Virtualization and VM Provisioning Assignment

Student Names: Nkwa Jude Tambe

Yemele Christian

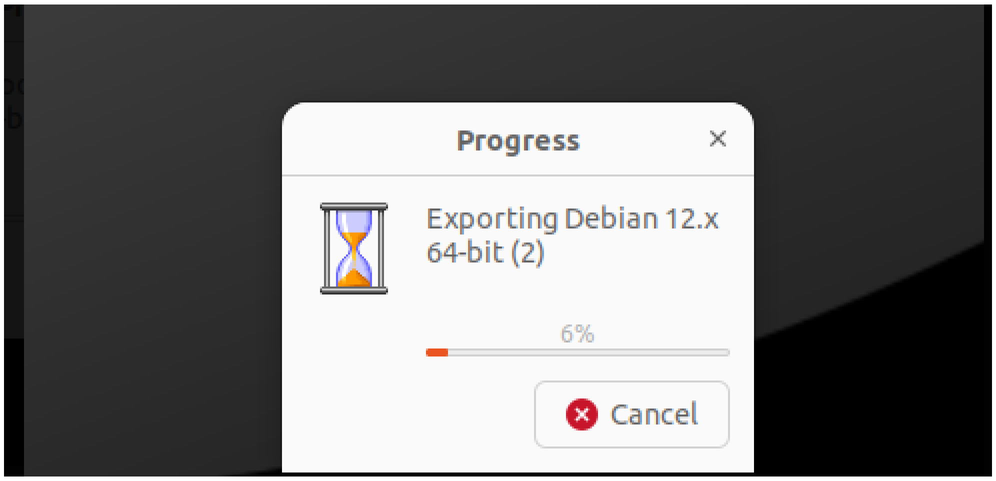
1. Exporting and Importing a VM

Objective:

Create a VM, export its configuration and disk, then import it into a new VM and verify it boots successfully.

Steps Performed:

1. **Created a Virtual Machine**
    - Resources: CPU [2], RAM [2 GB], Disk [20 GB]
    - Installed updates and basic packages.
2. **Exported the VM**
    - Format: OVF (Open Virtualization Format)



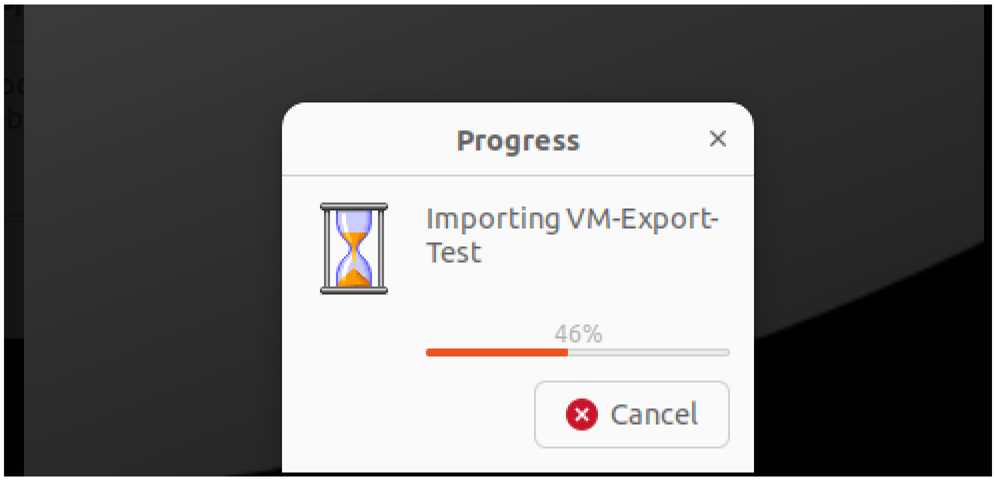
Progress

Exporting Debian 12.x 64-bit (2)

6%

- Included all disk files and configuration.

3. **Imported into a New VM**



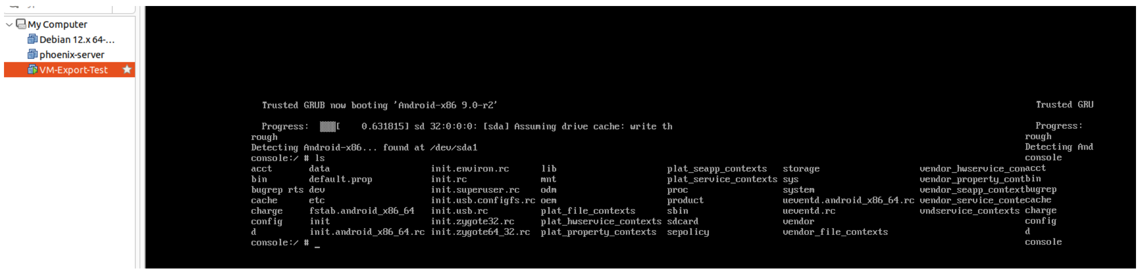
Importing VM-Export-Test

46%

- Verified imported VM resources matched original VM.

4. **Verification**

- Booted the new VM successfully.
- Screenshot of VM running:



```
  Trusted GRUB now booting 'Android-x86 9.0-r2'                                                              Trusted GRU

  Progress:  ▒▒[    0.631815] sd 32:0:0:0: [sda] Assuming drive cache: write th                            Progress:
rough                                                                                                       rough
Detecting Android-x86... found at /dev/sda1                                                                 Detecting And
console:/ # ls                                                                                              console
acct       data                  init.environ.rc      lib                     plat_seapp_contexts   storage                vendor_hwservice_conacct
bin        default.prop          init.rc              mnt                     plat_service_contexts sys                    vendor_property_contbin
bugrep rts dev                   init.superuser.rc    odm                     proc                  system                 vendor_seapp_contextbugrep
cache      etc                   init.usb.configfs.rc oem                     product               ueventd.android_x86_64.rc vendor_service_contecache
charge     fstab.android_x86_64  init.usb.rc          plat_file_contexts      sbin                  ueventd.rc             vndservice_contexts charge
config     init                  init.zygote32.rc     plat_hwservice_contexts sdcard                vendor                                     config
d          init.android_x86_64.rc init.zygote64_32.rc plat_property_contexts  sepolicy              vendor_file_contexts                       d
console:/ # _                                                                                               console
```

2. VM Provisioning using Vagrant

Objective: Automatically provision a VM using Vagrant.

Vagrantfile Used:

Vagrantfile

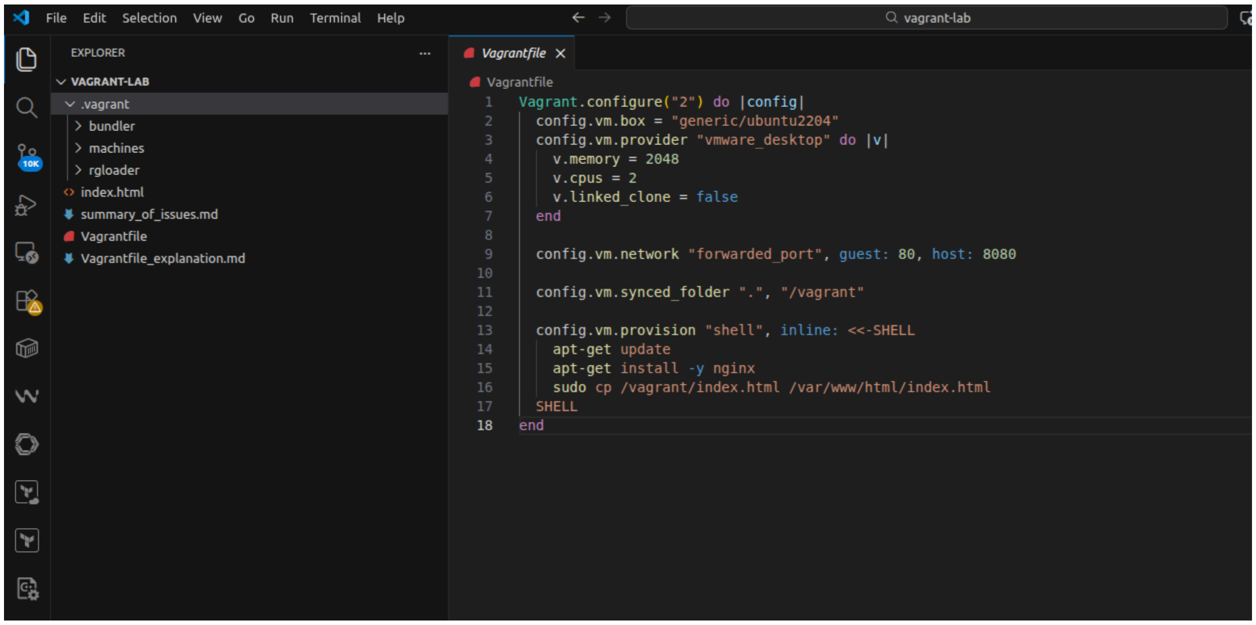
```ruby
Vagrant.configure("2") do |config|
  # Base box
  config.vm.box = "generic/ubuntu2204"

  # VMware provider settings
  config.vm.provider "vmware_desktop" do |v|
    v.memory = 2048   # 2 GB RAM
    v.cpus = 2        # 2 CPU cores
    v.linked_clone = false
  end

  # Forward HTTP port from guest to host
  config.vm.network "forwarded_port", guest: 80, host: 8080

  # Sync current folder to VM
  config.vm.synced_folder ".", "/vagrant"

  # Provision VM with Nginx and copy index.html
  config.vm.provision "shell", inline: <<-SHELL
    apt-get update
    apt-get install -y nginx
    sudo cp /vagrant/index.html /var/www/html/index.html
    sudo systemctl enable nginx
    sudo systemctl start nginx
  SHELL
end
```



```
Vagrant.configure("2") do |config|
  config.vm.box = "generic/ubuntu2204"
  config.vm.provider "vmware_desktop" do |v|
    v.memory = 2048
    v.cpus = 2
    v.linked_clone = false
  end

  config.vm.network "forwarded_port", guest: 80, host: 8080

  config.vm.synced_folder ".", "/vagrant"

  config.vm.provision "shell", inline: <<-SHELL
    apt-get update
    apt-get install -y nginx
    sudo cp /vagrant/index.html /var/www/html/index.html
  SHELL
end
```

Steps Performed:

1. Initialized Vagrant environment:

```
vagrant init
```

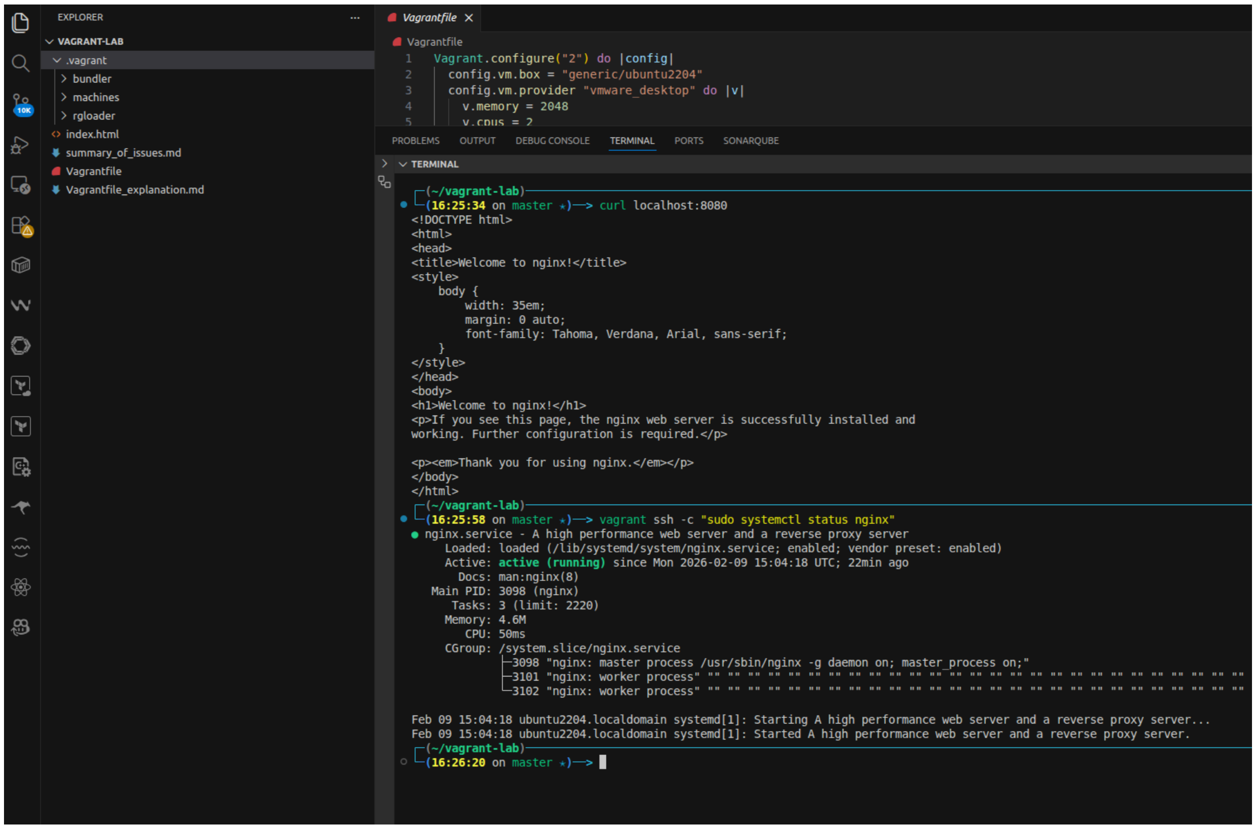
2. Started and provisioned VM:

```
vagrant up
```

3. Verified Apache service is running on VM.

Result: VM was automatically configured and Apache web server is running.

Screenshot of service status:



```
Vagrant.configure("2") do |config|
  config.vm.box = "generic/ubuntu2204"
  config.vm.provider "vmware_desktop" do |v|
    v.memory = 2048
    v.cpus = 2
```

```
┌(~/vagrant-lab)
└(16:25:34 on master ★)──> curl localhost:8080
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
┌(~/vagrant-lab)
└(16:25:58 on master ★)──> vagrant ssh -c "sudo systemctl status nginx"
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
     Active: active (running) since Mon 2026-02-09 15:04:18 UTC; 22min ago
       Docs: man:nginx(8)
   Main PID: 3098 (nginx)
      Tasks: 3 (limit: 2220)
     Memory: 4.6M
        CPU: 50ms
     CGroup: /system.slice/nginx.service
             ├─3098 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             ├─3101 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""
             └─3102 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""

Feb 09 15:04:18 ubuntu2204.localdomain systemd[1]: Starting A high performance web server and a reverse proxy server...
Feb 09 15:04:18 ubuntu2204.localdomain systemd[1]: Started A high performance web server and a reverse proxy server.
┌(~/vagrant-lab)
└(16:26:20 on master ★)──> 
```

---

3. VM Configuration using Cloud-Init

Objective:

Configure a VM on first boot using cloud-init, automating user creation, package installation, and service setup.

Cloud-Init Configuration File (`cloud-init.yaml`):

```
#cloud-config

# Create a user with sudo privileges
users:
  - name: student
    sudo: ['ALL=(ALL) NOPASSWD:ALL']
    shell: /bin/bash
```

```
# Install packages automatically on first boot
packages:
  - nginx                 # Web server
  - git                   # Version control
  - vim                   # Text editor
  - htop                  # System monitor
  - curl                  # Command-line HTTP client
  - wget                  # Download tool
  - build-essential       # Compiler and development tools
  - python3               # Python runtime
  - python3-pip           # Python package manager
  - unzip                 # Extract zip files
  - net-tools             # Network tools (ifconfig, netstat)
  - tree                  # Directory structure visualization

# Run commands at first boot
runcmd:
  # Set up a welcome page for Nginx
  - echo "<h1>Welcome to Multipass Nginx Lab!</h1>" > /home/s
tudent/index.html
  - sudo mv /home/student/index.html /var/www/html/index.html

  # Enable and start Nginx
  - sudo systemctl enable nginx
  - sudo systemctl start nginx

  # Create directories for practice
  - mkdir -p /home/student/projects
  - mkdir -p /home/student/scripts



  # Update pip and install a Python package
  - python3 -m pip install --upgrade pip
  - python3 -m pip install requests

  # Log system info in a welcome file
  - echo "Multipass VM initialized successfully on $(date)" >
> /home/student/welcome.log
  - uname -a >> /home/student/welcome.log
  - lsb_release -a >> /home/student/welcome.log
```

Steps Performed:

1. Created the file `cloud-init.yaml` in the local directory.
2. Launched the Multipass VM with cloud-init applied:

   ```
   multipass launch --name cloudinit --memory 1G --disk 5G --
   cpus 1 --cloud-init cloud-init.yaml
   ```

3. Connected to the VM to verify:

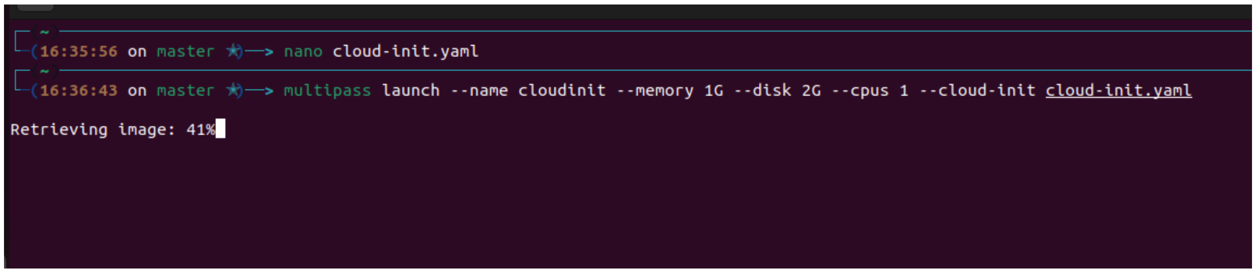
   ```
   multipass shell cloudinit
   ```

4. Verified:
   - User `student` exists with sudo privileges.
   - Nginx service is running (`systemctl status nginx`).
   - Packages like `git`, `vim`, `htop`, `python3`, `pip` are installed.
   - Welcome webpage is accessible via the VM IP.
   - `projects` and `scripts` directories created.
   - `welcome.log` file contains system info.

Result:

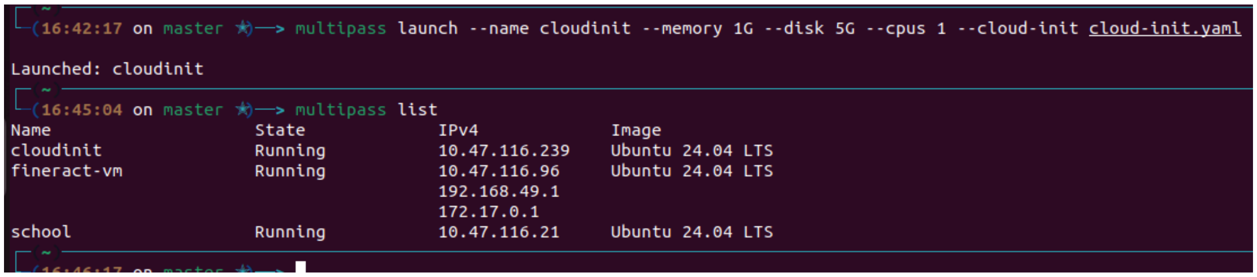
The VM is fully configured automatically on first boot, ready for development, testing, and learning exercises.

Screenshot



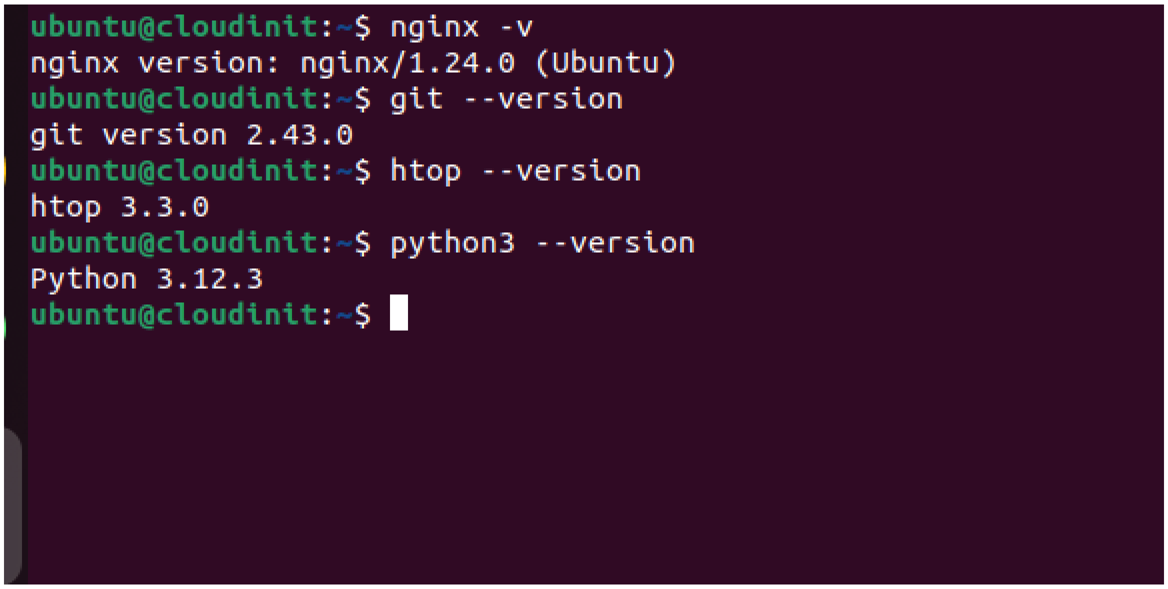
```
(16:35:56 on master ✯)──> nano cloud-init.yaml
(16:36:43 on master ✯)──> multipass launch --name cloudinit --memory 1G --disk 2G --cpus 1 --cloud-init cloud-init.yaml

Retrieving image: 41%
```

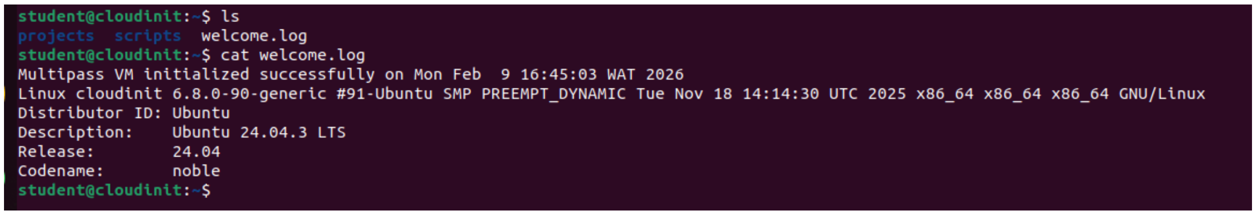


```
(16:42:17 on master ✯)──> multipass launch --name cloudinit --memory 1G --disk 5G --cpus 1 --cloud-init cloud-init.yaml

Launched: cloudinit
(16:45:04 on master ✯)──> multipass list
Name                    State             IPv4             Image
cloudinit               Running           10.47.116.239    Ubuntu 24.04 LTS
fineract-vm             Running           10.47.116.96     Ubuntu 24.04 LTS
                                          192.168.49.1
                                          172.17.0.1
school                  Running           10.47.116.21     Ubuntu 24.04 LTS
```



```
ubuntu@cloudinit:~$ nginx -v
nginx version: nginx/1.24.0 (Ubuntu)
ubuntu@cloudinit:~$ git --version
git version 2.43.0
ubuntu@cloudinit:~$ htop --version
htop 3.3.0
ubuntu@cloudinit:~$ python3 --version
Python 3.12.3
ubuntu@cloudinit:~$
```



```
student@cloudinit:~$ ls
projects  scripts  welcome.log
student@cloudinit:~$ cat welcome.log
Multipass VM initialized successfully on Mon Feb  9 16:45:03 WAT 2026
Linux cloudinit 6.8.0-90-generic #91-Ubuntu SMP PREEMPT_DYNAMIC Tue Nov 18 14:14:30 UTC 2025 x86_64 x86_64 x86_64 GNU/Linux
Distributor ID: Ubuntu
Description:    Ubuntu 24.04.3 LTS
Release:        24.04
Codename:       noble
student@cloudinit:~$
```