EXPERIMENT 1

Lab Exercise: Introduction to Vagrant and Vagrantfile

This exercise will guide them through setting up a virtual environment using Vagrant, configuring the environment via a Vagrantfile, and managing the virtual machines (VMs) with basic Vagrant commands.

Objective:

- Learn how to set up and configure virtual environments using Vagrant.
- Understand the structure and components of a Vagrantfile.
- Gain hands-on experience in managing virtual machines using Vagrant commands.

Prerequisites:

- Basic knowledge of virtualization concepts.
- Familiarity with command-line interfaces.
- Installation of Vagrant and VirtualBox (or any other supported provider) on your local machine.

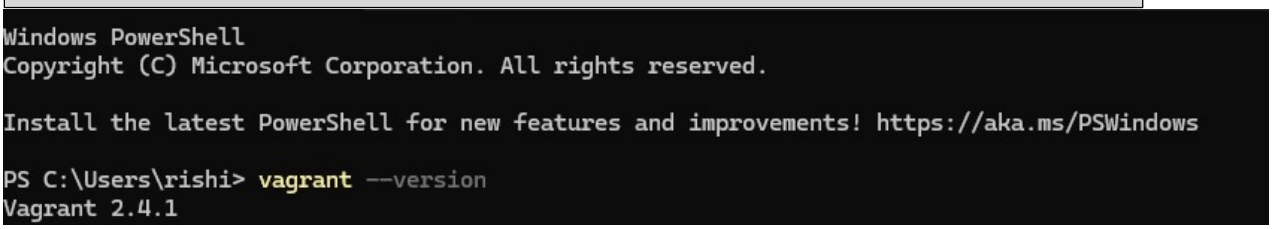
Step-by-Step Exercise:

1. Setting Up the Environment:

Install Vagrant:

- Download and install Vagrant from the official website.
- Ensure you have VirtualBox installed as it is a commonly used provider with Vagrant.
- Verify Installation:
- Open a terminal or command prompt.
- Run the following commands to verify the installation:

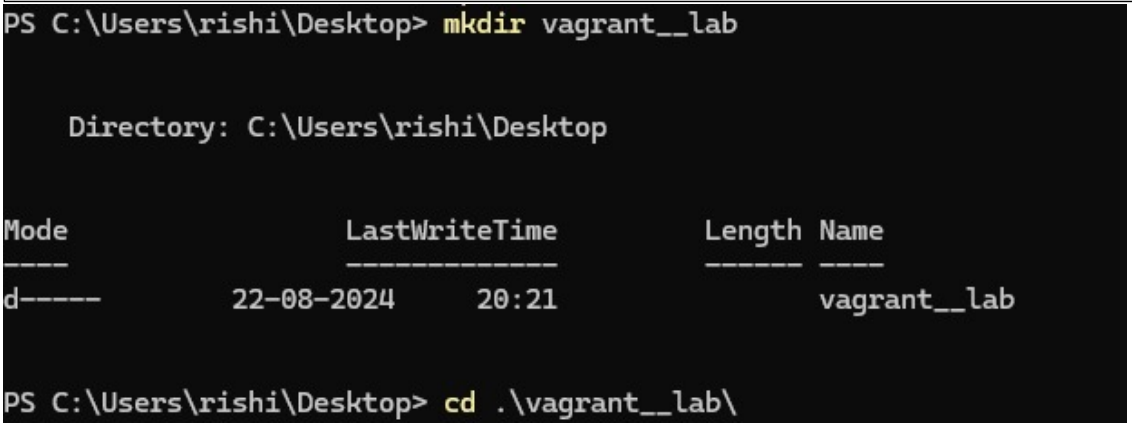
```
vagrant --version
```



Creating a New Vagrant Project:

- Create a Project Directory:
- In your terminal, create a new directory for your Vagrant project and navigate into it:

```
mkdir vagrant_lab

cd vagrant_lab
```

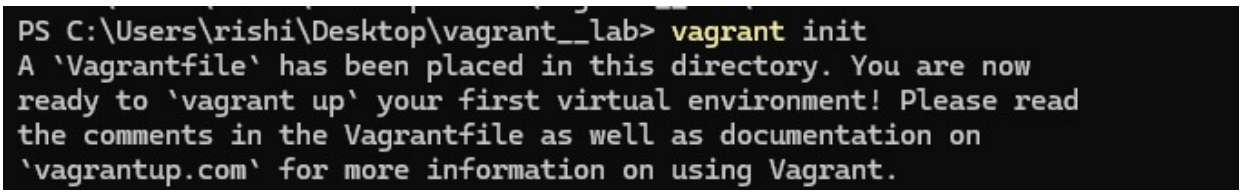


Initialize Vagrant:

- Run the following command to initialize a new Vagrantfile in your project directory:

```
vagrant init
```

This command will generate a Vagrantfile in the current directory.



3. Understanding the Vagrantfile:

- Open the Vagrantfile:
- Open the Vagrantfile in a text editor of your choice.
- The Vagrantfile is a Ruby-based configuration file used to define the virtual environment.
- Basic Vagrantfile Configuration:
- Modify the Vagrantfile to configure a basic virtual machine. For example:

\

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

  config.vm.box = "ubuntu/bionic64"  # Specifies the base box to use (Ubuntu 18.04)

  config.vm.network "private_network", type: "dhcp"  # Configures a private network

  config.vm.provider "virtualbox" do |vb|

    vb.memory = "1024"  # Allocates 1GB of RAM to the VM

  end

end
```

```
Vagrant.configure("2") do |config|
  # The most common configuration options are documented and commented below.
  # For a complete reference, please see the online documentation at
  # https://docs.vagrantup.com.

  # Every Vagrant development environment requires a box. You can search for
  # boxes at https://vagrantcloud.com/search.
  config.vm.box = "ubuntu/trusty64"

  # Disable automatic box update checking. If you disable this, then
  # boxes will only be checked for updates when the user runs
  # `vagrant box outdated`. This is not recommended.
  # config.vm.box_check_update = false

  # Create a forwarded port mapping which allows access to a specific port
  # within the machine from a port on the host machine. In the example below,
  # accessing "localhost:8080" will access port 80 on the guest machine.
  # NOTE: This will enable public access to the opened port
  # config.vm.network "forwarded_port", guest: 80, host: 8080
```

4. Launching and Managing the VM:

Start the VM:

In the terminal, start the VM using the following command:

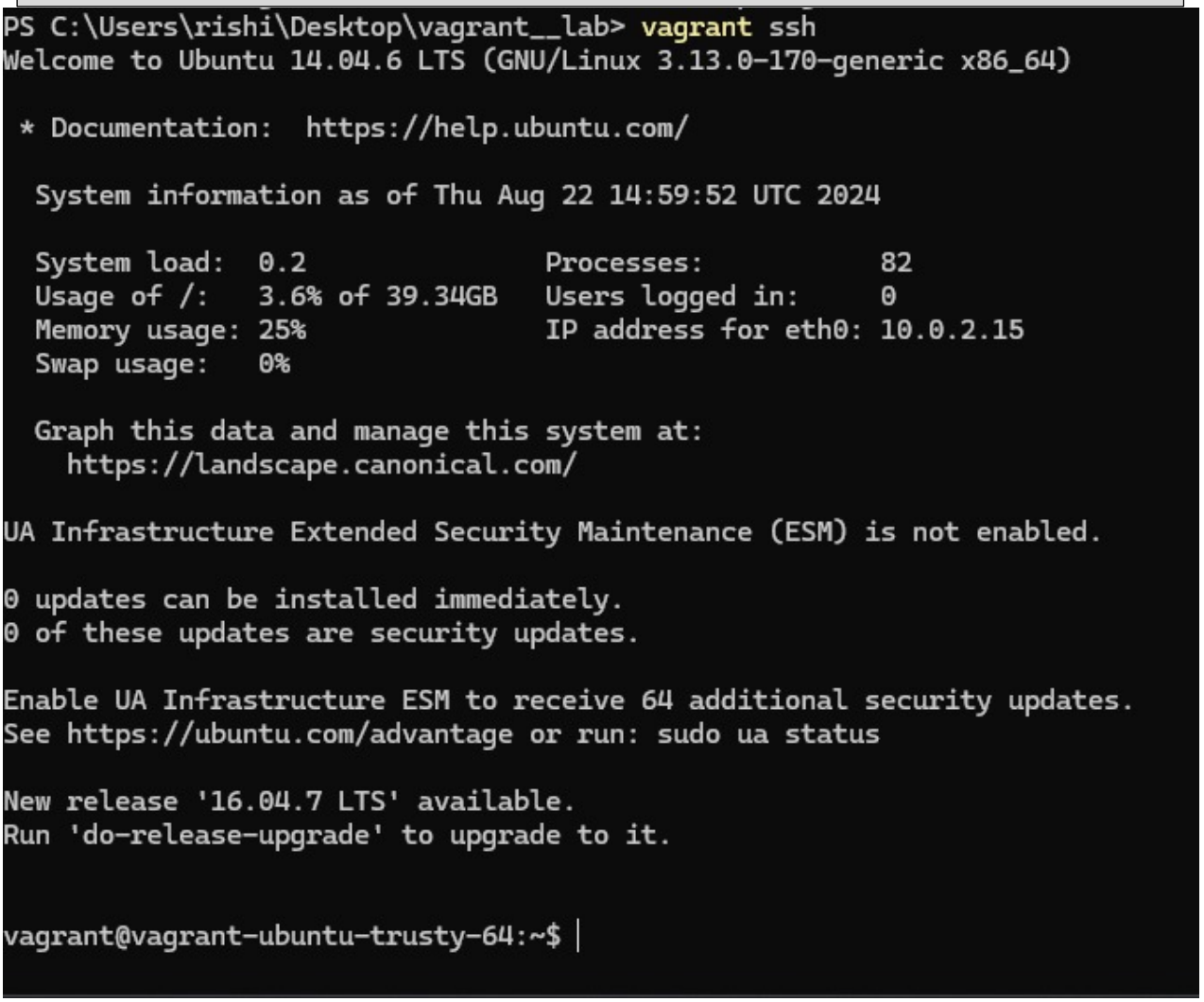
```
vagrant up
```

Vagrant will download the specified box (if not already downloaded) and launch the VM.

```
PS C:\Users\rishi\Desktop\vagrant__lab> vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Box 'ubuntu/trusty64' could not be found. Attempting to find and install...
    default: Box Provider: virtualbox
    default: Box Version: >= 0
==> default: Loading metadata for box 'ubuntu/trusty64'
    default: URL: https://vagrantcloud.com/api/v2/vagrant/ubuntu/trusty64
==> default: Adding box 'ubuntu/trusty64' (v20191107.0.0) for provider: virtualbox
    default: Downloading: https://vagrantcloud.com/ubuntu/boxes/trusty64/versions/20191107.0.0/providers/virtualbox/unknown/vagrant.box
Download redirected to host: cloud-images.ubuntu.com
    default:
==> default: Successfully added box 'ubuntu/trusty64' (v20191107.0.0) for 'virtualbox'!
==> default: Importing base box 'ubuntu/trusty64'...
==> default: Matching MAC address for NAT networking...
==> default: Checking if box 'ubuntu/trusty64' version '20191107.0.0' is up to date...
==> default: Setting the name of the VM: vagrant__lab_default_1724338766573_2419
==> default: Clearing any previously set forwarded ports...
Vagrant is currently configured to create VirtualBox synced folders with
the `SharedFoldersEnableSymlinksCreate` option enabled. If the Vagrant
guest is not trusted, you may want to disable this option. For more
information on this option, please refer to the VirtualBox manual:

  https://www.virtualbox.org/manual/ch04.html#sharedfolders

This option can be disabled globally with an environment variable:

  VAGRANT_DISABLE_VBOXSYMLINKCREATE=1

or on a per folder basis within the Vagrantfile:

  config.vm.synced_folder '/host/path', '/guest/path', SharedFoldersEnableSymlinksCreate: false
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
==> default: Forwarding ports...
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: vagrant
    default: SSH auth method: private key
    default:
    default: Vagrant insecure key detected. Vagrant will automatically replace
    default: this with a newly generated keypair for better security.
    default:
    default: Inserting generated public key within guest...
    default: Removing insecure key from the guest if it's present...
    default: Key inserted! Disconnecting and reconnecting using new SSH key...
==> default: Machine booted and ready!
```

- SSH into the VM:
- Connect to the running VM using SSH:

```
vagrant ssh
```



- This command will log you into the VM's shell.
- Exploring the VM:
- Inside the VM, explore the filesystem, install packages, and run commands to understand the environment.
- Stop the VM:
- Exit the SSH session by typing exit.
- Stop the VM with the following command:

```
vagrant halt
```

Destroy the VM (optional):

```
Last login: Sun Sep  8 09:16:01 2024 from 10.0.2.2
vagrant@vagrant-ubuntu-trusty-64:~$ exit
logout
Connection to 127.0.0.1 closed.
PS C:\Users\rishi\desktop\vagrant__lab> vagrant halt
==> default: Attempting graceful shutdown of VM...
```

To remove the VM completely, use the following command:

```
vagrant destroy
```

```
PS C:\Users\rishi\desktop\vagrant__lab> vagrant destroy
    default: Are you sure you want to destroy the 'default' VM? [y/N] Y
==> default: Destroying VM and associated drives...
PS C:\Users\rishi\desktop\vagrant__lab>
```

This will remove all traces of the VM, including any data stored on it.

Explore the benefits of using Vagrant for development and testing environments.

Submission:

- Submit a brief report including the Vagrantfile you configured, screenshots of the running VM, and the output of any commands run within the VM.
- Reflect on the learning experience and any challenges faced during the exercise.

This lab exercise provides a hands-on introduction to Vagrant, focusing on creating and managing virtual environments through a Vagrantfile. It offers both foundational learning and opportunities to explore more advanced features.