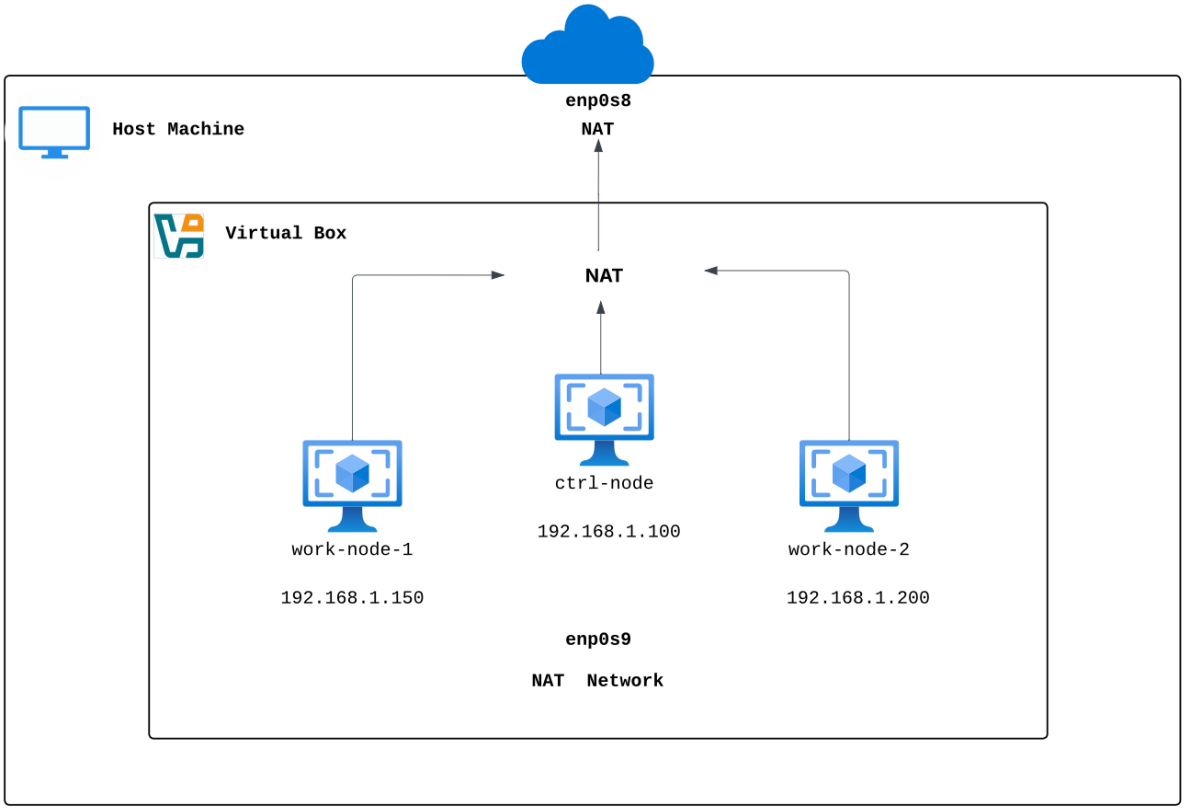
Linux VM Cluster

By: Nadia H.A.

Table of Contents

Network Topology



Project Objectives

- Create and configure three Ubuntu Server virtual machines in VirtualBox.
- Establish a dual network setup using NAT (internet access) and an internal network with Static IPs.
- Configure `/etc/hosts` on each node for name resolution.
- Set up SSH key-based access from the ctrl node to both work nodes.
- Validate connectivity to ensure cluster readiness.

Overview

This project focuses on designing and configuring a virtualized cluster of machines to simulate a distributed computing environment. The cluster comprises three virtual nodes:

- `ctrl-node`
- `work-node-1`
- `work-node-2`

Using **VirtualBox** and **Ubuntu Server**, this project walks through the process of:

- Creating virtual machines with consistent naming and system configuration
- Establishing isolated internal networking for inter-node communication
- Assigning static IPs and resolving hostnames
- Setting up passwordless SSH access from the master to nodes for secure remote management and automation

Purpose

The core purpose of this project is to provide a hands-on, foundational experience in setting up the infrastructure needed for distributed systems.

- **Understand Distributed Systems**: Learn the architecture, purpose, and advantages of distributed environments
- **Develop Virtualization Skills**: Set up and manage multiple VMs using VirtualBox
- **Configure Networking**: Establish NAT and Internal Network configurations to simulate multi-node communication
- **Implement Secure Remote Access**: Use SSH key-based authentication for automated, passwordless communication between nodes
- **Prepare for Real Deployments**: Build an environment capable of hosting distributed applications.

Phase 0: Prerequisites and Planning

1. Required Software

- **Oracle VirtualBox**
- **Ubuntu Server ISO Image**
  - Download the latest LTS version
  - *Note:* Used the **Server edition**—it's more lightweight and suitable for virtualization.

2. Naming Convention and User Setup

- **Virtual Machines (VM) Hostnames:**
  - `ctrl-node`
  - `work-node-1`
  - `work-node-2`
- **Username (consistent across all nodes)**

3. Network Configuration

Adapter 1: Internet Access

- **Mode:** NAT (Network Address Translation)
- **IP Assignment:** Automatic via VirtualBox DHCP (typically 10.0.2.x)
- **Configuration:** No static IP needed within Ubuntu for this adapter.

Adapter 2: Internal Cluster Network

- **Mode:** Internal Network (VirtualBox)
- **Static IP Address Scheme (Manual Configuration in Ubuntu):**
  - `ctrl-node: 192.168.1.100`
  - `work-node-1: 192.168.1.150`
  - `work-node-2: 192.168.1.200`

Phase 1: VirtualBox Network Setup – Internal Network

Create and Use a Consistent Internal Network Identifier

- VirtualBox automatically creates an Internal Network when a name is specified in a VM's network settings, must use the exact same name for all VMs to ensure proper communication.
- No additional configuration is required at the VirtualBox global settings level for internal networks.

Note: *Internal Networks in VirtualBox function like an isolated LAN for your VMs. They are only accessible between VMs using the same internal network name.*

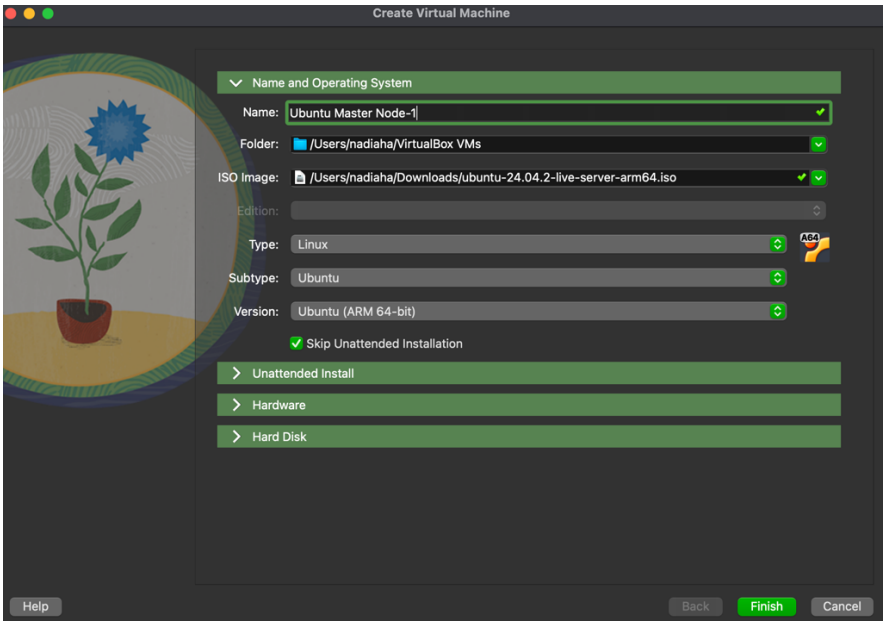
Phase 2: Virtual Machine Creation and Initial Configuration

Repeat the following steps for each VM:

- `ctrl-node`
- `work-node-1`
- `work-node-2`

1. Create the Virtual Machine

1. Open **VirtualBox Manager**.
2. Click **New** to create a new VM.
3. Fill out the initial setup:
    - **Name:** Use the appropriate name for the node (e.g., master-node).
    - **ISO Image:** Choose your downloaded **Ubuntu Server ISO**.
    - *(Recommended)* Check **"Skip Unattended Installation"** to manually control the installation process.
4. Click **Next** to proceed.



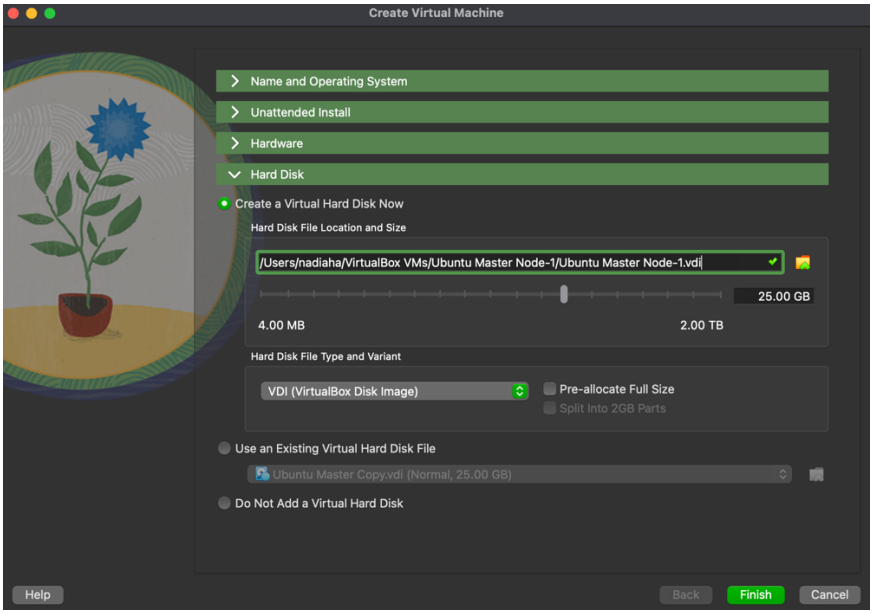
2. Configure Hardware Resources

- **Base Memory (RAM):**
  Allocate at least **2048 MB**.
- **Processors:**
  Assign **2 CPUs** if available for better performance.

3. Set Up the Virtual Hard Disk

- Choose **"Create a virtual hard disk now"**.
- Recommended settings:
  - **Disk Size:** At least **25 GB**.
  - **File Type:** VDI (VirtualBox Disk Image).
  - **Storage: Dynamically allocated**.

Click **Create** once the configuration is complete.

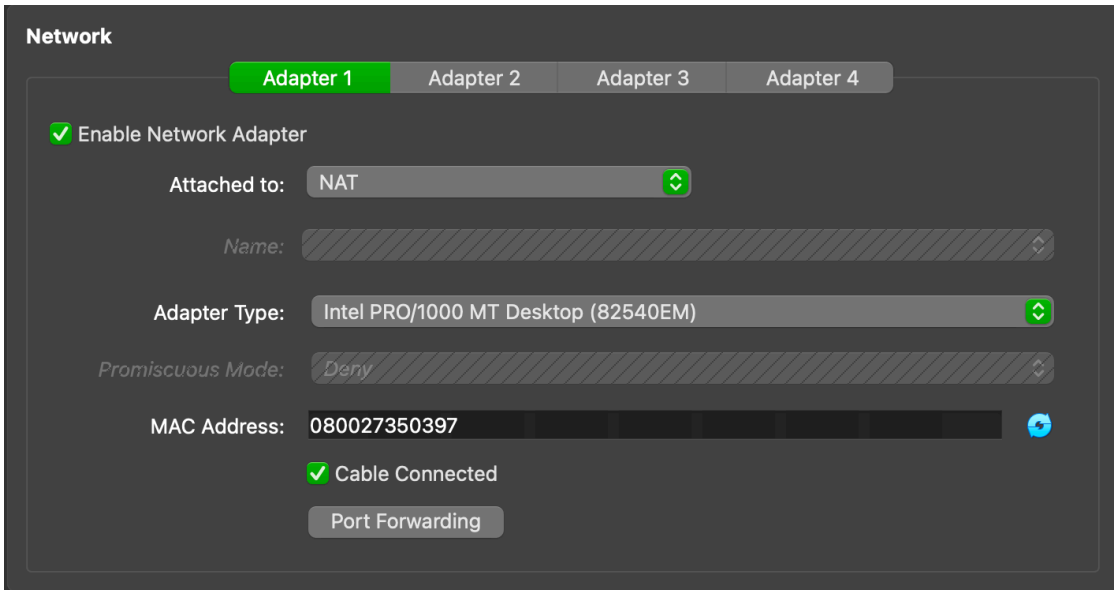


4. Configure Network Adapters (Before First Boot)

1. In **VirtualBox Manager**, select the newly created VM.
2. Click **Settings → Network**.
3. Configure the two adapters:

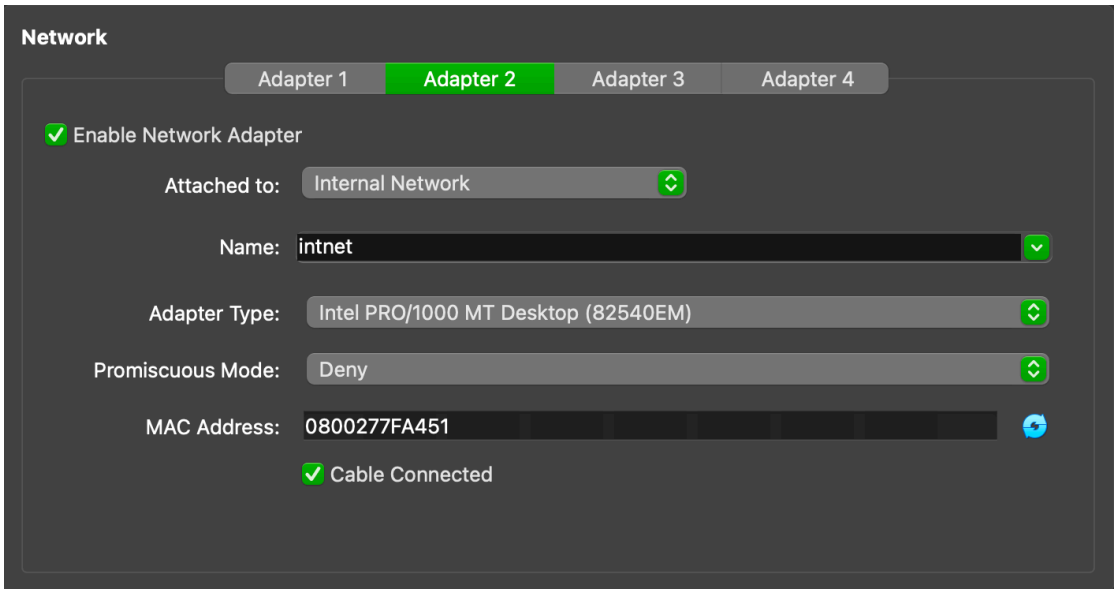
   Adapter 1 – Internet Access

   - **Enable Network Adapter**
   - Attached to: **NAT**



Adapter 2 – Internal Cluster Network

- **Enable Network Adapter**
- Attached to: **Internal Network**



Note: *The NAT adapter provides internet access to each virtual machine, while the internal network enables secure communication between the VMs without any exposure to external networks.*

4. Click **OK** to save changes.

Phase 3: Ubuntu Installation

Performed the following steps on each VM individually (`crtl-node, work-node-1, work-node-2`). However, some of us cloned the `crtl-node` to create `work-node-1` and `work-node-2.`

1. Boot the VM

- Start the VM. Automatically booted from the selected Ubuntu Server ISO.
- Selected **"Try or Install Ubuntu Server"**.

2. Language and Keyboard

- **Language:** English.
- **Keyboard Layout:** keep the default.

3. Installation Type

- Choose **"Ubuntu Server"** (recommended).
  - *Avoid "Minimized Ubuntu Server" unless you have specific needs.*

4. Network Connections

Two interfaces:

- **enp0s8** (connected to NAT):
  - Should automatically get a DHCP IP (e.g., `10.0.2.15/24`) – This is correct.
- **enp0s9** (connected to Internal Network):
  - Likely shows **no IP** – This is expected for now.
  - **Note: Network settings were not mortified at this stage.**
- Just select **Done** to proceed.

5. Configure Proxy

- Left it blank.
- Click **Done**.

6. Configure Ubuntu Archive Mirror

- Used the default mirror (recommended).
- Click **Done**.

7. Storage Configuration

- Choose **"Use an entire disk"**.

- Select the virtual disk you created earlier.
- Uncheck **"Set up this disk as an LVM group"** for simplicity
- Click **Done**, then **Confirm** the destructive action in the next prompt.

8. Profile Setup

Fill in the following details:

- **Your name:** nadiaha
- **Server's name:** Enter the **hostname** specific to the VM:
  - `crtl-node, work-node-1, or work-node-2`
- **Username:**
- **Password:**
- **Confirm Password:** Re-enter it.
- Click **Done** to continue.

9. SSH Setup

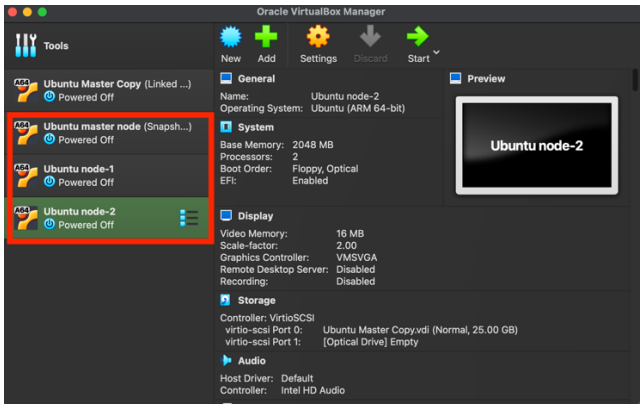
- Check the box to **Install OpenSSH server** (required for remote management).
- Do not import SSH identities at this stage.
- Click **Done**.

10. Server Snaps (Optional)

- Skip
- Click **Done**.

11. Installation and Reboot

- Wait for the installation to complete.
- Once finished, select **Reboot Now**.
- Prompted to **"remove the installation medium"**, press **Enter**.



***Note**: Three VM are created*

Phase 4: Post-Installation Configuration

This phase configuring Static IPs and Hostname Resolution for Each VM and repeated these steps on all three VMs: `crtl-node, work-node-1, and work-node-2`

1. Log In

- Used the credentials created during installation:
  - **Username:** `nadiaha` (this is my admin username)
  - **Password:** configured password

2. Identify Network Interface Names

Run the following command to list network interfaces:

```
ip a
```

Note: *this was before assigning the static IP addresses*

- Two interfaces:
  - **`enp0s8`** → connected to NAT (should have an IP like `10.0.2.x`)
  - **`enp0s9`** → connected to Internal Network (may show no IP or APIPA address)

3. Configure Netplan for Static Internal IP

Step 3.1 – Locate the Netplan Config File

```
ls /etc/netplan/
```

- Filename: `50-cloud-init. yaml`

Step 3.2 – Edit the Netplan File

```
sudo nano /etc/netplan/50-cloud-init. Yaml
```

Note: *This is before adjusting interface* names and IP address

Step 3.3 – Replace the File Content with the Following Template:

Note: adjusted the interface names and IP address per VM

```
network:
  version: 2
  ethernets:
    enp0s3:
      dhcp4: true
    enp0s8:
      dhcp4: no
      addresses:
        - 192.168.1.100/24 # ctrl-node (adjust for each VM)
```

Changed `IP address` to the correct static IP for each VM:

- `ctrl-node: 192.168.1.100`
- `work-node-1: 192.168.1.150`
- `work-node-2: 192.168.1.200`

Yaml file **`ctrl-node`**`:`

Yaml file **work-node-1**

Yaml file **work-node-2**

4. Apply Netplan Changes

- Save the file in Nano:
  - Press **Ctrl + O**, then **Enter**
  - Press **Ctrl + X** to exit
- Apply the configuration:

```
sudo netplan apply
```

5. Verify IP Configuration

Run:

```
ip a
```

ctrl-node

```
nadiaha@ctrl-node:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:35:03:97 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 metric 100 brd 10.0.2.255 scope global dynamic enp0s8
       valid_lft 86330sec preferred_lft 86330sec
    inet6 fd00::a00:27ff:fe35:397/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 86334sec preferred_lft 14334sec
    inet6 fe80::a00:27ff:fe35:397/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:7f:a4:51 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.100/24 brd 192.168.1.255 scope global enp0s9
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe7f:a451/64 scope link
       valid_lft forever preferred_lft forever
nadiaha@ctrl-node:~$ _
```

work-node-1

```
nadiaha@work-node-1:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:ac:08:86 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 metric 100 brd 10.0.2.255 scope global dynamic enp0s8
       valid_lft 86374sec preferred_lft 86374sec
    inet6 fd00::a00:27ff:feac:886/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 86377sec preferred_lft 14377sec
    inet6 fe80::a00:27ff:feac:886/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:c6:4f:17 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.150/24 brd 192.168.1.255 scope global enp0s9
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fec6:4f17/64 scope link
       valid_lft forever preferred_lft forever
```

work-node-2

```
nadiaha@work-node-2:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:af:44:33 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 metric 100 brd 10.0.2.255 scope global dynamic enp0s8
       valid_lft 86385sec preferred_lft 86385sec
    inet6 fd00::a00:27ff:feaf:4433/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 86387sec preferred_lft 14387sec
    inet6 fe80::a00:27ff:feaf:4433/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:69:98:7e brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.200/24 brd 192.168.1.255 scope global enp0s9
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe69:987e/64 scope link
       valid_lft forever preferred_lft forever
nadiaha@work-node-2:~$
```

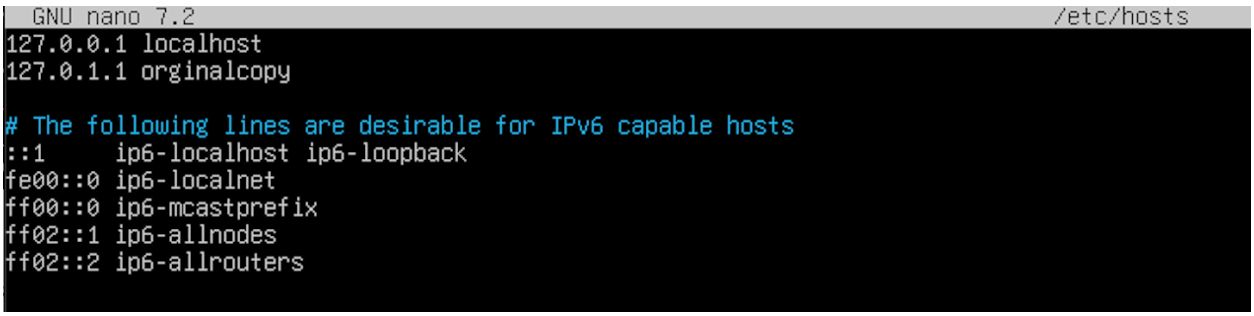
Check/ verify:

- **`enp0s8`** still has a NAT-assigned IP (e.g., `10.0.2.x`)
- **`enp0s9`** now has the configured **Static IP** (`192.168.1.100, .150`, or `.200`)

6. Configure Hostname Resolution

Edit the `/etc/hosts` file:

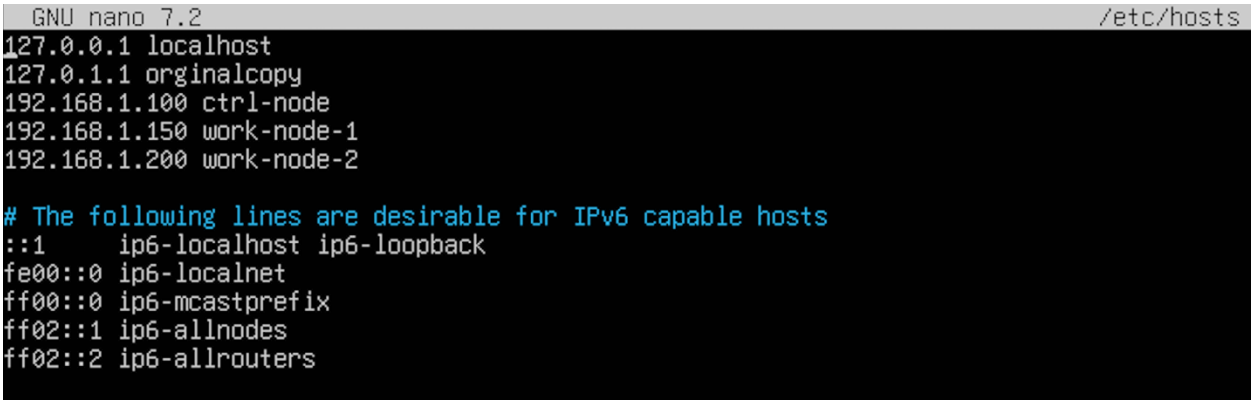
```
sudo nano /etc/hosts
```



Note: *This is before assigning Hostname*

Added the following lines below the `127.0.0.1` entry:

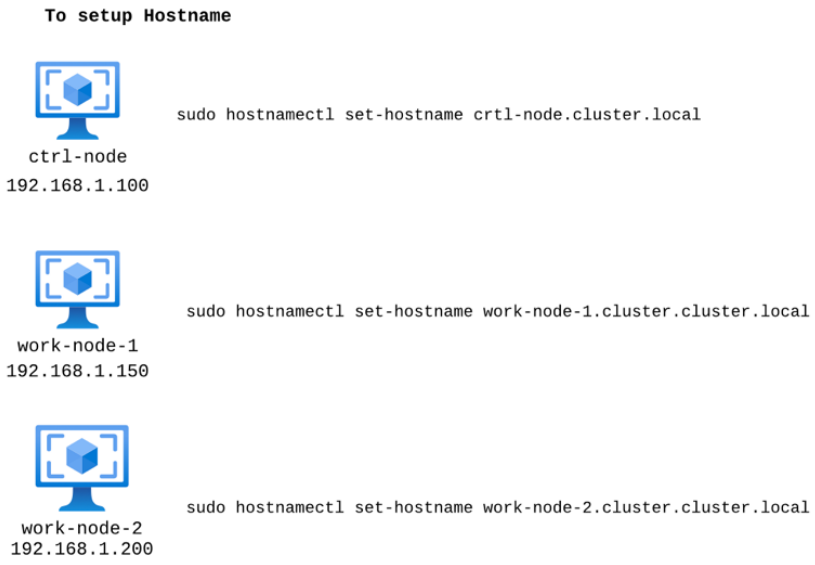
```
192.168.1.100 ctrl-node
192.168.1.150 work-node-1
192.168.1.200 work-node-2
```



- Save and exit Nano (**Ctrl + O**, **Enter**, then **Ctrl + X**)

Note*: Repeated this step to update on all three VMs to ensure consistent name resolution across the cluster.*

7. Setup Host name for each VM



from **crlt-node:**

```
sudo hostnamectl set-hostname crtl-node.cluster.local
```

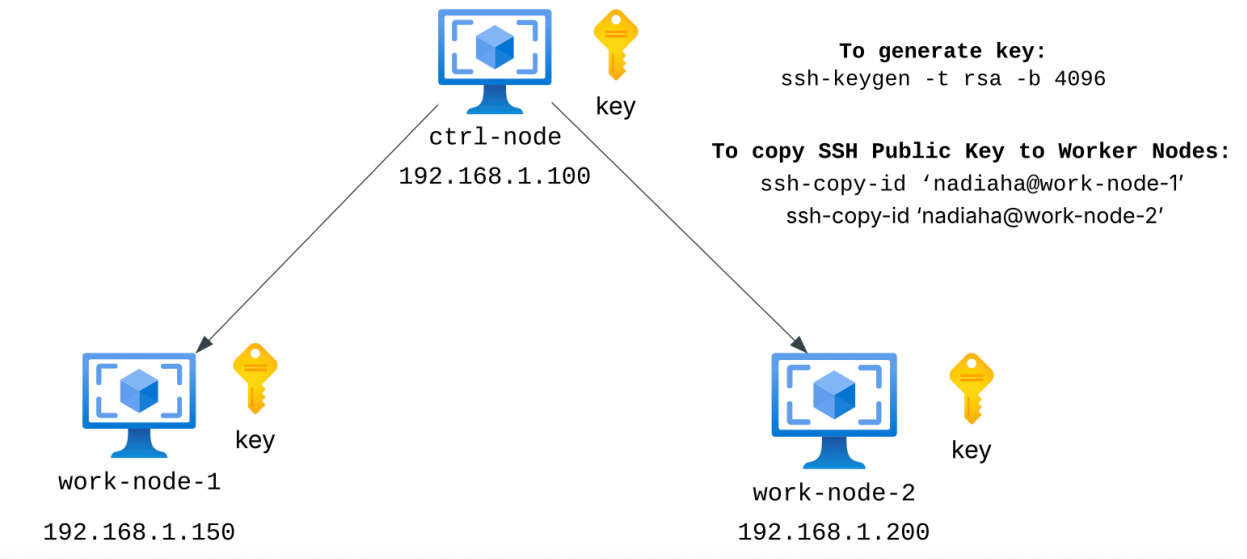
from **work-node-1:**

```
sudo hostnamectl set-hostname work-node-1.cluster.cluster.local
```

from **work-node-2:**

```
sudo hostnamectl set-hostname work-node-2.cluster.cluster.loca
```

Phase 5: SSH Configuration - Passwordless Access from Ctrl-Node



Performed these steps **only** on `ctrl-node`. The goal is to allow the `ctrl-node` to SSH into both `work-node-1` and `work-node-2` without a password.

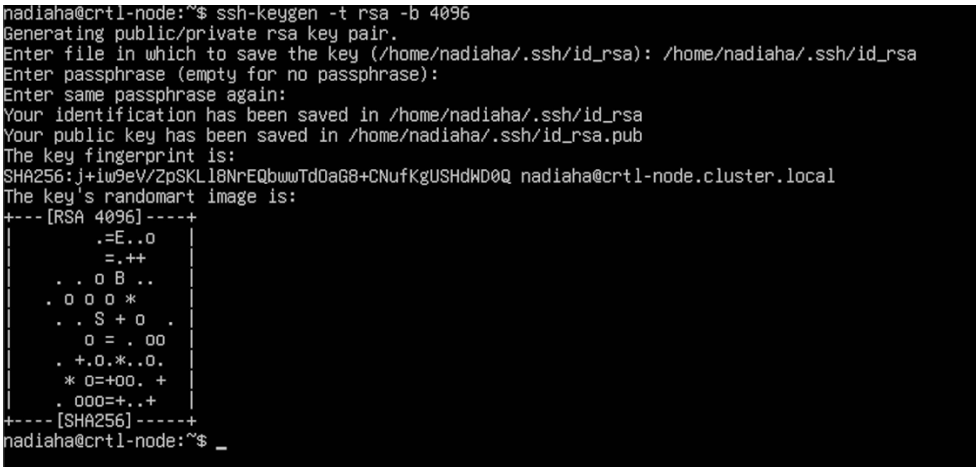
1. Log in to Master Node

- Login to `ctrl-node`

2. Generate SSH Key Pair

Use the following command to generate a new RSA key pair:

```
ssh-keygen -t rsa -b 4096
```



At the prompts:

- **File location:** Press **Enter** to accept the default (`/home/nadiaha/.ssh/id_rsa`)
- **Passphrase:** Press **Enter** twice to leave it blank (no passphrase)

This will generate:

- **Private Key**: `~/.ssh/id_rsa`
- **Public Key**: `~/.ssh/id_rsa.pub`

3. Copy SSH Public Key to Work-Nodes

Copy Key to `work-node-1` and `work-node-2` Command:

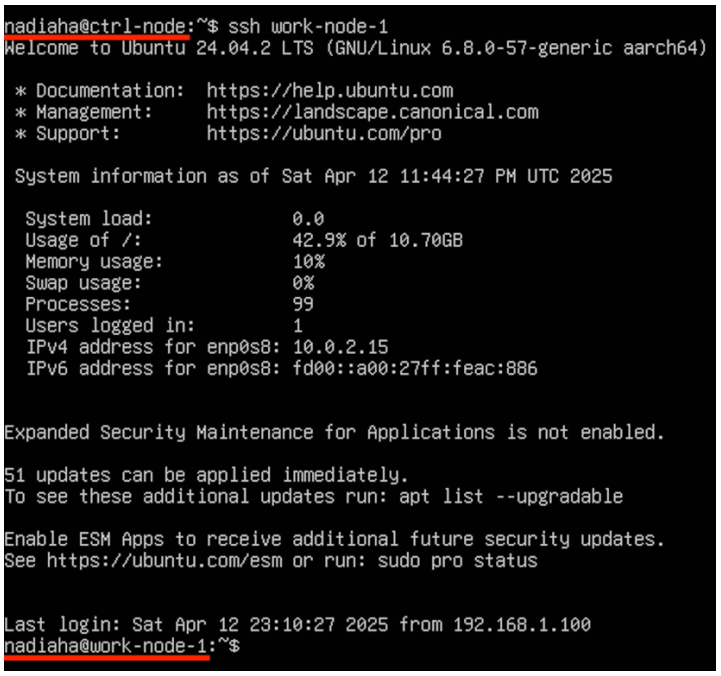
```
ssh-copy-id 'nadiaha@work-node-1'
ssh-copy-id 'nadiaha@work-node-2'
```

- Type `yes` to trust the host.
- Enter the `admin` password for `work-node-1` and `work-node-1` when prompted.

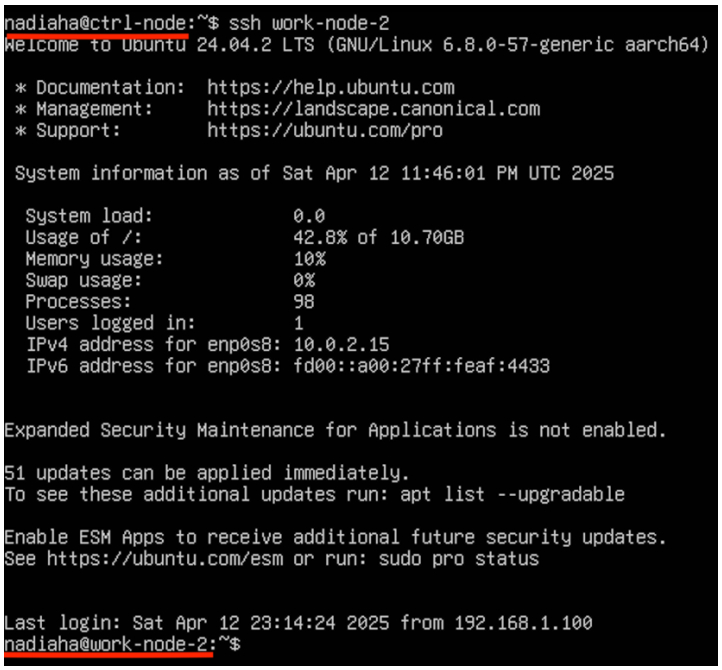
4. Verification:

After copying the keys, tested passwordless SSH from **`ctrl-node`**:

ssh work-node-1



ssh work-node-2



Note: *Passwordless SSH allows for secure and seamless remote access to worker nodes, which is crucial for efficient automation and effective cluster management.*

Phase 6: Verification and Testing

Performed the following steps to ensure your VM cluster is correctly configured and fully functional.

1. IP Configuration Check

On **each node**, run the following to verify interface names and assigned IPs:

```
ip a
```

- `enp0s8` should show a **DHCP address** (e.g., `10.0.2.x`)
- `enp0s9` should show the **assigned static IP** (`192.168.1.100`, `.150`, `.200`)

Note: This step was also done in phase 4

2. Internet Connectivity Test

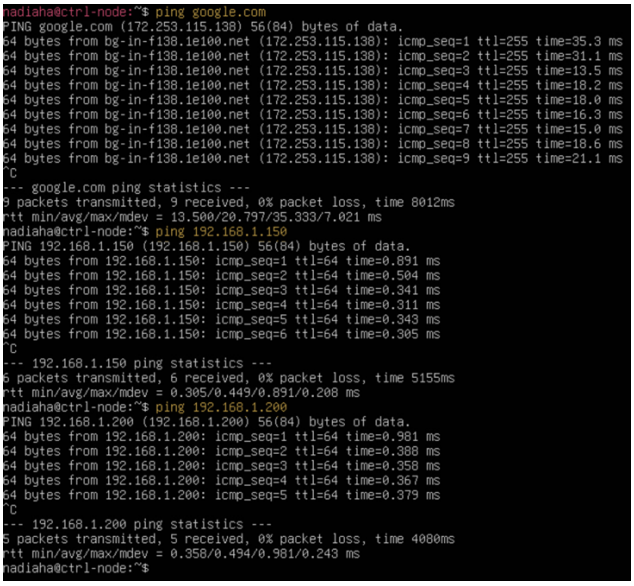
On **each node**, check internet access via NAT:

```
ping google.com
```

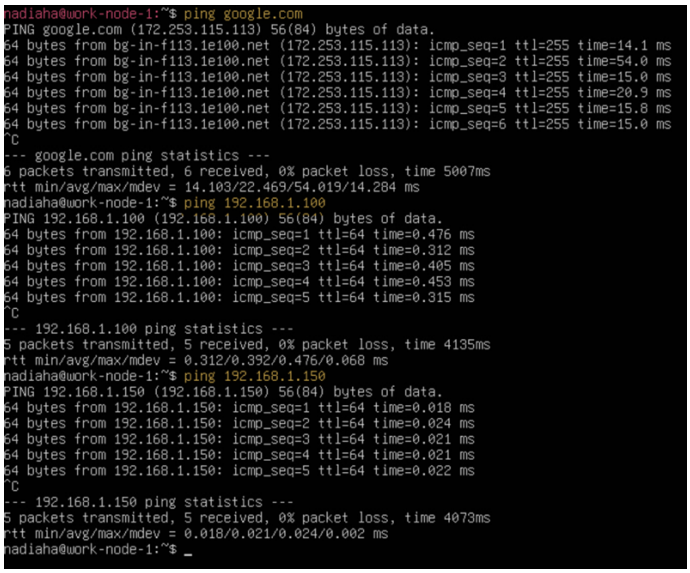
Response showed successful pings.

3. Internal Network Ping (By IP)

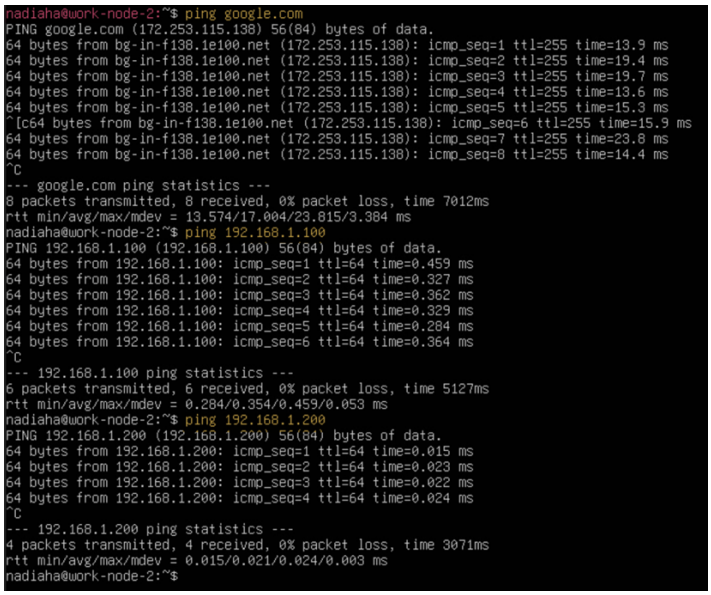
From **`ctrl-node: Ping google.com`** and **`IP address`** for work-node-1(**`192.168.1.150`**) & work-node-2 (**`192.168.1.200`**)



From **work-node-1: Ping google.com** and **IP address** for ctrl-node (**192.168.1.100**) & for work-node-2(**192.168.1.200**)



From **work-node-2: Ping google.com** and **IP address** for ctrl-node (**192.168.1.100**) & for work-node-1(**192.168.1.150**)

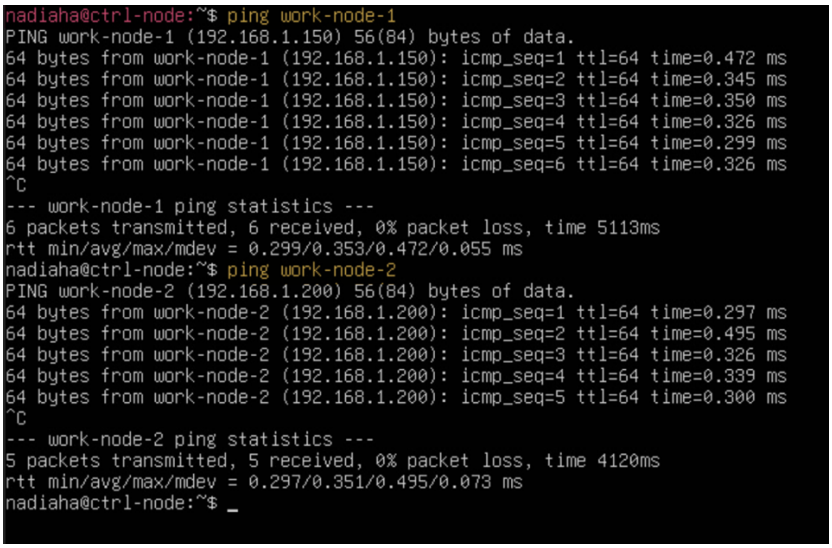


4. Internal Network Ping (By Hostname)

Check hostname resolution across the cluster, ping by hostname

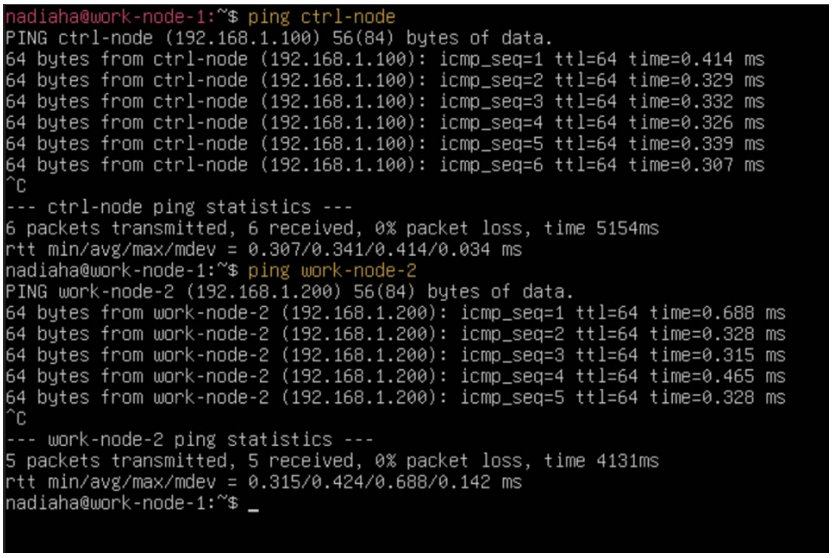
From **ctrl-node:**

```
ping work-node-1
ping work-node-2
```



From **work-node-1:**

```
ping ctrl-node
ping work-node-2
```



From **work-node-2:**

```
ping ctrl-node
ping work-node-1
```

```
nadiaha@work-node-2:~$ ping ctrl-node
PING ctrl-node (192.168.1.100) 56(84) bytes of data.
64 bytes from ctrl-node (192.168.1.100): icmp_seq=1 ttl=64 time=0.329 ms
64 bytes from ctrl-node (192.168.1.100): icmp_seq=2 ttl=64 time=0.291 ms
64 bytes from ctrl-node (192.168.1.100): icmp_seq=3 ttl=64 time=0.406 ms
64 bytes from ctrl-node (192.168.1.100): icmp_seq=4 ttl=64 time=0.344 ms
64 bytes from ctrl-node (192.168.1.100): icmp_seq=5 ttl=64 time=0.327 ms
64 bytes from ctrl-node (192.168.1.100): icmp_seq=6 ttl=64 time=0.289 ms
64 bytes from ctrl-node (192.168.1.100): icmp_seq=7 ttl=64 time=0.320 ms
^C
--- ctrl-node ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6180ms
rtt min/avg/max/mdev = 0.289/0.329/0.406/0.036 ms
nadiaha@work-node-2:~$ ping work-node-1
PING work-node-1 (192.168.1.150) 56(84) bytes of data.
64 bytes from work-node-1 (192.168.1.150): icmp_seq=1 ttl=64 time=0.354 ms
64 bytes from work-node-1 (192.168.1.150): icmp_seq=2 ttl=64 time=0.333 ms
64 bytes from work-node-1 (192.168.1.150): icmp_seq=3 ttl=64 time=0.370 ms
64 bytes from work-node-1 (192.168.1.150): icmp_seq=4 ttl=64 time=0.269 ms
64 bytes from work-node-1 (192.168.1.150): icmp_seq=5 ttl=64 time=0.249 ms
^C
--- work-node-1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4130ms
rtt min/avg/max/mdev = 0.249/0.315/0.370/0.047 ms
nadiaha@work-node-2:~$ _
```

Note: *Since the* **`/etc/hosts`** *file was configured correctly, all related operations were successful. Hostname resolution worked as expected.*

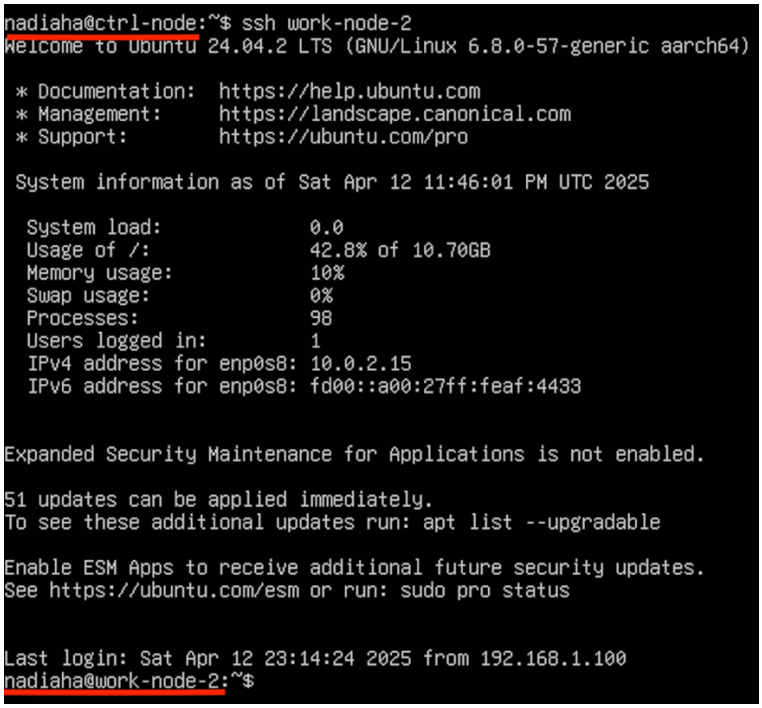
5. Passwordless SSH Verification

From **`ctrl-node`**, verify passwordless access to both worker nodes:

`ssh work-node-1`

```
nadiaha@ctrl-node:~$ ssh work-node-1
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-57-generic aarch64)

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

 System information as of Sat Apr 12 11:44:27 PM UTC 2025

  System load:             0.0
  Usage of /:              42.9% of 10.70GB
  Memory usage:            10%
  Swap usage:              0%
  Processes:               99
  Users logged in:         1
  IPv4 address for enp0s8: 10.0.2.15
  IPv6 address for enp0s8: fd00::a00:27ff:feac:886


Expanded Security Maintenance for Applications is not enabled.

51 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

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


Last login: Sat Apr 12 23:10:27 2025 from 192.168.1.100
nadiaha@work-node-1:~$
```

`ssh work-node-2`



Note: *The command successfully returned the correct hostname (`work-node-1` or `work-node-2`) without prompting for a password, confirming that passwordless SSH is working as intended.*

Challenges and Solutions:

Challenge:

- Initially, I ran into trouble when attempting to configure a static IP address on my virtual machine. I made the mistake of copying a template from an example without verifying if it matched my system's actual network interface. Specifically, I used `enp0s3` as the interface name, which did not exist on my VM. As a result, my configuration didn't apply correctly, and the network did not function as expected.

Solution:

- This experience taught me the importance of carefully reading and fully understanding the instructions before applying any configuration changes. It also reminded me to never assume that template values will work universally. To resolve the issue, I researched how to configure Netplan by watching YouTube tutorials and consulting external documentation. Doing this, I gained a clearer understanding of how to correctly identify the network interface and apply the correct settings for a static IP address.

Challenge:

- Another challenge I encountered was that my virtual machine wasn't responding when I tried to ping it using its hostname.

Solution:

- After investigating, I realized that I hadn't correctly associated the hostname with its corresponding IP address. Either the hostname was incorrect, or the IP address wasn't properly mapped. To fix this, I made sure the `/etc/hosts` file was updated with the correct IP-to-hostname mapping. This experience emphasized the importance of double-checking my configurations and ensuring that all dependencies, like name resolution, are properly set up.