Implement a hub-spoke network topology with shared services in Azure

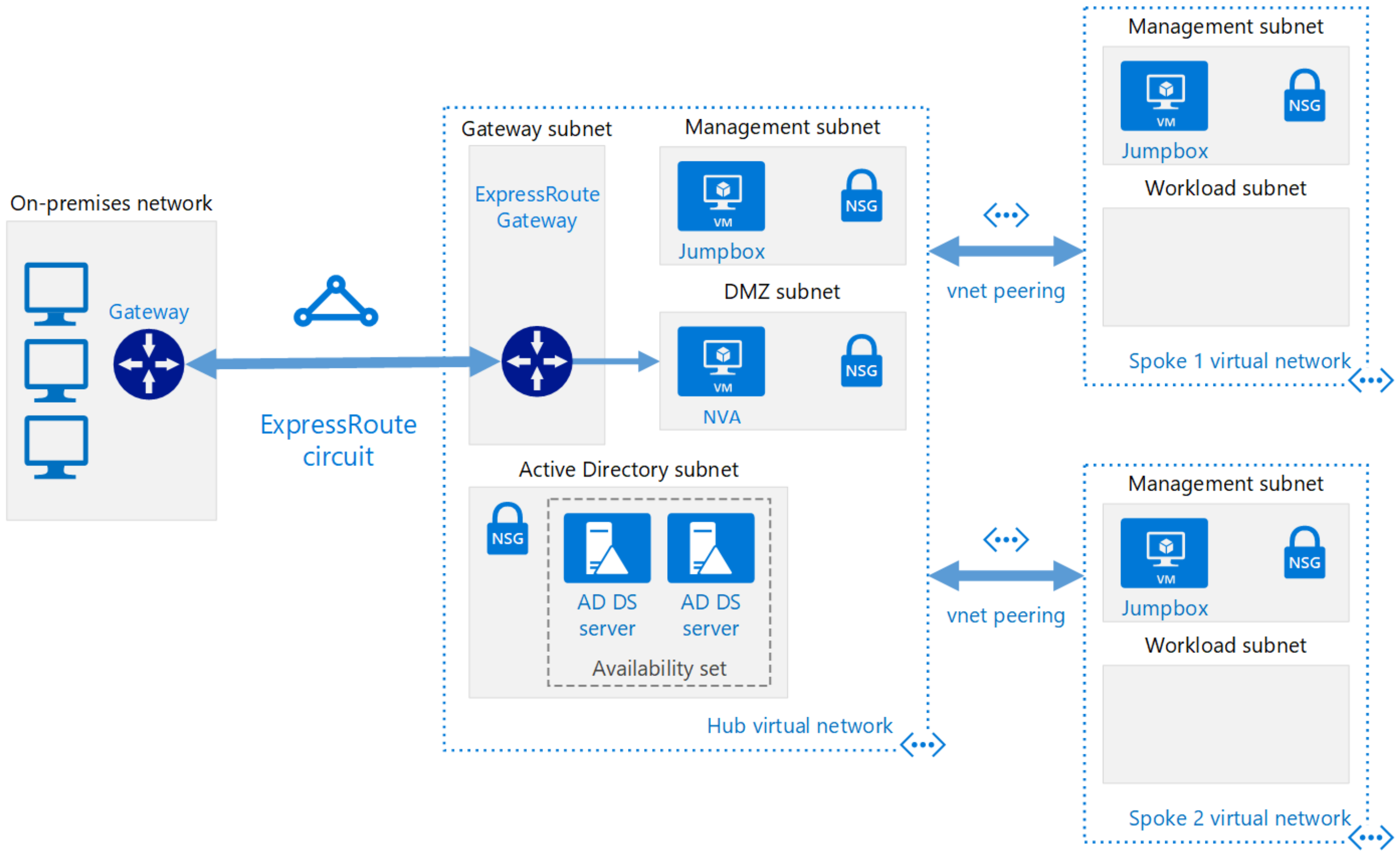
10/09/2018 • 8 minutes to read • Contributors all

In this article

This reference architecture builds on the hub-spoke reference architecture to include shared services in the hub that can be consumed by all spokes. As a first step toward migrating a datacenter to the cloud, and building a virtual datacenter, the first services you need to share are identity and security. This reference architecture shows you how to extend your Active Directory services from your on-premises datacenter to Azure, and how to add a network virtual appliance (NVA) that can act as a firewall, in a hub-spoke topology. **Deploy this solution**.

ⓘ **Note**

This scenario can also be accomplished using **Azure Firewall**, a cloud-based network security service.



Download a Visio file of this architecture

The benefits of this topology include:

- **Cost savings** by centralizing services that can be shared by multiple workloads, such as network virtual appliances (NVAs) and DNS servers, in a single location.
- **Overcome subscriptions limits** by peering VNets from different subscriptions to the central hub.
- **Separation of concerns** between central IT (SecOps, InfraOps) and workloads (DevOps).

Typical uses for this architecture include:

- Workloads deployed in different environments, such as development, testing, and production, that require shared services such as DNS, IDS, NTP, or AD DS. Shared services are placed in the hub VNet, while each environment is deployed to a spoke to maintain isolation.
- Workloads that do not require connectivity to each other, but require access to shared services.
- Enterprises that require central control over security aspects, such as a firewall in the hub as a DMZ, and segregated management for the workloads in each spoke.

Architecture

The architecture consists of the following components.

- **On-premises network**. A private local-area network running within an organization.

- **VPN device**. A device or service that provides external connectivity to the on-premises network. The VPN device may be a hardware device, or a software solution such as the Routing and Remote Access Service (RRAS) in Windows Server 2012. For a list of supported VPN appliances and information on configuring selected VPN appliances for connecting to Azure, see About VPN devices for Site-to-Site VPN Gateway connections.

- **VPN virtual network gateway or ExpressRoute gateway**. The virtual network gateway enables the VNet to connect to the VPN device, or ExpressRoute circuit, used for connectivity with your on-premises network. For more information, see Connect an on-premises network to a Microsoft Azure virtual network.

ⓘ Note

The deployment scripts for this reference architecture use a VPN gateway for connectivity, and a VNet in Azure to simulate your on-premises network.

- **Hub VNet**. Azure VNet used as the hub in the hub-spoke topology. The hub is the central point of connectivity to your on-premises network, and a place to host services that can be consumed by the different workloads hosted in the spoke VNets.

- **Gateway subnet**. The virtual network gateways are held in the same subnet.

- **Shared services subnet**. A subnet in the hub VNet used to host services that can be shared among all spokes, such as DNS or AD DS.

- **DMZ subnet**. A subnet in the hub VNet used to host NVAs that can act as security appliances, such as firewalls.

- **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, managed separately from other spokes. Each workload might include multiple tiers, with multiple subnets connected through Azure load balancers. For more information about the application infrastructure, see Running Windows VM workloads and Running Linux VM workloads.

- **VNet peering**. Two VNets can be connected using a peering connection. Peering connections are non-transitive, low latency connections between VNets. Once peered, the VNets exchange traffic by using the Azure backbone, without the need for a router. In a hub-spoke network topology, you use VNet peering to connect the hub to each spoke. You can peer virtual networks in the same region, or different regions (Global VNet Peering). For more information, see Requirements and constraints.

ⓘ Note

This article only covers Resource Manager deployments, but you can also connect a classic VNet to a Resource Manager VNet in the same subscription. That way, your spokes can host classic deployments and still benefit from services shared in the hub.

Recommendations

All the recommendations for the hub-spoke reference architecture also apply to the shared services reference architecture.

Also, the following recommendations apply for most scenarios under shared services. Follow these recommendations unless you have a specific requirement that overrides them.

Identity

Most enterprise organizations have an Active Directory Directory Services (ADDS) environment in their on-premises datacenter. To facilitate management of assets moved to Azure from your on-premises network that depend on ADDS, it is recommended to host ADDS domain controllers in Azure.

If you use Group Policy Objects, that you want to control separately for Azure and your on-premises environment, use a different AD site for each Azure region. Place your domain controllers in a central VNet (hub) that dependent workloads can access.

Security

As you move workloads from your on-premises environment to Azure, some of these workloads will require to be hosted in VMs. For compliance reasons, you may need to enforce restrictions on traffic traversing those workloads.

You can use network virtual appliances (NVAs) in Azure to host different types of security and performance services. If you are familiar with a given set of appliances on-premises today, it is recommended to use the same virtualized appliances in Azure, where applicable.

Note

The deployment scripts for this reference architecture use an Ubuntu VM with IP forwarding enabled to mimic a network virtual appliance.

Deploy the solution

A deployment for this architecture is available on GitHub. The deployment creates the following resource groups in your subscription:

- hub-adds-rg
- hub-nva-rg
- hub-vnet-rg
- onprem-vnet-rg
- spoke1-vnet-rg
- spoke2-vnet-rg

The template parameter files refer to these names, so if you change them, update the parameter files to match.

Prerequisites

1. Clone, fork, or download the zip file for the reference architectures GitHub repository.

2. Install Azure CLI 2.0.

3. Install Node and NPM

4. Install the Azure building blocks npm package.

```bash
npm install -g @mspnp/azure-building-blocks
```

5. From a command prompt, bash prompt, or PowerShell prompt, sign into your Azure account as follows:

```bash
az login
```

Deploy the simulated on-premises datacenter using azbb

This step deploys the simulated on-premises datacenter as an Azure VNet.

1. Navigate to the `hybrid-networking\shared-services-stack\` folder of the GitHub repository.

2. Open the `onprem.json` file.

3. Search for all instances of `UserName`, `adminUserName`,`Password`, and `adminPassword`. Enter values for the user name and password in the parameters and save the file.

4. Run the following command:

```bash
azbb -s <subscription_id> -g onprem-vnet-rg -l <location> -p onprem.json --deploy
```

5. Wait for the deployment to finish. This deployment creates a virtual network, a virtual machine running Windows, and a VPN gateway. The VPN gateway creation can take more than 40 minutes to complete.

Deploy the hub VNet

This step deploys the hub VNet and connects it to the simulated on-premises VNet.

1. Open the `hub-vnet.json` file.

2. Search for `adminPassword` and enter a user name and password in the parameters.

3. Search for all instances of `sharedKey` and enter a value for a shared key. Save the file.

```JSON
"sharedKey": "abc123",
```

4. Run the following command:

```bash
azbb -s <subscription_id> -g hub-vnet-rg -l <location> -p hub-vnet.json --deploy
```

5. Wait for the deployment to finish. This deployment creates a virtual network, a virtual machine, a VPN gateway, and a connection to the gateway created in the previous section. The VPN gateway can take more than 40 minutes to complete.

Deploy AD DS in Azure

This step deploys AD DS domain controllers in Azure.

1. Open the `hub-adds.json` file.

2. Search for all instances of `Password` and `adminPassword`. Enter values for the user name and password in the parameters and save the file.

3. Run the following command:

   bash Copy

   ```bash
   azbb -s <subscription_id> -g hub-adds-rg -l <location> -p hub-adds.json --deploy
   ```

This deployment step may take several minutes, because it joins the two VMs to the domain hosted in the simulated on-premises datacenter, and installs AD DS on them.

Deploy the spoke VNets

This step deploys the spoke VNets.

1. Open the `spoke1.json` file.

2. Search for `adminPassword` and enter a user name and password in the parameters.

3. Run the following command:

   bash Copy

   ```bash
   azbb -s <subscription_id> -g spoke1-vnet-rg -l <location> -p spoke1.json --deploy
   ```

4. Repeat steps 1 and 2 for the file `spoke2.json`.

5. Run the following command:

   bash Copy

   ```bash
   azbb -s <subscription_id> -g spoke2-vnet-rg -l <location> -p spoke2.json --deploy
   ```

Peer the hub VNet to the spoke VNets

To create a peering connection from the hub VNet to the spoke VNets, run the following command:

bash Copy

```bash
azbb -s <subscription_id> -g hub-vnet-rg -l <location> -p hub-vnet-peering.json --deploy
```

Deploy the NVA

This step deploys an NVA in the `dmz` subnet.

1. Open the `hub-nva.json` file.

2. Search for `adminPassword` and enter a user name and password in the parameters.

3. Run the following command:

bash Copy

```bash
azbb -s <subscription_id> -g hub-nva-rg -l <location> -p hub-nva.json --deploy
```

Test connectivity

Test connectivity from the simulated on-premises environment to the hub VNet.

1. Use the Azure portal to find the VM named `jb-vm1` in the `onprem-jb-rg` resource group.

2. Click `Connect` to open a remote desktop session to the VM. Use the password that you specified in the `onprem.json` parameter file.

3. Open a PowerShell console in the VM, and use the `Test-NetConnection` cmdlet to verify that you can connect to the jumpbox VM in the hub VNet.

   PowerShell Copy

   ```powershell
   Test-NetConnection 10.0.0.68 -CommonTCPPort RDP
   ```

The output should look similar to the following:

PowerShell Copy

```powershell
ComputerName     : 10.0.0.68
RemoteAddress    : 10.0.0.68
RemotePort       : 3389
InterfaceAlias   : Ethernet 2
SourceAddress    : 192.168.1.000
TcpTestSucceeded : True
```

Note

By default, Windows Server VMs do not allow ICMP responses in Azure. If you want to use `ping` to test connectivity, you need to enable ICMP traffic in the Windows Advanced Firewall for each VM.

Repeat the sames steps to test connectivity to the spoke VNets:

PowerShell Copy

```powershell
Test-NetConnection 10.1.0.68 -CommonTCPPort RDP
Test-NetConnection 10.2.0.68 -CommonTCPPort RDP
```