➔ Follow the installation procedure and setup the vm.

Snapshots

With snapshots, you can save a particular state of a virtual machine for later use. At any later time, you can revert to that state, even though you may have changed the VM considerably since then.

To take snapshot of a machine right click on the vm and click on snapshot.

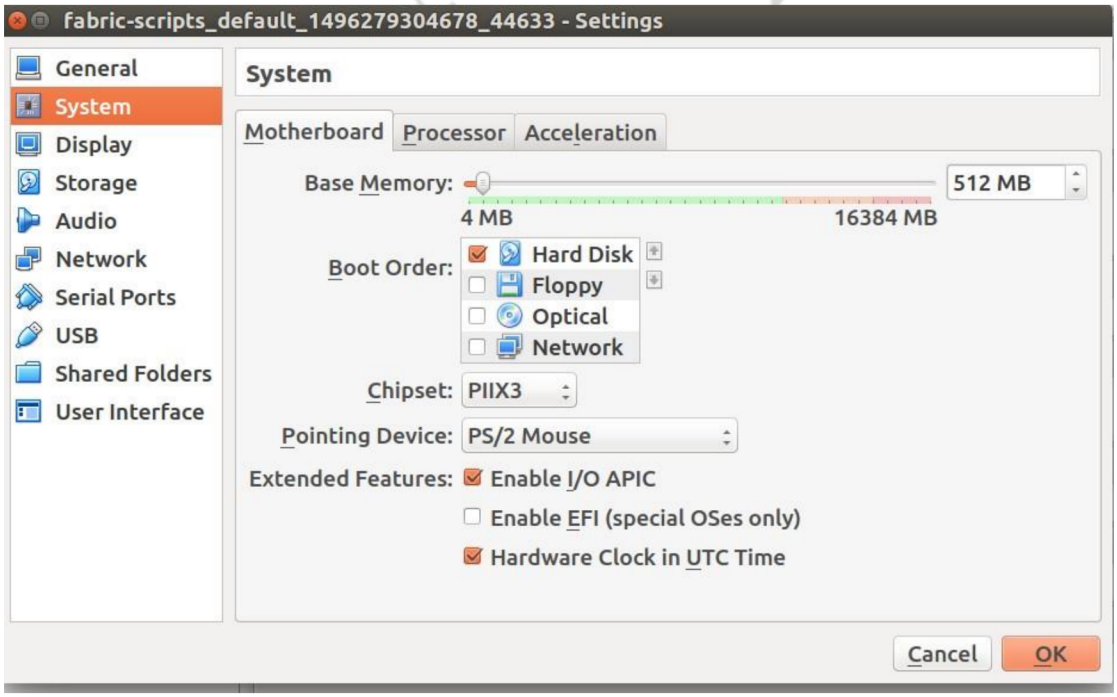
When you want to restore machine from snapshot, find the snapshot location on your filesystem and simply double click on it.

Virtual Machine configuration

When you select a virtual machine from the list in the Manager window, you will see a summary of that machine's settings on the right.

Right click on the vm and open settings window.

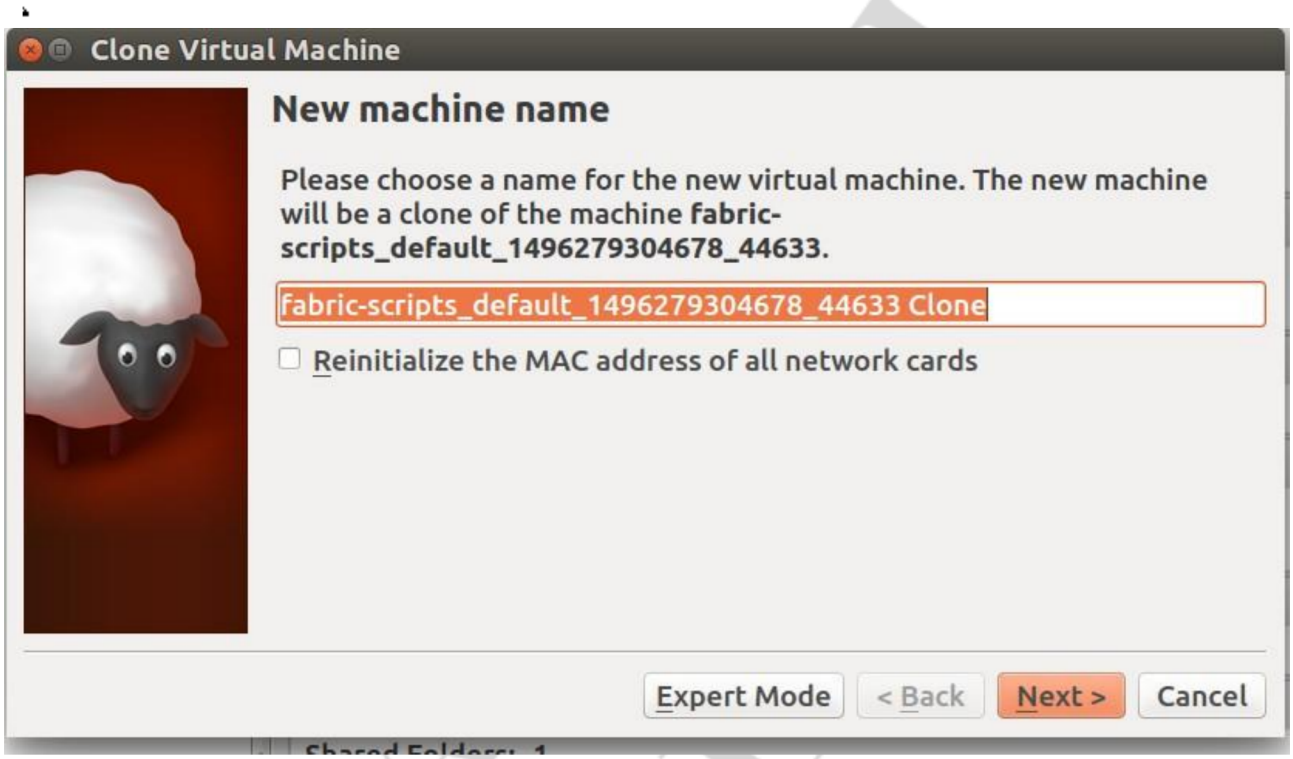
You can configure various configuration of the vm's like RAM size, number of CPU's, Network etc.



Cloning virtual machines

To experiment with a VM configuration, test different guest OS levels or to simply backup a VM, VirtualBox can create a full or a linked copy of an existing VM.[5]

A wizard will guide you through the clone process:



Full clone: In this mode, all depending disk images are copied to the new VM folder. The clone can fully operate without the source VM.

Linked clone: In this mode, new differencing disk images are created where the parent disk images are the source disk images. If you selected the current state of the source VM as clone point, a new snapshot will be created implicitly.

Virtualbox gives you many more features which is out of the scope of this tutorial to cover.

Creating one single vm with the OS and application setup is time consuming process. You may not have great deal of issue with it if we are talking about one or two vm's but being a devops you may need to setup lot of vm's on regular basis. You may test your script or automation tool execution on variety of applications and for that you may have to setup multiple vm's. One of your job is to research on tools, lots of tools and to learn and implement those tools you would be setting up lot of vm's and even share the same setup with your team.

To get around this very problem of setting up vm's manually we can automate vm setup and management process. This is the place where I introduce to a VM lifecycle automation tool known as Vagrant.

IV. Introduction to Vagrant

Vagrant is a tool for building and managing virtual machine environments in a single workflow. With an easy-to-use workflow and focus on automation, Vagrant lowers development environment setup time, increases productivity.

1. Why Vagrant?

As per Vagrant Documentation.

If I put it in a single sentence, there is no such tool as vagrant for managing virtual machine lifecycle. Vagrant gets integrated with hypervisors like VirtualBox and gives as a command line interface to automate vm lifecycle.

Then, industry-standard provisioning tools such as shell scripts, Ansible, Chef, or Puppet, can automatically install and configure software on the virtual machine.

For Developers

If you are a **developer**, Vagrant will isolate dependencies and their configuration within a single disposable, consistent environment, without sacrificing any of the tools you are used to working with (editors, browsers, debuggers, etc.). Once you or someone else creates a single Vagrantfile, you just need to `vagrant up` and everything is installed and configured for you to work. Other members of your team create their development environments from the same configuration, so whether you are working on Linux, Mac OS X, or Windows, all your team members are running code in the same environment, against the same dependencies, all configured the same way. Say goodbye to "works on my machine" bugs.

For Operators

If you are an **operations engineer** or **DevOps engineer**, Vagrant gives you a disposable environment and consistent workflow for developing and testing infrastructure management scripts. You can quickly test things like shell scripts, Chef cookbooks, Puppet modules, and more using local virtualization such as VirtualBox or VMware. Then, with the *same configuration*, you can test these scripts on remote clouds such as AWS or RackSpace with the *same workflow*. Ditch your custom scripts to recycle EC2 instances, stop juggling SSH prompts to various machines, and start using Vagrant to bring sanity to your life.

For Designers

If you are a **designer**, Vagrant will automatically set everything up that is required for that web app in order for you to focus on doing what you do best: design. Once a developer configures Vagrant, you do not need to worry about how to get that app running ever again. No more bothering other developers to help you fix your environment so you can test designs. Just check out the code, `vagrant up`, and start designing.

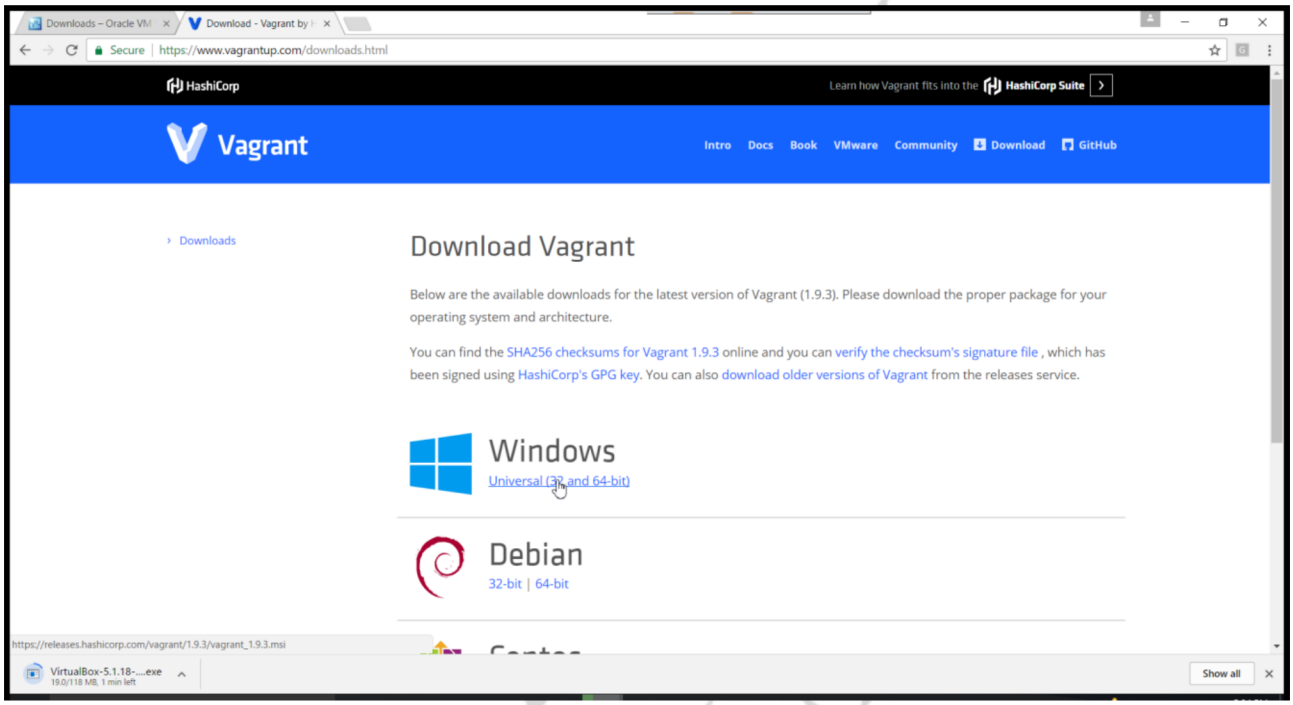
For Everyone

Vagrant is designed for everyone as the easiest and fastest way to create a virtualized environment!

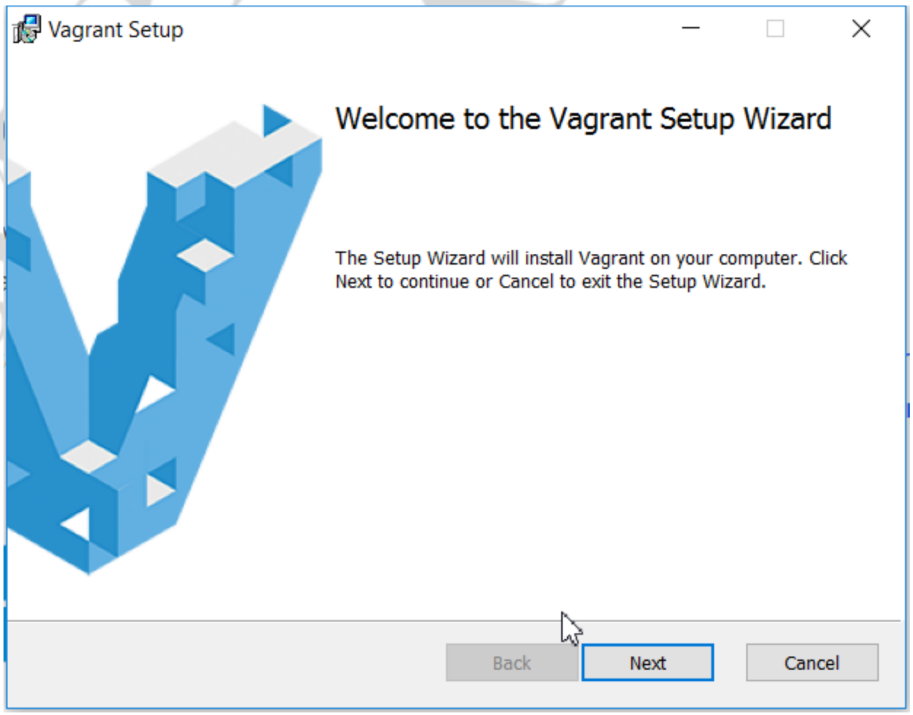
2. Installing Vagrant on windows.

Download Vagrant.

➔ Go to Vagrant Download page, Click Windows Installer.



➔ Open vagrant msi file and follow the installation wizard.

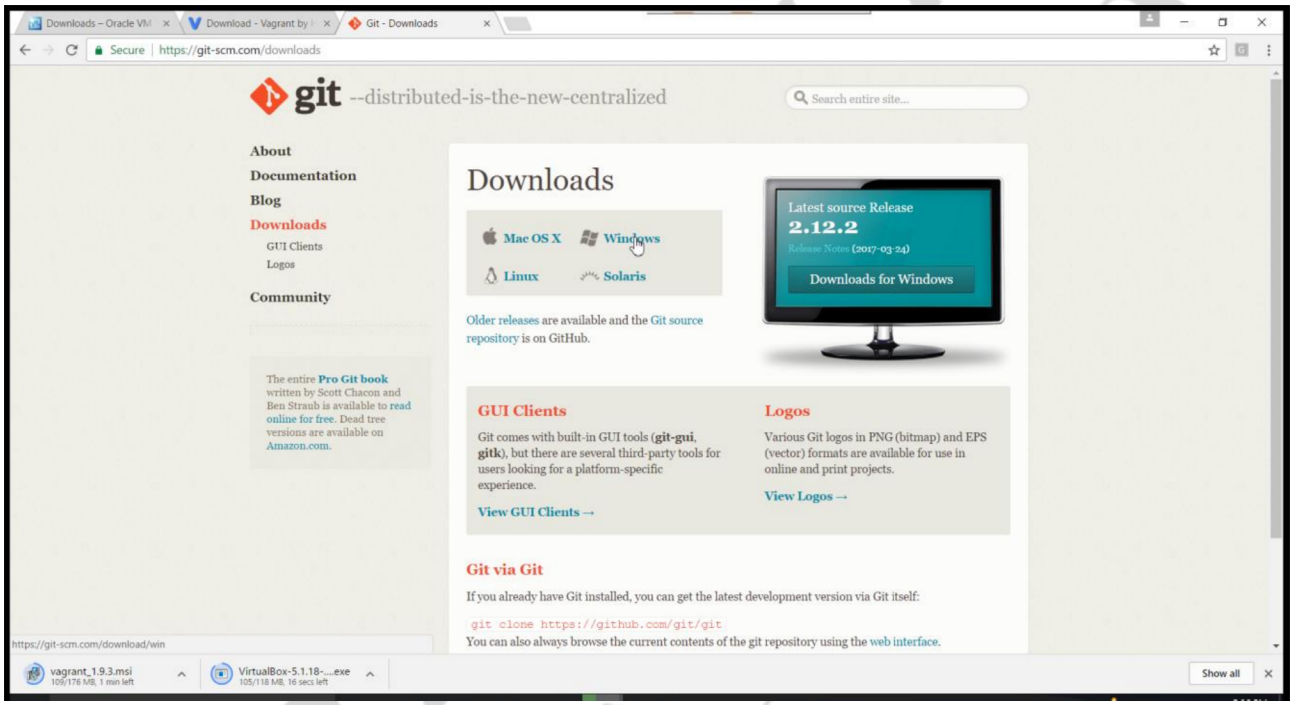


3.Git Bash CLI

As stated earlier vagrant is a command line tool and windows command prompt is just sluggish to run vagrant commands. So, we will need a better command line utility, one of them is Git bash.

Download git for windows.

➔ Go to git scm download page, Select windows.



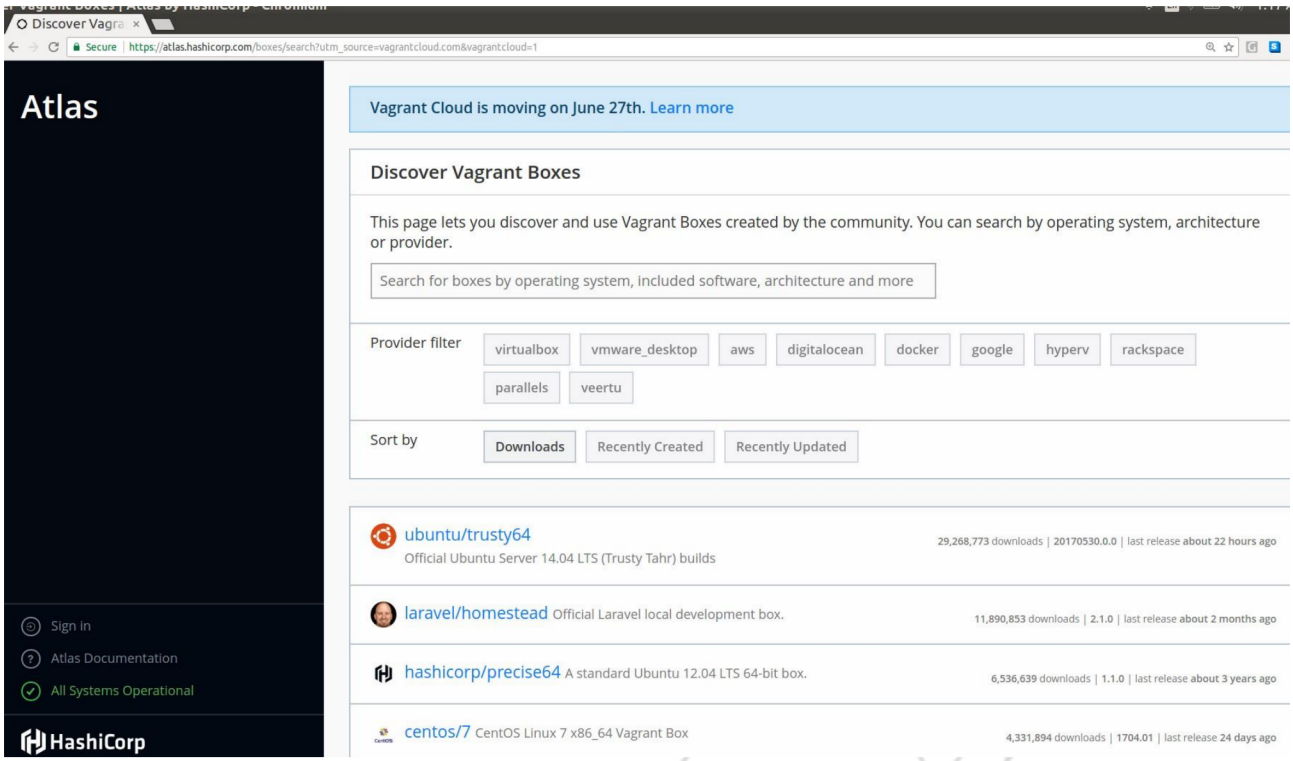
➔ Open git installable and follow the Installation wizard, take all the default settings in the wizard.

4.Vagrant Cloud.

The biggest USP of vagrant is the vagrant cloud. There is no OS installation procedures when we use vagrant. Vagrant solves that problem by giving us vagrant boxes.

Vagrant boxes are VM images which already has the OS and softwares installed in them. We just need to download these boxes from vagrant cloud by using our vagrant CLI. Once the box is downloaded you can create as many as VM's from them.

https://atlas.hashicorp.com/boxes/search?utm_source=vagrantcloud.com&vagrantcloud=1



Majority of the boxes are of VirtualBox but there are many other boxes that supports other hypervisors like Vmware workstation, livirt, hyperv etc.

In the next exercise, we will setup an Ubuntu and Centos VM with vagrant.

5. Vagrantfile

The first step of setting up VM with vagrant is to create Vagrantfile. Vagrantfile has all the information about the virtual machine, which vagrant will read and configure vm for us.

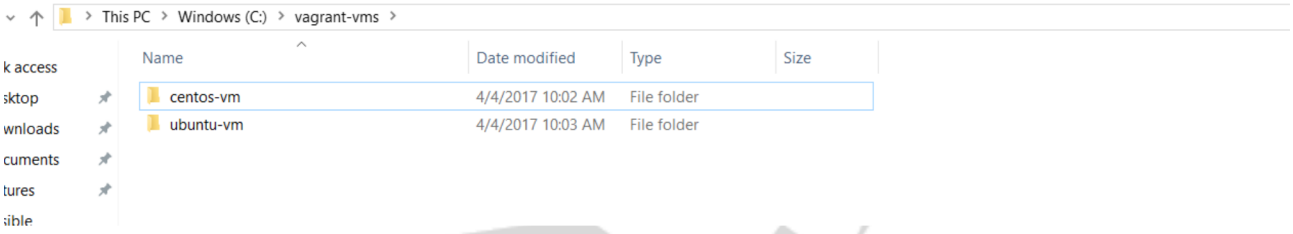
The primary function of the Vagrantfile is to describe the type of machine required for a project, and how to configure and provision these machines. Vagrantfiles are called Vagrantfiles because the actual literal filename for the file is `Vagrantfile`.

The syntax of Vagrantfiles is Ruby, but knowledge of the Ruby programming language is not necessary to make modifications to the Vagrantfile, since it is mostly simple variable assignment. In fact, Ruby is not even the most popular community Vagrant is used within, which should help show you that despite not having Ruby knowledge, people are very successful with Vagrant.

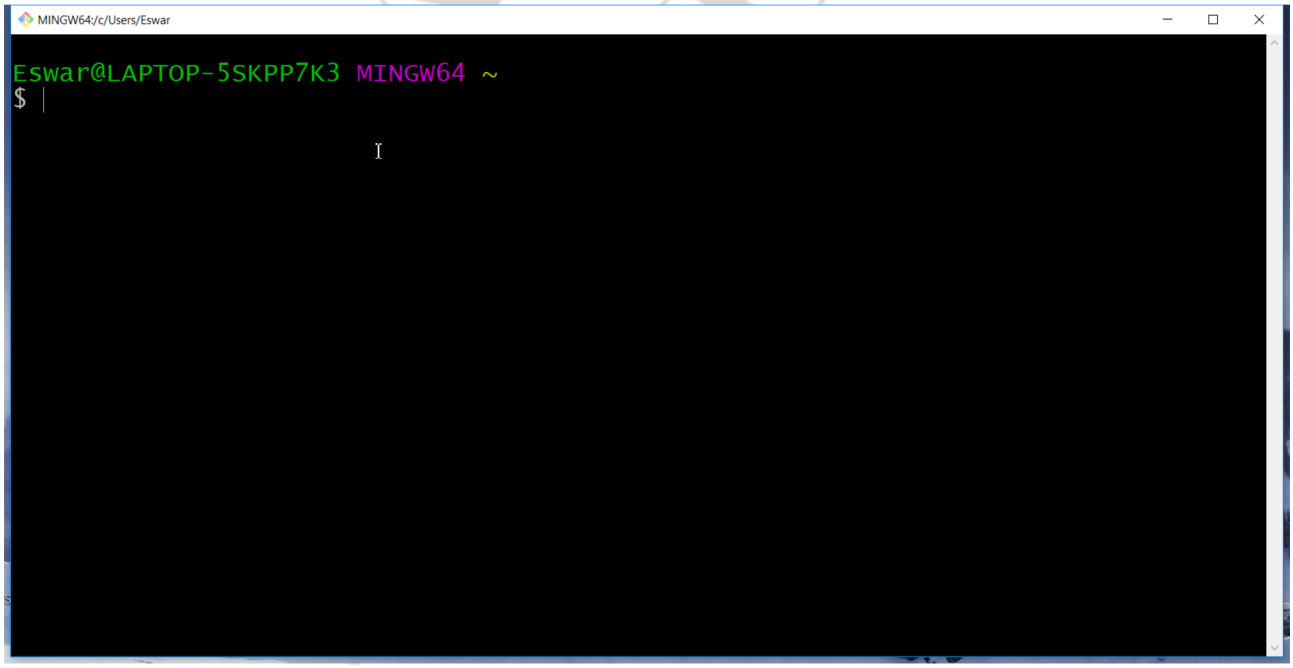
6. Creating Centos & Ubuntu VM's using Vagrant tool.

➔ **Go to C drive => Create folder named vagrant-vms**

➔ **Go to C:\vagrant-vms => Create two folders named centos-vm & ubuntu-vm**



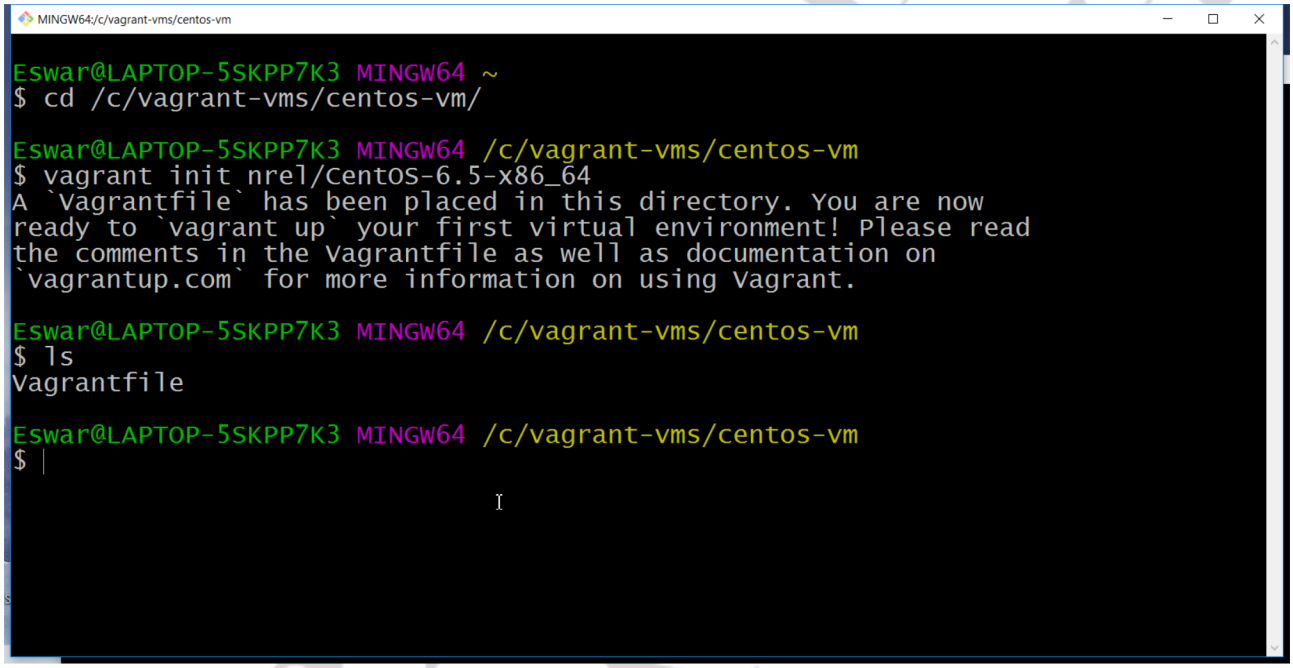
➔ **Open git bash.**



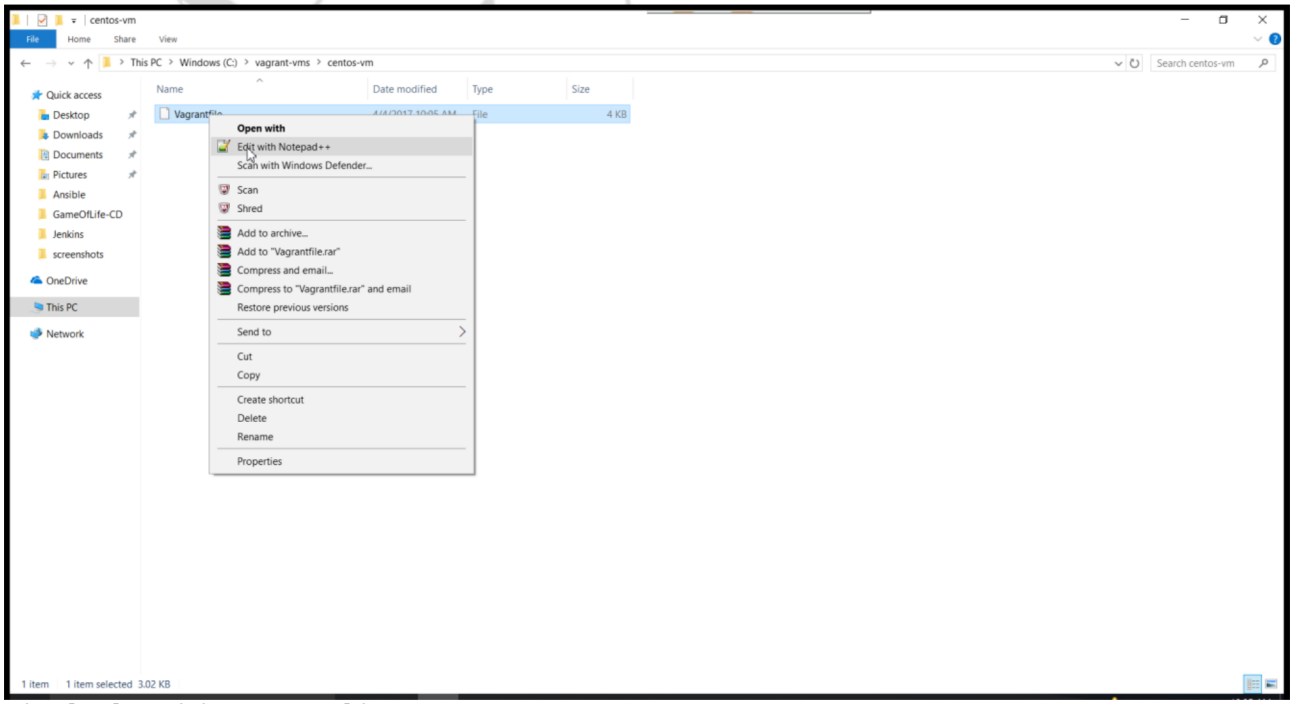
7. Creating a Vagrantfile

Vagrant init command creates a sample vagrant file. We can create a Vagrantfile manually also and put all the configuration of the vm. But just to keep it simple initially we will use vagrant init command.

1) cd /c/vagrant-vms/centos-vm
2) vagrant init nrel/Centos-6.5-x86_64
3) ls

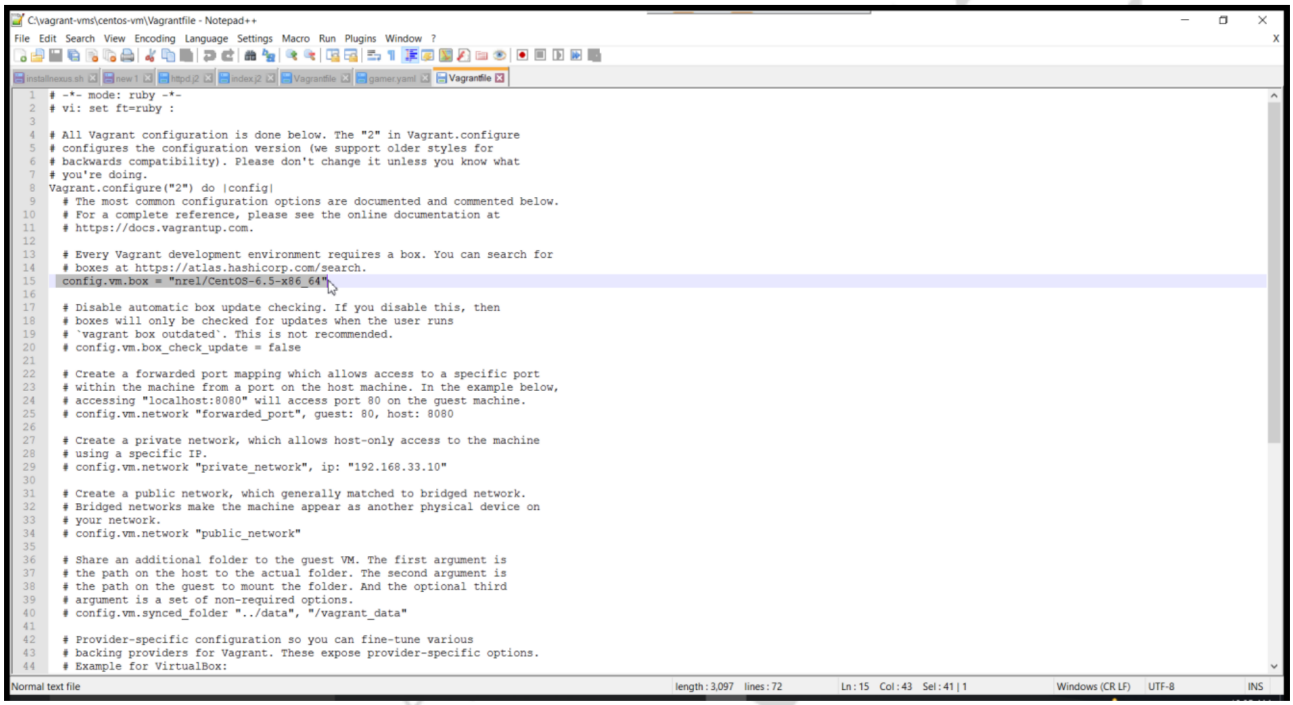


➔ **Go to centos-vm folder and check the file => Open it with Notepad++ editor.**



8. First look at a Vagrantfile.

As we have hit vagrant init <boxname> command, it created a sample Vagrantfile and placed box name as shown in the screenshot. Box name should match with the name of box in the vagrant cloud, as it is going to download the box from vagrant cloud.



➔ **Close notepad++ after verification & Open git bash again.**

9. Vagrant up

➔ **Go to centos-vm folder from git bash and run "vagrant up" command.**
Vagrant up commands read the configuration from the Vagrantfile.

This is the single most important command in Vagrant, since it is how any Vagrant machine is created. Anyone using Vagrant must use this command on a day-to-day basis.

As we just specified the boxname its going to take other settings as defaults. Default settings include RAM size, CPU, hard disk, network etc.