

R&D Report

CIDR Ranges, Subnet Configuration, and VNet Peering

Prepared by: Virat Pandey

Internship: Cloud Infrastructure & Security – Celebal Technology

Date: June 2025

Table of Contents

- 1. Introduction
- 2. Overview of CIDR and VNet Concepts
- 3. Implementation Plan
 - 3.1 Creating Virtual Network and Subnets
 - 3.2 Deploying VMs in Separate Subnets
 - 3.3 Enabling Internal Communication Between VMs
 - 3.4 Creating Second VNet for Peering
 - 3.5 Peering VNet-Project-A and VNet-Project-B
 - 3.6 Deploying VM in VNet-B
 - 3.7 Testing Ping Across Peered VNets
- 4. Result Verification
- 5. Conclusion

1. Introduction

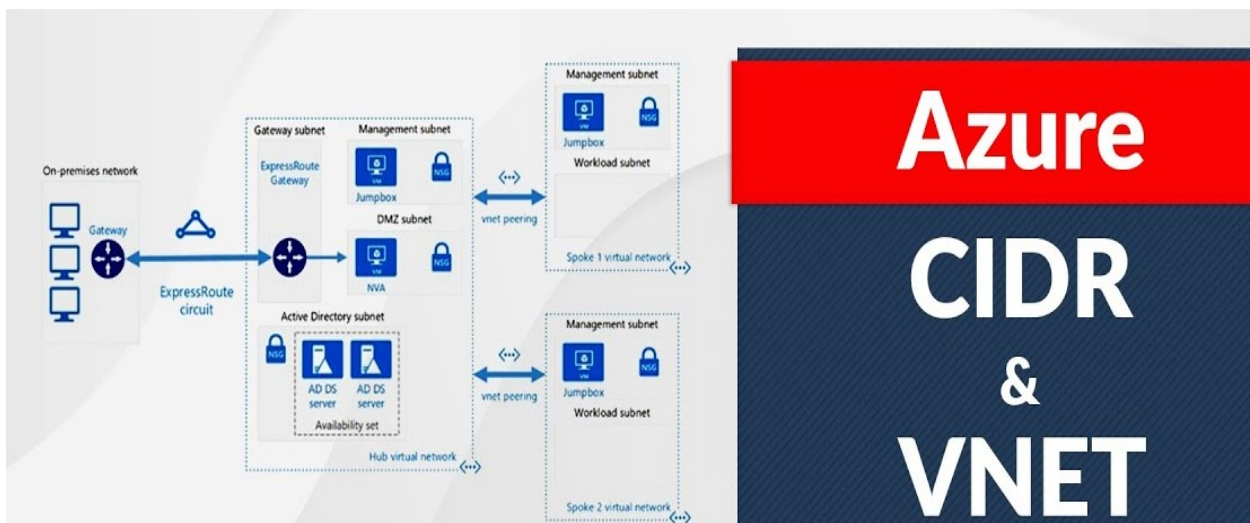
This report documents the practical implementation of Azure Virtual Network (VNet) configuration, CIDR-based subnetting, and inter-VNet peering. The goal was to build a simulated cloud network environment where multiple virtual machines (VMs) are deployed in isolated subnets across different VNets and can communicate securely and efficiently through private IP ranges.

The implementation showcases real-world usage of CIDR blocks, subnet design, and Azure VNet peering for scalable and secure network architectures.

2. Overview of CIDR and VNet Concepts

CIDR (Classless Inter-Domain Routing) allows flexible allocation of IP addresses and efficient subnetting. A VNet in Azure uses CIDR ranges to define its IP space, and each subnet is a subdivision of that space.

In this assignment, multiple subnets were created within a VNet using different CIDR blocks. Network Security Groups (NSGs) were configured to allow ping (ICMP), and a second VNet was created and peered with the first to allow inter-VNet communication.



3.1 Creating Virtual Network and Subnets

- Resource Group: VNetPeeringLab
- Virtual Network Name: VNet-Project-A
- Address Space: 10.0.0.0/16
- Subnets Created:
 - Web-Subnet: 10.0.0.0/24
 - App-Subnet: 10.0.1.0/24

The screenshot shows the 'Create virtual network' page in the Azure portal, specifically the 'IP addresses' tab. The page has a dark theme. At the top, there's a title 'Create virtual network' followed by a three-dot menu. Below the title are tabs: 'Basics', 'Security', 'IP addresses' (which is selected and underlined), 'Tags', and 'Review + create'. A paragraph of text explains how to configure the virtual network address space with IPv4 and IPv6 addresses and subnets, with a 'Learn more' link. Below this is a '+ Add a subnet' button. The main content area shows a configuration for the address space '10.0.0.0/16'. It includes a warning that this prefix overlaps with a virtual network named 'Threetier-VNet'. There are input fields for the IP address '10.0.0.0' and the prefix '/16', showing the resulting range '10.0.0.0 - 10.0.255.255' and '65,536 addresses'. Below this is a table with columns: 'Subnets', 'IP address range', 'Size', and 'NAT gateway'. The table lists two subnets: 'web-subnet' and 'App-Subnet', each with their respective IP ranges and sizes of 256 addresses. Each row has edit and delete icons.

Create virtual network ...

Basics Security IP addresses Tags Review + create

Configure your virtual network address space with the IPv4 and IPv6 addresses and subnets you need. [Learn more](#)

Define the address space of your virtual network with one or more IPv4 or IPv6 address ranges. Create subnets to segment the virtual network address space into smaller ranges for use by your applications. When you deploy resources into a subnet, Azure assigns the resource an IP address from the subnet. [Learn more](#)

+ Add a subnet

10.0.0.0/16 [Delete address space](#)

This address prefix overlaps with virtual network 'Threetier-VNet'. If you intend to peer these virtual networks, change the address space. [Learn more](#)

10.0.0.0 /16
10.0.0.0 - 10.0.255.255 65,536 addresses

Subnets	IP address range	Size	NAT gateway
web-subnet	10.0.0.0 - 10.0.0.255	/24 (256 addresses)	-
App-Subnet	10.0.1.0 - 10.0.1.255	/24 (256 addresses)	-

As part of the implementation, a virtual network named **VNet-Project-A** was created to define a custom IP address space using the CIDR block 10.0.0.0/16. This VNet acts as the foundational layer for deploying subnets and virtual machines. It enables isolated and secure communication between resources, and serves as the core network for configuring address ranges, routing, and connectivity in the simulated cloud environment.

3.2 Deploying VMs in Separate Subnets

- win-vm-A: Deployed in Web-Subnet
- linux-vm-A: Deployed in App-Subnet
- Size: Standard B1s
- Public IP: Assigned
- Inbound Port Rule: RDP (Windows), SSH (Linux)

Create a virtual machine ...

Help me create a low cost VM | Help me create a VM optimized for high availability | Help me choose the right VM s

Virtual machine name * ⓘ win-vm-A ✓

Region * ⓘ (Asia Pacific) Central India ✓

Availability options ⓘ Availability zone ✓

Zone options ⓘ

- ☒ Self-selected zone
Choose up to 3 availability zones, one VM per zone
- ☐ Azure-selected zone (Preview)
Let Azure assign the best zone for your needs

i Using an Azure-selected zone is not supported in region 'Central India'.

Availability zone * ⓘ Zone 1 ✓

🔧 You can now select multiple zones. Selecting multiple zones will create one VM per zone. [Learn more](#)

Security type ⓘ Trusted launch virtual machines ✓
[Configure security features](#)

Image * ⓘ Ubuntu Server 24.04 LTS - x64 Gen2 (free services eligible) ✓
[See all images](#) | [Configure VM generation](#)

VM architecture ⓘ

- ☐ Arm64
- ☒ x64


Run with Azure Spot discount ⓘ ☐

Here we have created our first virtual machine that is windows virtual machine in the web subnet now we will do the same for linux based virtual machine now we will create another virtual machine and it will be placed in the app subnet. Remember both the vm are inside different subnets but are within the same virtual network so they can ping each other which we will see in the upcoming screenshots that how we peered between to virtual machines in different subnets inside a same network.

DeleteCancelRedeployDownloadRefresh

✓

Your deployment is complete




Deployment name: CreateVm-MicrosoftWindowsServer.WindowsSe...

Subscription: [Azure subscription 1](#)

Resource group: [VNetPeeringLab](#)

Start time: 6/29/2025, 10:39:29 AM

Correlation ID: 34f01233-9614-424a-bfc2-50617e2d160c



Deployment details

Resource	Type	Status	Operation details
✓ win-vm-A	Microsoft.Compute/virtualMachines	OK	Operation details
✓ win-vm-a973	Microsoft.Network/networkInterfa...	OK	Operation details
✓ win-vm-A-nsg	Microsoft.Network/networkSecurit...	OK	Operation details
✓ win-vm-A-ip	Microsoft.Network/publicIpAddre...	OK	Operation details

Next steps

Setup auto-shutdown

Recommended

Monitor VM health, performance and network dependencies

Recommended

Run a script inside the virtual machine

Recommended

Go to resource

Create another VM


Here this is the windows vm in the web subnet we have deployed this

Now the linux vm inside the app subnet-

DeleteCancelRedeployDownloadRefresh

✓

Your deployment is complete




Deployment name: CreateVm-canonical.0001-com-ubuntu-server-j...

Subscription: [Azure subscription 1](#)

Resource group: [VNetPeeringLab](#)

Start time: 6/29/2025, 11:40:53 AM

Correlation ID: 29a2daef-d669-4a79-815d-30ceae7df797



Deployment details

Next steps

Setup auto-shutdown

Recommended

Monitor VM health, performance and network dependencies

Recommended

Run a script inside the virtual machine

Recommended

Go to resource

Create another VM

This concludes the creation of virtual machines inside different subnets but within the same Virtual network. Now we will enable the internal communication between these two virtual machines we will be sending a ping request from the windows vm to the linux based vm

3.3 Enabling Internal Communication Between VMs

- ICMP (ping) was not enabled by default.
- Added inbound NSG rule:
 - Protocol: ICMP
 - Source: Any
 - Destination: Any
 - Port: *
 - Action: Allow

The screenshot displays the configuration for a Network Security Group (NSG) named 'win-vm-A-nsg' in the Azure portal. The NSG is attached to the network interface 'win-vm-a973' and impacts 0 subnets and 1 network interface. A '+ Create port rule' button is visible in the top right. Below the header, there is a search bar and filter buttons for 'Source == all', 'Destination == all', 'Protocol == all', and 'Action == all'. The main section shows a table of port rules, categorized into 'Inbound port rules (5)' and 'Outbound port rules (3)'. The inbound rules table has columns for Priority, Name, Port, and Protocol. The rule 'Allow-ICMP' at priority 310 is highlighted, showing it allows ICMP traffic on any port. Other rules include RDP (priority 300), AllowVnetInBound (priority 65000), AllowAzureLoadBalancerInBound (priority 65001), and DenyAllInBound (priority 65500).

Priority ↑	Name	Port	Protocol
Inbound port rules (5)			
300	RDP	3389	TCP
310	Allow-ICMP	Any	ICMP
65000	AllowVnetInBound ⓘ	Any	Any
65001	AllowAzureLoadBalancerInBound ⓘ	Any	Any
65500	DenyAllInBound ⓘ	Any	Any
Outbound port rules (3)			

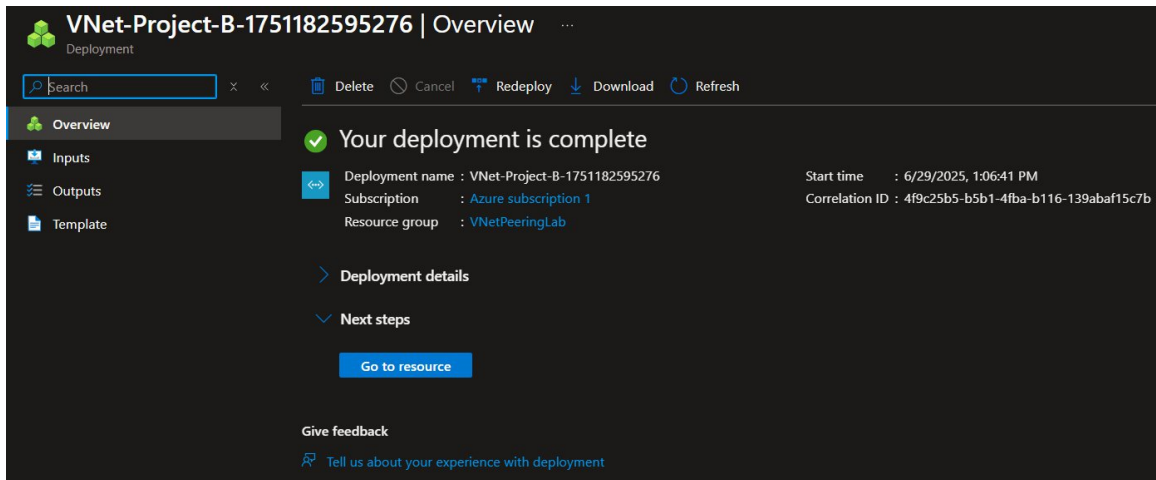
Here we created a new port rule inside the network security group so that the verifying can be carried out as the ICMP (ping) was not enabled by default. So first we enabled that and then we proceeded with creating a second virtual network for peering.

We already have a virtual network-A so for peering and connection testing we have to create one more virtual network

We will name this as virtual network-B and deploy a virtual machine in this vnet also via adding a subnet first.

3.4 Creating Second VNet for Peering

- Virtual Network Name: VNet-Project-B
- Address Space: 10.1.0.0/16
- Subnet: b-subnet (10.1.0.0/24)



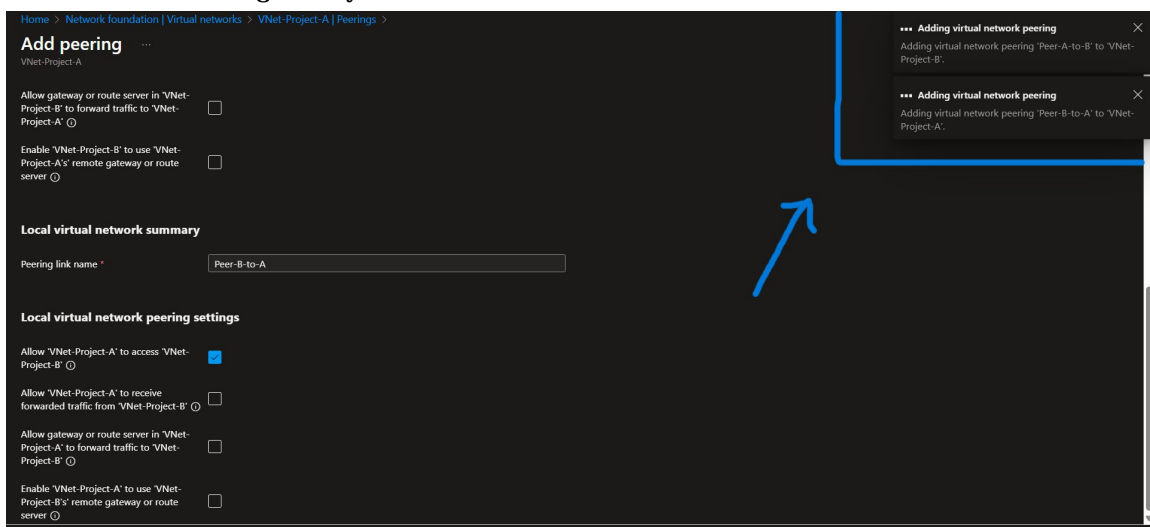
3.5 Peering VNet-Project-A and VNet-Project-B

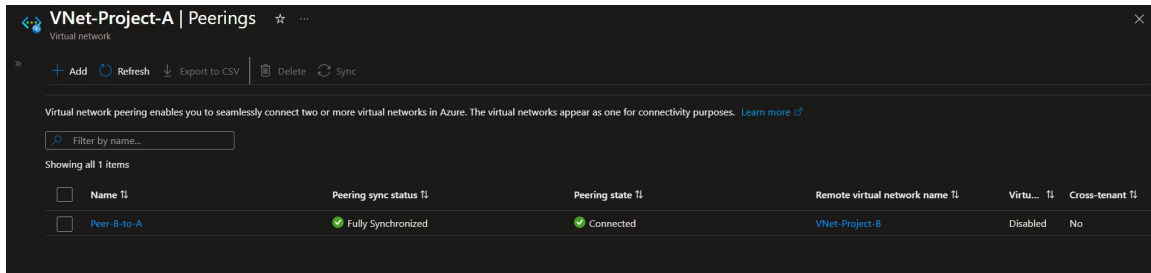
Peering A → B:

- Name: Peer-A-to-B
- Allow traffic: ✓
- Forwarded traffic, gateway: ✗

Peering B → A:

- Name: Peer-B-to-A
- Allow traffic: ✓
- Forwarded traffic, gateway: ✗*





Here we have configured the peering setting in the above picture.

3.6 Deploying VM in VNet-B

- VM Name: linux-vm-B
- Subnet: b-subnet
- Size: Standard B1s (or alternative due to quota)
- Public IP: Assigned
- Inbound Rule: SSH (Port 22)

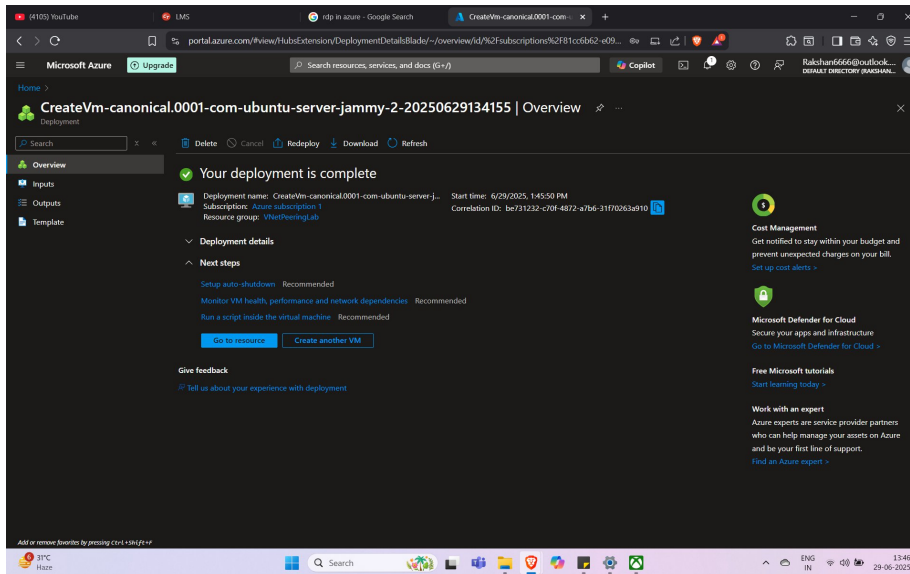
Create a virtual machine

✓ Validation passed

[Help me create a low cost VM](#)
[Help me create a VM optimized for high availability](#)
[Help me choose the right VM size for my workload](#)

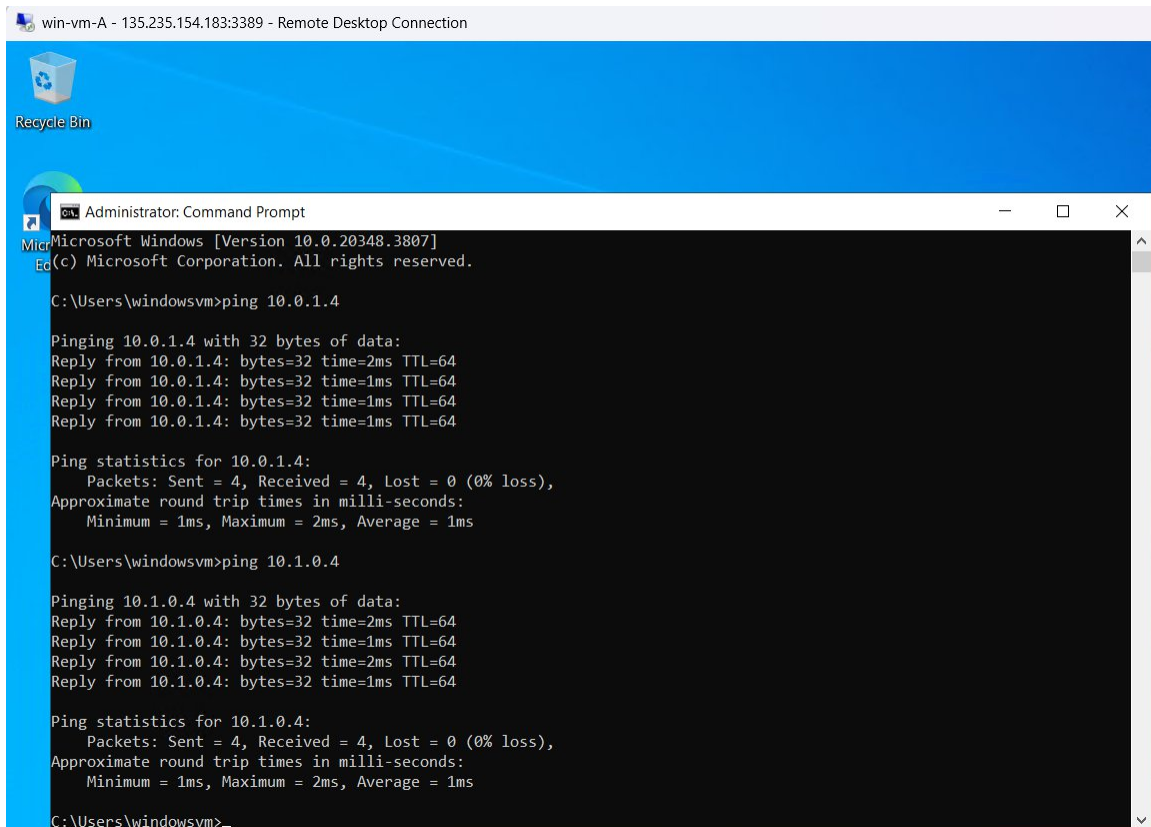
Basics

Subscription	Azure subscription 1
Resource group	VNetPeeringLab
Virtual machine name	linux-vm-B
Region	Central India
Availability options	No infrastructure redundancy required
Zone options	Self-selected zone
Security type	Trusted launch virtual machines
Enable secure boot	Yes
Enable vTPM	Yes
Integrity monitoring	No
Image	Ubuntu Server 22.04 LTS - Gen2
VM architecture	x64
Size	Standard B1s (1 vcpu, 1 GiB memory)
Enable Hibernation	No
Authentication type	Password
Username	linuxvm2
Public inbound ports	SSH
Azure Spot	No



3.7 Testing Ping Across Peered VNets

- From win-vm-A (in VNet-A) → pinged linux-vm-B (in VNet-B) using private IP
- Result: Successful reply



The above screenshot includes the terminal responses of two successful ping operations: first, between the Windows and Linux VMs within VNet-Project-A (intra-VNet communication), and second, between the Windows VM in VNet-Project-A and the Linux VM in VNet-Project-B (inter-VNet communication using VNet Peering). This confirms that both subnet-level and VNet-level connectivity have been successfully implemented.

4. Result Verification

All steps were executed successfully. VMs in separate subnets were able to communicate through private IPs after NSG modification. Peering between VNet-A and VNet-B enabled cross-VNet communication, verified via ping.

4. Conclusion

This implementation demonstrated a clear understanding of CIDR-based subnetting, NSG configurations, and VNet peering within Microsoft Azure. These skills are essential for designing scalable, secure, and modular cloud infrastructures that mimic real-world enterprise setups.