Starting with Vagrant and VirtualBox: A Basic Ubuntu 14.04 LTS Server

As a part of my #vDM30in30 post series, I've got some lab work that we will be doing together. This is the start which will be the foundation for the next set of posts that work with Vagrant and VirtualBox.

We are about to go on an adventure together! Using Vagrant is becoming a standard part of every day for me. As a frequent tester of new technology, one of the most important tools in my toolkit is automation. My tool of choice for that is nearly always Vagrant.

Install Vagrant for your OS of choice from here: https://www.vagrantup.com/downloads.html

Install VirtualBox for your OS of choice from here: https://www.virtualbox.org/wiki/Downloads

Once you're installed and ready, we will create our new Vagrant configuration together.

Create a Directory and Initialize a Vagrant Configuration

First, we need to create our directory where our configuration will go, and in my case I've created a simple folder named VagrantIntro on my OS X machine. The same process holds true for Windows systems, so this will make things easier as we go.

Now we need to initialize a Vagrant configuration with a very simple command `vagrant init`

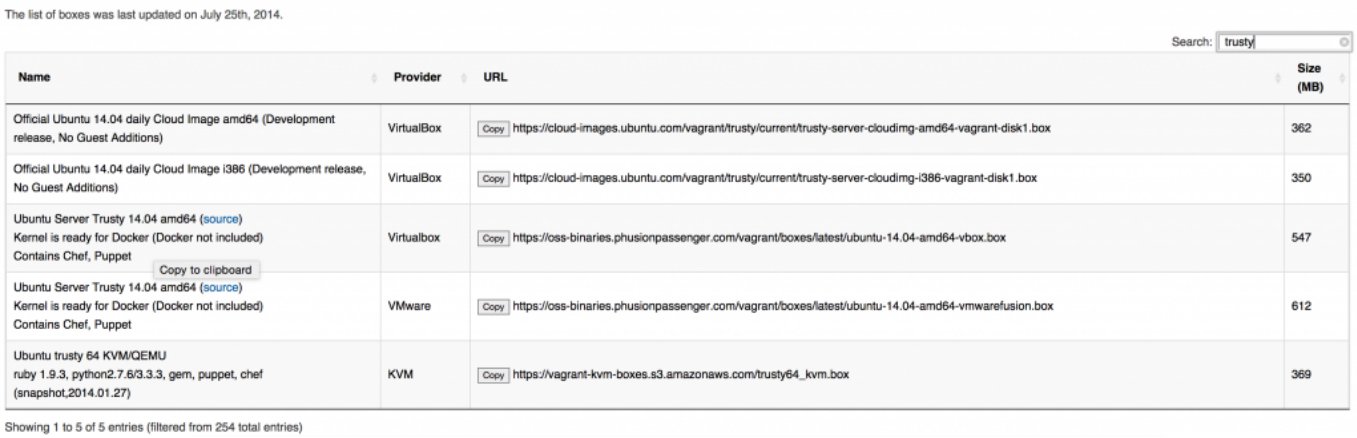
You will see on screen that there has been a 'Vagrantfile' created. This is the configuration for your Vagrant deployment which we will be building to set up our first Ubuntu server.

We are going to edit that file shortly, but first let's jump into what the Vagrant Box list is and how we are going to choose which server to deploy from it.

VagrantBox.es

To figure out which Vagrant box to use, we can scan through the online catalog of Vagrant boxes, neatly found at VagrantBox.es which holds a current list of available images. There are lots of images there to choose from, but in our case we want to narrow the search to show only ones that are the Ubuntu 14.04 Trusty server.

Using the search field above the table, type in Trusty and you will see a more narrowed down list for us:



The list of boxes was last updated on July 25th, 2014.

Search: trusty

Name	Provider	URL	Size (MB)
Official Ubuntu 14.04 daily Cloud Image amd64 (Development release, No Guest Additions)	VirtualBox	Copy https://cloud-images.ubuntu.com/vagrant/trusty/current/trusty-server-cloudimg-amd64-vagrant-disk1.box	362
Official Ubuntu 14.04 daily Cloud Image i386 (Development release, No Guest Additions)	VirtualBox	Copy https://cloud-images.ubuntu.com/vagrant/trusty/current/trusty-server-cloudimg-i386-vagrant-disk1.box	350
Ubuntu Server Trusty 14.04 amd64 (source) Kernel is ready for Docker (Docker not included) Contains Chef, Puppet	Virtualbox	Copy https://oss-binaries.phusionpassenger.com/vagrant/boxes/latest/ubuntu-14.04-amd64-vbox.box	547
Ubuntu Server Trusty 14.04 amd64 (source) Kernel is ready for Docker (Docker not included) Contains Chef, Puppet	VMware	Copy https://oss-binaries.phusionpassenger.com/vagrant/boxes/latest/ubuntu-14.04-amd64-vmwarefusion.box	612
Ubuntu trusty 64 KVM/QEMU ruby 1.9.3, python2.7.6/3.3.3, gem, puppet, chef (snapshot,2014.01.27)	KVM	Copy https://vagrant-kvm-boxes.s3.amazonaws.com/trusty64_kvm.box	369

Copy to clipboard

Showing 1 to 5 of 5 entries (filtered from 254 total entries)

You can see that there are a few fields there to show what the configuration for each box is. These are the Name, Provider, URL, and Size in MB. For our Trusty images, we have a list of 5 entries to choose from which include two cloud images for x64 and x86 editions, a full Virtual Box image which includes Chef and Puppet already (Yay!), and also a VMware version of the same file, and finally a KVM edition which is a traditional Ubuntu server build.

We choose, or rather I choose, to use the 14.04 LTS edition. LTS stands for Long Term Support, and these tend to be very stable and popular images for building servers.

For our configuration, we will use the 14.04 amd64 edition for VirtualBox. Make sure you keep the URL handy `https://oss-binaries.phusionpassenger.com/vagrant/boxes/latest/ubuntu-14.04-amd64-vbox.box`, as we will be needing it to add to our Vagrantfile next.

Editing the Vagrantfile

For our experiment, we want a completely empty Vagrantfile to start off so that we have a simple version to build on. Copy your current Vagrantfile to another file and rename it to Vagrantfile.old and then open up your Vagrantfile with the editor of your choice (I use Sublime), and empty out the contents of the file.

Now we have a basic Vagrant file to create which will look like this:

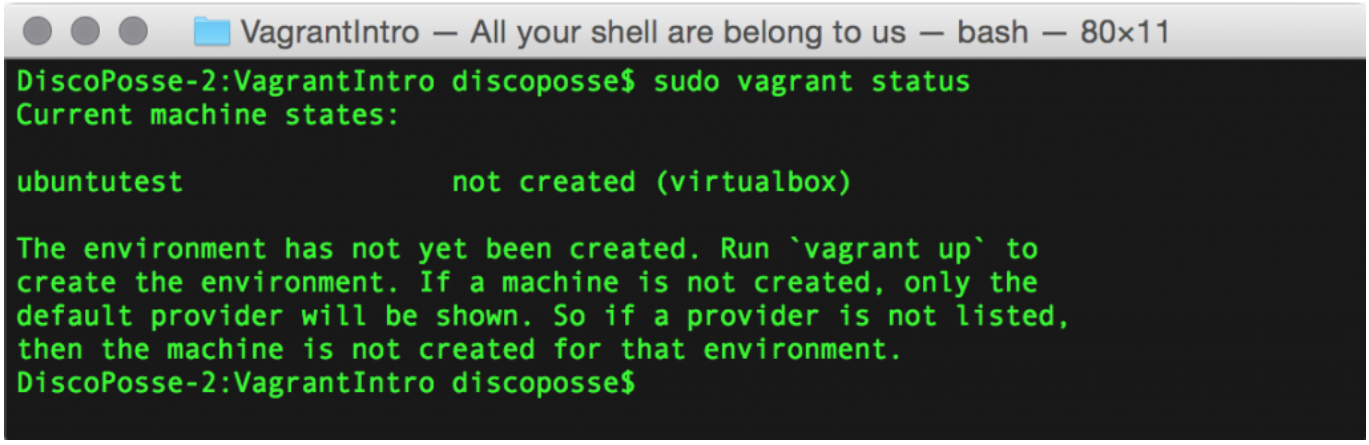
```
# -*- mode: ruby -*-
# vi: set ft=ruby :

VAGRANTFILE_API_VERSION = "2"
ENV['VAGRANT_DEFAULT_PROVIDER'] = 'virtualbox'

Vagrant.configure(VAGRANTFILE_API_VERSION) do |config|
config.vm.define "ubuntutest" do |ubuntutest|
ubuntutest.vm.hostname = "ubuntutest"
ubuntutest.vm.box = "trusty-server"
ubuntutest.vm.box_url =
"https://oss-binaries.phusionpassenger.com/vagrant/boxes/latest/ubuntu-14.04-amd64-vbox.box"
end
end
```

As we move through the series, we will deconstruct the Vagrantfile and really dive into how each

code block works. For now, this is meant to get our first machine up and running.

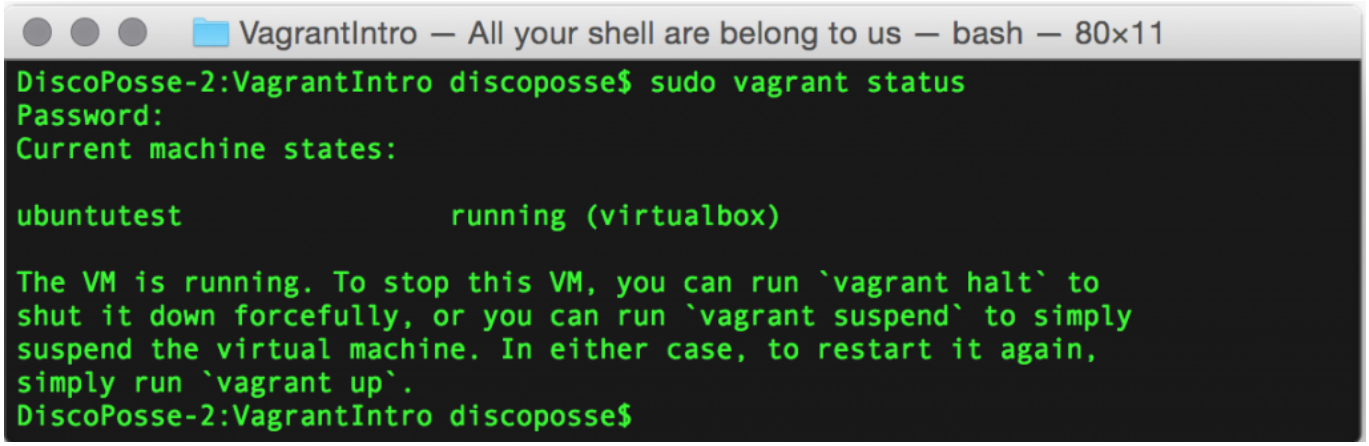
Vagrant Up!

Now that we have our configuration file ready, we can check the status to be sure that it is ready to go. This is done with the `vagrant status` command. Because I'm on a Mac and I don't use root privileges, you can see from my screen shot that it will show me using `sudo vagrant status` which may be different than what you so.

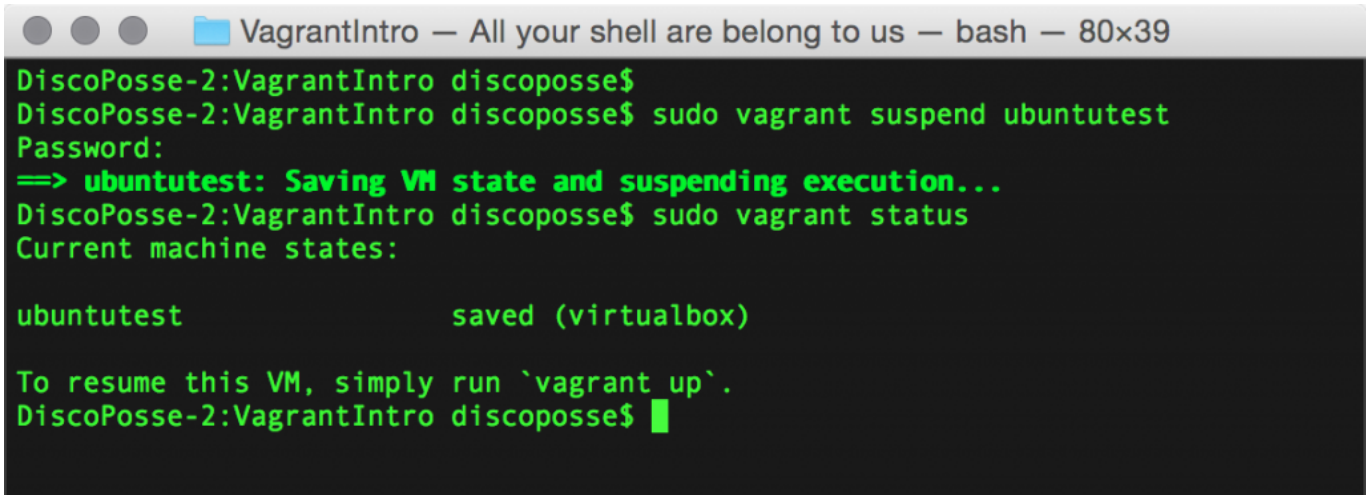


Next we run my favorite command to kick everything off which is quite simply `vagrant up` which will run through our configuration file and create a machine named **ubuntutest** based on the Ubuntu 14.04 Trusty image. Yup, it's just that easy:

We can check on our machine status by running the `vagrant status` again, and this time you will se a little bit of magic has happened because we now have a running machine!



Now we need to clean up after ourselves because this is just the first step and we may want to take a break between things. We are going to suspend our virtual guest because it doesn't need to be running all of the time. This is easily done with the `vagrant suspend ubuntutest` command, after which we will run another `vagrant status` command to confirm that the machine is in a saved state.



There we go! Now pat yourself on the back, take a break and enjoy because we are going to keep on rolling ahead with this Vagrant configuration and grow it much larger in the series.