Vagrant Workshop

Parsa Mohammadian

November 17, 2022

Contents

Figure 1: Vagrant logo

1 Introduction

1.1 What is Vagrant?

- Vagrant is a tool for building and managing virtual machine environments

- With an easy-to-use workflow and focus on automation

  - Vagrant lowers development environment setup time
  - Increases production parity
  - Makes the "works on my machine" excuse a relic of the past

1.2 How does it works?

- To achieve its magic, Vagrant stands on the shoulders of giants

- Machines are provisioned on top of standard providers

  - VirtualBox
  - VMware
  - AWS
  - Hyper-V
  - Docker

- Standard provisioning tools can automatically install and configure software on VM

  - Shell Script
  - Ansible
  - Chef
  - Puppet

1.3 Prerequisite

- Install VirtualBox Download VirtualBox Note that current version of Vagrant (2.3.0) is compatible with VirtualBox version 6.1 and bellow

- Install Vagrant Download Vagrant

2 Get started

2.1 Initialize a project directory

2.1.1 Create a directory

```
mkdir my_first_vagrant_project
cd my_first_vagrant_project
```

2.1.2 Initialize the project

```
vagrant init ubuntu/jammy64
```

- This command will create a file named Vagrantfile under the working directory using ubuntu/jammy64 as the base box

  - Boxes are the package format for Vagrant environments. There are ready to use boxes for ubuntu, centos, . . .

- Vagrantfile describes the type of machine required for a project, and how to configure and provision these machines

- We explore the content of this Vagrantfile later in the presentation

2.2 Boot the environment

2.2.1 Bring up the previously initialized project

```
vagrant up
```

- First time you boot into a box, Vagrant will download the box from HashiCorp's Vagrant Cloud box catalog

  - The ubuntu/jammy64 box is about 600 MB. So be patient :D

- After this command finished, you will not actually see anything, because Vagrant runs the VM without UI

  - So only check for errors. If there was any, try rerunning the command

2.2.2 SSH into the machine

`vagrant ssh`

- This command needs neither passowrd nor manual SSH key in order to login

    - Vagrant maintains an internal SSH key for that automatically!
    - Vagrant also use proper port forward for the created VM

- Vagrant also shares project directory with the VM in /vagrant

    - Easily copy files between host and VM!

- Exit the SSH session however you wish

2.2.3 Turn off the machine and save its state

`vagrant halt`

- This command will save state of the machine and release hardware resources

- The command will bring up the machine with saved state

2.2.4 Destroy the machine

`vagrant destroy`

- This command stops the machine and remove all the associated resources

- Keep in mind that Vagrantfile won't be deleted, since it contains the blueprint for the VMs

3 Advanced topics

3.1 Vagrantfile

Lets take a look at previous section Vagrantfile

```
# Print Vagrantfile without empty lines and comments
cat Vagrantfile | egrep -v "#|^$"
```

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

- The syntax of Vagrantfiles is Ruby, which is a dynamic, open source programming language

- This allows lots of flexabilties when working with Vagrantfiles

- Structure of this simple Vagrantfile

  – The "2" in the first line above represents the version of the configuration object that will be used for configuration

  – After that, there is a block (the section between the do and the end) that defines configuration object
    * This object can be very different from version to version
    * All of the configurations should be inserted here

  – Then there is a simple field assignment which sets the base box

3.1.1 Tips and tricks

1. Configure VM hardware resources

   ```
   Vagrant.configure("2") do |config|
     config.vm.box = "ubuntu/jammy64"
     config.vm.provider "virtualbox" do |vb|
       vb.cpus = 2
       vb.memory = "2048"
     end
   end
   ```

   Explanation:

   - The config.vm.provider directive is a function which is called with two parameters
     – Name of the provider, here is "virtualbox"
     – A block which contains configurations
       * A simple field assignment which sets number of cpu cores
       * Anoter simple field assignment which sets amount of memory (in MB)

2. Define multiple VMs in a Vagrantfile

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/jammy64"
  config.vm.define "First VM" do |first_vm|
    first_vm.vm.hostname = "firstvm"
  end
end
```

Explanation:

- Everything is like previous example
- The $first_{vm.vm.hostname}$ variable sets the hostname of the VM

3. Loop over VM definitions

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/jammy64"
  (1..3).each do |i|
    config.vm.define "node-#{i}" do |node|
    node.vm.hostname = "machine-#{i}"
  end
end
```

3.2   Provisioning

- Provisioners in Vagrant allow you to automatically install software, alter configurations, and more on the machine as part of the first vagrant up process

- This is useful since boxes typically are not built perfectly for your use case

- Of course, if you want to just use vagrant ssh and install the software by hand, that works

  - But by using the provisioning systems built-in to Vagrant, it automates the process so that it is repeatable and without any human interaction

- There are lots of provisioners for Vagrant

  - File provisioner

- Shell provisioner
- Ansible provisioner
- Chef provisioner
- Docker provisioner
- Podman provisioner
- Puppet provisioner
- Salt provisioner

3.2.1 File provisioner

The Vagrant file provisioner allows you to upload a file or directory from the host machine to the guest machine

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/jammy64"
  config.vm.provision "file", source: "~/.gitconfig", destination: ".gitconfig"
end
```

3.2.2 Shell provisiner

The Vagrant Shell provisioner allows you to upload and execute a script within the guest machine

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/jammy64"
  config.vm.provision "shell", inline: <<-SCRIPT
    sudo apt update
    sudo apt install nginx
  SCRIPT
end
```

4 Credits

- Vagrant official documentation

- Vagrant quick start tutorial