Install Vagrant on Ubuntu

To install Vagrant on your Ubuntu system, follow these steps:

1. Installing VirtualBox

The first step is to install the VirtualBox package which is available in the Ubuntu’s repositories:

**sudo apt install virtualbox**

1. Installing Vagrant

We’ll download and install the latest version of Vagrant from the official Vagrant site.

Check the Vagrant Download page to see if a newer version is available.

Start by updating the package list with:

**sudo apt update**

Download the Vagrant package using the following curl command:

**curl -O https://releases.hashicorp.com/vagrant/2.2.6/vagrant\_2.2.6\_x86\_64.deb**

Once the .deb file is downloaded, install it by typing:

**sudo apt install ./vagrant\_2.2.6\_x86\_64.deb**

1. Verify Vagrant installation

To verify that the installation was successful, run the following command which prints the Vagrant version:

**vagrant --version**

The output should look something like this:

Vagrant 2.2.6

Getting Started with Vagrant

Now that Vagrant is installed on your Ubuntu system let’s create a development environment.

The first step is to create a directory which will be the project root directory and hold the Vagrantfile file. Vagrantfile is a Ruby file that describes how to configure and provision the virtual machine.

Create the project directory and switch to it with:

**mkdir ~/vagrant**

**cd ~/vagrant**

Next, initialize a new Vagrantfile using the vagrant init command and specify the box you want to use.

Boxes are the package format for the Vagrant environments and are provider-specific. You can find a list of publicly available Vagrant Boxes on the Vagrant box catalog page.

In this project, we will use the centos/7 box. Run the following command to initialize a new Vagrantfile:

**vagrant init centos/7**

A `Vagrantfile` has been placed in this directory. You are now

ready to `vagrant up` your first virtual environment!

You can open the Vagrantfile, read the comments and make adjustments according to your needs.

For our project we have updated our Vagrantfile with this

# -\*- mode: ruby -\*-

# vi: set ft=ruby :

# All Vagrant configuration is done below. The "2" in Vagrant.configure

# configures the configuration version (we support older styles for

# backwards compatibility). Please don't change it unless you know what

# you're doing.

# variables

NUM\_MASTER\_NODES = 1

NUM\_WORKER\_NODES = 2

IP\_NW = "192.168.5."

MASTER\_IP\_START = 10

NODE\_IP\_START = 20

LB\_IP\_START = 30

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/bionic64"

# 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

# Provision master nodes

(1..NUM\_MASTER\_NODES).each do |i|

config.vm.define "master-#{i}" do |node|

# Name shown in the GUI

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

vb.name = "kubernetes-ha-master-#{i}"

vb.memory = 2048

vb.cpus = 2

end

node.vm.hostname = "master-#{i}"

node.vm.network :private\_network, ip: IP\_NW + "#{MASTER\_IP\_START + i}"

node.vm.network "forwarded\_port", guest: 22, host: "#{2710 + i}"

node.vm.provision "setup-hosts", :type => "shell", :path => "ubuntu/vagrant/setup-hosts.sh" do |s|

s.args = ["enp0s8"]

end

node.vm.provision "setup-dns", type: "shell", :path => "ubuntu/update-dns.sh"

end

end

# Provision worker nodes

(1..NUM\_WORKER\_NODES).each do |i|

config.vm.define "worker-#{i}" do |node|

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

vb.name = "kubernetes-ha-worker-#{i}"

vb.memory = 512

vb.cpus = 1

end

node.vm.hostname = "worker-#{i}"

node.vm.network :private\_network, ip: IP\_NW + "#{NODE\_IP\_START + i}"

node.vm.network "forwarded\_port", guest: 22, host: "#{2720 + i}"

node.vm.provision "setup-hosts", :type => "shell", :path => "ubuntu/vagrant/setup-hosts.sh" do |s|

s.args = ["enp0s8"]

end

node.vm.provision "setup-dns", type: "shell", :path => "ubuntu/update-dns.sh"

node.vm.provision "install-docker", type: "shell", :path => "ubuntu/install-docker.sh"

node.vm.provision "allow-bridge-nf-traffic", :type => "shell", :path => "ubuntu/allow-bridge-nf-traffic.sh"

end

end

# Provision load balancer

config.vm.define "loadbalancer" do |node|

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

vb.name = "kubernetes-ha-lb"

vb.memory = 512

vb.cpus = 1

end

node.vm.hostname = "loadbalancer"

node.vm.network :private\_network, ip: IP\_NW + "#{LB\_IP\_START}"

node.vm.network "forwarded\_port", guest: 22, host: 2730

node.vm.provision "setup-hosts", :type => "shell", :path => "ubuntu/vagrant/setup-hosts.sh" do |s|

s.args = ["enp0s8"]

end

node.vm.provision "setup-dns", type: "shell", :path => "ubuntu/update-dns.sh"

end

# 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

# Create a forwarded port mapping which allows access to a specific port

# within the machine from a port on the host machine and only allow access

# via 127.0.0.1 to disable public access

# config.vm.network "forwarded\_port", guest: 80, host: 8080, host\_ip: "127.0.0.1"

# Create a private network, which allows host-only access to the machine

# using a specific IP.

# config.vm.network "private\_network", ip: "192.168.33.10"

# 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"

# Share an additional folder to the guest VM. The first argument is

# the path on the host to the actual folder. The second argument is

# the path on the guest to mount the folder. And the optional third

# argument is a set of non-required options.

# config.vm.synced\_folder "../data", "/vagrant\_data"

# Provider-specific configuration so you can fine-tune various

# backing providers for Vagrant. These expose provider-specific options.

# Example for VirtualBox:

#

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

# # Display the VirtualBox GUI when booting the machine

# vb.gui = true

#

# # Customize the amount of memory on the VM:

# vb.memory = "1024"

# end

#

# View the documentation for the provider you are using for more

# information on available options.

# Enable provisioning with a shell script. Additional provisioners such as

# Ansible, Chef, Docker, Puppet and Salt are also available. Please see the

# documentation for more information about their specific syntax and use.

# config.vm.provision "shell", inline: <<-SHELL

# apt-get update

# apt-get install -y apache2

# SHELL

end

Run the vagrant up command to create and configure the virtual machine as specified in the Vagrantfile:

vagrant up

==> default: Configuring and enabling network interfaces...

default: SSH address: 192.168.121.74:22

default: SSH username: vagrant

default: SSH auth method: private key.

To ssh into the virtual machine, run:

vagrant ssh

You can stop the virtual machine with the following command:

vagrant halt

The following command stops the machine if it is running, and destroys all resources created during the creation of the machine: