Lab: Automating Ubuntu VM Setup with Vagrant on Linux

Step 1: Create and Enter Your Project Directory

mkdir -p /Testdir/vagrant/lab
cd /Testdir/vagrant/lab

Explanation: This creates your isolated Vagrant environment folder.

```
tiago-paquete@Ubuntu1:~$ mkdir -p /Testdir/vagrant/lab
========================================================================
mkdir: cannot create directory ‘/Testdir’: Permission denied
========================================================================
```

```
tiago-paquete@Ubuntu1:~$ sudo !!
========================================================================
sudo mkdir -p /Testdir/vagrant/lab
[sudo] password for tiago-paquete:
========================================================================
```

Step 2: Initialize the Vagrant Environment

vagrant init ubuntu/jammy64

Explanation: This generates a Vagrantfile configured to use the Ubuntu 22.04 LTS base box (jammy64).

```
tiago-paquete@Ubuntu1:/Testdir/vagrant/lab$ vagrant init ubuntu/jammy64
========================================================================
The user that is running Vagrant doesn't have the proper permissions
to write a Vagrantfile to the specified location. Please ensure that
you call `vagrant init` in a location where the proper permissions
are in place to create a Vagrantfile.
========================================================================

tiago-paquete@Ubuntu1:/Testdir/vagrant/lab$ sudo !!
========================================================================
sudo vagrant init ubuntu/jammy64
==> vagrant: A new version of Vagrant is available: 2.4.6 (installed
version: 2.4.5)!
==> vagrant: To upgrade visit: https://www.vagrantup.com/downloads.html

A `Vagrantfile` has been placed in this directory. You are now
ready to `vagrant up` your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
`vagrantup.com` for more information on using Vagrant.
========================================================================
```

Step 3: Edit the Vagrantfile

Replace the contents with the following:

```
============================================================================
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/jammy64"

  # Set hostname
  config.vm.hostname = "vagrant-lab"

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

  # Set up a private network with a static IP
  config.vm.network "private_network", ip: "192.168.56.50"

  # Allocate system resources
  config.vm.provider "virtualbox" do |vb|
    vb.memory = "1024"
    vb.cpus = 2
  end

  # Sync current folder to /vagrant_data inside the VM
  config.vm.synced_folder ".", "/vagrant_data"

  # Provision VM with a shell script
  config.vm.provision "shell", inline: <<-SHELL
    sudo apt update
    sudo apt install -y apache2
    echo "Hello from Vagrant" | sudo tee /var/www/html/index.html
    sudo systemctl enable apache2
    sudo systemctl start apache2
  SHELL
end

============================================================================
```

Base Configuration Block

```
Vagrant.configure("2") do |config|
```

- This line begins the configuration block for Vagrant using version **2** of the configuration syntax.
- The block variable config is used to set up all the VM properties and behaviors.

Base Box Declaration

```
config.vm.box = "ubuntu/jammy64"
```

- This specifies the **base box** that Vagrant will use to create the VM.
- "ubuntu/jammy64" refers to the official **Ubuntu 22.04 LTS** image provided by HashiCorp.
- Vagrant will download this box the first time it's used.

Hostname Setup

```
config.vm.hostname = "vagrant-lab"
```

- Sets the **internal hostname** of the VM to "vagrant-lab".
- Useful for network identification, provisioning scripts, and internal tooling.

Port Forwarding

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

- Maps port **80 on the VM (guest)** to port **8080 on the host**.
- This means accessing http://localhost:8080 on the host will reach a web server running on port 80 in the VM.
- Useful for exposing services (e.g., Apache, Nginx) to your host machine.

Private Network Configuration

```
config.vm.network "private_network", ip: "192.168.56.50"
```

- Assigns the VM a **static private IP** address.
- The VM is **reachable only from the host** and other VMs on the same host-only network.
- Useful for internal access, testing private APIs, or multiple VM setups.

Resource Allocation for VirtualBox

```
config.vm.provider "virtualbox" do |vb|
  vb.memory = "1024"
  vb.cpus = 2
end
```

- Configures **VirtualBox-specific** VM resources:
  - vb.memory = "1024" allocates **1 GB of RAM**.
  - vb.cpus = 2 allocates **2 virtual CPU cores**.
- These values ensure the VM runs smoothly for lightweight development and testing.

Synced Folder

```
config.vm.synced_folder ".", "/vagrant_data"
```

- Syncs the **current host directory** (denoted by ".") to the **VM's internal path** /vagrant_data.
- Any file placed in the host project directory will automatically be available inside the VM at that path.
- Useful for code sharing, data manipulation, and development tools that require real-time access.

Provisioning Script (Shell Script)

```
config.vm.provision "shell", inline: <<-SHELL
  sudo apt update
  sudo apt install -y apache2
  echo "Hello from Vagrant" | sudo tee /var/www/html/index.html
  sudo systemctl enable apache2
  sudo systemctl start apache2

SHELL
```

- This block uses a **shell provisioner** to automate the VM setup.
- Here's what each command does:
  - sudo apt update: Updates the package index.
  - sudo apt install -y apache2: Installs the Apache2 web server.
  - echo "Hello from Vagrant": Writes a simple HTML page to the Apache web root.
  - systemctl enable apache2: Ensures Apache starts automatically on boot.
  - systemctl start apache2: Starts the Apache service immediately.

End of Configuration Block

```
end
```

- This closes the Vagrant.configure block.

✅ Summary of Concepts Covered

Feature	Purpose
Base Box	Standard starting point (Ubuntu 22.04 LTS)
Hostname	Identifies VM internally
Port Forwarding	Access guest services from host (localhost:8080)
Private Networking	Allows static IP access from host (192.168.56.50)
VirtualBox Resources	Controls RAM and CPU usage
Synced Folder	Syncs files between host and VM
Shell Provisioning	Automates software installation and service configuration

```
tiago-paquete@Ubuntu1:/Testdir/vagrant/lab$ sudo vim Vagrantfile
========================================================================
[sudo] password for tiago-paquete:
========================================================================

tiago-paquete@Ubuntu1:/Testdir/vagrant/lab$ cat Vagrantfile
========================================================================
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/jammy64"

  # Set hostname
  config.vm.hostname = "vagrant-lab"

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

  # Set up a private network with a static IP
  config.vm.network "private_network", ip: "192.168.56.50"

  # Allocate system resources
  config.vm.provider "virtualbox" do |vb|
    vb.memory = "1024"
    vb.cpus = 2
  end

  # Sync current folder to /vagrant_data inside the VM
  config.vm.synced_folder ".", "/vagrant_data"

  # Provision VM with a shell script
  config.vm.provision "shell", inline: <<-SHELL
    sudo apt update
    sudo apt install -y apache2
    echo "Hello from Vagrant" | sudo tee /var/www/html/index.html
    sudo systemctl enable apache2
    sudo systemctl start apache2
  SHELL
end
========================================================================
```

Step 4: Start and Provision the Virtual Machine

vagrant up

Explanation: This command:
Downloads the Ubuntu box if it’s not already available
Creates and boots the VM
Applies all configurations and provisioning defined in the Vagrantfile

```
tiago-paquete@Ubuntu1:/Testdir/vagrant/lab$ sudo vagrant up
========================================================================
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Box 'ubuntu/jammy64' could not be found. Attempting to find
and install...
    default: Box Provider: virtualbox
    default: Box Version: >= 0
==> default: Loading metadata for box ‘ubuntu/jammy64’

…

========================================================================
```

Step 5: Verify VM Is Running

vagrant global-status

Look for your VM's status (running) and its ID.

vagrant ssh

Explanation: SSH into your VM to validate configurations manually.

```
tiago-paquete@Ubuntu1:/Testdir/vagrant/lab$ vagrant global-status
========================================================================
id       name    provider   state   directory
------------------------------------------------------------------------
e84f295  default virtualbox running /home/tiago-paquete/Testdir/vagrant/
myvirtualmachines

The above shows information about all known Vagrant environments
on this machine. This data is cached and may not be completely
up-to-date (use "vagrant global-status --prune" to prune invalid
entries). To interact with any of the machines, you can go to that
directory and run Vagrant, or you can use the ID directly with
Vagrant commands from any directory. For example:
"vagrant destroy 1a2b3c4d”
========================================================================

tiago-paquete@Ubuntu1:/Testdir/vagrant/lab$ vagrant ssh
========================================================================
The VirtualBox VM was created with a user that doesn't match the
current user running Vagrant. VirtualBox requires that the same user
be used to manage the VM that was created. Please re-run Vagrant with
that user. This is not a Vagrant issue.

The UID used to create the VM was: 0
Your UID is: 1000
========================================================================
```

Avoid This in Future
Always run vagrant up and vagrant ssh using the same user that created the VM. If you use sudo, you’re using root — which should generally be avoided for Vagrant. Let me know if you want help automating the cleanup with a script.

```
tiago-paquete@Ubuntu1:/Testdir/vagrant/lab$ sudo vagrant ssh
========================================================================
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-140-generic x86_64)

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

 System information as of Fri May 23 10:28:58 UTC 2025

  System load:  0.03              Processes:               114
  Usage of /:   4.4% of 38.70GB   Users logged in:         0
  Memory usage: 24%               IPv4 address for enp0s3: 10.0.2.15
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

1 update can be applied immediately.
1 of these updates is a standard security update.
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

New release '24.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
========================================================================
```

Step 6: Verify Apache Web Server

Inside the VM:
curl http://localhost

You should see:
Hello from Vagrant

On your host machine:
curl http://localhost:8080

Confirms that **port forwarding** is working.

Also, test the **private network** (from the host):
ping 192.168.56.50

```
vagrant@vagrant-lab:~$ curl http://localhost
========================================================================
Hello from Vagrant
========================================================================

tiago-paquete@Ubuntu1:~$ curl http://localhost:8080
========================================================================
Hello from Vagrant
========================================================================

tiago-paquete@Ubuntu1:~$ ping 192.168.56.50
========================================================================
PING 192.168.56.50 (192.168.56.50) 56(84) bytes of data.
64 bytes from 192.168.56.50: icmp_seq=1 ttl=64 time=1.21 ms
64 bytes from 192.168.56.50: icmp_seq=2 ttl=64 time=0.977 ms
64 bytes from 192.168.56.50: icmp_seq=3 ttl=64 time=1.08 ms
64 bytes from 192.168.56.50: icmp_seq=4 ttl=64 time=1.10 ms
64 bytes from 192.168.56.50: icmp_seq=5 ttl=64 time=0.938 ms
64 bytes from 192.168.56.50: icmp_seq=6 ttl=64 time=1.15 ms
64 bytes from 192.168.56.50: icmp_seq=7 ttl=64 time=1.42 ms
^C
--- 192.168.56.50 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6009ms
rtt min/avg/max/mdev = 0.938/1.124/1.417/0.147 ms
========================================================================
```