**Deploy Jenkins on an Azure VM — Step‑by‑Step (Windows + Git Bash)**

This guide documents exactly what you did (create Resource Group → create VM → SSH from Windows Git Bash → install Jenkins → test locally → open Azure Networking port rules) and adds best‑practice notes and troubleshooting tips.

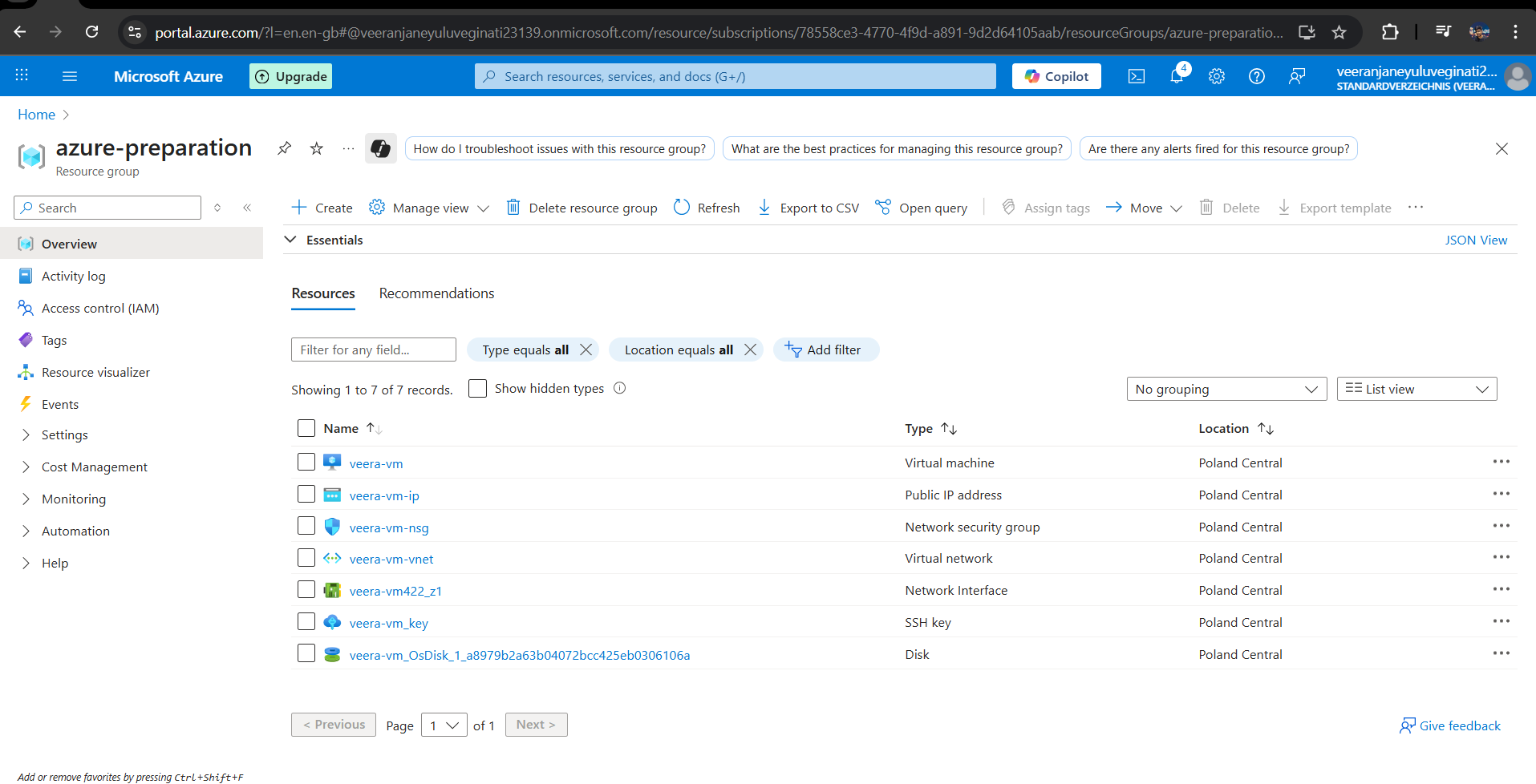
**0) Prerequisites**

* An **Azure account** with permission to create resources.
* Your **Windows laptop** with **Git Bash** installed.
* An **SSH private key** (.pem) that corresponds to the public key on the VM, or you will generate one during VM creation.
* Recommended OS for VM: **Ubuntu 22.04 LTS** or **Ubuntu 24.04 LTS**.

Placeholders you’ll replace below:

* <RESOURCE\_GROUP> — your resource group name (e.g., rg-jenkins-demo).
* <VM\_NAME> — your VM name (e.g., vm-jenkins-ubuntu).
* <PUBLIC\_IP> — your VM’s public IP (shown on the VM Overview page).
* <PATH\_TO\_PEM> — your local key path (e.g., /c/Users/veera/Downloads/veera-vm\_key.pem).
* Default Linux username used below: azureuser (change if you set a different one).

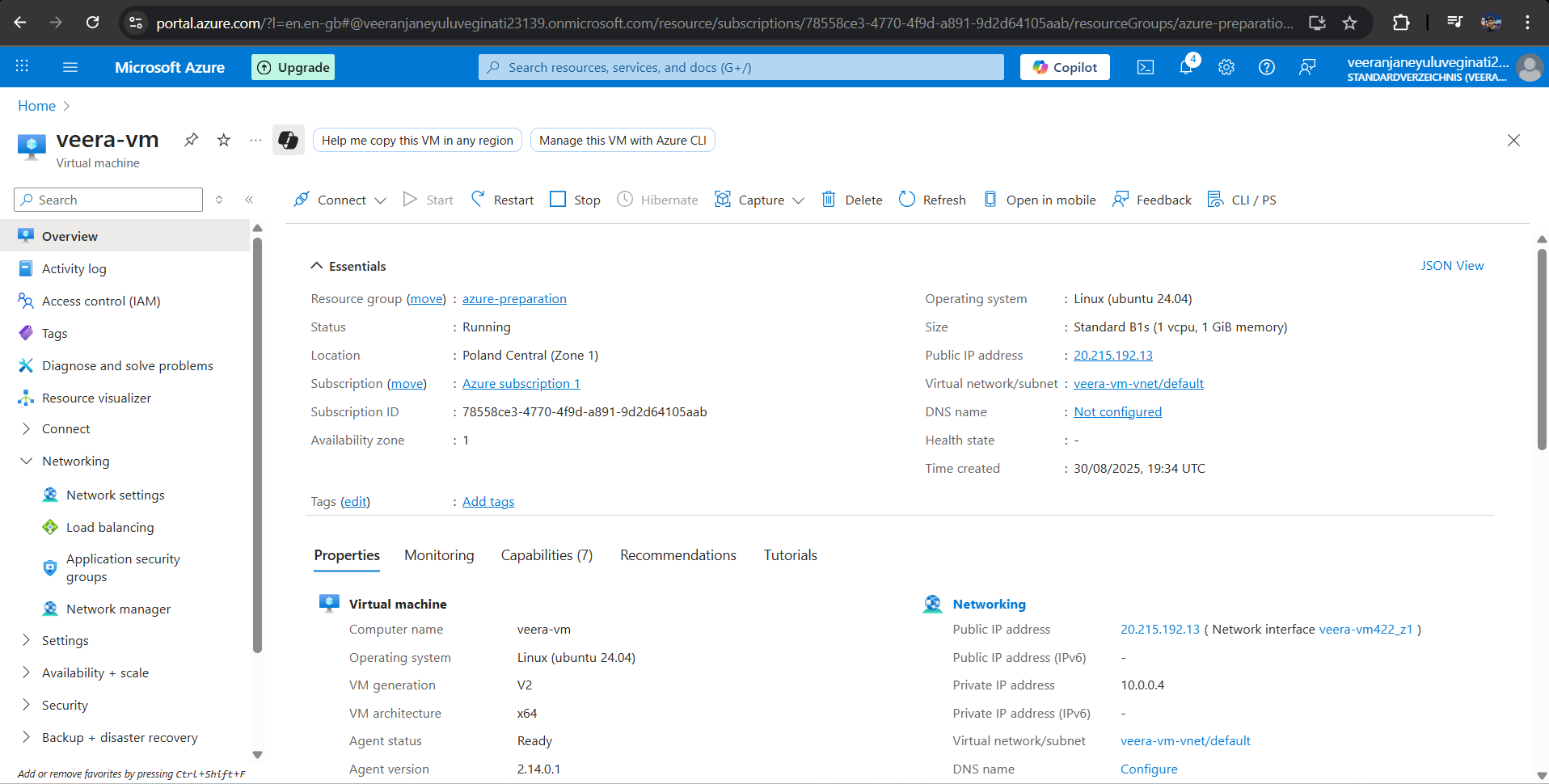
**1) Create a Resource Group (Azure Portal)**

1. Sign in to **Azure Portal**.
2. Search **Resource groups** → **Create**.
3. **Subscription**: choose yours.
4. **Resource group**: <azure\_prepration>.
5. **Region**: closest to you (e.g., West Europe).
6. **Review + create** → **Create**.  
     
   

**2) Create an Ubuntu VM (Azure Portal)**

1. Go to **Virtual machines** → **Create** → **Azure virtual machine**.
2. **Basics**
   * Subscription: your subscription.
   * Resource group: <RESOURCE\_GROUP>.
   * Virtual machine name: <VM\_NAME>.
   * Region: same as RG.
   * Image: **Ubuntu Server 22.04 LTS** (or 24.04 LTS).
   * Size: start with Standard\_B2s (2 vCPU, 4 GB RAM) for Jenkins demo.
   * Authentication type: **SSH public key**.
   * Username: azureuser (or your choice).
   * SSH public key source: **Generate new key pair** *or* **Use existing public key** (paste your .pub content).
3. **Inbound port rules** (Basics tab)
   * Select **Allow selected ports**.
   * Check **SSH (22)** only for now (we’ll open 8080 later to keep it secure during setup).
4. **Disks**: Default Standard SSD is fine for demos.
5. **Networking**: Keep defaults; ensure an NSG is associated (Basic networking includes one automatically).
6. **Review + create** → **Create**. If you generated a key pair, **download the private key** (.pem).

Tip: If you already have a .pem, make sure your **public** key was used during VM creation. Private key stays on your laptop.



**3) Prepare your SSH key on Windows (Git Bash)**

* Convert Windows path to Git Bash (Unix-style):
  + C:\Users\veera\Downloads\veera-vm\_key.pem → /c/Users/veera/Downloads/veera-vm\_key.pem
* Restrict file permissions so SSH accepts the key:

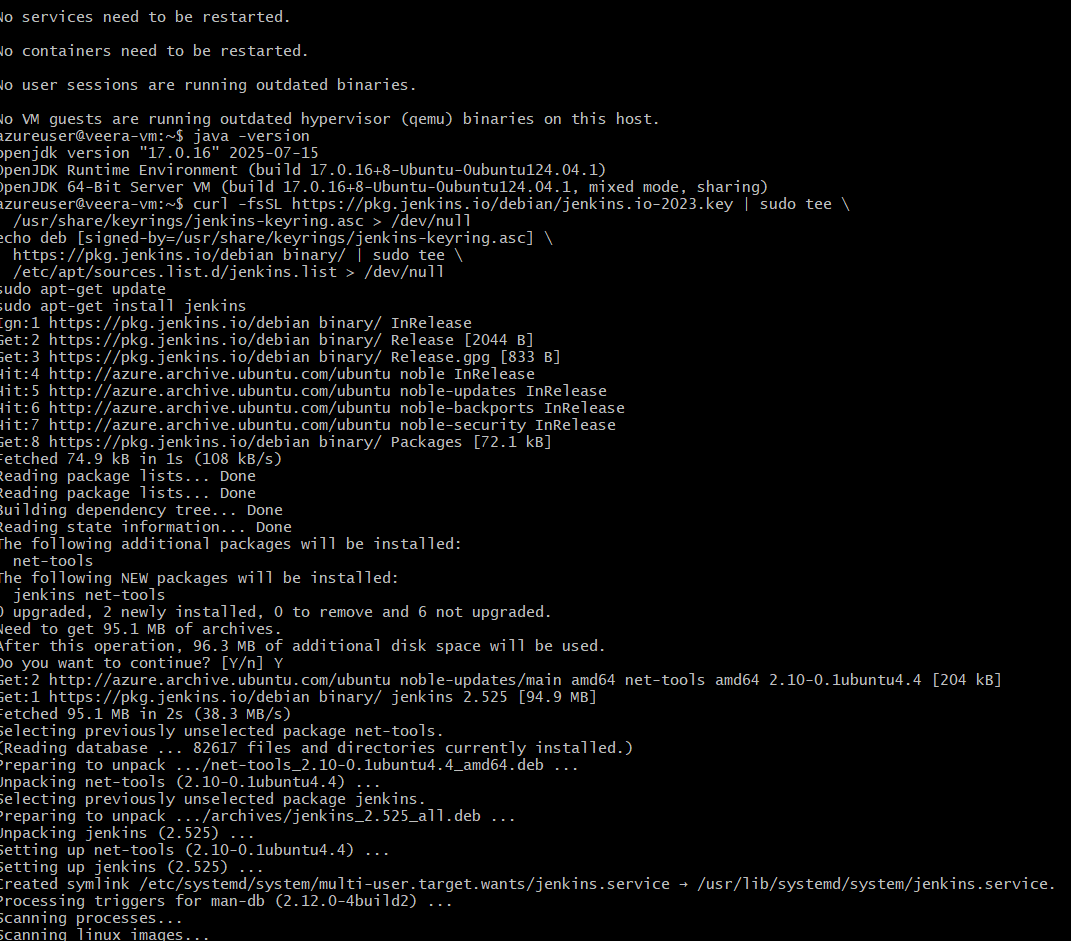
chmod 600 /c/Users/veera/Downloads/veera-vm\_key.pem

If you see **“Unprotected private key file”** on Windows, open **File → Properties → Security → Advanced**, remove unnecessary users, keep only your account with Full control.

**4) SSH into the VM**

Find the VM’s **Public IP** on the VM’s **Overview** page, then:

ssh -i /c/Users/veera/Downloads/veera-vm\_key.pem azureuser@20.215.192.13

First connection will ask to trust the host fingerprint → type yes.  
  
  


**5) Update the system**

sudo apt update && sudo apt -y upgrade

sudo reboot

Reconnect via SSH after the reboot:

ssh -i /c/Users/veera/Downloads/veera-vm\_key.pem azureuser@<PUBLIC\_IP>

**6) Install Java (Jenkins requires Java 17+)**

sudo apt update

sudo apt install -y fontconfig openjdk-17-jre

java -version

**7) Add Jenkins repository and install Jenkins**

# Import Jenkins repo key

curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | \

sudo tee /usr/share/keyrings/jenkins-keyring.asc > /dev/null

# Add repo

echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-stable binary/" | \

sudo tee /etc/apt/sources.list.d/jenkins.list > /dev/null

# Install Jenkins

sudo apt update

sudo apt install -y jenkins

# Enable and start service

sudo systemctl enable --now jenkins

# Verify

systemctl status jenkins --no-pager

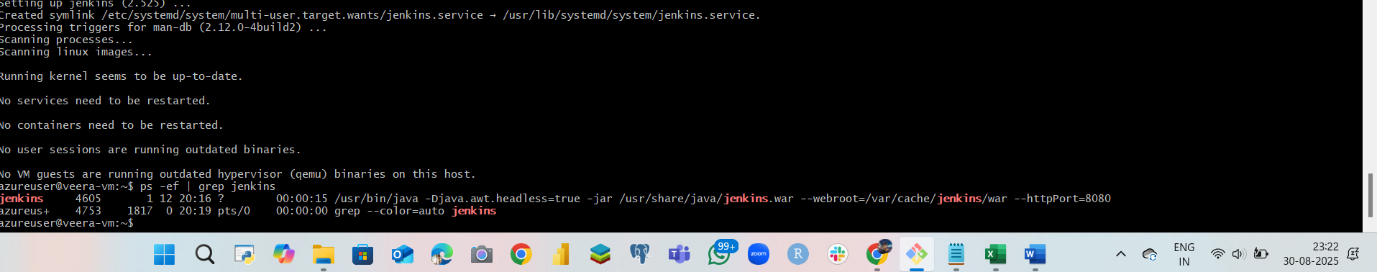
You should see active (running).

**8) Test Jenkins locally on the VM**

Check that the service responds on localhost (port **8080** by default):

curl -I http://localhost:8080

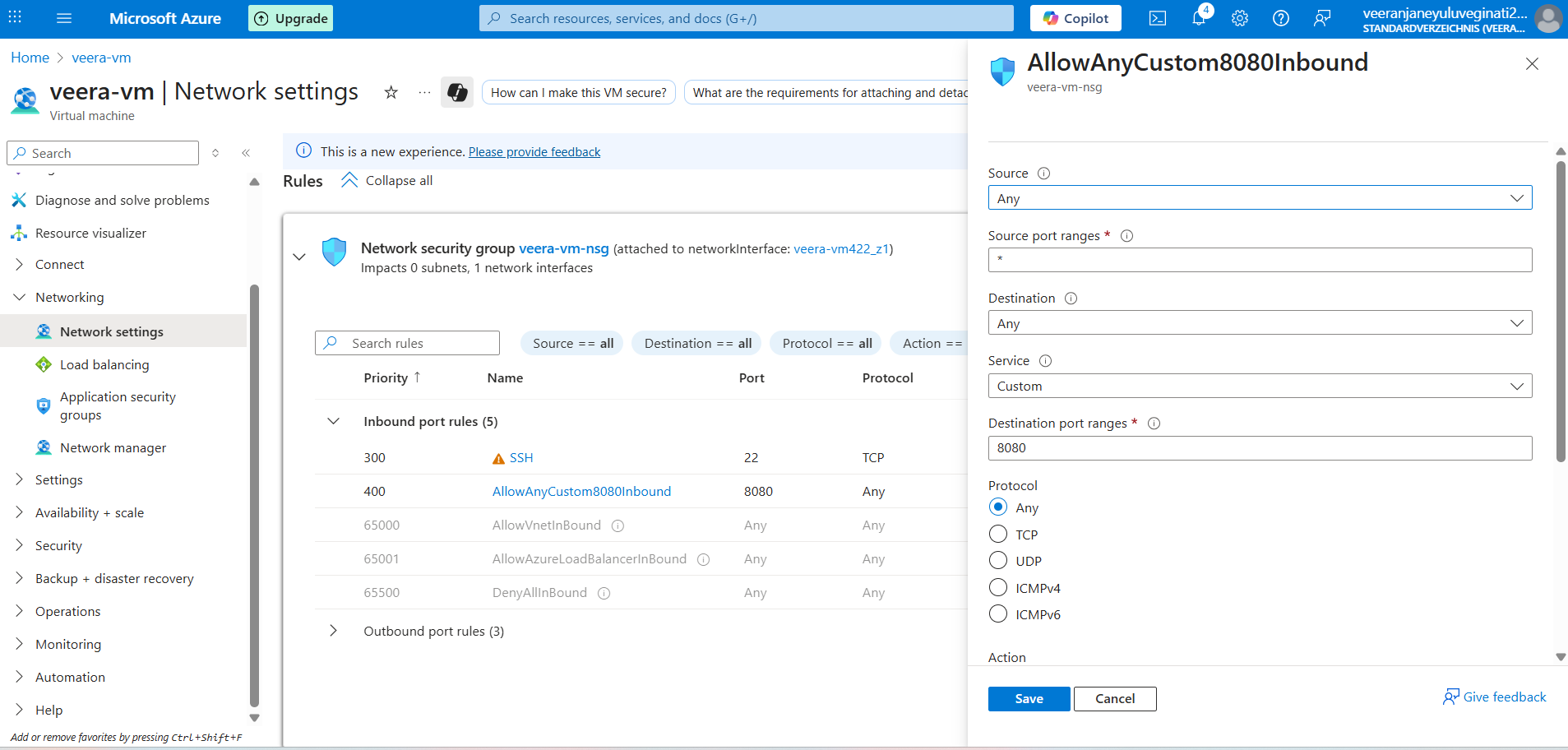
You should see an HTTP response (e.g., HTTP/1.1 403 or 200), which confirms Jenkins is listening.

jfje  
  
  
  
  
  
  
**9) Open Azure Networking to allow external access (port 8080)**

**From Azure Portal:**

1. Open your VM → **Networking**.
2. Under **Inbound port rules** → **Add inbound port rule**.
3. Set:
   * **Source**: Any *(or restrict to your IP for better security)*
   * **Source port ranges**: \*
   * **Destination**: Any
   * **Service**: Custom
   * **Protocol**: TCP
   * **Destination port ranges**: 8080
   * **Action**: Allow
   * **Priority**: e.g., 1001 (must be unique and lower than broad deny rules)
   * **Name**: allow-jenkins-8080
4. **Add** the rule.

Ubuntu’s default is to have UFW disabled; if you enabled it, also run:



**10) Access Jenkins from your laptop**

Open your browser and go to:

http://<PUBLIC\_IP>:8080

You’ll see the **Unlock Jenkins** page.

**11) Unlock Jenkins and initial setup**

1. SSH into the VM (if not already):

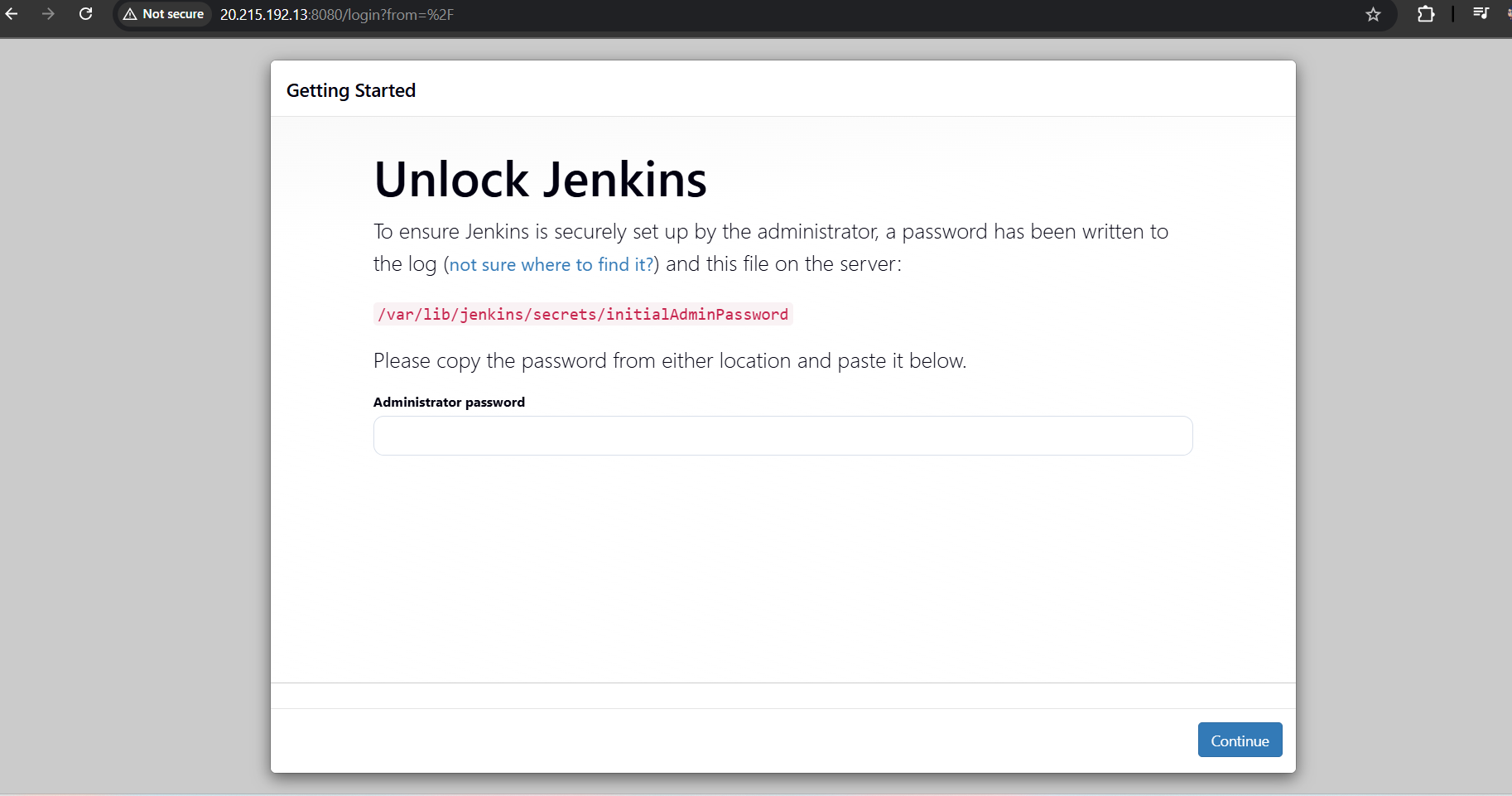
ssh -i /c/Users/veera/Downloads/veera-vm\_key.pem azureuser@<PUBLIC\_IP>

1. Get the initial admin password:

sudo cat /var/lib/jenkins/secrets/initialAdminPassword

1. Paste it into the **Unlock Jenkins** page.
2. Choose **Install suggested plugins**.
3. Create your **first admin user**.
4. Confirm **Jenkins URL** (use http://20.215.192.13:8080/ for now).

You now have a working Jenkins controller.

;;;;  
  
**How to use Jenkins (quick start)**

* Connect to your source control (GitHub/GitLab/Bitbucket) via **Manage Jenkins → Plugins** (install Git-related plugins as needed).
* Create a **Freestyle** or **Pipeline** job.
* For GitHub webhooks: expose Jenkins at a stable URL (public IP works, but a domain + HTTPS is recommended) and add a webhook pointing to http://<PUBLIC\_IP>:8080/github-webhook/.

**13) Recommended hardening (after it works)**

* **Restrict Azure NSG rule** for port 8080 to **your office/home IP** instead of Any.
* Put Jenkins **behind Nginx reverse proxy** and enable **HTTPS** (Let’s Encrypt).
* Create a non‑admin user for daily use; keep the admin account only for administration.
* Regularly update: sudo apt update && sudo apt -y upgrade and keep plugins up to date.

**14) Troubleshooting**

**SSH key path errors on Windows**

* Use Unix path in Git Bash: C:\Users\veera\Downloads\x.pem → /c/Users/veera/Downloads/x.pem.
* chmod 600 /c/Users/.../x.pem and ensure Windows file permissions restrict access to your user.

**Permission denied (publickey)**

* Ensure the **public key** used during VM creation matches your **private key**.
* Verify username: Azure Ubuntu defaults to azureuser (unless you chose a different one).

**Jenkins service not running**

sudo systemctl status jenkins

sudo journalctl -u jenkins -n 100 --no-pager

* Confirm Java 17+: java -version.

**Can’t reach http://<PUBLIC\_IP>:8080**

* Azure Portal → VM → **Networking**: confirm inbound **Allow TCP 8080** rule exists and is above any deny rules.
* If you enabled UFW: sudo ufw allow 8080/tcp.
* From the VM: curl -I http://localhost:8080 to confirm Jenkins is listening.

**Port already in use**

sudo lsof -i :8080

* Change Jenkins port in /etc/default/jenkins (HTTP\_PORT=8081) → sudo systemctl restart jenkins and update NSG to match.

**15) Clean up to avoid costs**

* Stop VM when not in use: Azure Portal → VM → **Stop** (deallocate).
* To remove everything: delete the **resource group** <RESOURCE\_GROUP> (this deletes the VM, disk, NIC, IP, NSG, etc.).

**16) Summary — What, Why, How**

* **What is it?** Jenkins is a CI/CD automation server running on your Azure VM.
* **Why Azure VM?** Always‑on, scalable, accessible from anywhere, separated from your laptop.
* **How to access?** http://<PUBLIC\_IP>:8080 after opening port 8080 in the VM’s NSG; login with your Jenkins admin user.

You now have a clear, reproducible path from Azure resource creation to a reachable Jenkins instance.