R&D Document

Topic: Azure Virtual Network – CIDR Ranges, Subnets, and VNet Peering

1. Objective

This document outlines the steps to:

• Create a Virtual Network (VNet) and subnets
• Launch VMs (Windows/Linux) in subnets
• Enable communication between VMs in the same VNet
• Create two VNets and set up VNet Peering
• Enable cross-VNet VM communication
• Use Azure Bastion for secure VM access
• Includes screenshots for each major configuration step

2. Key Concepts

CIDR (Classless Inter-Domain Routing)
• CIDR defines IP address ranges
• Format: <IP>/<prefix> e.g. 10.0.0.0/16
• Helps create large or small address spaces as needed

Azure Virtual Network (VNet)
• Logical isolation in Azure
• Hosts subnets and resources

Subnet
• Logical partition within a VNet
• Helps isolate services and manage IP ranges

Azure Bastion
• Allows VM access via RDP/SSH without public IP
• Uses browser-based secure access

VNet Peering
• Connects two VNets privately
• Types:
  - Intra-region Peering (same region)
  - Global Peering (across regions)

3. Prerequisites

• Azure account with valid subscription
• Azure Portal access
• Basic understanding of VMs, networks

4. Task 1: Single VNet with Subnets and VMs

Step 1: Create VNet and Subnets
• Go to Azure Portal → Virtual Networks → +Create
• Name: Vnet-Demo
• Address space: 10.0.0.0/16
• Create two subnets:
  - Subnet-1: 10.0.1.0/24 (for Windows VM)
  - Subnet-2: 10.0.2.0/24 (for Linux VM)

Step 2: Launch VMs
• Create Windows VM in Subnet-1 (Name: Win-VM, OS: Windows Server 2019)
• Create Linux VM in Subnet-2 (Name: Linux-VM, OS: Ubuntu 22.04 LTS)
• No Public IPs — use Bastion

Step 3: Use Azure Bastion
• Enable Azure Bastion when creating VNet
• Use browser-based Bastion to access both VMs securely

Step 4: Test Connectivity
• Connect to Linux VM → ping private IP of Windows VM
• Connect to Windows VM → ping private IP of Linux VM

5. Task 2: Two VNets and VNet Peering

Step 1: Create VNet-1
• Name: vnet-1
• Address space: 10.0.0.0/16
• Subnet: 10.0.0.0/24

Step 2: Create VNet-2
• Name: vnet-2
• Address space: 10.1.0.0/16
• Subnet: 10.1.0.0/24

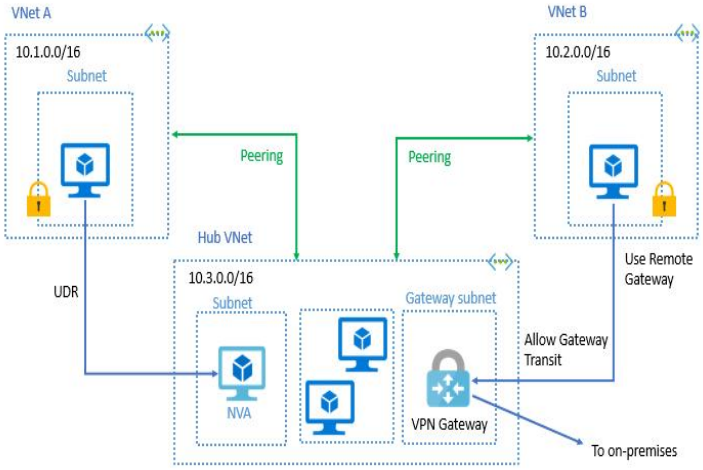
Step 3: Launch VMs
• VM-1 in vnet-1 and VM-2 in vnet-2
• Both use Ubuntu image
• No Public IPs

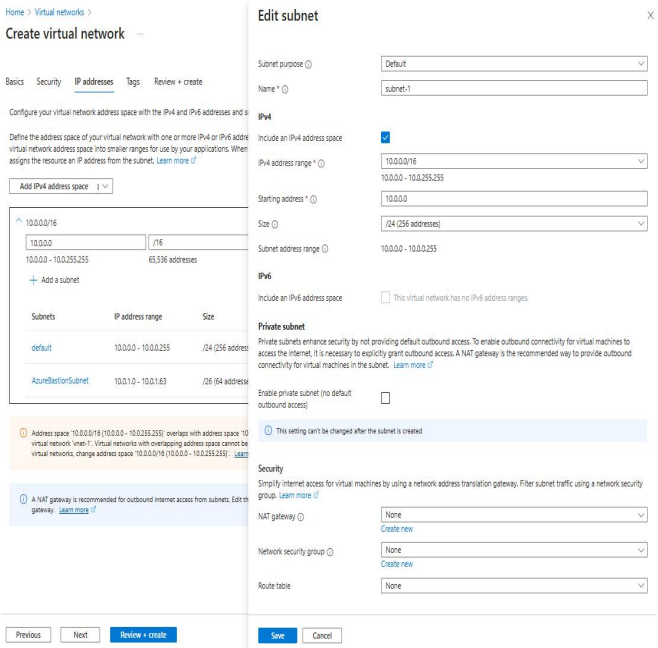
Step 4: Configure VNet Peering
• Go to vnet-1 → Peerings → Add:
  - Name: vnet-1-to-vnet-2
  - Remote: vnet-2
  - Allow forwarded traffic ✔

• Go to vnet-2 → Peerings → Add:
  - Name: vnet-2-to-vnet-1
  - Remote: vnet-1
  - Allow forwarded traffic ✔

Step 5: Test Communication
• SSH into VM-1 and run:
  ping <private-ip-of-vm-2>
• SSH into VM-2 and run:
  ping <private-ip-of-vm-1>

6. Clean Up
• Go to Resource Groups → test-rg
• Click “Delete Resource Group” to remove all related resources





Add peering

vnet-1

Virtual network peering enables you to seamlessly connect two or more virtual networks in Azure. This will allow resources in either virtual network to directly connect and communicate with resources in the peered virtual network.

Remote virtual network summary

Peering link name * vnet-2-to-vnet-1

Virtual network deployment model
- (•) Resource manager
- ( ) Classic

I know my resource ID [ ]

Subscription * Contoso Subscription

Virtual network * vnet-2 (test-rg)

Remote virtual network peering settings

Allow 'vnet-2' to access 'vnet-1' [x]

Allow 'vnet-2' to receive forwarded traffic from 'vnet-1' [x]

Allow gateway or route server in 'vnet-2' to forward traffic to 'vnet-1' [ ]

Enable 'vnet-2' to use 'vnet-1's' remote gateway or route server [ ]

Local virtual network summary

Peering link name * vnet-1-to-vnet-2

Local virtual network peering settings

Allow 'vnet-1' to access 'vnet-2' [x]

Allow 'vnet-1' to receive forwarded traffic from 'vnet-2' [x]

Allow gateway or route server in 'vnet-1' to forward traffic to 'vnet-2' [ ]

Enable 'vnet-1' to use 'vnet-2's' remote gateway or route server [ ]

Add Cancel

Give feedback

Communicate between VMs

At the bash prompt for vm-1, enter ping -c 4 10.1.0.4.

```
Output
azureuser@vm-1:~$ ping -c 4 10.1.0.4

PING 10.1.0.4 (10.1.0.4) 56(84) bytes of data.

64 bytes from 10.1.0.4: icmp_seq=1 ttl=64 time=2.29 ms

64 bytes from 10.1.0.4: icmp_seq=2 ttl=64 time=1.06 ms

64 bytes from 10.1.0.4: icmp_seq=3 ttl=64 time=1.30 ms

64 bytes from 10.1.0.4: icmp_seq=4 ttl=64 time=0.998 ms


--- 10.1.0.4 ping statistics ---

4 packets transmitted, 4 received, 0% packet loss, time 3004ms

rtt min/avg/max/mdev = 0.998/1.411/2.292/0.520 ms
```

Close the Bastion connection to vm-1.

Repeat the steps in Connect to a virtual machine to connect to vm-2.

At the bash prompt for vm-2, enter ping -c 4 10.0.0.4.

```
Output
azureuser@vm-2:~$ ping -c 4 10.0.0.4

PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.

64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=1.81 ms

64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=3.35 ms

64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=0.811 ms

64 bytes from 10.0.0.4: icmp_seq=4 ttl=64 time=1.28 ms
```