**ASSIGNMENT 8**

# Introduction to Azure VPN Solutions

In the intricate landscape of modern IT infrastructure, robust and secure network connectivity forms the cornerstone of operational efficiency and data integrity. As businesses increasingly migrate and host their critical resources within cloud platforms like Microsoft Azure, the demand for seamless and secure communication between on-premises environments and the cloud has intensified. Virtual Private Network (VPN) solutions, particularly Point-to-Site (P2S) and Site-to-Site (S2S) configurations, emerge as indispensable tools to bridge this gap, offering encrypted and private communication channels over the public Internet.

These VPN configurations are pivotal in enabling:

* **Encrypted tunnels over the public Internet:** Protecting data in transit from eavesdropping and tampering by establishing secure, encrypted pathways.
* **Private IP access to Azure-hosted resources:** Allowing on-premises systems and remote users to interact with Azure Virtual Network (VNet) resources as if they were part of the same local network, using private IP addresses.
* **Scalable remote access for individuals and entire networks:** Providing flexible solutions for diverse connectivity needs, from a single remote user to an entire branch office.

This document serves as a comprehensive guide, meticulously detailing both Point-to-Site and Site-to-Site VPN configurations within Microsoft Azure. It provides step-by-step instructions, essential Azure CLI commands, practical considerations, and an in-depth comparison to empower IT professionals and aspiring cloud engineers with the knowledge to implement secure hybrid cloud connectivity.

# Point-to-Site (P2S) VPN – Azure Setup

## 🔷 What is Point-to-Site (P2S) VPN?

A Point-to-Site (P2S) VPN establishes a client-based connection, allowing individual users to securely connect to an Azure Virtual Network (VNet) from a remote location. Unlike Site-to-Site VPNs, P2S connections do not require a dedicated VPN device on the client's end or on the remote network. Instead, the connection is initiated directly from a client computer (e.g., a personal laptop, a developer workstation) using a built-in VPN client, such as the Windows native VPN client, or third-party compatible clients.

This setup is conceptually similar to "connecting your laptop to a company LAN" over a secure tunnel, making it ideal for remote employees, developers, and administrators who need secure access to Azure-hosted resources without requiring a full site-to-site VPN. P2S VPNs are flexible and support various authentication methods, including Azure native certificate authentication and RADIUS server authentication.

## 🧱 Key Components of a P2S VPN

Understanding the core components is crucial for a successful P2S VPN deployment:

* **Virtual Network (VNet):** This is the fundamental building block for your private network in Azure. It's an isolated network within the Azure cloud where your virtual machines and other resources reside. The P2S VPN connects remote clients directly into this private network.
* **Gateway Subnet:** A specially named subnet (GatewaySubnet) within your VNet. Azure VPN Gateway resources are deployed into this subnet. It must be named exactly GatewaySubnet for the deployment to succeed, and it requires a specific minimum address range (e.g., /27 or larger).
* **VPN Gateway:** The heart of the VPN solution in Azure. It's a network gateway that encrypts cross-premises traffic and sends it across a public connection. For P2S, it acts as the termination point for all client connections, managing the encrypted tunnels and routing traffic between the connected clients and the Azure VNet. It needs a public IP address to be reachable from the Internet.
* **Root Certificate:** When using Azure native certificate authentication (a common method for P2S), a self-signed root certificate (or a certificate from a trusted CA) is required. The public key portion of this root certificate is uploaded to the Azure VPN Gateway. All client certificates that are issued by or chained to this root certificate will be trusted by the VPN Gateway for authentication.
* **VPN Client:** This is the software installed on the end-user's device (e.g., Windows, macOS, Linux) that initiates and maintains the VPN connection to the Azure VPN Gateway. Azure provides a downloadable VPN client configuration package that simplifies the setup process on client machines.

## 🛠 Step-by-Step Setup: P2S VPN in Azure

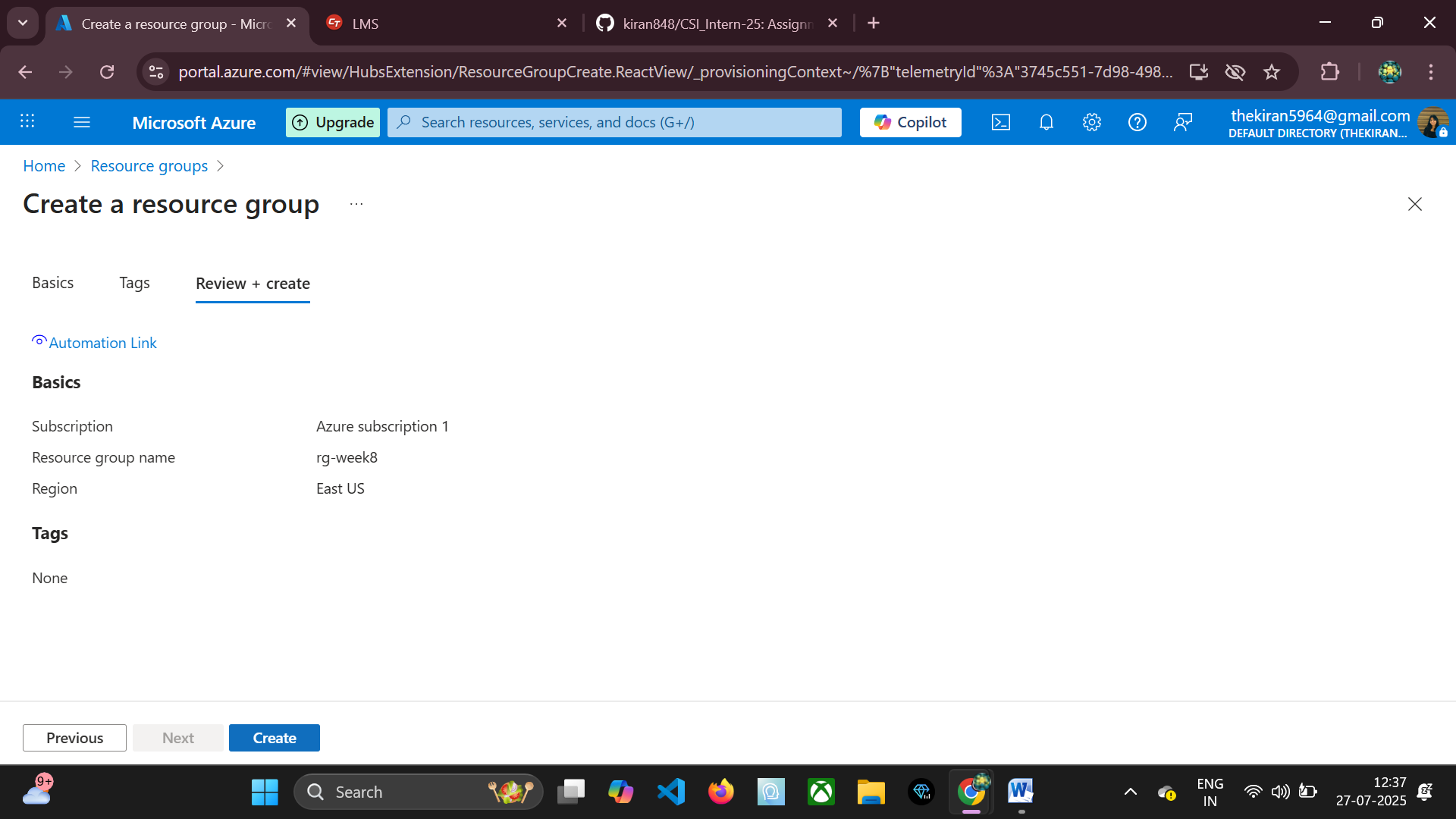
The following steps outline the process of configuring a Point-to-Site VPN using Azure CLI and PowerShell.

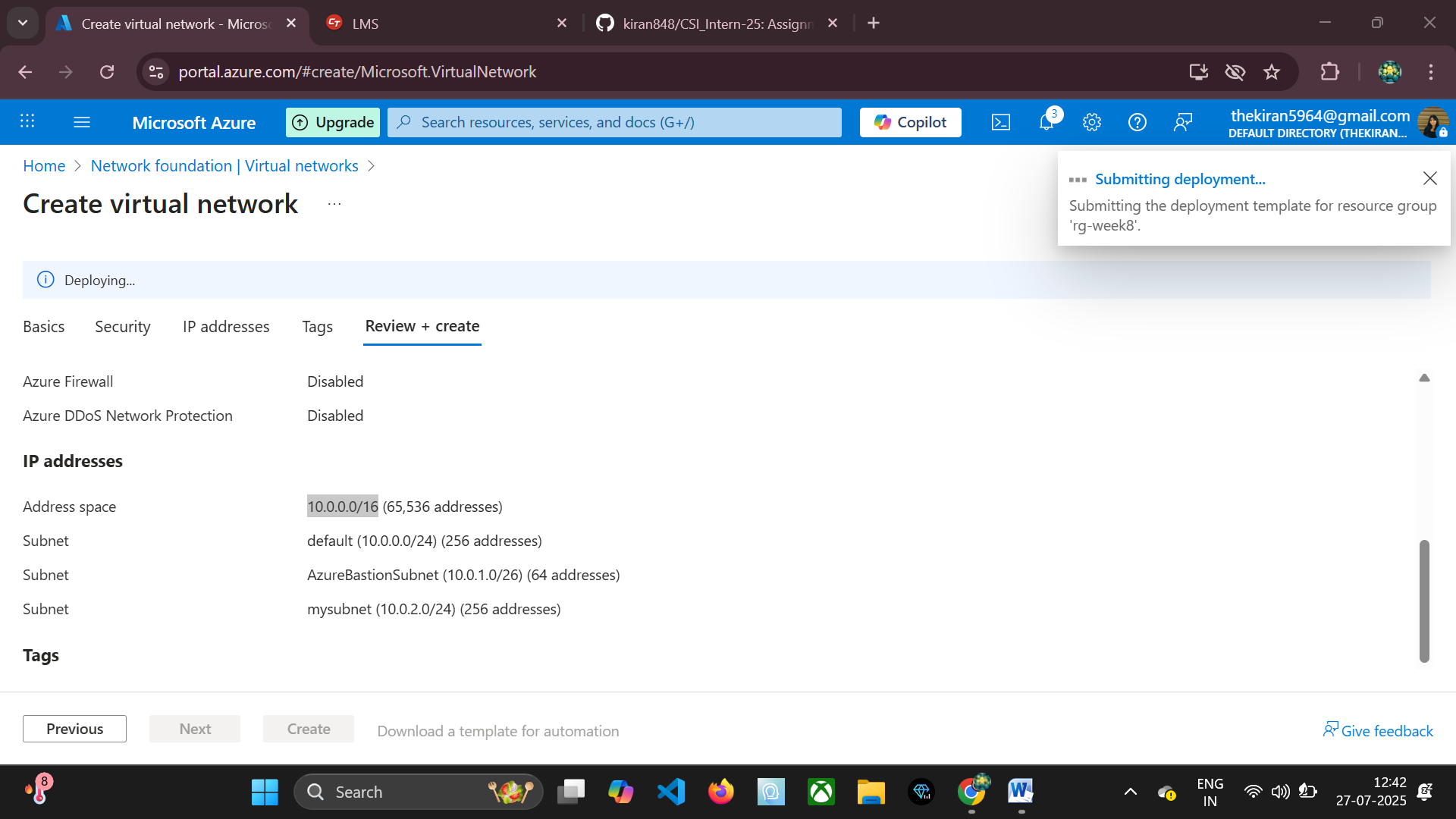
### ✅ Step 1: Create Virtual Network (VNet) and a General Subnet

First, you need a Virtual Network to host your Azure resources and a subnet for your virtual machines or other services.

az network vnet create --resource-group rg-week8 --name myVNet --address-prefix 10.1.0.0/16 --subnet-name mySubnet --subnet-prefix 10.0.2.0/24--location eastus

* **--resource-group rg-week8:** Specifies the resource group where the VNet will be created. A resource group is a logical container for Azure resources.
* **--name myVNet:** Sets the name of your Virtual Network.
* **--address-prefix 10.1.0.0/16:** Defines the overall private IP address space for your VNet. This is a larger range from which subnets will be carved out.
* **--subnet-name mySubnet:** Names the first general-purpose subnet within your VNet.
* **--subnet-prefix 10.0.2.0/24:** Defines the IP address range for mySubnet.
* **--location eastus:** Specifies the Azure region where the VNet will be deployed. Ensure all related resources are in the same region.



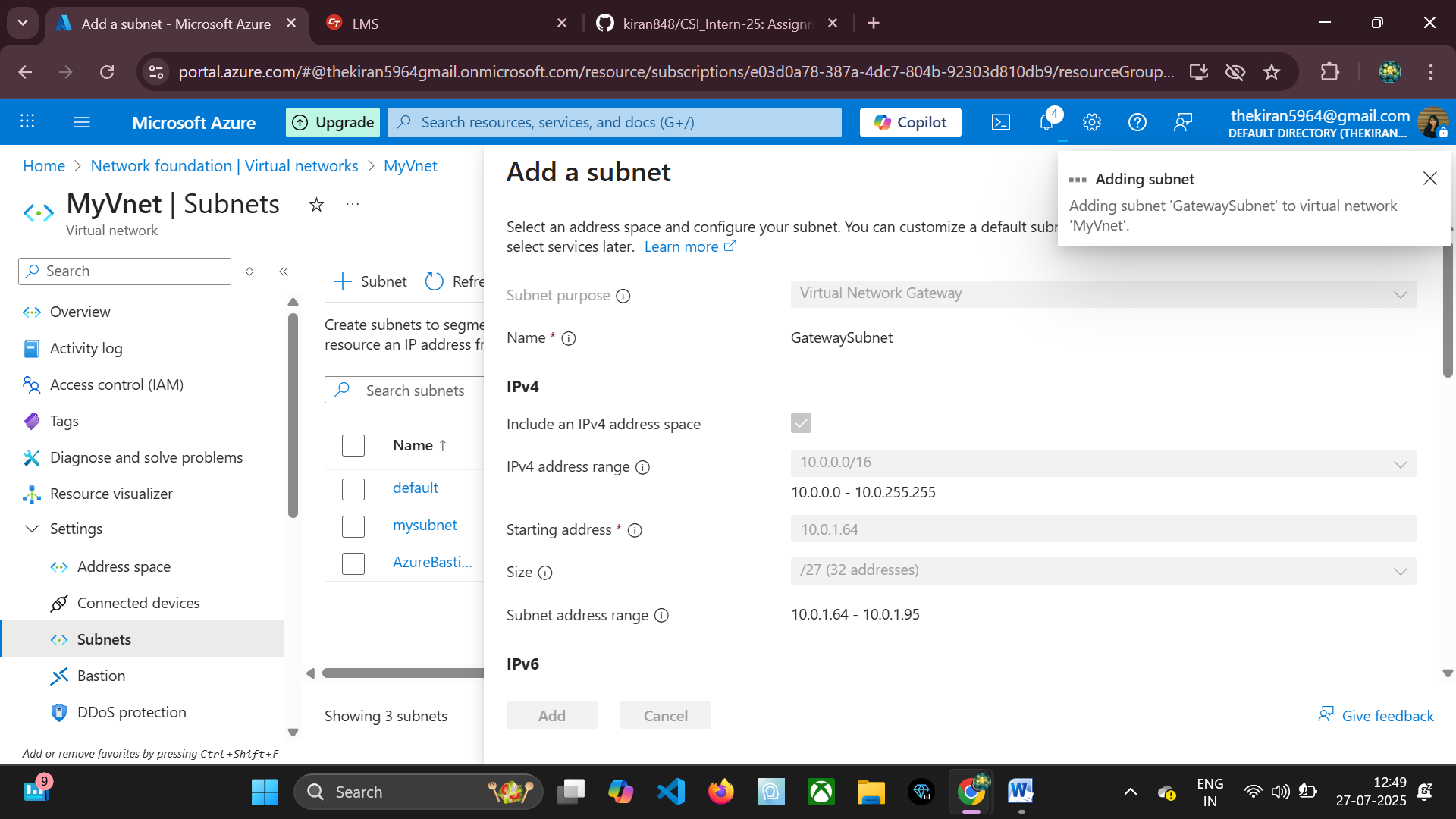


### ✅ Step 2: Add GatewaySubnet

A dedicated subnet named GatewaySubnet is mandatory for deploying the Azure VPN Gateway. This subnet is reserved exclusively for the VPN Gateway resources and cannot be used for any other purpose.

az network vnet subnet create --name GatewaySubnet --resource-group rg-week8 --vnet-name myVNet --address-prefix 10.0.1.64/27

* **--name GatewaySubnet:** This exact name is critical. Azure recognizes this name as the subnet dedicated to the VPN Gateway.
* **--address-prefix 10.0.1.64/27:** The address prefix for the GatewaySubnet. A minimum size of /27 is required, which provides 32 IP addresses. Using a larger subnet like /24 is recommended for future scalability and redundancy purposes, as the gateway may use multiple IP addresses internally.

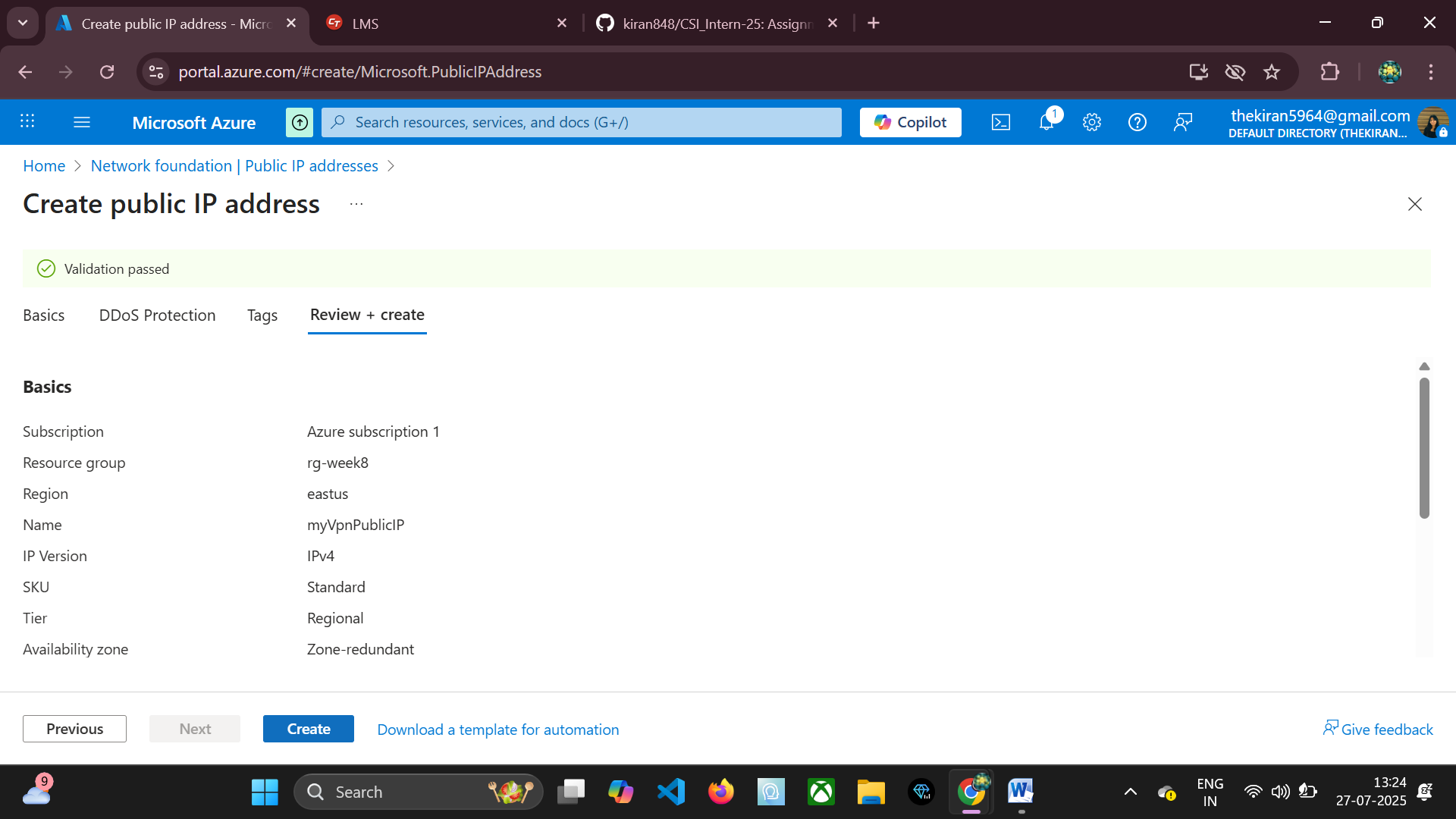


### ✅ Step 3: Create Public IP for Gateway

The Azure VPN Gateway requires a public IP address to allow client connections from the Internet. This IP address will be the destination for your P2S VPN clients.

az network public-ip create --name myVpnPublicIP --resource-group rg-week8 --allocation-method Dynamic --sku Basic --location eastus

* **--name myVpnPublicIP:** Names the public IP address resource.
* **--allocation-method Dynamic:** Specifies that the IP address is dynamically assigned. For VPN gateways, a dynamic IP is typically sufficient. A Static allocation ensures the IP doesn't change after a gateway resize or reboot, which can be useful but is not strictly necessary for basic P2S.
* **--sku Basic:** Defines the SKU for the public IP. For Basic VPN Gateways, a Basic SKU public IP is required. For higher performance gateways (VpnGw2+), a Standard SKU public IP is required.



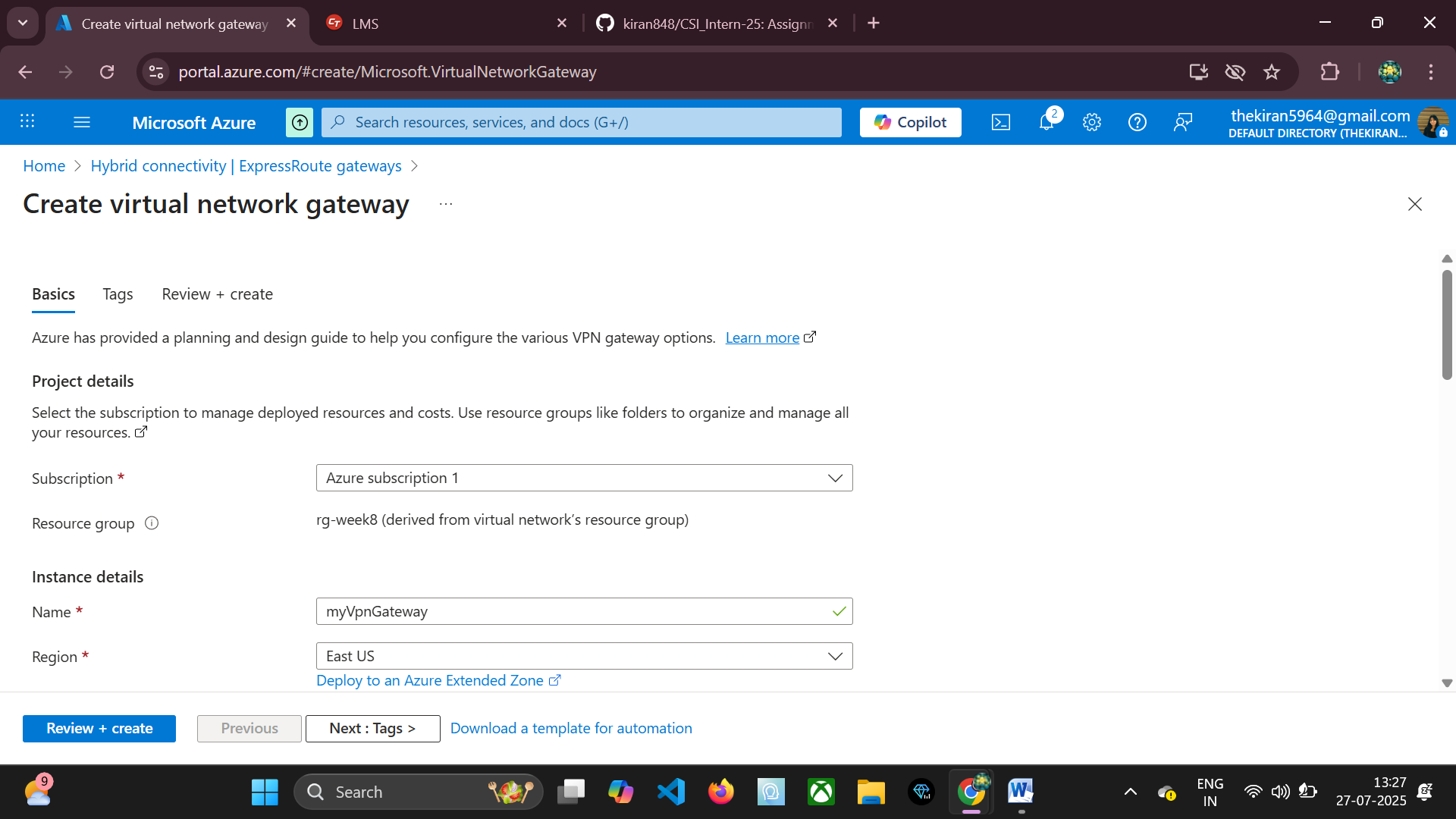
### ✅ Step 4: Create VPN Gateway

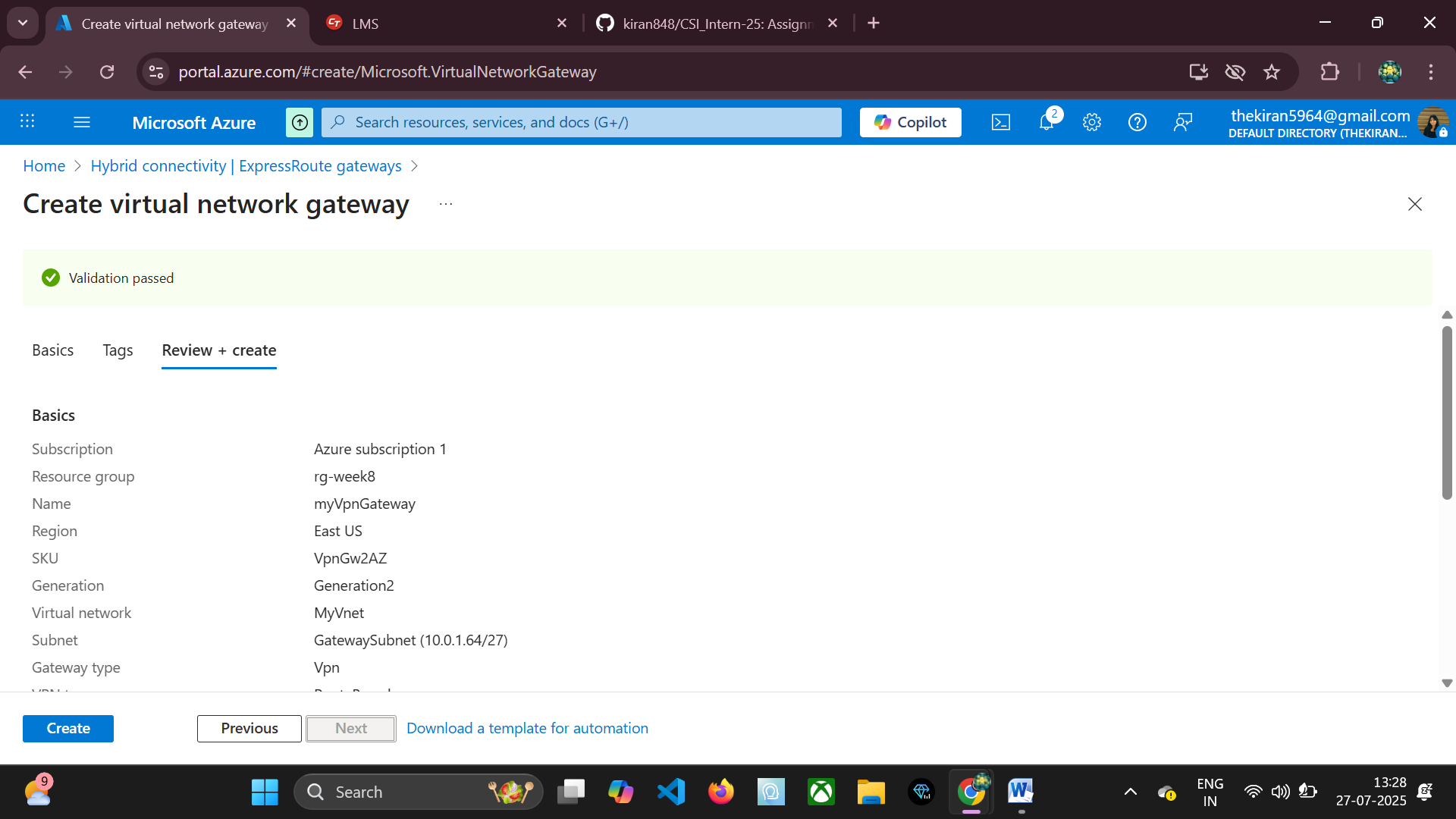
This is the most time-consuming step. The VPN Gateway is provisioned into the GatewaySubnet and associated with the public IP address.

az network vnet-gateway create --name myVpnGateway --resource-group rg-week8 --vnet myVNet --public-ip-address myVpnPublicIP --gateway-type Vpn --vpn-type RouteBased --sku VpnGw1 --location eastus --no-wait # To return control to the terminal immediately

* **--name myVpnGateway:** Names the VPN Gateway resource.
* **--vnet myVNet:** Links the gateway to your Virtual Network.
* **--public-ip-address myVpnPublicIP:** Associates the public IP created in the previous step.
* **--gateway-type Vpn:** Specifies that this is a VPN gateway (as opposed to an ExpressRoute gateway).
* **--vpn-type RouteBased:** Crucial for P2S and often for S2S. Route-based VPNs use routing or forwarding tables to direct traffic. This is required for IKEv2 P2S connections. Policy-based VPNs use policies to determine which traffic to encrypt.
* **--sku VpnGw1:** Defines the SKU (performance tier) of the VPN Gateway. VpnGw1 is suitable for moderate P2S loads. Other SKUs (Basic, VpnGw2, VpnGw3, VpnGw4, VpnGw5) offer varying levels of throughput, connections, and features.
* **--no-wait:** Allows the command to return immediately while the gateway deployment continues in the background.

**⚠️ Note:** The creation of a VPN Gateway is a resource-intensive operation and typically takes 20–45 minutes to complete. It is important to wait for this process to finish before proceeding with P2S configuration.





### 🔐 Step 5: Generate Root Certificate and Client Certificate

For Azure native certificate authentication, you need a root certificate. Self-signed certificates are commonly used for testing and development. You will export the public key of the root certificate (as a .cer file) and upload it to Azure. You will also need to issue client certificates from this root certificate, which will be installed on the client machines.

Open PowerShell as Administrator on a Windows machine:

# Create a self-signed root certificate

$cert = New-SelfSignedCertificate -Type Custom -KeySpec Signature -Subject "CN=AzureRootCert" -KeyExportPolicy Exportable -HashAlgorithm sha256 -KeyLength 2048 -CertStoreLocation "Cert:\\CurrentUser\\My" -KeyUsageProperty Sign -KeyUsage CertSign

# Export the public key (.cer) of the root certificate

Export-Certificate -Cert $cert -FilePath C:\\temp\\AzureRootCert.cer

# (Optional) Create a client certificate signed by the root certificate

# You would repeat this for each user or device that needs P2S access

$clientCert = New-SelfSignedCertificate -Type Custom -KeySpec Signature -Subject "CN=AzureClientCert" -KeyExportPolicy Exportable -HashAlgorithm sha256 -KeyLength 2048 -CertStoreLocation "Cert:\\CurrentUser\\My" -Signer $cert

# Export the client certificate with private key (PFX) for installation on client machines

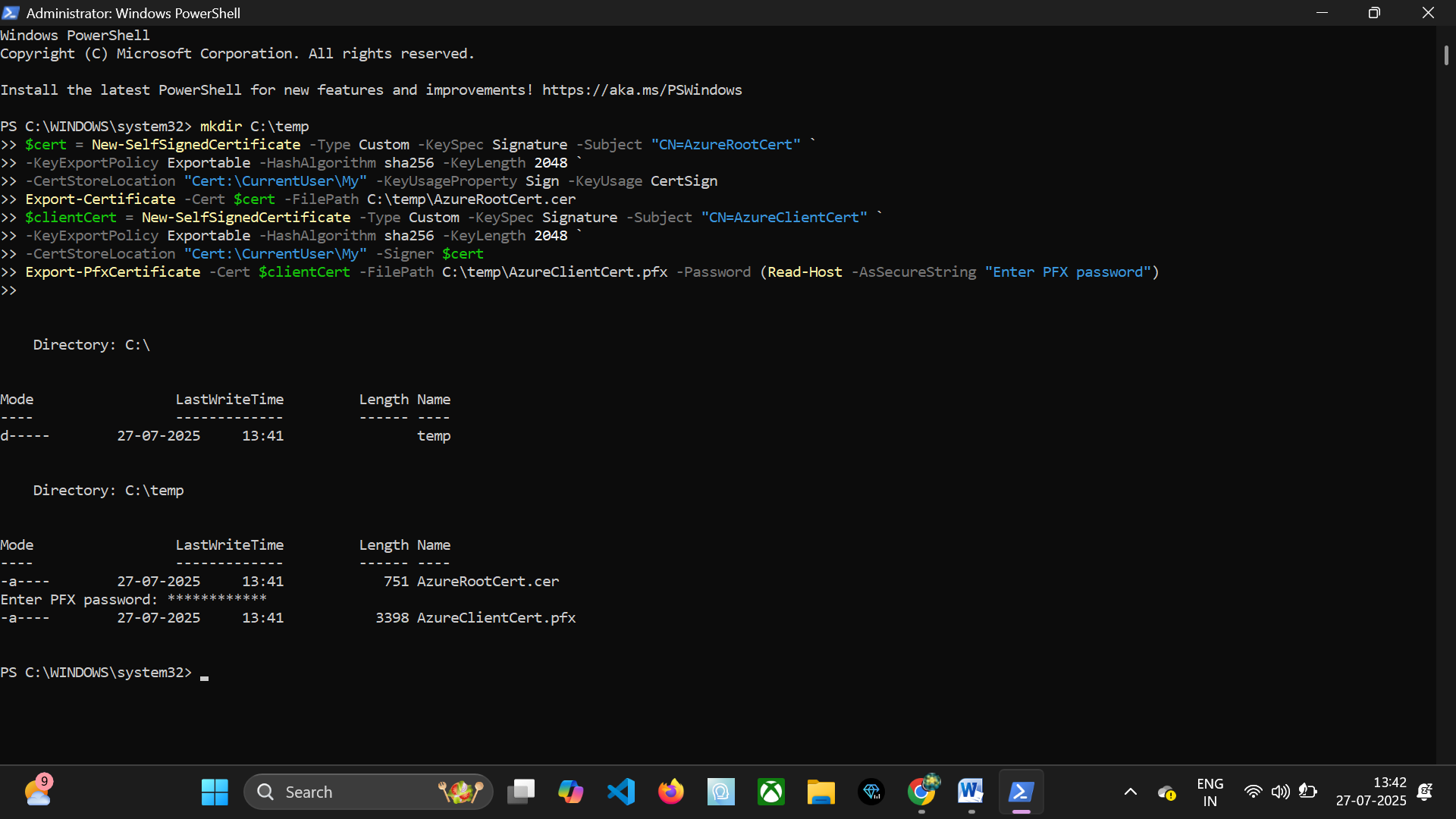
# This file must be protected as it contains the private key.

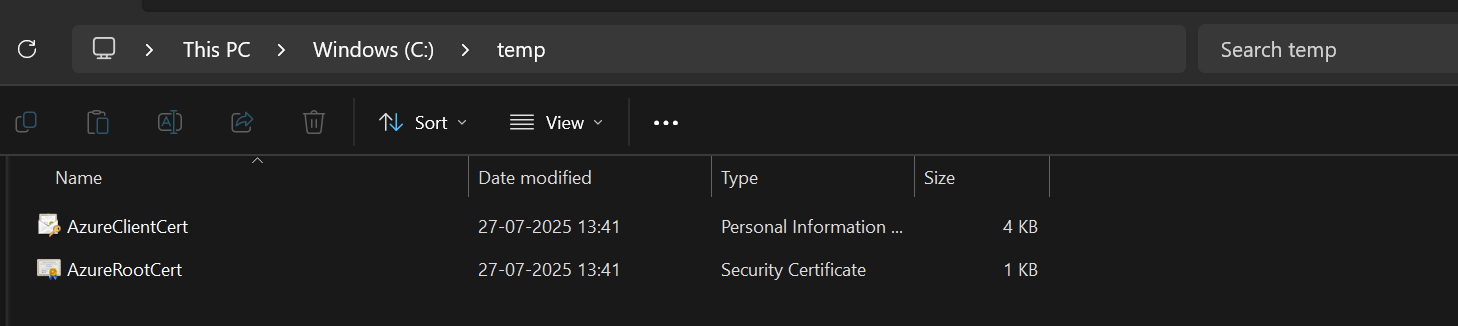
Export-PfxCertificate -Cert $clientCert -FilePath C:\\temp\\AzureClientCert.pfx -Password (Read-Host -AsSecureString "Enter PFX password")

Write-Host "Certificates generated. Check C:\\temp for .cer and .pfx files."

* **New-SelfSignedCertificate:** PowerShell cmdlet to create a self-signed certificate.
* **-KeySpec Signature:** Specifies that the private key can be used for digital signatures.
* **-Subject "CN=AzureRootCert":** Sets the subject name of the certificate. This is what identifies it.
* **-KeyExportPolicy Exportable:** Allows the private key to be exported, which is useful for backups or for signing client certificates from another machine.
* **-HashAlgorithm sha256:** Specifies the hashing algorithm. SHA256 is recommended for security.
* **-KeyLength 2048:** Specifies the length of the RSA key. 2048-bit is standard.
* **-CertStoreLocation "Cert:\\CurrentUser\\My":** Stores the certificate in the current user's personal certificate store.
* **-KeyUsageProperty Sign -KeyUsage CertSign:** These properties designate the certificate as a Certificate Authority (CA), allowing it to sign other certificates (client certificates).
* **Export-Certificate:** Exports the public key of the certificate to a .cer file. This is the file you will upload to Azure.
* **Export-PfxCertificate:** Exports the certificate along with its private key to a .pfx file. This is used when you need to install a client certificate on a user's machine. The PFX file should be password-protected.

**Action Required:** Upload the AzureRootCert.cer file to your Azure VPN Gateway's Point-to-Site configuration. The content of the .cer file (base-64 encoded public key) is what Azure uses to trust clients.



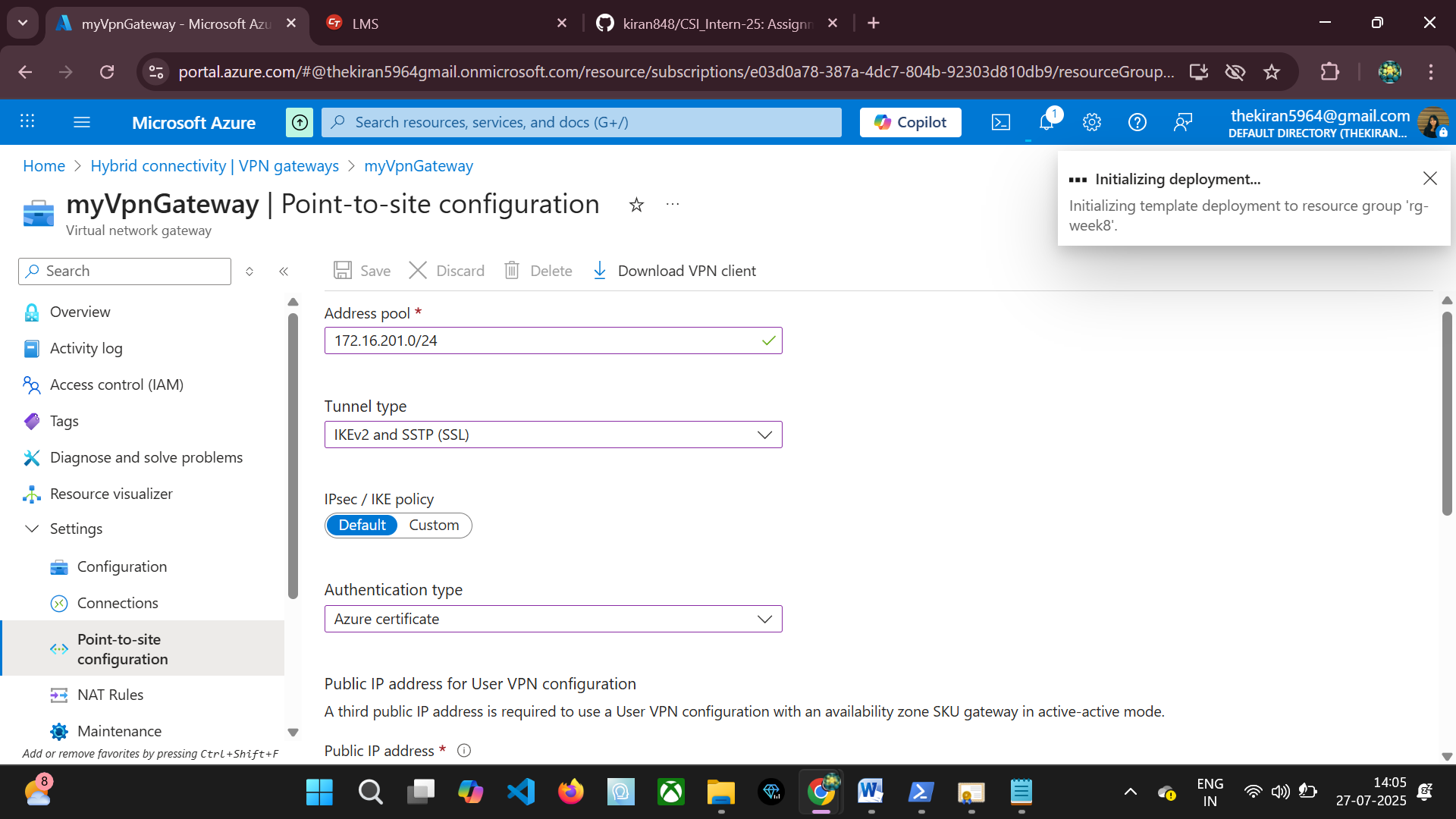


### 🔄 Step 6: Configure P2S Settings in Azure Portal or CLI

Once the VPN Gateway is provisioned and the root certificate is ready, you can configure the P2S settings.

**Using Azure Portal:**

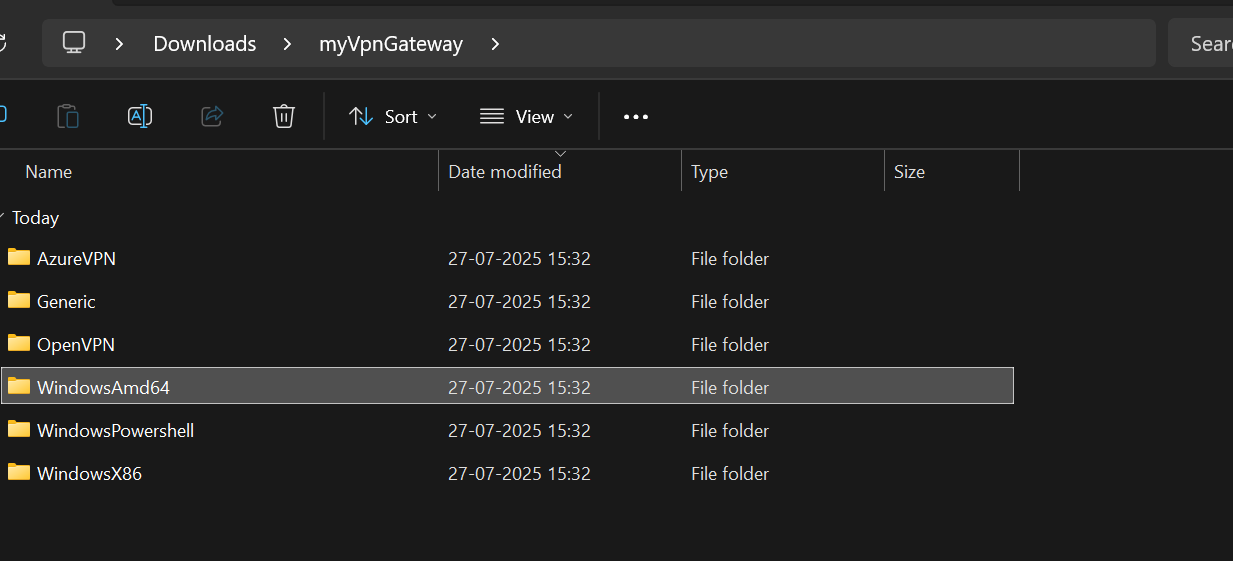
* Navigate to your Virtual Network Gateway (e.g., myVpnGateway).
* In the left-hand menu, under Settings, click Point-to-site configuration.
* Click Configure now.
* **Address pool:** Enter an address range (e.g., 172.16.201.0/24). This is the IP address range from which VPN clients will be assigned IP addresses when they connect. This range must not overlap with your VNet address space or any on-premises network ranges connected via S2S VPN.
* **Tunnel type:** Select IKEv2 and SSTP (SSL) for maximum compatibility and security. IKEv2 is generally preferred for performance and stability, while SSTP provides fallback for networks with strict firewalls.
* **Authentication type:** Select Azure certificate.
* **Root certificate public certificate data:** Open your AzureRootCert.cer file with a text editor (like Notepad), copy the entire content (excluding -----BEGIN CERTIFICATE----- and -----END CERTIFICATE-----), and paste it into this field. Provide a name for the certificate (e.g., AzureRootCert).
* Click Save.



### 📥 Step 6: Download and Install VPN Client

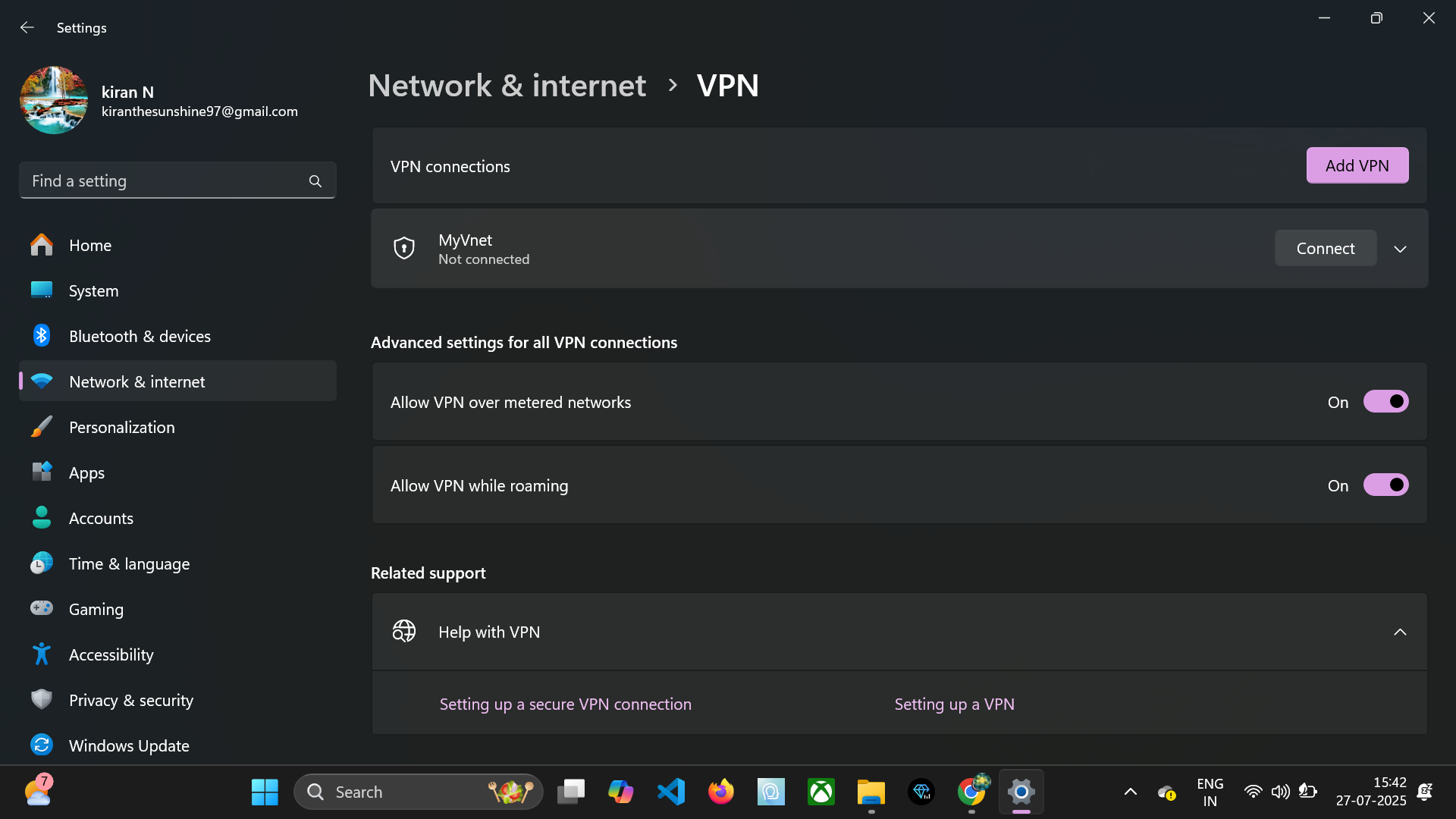
After configuring P2S settings, Azure generates a VPN client configuration package.

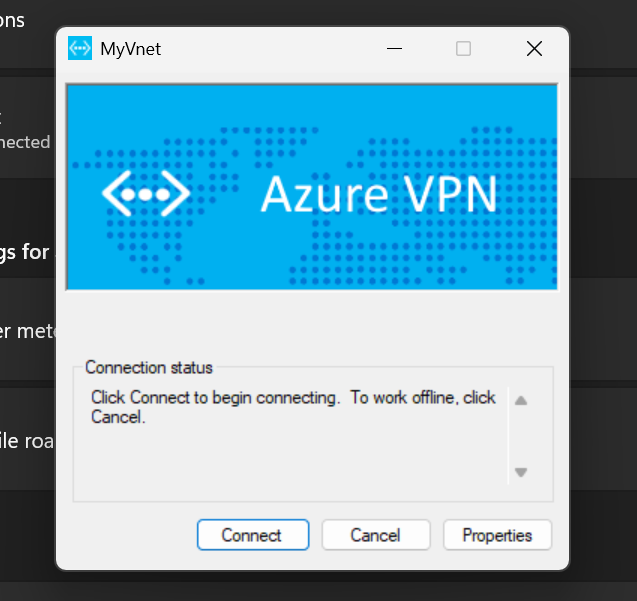
* Go to your VPN Gateway > Point-to-site configuration.
* Click Download VPN Client.
* Save the .zip file to your local machine.
* Extract the contents of the .zip file.
* Inside the extracted folder, navigate to the WindowsAmd64 (for 64-bit Windows) or WindowsX86 (for 32-bit Windows) folder.
* Run the VpnClientSetupAmd64.exe (or VpnClientSetupX86.exe) installer. This installs the Azure VPN client profile and the necessary certificates into your Windows native VPN client.
* **Install Client Certificate:** If you generated a client certificate (e.g., AzureClientCert.pfx), you must install it into the Personal certificate store of the current user on the client machine. Double-click the .pfx file and follow the import wizard. Ensure you mark the private key as exportable if you plan to move it, though for client certificates, it's generally not needed.



Once installed, go to Network & Internet settings > VPN in Windows. You should see a new VPN connection (e.g., myVNet).

Click on the VPN connection and then Connect. You may be prompted to select a client certificate if multiple are present.





## 💡 Benefits of P2S VPN

* **No dedicated hardware/router needed on client side:** Simplifies deployment for individual users, as they only need software.
* **Good for remote users, developers, and testers:** Provides secure access for employees working from home, contractors, or development teams needing secure connectivity to isolated Azure environments.
* **Supports certificate or RADIUS-based authentication:** Offers flexibility in authentication mechanisms, allowing integration with existing identity systems.
* **Cost-effective for smaller numbers of remote users:** Avoids the overhead and cost of deploying and managing dedicated VPN hardware at each remote location.
* **Quick to deploy:** Once the gateway is up, configuring P2S and distributing clients is relatively fast.

# Site-to-Site (S2S) VPN Using Hyper-V + RRAS

## 🔷 What is Site-to-Site (S2S) VPN?

A Site-to-Site (S2S) VPN connects an entire on-premises network to an Azure Virtual Network (VNet), effectively extending your on-premises network into the cloud. This type of VPN creates a persistent, encrypted tunnel between a VPN device or router on your physical premises (e.g., a hardware VPN appliance, a software-based VPN server like RRAS) and an Azure VPN Gateway. All traffic between the specified on-premises network ranges and the Azure VNet is routed through this secure tunnel.

S2S VPNs are foundational for hybrid cloud architectures, enabling seamless communication between on-premises applications and cloud resources, supporting scenarios like data replication, disaster recovery, and extending Active Directory domains to Azure. They allow multiple devices on the on-premises network to access Azure resources without each device needing its own VPN client.

## 📌 Tools Used for Simulation

To demonstrate a Site-to-Site VPN in a lab environment without physical hardware, we simulate the on-premises network using virtualized components:

* **Hyper-V:** Microsoft's virtualization platform. We will use Hyper-V to host a Windows Server Virtual Machine (VM) that will act as our on-premises VPN device.
* **RRAS (Routing and Remote Access Service):** A Windows Server role that provides routing and remote access capabilities, including VPN server functionality. RRAS will be configured on the Hyper-V VM to establish and manage the VPN tunnel to the Azure VPN Gateway.
* **Azure Virtual Network Gateway:** This is the Azure-side component that terminates the S2S VPN tunnel and routes traffic to the Azure VNet.

## 🛠 Step-by-Step Configuration: S2S VPN

The setup involves configuring both the Azure environment and the simulated on-premises environment (Hyper-V VM with RRAS).

### ✅ Step 1: Create Azure Resources (VNet, Gateway Subnet, Public IP, VPN Gateway)

These initial steps are identical to the P2S VPN setup, as a VPN Gateway is required for both P2S and S2S connections. If you already completed Section 2, you can skip this, assuming myVNet, GatewaySubnet, myVpnPublicIP, and myVpnGateway are already created.

**Create VNet and Subnet:**

az network vnet create --resource-group rg-week8 --name myVNet --address-prefix 10.1.0.0/16 --subnet-name mySubnet --subnet-prefix 10.1.0.0/24 --location eastus

**Add GatewaySubnet:**

az network vnet subnet create --name GatewaySubnet --resource-group rg-week8 --vnet-name myVNet --address-prefix 10.1.255.0/27

**Create Public IP for Gateway:**

az network public-ip create --name myVpnPublicIP --resource-group rg-week8 --allocation-method Dynamic --sku Basic --location eastus

**Create VPN Gateway:**

az network vnet-gateway create --name myVpnGateway --resource-group rg-week8 --vnet myVNet --public-ip-address myVpnPublicIP --gateway-type Vpn --vpn-type RouteBased --sku VpnGw1 --location eastus --no-wait

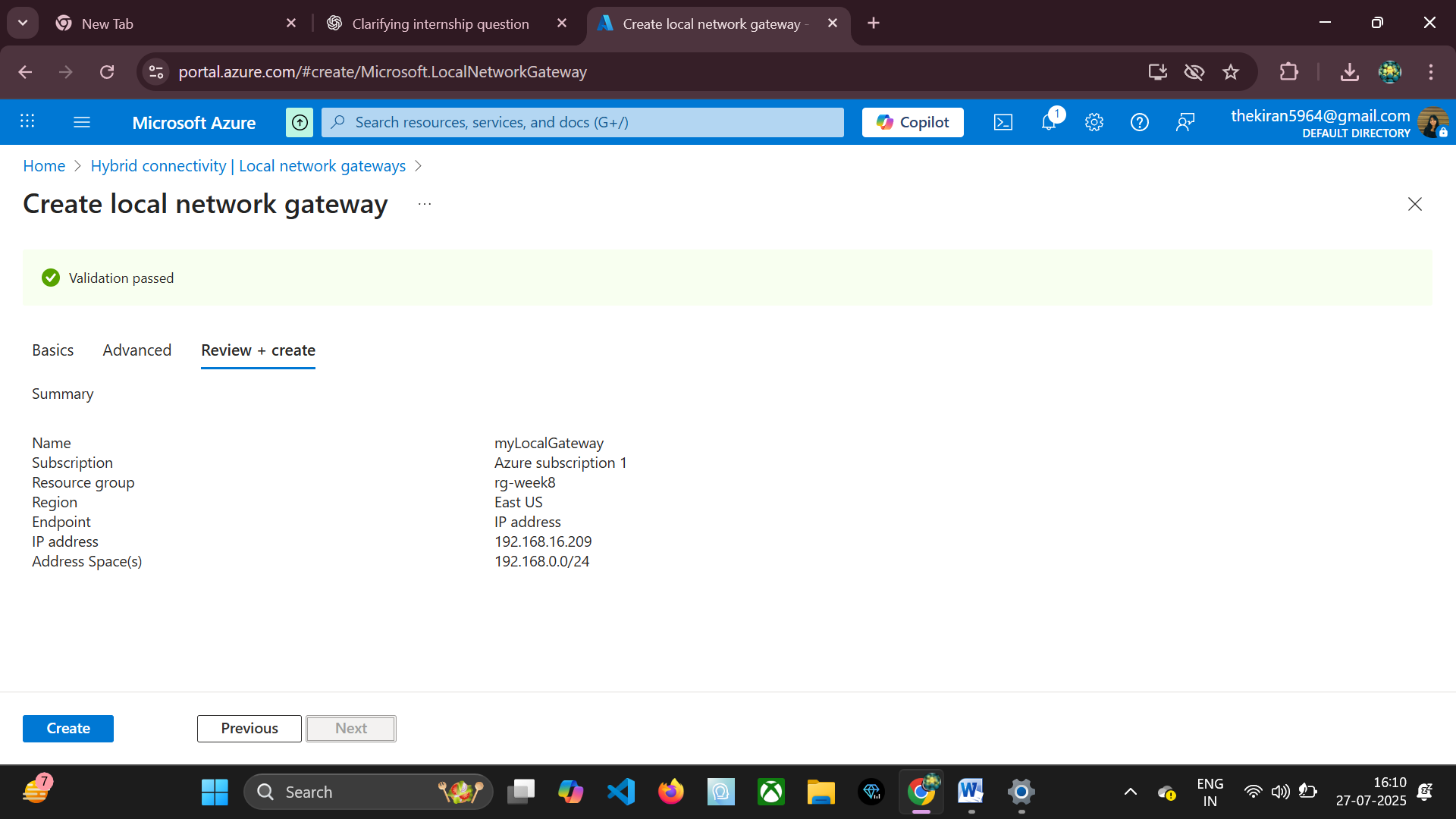
**Note on vpn-type:** For S2S VPNs, RouteBased is highly recommended as it supports IKEv2 (preferred) and allows for more flexible routing configurations. Policy-based VPNs are simpler but less flexible and generally use IKEv1.

### ✅ Step 2: Create Local Network Gateway

A Local Network Gateway in Azure represents your on-premises VPN device and network. It provides Azure with information about your on-premises network configuration, including the public IP address of your VPN device and the address prefixes of your on-premises network.

az network local-gateway create --resource-group rg-week8 --name OnPremGateway --gateway-ip-address --local-address-prefixes 192.168.0.0/16 --location eastus

* **--name OnPremGateway:** A logical name for your on-premises network gateway.
* **--gateway-ip-address :** This is the public IP address of your Hyper-V VM running RRAS. If your Hyper-V host is behind a NAT, you'll need to use the public IP of your router and configure port forwarding for UDP ports 500 and 4500 to the RRAS VM. In a typical lab scenario, the Hyper-V VM might have a publicly routable IP or be behind a simple NAT where the host's public IP is used.
* **--local-address-prefixes 192.168.0.0/16:** This specifies the IP address range(s) of your on-premises network that you want to connect to Azure. All traffic destined for the Azure VNet from these prefixes will be routed through the S2S VPN tunnel.

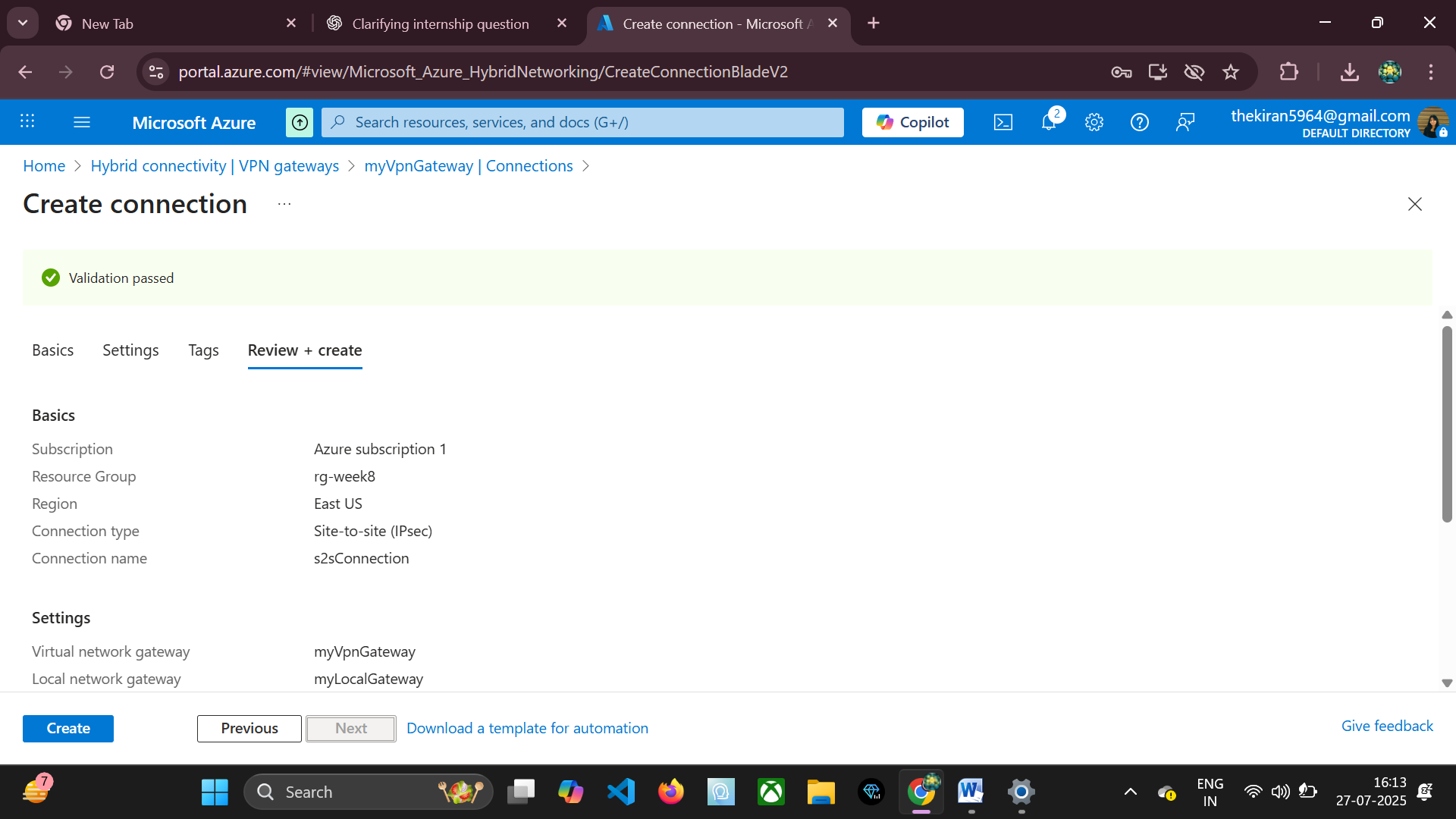


### ✅ Step 3: Create VPN Connection (S2S)

This step creates the actual S2S VPN connection resource in Azure, linking your Azure VPN Gateway to your Local Network Gateway and defining the shared key.

az network vpn-connection create --name S2SConnection --resource-group rg-week8 --vnet-gateway1 myVpnGateway --local-gateway2 OnPremGateway --shared-key "MySecretKey123" --connection-type IPsec --location eastus

* **--name S2SConnection:** The name of the connection resource.
* **--vnet-gateway1 myVpnGateway:** Refers to the Azure VPN Gateway created earlier.
* **--local-gateway2 OnPremGateway:** Refers to the Local Network Gateway representing your on-premises network.
* **--shared-key "MySecretKey123":** This is a pre-shared key (PSK) that must be identical on both the Azure VPN Gateway and your on-premises VPN device (RRAS in this case). This key is crucial for establishing the initial secure handshake (Phase 1 of IKE). Choose a strong, complex key.
* **--connection-type IPsec:** Specifies that this is an IPsec connection, which is the protocol used for S2S VPNs.



## 🧑‍💻 On Hyper-V VM (Windows Server) Configuration

Now, we switch to configuring the on-premises component, which is a Windows Server VM running on Hyper-V. This VM will simulate your on-premises VPN router. It should have at least two network adapters: one connected to an external network (simulating the Internet connection) and another connected to an internal network (simulating the on-premises LAN).

### Step 1: Install RRAS Role

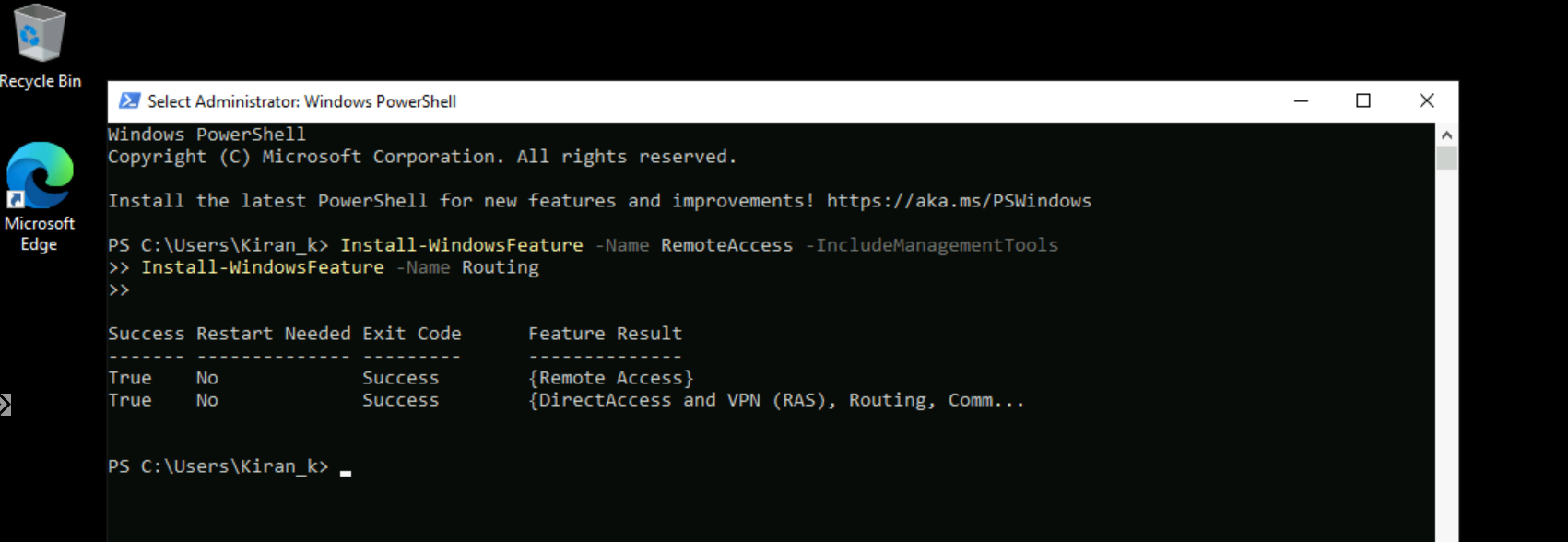
Install the Routing and Remote Access Service role on your Windows Server VM.

Install-WindowsFeature RemoteAccess -IncludeManagementTools

Install-WindowsFeature Routing

* **RemoteAccess:** Installs the core Remote Access role, which includes VPN capabilities.
* **Routing:** Installs components necessary for network routing, including NAT and static routing.
* **IncludeManagementTools:** Installs the graphical management console for RRAS.

After installation, open Server Manager > Tools > Routing and Remote Access.



### Step 2: Configure RRAS

Configure the RRAS service to act as a VPN server and a router.

In the RRAS management console, right-click on the server name (e.g., SERVER01) and select Configure and Enable Routing and Remote Access.

The Routing and Remote Access Server Setup Wizard will appear. Click Next.

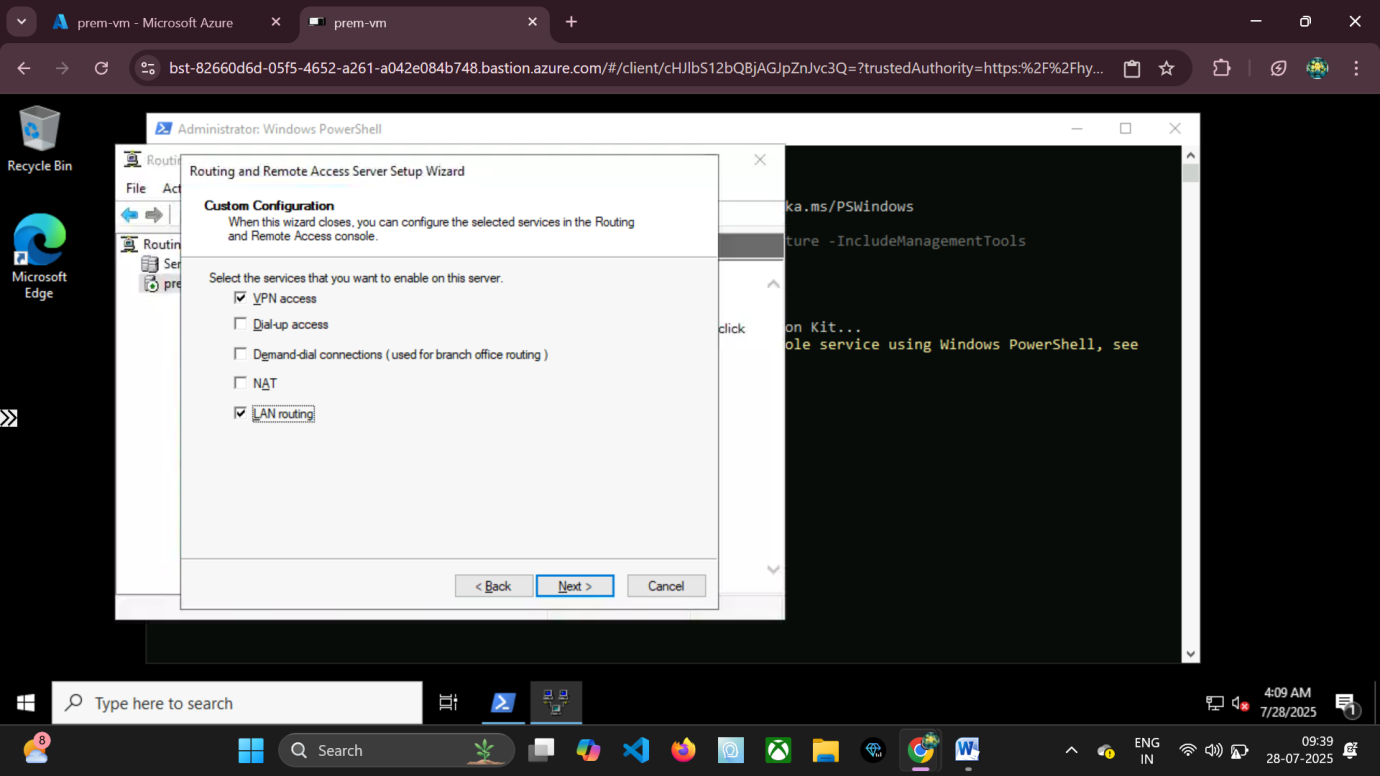
Choose Custom Configuration. This allows you to select specific features.

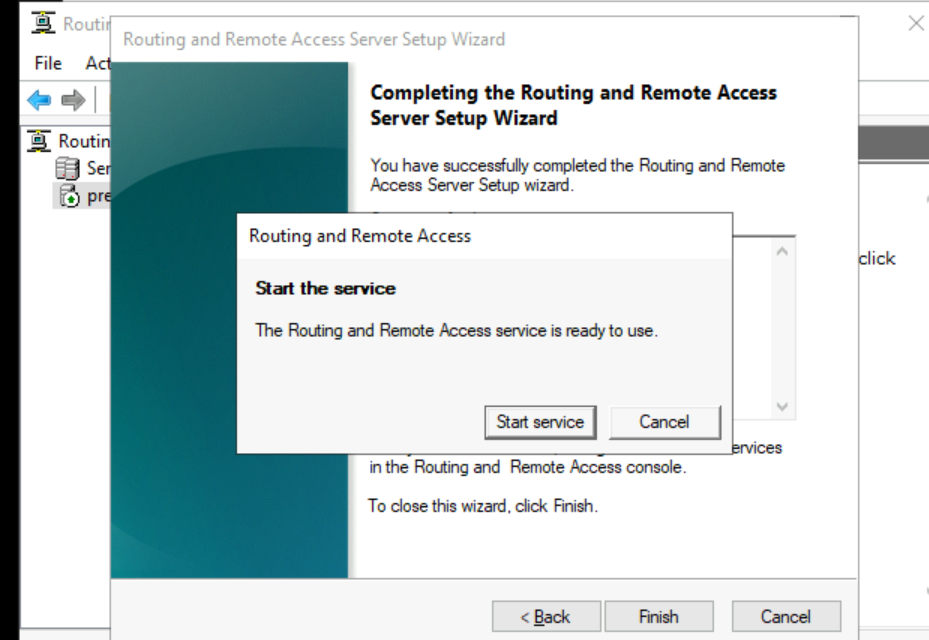
Select VPN and LAN Routing.

* **VPN:** Enables the server to act as a VPN endpoint.
* **LAN Routing:** Allows the server to route traffic between different network interfaces (e.g., between the "Internet" interface and the "LAN" interface).

Click Next, then Finish.

When prompted, choose Start service.





### Step 3: Add Demand-Dial Interface for S2S VPN

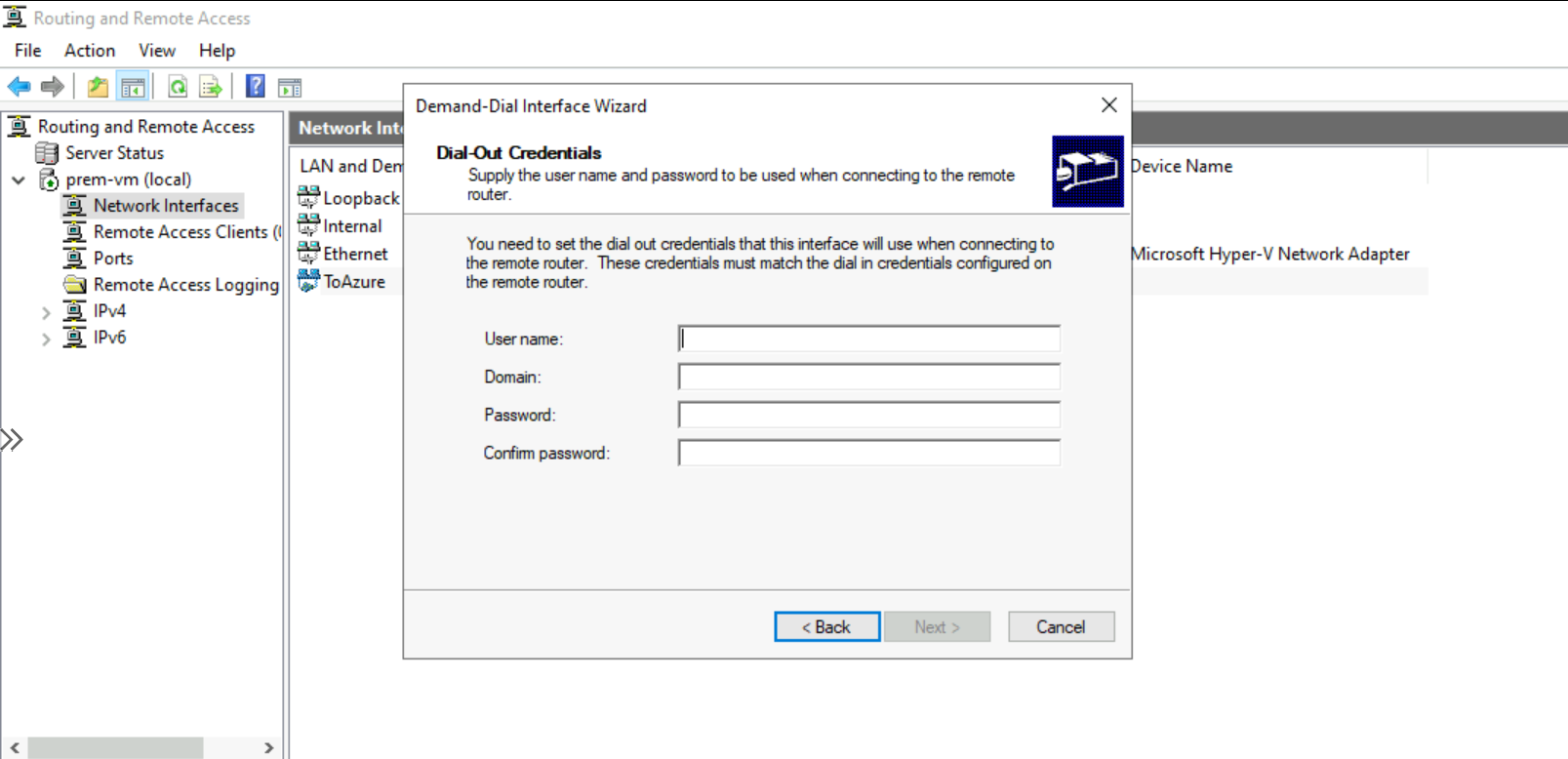
A demand-dial interface in RRAS is used to initiate a VPN connection to a remote network when traffic for that network is detected.

In the RRAS console, expand your server, then right-click Network Interfaces and select New Demand-Dial Interface.

The Demand-Dial Interface Wizard starts. Click Next.

* **Interface Name:** Enter AzureTunnel (or any descriptive name).
* **Connection Type:** Select Connect using virtual private networking (VPN).
* **VPN Type:** Choose IKEv2. This is the recommended VPN type for Azure S2S VPNs due to its robustness and support for MOBIKE.
* **Destination Address:** Enter the Public IP address of your Azure VPN Gateway. You can find this in the Azure Portal by navigating to your myVpnGateway resource.
* **Protocols and Security:** Keep the default settings.
* **Dial Out Credentials:** Ensure "Use pre-shared key for authentication" is selected. Enter the exact same pre-shared key (MySecretKey123) that you defined in Azure when creating the VPN connection.

Click Next, then Finish. The interface will appear under Network Interfaces.



### Step 4: Configure Static Route to Azure VNet

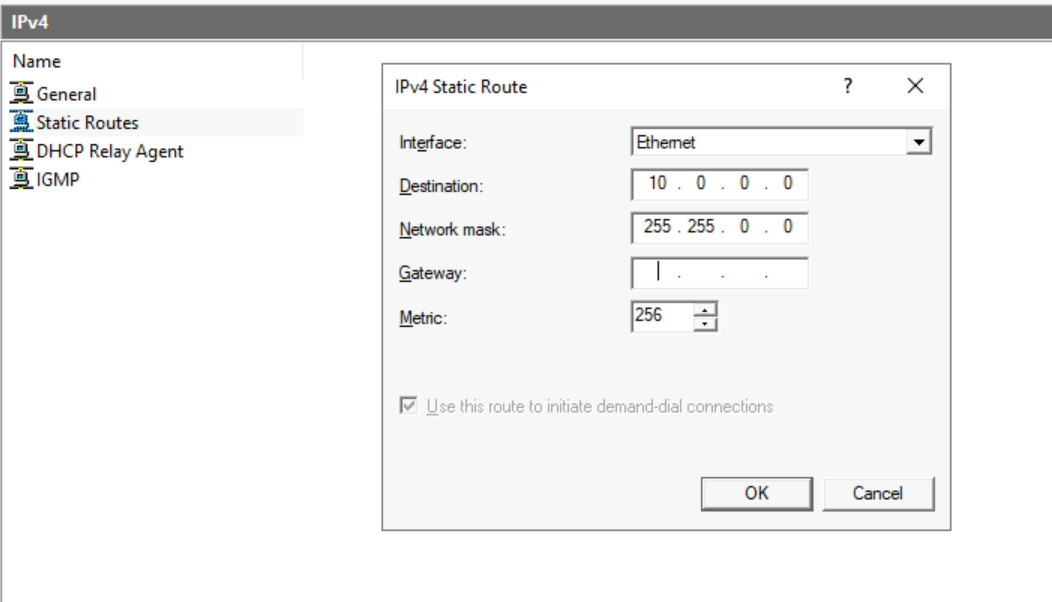
Finally, you need to tell your RRAS server how to reach the Azure VNet through the newly created VPN tunnel. This is done by adding a static route.

In the RRAS console, expand your server, then expand IPv4 (or IPv6 if applicable).

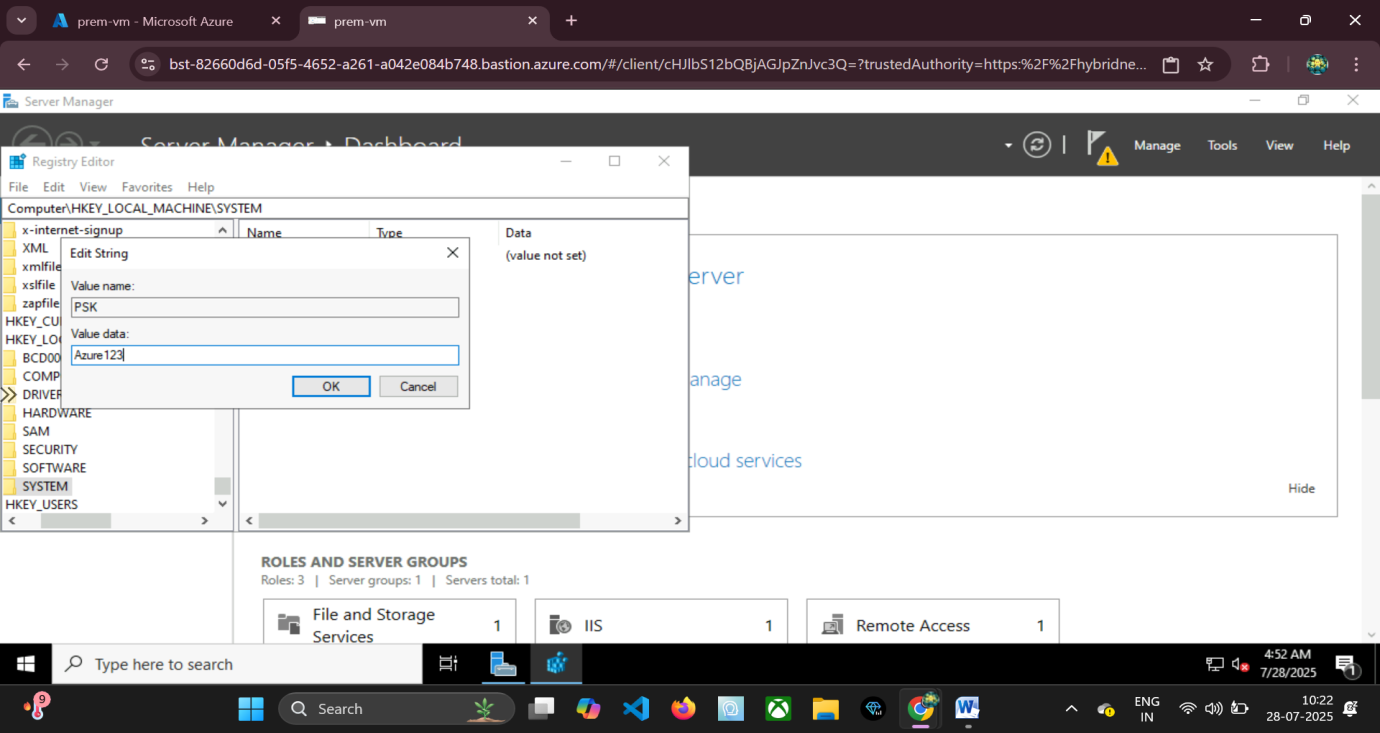
Right-click Static Routes and select New Static Route.

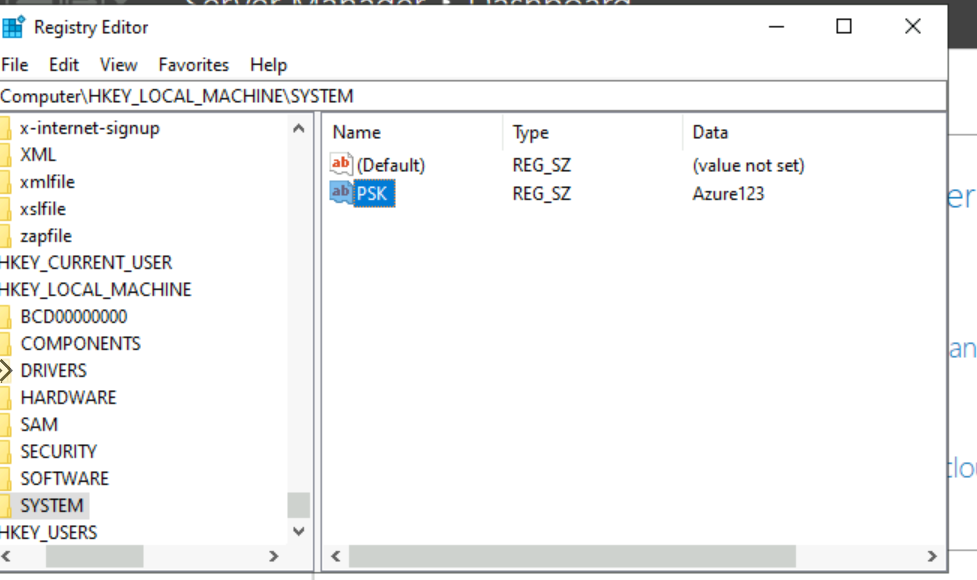
* **Interface:** Select AzureTunnel (the demand-dial interface you just created).
* **Destination:** Enter 10.0.0.0 (the network address of your Azure VNet).
* **Network mask:** Enter 255.255.0.0 (corresponding to the /16 of your Azure VNet).
* **Gateway:** Leave blank or set to 0.0.0.0 as the demand-dial interface handles the routing through the tunnel.

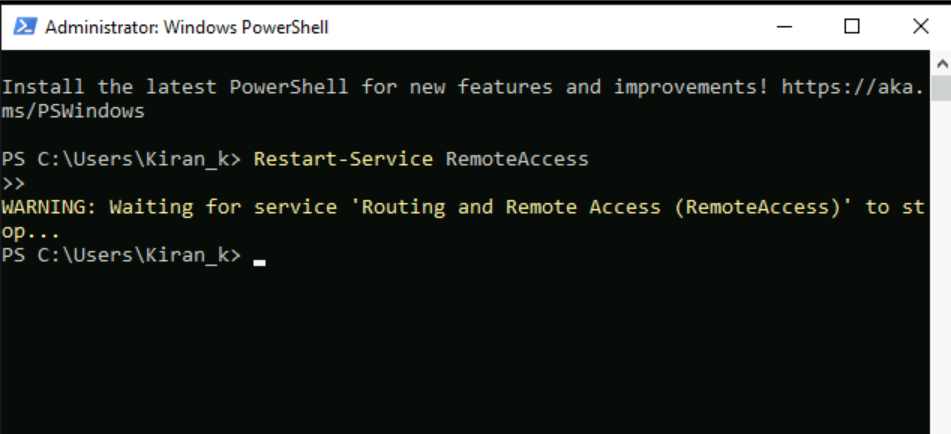
Click OK.



Once this is configured, any traffic originating from your Hyper-V VM or its simulated LAN (e.g., another VM connected to its internal switch) destined for 10.1.0.0/16 will trigger the AzureTunnel demand-dial interface, establish the VPN connection, and route the traffic through the encrypted tunnel to Azure.







**Firewall Considerations on RRAS VM:** Ensure that Windows Firewall on the RRAS VM allows inbound UDP traffic on ports 500 (IKE) and 4500 (IPsec NAT-T) from the Internet to allow the Azure VPN Gateway to initiate or respond to the connection.

## ⚠️ Troubleshooting Tips for S2S VPN

Establishing S2S VPNs can sometimes be challenging. Here are common issues and their fixes:

### VPN tunnel not forming:

* **Check pre-shared key:** Ensure the pre-shared key is identical on both the Azure VPN Connection and the RRAS demand-dial interface. Even a single character mismatch will prevent the tunnel from forming.
* **Public IP addresses:** Verify that the gateway-ip-address in the Azure Local Network Gateway matches the actual public IP of your RRAS VM (or the NAT device in front of it). Also, confirm the Destination Address in RRAS demand-dial interface is the correct public IP of the Azure VPN Gateway.
* **Firewall:** Ensure no firewalls (Windows Firewall on RRAS VM, network firewalls, router ACLs) are blocking UDP ports 500 (IKE) and 4500 (IPsec NAT-T) in both directions.

### Traffic blocked after tunnel forms:

* **Network Security Groups (NSGs):** Check NSGs applied to subnets in your Azure VNet. Ensure they allow inbound traffic from your on-premises IP ranges (e.g., 192.168.0.0/16).
* **User Defined Routes (UDRs):** If you have custom routing tables, ensure they are not overriding the VPN gateway route.

### Wrong address space:

* **Match local and Azure address prefixes:** The local-address-prefixes defined in the Azure Local Network Gateway must accurately represent your on-premises network. Similarly, the static route on RRAS must correctly point to the Azure VNet address space. Any mismatch means traffic won't be routed correctly.
* **Overlapping IP ranges:** Ensure your on-premises IP address range and your Azure VNet IP address range do not overlap. Overlapping ranges lead to routing conflicts.

### RRAS errors:

* **Restart RRAS service:** Sometimes, a simple restart of the "Routing and Remote Access" service can resolve transient issues.
* **Check IKE logs:** Look into the Windows Event Viewer on the RRAS VM, specifically under "Applications and Services Logs" -> "Microsoft" -> "Windows" -> "RasClient" or "IKEv2." These logs provide detailed information about VPN connection attempts and failures.

### Azure Monitoring:

Use Azure Monitor for your VPN Gateway. Metrics like "Gateway P2S Connection Count" (for P2S) and "Gateway S2S Bandwidth" (for S2S) can indicate if the tunnel is up and passing traffic. Check the "Connections" blade on your VPN Gateway in Azure Portal to see the connection status.

## ✅ Benefits of S2S VPN

* **Ideal for enterprise branch offices:** Connects entire remote networks securely to Azure, suitable for corporate environments with multiple devices needing cloud access.
* **No need to install client on each machine:** Once the S2S tunnel is established, all devices on the on-premises network can access Azure resources (and vice-versa) transparently, without individual VPN client software.
* **Enables hybrid network architecture:** Facilitates seamless integration of on-premises data centers with Azure cloud infrastructure, allowing for shared resources and services.
* **Persistent connection, no manual login:** The tunnel remains active, providing always-on connectivity between the two networks.
* **Supports data replication and disaster recovery:** Essential for scenarios where large volumes of data need to be synchronized or replicated between on-premises and Azure for business continuity.

# P2S vs S2S: Comparison Table

|  |  |  |
| --- | --- | --- |
| **Feature** | **Point-to-Site (P2S)** | **Site-to-Site (S2S)** |
| Connection Type | Individual client (e.g., laptop, desktop) | Whole on-premises network (e.g., branch office, data center) |
| Endpoint Device | Client software on user's machine | VPN device/router (hardware appliance or software like RRAS) |
| Setup Complexity | Simple (client-side installation) | Moderate to Complex (on-premises device configuration) |
| Cost | Lower (no dedicated on-prem hardware) | Higher (requires dedicated on-prem device/server and continuous gateway cost) |
| Suitable For | Remote users, developers, testers, administrators, temporary access | Branch offices, enterprises, hybrid cloud deployments, disaster recovery |
| Authentication | Certificates (Azure native) or RADIUS | Pre-shared Key (PSK) or BGP with IPsec/IKEv2 (requires Policy/Route-based VPN) |
| Always On? | No (connection initiated by user, session-based) | Yes (persistent tunnel between networks) |
| Scalability | Scalable for many individual users | Scalable for connecting multiple branch offices, complex network routing |
| Traffic Flow | Client IP to VNet IP | On-prem Subnet to Azure VNet Subnet |
| Routing | Clients get an IP from the P2S address pool, routes configured automatically by client | Static or BGP-based routing configured on both VPN endpoints |
| NAT Support | Client side can be behind NAT | Azure Gateway supports NAT, on-prem device often handles its own NAT |

# 8. Conclusion

In this intensive internship task, we embarked on a detailed exploration of two fundamental and widely deployed networking methodologies crucial for robust hybrid cloud environments: Point-to-Site (P2S) VPN and Site-to-Site (S2S) VPN within Microsoft Azure.

We delved into:

* **Point-to-Site VPN:** Understanding its role in enabling secure, client-based access for individual users to Azure Virtual Networks without the need for on-premises VPN hardware. We covered its step-by-step setup, from VNet creation to certificate generation and client installation, highlighting its suitability for remote workers, developers, and administrators.
* **Site-to-Site VPN:** Examining how it connects entire on-premises networks to Azure VNets, forming persistent, encrypted tunnels. Through a practical simulation using Hyper-V and Windows Server's Routing and Remote Access Service (RRAS), we walked through the configurations required on both the Azure and on-premises sides, demonstrating its value for enterprise-grade hybrid architectures.

These VPN configurations are not merely theoretical concepts but are extensively utilized in real-world scenarios, including:

* **Hybrid Cloud Setups:** Seamlessly extending on-premises data centers into Azure, allowing applications to leverage cloud scalability while accessing on-premises resources.
* **Disaster Recovery (DR):** Establishing secure links for data replication and rapid failover between on-premises environments and Azure DR sites.
* **DevOps Pipelines:** Providing secure access for development and testing teams to isolated cloud environments.
* **Secure Development Environments:** Ensuring that sensitive development resources in Azure are only accessible via encrypted and authenticated channels.