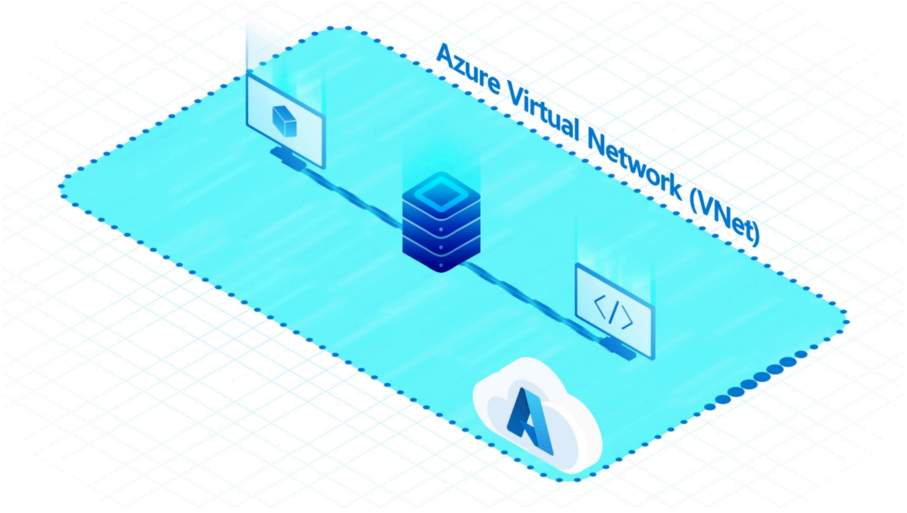
CIDR Ranges of a VNet, Subnet & VNet Peering

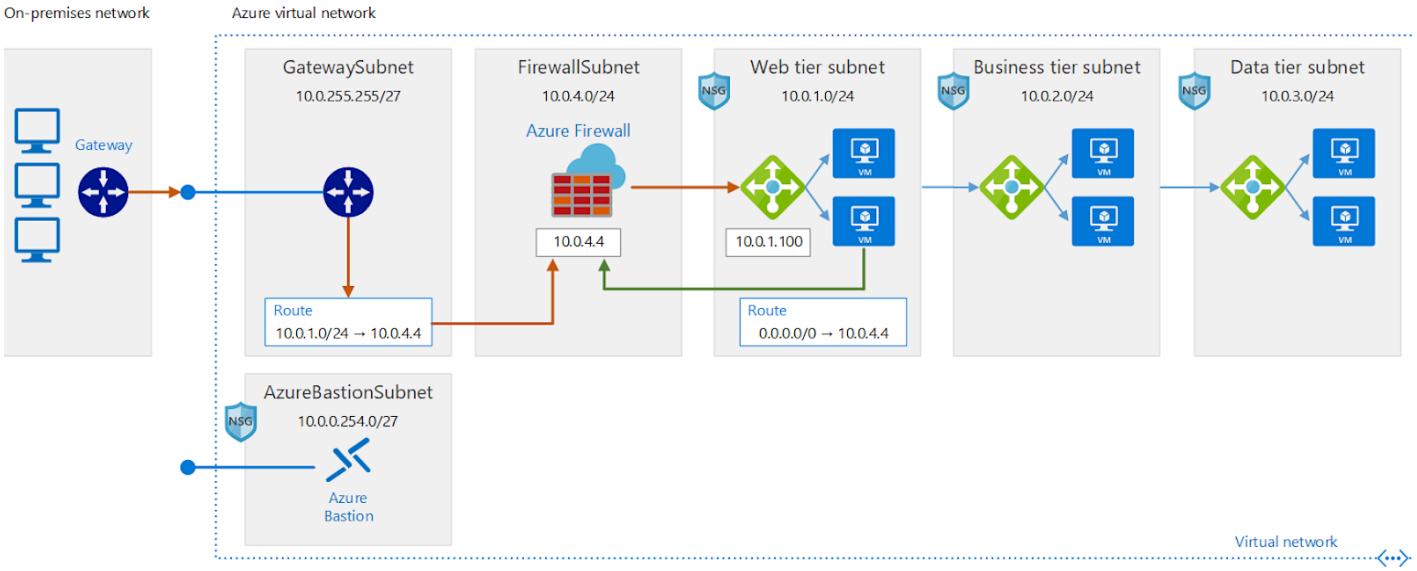
1. Overview

In the Azure ecosystem, **Virtual Networks (VNets)** represent the core infrastructure layer that enables cloud-based applications and services to communicate securely and efficiently. VNets act as a logical, isolated section of the Azure cloud, scoped to a specific subscription and region, where resources such as Virtual Machines (VMs), Application Gateways, Load Balancers, Databases, and more can be deployed and connected.

A VNet provides the same networking functionalities found in an on-premises network, such as IP addressing, name resolution, routing, and security, but with the elasticity, scalability, and cost efficiency of the cloud. With VNets, organizations can build highly customizable network topologies tailored to specific workloads, while maintaining complete control over traffic flow and security boundaries.

VNets are pivotal in designing hybrid and multi-cloud architectures. They support VPN Gateways and ExpressRoute connections to securely bridge on-premises environments with cloud deployments. Moreover, VNets can be interconnected using VNet Peering, enabling a mesh of secure, high-performance communication channels between isolated environments or across different regions.





2. CIDR Ranges in VNets and Subnets

2.1 What is CIDR?

CIDR (Classless Inter-Domain Routing) is a notation method for IP address allocation and routing. It replaces the older classful addressing scheme by allowing **variable-length subnet masks**. CIDR notation uses the format: `<IP_address>/<prefix_length>`

`Example: 10.0.0.0/16`

2.2 CIDR in Azure VNets

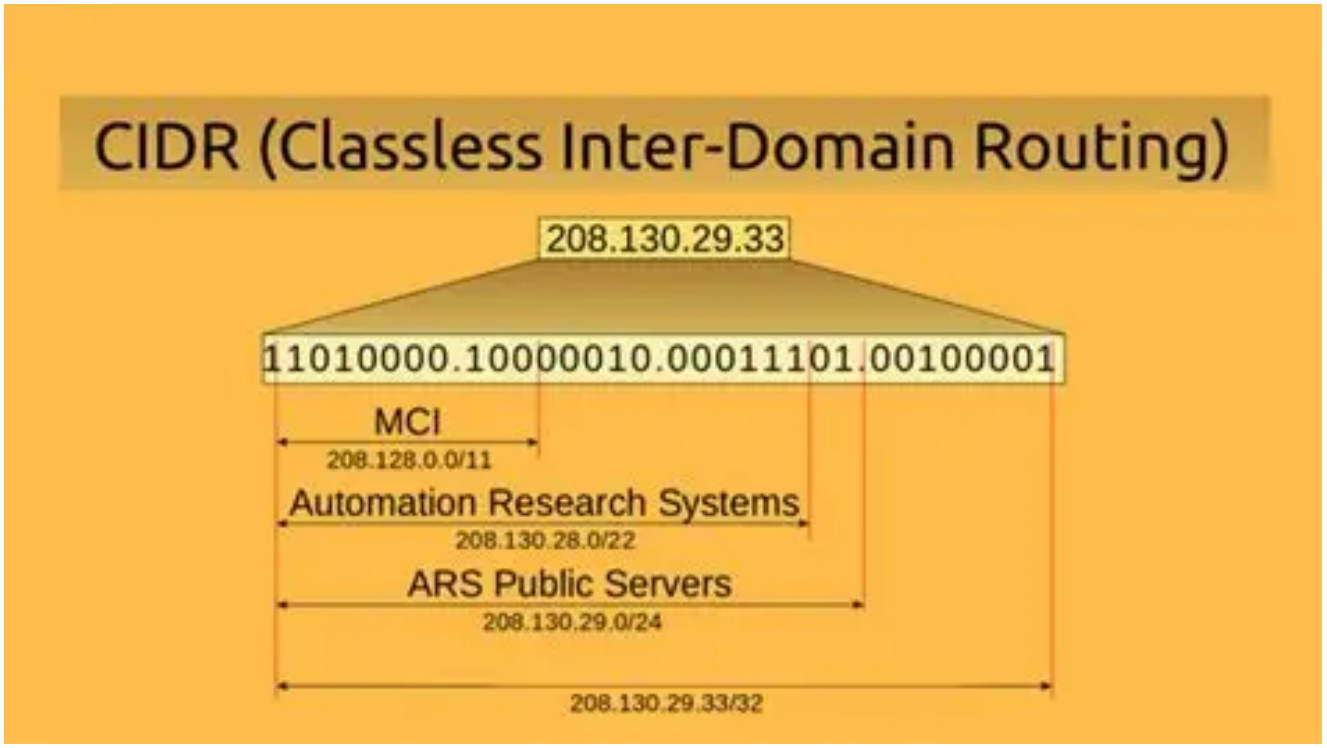
- VNets in Azure **must have a defined IP address space** in CIDR format.
- CIDR blocks can range from `/8` to `/29`.
- A VNet can have multiple **non-overlapping** address spaces (useful for peering later).
- The address space should be chosen with future subnetting and peering in mind.

2.3 CIDR in Azure Subnets

- Subnets are segments of a VNet's CIDR space.
- Each subnet requires a unique non-overlapping CIDR block within the VNet.
- Azure reserves the first four and the last IP address of each subnet, reducing usable IPs.

Example

VNet: `10.0.0.0/16`
Subnet 1: `10.0.1.0/24`
Subnet 2: `10.0.2.0/24`



3. Subnet Architecture and Purpose

3.1 What is a Subnet?

A **Subnet**, short for *subnetwork*, is a logically defined segment of an Azure Virtual Network (VNet) that helps in organizing, isolating, and securing resources. Subnetting enables fine-grained control over traffic routing, security, and IP address management.

In Azure, a subnet is defined by a **CIDR block** that is a subset of the parent VNet's address space. All resources deployed within a subnet receive an IP address from that range.

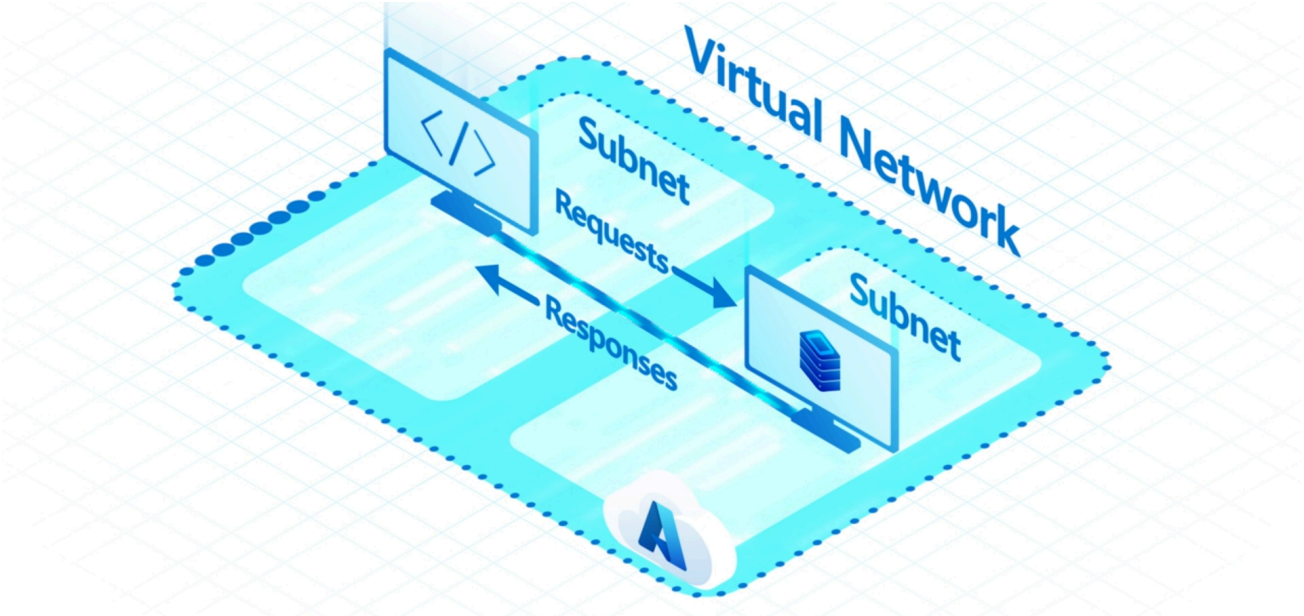
Key Capabilities of Subnets:

- **Logical Isolation:** Each subnet can represent a unique tier (e.g., web, app, database) or workload type, enhancing clarity and manageability.
- **Security Enforcement:** Apply Network Security Groups (NSGs) to subnets to control inbound and outbound traffic at Layer 3/4.
- **Custom Routing:** Attach User-Defined Routes (UDRs) to subnets for traffic redirection, inspection, or firewall enforcement.
- **Policy Enforcement:** Deploy Azure Policies at the subnet level to enforce resource tagging, allowed VM sizes, etc.
- **Network Virtual Appliances (NVAs):** Subnets can host custom routing devices for firewall, NAT, or WAN optimization.

3.2 Types of Subnets

While Azure doesn't classify subnets explicitly, they can be designed based on purpose:

- **Front-end subnet:** Hosts web servers or public-facing apps.
- **Back-end subnet:** Contains databases or internal services.
- **Gateway subnet:** Required for VPN Gateway or ExpressRoute connections.
- **Bastion subnet:** Required for Azure Bastion service.



4. VNet Peering

Virtual Network (VNet) Peering in Azure enables seamless connectivity between two VNets, allowing virtual machines and services in each network to communicate over private IP addresses. This is achieved through Azure's high-speed, low-latency backbone network, not the public internet. Peering abstracts the complexity of VPN or gateway configurations and provides a straightforward method for enabling secure cross-network communication.

Peered VNets:

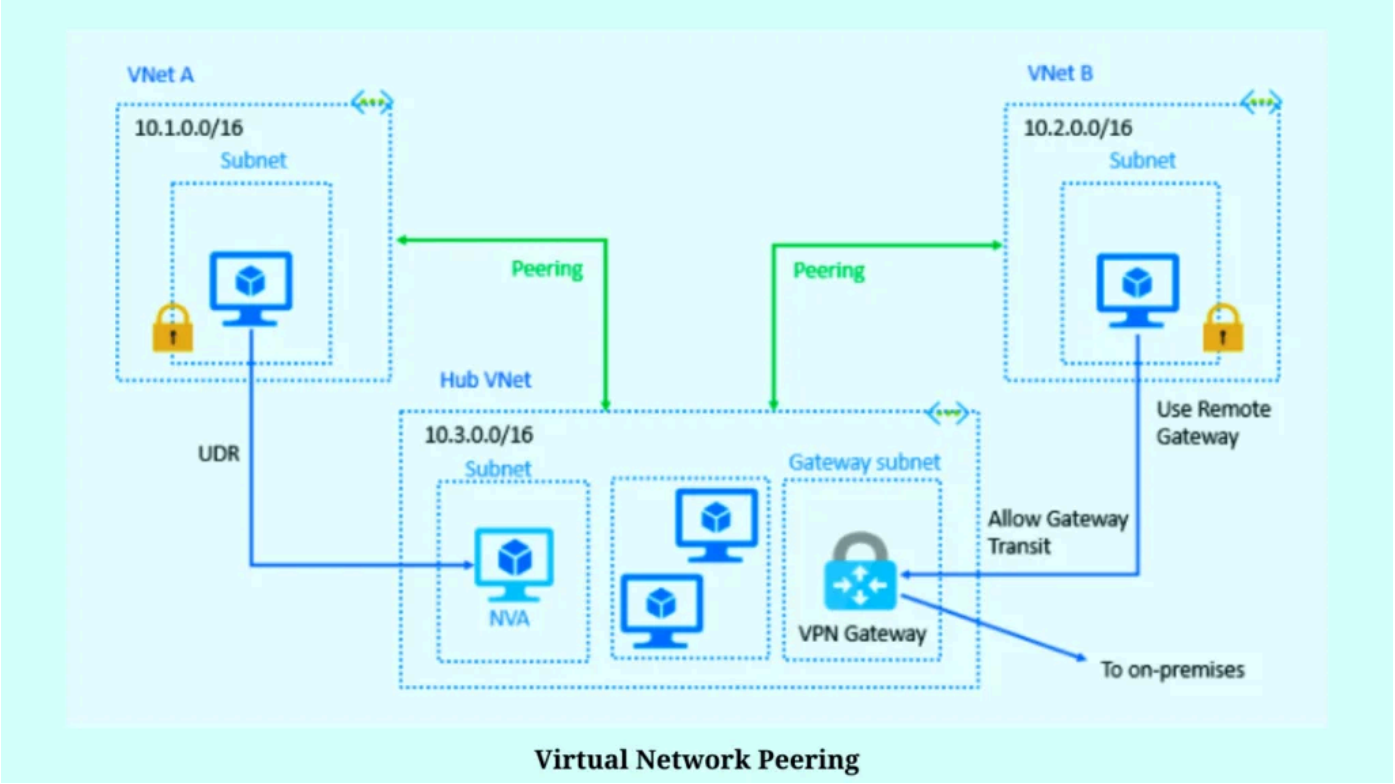
- Can be in **same or different regions**
- Must have **non-overlapping address spaces**
- Can **forward traffic** between them if explicitly configured

Types of VNet Peering

Type	Description
Intra-Region Peering	Connects two VNets in the **same Azure region**
Global VNet Peering	Connects VNets across **different regions**

Key Features

- **No public internet involvement:** Traffic stays on Azure's private backbone.
- **Low-latency & high-speed:** No bottlenecks due to NAT.
- **Transitive peering not supported:** You must manually peer each VNet that needs connectivity.
- **Network Security Group rules still apply:** Even if peered, explicit NSG rules are needed for communication.



Virtual Network Peering

5. Use Case: Deploying VNets, Subnets, and VMs with Peering for Cross-Network Communication

Objective Overview

To design and implement a secure, logically segmented Azure network infrastructure using Virtual Networks (VNets) and Subnets to simulate a real-world enterprise network topology. The goal is to deploy Linux-based Virtual Machines (VMs) in isolated subnets within a VNet, ensuring they can communicate with each other internally via private IPs. Additionally, establish VNet Peering between two separate VNets to enable cross-network communication without exposing resources to the public internet. This setup aims to demonstrate secure, scalable, and private intra-network and inter-network connectivity in Azure.

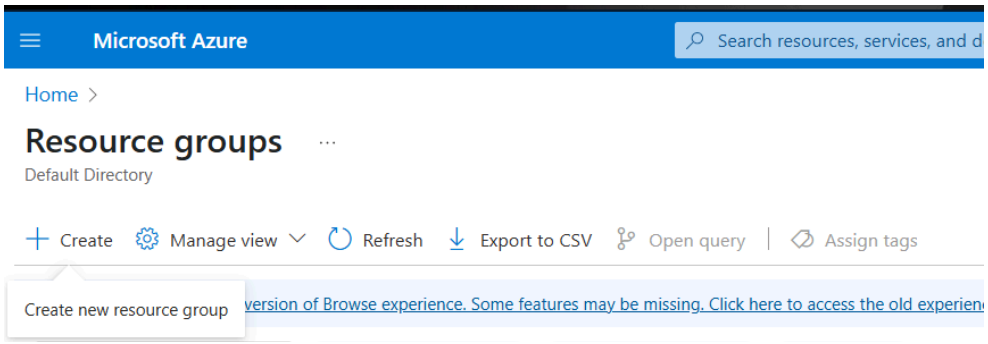
Specifically, we will:

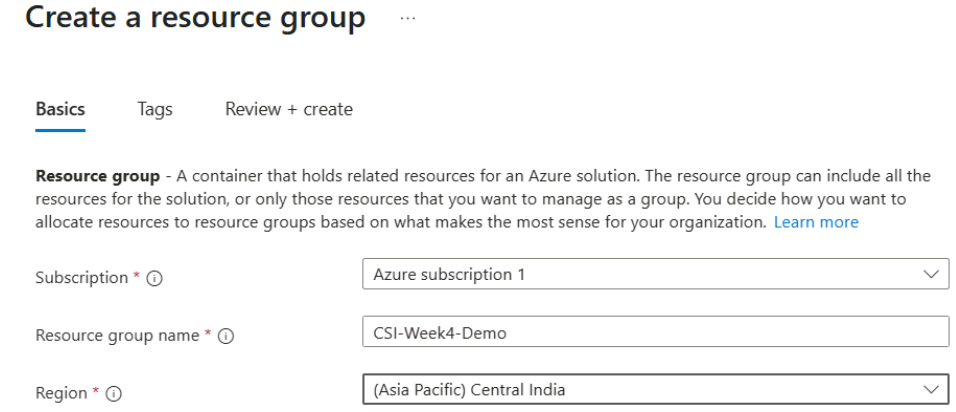
- Create a Virtual Network (VNet1) with two subnets.
- Launch one Linux VM in Subnet-1 and one Windows VM in Subnet-2.
- Validate ping/ICMP connectivity between these VMs (within the same VNet).
- Create a second VNet (VNet2) in the same region with its own subnet and Linux VM.
- Establish VNet Peering between VNet1 and VNet2.
- Confirm cross-VNet communication by verifying ping between the VMs in both VNets.

Step-by-Step Implementation on Azure Portal

Step 1: Create a Resource Group

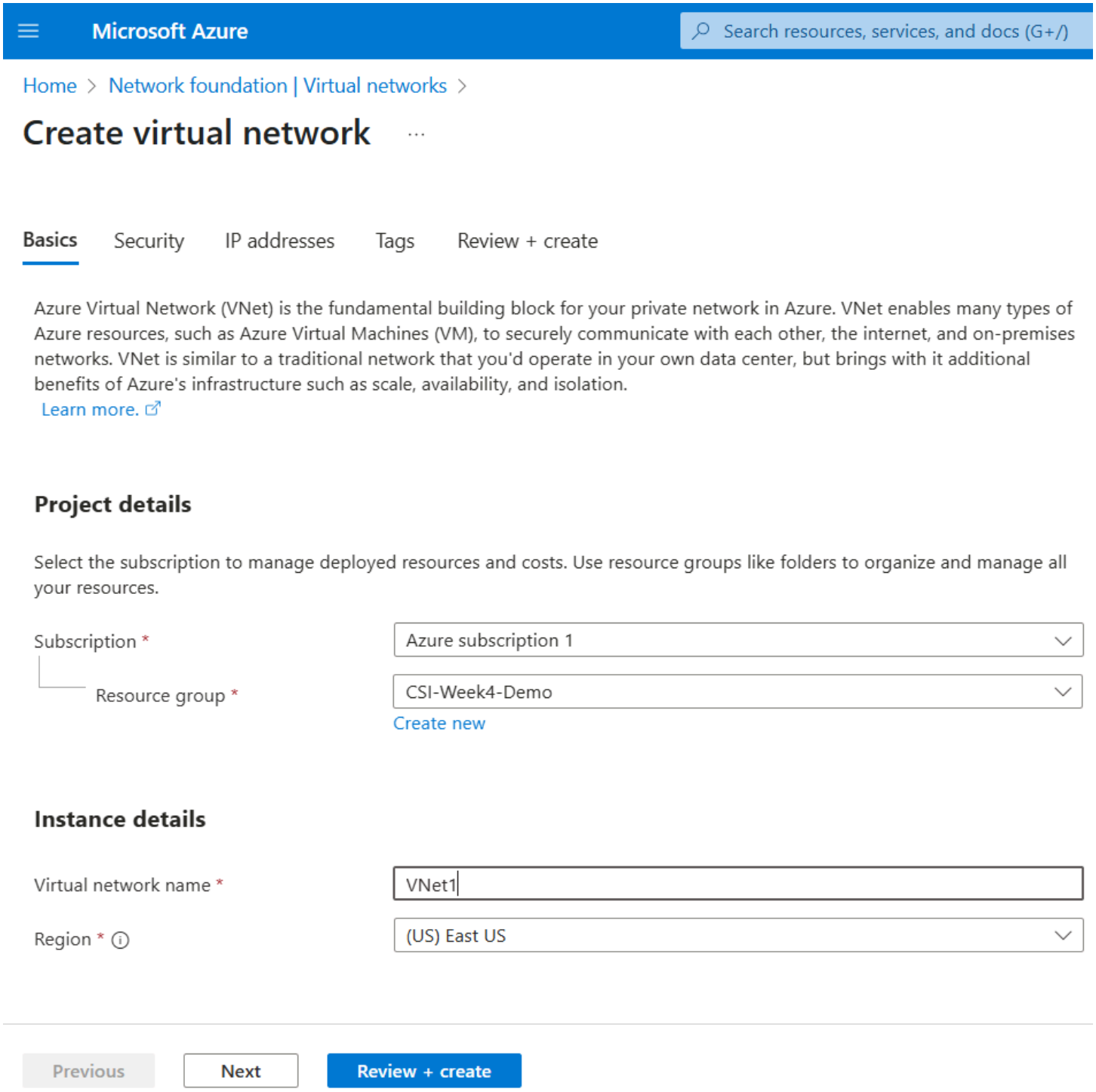
- Go to **Azure Portal > Resource Groups > Create**
- Provide a name and region.
- Click **Review + Create > Create**

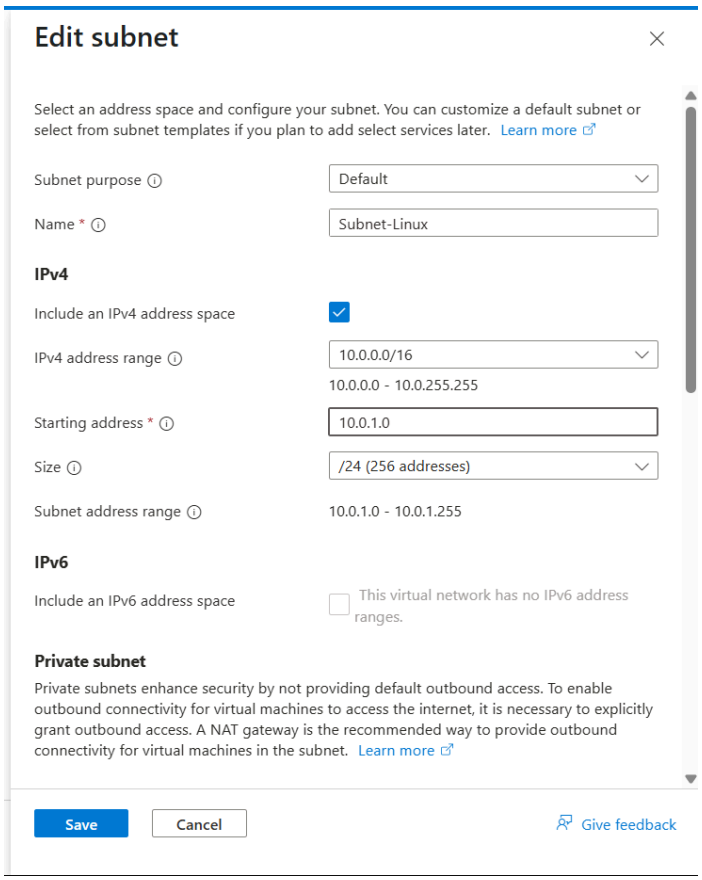


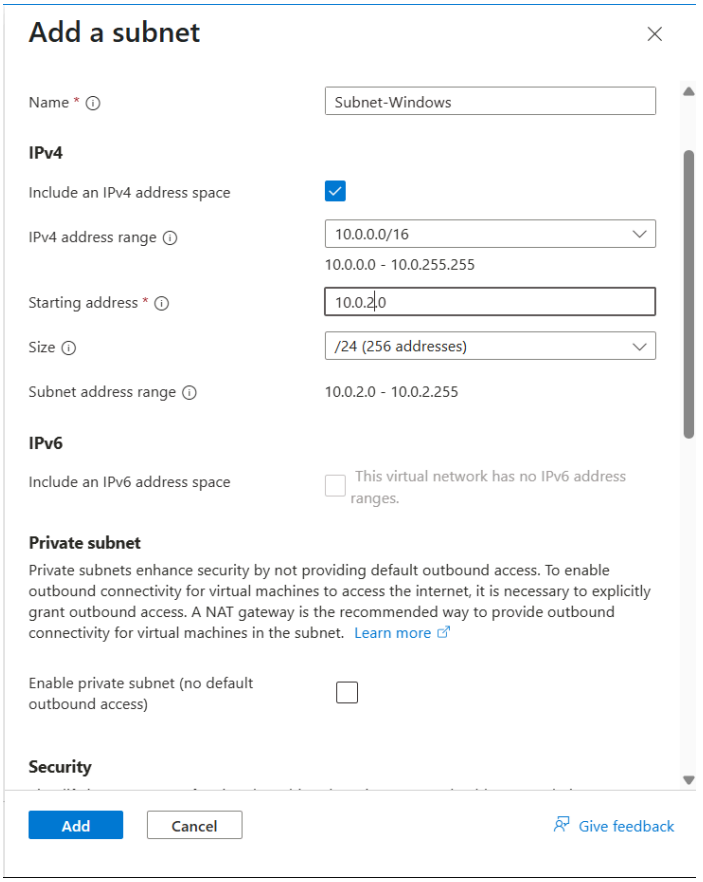


Step 2: Create VNet1 with Two Subnets

- Go to **Virtual Networks > Create**
- Define address space (e.g., `10.0.0.0/16`)
- Add two subnets:
  - Subnet-Linux (`10.0.1.0/24`)
  - Subnet-Windows (`10.0.2.0/24`)
- Click **Review + Create > Create**

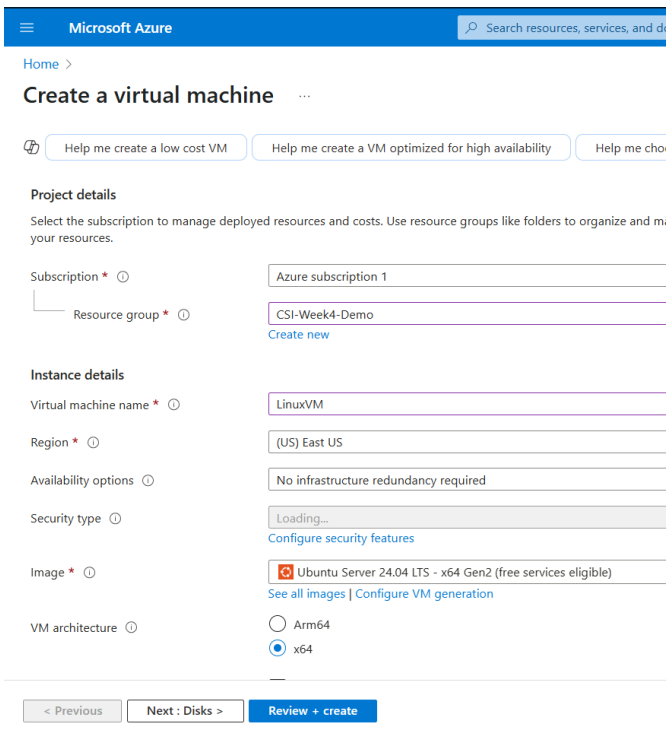


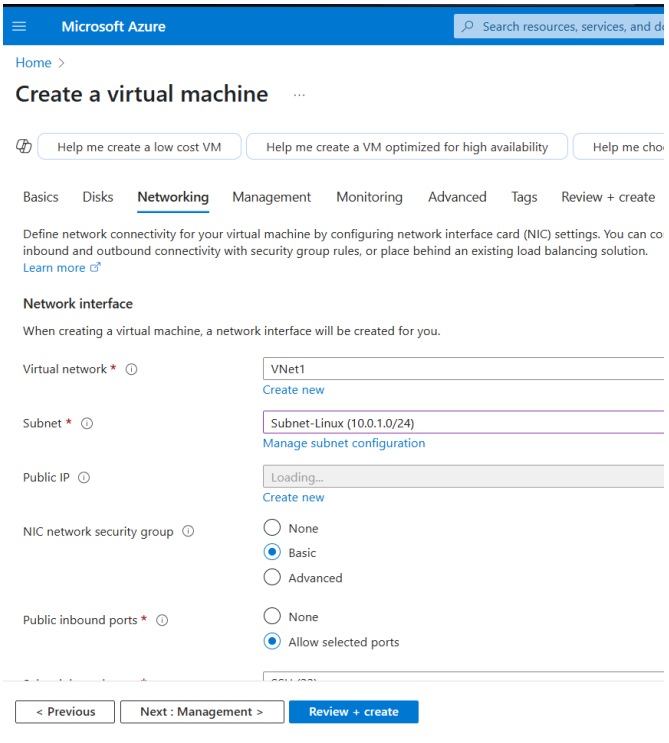




Step 3: Launch Linux VM in Subnet-Linux

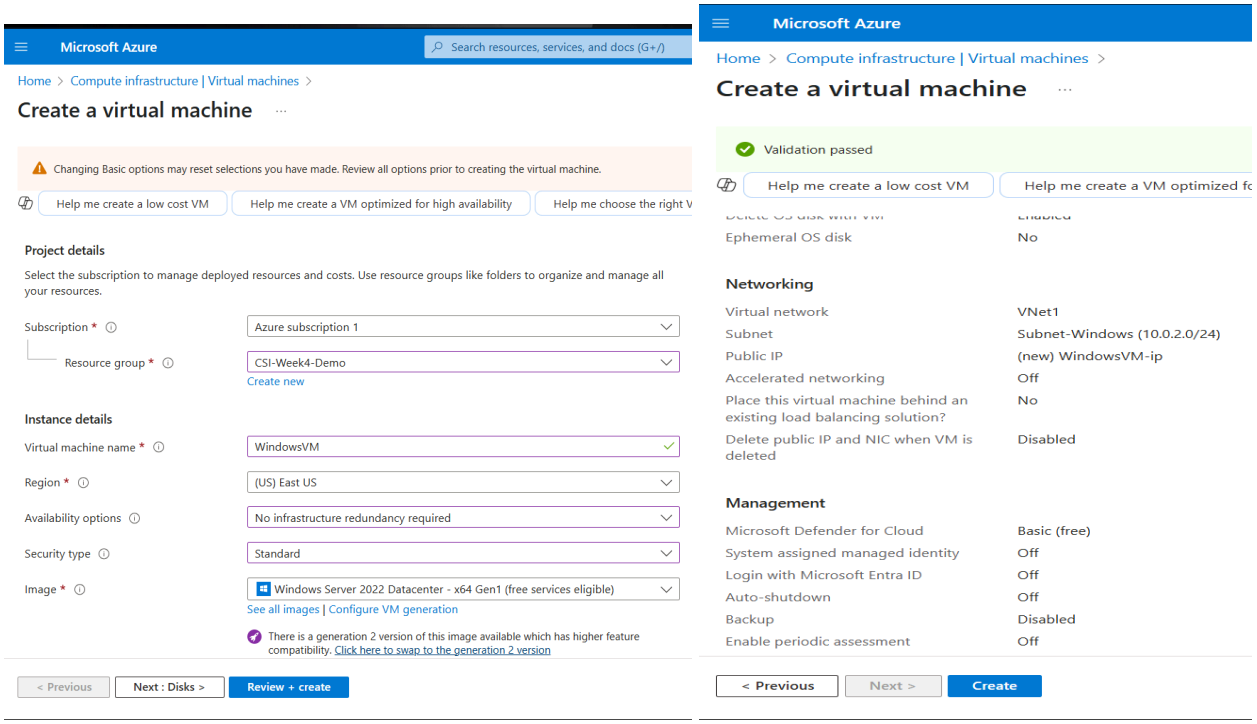
- Go to **Virtual Machines > Create**
- Select Ubuntu image and place it in **Subnet-Linux** of **VNet1**
- Enable SSH access
- Deploy the VM





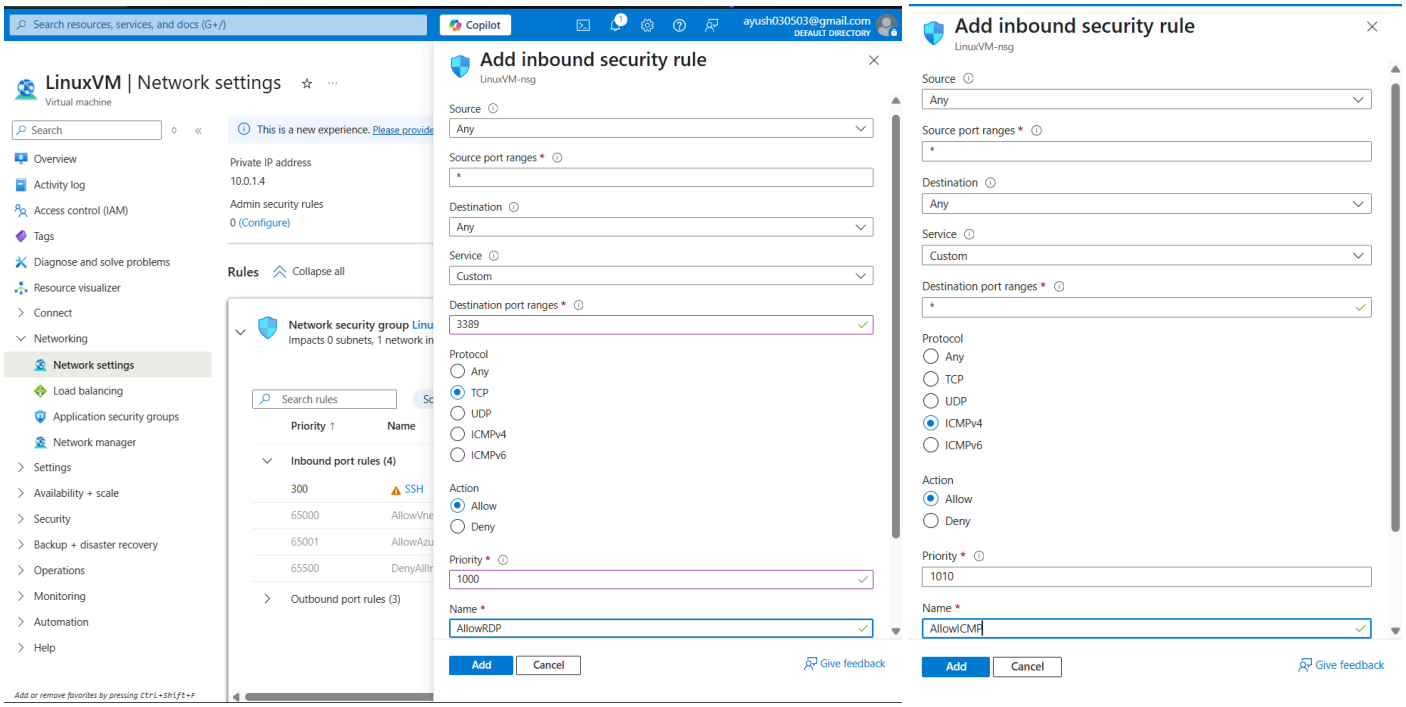
Step 4: Launch Windows VM in Subnet-Windows

- Create a new VM using Windows Server image
- Place it in **Subnet-Windows** of **VNet1**
- Enable RDP access
- Add an NSG rule to allow **ICMP (ping)**



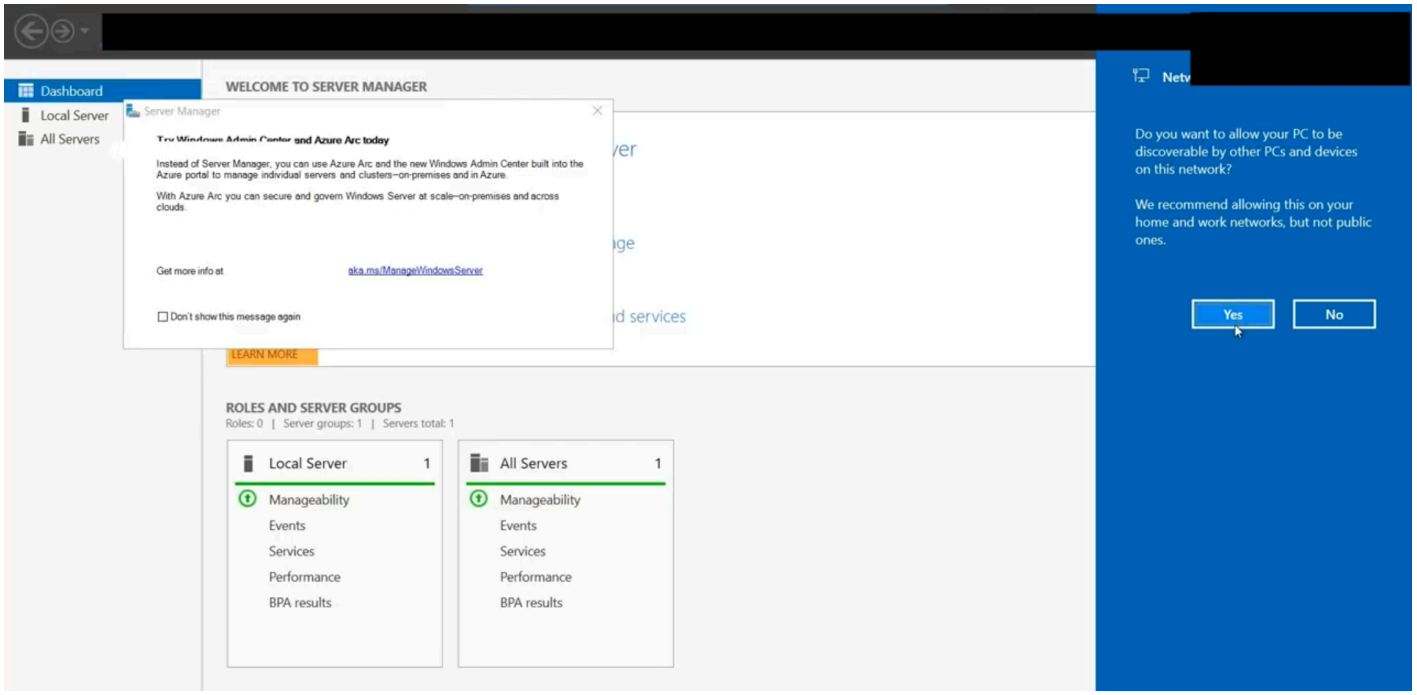
Step 5: Enable ICMP (Ping) Between VMs

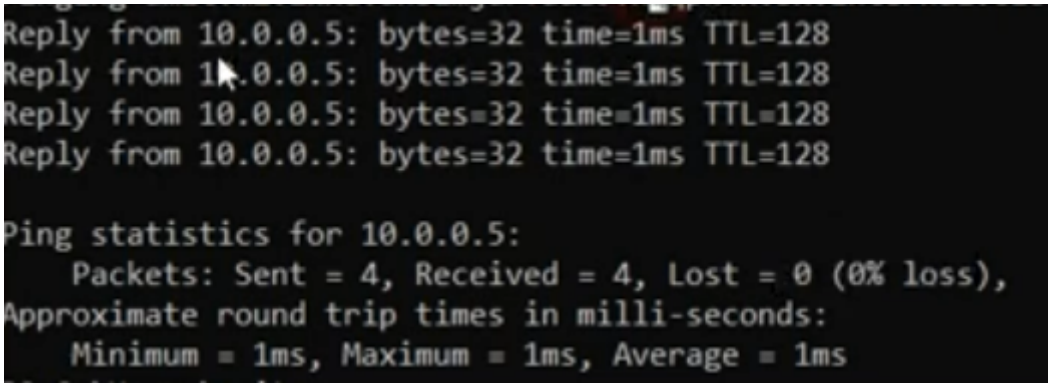
- Add custom **inbound NSG rule** to both VMs:
  - Protocol: ICMP
  - Source: Any
  - Action: Allow



Step 6: Test Intra-VNet Communication

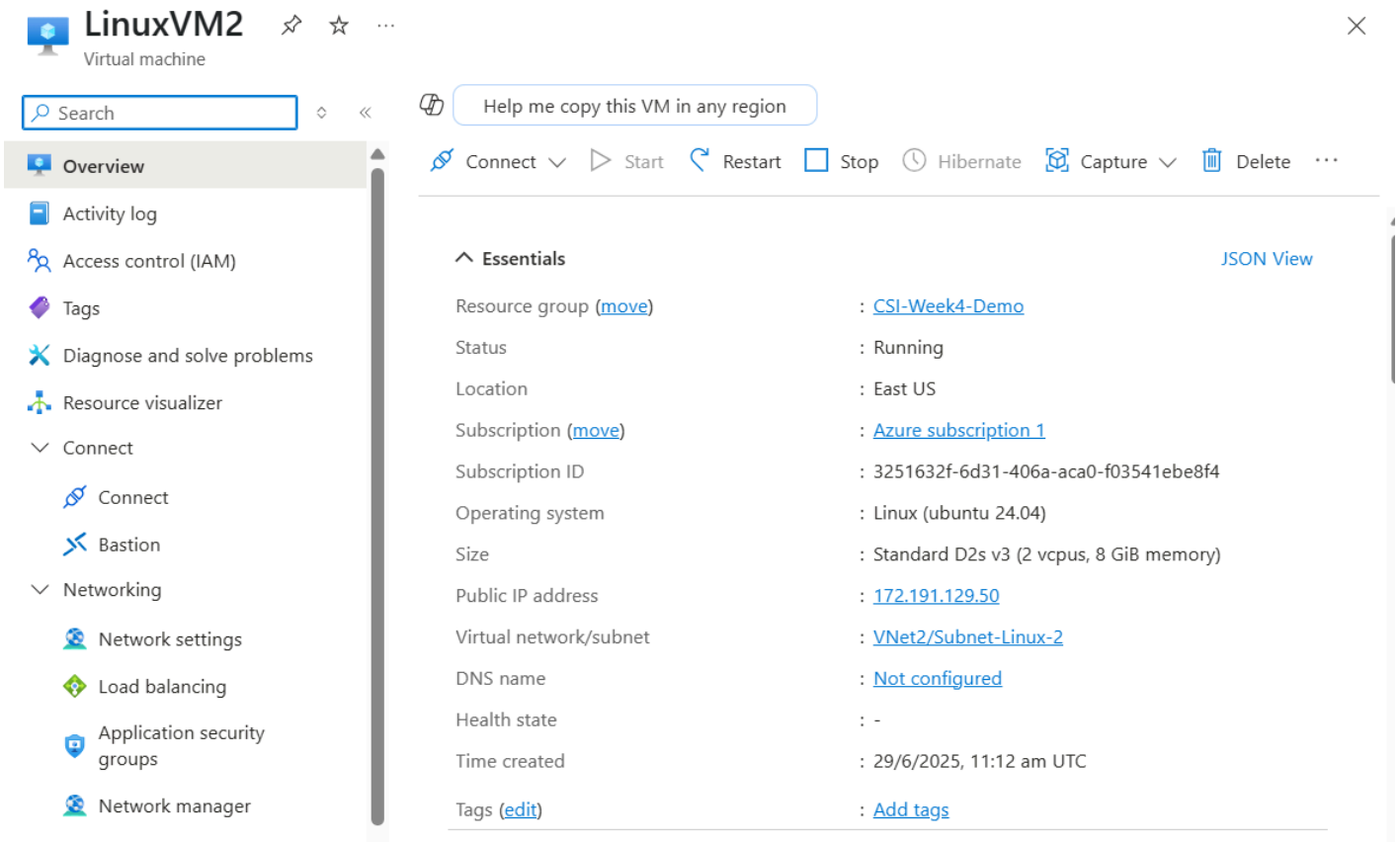
- SSH into Linux VM and ping Windows VM, use `ping <WindowsVM Private IP>`
- Or RDP into Windows VM and ping Linux VM
- Confirm successful ping response





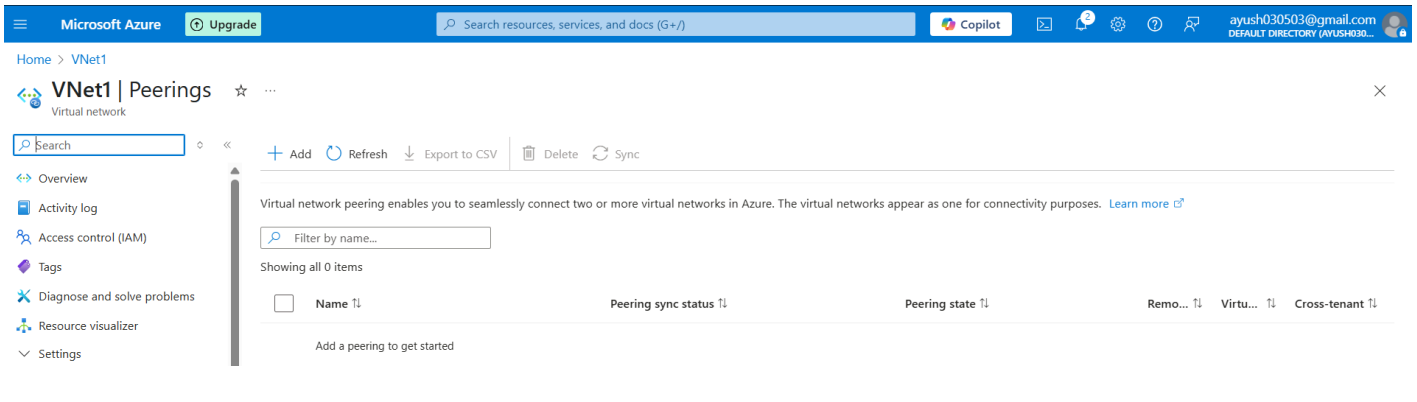
Step 7: Create VNet2 and a Linux VM

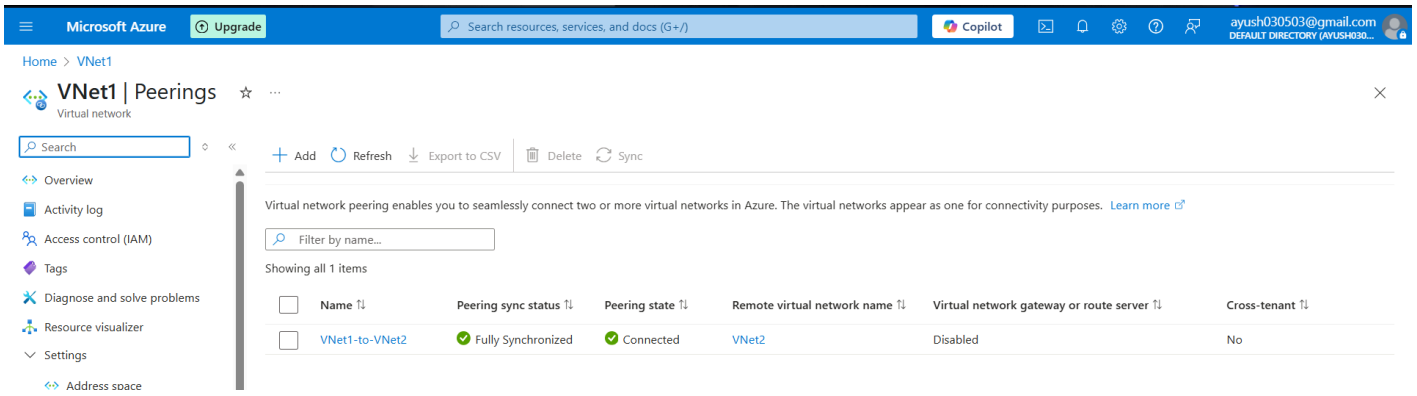
- Create **VNet2** with a non-overlapping address space (e.g., `10.1.0.0/16`)
- Add a subnet (e.g., `10.1.1.0/24`)
- Launch a **Linux VM** in this subnet
- Enable SSH and ICMP via NSG



Step 8: Configure VNet Peering

- In **VNet1**, create peering to **VNet2**
- In **VNet2**, create peering to **VNet1**
- Enable traffic in both directions





Step 9: Test Cross-VNet Communication

- SSH into either Linux VM and ping the other's private IP
    - Confirm connectivity