Project Title: \*Secure Remote Access Setup and Configuration for Virtual Machines

Duration: 3 weeks

\*Objective:\*

The objective of this project is to enable you to understand, configure, and secure remote access to virtual machines (VMs) running Linux operating systems. This group will be responsible for setting up a VM, ensuring it is securely accessible from a physical computer, and documenting the process and security measures.

\*Project Tasks:\*

\*Group A: Linux VM Configuration and Access\*

1. \*Virtual Machine Setup:\*

- Create a Linux-based virtual machine using your Azure portal

- Configure the network settings of the VM to ensure it is accessible from the network.

2. \*User Account Management:\*

- Create user accounts on the Linux VM for each group member.

- Assign appropriate permissions and configure sudo privileges as necessary.

3. \*Enable Secure Shell (SSH):\*

- Ensure SSH is installed and running on the Linux VM.

- Configure the firewall to allow SSH connections (default port TCP/22).

4. \*Generate and Deploy SSH Keys:\*

- Generate SSH key pairs for each group member.

- Deploy the public keys to the ~/.ssh/authorized\_keys file on the Linux VM.

5. \*Secure the Connection:\*

- Set up a Virtual Private Network (VPN) or Network Security Group (NSG) to restrict access to the VM to specific IP addresses.

- Implement SSH key-based authentication and disable password-based logins for SSH.

- Document the steps taken to secure the VM and the SSH connection.

6. \*Test Remote Access:\*

- Each group member should test remote access to the Linux VM using SSH.

- Troubleshoot and resolve any connection issues.

7. \*Monitoring and Logging:\*

- Enable and configure logging on the Linux VM to track remote access attempts (syslog or other logging service).

- Monitor the logs for any unauthorized access attempts and document the findings.

8. \*Documentation:\*

- Prepare a comprehensive report detailing the setup, security measures, and testing process.

- Include screenshots and command outputs where applicable.

\*Deliverables:\*

1. \*Group Report:\*

- Each person in this group should submit a detailed report of their work, including the following:

- VM setup process.

- User account creation and management.

- Security measures implemented.

- Remote access testing procedures and results.

- Monitoring and logging configuration.

- Any issues encountered and how they were resolved.

2. \*Presentation:\*

- This group should prepare a presentation summarizing their findings and demonstrating how they configured and secured the VM.

- The presentation should include a live demo or video showing the remote access process.

3. \*Peer Review:\*

- You are to peer-review each other's work, providing feedback on the setup, security measures, and documentation.

Project Title: \*Secure Remote Access Setup and Configuration for Virtual Machines

Duration: 3 weeks

\*Objective:\*

The objective of this project is to enable you to understand, configure, and secure remote access to virtual machines (VMs) running Linux operating systems. This group will be responsible for setting up a VM, ensuring it is securely accessible from a physical computer, and documenting the process and security measures.

\*Project Tasks:\*

\*Group A: Linux VM Configuration and Access\*

1. \*Virtual Machine Setup:\*

- Create a Linux-based virtual machine using your Azure portal

- Configure the network settings of the VM to ensure it is accessible from the network.

Solution

**Step 1: Sign in to Azure Portal**

1. Go to the Azure portal and sign in with your Azure account.

**Step 2: Create a Virtual Machine**

1. In the Azure portal, search for **Virtual machines** in the search bar and select it.
2. Click on **Create** and then **Azure virtual machine**.

**Step 3: Configure Basic Settings**

1. **Subscription**: Select your subscription.
2. **Resource group**: Create a new resource group or select an existing one.
3. **Virtual machine name**: Enter a name for your VM.
4. **Region**: Choose the region where you want to deploy the VM.
5. **Image**: Select the Linux distribution you prefer (e.g., Ubuntu Server 22.04 LTS).
6. **Size**: Choose the size of the VM based on your requirements.
7. **Authentication type**: Select Password.
8. **Username**: Enter a username.
9. **Password:** Enter a password

**Step 4**: Configure Disk

1. **VM disk encryption:** Check the box for Encryption at the host. However, my subscription do not support it.
2. OS disk size: select the disk size for the VM.
3. OS disk type: Select the type of disk for the VM depending on your workload. SSD preferably.
4. Delete with VM: check the box if you want it to be deleted with VM.
5. Key management: select the key management of your choice.

**Step 5: Configure Networking**

1. **Virtual network**: Create a new virtual network or select an existing one.
2. **Subnet**: Create a new subnet or select an existing one.
3. **Public IP**: Ensure a public IP address is assigned to the VM.
4. **NIC network security group**: Select **Basic** and allow SSH (22) and HTTP (80) inbound ports.

**Step 5: Review and Create**

1. Click on **Review + create**.
2. Review all the settings and click on **Create**.

**Step 6: Connect to Your VM**

1. Once the VM is created, go to the **Virtual machines** section and select your VM.
2. Copy the public IP address of your VM.
3. Use an SSH client to connect to your VM:
4. ssh username@your\_vm\_public\_ip

**Configuration Parameter**

**Basics Configuration**

**Subscription**: Azure subscription 1

**Resource group**: project\_group\_a

**Virtual machine name**: Project

**Region**: West US 2

**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 24.04 LTS - Gen2

**VM architecture**: x64

**Size**: Standard B1s (1 vcpu, 1 GiB memory)

**Enable Hibernation**: No

**Authentication type**: Password

**Username**: group\_a@[52.143.68.146](https://portal.azure.com/" \t "_blank)

**pass** : group\_a12345

**Public inbound ports**: SSH, RDP

**Azure Spot**: No

**Disks Configuration**

**OS disk size:** Image default

**OS disk type**: Premium SSD LRS

**Use managed disks**: Yes

**Delete OS disk with VM**: Enabled

**Ephemeral OS disk**: No

**Networking Configuration**

**Virtual network**: (new) Project-vnet

**Subnet**: (new) default (10.0.0.0/24)

**Public IP**: (new) Project-ip

**Accelerated networking**: Off

**Place this virtual machine behind an existing load balancing solution?** No

**Delete public IP and NIC when VM is deleted**: Disabled

**Management**

**Microsoft Defender for Cloud**: Basic (free)

**System assigned managed identity**: Off

**Login with Microsoft Entra ID**: Off

**Auto-shutdown**: Off

**Backup**: Disabled

**Enable hotpatch**: Off

**Patch orchestration options**: Image Default

**Monitoring Configuration**

**Alerts**: Off

**Boot diagnostics**: On

**Enable OS guest diagnostics**: Off

**Enable application health monitoring**: Off

**Advanced**

**Extensions**: None

**VM applications**: None

**Cloud init**: No

**User data**: No

**Disk controller type**: SCSI

**Proximity placement group**: None

**Capacity reservation group**: None

2. \*User Account Management:\*

- Create user accounts on the Linux VM for each group member.

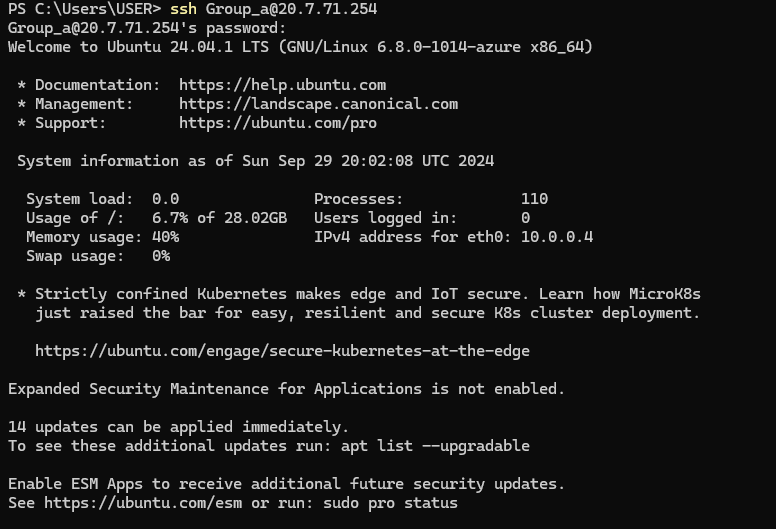
- Assign appropriate permissions and configure sudo privileges as necessary.

Solution

Here’s a step-by-step guide to help you set up user accounts for each group member:

**Step 1: Connect to Your Linux VM**

First, you need to connect to your Linux VM using password. You can do this from your local machine’s terminal or using Azure Cloud Shell.



**Step 2: Create Folder and sub folder for User Accounts**

# Make folder called matthew and a subfolder called .ssh

#.ssh subfolder is where the authorised keys for ssh is stored

sudo mkdir -p /home/matthew/.ssh

**Step3: Create an empty file called authorized\_keys in sub folder for User Accounts**

# create an empty file called authorized\_key where the ssh key will be stored

sudo touch /home/matthew/.ssh/authorized\_keys

**Step4: Create an administrator User Accounts**

# Create an administrator user account

sudo useradd -d /home/matthew matthew

**Step 5: Set Password for the New User**

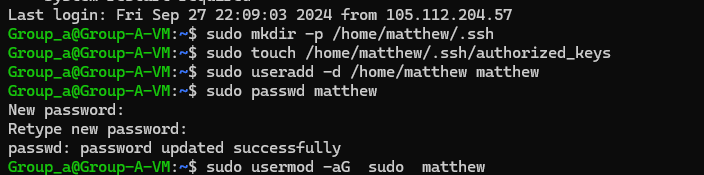
#set the user password

Command: sudo passwd matthew

**Step 7: Grant the user a Sudo privileged as an administrator**

#Add all user in group A to sudo group

Command: sudo usermod -aG sudo matthew



# To Verify User Belongs to Sudo Group

Command: groups matthew or sudo cat /etc/group

# To know the group User Belongs to

Command: Id



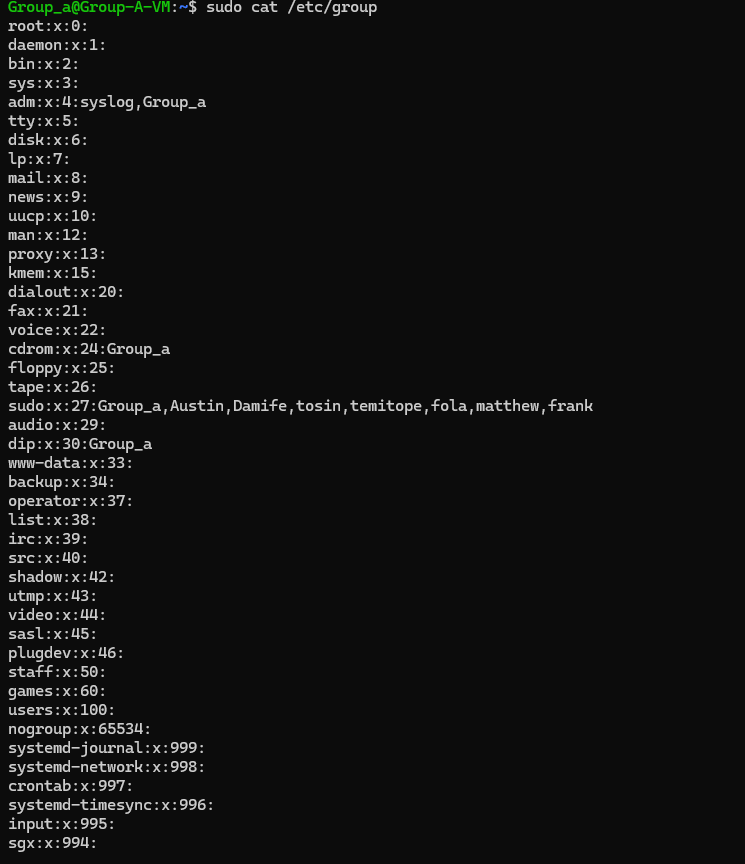
**Step 8**

# Switch to your user account.

Command: su matthew (it prompts you to put in your password)

A computer screen with white text

Description automatically generated



**Step 9: Change Ownership for Directory**

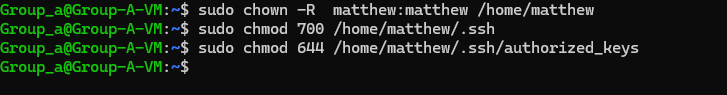
# To assign ownership or changing ownership for the directory for each user

Command: sudo chown -R matthew:matthew /home/matthew

**Step 10: Grant user Permission require for directory and file**

Command: sudo chmod 700 /home/matthew/.ssh

Command: sudo chmod 644 /home/matthew/.ssh/authorized\_keys



**Step 11: Verify User Creation**

You can verify that the user has been created and check their details using the id command:

id new\_username

A screenshot of a computer

Description automatically generated



A black screen with white text

Description automatically generated



**A screenshot of a computer program

Description automatically generated**

3. \*Enable Secure Shell (SSH):\*

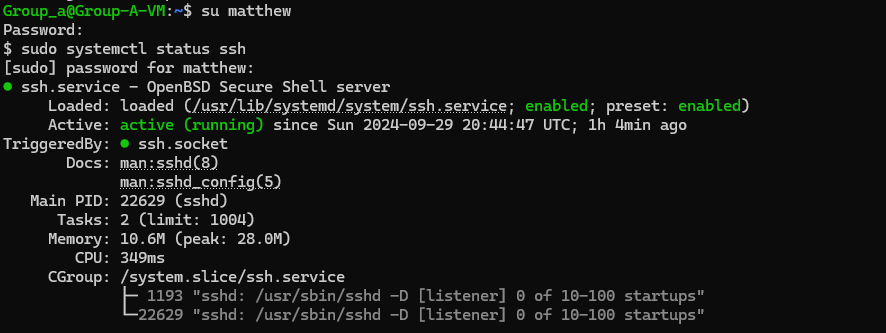
- Ensure SSH is installed and running on the Linux VM.

- Configure the firewall to allow SSH connections (default port TCP/22).

Solutiion

**Step 1: Ensure SSH is Installed and Running**

1. **Connect to your VM**:
2. ssh username@your\_vm\_public\_ip
3. **Check if SSH is installed**:
4. sudo systemctl status ssh
5. Enter the Ctrl + C to go back to command prompt



If SSH is not installed, you can install it using:

sudo apt update

sudo apt install openssh-server

1. **Start and enable the SSH service**:
2. sudo systemctl start ssh
3. sudo systemctl enable ssh

**Step 2: Configure the Firewall to Allow SSH Connections**

1. **Check the status of the firewall**:
2. sudo ufw status
3. **Allow SSH connections (default port TCP/22)**:
4. sudo ufw allow ssh

Alternatively, you can specify the port explicitly:

sudo ufw allow 22/tcp

1. **Enable the firewall**:
2. sudo ufw enable
3. **Verify the firewall rules**:
4. sudo ufw status

A black background with white text

Description automatically generated

4. \*Generate and Deploy SSH Keys:\*

- Generate SSH key pairs for each group member.

- Deploy the public keys to the ~/.ssh/authorized\_keys file on the Linux VM.

Solution

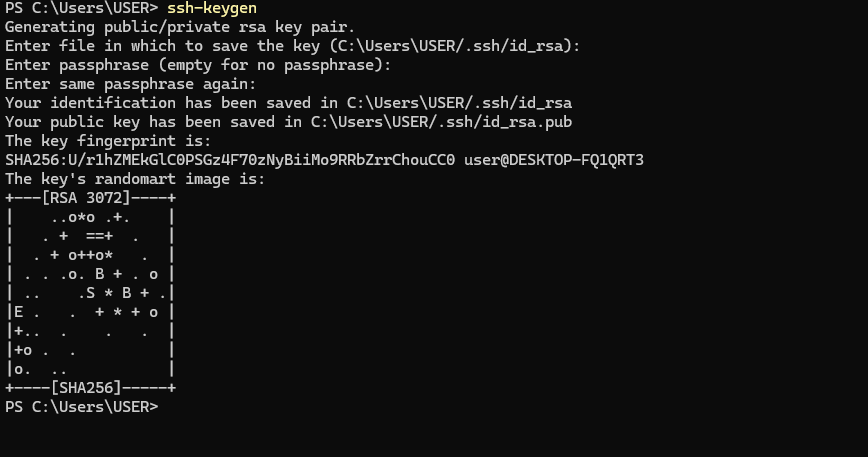
Here’s how each group member generates SSH key pairs on your local machine and deploy the public keys to the ~/.ssh/authorized\_keys file on the Linux VM:

**Step 1: Generate SSH Key Pairs**

Each group member needs to generate their own SSH key pair on their local machine. Here’s how to do it:

1. **Open a terminal** on the local machine. e.g Powershell
2. **Generate the SSH key pair**:

Command: ssh-keygen



1. **Follow the prompts** to save the key pair. By default, it will be saved in ~/.ssh/id\_rsa and ~/.ssh/id\_rsa.pub by press Enter key on the keyboard.

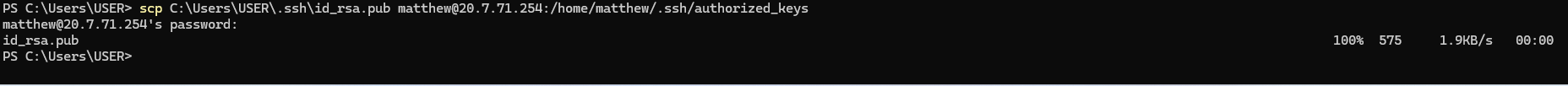
**Step 2: Copy the Public Key to the VM**

Each group member needs to copy their public key to their directory that is assigned to thmem ~/.ssh/authorized\_keys file on the Linux VM. Here’s how:

1. **Use the**scp(secure copy)**command** to copy the public key:
2. Scp C:\~/.ssh/id\_rsa.pub username@ your\_vm\_Public IP address: home/your directory/.ssh/authorized\_keys

Example:

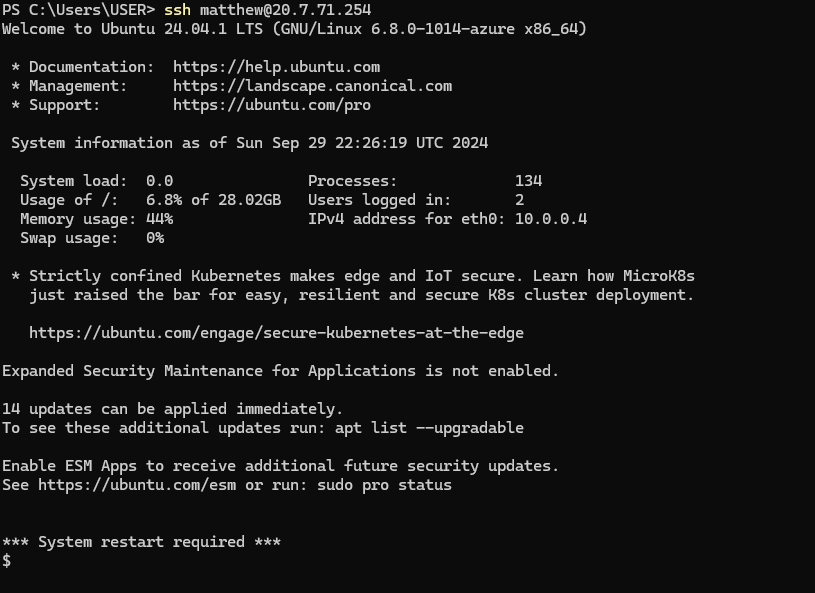
Command scp C:\Users\USER\.ssh\id\_rsa.pub [matthew@20.7.71.254:/home/matthew/.ssh/authorized\_keys](mailto:matthew@20.7.71.254:/home/matthew/.ssh/authorized_keys)



**Step 3: Verify the Setup**

1. **Connect to the VM using SSH**:
2. ssh username@your\_vm\_public\_ip

If the setup is correct, you should be able to log in without being prompted for a password



5. \*Secure the Connection:\*

- Set up a Virtual Private Network (VPN) or Network Security Group (NSG) to restrict access to the VM to specific IP addresses.

- Implement SSH key-based authentication and disable password-based logins for SSH.

- Document the steps taken to secure the VM and the SSH connection.

Solution

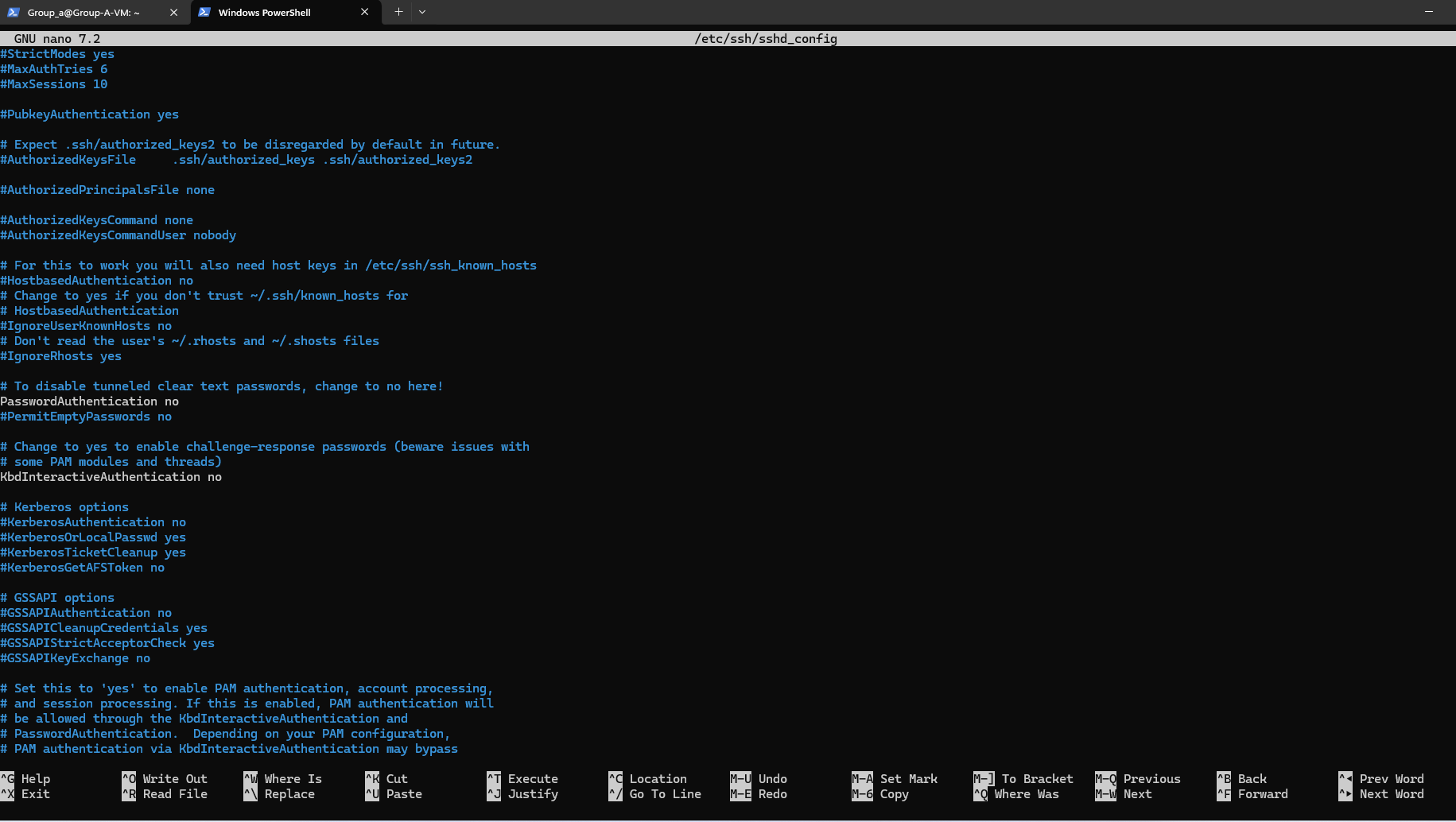
Let’s go through the steps to set up a Network Security Group (NSG) to restrict access to your VM, implement SSH key-based authentication, disable password-based logins, and document the entire process.

**Step 1: Set Up a Network Security Group (NSG)**

1. **Create an NSG**:
   * In the Azure portal, search for **Network security groups** and select it.
   * Click on **Create**.
   * Fill in the required details such as **Subscription**, **Resource group**, **Name**, and **Region**.
   * Click on **Review + create** and then **Create**.
2. **Add Inbound Security Rules**:
   * Go to your newly created NSG.
   * Under **Settings**, select **Inbound security rules**.
   * Click on **Add**.
   * Configure the rule to allow SSH (port 22) from specific (Public) IP addresses:
     + **Source**: IP Addresses
     + **Source IP addresses/CIDR ranges**: Enter the specific IP addresses or ranges.
     + **Destination**: Any
     + **Destination port ranges**: 22
     + **Protocol**: TCP
     + **Action**: Allow
     + **Priority**: Set a priority (e.g., 100).
     + **Name**: Give the rule a name (e.g., Allow-SSH).
3. **Associate NSG with VM’s Network Interface**:
   * Go to your VM’s **Networking** settings.
   * Under **Network interface**, select the network interface associated with your VM.
   * Under **Settings**, select **Network security group**.
   * Click on **Associate** and select the NSG you created.

**Step 3: Disable Password-Based Logins**

1. **Edit SSH Configuration**:
   * Connect to your VM:
   * ssh username@your\_vm\_public\_ip
   * Open the SSH configuration file:
   * sudo nano /etc/ssh/sshd\_config
   * Find the line #PasswordAuthentication yes and change it to:
   * PasswordAuthentication no
   * Save and exit the editor.



1. **Restart SSH Service**:
   * Restart the SSH service to apply the changes:
   * sudo systemctl restart ssh