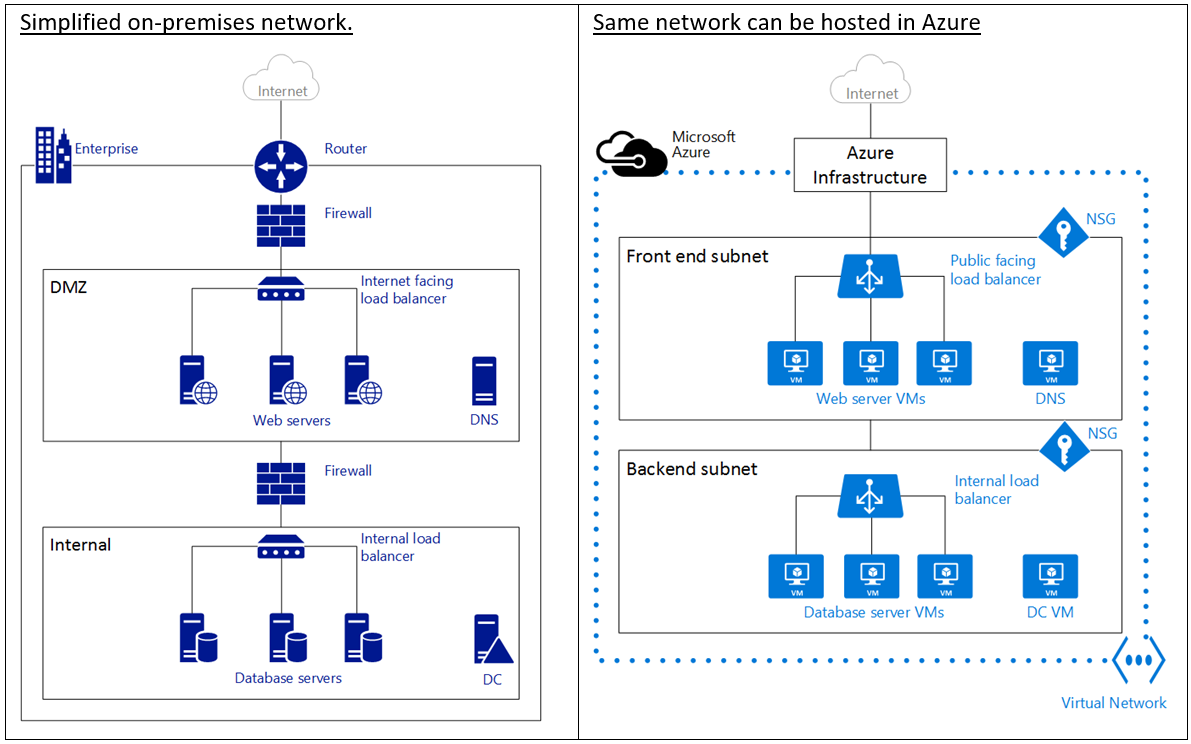
**Configure and Manage Azure Virtual Networks**

* Overview of Azure Networking
* Virtual Network Benefits
* Understanding Network Resources
* Implement and manage virtual networking
* Network Security Group
* Application Security Groups

**Overview of Azure Networking**

* An Azure virtual network (VNet) is a representation of your own network in the cloud.
* It is **a logical isolation** of the Azure cloud dedicated to your subscription. You can fully control the IP address blocks, DNS settings, security policies, and route tables within this network.
* You can also further segment your VNet into **subnets** and launch Azure virtual machines (VMs).
* You can connect the virtual network to your on-premises network using one of the connectivity options available in Azure. In essence, you can expand your network to Azure, with complete control on IP address blocks with the benefit of enterprise scale Azure provides.



\*In computer **networks**, a **DMZ** (**demilitarized zone**) is a physical or logical **sub-network** that separates an internal local area **network** (LAN) from other untrusted **networks**, usually the Internet.

**Virtual Network Characteristics**

* **Isolation**. VNets are completely isolated from one another. That allows you to create disjoint networks for **development, testing, and production** that use the same CIDR address blocks.
* **Connectivity**. VNets can be connected to each other, and even to your on-premises datacenter, by using a site-to-site VPN connection, or ExpressRoute connection.
* **Access to the public Internet**. All VMs in a VNet can access the public Internet by default. You can control access by using Network Security Groups (NSGs).
* **Security**. Traffic entering and exiting the virtual machines in a VNet can be controlled using Network Security groups and Azure Firewall.
* **Access to VMs within the VNet**. VMs can be launched in the same virtual network and they can connect to each other using private IP addresses even if they are in different subnets without the need to configure a gateway or use public IP addresses.
* **Name resolution**. Azure provides internal name resolution for IaaS VMs deployed in your VNet. You can also deploy your own DNS servers and configure the VNet to use them.

**Subnets**:

Subnet is a **range of IP addresses** in the VNet, you can divide a VNet into multiple subnets for organization and security. VMs deployed to subnets (same or different) within a VNet can communicate with each other without any extra configuration. You can also configure **route tables and NSGs** to a subnet.

**Understanding IP Address Space**

* **IP addresses**: There are two types of IP addresses assigned to resources in Azure: public and private.
  1. **Public IP Addresses** allow Azure resources to communicate with Internet and other Azure public-facing services like Azure Redis Cache.
  2. **Private IP Addresses** allows communication between resources in a virtual network, along with those connected through a VPN, without using an Internet-routable IP addresses.

**Preferred IP Series for Intranets (Private IP):**

Small Network1: 192.168.0.X – for 28 Systems – IP Address Range = 192.168.0.0/24 (Only last byte changes)

Small Network2: 192.168.1.X –for 28 Systems – IP Address Range = 192.168.1.0/24 (Only last byte changes)

Large Network: 172.16.X.X – for 216 Systems - IP Address Range = 172.16.0.0/16 (last 2 bytes change)

Very Large Network: 10.X.X.X – for 224 Systems – IP Address Range = 10.0.0.0/8 (last 3 bytes change)

**Classless Inter-Domain Routing** (**CIDR) notation** is a compact representation of an IP address and its associated routing prefix. The **notation** is constructed from an IP address, a slash ('/') character, and a decimal number. The number is the count of leading 1 bits in the routing mask, traditionally called the network mask.

202.123.56.123 to 202.123.56.123 = 202.123.56.123/32

0.0.0.0 to 255.255.255.255 = 0.0.0.0/0

192.168.0.0 to 192.168.0.255 = 192.168.0.0/24 = 256

192.168.170.0 to 192.168.170.255 = 192.168.170.0/24 = 256

172.16.0.0 to 172.16.255.255 = 172.16.0.0/16 = 256 \* 256

202.34.0.0/16 = 202.34.0.0 to 202.34.255.255 = 256 \* 256

10.0.0.0/16 = 10.0.0.0 to 10.255.255.255 = 256\*256\*256

10.0.0.0/28

10.0.0.4

10.0.0.5

10.0.0.6

10.0.1.5

10.0.1.0/24

10.0.1.4

10.0.1.5

10.0.1.6

10.0.2.0/24

10.0.3.0/24

10.0.4.0/24

10.1.0.0/16

10.1.0.0/24

10.1.0.4

10.1.0.5

10.1.0.6

10.1.1.5

10.1.1.0/24

10.1.1.4

10.1.1.5

10.1.1.6

10.1.2.0/24

10.1.3.0/24

10.1.4.0/24

10.2.0.0/16

**Public IP Addresses**

* There are two methods in which an IP address is allocated to a public IP resource - **dynamic** or **static.**
* In the **dynamic** allocation method the IP address is **not** allocated at the time of its creation. Instead, the public IP address is allocated when you start (or create) the associated resource (like a VM or load balancer). The IP address is released when you stop (or delete) the resource. This means the IP address can change.
* In the **static** allocation method the IP address for the associated resource does not change. In this case an IP address is assigned immediately. It is released only when you delete the resource or change its allocation method to dynamic.
* Public IP addresses allow Azure resources to communicate with Internet and Azure public-facing services such as Azure Redis Cache, Azure Event Hubs, SQL databases and Azure storage.
* In Azure Resource Manager, a public IP address is a resource that has its own properties. You can associate a public IP address resource with any of the following resources:
  + Internet-facing Virtual machines (VM)
  + Internet-facing load balancers
  + VPN gateways
  + Application gateways
* Public IP address is paid service.

**Private IP Addresses**

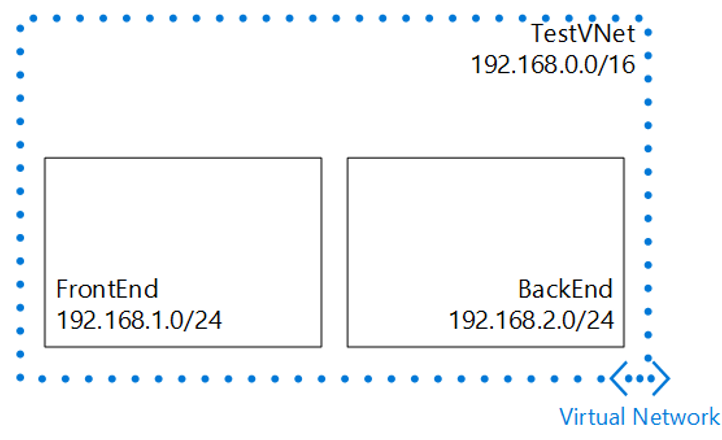
1. IP address is allocated from the address range of the subnet to which the resource is attached.
2. The default allocation method is dynamic, where the IP address is automatically allocated from the resource's subnet (using DHCP). This IP address can change when you stop and start the resource.
3. You can set the allocation method to static to ensure the IP address remains the same. In this case, you also need to provide a valid IP address that is part of the resource's subnet.
4. Private IP addresses allow Azure resources to communicate with other resources in a virtual network or an on-premises network through a VPN gateway or ExpressRoute circuit, without using an Internet-reachable IP address.
5. In the Azure Resource Manager deployment model, a private IP address is associated to the following types of Azure resources:
   * VMs
   * Internal load balancers (ILBs)
   * Application gateways

**Create a Virtual Network (VNet) using the Azure portal**

In this scenario we will create a VNet named **TestVNet** with a reserved CIDR block of **192.168.0.0/16**.

Your VNet will contain the following **subnets**:

* **FrontEnd**, using **192.168.1.0/24** as its CIDR block.
* **BackEnd**, using **192.168.2.0/24** as its CIDR block.



1. Click **Search 🡪 Virtual networks 🡪 +Create**

A screenshot of a computer

Description automatically generated

1. Select your Subscription, Resource Group 🡪 Create new (Demo-eastus-rg), Virtual network name=Demo-eastus-vnet, Region = East US

A screenshot of a computer

Description automatically generated

1. Go to IP address tab 🡪 Change CIDR block to 192.168.0.0/16, delete existing Subnet and click + Add a subnet

A screenshot of a computer

Description automatically generated

1. IP Address Space=192.168.0.0/16, Subnet name="**Frontend**-**subnet**", Subnet Address Range=192.168.1.0/24, click Add button

A screenshot of a computer

Description automatically generated

1. Click Review + create button, then Create button, wait for the VNet to be created. Click Go to resource button

A screenshot of a computer

Description automatically generated

1. In the **Virtual network** blade, Settings section, click **Subnets** 🡪 + Subnet

A screenshot of a computer

Description automatically generated

1. Name=**Backend-subnet**, Subnet address range=192.168.2.0/24, Leave NAT Gateway, Network security group and Route table as None 🡪 Save

A screenshot of a computer

Description automatically generated

1. View VNET Diagram

**Network Security Group**

NSGs are simple, stateful packet inspection devices that use the 5-tuple (the source IP, source port, destination IP, destination port, and layer 4 protocol) approach to create **allow/deny rules** for network traffic. You allow or deny traffic to and from a single IP address, to and from multiple IP addresses, or to and from entire subnets.

In this scenario you will create an NSG for each subnet in the **Demo-vnet** virtual network, as described below:

|  |
| --- |
| * **Frontend-nsg**. The front end NSG will be applied to the *FrontEnd* subnet, and contain two rules:   + **rdp-allow**. This rule will allow RDP (3389) traffic to the *FrontEnd* subnet.   + **web-allow**. This rule will allow HTTP (80) traffic to the *FrontEnd* subnet. * **Backend-nsg**. The back end NSG will be applied to the *BackEnd* subnet, and contain two rules:   + **sql-allow**. This rule allows SQL (1433) traffic only from the *FrontEnd* subnet.   + **rdp-allow**: This rule will allow RDP (3389) traffic to the *BackEnd* subnet   + **web-deny**. This is Outbound rule **denies all internet bound** traffic **from** the *BackEnd* subnet.   A screenshot of a computer  Description automatically generated |

**Create NSG for Frontend-subnet:**

1. Search 🡪 Network Security Groups 🡪 + Create

A screenshot of a computer

Description automatically generated

1. Name=**Frontend-nsg** 🡪 Review + Create button

Go to resource after creation

A screenshot of a computer

Description automatically generated

**Create HTTP and RDP Rules**

1. Select Frontend-nsg 🡪 Settings 🡪
   1. **Inbound security rules 🡪 Add** 
      1. **Name=AllowHTTP, Source=Any, Source port range=\*, Protocol=TCP, Destination=Any, Destination port range=80, Action=Allow, Priority=1000 🡪 Add**

A screenshot of a computer

Description automatically generated

* 1. **Inbound security rules 🡪 Add,** 
     1. **Name=AllowRDP,**
     2. **Source=Any,**
     3. **Source port range=\*,**
     4. **Protocol=TCP,**
     5. **Destination=Any,**
     6. **Destination port range=3389,**
     7. **Action=Allow**
     8. **Priority=1002,**
     9. **Add**

**Associate the NSG to the FrontEnd subnet**

1. Select Demo-east-vnet 🡪 Subnets 🡪 **Frontend-subnet** 🡪 Network security group 🡪 Select Frontend-nsg 🡪 Save

A screenshot of a computer

Description automatically generated

1. **Create NSG for Backend:** Browse 🡪 Network Security Groups 🡪 Add 🡪 Name=Backend-nsg 🡪 Create

**Configuring rules for Backend-subnet**

1. Select Backend-nsg 🡪
   1. **Inbound security rules 🡪 Add,** 
      1. **Name=AllowSQL**
      2. **Source=IP Addresses**
      3. **Source IP address range=192.168.1.0/24**
      4. **Source port range=\*,**
      5. **Destination=Any,**
      6. **Destination port range=1433,**
      7. **Protocol=TCP,**
      8. **Action=Allow**
      9. **Priority=1001**
      10. **Add**
   2. **Inbound security rules 🡪 Add,** 
      1. **Name=AllowRDP,**
      2. **Source=Any,**
      3. **Source port range=\*,**
      4. **Protocol=TCP,**
      5. **Destination=Any,**
      6. **Destination port range=3389,**
      7. **Action=Allow**
      8. **Priority=1002,**
      9. **Add**
   3. **Outbound security rules 🡪 Add, Name=DenyWeb, priority, Priority=1000, Destination=Tag, destination Tag=Internet, Destination port range=80, Source=Any, Protocol=Any, Source port range=\*, Action=Deny 🡪 OK**

A screenshot of a computer

Description automatically generated

**Associate the NSG to the BackEnd subnet**

1. Select Demo-eastus-vnet 🡪 Subnets 🡪 Backend-subnet 🡪 Network security group 🡪 Select Backend-nsg 🡪 Save

**Summary:**

Virtual Network (192.168.0.0/16)

Frontend-subnet (192.168.1.0/24)

Frontend-nsg

Allow RDP / HTTP (Inbound)

Backend-subnet (192.168.2.0/24)

Backend-nsg

Allow RDP / SQL (Inbound)

Deny Internet (Outbound)

**Creating a Virtual Machine**

1. Azure portal 🡪 Search 🡪 Compute 🡪 Virtual machines
2. Click on + Create 🡪 Azure virtual machine

**Basics tab 🡪**

* Select same Subscription, Resource group, Region as that of Virtual network created above.
* Virtual machine name = Web1-vm,
* Image = "Windows Server 2019 Datacenter x64 Gen 2"
* Username = dssadmin,
* Password = Password@123,
* Public Inbound Port = Allow Selected ports (3389).

A screenshot of a computer

Description automatically generated

**Disks tab 🡪 All Defaults**

**Networking tab 🡪**

Virtual Network = Select the earlier created VNET,

Subnet=Frontend-subnet

NSG = None

A screenshot of a computer

Description automatically generated

Leave the other tabs as is.

Click **Review + Create** 🡪 Review your configuration and Click **Create**

1. Create another VM (All steps same as Web1-vm Except)
   1. Name=Database1-vm
   2. Subnet = Backend-subnet
2. Connect to Web1-vm🡪 Server Manager 🡪 Local Server 🡪 Dashboard 🡪 Add roles and features 🡪 Next 🡪 Next 🡪 Next 🡪Select **Web server** 🡪 Next 🡪 Next 🡪 Install.
3. Azure Portal 🡪 Find the IP Address of VM and visit in browser: http://<IP-of-VM>
4. RDP to Database1-vm 🡪 Open Browser 🡪 Visit: <https://www.google.com>.
   1. Note that you will get error page.
5. From Frontend-nsg 🡪 Delete HTTP Rule
6. Azure Portal 🡪 Find the IP Address of VM and visit in browser: http://<IP-of-VM> (Refresh page using Ctrl+F5).
   1. Note that we will get error

**Summary:**

Demo-VNet

Frontend-subnet

Frontend-nsg

Allowed HTTP and RDP

Demo-vm

NO NSG (NIC Level)

Remote Login and installed IIS

edit wwwroot\iisstart.png - Added ONE

Web1-vm-ip

DNS Name

Backend-sub

Backend-nsg

Allowed RDP Inbound

Denied Internet: OutBound

Database-vm

NO NSG (NIC Level)

Accessed

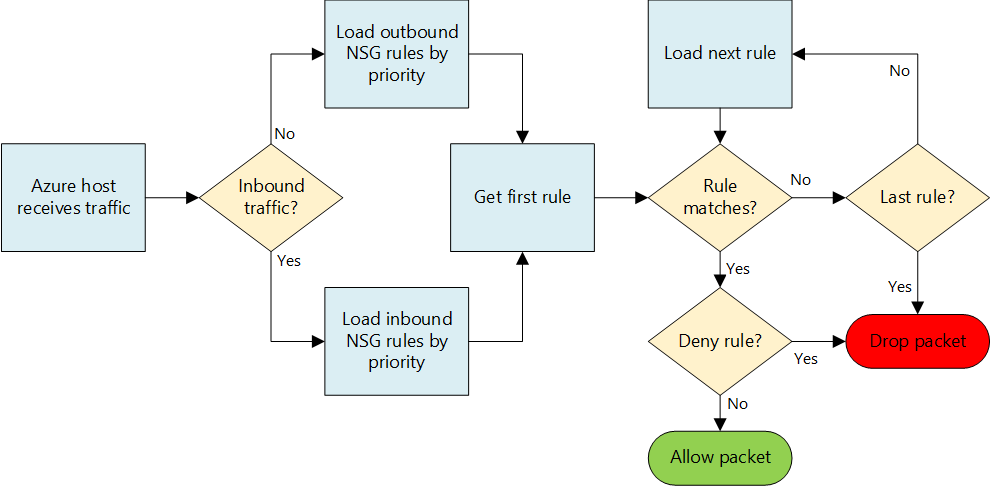
Web1 http://<ip> or http://<dnsname>

**NSG: Evaluate effective security rules**

Be very careful when you want to apply NSG to both VM (NIC) and subnet level at the same time. NSGs are evaluated independently, and an “allow” rule must exist at **both levels** otherwise traffic will not be admitted.

|  |  |
| --- | --- |
|  | If there was incoming traffic on port 80, you would need to have the NSG at subnet level ALLOW port 80, and you would also need another NSG with ALLOW rule on port 80 at the NIC level. For incoming traffic, the NSG set at the subnet level is evaluated first, then the NSG set at the NIC level is evaluated. For outgoing traffic, it is the converse. |

The picture below should even clarify this concept more: you can see how rules are evaluated for network packets, once again remember that you need to **evaluate this diagram two times**: once for subnet level NSG rules, and once for NIC level NSG rules.



**To see the Effective Rules:**

Select the VM 🡪 Settings 🡪 Networking 🡪 Click on **Effective security rules**

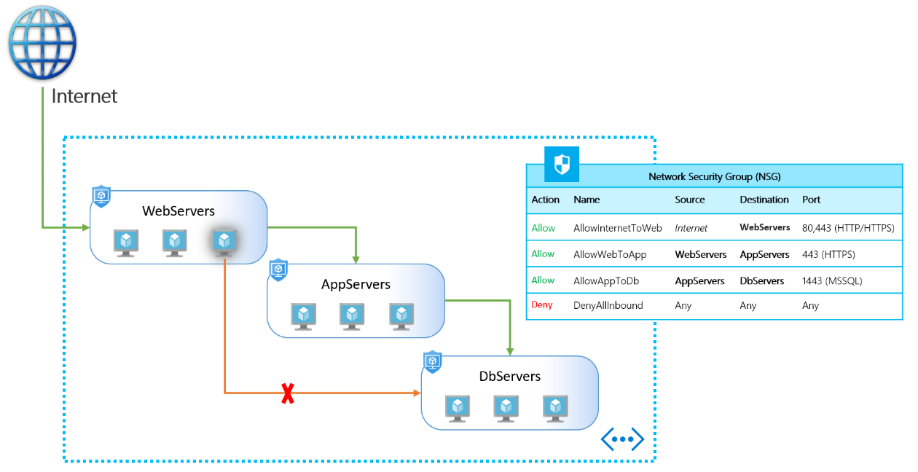
Now you get an overview which NSGs are associated with the VM’s NIC and which rules are applied to it.

For an offline analysis there is a download option, that generates a CSV file of the output.

**Application Security Groups**

Application security groups enable you to configure network security as a natural extension of an application's structure, allowing you to group virtual machines and define network security policies based on those groups.

This feature allows you to reuse your security policy at scale without manual maintenance of explicit IP addresses. The platform handles the complexity of explicit IP addresses and multiple rule sets, allowing you to focus on your business logic.



**By Default Web1-vm will be able to connect to Database1-vm**

1. Use **Network Watcher** and note that **IP Flow verify** succeeds from Web1-vm to Database1-vm

Search Network Watcher 🡪 IP flow verify

* 1. Virtual Machine = Web1-vm
  2. Local Port = 1234
  3. Remote IP Address = 192.168.1.4
  4. Remote Port = 1234
  5. Verify IP flow

Create two new Application Secuirty Groups

* + 1. WebServers-asg
    2. DbServers-asg

1. Search 🡪 Application Security Groups 🡪 + Create

Name=**WebServers-asg**

Region = East US

Review + Create

A screenshot of a computer

Description automatically generated

1. Similarly create another Application security group = **DatabaseServers-asg**

**Attach them to respective VM:**

1. Web1-vm 🡪 Networking 🡪 **Application Securty Group** tab 🡪 click on Configure the application security groups 🡪 Select WebServers-asp 🡪 Save

A screenshot of a computer

Description automatically generated

1. Database1-vm 🡪 Networking 🡪 **Application Securty Group** tab 🡪 click on Configure the application security groups 🡪 Select DatabaseServers-asg 🡪 Save

Create NSG rule to block traffic from all VMs with ASG=WebServers-asg to all VMs with ASG=DatabaseServers-asg

1. Select **Frontend-nsg 🡪 Outbound security rules** 🡪 Add
2. Select values as in image below:

A screenshot of a computer

Description automatically generated

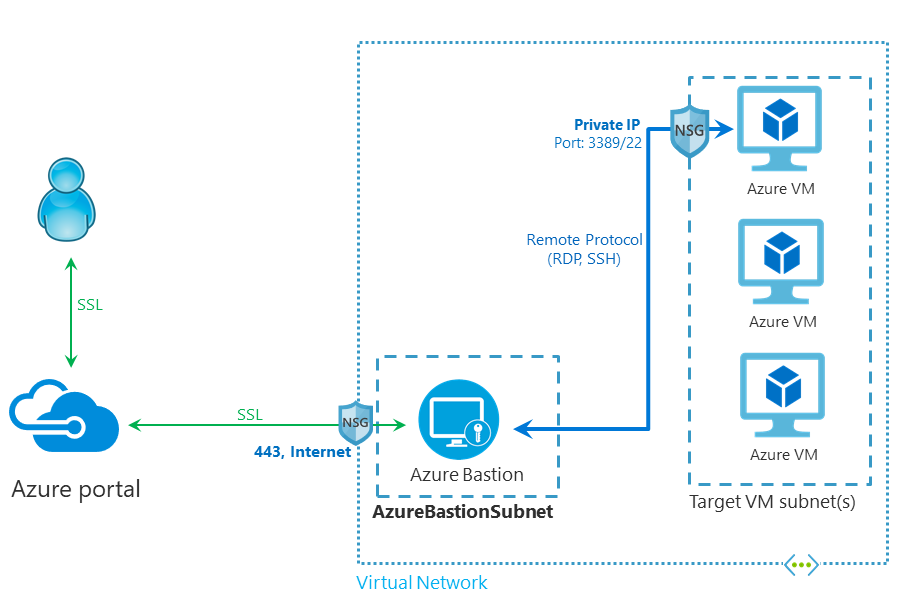
1. Wait for couple of minutes.
2. Use **Network Watcher** and note that **IP Flow verify** is **failed** from Web1-vm to Database1-vm
3. Search Network Watcher 🡪 IP flow verify
   1. Virtual Machine = Web1-vm
   2. Local Port = 1234
   3. Remote IP Address = 192.168.2.4
   4. Remote Port = 1234
   5. Verify IP flow

A screenshot of a computer

Description automatically generated

**Azure Bastion**

Azure Bastion is a fully managed PaaS service that provides secure and seamless RDP and SSH access to your virtual machines directly through the Azure Portal. Azure Bastion is provisioned directly in your Virtual Network (VNet) and supports all VMs in your Virtual Network (VNet) using SSL without any exposure through public IP addresses.



**In this diagram:**

* The Bastion host is deployed in the virtual network and in subnet **AzureBastionSubnet**.
* The user connects to the Azure portal using any **HTML5** browser.
* The user selects the virtual machine to **connect** to.
* With a single click, the **RDP/SSH session** opens in the browser.
* No public IP is required on the Azure VM.

**The following features are available:**

* **RDP and SSH directly in Azure portal:** You can directly get to the RDP and SSH session directly in the Azure portal using a single click seamless experience.
* **Remote Session over SSL and firewall traversal for RDP/SSH:** Azure Bastion uses an HTML5 based web client that is automatically streamed to your local device, so that you get your RDP/SSH session over SSL on port 443 enabling you to traverse corporate firewalls securely.
* **No Public IP required on the Azure VM:** Azure Bastion opens the RDP/SSH connection to your Azure virtual machine using private IP on your VM. You don't need a public IP on your virtual machine.
* **No hassle of managing NSGs:** Azure Bastion is a fully managed platform PaaS service from Azure that is hardened internally to provide you secure RDP/SSH connectivity. You don't need to apply any NSGs on Azure Bastion subnet. Because Azure Bastion connects to your virtual machines over private IP, you can configure your NSGs to allow RDP/SSH from Azure Bastion only. This removes the hassle of managing NSGs each time you need to securely connect to your virtual machines.
* Azure Bastion can support up to **25 concurrent RDP**, this is still dependent on the Azure Virtual Machines. Azure Virtual Machine doesn’t support more than 2 concurrent RDP connections and these must be from two different user accounts.
* Each instance can support 20 concurrent RDP connections and 40 concurrent SSH connections for medium workloads.

**You can use the Shortcut:**

Azure Virtual Network 🡪 Demo-eastus-vnet 🡪 Bastion 🡪 Deploy Bastion

A screenshot of a computer

Description automatically generated

OR

**Create a bastion host manually**

1. Select Demo-eastus-vnet 🡪 Subnets 🡪 + Subnet
   1. Name **AzureBastionSubnet** (You must use a subnet of at least /26 or larger eg: /26, /25 and …)
   2. Subnet address range = 192.168.3.0/24
2. Search Bastions 🡪 + Create
   1. Name = **Demo-eastus-bastion**
   2. Region = eastus
   3. Tier = Basic
   4. Virtual network = Demo-eastus-vnet
   5. Subnet = AzureBastionSubnet
   6. Public IP address = Create new
   7. Public IP address name = Demo-eastus-bastion-ip

A screenshot of a computer

Description automatically generated

OR

**Connect to VM**

1. Azure VM 🡪 **Connect** 🡪 **Bastion** tab 🡪 Use Bastion 🡪 Provide the RDP Username and Password 🡪 Connect

**Azure DNS**

* The Domain Name Service, or DNS, is responsible for translating (or resolving) a website or service name to its IP address.
* Azure DNS is a hosting service for DNS domains, providing name resolution using Microsoft Azure infrastructure.
* Applications requiring automatic DNS management can integrate with the service via the REST API and SDKs.
* When you add a new DNS record, the Azure DNS name servers are updated in a few seconds so you don’t have to wait long before that DNS record can be used.
* Azure DNS does not currently support purchasing of domain names.

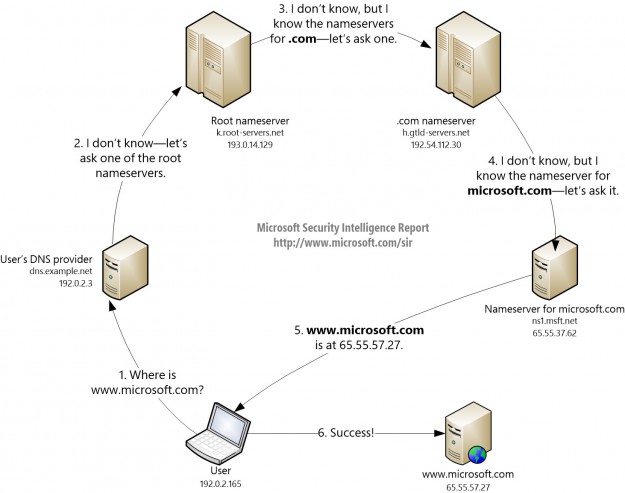
**DNS Domains:**

The DNS is a hierarchy of domains. The hierarchy starts from the ‘root’ domain, whose name is simply '.'. Below this come top-level domains, such as ‘com’, 'net', ‘org’, 'uk' or ‘jp’. Below these are second-level domains, such as 'org.uk' or ‘co.jp’. The domains in the DNS hierarchy are globally distributed, hosted by DNS name servers around the world.

A diagram of a company

Description automatically generated

**DNS Resolution:** To answer queries, it uses aspecial type of DNS record called a Name Server (NS)record. For example, the root zone contains NS records for ‘com’and shows the name servers for the ‘com’ zone. In turn,the ‘com’ zone contains NS records for ‘contoso.com’, whichshows the name servers for the ‘contoso.com’ zone. Settingup the NS records is called delegating the domain.



**How DNS Server Works**   
In browser http://www.bestazuretraining.com

* 1. Browser will send request to DNS Server as configured in your machine for finding IP of [www.bestazuretraining.com](http://www.bestazuretraining.com)
  2. DNS, if has IP - It immediately returns
  3. DNS does'nt have IP - It will send the request to ROOT Name Server
  4. Root Name Server will query -> .com Name Server
  5. .com Name Server will sent the request bestazuretraining.com name server
  6. In bestazuretraining.com Name Server it will search for the required Recordset and return the value...
  7. If IP is returned browser will directly send the request to target machine...
  8. If Alias (CName) is returned then it again starts from Step 2...

**DNS Zone:**

A DNS zone is used to host the DNS records for a particular domain (purchased from registrar like godaddy.com). In order to start hosting your domain, you need to create a DNS zone. Any DNS record created for a particular domain will be inside a DNS zone for the domain.

For example, the domain "contoso.com" may contain a number of DNS records, such as "mail.contoso.com" (for a mail server) and "www.contoso.com" (for a web site).

**Steps to Create a DNS Zone and Map Name to IP Address:**

1. **Buy a Domain Name from a Registrar (eg: godaddy.com is registrar)**
2. **Azure Portal 🡪 New 🡪 Networking 🡪 DNS zone**
3. **Name = deccansoft.net, Provide other details 🡪 Create**
4. **Goto Registrar Website 🡪 Login**
5. **DNS Delegation: Map domain Name Server to NS records for the DNS Zone created**

A screenshot of a computer

Description automatically generated

1. **Select the DNS Zone 🡪 + Record set 🡪 Name=www, Type=”A”, TTL=1, IP Address=<Public IP of VM Created> 🡪 OK**

**Private DNS Zones**

* A Private DNS Zone can be connected to multiple Virtual Networks (in any ADTenant/Subscriptions/Region).
* Private DNS Zone is a Global Services (Independent of Region)
* Here Domain Names are mapped **Private** IP Address.

Web1-vm = 192.168.1.4

Database1-vm = 192.168.2.4

**Create Private DNS Zone**

1. Search 🡪 Private DNS zones 🡪 + Create
   * Resource group = Select **Demo-eastus-rg**
   * Name = **mydomain.com**
   * **Review + Create 🡪 Create**

A screenshot of a computer

Description automatically generated

1. Select mydomain.com 🡪 Virtual network links 🡪 + Add
   * Link name = Demo-eastus-vnet
   * Virtual network = Select Demo-eastus-vnet
   * Check Enable auto registration
   * OK

A screenshot of a computer

Description automatically generated

1. Select mydomain.com 🡪 Overview 🡪 +Record set
   * Name = web1
   * IP address = 192.168.1.4
   * OK
2. Select mydomain.com 🡪 Overview 🡪 +Record set
   * Name = database1
   * IP address = 192.168.2.4
   * OK
3. RDP to Database1-vm
   * Open Browser 🡪 http://web1.mydomain.com

**Create connectivity between virtual networks**

There are multiple ways to connect VNets. The sections below describe different ways to connect virtual networks.

A diagram of a computer network

Description automatically generated

**Cloud-Only Virtual Networks:**

You can choose not to make any kind of virtual private network (VPN) connection to a VNet. Instead, when you create a VM or cloud service, you can specify endpoints that external clients can connect to. An endpoint is a VIP and a port number. Therefore an endpoint can be used only for a specific protocol, such as connecting a Remote Desktop Protocol (RDP) client or browsing a website. These VNets are known as cloud-only virtual networks. A dynamic routing gateway is not required in the VNet. Endpoints are published to the Internet, so they can be used by anyone with an Internet connection, including your on-premises computers.

**Point-to-Site VPNs**

A simple way to connect a VPN to an Azure VNet is to use a Point-to-Site VPN. In these VPNs, you configure the connection on individual on-premises computers. No extra hardware is required but you must complete the configuration procedure on every computer that you want to connect to the VNet. Point-to-site VPNs can be used by the client computer to connect to a VNet from any location with an Internet connection. Once the VPN is connected, the client computer can access all VMs and cloud services in the VNet as if they were running on the local network.

**VNet-to-VNet**

Connecting a virtual network to another virtual network (VNet-to-VNet) is similar to connecting a virtual network to an on-premises site location. Both connectivity types use a VPN gateway to provide a secure tunnel using IPsec/IKE. The VNets you connect can be in **different subscriptions** and **different regions**.

The difference between the S2S AND V2V connection types is the way the local network gateway is configured. When you create a VNet-to-VNet connection, you do not see the local network gateway address space. It is automatically created and populated. If you update the address space for one VNet, the other VNet automatically knows to route to the updated address space. Creating a VNet-to-VNet connection is typically faster and easier than creating a Site-to-Site connection between VNets.

You can combine VNet to VNet communication with multi-site configurations. This lets you establish network topologies that combine cross-premises connectivity with inter-virtual network connectivity.

### **VNet peering**

You may want to consider connecting your VNets using VNet Peering. VNet peering does not use a VPN gateway and has different constraints. Additionally, VNet peering pricing is calculated differently than VNet-to-VNet VPN Gateway pricing.

**Site-to-Site VPNs**

To connect **all the computers** in a physical site to an Azure VNet, you can create a Site-to-Site VPN. In this configuration, you do not need to configure individual computers to connect to the VNet, **instead you configure a VPN device**, which acts as a gateway to the VNet.

When you use the Site-to-Site IPsec steps, you create and configure the local network gateways manually. The local network gateway for each VNet treats the other VNet as a local site. This lets you specify additional address space for the local network gateway in order to route traffic. If the address space for a VNet changes, you need to update the corresponding local network gateway to reflect that. It does not automatically update.

**ExpressRoute**

ExpressRoute is a service that enables Azure customers to create a dedicated connection to Azure, which does not connect through the public Internet. This contrasts with VPNs, which use encryption to tunnel securely through the public Internet. Because ExpressRoute connections are dedicated, they can offer faster speeds, higher security, lower latencies, and higher reliability than VPNs.

**Create and Configure VNet Peering**

Virtual network peering enables you to seamlessly connect two Azure virtual networks. Once peered, the virtual networks appear as one, for connectivity purposes.

The traffic between virtual machines in the peered virtual networks is routed through the **Microsoft backbone infrastructure**, much like traffic is routed between virtual machines in the same virtual network, through private IP addresses only.

Azure supports:

* **VNet peering** - connecting VNets within the **same Azure region.**
* **Global VNet peering** - connecting VNets across different Azure regions.

**Benefits**

1. Its best alternative to VPN for vNets because all network traffic between peered virtual networks is **private and routed over Azure internal networks** instead of public internet.
2. A low-latency, high-bandwidth connection between resources in different virtual networks.
3. The ability to transfer data across Azure subscriptions, deployment models, and across Azure regions.

**Pros and Cons over VPN Gateway**

**Pros**

1. Faster and easier to setup than VPN
2. No Public IP required.

**Cons**

1. Peering relationships are not transitive.

If you create peerings between:

* + VirtualNetwork1 & VirtualNetwork2
  + VirtualNetwork2 & VirtualNetwork3

There is no peering between VirtualNetwork1 and VirtualNetwork3 through VirtualNetwork2.

1. Cannot use overlapping address spaces.

**Pricing:**

[**https://azure.microsoft.com/en-us/pricing/details/virtual-network/**](https://azure.microsoft.com/en-us/pricing/details/virtual-network/)

**Configuring a Peering**

1. Portal 🡪 Search 🡪 Virtual Machine 🡪 +Create 🡪 Azure Virtual Machine
2. Basic Tab
   * Resource Group = Demo-westus-rg
   * Virual machine name = Demo-westus-vm
   * Region = WestUS
   * Image = Windows Server 2022 Datacenter: Azure Edition – x64 Gen2
   * Username = dssadmin
   * Password = \*\*\*\*\*\*\*

A screenshot of a computer

Description automatically generated

1. Next: Disks >
2. Next: Networking >
   * Virtual network = Create new
     1. Name = Demo-westus-vnet
     2. OK
   * A screenshot of a computer screen

     Description automatically generated
3. Review + create
4. Select the Demo-eastus-vnet 🡪 Settings 🡪 **Peerings 🡪 +Add**
   * This virtual network, Peering link name = Eastus-to-Westus
   * Remote virtual network, Peering link name = Westus-to-Eastus
   * Virtual network = Demo-westus-rg
   * Add
   * A screenshot of a computer

     Description automatically generated
5. Verify that Peering is showing status=**Connected**
6. **RDP to Web1-vm** 
   * **RDP to Demo-westus-vm (10.0.0.4)**

**Create a Point-to-Site VPN**

Diagram

Description automatically generated

**Generate Certificates – Self signed root certificate for P2S connection**

1. Open the Powershell command window and execute the following (Do not close the window)

**Windows Start 🡪 Windows Powershell ISE**

1. **Generate server and client certificate:**

**Execute the following command in the same PowerShell window opened earlier.**

$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 -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")

1. **🡪 Run Script**
2. **To obtain the public key (.cer file) of Root Certificate**
   1. Windows Start 🡪 **Manage User Certificates** 🡪 Personal 🡪 Certificates
   2. Locate the self-signed root certificate **(P2SRootCert)** 🡪 Right Click 🡪 **All Tasks**, and then click **Export**. This opens the **Certificate Export Wizard**.
   3. In the Wizard, click **Next**. Select **No, do not export the private key**, and then click **Next**.
   4. On the **Export File Format** page, select **Base-64 encoded X.509 (.CER).**, and then click **Next**.
   5. **File to Export**, **Browse** = d:\P2SRootCert.cer 🡪 **Next**.
   6. Click **Finish** to export the certificate. You will see **The export was successful**. Click **OK** to close the wizard.

A client certificate that is present on the device is used to authenticate the connecting user. Client certificates are generated from a trusted root certificate and then installed on each client computer. You can use a root certificate that was generated using an Enterprise solution, or you can generate a self-signed certificate.

The validation of the client certificate is performed by the VPN gateway and happens during establishment of the P2S VPN connection. The root certificate is required for the validation and must be uploaded to Azure

**Optional: Ensure that a Gateway Subnet is already created in a VNet**

1. Select Demo-eastus-vnet 🡪 Subnets **🡪 + Gateway subnet** 🡪 OK

**Create a Virtual Network Gateway**

1. Search 🡪 **Virtual network gateways** 🡪 +Create
   1. Name= Demo-eastus-vnet-gateway, . . . ,
   2. Gateway type = VPN
   3. VPN type = Route-based
   4. SKU = VPNGw1 (Don't select Basic as it doesn’t support Azure Certificate Authentication)
   5. Virtual network = Demo-eastus-vnet
   6. Gateway subnet address range = 192.168.0.0/24 (Or select already created Gateway Subnet)
   7. Public IP address = Create new
   8. Public IP address name = Demo-eastus-vnet-gateway-ip
   9. Enable active-active mode = Disabled
   10. Configure BGP = Disabled
   11. Review + Create 🡪 Create

A screenshot of a computer

Description automatically generated

### Note: Provisioning a virtual network gateway may take up to 15 minutes.

**Upload the root certificate .cer file**

1. Open the Root certificate (not child) with a text editor, such as Notepad. Copy the content between and **excluding** -----BEGIN CERTIFICATE------ and -----END CERTIFICATE------

A screenshot of a computer code

Description automatically generated

1. Select **Demo-eastus-vnet-gateway** created earlier 🡪 **Point-to-site configuration**,
   1. **Address Pool**=172.16.201.0/24 (is the pool of IP addresses from which clients that connect will receive an IP address.)
   2. Tunnel type = **IKEv2 and OpeVPN (SSL)**
   3. **Root Certificates**: Name=RootCert1, Public Certficate Data <Value copied in step 6>

A screenshot of a computer

Description automatically generated

Note: You can add up to **20 trusted** **root certificates**.

**Download and Install VPN client**

1. **Demo-eastus-vnet-gateway** 🡪 **Point-to-site configuration** 🡪 **Download VPN client**
2. Select X64 🡪 Download
3. Execute the downloaded EXE file

**Connect to Virtual Networking using VPN Client**

1. Windows Start 🡪 VPN Settings 🡪 Click on Connect



1. Conntect 🡪 Yes
2. To verify that your VPN connection is active, open an elevated command prompt, and **run ipconfig/all**.

* Note the Private IP Assgined to your Machine eg: **172.16.201.3**

A screenshot of a computer

Description automatically generated

1. From your local machine browser visit <http://192.168.1.4> (Private IP Address of Web1-vm)
2. From your local machine, You can RDP to **192.168.0.1 (Web1-vm)**

**Microsoft Labs:**

1. **Implement Virtual Networking:**

<https://microsoftlearning.github.io/AZ-104-MicrosoftAzureAdministrator/Instructions/Labs/LAB_04-Implement_Virtual_Networking.html>

1. Implement Intersite Connectivity

<https://microsoftlearning.github.io/AZ-104-MicrosoftAzureAdministrator/Instructions/Labs/LAB_05-Implement_Intersite_Connectivity.html>