**Name of Title:** Learning Nginx

**Video Name:** 00\_06 Using Vagrant to create a Ubuntu VM

**Estimated Length:**

**Author Name:** Michael Jenkins

**Chapter\_Section\_Video:**

**Video Objective:**

At the end of this video the learner will be able to create a Ubuntu VM using vagrant and virtualbox

**Introductory Statement:**

Type your introductory statement here.

**Speaking Points:**

1. Make sure vagrant and virtualbox are installed correctly
   1. Refer to previous lesson
2. Download/Create Vagrantfile
   1. Put in folder on desktop or workspace
3. Run `vagrant up`
4. SSH to VM using `vagrant ssh`
5. Disconnect from VM
6. Stop VM using `vagrant halt`
   1. Show how VM can be restarted with `vagrant up`
7. Destroy VM using `vagrant destroy`

**Script:**

Now that we have Virtualbox and Vagrant installed, let’s create our first VM.

On my system, I’ve downloaded the exercise files and unzipped the contents into my downloads folder. I’ve opened a terminal and navigated to the folder for chapter 1 and lesson 1.

This directory has a Vagrantfile that describes the VM we’ll be creating. Let’s take a look inside this file:

cat Vagrantfile

Even if you don’t know Ruby, the programming language that Vagrantfiles are written in, its fairly straightforward to figure out what’s going on.

The first line in this file set the IP address that the VM will be configured to use.

**If you already have something running on your system that uses this IP address, you can avoid IP address conflicts by using a different IP address for your vagrant box.**

The following lines configure the VM by calling out the base box and the network configuration. In this case, the base box is the Ubuntu 18.04 box from the bento library. This base box and others like it are freely available on the web.

The last few lines of the Vagrantfile print out a message with a URL with the VM’s IP address.

Exit file and/or clear screen

To start the VM, we simply run “vagrant up”.

This will start the process of loading the base box. If you haven’t downloaded the base box before, vagrant will download it for you now. After this download, we won’t have to down the base box again for other VMs.

After the vagrant up command completes, the VM is running and we need to connect to it.

We do that with the “vagrant ssh” command. This will log us into the VM with SSH, or Secure Shell, as the default user which in this case is named vagrant.

Once we’re logged in we have full control of the VM just like any other linux server. We log out of the vm by typing exit or CTRL+D.

To stop the VM, we run the command “vagrant halt”. Doing this powers down the VM and preserves the current state. If we want to pick up where we left off, we can restart the VM by running vagrant up again.

TYPE VAGRANT UP BUT DON”T RUN IT

DELETE AND THEN TYPE VAGRANT DESTROY

If we’re done with the VM, we can completely remove it by running “vagrant destroy”. This stops the VM if its running and then removes any files associated with the VM. This command does not remove the Vagrantfile.

**I’ll type the vagrant destroy command here, but you don’t need to run this command at this time.**

Using vagrant and these few commands, we can easily create and manage VMs for our sandbox environment.

For a more indepth lesson on using vagrant, you can review “Learning Vagrant” in the course library.

EDITOR: FLY IN “Learning Vagrant: <https://www.lynda.com/Vagrant-tutorials/Learning-Vagrant/685028-2.html>

**Conclusion:**

Type your conclusion statement here.

**Script and Media:**

Break the script up into parts and align it with any media (slides, web, CLI, etc.)

| **Part** | **Script** | **Media** |
| --- | --- | --- |
|  |  |  |

**Exercise Files:**

Vagrantfile:

guest\_ip = "192.168.0.3"

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

config.vm.box = "bento/ubuntu-18.04"

config.vm.network "private\_network", ip: guest\_ip

end

puts "-------------------------------------------------"

puts "Default Homepage : http://#{guest\_ip}"

puts "-------------------------------------------------"

**Basement:**

Open CMD or powershell on Windows or terminal on Mac and run “VBoxManage --version” and “vagrant --version” this will print the version for each application.