports Microsoft Azure to Provide Multi-Cloud Security Posture Management
July 17, 2019

Cloud Network Security 101: AWS VPC Endpoints
September 12, 2019

Cloud Network Security 101: AWS Security Groups vs NACLs
September 19, 2019

Securing Microsoft Azure Virtual Networks and Network Security Groups
November 1, 2019

Cloud Network Security 101: Azure Virtual Network Service Endpoints
September 17, 2020

Cloud Network Security 101: Azure Service Endpoints vs. Private Endpoints
October 8, 2020

POPULAR CATEGORIES

Security & Compliance | Cloud | AWS | Fugue | cloud security

Featured Posts

Introducing the Engineer's Handbook on Cloud Security
Drew Wright
August 26, 2020

3 Big Amazon S3 Vulnerabilities You May Be Missing
Drew Wright
May 21, 2020

Cloud Security for Newly Distributed Engineering Teams
Drew Wright
March 19, 2020