|  |
| --- |
| Some Notes on Installation |
| Installation Notes |

|  |
| --- |
|  |

# Introduction

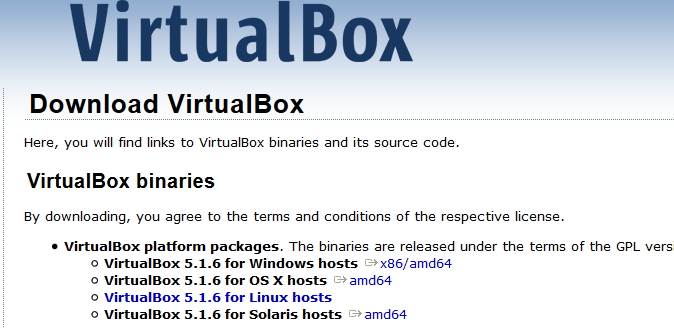
The course requires 4 virtual box instance. These are as follows:

1. Box 1 contains the DNS Master, Jenkins and Puppet Master to be installed
2. Box 2 contains docker from where 2 containers are spawned off for performing acceptance and web tests.
3. Box 3 contains a puppet agent which ensures tomcat 7 and production version of application is installed.
4. Box 4 contains docker which spawns off Nagios and which ensures that Box 3 is being monitored.

The rest of the document will cover the basics of installation of the required software. The instructions are specific to Windows 10 and Ubuntu 14.04 (both 64 bit). Please adjust instructions accordingly.

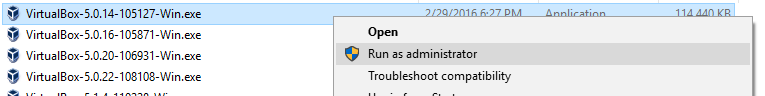
# Install Virtual Box

**Step 1**: Download VirtualBox from- <https://www.virtualbox.org/wiki/Downloads>

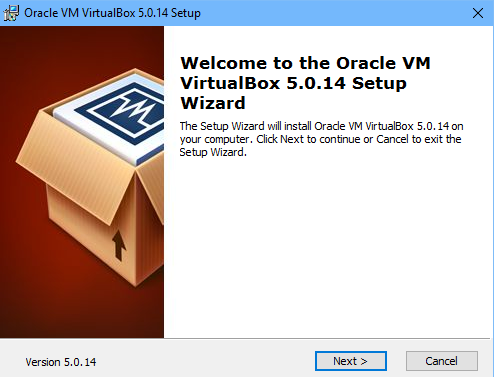


**Step 2**: Run the setup

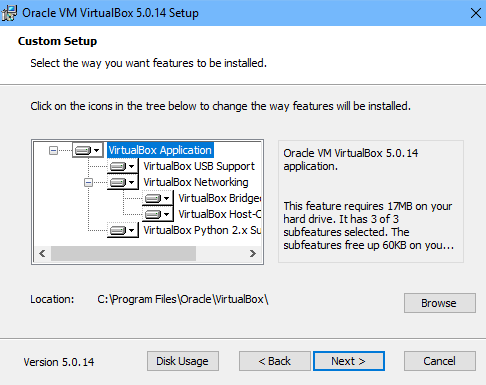
Run the downloaded file as an administrator. Keep default settings. There is no need to change any settings.



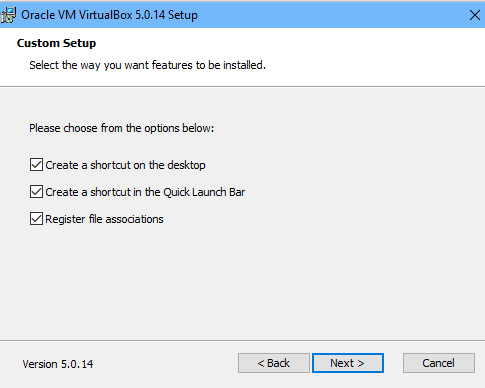
**Step 3**: Click “Next”. ©



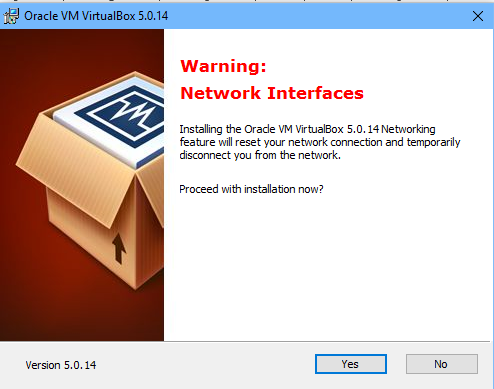
**Step 4**: Select the way you want your features to be installed and click “Next”. Keep the default settings – unless you want to change them.



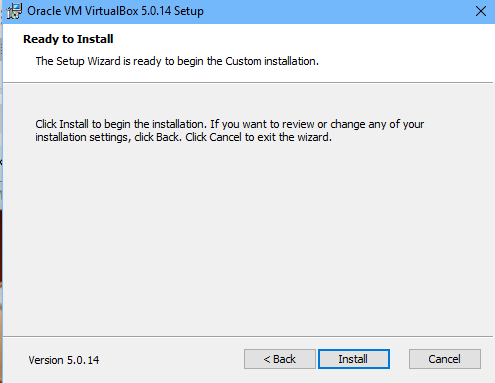
**Step 5**: Chose all the options and click “Next”.



**Step 6**: Click “Yes” to install Virtual box.



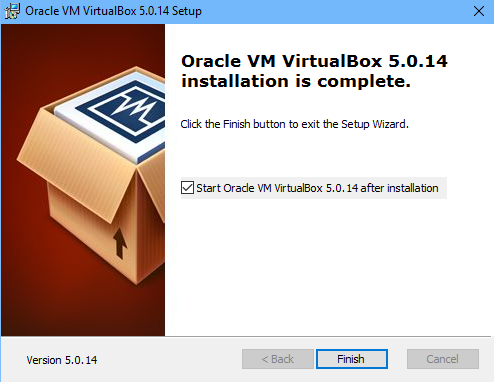
**Step 7**: Click “Install” to begin the installation.

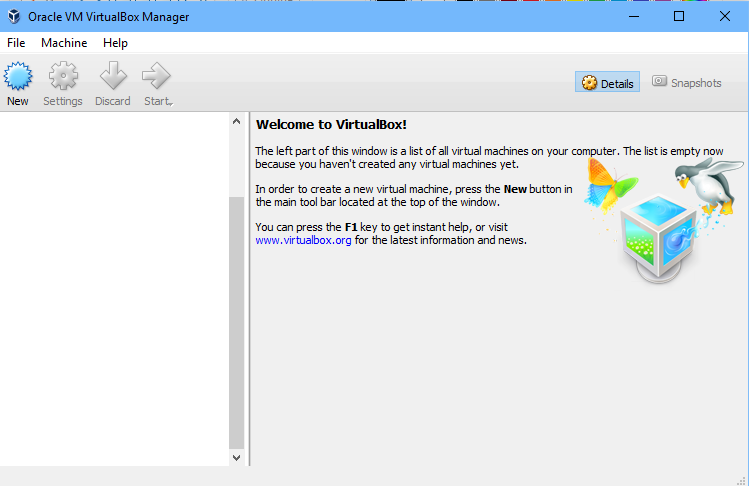


**Step 8**: Click “Install” on windows security popup. (You may or may not get this message Also note that on Window 10, the message will come from UAC, not Windows Security).



**Step 9**: Select “Start Oracle VM VirtualBox 5.0.14 after installation and click “Finish”.

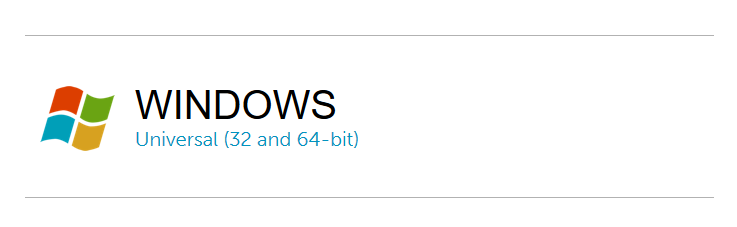




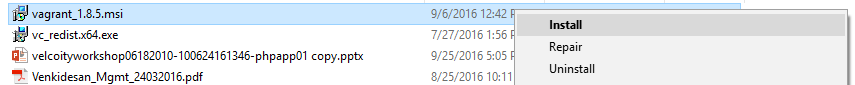
Congratulations! Oracle Virtual Box Manager is successfully installed.

# Installing Vagrant

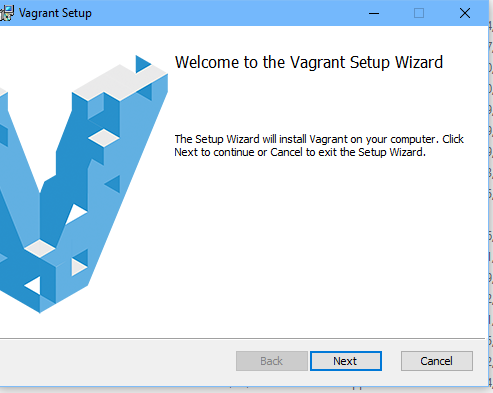
**Step 1**: Download Vagrant from- <https://www.vagrantup.com/downloads.html>



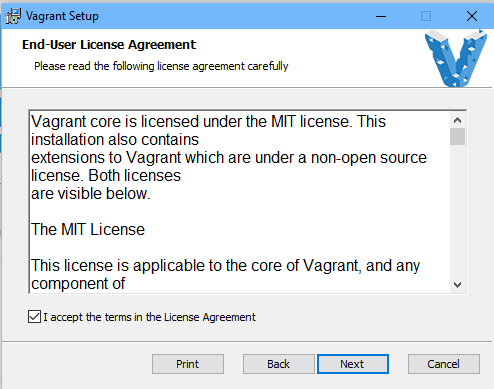
**Step 2**: Double click on installer or select and Chose install



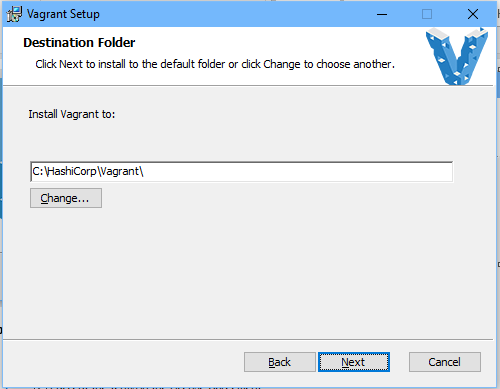
**Step 3**: Click on Next.



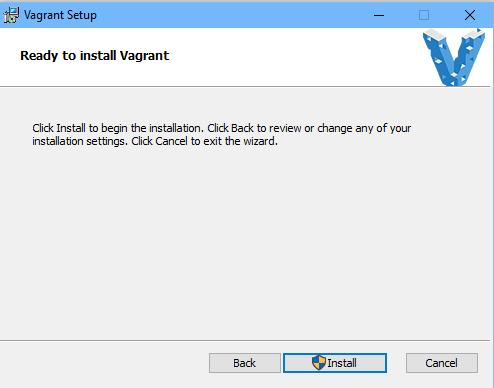
**Step 4**: Accept the license text and click on Next.

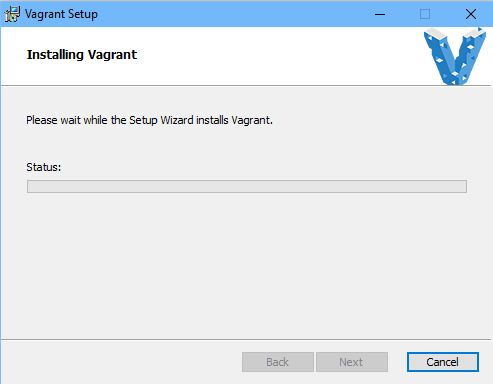


**Step 5: (optional)** – Change installation folder and click on Next. We recommend the defaults be kept as is.



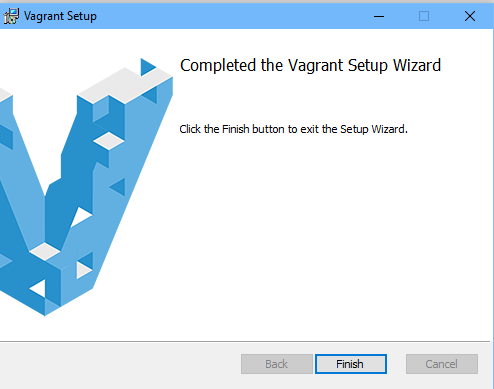
**Step 6:** – Click on install button.



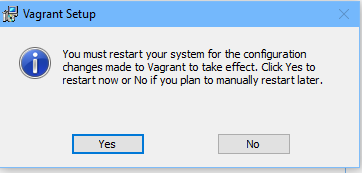


**Step 7:** – Accept to run software from Hashicorp.

**Step 8**: Click on finish.



**Step 9:** Restart your machine. Congratulations! Vagrant is now setup on your machine.

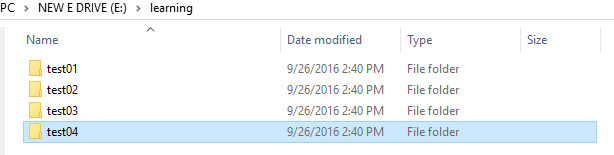


# Setup of the instances

The details of the 4 instances are as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Machine | IP | OS | SSH Port | Purpose |
| test01.demo.org | 192.168.33.20 | Ubuntu 14.04 | 8222 | DNS Server, Puppet Master |
| test02.demo.org | 192.168.33.21 | Ubuntu 14.04 | 9222 | Jenkins Slaves |
| test03.demo.org | 192.168.33.22 | Ubuntu 14.04 | 10222 | Production Tomcat Server |
| test04.demo.org | 192.168.33.23 | Ubuntu 14.04 | 11222 | Nagios Server |

Initially you will create a folder structure similar to the following:



The names are arbitrary, but we do recommend that instance names have a pattern which can be scripted. A sample script to create this structure is given below (**Adapt as required**)

@echo off

mkdir e:\learning

for /l %%c in (1,1,4) do mkdir "e:\learning\test0%%c"

Now for each directory, you need to create a vagrant file inside. This creates a sample vagrant file inside each of these folders. A sample code is given below: (**Adapt as required**)

@echo off

for /l %%c in (1,1,4) do (

echo "\*\*\* init in directory e:\learning\test0%%c"

cd "e:\learning\test0%%c"

vagrant init ubuntu/trusty64

)

cd e:\

Now for each folder, you need to edit the vagrant file to reflect correct

1. IP Address
2. SSH Port
3. Hostname

A sample vagrant file is attached for your reference.



After making the changes, you can start all machines by either manually going to each folder and typing the following at command prompt:

|  |
| --- |
| Vagrant up  @rem to stop a machine: use vagrant halt  **@rem the first time vagrant up is called, it has to download the full image so it will take time.** |

A sample script to start all of these machines is given below: (**Adapt as required**)

@echo off

for /l %%c in (1,1,4) do (

echo "\*\*\* init in directory e:\learning\test0%%c"

cd "e:\learning\test0%%c"

vagrant up

)

cd e:\

A sample script to stop all of these machines is given below: (**Adapt as required**)

@echo off

for /l %%c in (1,1,4) do (

echo "\*\*\* init in directory e:\learning\test0%%c"

cd "e:\learning\test0%%c"

vagrant halt

)

cd e:\

There is still one final step. You need to be able to SSH to these machines. To do so, create a folder called

E:\learning\keys. Then from each folder, navigate to folder: **.vagrant\machines\default\virtualbox**

Eg. *E:\learning\test01\.vagrant\machines\default\virtualbox*

Copy the file private\_key to e:\learning\keys giving it a unique name.

Eg. *copy e:\learning\test01\.vagrant\machines\default\virtualbox\private\_key e:\learning\keys*

Once this has been done, you can use any SSH client to connect to the instance as follows:

*Ssh -i e:\learning\keys\pk\_test01 -p 8222* [*vagrant@127.0.0.1*](mailto:vagrant@127.0.0.1)

A sample script to copy keys for all of these machines is given below: (**Adapt as required**)

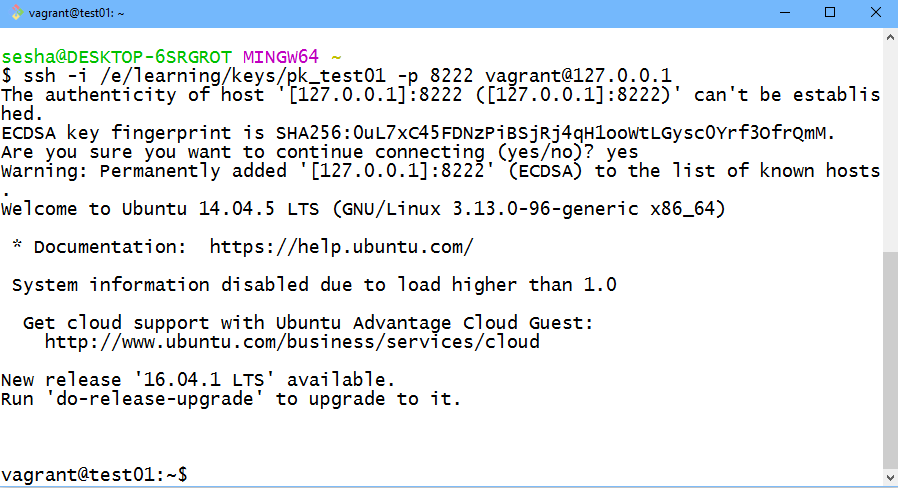
@echo off

for /l %%c in (1,1,4) do (

echo "\*\*\* init in directory e:\learning\test0%%c"

copy e:\learning\test0%%c*\.vagrant\machines\default\virtualbox\private\_key e:\learning\keys\pk\_test0%%c*

)



# Install BIND/DNS on Ubuntu 14 (BOX#1)

To install Bind9 on Ubuntu 14, do the following steps: (You do need to be logged in via SSH to BOX #1 as explained in previous section)

|  |
| --- |
| * Perform an update of repository   *sudo apt-get update*   * Install BIND software   *sudo apt-get install bind9 bind9utils bind9-doc dnsutils -y*   * Edit /etc/default/bind9 and add/modify line that contains OPTIONS to include support for IPV4   E,g. OPTIONS="-4 -u bind"   * Edit /etc/bind/named.conf.options and set any global level configurations. A sample named.conf.options is attached herwith      * Restart bind, Bind is UNIX implementation of DNS.   *sudo /etc/init.d/bind9 start*   * Verify core configuration using dig   dig @127.0.0.1 ubuntu.com   * Edit /etc/bind/named.conf.local. In our case, we are going to add zones for demo.org. So add the following lines to this file.   *zone "demo.org" {*  *type master;*  *file "/etc/bind/zones/db.demo.org";*  *#allow-transfer { 192.168.33.84; }; # ns2 secondary Name Server*  *};*  *zone "168.192.in-addr.arpa" {*  *type master;*  *file "/etc/bind/zones/db.192.168"; # 192.168.0.0/16 subnet*  *#allow-transfer { 192.168.33.84; }; # ns2 private IP address - secondary*  *};*   * Make the directory /etc/bind/zones and create empty files db.demo.org and db.192.168 under them. Sample content is attached below:   *sudo mkdir /etc/bind/zones*  *sudo touch /etc/bind/zones/db.demo.org*  *sudo touch /etc/bind/zones/db.192.168*     * Check Configuration.   *sudo named-checkconf*  *sudo named-checkzone sridemo.com db.sridemo.com*  *sudo named-checkzone 168.192.in-addr.arpa /etc/bind/zones/db.192.168*   * if all fine, restart   *sudo service bind9 restart*   * verify command   *dig demo.org @127.0.0.1*  *dig -tAXFR demo.org*  **Optionally: verify /etc/resolv.conf has your domain name and name server listed in it. This needs to be verified on each client.**  **If all is fine, a dig command should show output similar as below:** |

# Install Puppet Master on Ubuntu 14 (BOX#1)

To install puppet master on Ubuntu 14, do the following steps: (You do need to be logged in via SSH to BOX #1 as explained previously.

|  |
| --- |
| * Get the latest package   *wget https://apt.puppetlabs.com/puppetlabs-release-trusty.deb*   * Install the same   *sudo dpkg -i puppetlabs-release-trusty.deb*  *sudo apt-get update*  *sudo apt-get install -y puppetmaster*   * **Edir /etc/puppet/puppet.conf** edit below 2 lines   [main]  dns\_alt\_names = puppet,test01,test01.sridemo.com  #Also, remove the line templatedir=$confdir/templates, which has been deprecated.   * Restart Server   *sudo service puppetmaster start* |

# Install JDK, Ant, Maven, GIT and Jenkins on Ubuntu 14 (BOX#1)

To install components required for on Ubuntu 14, do the following steps: (You do need to be logged in via SSH to BOX #1 as explained previously.

|  |
| --- |
| * Execute the following commands   *sudo apt-get install -y default-jdk ant maven git*  **Configure git with following commands:**  *git config --global user.name EdurekaDemo*  *git config --global* [*user.email demouser@edureka.com*](mailto:user.email%20demouser@edureka.com)  *git config --global push.default simple*  **Configure git with following commands:**   * Verify ant with   *ant -version*   * Verify maven with   *mvn –version*  **Installing Jenkins**  *wget -q -O - https://pkg.jenkins.io/debian/jenkins-ci.org.key | sudo apt-key add -*  *sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'*  *sudo apt-get update*  *sudo apt-get install -y jenkins*  Alternatively, install Tomcat 7 and copy Jenkins.war file to webapps folder of Tomcat (**THIS METHOD IS RECOMMENDED)** |

# Install Nagios Server on Ubuntu 14 (BOX#4)

To install Nagios server components required for on Ubuntu 14, do the following steps: (You do need to be logged in via ssh

|  |
| --- |
| * Add the Nagios user and group nagcmd   *sudo useradd nagios*  *sudo groupadd nagcmd*  *sudo usermod -a -G nagcmd nagios*   * Install Apache and required modules   *sudo apt-get update*  *sudo apt-get install -y apache2 build-essential libgd2-xpm-dev openssl libssl-dev xinetd apache2-utils unzip*  *sudo apt-get install -y mysql-server php5 php5-mysql php5-gd libapache2-mod-php5*   * Download and build Nagios 4.2   *cd ~*  *curl -L -O https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.2.1.tar.gz*  *tar xvf nagios-\*.tar.gz*  *cd nagios-\**  *./configure --with-nagios-group=nagios --with-command-group=nagcmd*  *make all*  *sudo make install*  *sudo make install-commandmode*  *sudo make install-init*  *sudo make install-config*  *sudo /usr/bin/install -c -m 644 sample-config/httpd.conf /etc/apache2/sites-available/nagios.conf*  *sudo usermod -G nagcmd www-data*   * Download and build Plugins   *cd ~*  *curl -L -O http://nagios-plugins.org/download/nagios-plugins-2.1.2.tar.gz*  *tar xvf nagios-plugins-\*.tar.gz*  *cd nagios-plugins-\**  *./configure --with-nagios-user=nagios --with-nagios-group=nagios --with-openssl*  *make*  *sudo make install*   * Download and build NRPE   *cd ~*  *curl -L -O http://downloads.sourceforge.net/project/nagios/nrpe-2.x/nrpe-2.15/nrpe-2.15.tar.gz*  *tar xvf nrpe-\*.tar.gz*  *cd nrpe-\**  *./configure --enable-command-args --with-nagios-user=nagios --with-nagios-group=nagios --with-ssl=/usr/bin/openssl --with-ssl-lib=/usr/lib/x86\_64-linux-gnu*  *make all*  *sudo make install*  *sudo make install-xinetd*  *sudo make install-daemon-config*   * Configure Xinetd and restart it.   Edit vi /etc/xinetd.d/nrpe and edit line containing only\_allow to allow your servers to connect to it.  Restart xinetd using: ***service xinetd restart***   * Configure Nagios   *sudo vi /usr/local/nagios/etc/nagios.cfg #enable servers folder*  *sudo mkdir /usr/local/nagios/etc/servers*  *sudo vi /usr/local/nagios/etc/objects/commands.cfg*  #Add below lines  *define command{*  *command\_name check\_nrpe*  *command\_line $USER1$/check\_nrpe -H $HOSTADDRESS$ -c $ARG1$*  *}*   * Confgure Apache   *sudo a2enmod rewrite*  *sudo a2enmod cgi*  *sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin*  *sudo ln -s /etc/apache2/sites-available/nagios.conf /etc/apache2/sites-enabled/*  *sudo service nagios start*  *sudo service apache2 restart*  *sudo ln -s /etc/init.d/nagios /etc/rcS.d/S99nagios* |

# Install Puppet Client on Ubuntu 14 (BOX#2 and BOX#3)

To install puppet master on Ubuntu 14, do the following steps: (You do need to be logged in via SSH to BOX #1 as explained previously.

|  |
| --- |
| * Get the latest package   *wget https://apt.puppetlabs.com/puppetlabs-release-trusty.deb*   * Install the same   *sudo dpkg -i puppetlabs-release-trusty.deb*  *sudo apt-get update*  *sudo apt-get install -y puppetclient*   * **Edir /etc/puppet/puppet.conf** edit below 2 lines   [agent]  server = test01.sridemo.com  #Also, remove the line templatedir=$confdir/templates, which has been deprecated.   * Restart client   *sudo service puppet restart*   * Generate a client Cert Request   puppet agent -t   * On Server, sign the certificate   *puppet cert sign <Fqdn> # replace FQDN with FQDN of client*  Repeat step with other box |

# Miscellaneous Installation

Both class and course project require Tomcat 8 and JDK8 for compiling and running the code. Since Ubuntu 14.04 does not recognize Tomcat 8 as a service and has no “official” OpenJDK Deb release, here are the steps to install the same. (ALWAYS install JDK before Tomcat)

|  |
| --- |
| **Install OpenJDK 8 on Ubuntu 14**  Add a repository, perform an apt-update and install JDK. The commands are as below:  *sudo add-apt-repository ppa:openjdk-r/ppa*  *sudo apt-get update*  *sudo apt-get install -y default-jdk*  *sudo apt-get install -y openjdk-8-jdk*  *sudo update-alternatives --config java # configure which version of the JRE you want*  *sudo update-alternatives --config javac # configure which version of JDK you want*  **Install Tomcat 8 on Ubuntu 14**  Perform an update and add an user called tomcat  *sudo apt-get update*  *sudo groupadd tomcat*  *sudo useradd -s /bin/false -g tomcat -d /opt/tomcat tomcat*  **Download Tomcat and Change permissions**  *wget http://mirror.sdunix.com/apache/tomcat/tomcat-8/v8.0.23/bin/apache-tomcat-8.0.23.tar.gz*  *sudo mkdir /opt/tomcat*  *sudo tar xvf apache-tomcat-8\*tar.gz -C /opt/tomcat --strip-components=1*  *cd /opt/tomcat*  *sudo chgrp -R tomcat conf*  *sudo chmod g+rwx conf*  *sudo chmod g+r conf/\**  *sudo chown -R tomcat work/ temp/ logs/*  **Create a file to start tomcat as a service**  *sudo vi /etc/init/tomcat.conf*  A sample file is listed below    **Restart Tomcat8**  *sudo initctl reload-configuration*  *sudo initctl start tomcat* |

# Install Dockers on Ubuntu 14 (BOX#2)

To install puppet master on Ubuntu 14, do the following steps: (You do need to be logged in via SSH to BOX #2 as explained previously.

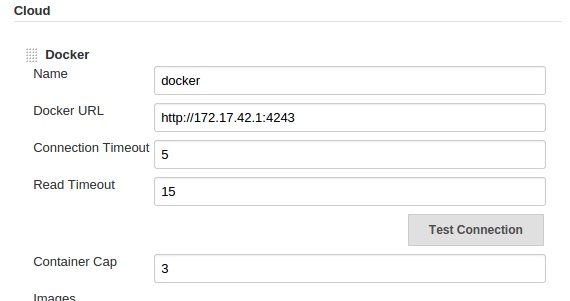
|  |
| --- |
| * Verify your kernel version   ***uname -r***  This has to be 3.13 and upwards.   * Update your Apt sources   ***sudo apt-get update && sudo apt-get install -y apt-transport-https ca-certificates***   * Add new GPG key   ***sudo apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 --recv-keys 58118E89F3A912897C070ADBF76221572C52609D***   * Edit /etc/apt/sources.list.d/docker.list (If it does not exist, add this) * Add this line:   ***deb https://apt.dockerproject.org/repo ubuntu-trusty main***   * Update repository   ***sudo apt-get update***   * Purge the old repo if it exists.   ***sudo apt-get purge lxc-docker***   * Verify it is pulling correct data   ***apt-cache policy docker-engine***   * Install linux extra packages   ***sudo apt-get install -y linux-image-extra-$(uname -r) linux-image-extra-virtual***   * install Docker   ***sudo apt-get install -y docker-engine***   * Start the daemon   ***sudo service docker start***   * verify It is correct   ***sudo docker run hello-world***   * Add user and group   ***sudo groupadd docker***  ***sudo usermod -aG docker $USER***   * Edit /etc/default/docker * Add line for ***DOCKER\_OPTS="--dns 8.8.8.8"***   Replace with your own DNS server   * Restart Daemon   ***sudo service docker restart***   * **to Remove docker**   ***sudo apt-get purge docker-engine***  ***sudo apt-get autoremove --purge docker-engine*** |

# Setup Dockers for Jenkins

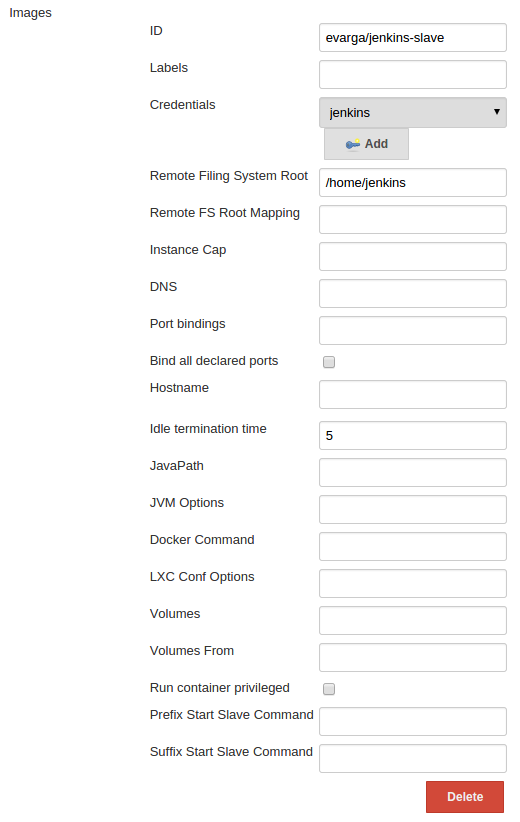
Connect to the box where dockers has been installed. Do the following steps:

|  |
| --- |
| ***docker pull Ubuntu***  ***docker run -i -t ubuntu /bin/bash***  ***apt-get update && apt-get install-y openssh-server***  ***mkdir /var/run/sshd***  ***apt-get install openjdk-8-jdk***  ***adduser jenkins***  ***/usr/sbin/sshd***  ***exit***  Now in host, run  ***docker ps -a***  ***docker commit <id> Jenkins-0 # replace <id> with the id of the container id***  Edit /etc/default/docker and enable the docker daemon to run on TCP port 2375. Restart Docker. |

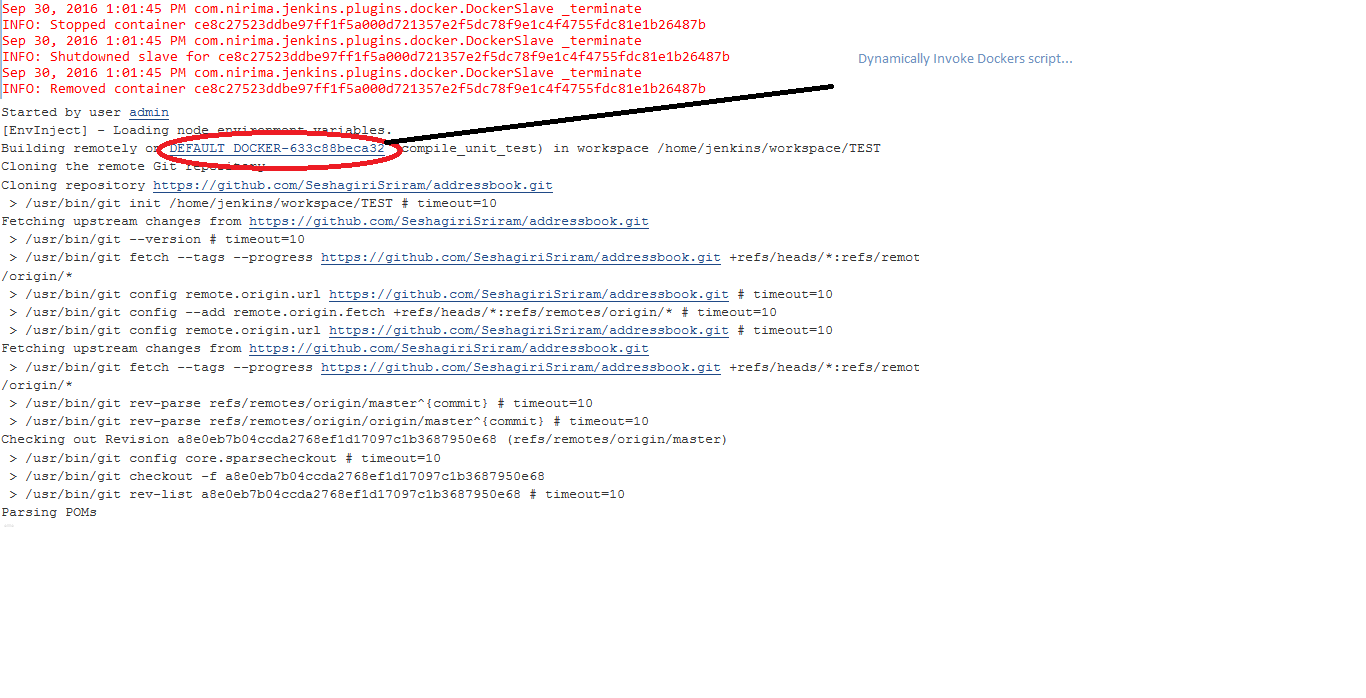
In Jenkins, navigate to Manage Jenkins->Configure System and add docker as the cloud provider.



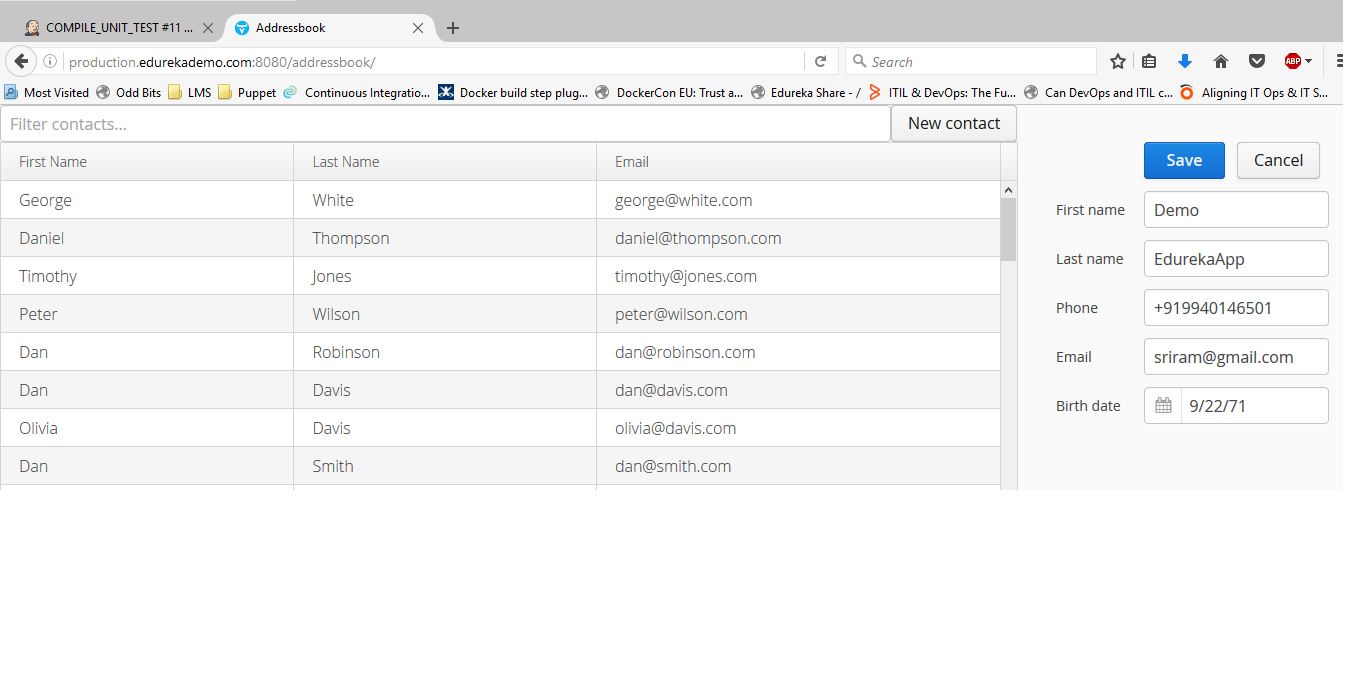
Click on Add button to add new images. Specify jenkin-0 (instead of evarga/Jenkins-slave), add volumes for maven/java, set a label for this image (e.g. compile\_unit\_test) and create a new maven job with this label.



That’s it! Run your job and you should see the output as below in Tomcat Console:



As part of the build pipeline, set up a job that copies the artifact from testing and deploys it to a Puppet server and runs puppet-agent on Production server using SSH (**HINT:** Make sure SSH connectivity exists between Jenkin server and Production Server).



<< END OF DOCUMENT >>