Prepared by: Nandini Goyal
Intern at Celebal Technologies
Department: Cloud Infra & Security
Date: 28 June 2025

R&D Document

Practical Implementation: Deploying VMs Across Subnets and Peered VNets in Azure

Table of Contents

Serial Number	Content	Page Number
1	Introduction	3
2	CIDR Ranges in Azure VNets 2.1 What is CIDR? 2.2 Common CIDR Examples in Azure 2.3 CIDR Best Practices	3
3	Azure Virtual Network (VNet) 3.1 What is a VNet? 3.2 Features of Azure VNet	3
4	Subnets in Azure 4.1 What is a Subnet? 4.2 Benefits of Subnetting	4
5	VNet Peering and Its Types 5.1 What is VNet Peering? 5.2 Types of VNet Peering 5.3 Key Points	4
6	Practical Implementation 6.1 Create Virtual Network and Subnets 6.2 Launch VMs in Subnets 6.3 Configure NSG and Inbound Rules 6.4 Enable and Test Ping 6.5 Create a Second VNet (VNet2) 6.6 VNet Peering Between VNet1 and VNet2 6.7 Test Cross-VNet Communication	5
7	Troubleshooting and Fixes	19
8	Learnings	19
9	Conclusion	19
10	References	19

1. Introduction

This R&D document covers key Azure networking concepts including CIDR ranges, subnets, virtual networks (VNets), and VNet peering. These components are essential for designing secure and scalable cloud networks.

2. CIDR Ranges in Azure VNets

2.1 What is CIDR?

CIDR stands for **Classless Inter-Domain Routing**. It is a method of representing IP address ranges using a compact format like:

10.0.0.0/16

- The /16 means the first 16 bits are the network portion.
- This gives **65,536** addresses (2^{16}).

2.2 Common CIDR Examples in Azure

CIDR Range	Total IPs	Use Case
10.0.0.0/16	65,536	Entire VNet
10.0.0.0/24	256	Subnet
10.0.1.0/24	256	Another Subnet
10.1.0.0/16	65,536	VNet2

2.3 CIDR Best Practices

- Avoid overlapping CIDR ranges if planning VNet peering.
- Reserve IPs for Azure (5 per subnet are reserved).
- Plan ahead for scalability.

3. Azure Virtual Network (VNet)

3.1 What is a VNet?

An **Azure Virtual Network** is a logically isolated network in Azure where you can launch resources like VMs, web apps, databases, etc.

3.2 Features of Azure VNet:

Feature	Description
Isolation	VNets are isolated from each other by default
Subnetting	You can divide a VNet into multiple subnets
NSGs	Use Network Security Groups to filter traffic
Peering	Connect multiple VNets with VNet peering
Routing	Azure handles routing between subnets
Hybrid Connectivity	Supports VPN Gateway, ExpressRoute

4. Subnets in Azure

4.1 What is a Subnet?

A **subnet** divides a virtual network into smaller IP ranges. Each VM resides in a subnet and can communicate with others.

Example:

- VNet: 10.0.0.0/16
- Subnet1: 10.0.0.0/24
- Subnet2: 10.0.1.0/24

4.2 Benefits of Subnetting:

- Logical grouping (e.g., web tier, DB tier)
- Apply NSG at subnet level
- Segment network traffic

5. VNet Peering and Its Types

5.1 What is VNet Peering?

VNet Peering allows traffic to flow between two Azure virtual networks **securely and privately**, using Azure's backbone network.

5.2 Types of VNet Peering:

Type	Description
Intra-region Peering	VNets in the same region (e.g., East US)
Global Peering	VNets in different regions
One-way Peering	Only one VNet initiates peering
Two-way Peering	Both VNets initiate and accept traffic

5.3 Key Points:

- No need for gateways
- Low latency, high bandwidth
- Cannot use overlapping IP ranges
- Supports transitive routing via NVA or gateways

6. Practical Implementation

Now we put the theory into action.

We will:

- Create two VNets
- Launch Windows and Linux VMs in each
- Enable communication (ping)
- Configure peering

6.1 Create Virtual Network and Subnets

Go to Azure Portal → **Create a Resource → Virtual Network**

Set:

- Name: VNet1
- Address space: 10.0.0.0/16

Subnets:

- LinuxSubnet → 10.0.0.0/24
- WindowsSubnet → 10.0.1.0/24

Create Resource Group: Week4-Networking-RG

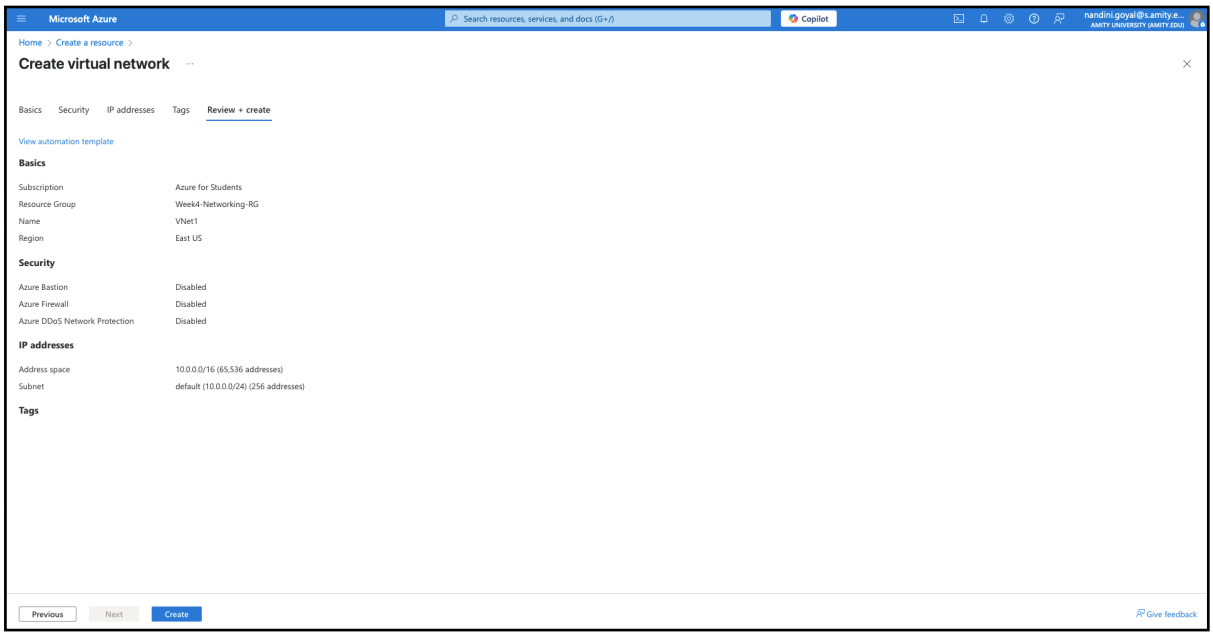

Figure 1: VNet1 creation summary

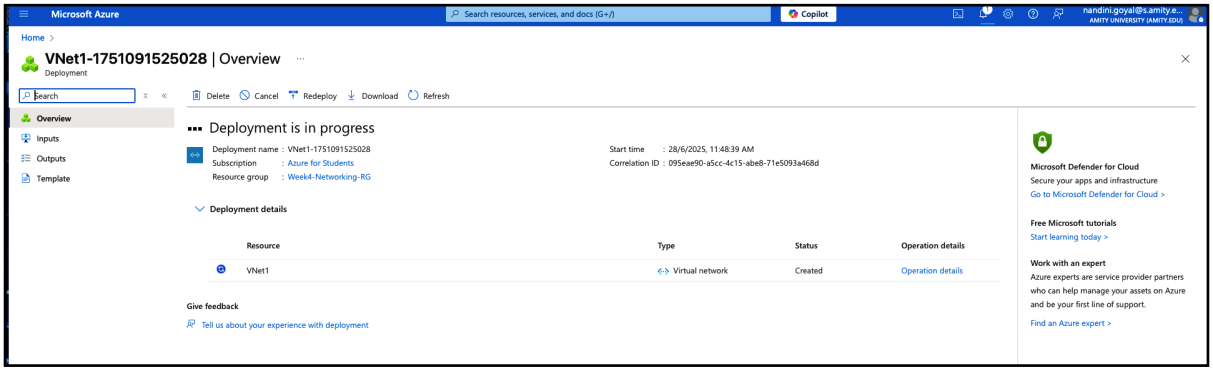

Figure 2: VNet1 Deployed

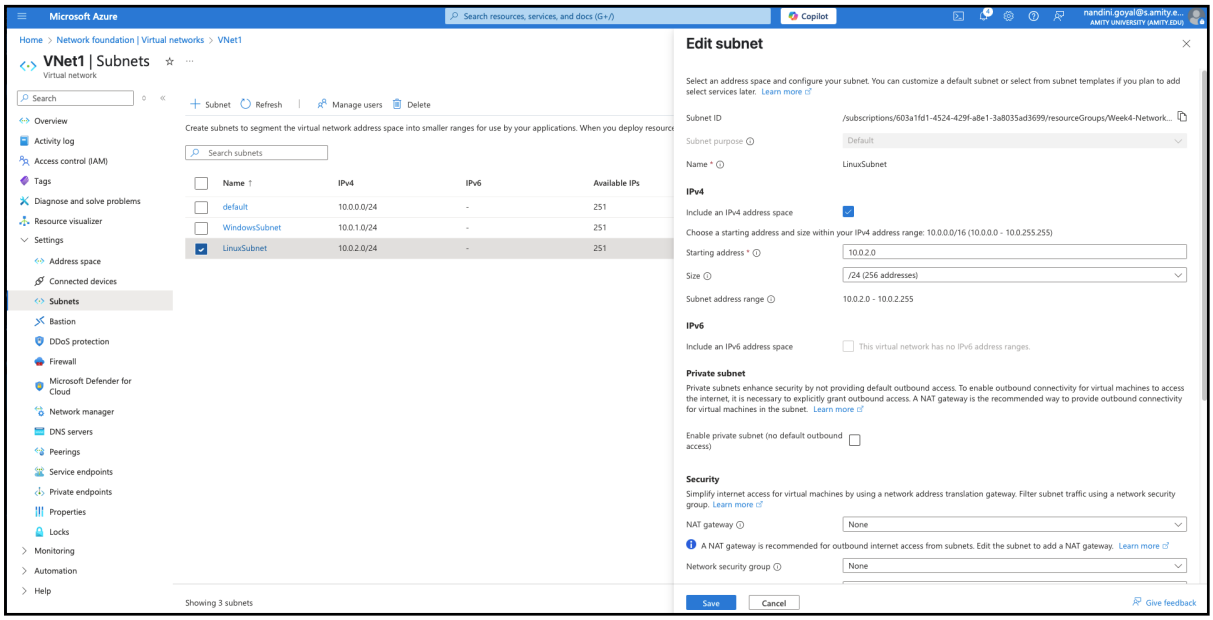

Figure 3: Subnet Configuration (WindowsSubnet)

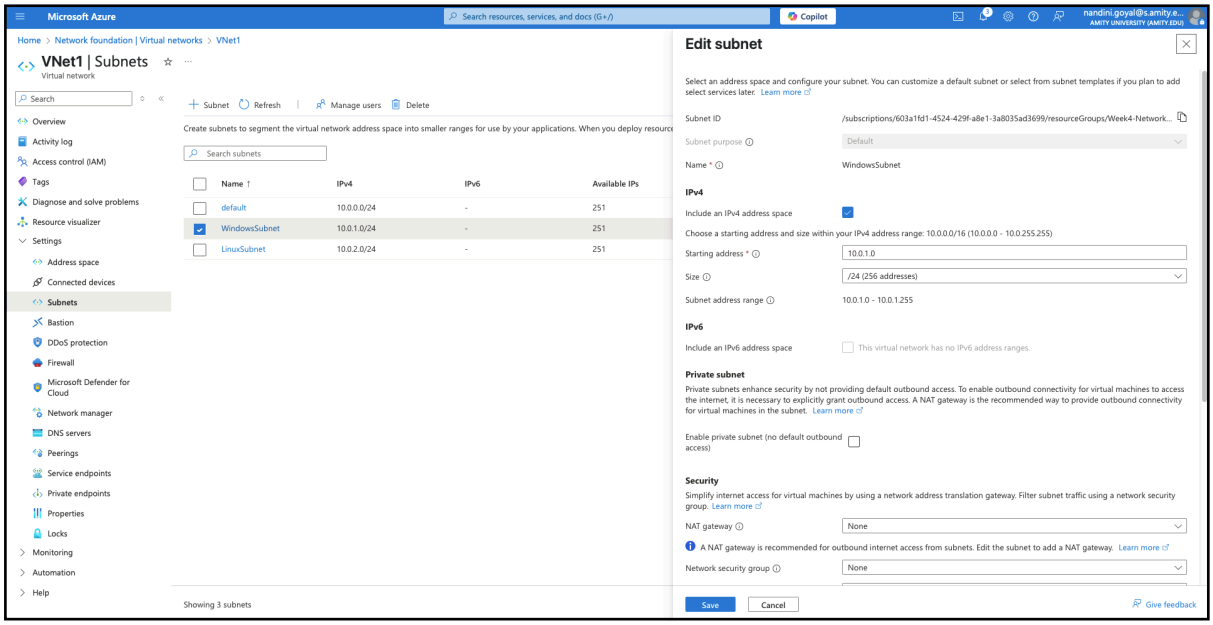

Figure 4: Subnet Configuration (LinuxSubnet)

6.2 Launch VMs in Subnets

- **Create Linux VM (Ubuntu)**
  - Image: Ubuntu Server 24.04 LTS - x64 Gen2
  - Size: Standard B1s
  - Auth: SSH Key (generated)
  - Subnet: LinuxSubnet
  - Public Port: **SSH (22)**

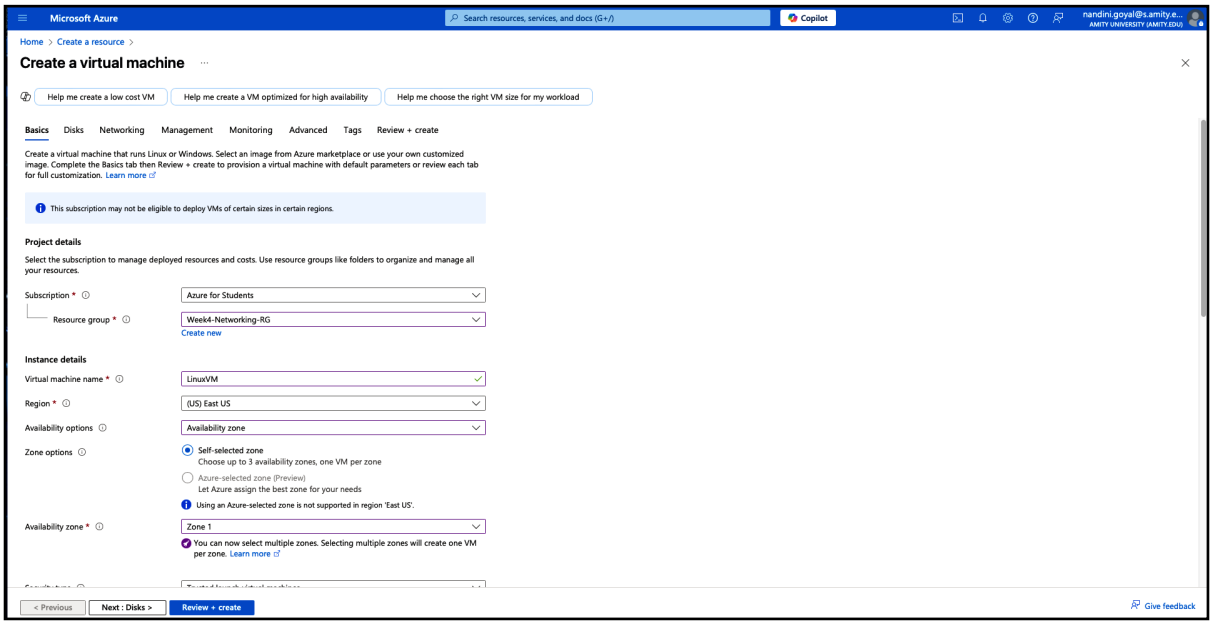

Figure 5: Making of Linux VM

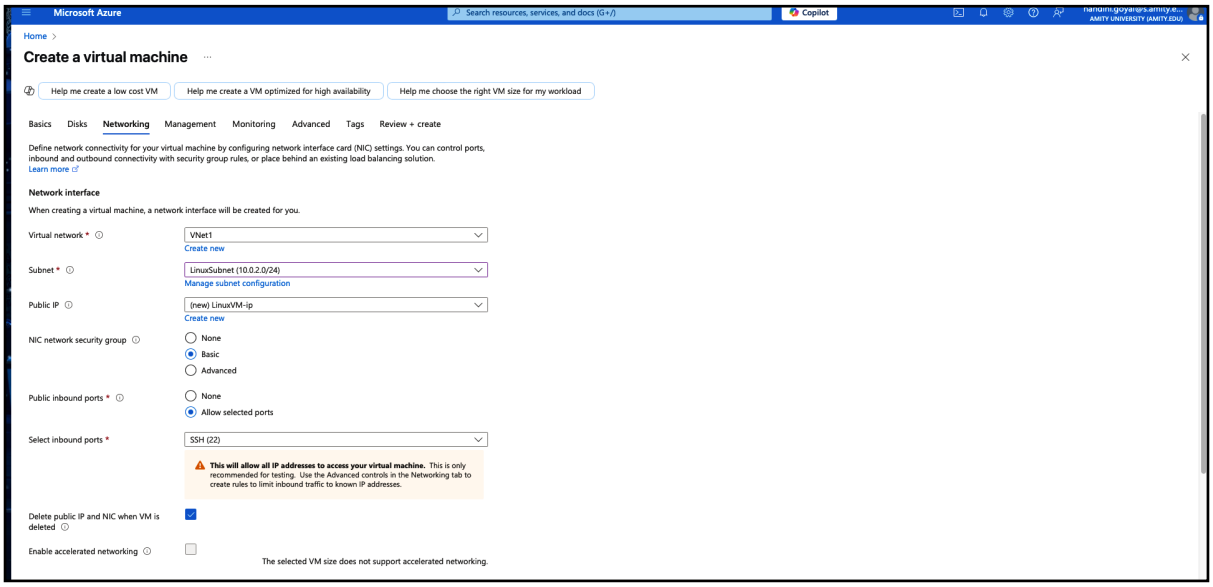

Figure 6: Networking tab with Linux Subnet

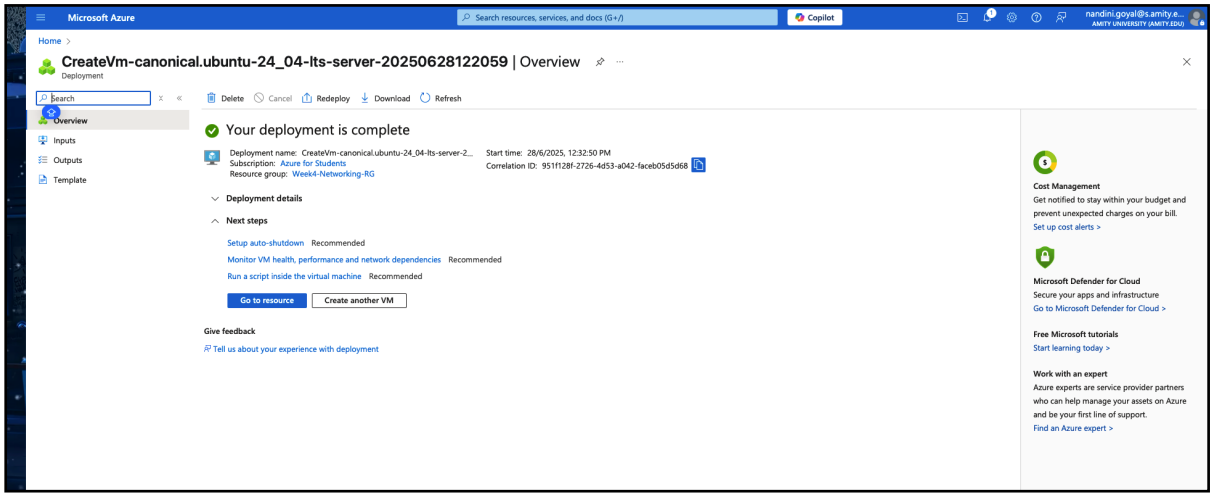

Figure 7: Deployment of Linux VM

```
Last login: Sat Jun 28 11:11:24 on console
Matching Java Virtual Machines (2):
    1.8.202.08 (x86_64) "Oracle Corporation" - "Java" /Library/Internet Plug-Ins/JavaAppletPlugin.plugin/Contents/Home
    1.8.0_202 (x86_64) "Oracle Corporation" - "Java SE 8" /Library/Java/JavaVirtualMachines/jdk1.8.0_202.jdk/Contents/Home
(base) kesar@Kunjals-Mac ~ % chmod 400 ~/Downloads/my-key.pem

chmod: /Users/kesar/Downloads/my-key.pem: No such file or directory
(base) kesar@Kunjals-Mac ~ % chmod 400 ~/Downloads/linuxvm_key.pem

(base) kesar@Kunjals-Mac ~ % ssh -i ~/Downloads/linuxvm_keykey.pem linuxadmin@20.121.45.156

Warning: Identity file /Users/kesar/Downloads/linuxvm_keykey.pem not accessible: No such file or directory.
The authenticity of host '20.121.45.156 (20.121.45.156)' can't be established.
ED25519 key fingerprint is SHA256:0VbjP5ptw2SWiuSqfqAQ19EjLEj+ktmKY9HYX3fgHg4.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '20.121.45.156' (ED25519) to the list of known hosts.
linuxadmin@20.121.45.156: Permission denied (publickey).
(base) kesar@Kunjals-Mac ~ % ssh -i ~/Downloads/linuxvm_key.pem linuxadmin@20.121.45.156

Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.11.0-1015-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

  System information as of Sat Jun 28 07:14:52 UTC 2025

   System load:  0.0               Processes:             110
   Usage of /:   5.5% of 28.02GB   Users logged in:       0
   Memory usage: 29%               IPv4 address for eth0: 10.0.2.4
   Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

linuxadmin@LinuxVM:~$ whoami
linuxadmin
linuxadmin@LinuxVM:~$ ls
linuxadmin@LinuxVM:~$ uname -a
Linux LinuxVM 6.11.0-1015-azure #15~24.04.1-Ubuntu SMP Thu May  1 02:52:08 UTC 2025 x86_64 x86_64 x86_64 GNU/Linux
linuxadmin@LinuxVM:~$
```

Figure 8: Linux VM through terminal

- **Create Windows VM (Windows Server 2022)**
  - Image: Windows Server 2022 Datacenter
  - Auth: Username + Password
  - Subnet: WindowsSubnet
  - Public Port: **RDP (3389)**

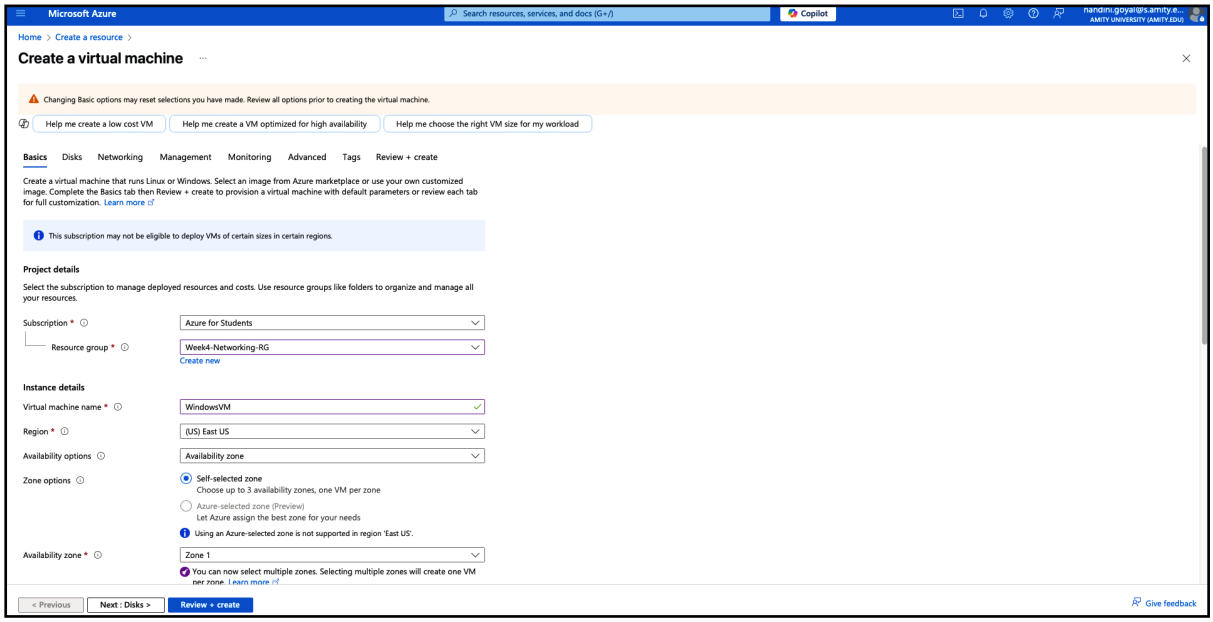

Figure 9: Making of Windows VM

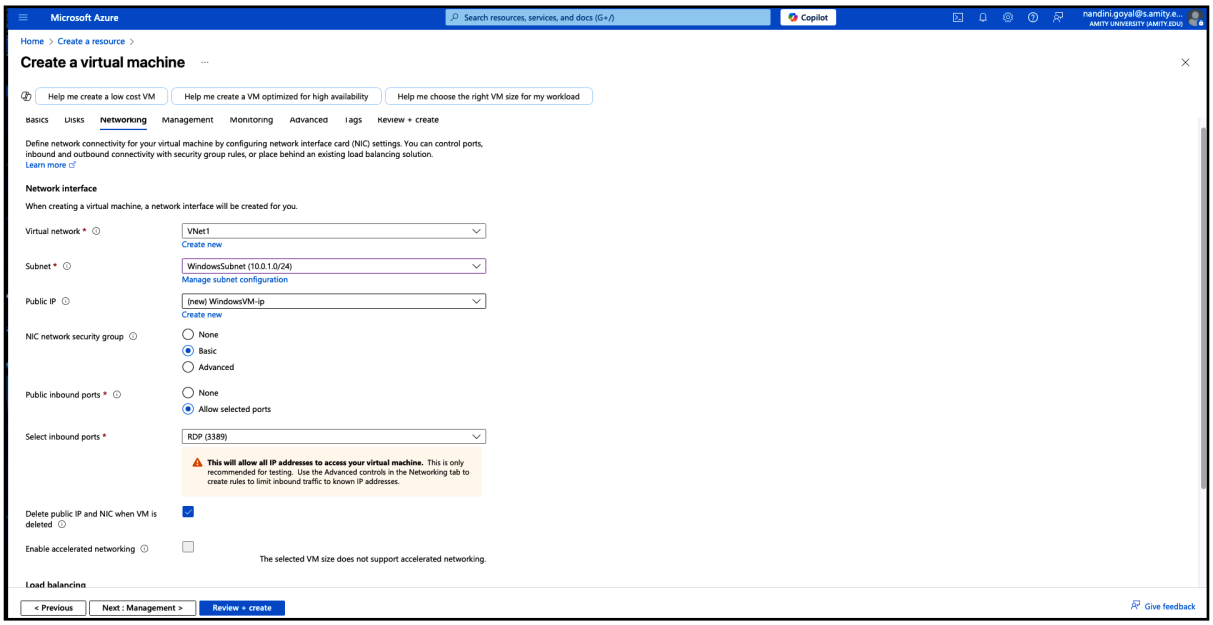

Figure 10: Networking tab with Windows Subnet

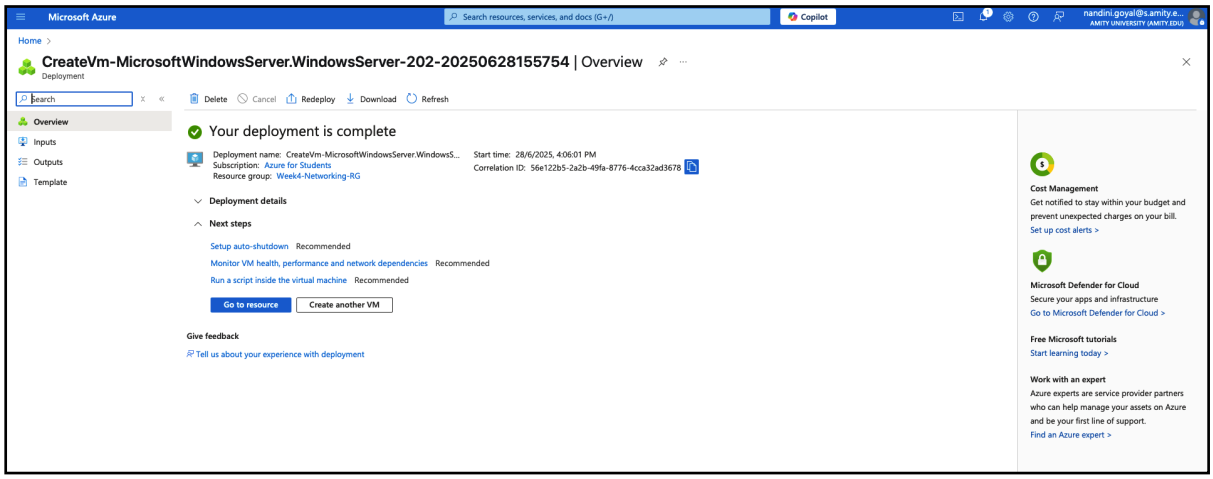

Figure 11: Deployment of Windows VM

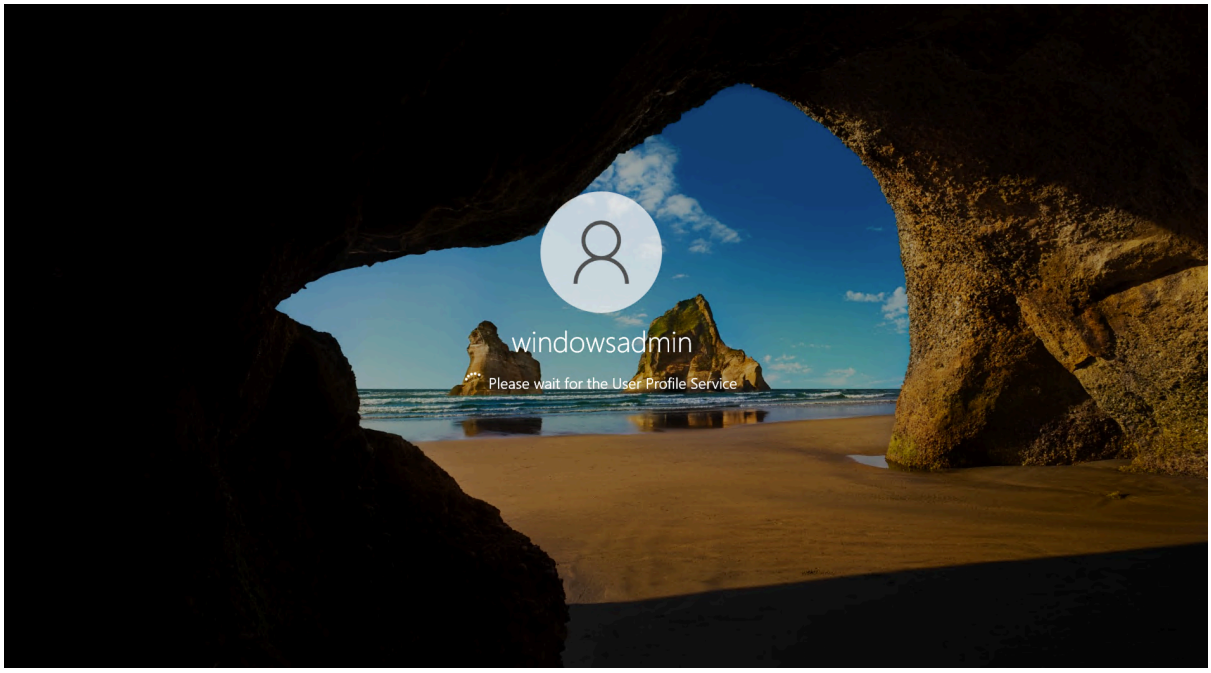

Figure 12 : Windows VM using RDP

6.3 Configure NSG and Inbound Rules

By default, Azure blocks ICMP (ping) traffic. So, we had to manually enable it in NSGs.

- **Add Inbound Rule in LinuxVM-nsg:**
  - Protocol: ICMP
  - Action: Allow
  - Priority: 100
  - Name: AllowPing
- **Add Inbound Rule in WindowsVM-nsg:**
  - Same as above
  - We first added an inbound rule for ICMP in the **Network Security Group** attached to the Windows VM, to allow ping requests.

Steps:

Go to **Networking** section of WindowsVM.

Click on **WindowsVM-nsg → Inbound Security Rules → Add**.

Configure as below:

A. **Source**: Any

B. **Source Port Ranges**: * (leave default)

C. **Destination**: Any

D. **Service**: Custom

E. **Protocol**: ICMP

F. **Action**: Allow

G. **Priority**: 101

H. **Name**: AllowPing

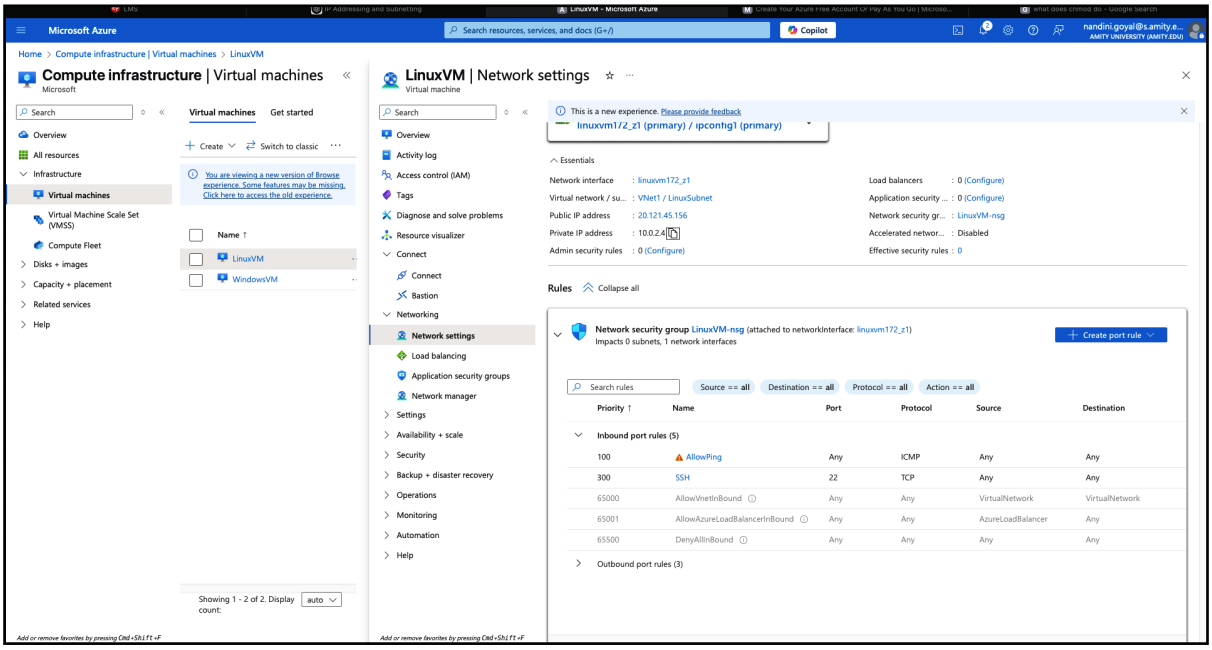

Figure 13: Linux NSG Rule for ICMP

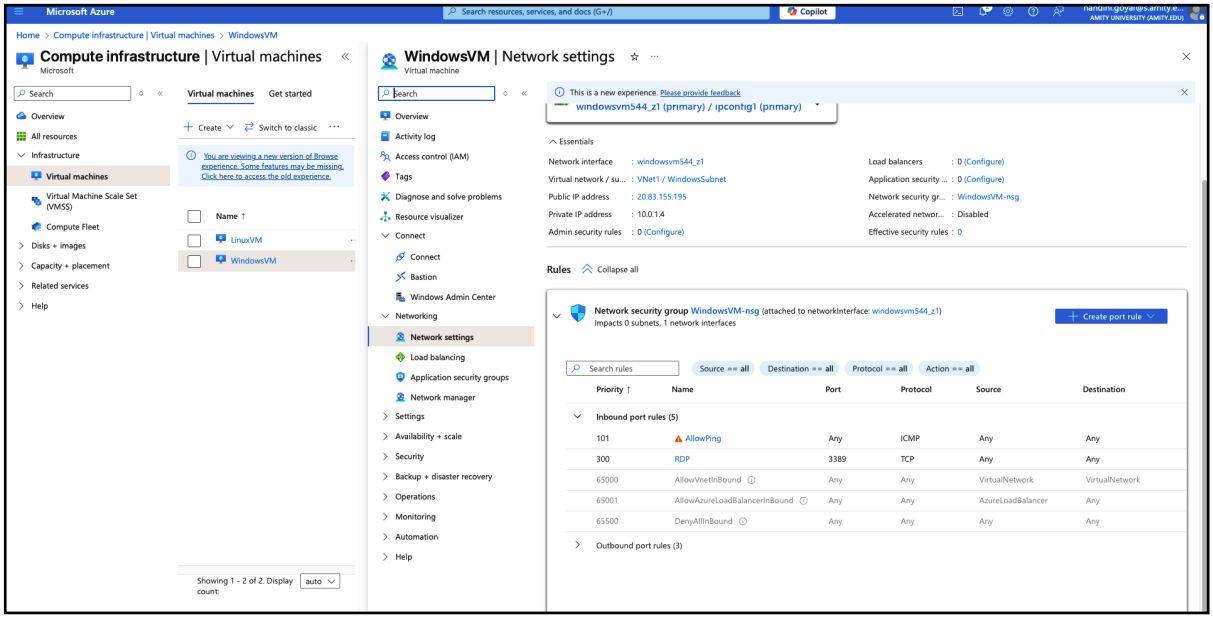

Figure 14: Windows NSG Rule for ICMP

- **Enable ICMP in Windows Firewall (Troubleshooting Step)**

Even after adding the NSG rule in Azure, sometimes the Windows VM may still **not respond to ping** from other machines. That's because **Windows Defender Firewall blocks ICMP (ping) traffic** by default.

You can allow it in **two ways**: using the GUI or using a command.

- **Using Windows Defender Firewall GUI (Manual)**

1. Inside the Windows VM, press **Win + R**, type:
2.

   wf.msc

3.
4. and press **Enter**.
5. This opens **Windows Defender Firewall with Advanced Security**.
6. In the left pane, click on **Inbound Rules**.
7. In the right pane, click **New Rule**.
8. Choose **Custom**, then click **Next**.
9. In **Protocol and Ports**, choose **ICMPv4** from the drop-down → click **Next**.
10. Leave **Scope** as default → click **Next**.
11. Leave **Action** as **Allow the connection** → click **Next**.
12. Apply the rule to **Domain, Private, and Public** → click **Next**.
13. Give the rule a name like:

    Allow ICMP Inbound → click **Finish**.

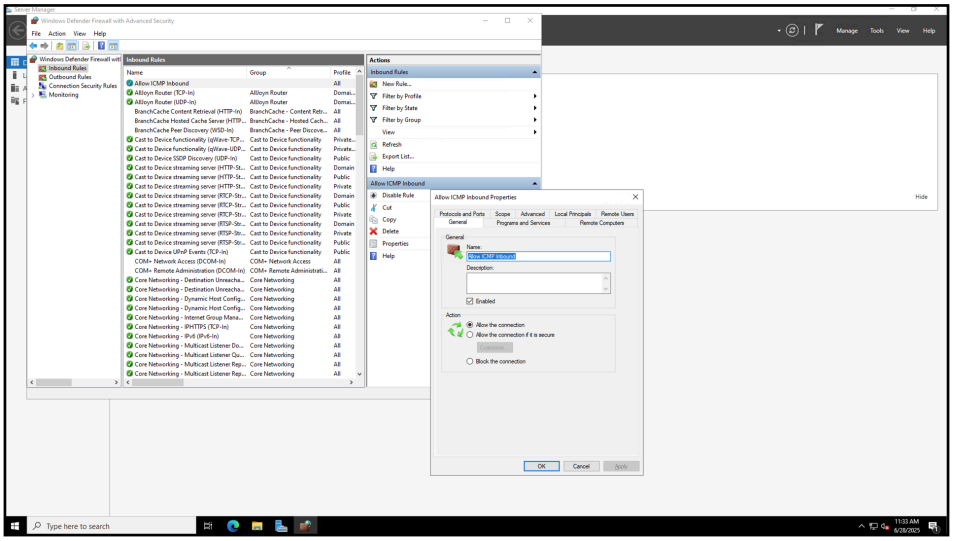

Figure 15: Custom Rule for ICMPv4 in Windows

6.4 Enable and Test Ping

- **Find Private IPs from VM Overview page. Example:**
  - Linux: 10.0.1.4
  - Windows: 10.0.2.4
- **From Linux VM:**

bash

ping 10.0.2.4

- **From Windows VM:**

cmd

ping 10.0.1.4

Figure 16: Ping from Linux to Windows

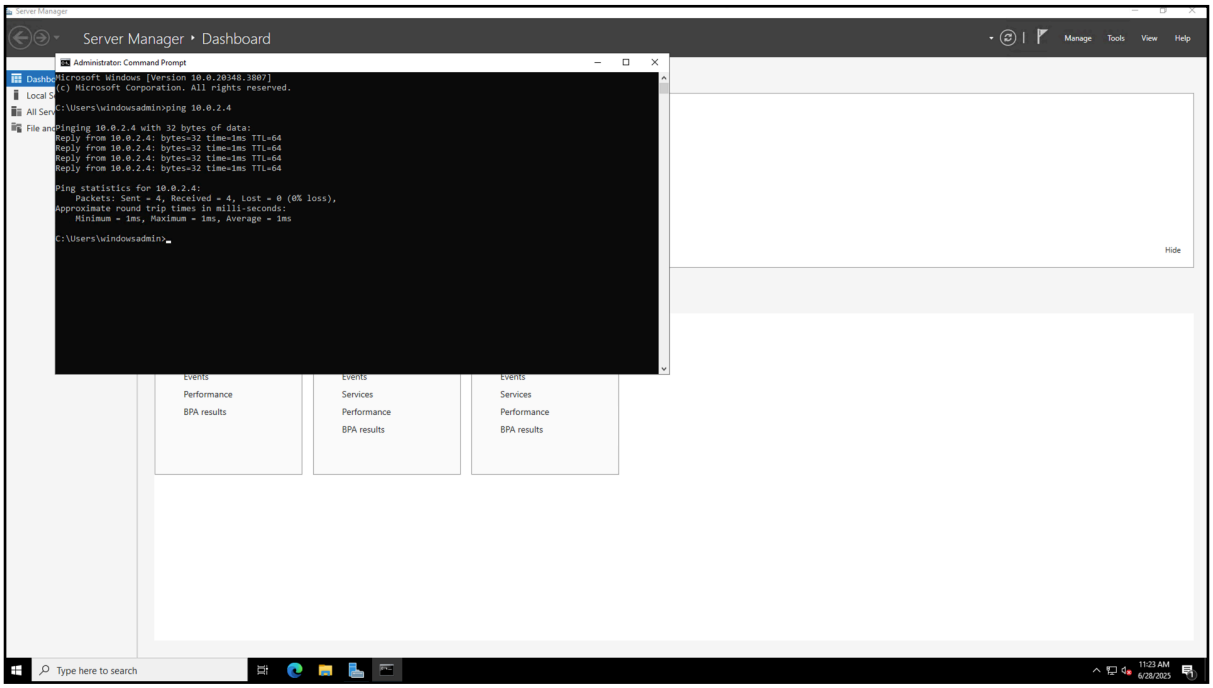

Figure 17: Ping from Windows to Linux

6.5 Create a Second VNet (VNet2)

- VNet2 → Address space: 10.1.0.0/16
- Subnet: LinuxSubnet2 → 10.1.1.0/24
- Launch another Linux VM (LinuxVM2) in this subnet

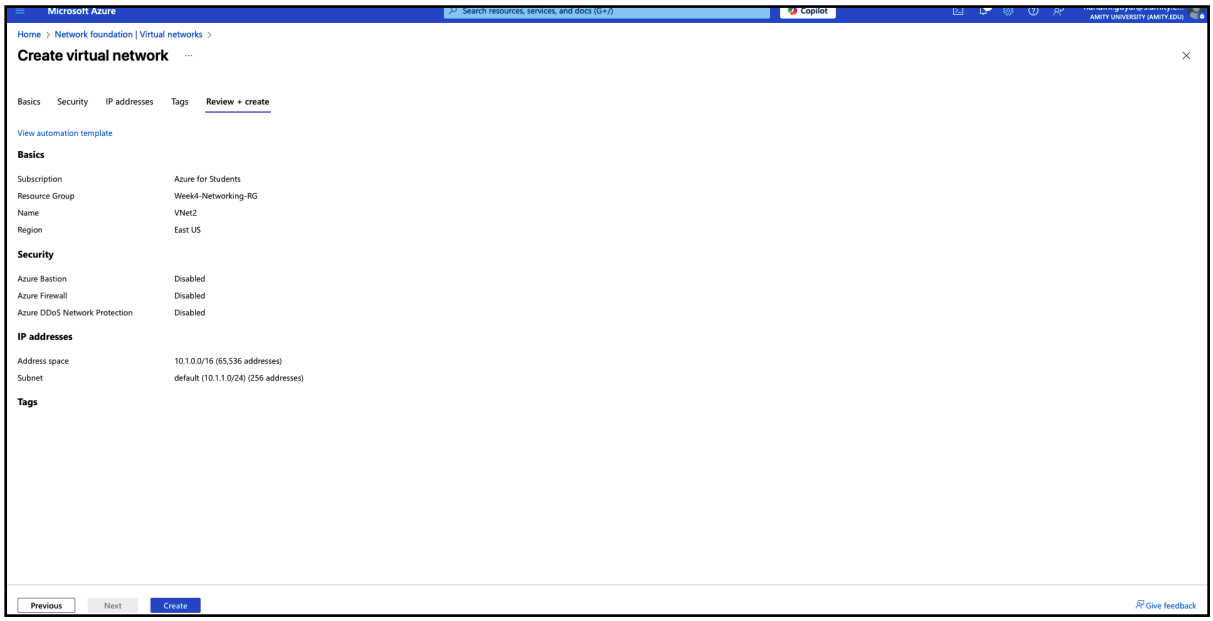

Figure 18: Setting up VNet2

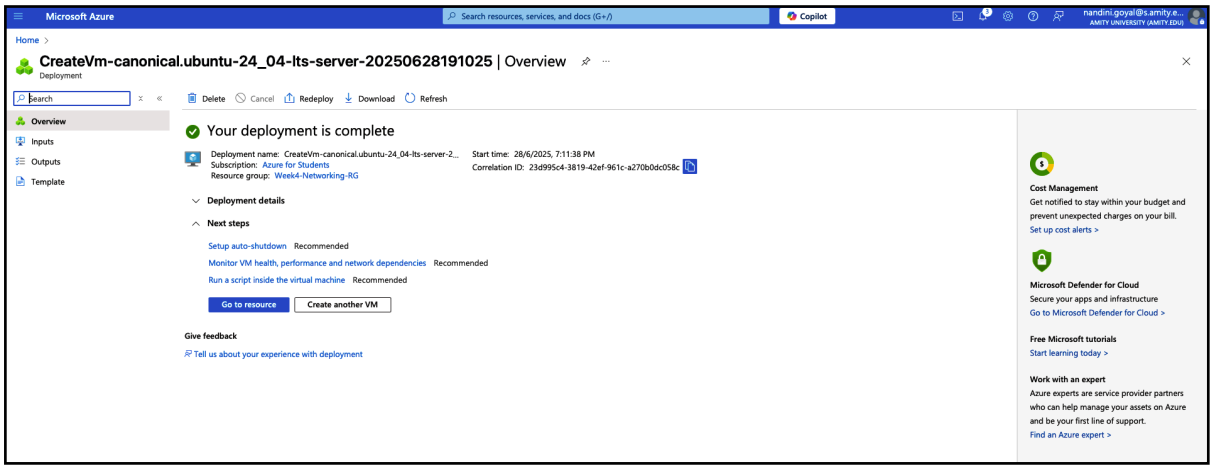

Figure 19: Deployed VNet2

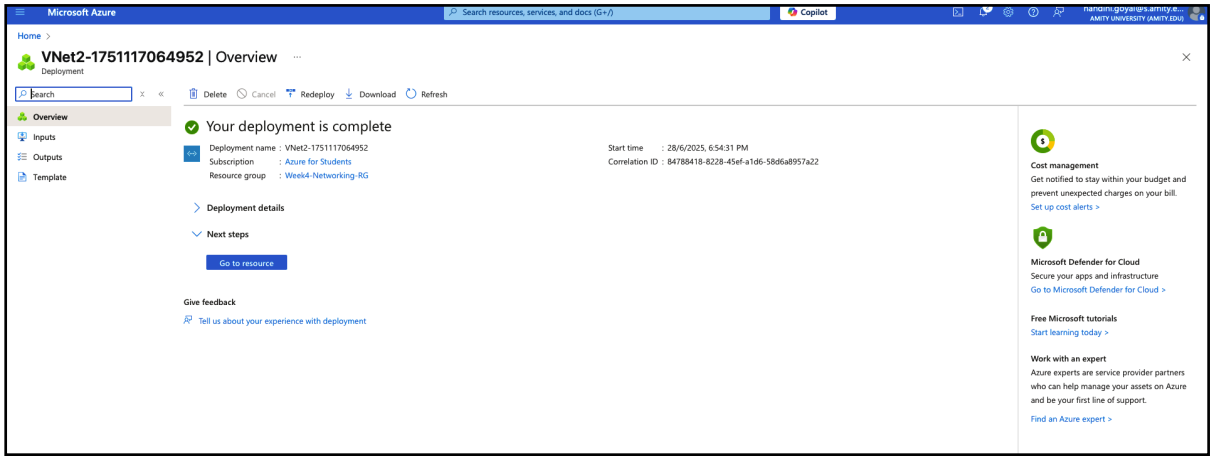

Figure 20: Launched Linux VM2 in VNet2

6.6 VNet Peering Between VNet1 and VNet2

1. Go to **VNet1** → Peerings → Add
2. Peering name: VNet1ToVNet2
3. Select VNet2 in Remote
4. Allow traffic in both directions

Repeat steps for **VNet2 to VNet1** if not done automatically.

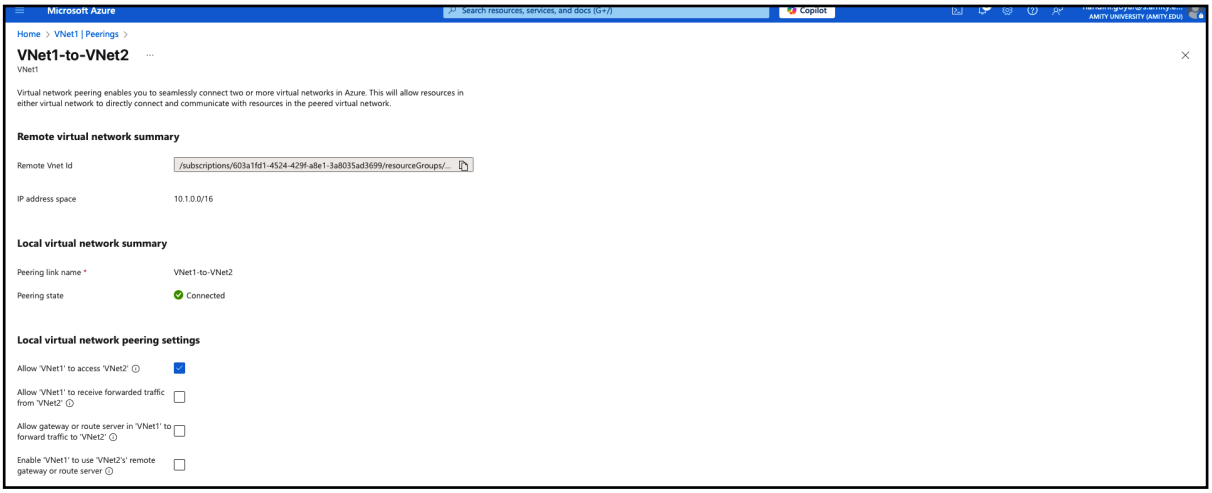

Figure 21: VNet1 peering configuration

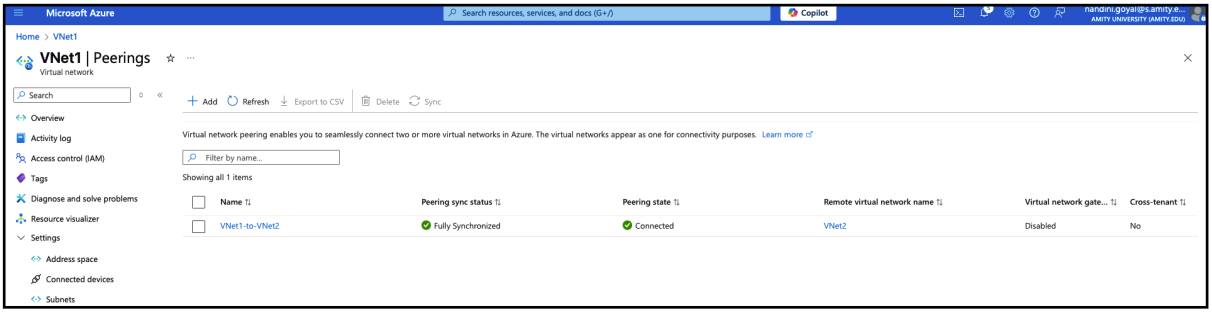

Figure 22: VNet1 to VNet2 peering

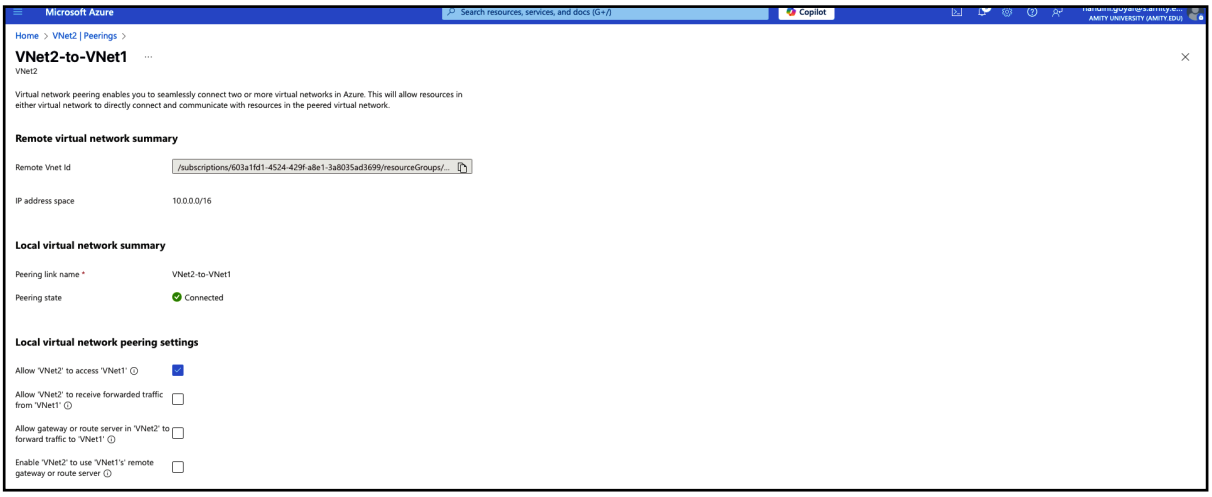

Figure 23: VNet2 peering configuration

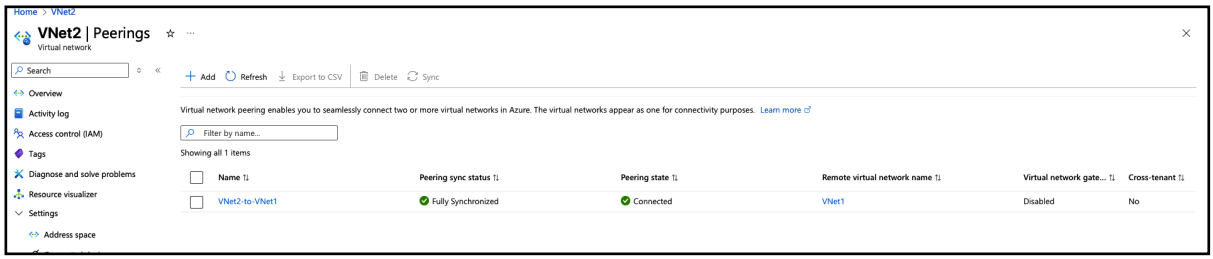

Figure 24: VNet2 to VNet1 peering

6.7 Test Cross-VNet Communication

Ping from LinuxVM1 to LinuxVM2 and vice versa using their **private IPs**.

- **Private ID of LinuxVM1:** 10.0.2.4
- **Private ID of LinuxVM2:** 10.1.1.4

```
linuxadmin@LinuxVM:~$ ping 10.1.1.4
PING 10.1.1.4 (10.1.1.4) 56(84) bytes of data.
64 bytes from 10.1.1.4: icmp_seq=1 ttl=64 time=1.79 ms
64 bytes from 10.1.1.4: icmp_seq=2 ttl=64 time=0.994 ms
64 bytes from 10.1.1.4: icmp_seq=3 ttl=64 time=1.30 ms
64 bytes from 10.1.1.4: icmp_seq=4 ttl=64 time=1.17 ms
64 bytes from 10.1.1.4: icmp_seq=5 ttl=64 time=1.22 ms
64 bytes from 10.1.1.4: icmp_seq=6 ttl=64 time=1.27 ms
64 bytes from 10.1.1.4: icmp_seq=7 ttl=64 time=1.15 ms
64 bytes from 10.1.1.4: icmp_seq=8 ttl=64 time=1.33 ms
^C
--- 10.1.1.4 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7010ms
rtt min/avg/max/mdev = 0.994/1.276/1.785/0.215 ms
linuxadmin@LinuxVM:~$
```

Figure 25: Pinging from Linux VM1 to Linux VM2

```
linuxadmin2@LinuxVM2:~$ ping 10.0.2.4
PING 10.0.2.4 (10.0.2.4) 56(84) bytes of data.
64 bytes from 10.0.2.4: icmp_seq=1 ttl=64 time=2.27 ms
64 bytes from 10.0.2.4: icmp_seq=2 ttl=64 time=0.979 ms
64 bytes from 10.0.2.4: icmp_seq=3 ttl=64 time=1.12 ms
64 bytes from 10.0.2.4: icmp_seq=4 ttl=64 time=2.53 ms
64 bytes from 10.0.2.4: icmp_seq=5 ttl=64 time=0.901 ms
64 bytes from 10.0.2.4: icmp_seq=6 ttl=64 time=1.36 ms
64 bytes from 10.0.2.4: icmp_seq=7 ttl=64 time=1.01 ms
64 bytes from 10.0.2.4: icmp_seq=8 ttl=64 time=1.14 ms
64 bytes from 10.0.2.4: icmp_seq=9 ttl=64 time=2.48 ms
64 bytes from 10.0.2.4: icmp_seq=10 ttl=64 time=1.99 ms
^C
--- 10.0.2.4 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9012ms
rtt min/avg/max/mdev = 0.901/1.578/2.534/0.630 ms
linuxadmin2@LinuxVM2:~$
```

Figure 26: Pinging from Linux VM2 to Linux VM1

7. Troubleshooting and Fixes

Issue	Fix
RDP not working	Checked NSG for port 3389, restarted VM
Ping failed	Allowed ICMP in NSGs, enabled Windows firewall rule
Subnet not visible	Manually added it later via portal
Peering error	Fixed by setting proper permissions both ways

8. Learnings

- Importance of CIDR ranges in cloud networking
- How to create and manage Azure VNets and subnets
- Control traffic using NSG and firewall rules
- Set up peering for private communication
- Debugging connectivity issues in cloud infrastructure

9. Conclusion

This R&D document provided an in-depth understanding and hands-on implementation of Azure Virtual Networking components, particularly focusing on:

- **CIDR ranges**, subnets, and IP addressing within a Virtual Network (VNet),
- **VNet Peering**, its types, and how it allows inter-VNet communication,
- Creating and configuring a **Linux and Windows VM**, placing them in **separate subnets**,
- Ensuring **bidirectional ping connectivity** through Network Security Group (NSG) rules and Windows Firewall settings,
- **Troubleshooting connectivity** issues using GUI and command-line techniques,
- Creating **two separate VNets** and establishing **VNet Peering** to enable seamless communication.

10. Reference

- **Microsoft Learn Documentation**
  https://learn.microsoft.com/en-us/azure/virtual-network/

- **Azure CIDR IP Addressing Guide**
  https://learn.microsoft.com/en-us/azure/virtual-network/virtual-networks-ip-addressing

- **VNet Peering in Azure**
  https://learn.microsoft.com/en-us/azure/virtual-network/virtual-network-peering-overview
- **Create and Manage Subnets in Azure**
  https://learn.microsoft.com/en-us/azure/virtual-network/manage-subnets
- **Enable Ping (ICMP) in Windows VM**
  https://learn.microsoft.com/en-us/answers/questions/225769/enable-ping-on-azure-vm
- **Azure Network Security Groups (NSGs)**
  https://learn.microsoft.com/en-us/azure/virtual-network/network-security-groups-overview
- **Azure VM Pricing Calculator**
  https://azure.microsoft.com/en-in/pricing/calculator/
- **Azure for Students Subscription Details**
  https://azure.microsoft.com/en-in/free/students/