TASK: Installation of Virtual Machines using Vagrant

PGCSEDS

IIIT-Hyderabad

Task: To understand the open-source software Vagrant and its features.

Overview:

In a nutshell, Vagrant is a tool for working with virtual environments, and in most circumstances, this means working with virtual machines. Vagrant provides a simple and easy to use command-line client for managing these environments, and an interpreter for the text-based definitions of what each environment looks like, called Vagrantfiles. Vagrant is open source, which means that anyone can download it, modify it, and share it freely.

System Requirements:

- Laptop or a PC with minimum 8GB ram and 250 or above Hard-disk.
- Virtualization and other features (Hyper-V) enabled.
- Good Internet connectivity.

Downloading the Software:

- It is required to download the Vagrant software for both windows, Linux and Mac users.
- For Windows, Linux and Mac based installation and setup: Vagrant by HashiCorp (vagrantup.com)
- For the download of the vagrant software using CLI or GUI mode it is available at: Downloads | Vagrant by HashiCorp (vagrantup.com)

Installation of the Vagrant

- For usage of Vagrant software on windows machine you need to have an administrator access to PowerShell.
  - The setup file would be executable so just follow the installation steps as mentioned in the window while installing the software and on finish, please restart the system.
- For usage of Vagrant software on Linux machine, we need to have an administrator access to the terminal.
  - Command 1 : curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add –
  - Command 2: sudo apt-add-repository "deb [arch=amd64] https://apt.releases.hashicorp.com $(lsb_release -cs) main"
  - Command 3: sudo apt-get update && sudo apt-get install vagrant
- For usage of Vagrant software on mac machine, we need to have administrator access to the shell.
  - Command 1: brew tap hashicorp/tap
  - Command 2 : brew install vagrant

Installation of OS using boxes in Vagrant

After successful installation of the software, its time to create a folder for the vm to be installed using Vagrant.

Note:

- For windows-based machine: Please use the powershell for using the vagrant software.

- For Linux based machine please use terminal with administrative rights.
- For Mac based machine please use the terminal with administrative rights.

Basic Commands to run the virtual machine on the vagrant

  - Command 1 : mkdir ubuntuvm1 (Creates a empty folder with ubuntuvm1)
  - Command 2 : vagrant init hashicorp/bionic64 (Extracts the OS files and initializes the OS installation)
  - Command 3: vagrant up (After installation it starts the VM)
  - Command 4 : vagrant ssh (Use to login to the created VM using vagrant)

Note:

- The commands should be used under the administrative rights.
- Ensure that the virtualization and Hyper-V all are enabled.

Network Configuration for VM's:

This part is mainly to understand the ideology behind the working of distributed systems. Here we configure the installed Virtual Machine and try to get access to the other virtual machine for executing our tasks.

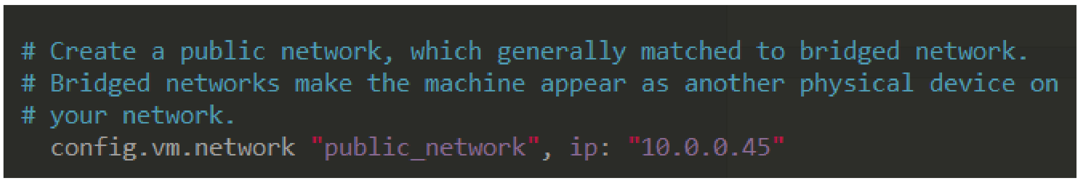
There are commands which you need to follow:

For Virtual Machine 1 :

- Command 1: sudo apt-get update (Will update the necessary softwares)
- Command 2: ifconfig (Note the ip address)
- Command 3 : sudo apt-get install vim
- Command 4: logout
- Command 5 : sudo vim Vagrantfile. (Open's vagrantfile)

Under the Vagrant file, please change the configuration to:

You will find the configuration as shown below:



```
# Create a public network, which generally matched to bridged network.
# Bridged networks make the machine appear as another physical device on
# your network.
  config.vm.network "public_network", ip: "10.0.0.45"
```

- Please use the ip address that you have captured in the above step and replace it with your ipaddress.
- Save the file and use the command : vagrant reload (It reloads the VM)

This would reload the VM and assign you a network with which you can communicate via using ping to other virtual machine.

Installation and Network Configuration of VM 2:

- Please follow the same instructions as mentioned in the section **Installation of OS using boxes in Vagrant.**

- You just need to use a new directory to install and run the new virtual machine.
- Infact, you call also just change the last digit of the ip address in the vagrant file of the VM2. By which that would be your private Ip address for VM2.
- Check the communication using the ping between two VM's.

Python, Flask based web application on the Virtual Machine.

- After the VM is completely setup, create a folder of your choice and follow the commands to deploy the web application.

Objective:

1. **Setup a python environment**
2. **Install the flask package.**
3. **Run a web-based application and deploy in the other Virtual Machines.**

Installation of Python3 and its packages:

- Please follow the below commands, if there is any difficulty in installing the python packages.
    - Command 1: sudo apt update
    - Command 2: sudo apt install software-properties-common
    - Command 3: sudo add-apt-repository ppa:deadsnakes/ppa
    - Command 4: sudo apt install python3.9
- Verification of the installation was successful.
    - Command: python3.9 –version

That's it. Python 3.9 is installed on your Ubuntu, and you can start using it.

Setup and configuration:

- To set up a Python 3 virtual environment, navigate to your project folder on your terminal and type the following command:
    - Command: python3 -m venv venv

This will create a new virtual environment named venv using the version of Python 3 that you have installed on your system. Next, you need to activate the virtual environment by sourcing the activation script:

- Command: source venv/bin/activate

After executing this command, your prompt will change to indicate that you're now operating from within the virtual environment.

The next file to look at is requirements.txt. Since Flask is the only dependency of this project, that's all you need to specify:

- Command 1: sudo apt-get install vim
- Command 2: vim requirements.txt
- Under requirements.txt -> Flask==1.1.2

If your app has other dependencies, then you'll need to add them to your requirements.txt file as well.

After you successfully set up and activate your virtual environment, you're ready to install Flask:

- Command: python3 -m pip install -r requirements.txt

This command fetches all packages listed in requirements.txt from PyPI and installs them in your virtual environment. In this case, the only package installed will be Flask.

Wait for the installation to complete, then open up main.py and add the following lines of code to the file:

```
from flask import Flask
app = Flask(__name__)
@app.route("/")
def index():
   return "Congratulations, it's a web app!"
if __name __ == "__main__":
   app.run(host="0.0.0.0", port=5000, debug=True)
```

You can now start Flask's development server and interact with your Python app in your browser. To do so, you need to run the Python script that starts the Flask app by typing the following command:

- Command: python3 main.py

Flask starts up the development server, and your terminal will display output similar to the text shown below:

```
Serving Flask app "main" (lazy loading)
 * Environment: production
   WARNING: This is a development server.
   Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://<ip-address>:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: <random generated>
```

Installation and usage of Multiple Virtual Machine:

Virtual machine 1 (vm1): This virtual machine can be used for the execution of the sample code as described above.

Virtual Machine 2 (vm2): This machine can be used to render the output of the tasks that are being performed on the Virtual Machine 1 (vm1).

Services that run on Virtual Machine 1 (vm1): (Development)

- This virtual machine will be exclusively used for developing and executing of the sample python based application as described in the previous sections.

- This machine would also have mongodb services configured which later is used to create and integrate with the flask-based application. Detailed setup and configuration information are available at abhaymehtre/Mongodb-with-Flask-Web-Application (github.com)

Services that run on Virtual Machine 2 (vm2) (Deployment and Delivery)

- This virtual machine will be used only for the execution purpose and also to analyse the connectivity between two machines connected to each other.

Note :

- You need to follow the installation steps for all the VM machine you create as per the details mentioned in the section "**Installation Steps for Windows:**" and "**Python, Flask based web application on the Virtual Machine.**"
- If you have an linux based system/machine please follow the initial steps from the section **Installation of OS using boxes in Vagrant, then follow the similar steps as mentioned in the network configuration for VM's.**
- The Virtual Machine 1 will have the flask-based app developed and deployed using the ip address mentioned in previous section. You can also refer to the github repository, for the mongodb and flsk integration. Link : abhaymehtre/Mongodb-with-Flask-Web-Application (github.com)
- The Result or the execution part of the web application can be rendered using the ip address obtained after running the flask application on the virtual machine 2.