**Initial Lab Setup Document**

**Author: Avik Deb (avikdeb@gmail.com)**

**Version: 1.0**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Contents

[1. Host System Configuration 2](#_Toc462009318)

[1.1 System requirement 2](#_Toc462009319)

[1.2 Prepare system for virtualization 2](#_Toc462009320)

[1.3 List of software to be downloaded 3](#_Toc462009321)

[1.4 Some useful tips 4](#_Toc462009322)

[2. Oracle Virtual Box Installation and Configuration 5](#_Toc462009323)

[3. Vagrant Installation and Configuration 5](#_Toc462009324)

[4. Lab Exercises 8](#_Toc462009325)

# Host System Configuration

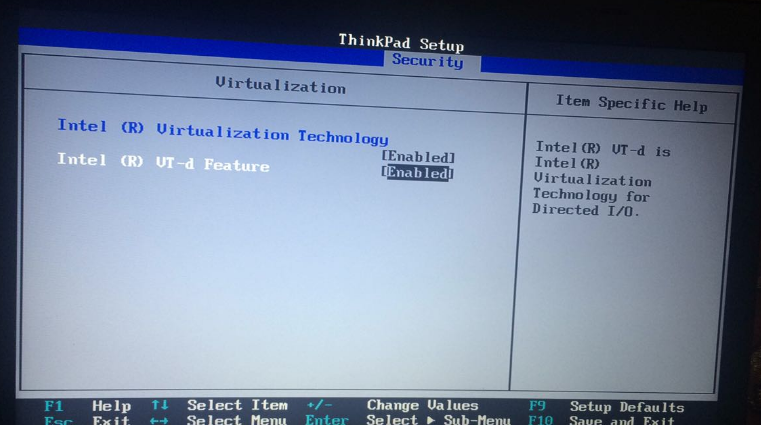
## System requirement

|  |  |
| --- | --- |
| **Developer Workstation (No. of participants) – Must have** | |
| Operating System | Windows 7 64-bit |
| RAM | 4GB (min) / 8GB (preferred) |
| Available Space | 5 GB min |
| Role | Administrator |
| Connectivity | Internet required |
| Virtualization | Enabled |
| **Lab Server (2) – Good to have** | |
| Operating System | Ubuntu Trusty 64-bit or equivalent |
| RAM | 4GB (min) / 8GB (preferred) |
| Available Space | 5 GB min |
| Role | Sudo or Root |
| Connectivity | Internet enabled |
| Virtualization | Not required |

Developers will connect to the lab linux servers if available or they will use virtualized linux hosted on their own workstation. Linux servers will host build, svn, deployment and monitoring environments. All systems must be available over LAN.

## Prepare system for virtualization

In host system BIOS set-up, virtualization must be enabled in order to run any 64-bit virtual OS within Virtual Box. Open BIOS just before booting with F2 or F12 or DEL key, depending on machine type (e.g. in Lenovo Thinkpad T450 it is F12, in DELL Inspiron 1532 it is F2)



## List of software to be downloaded

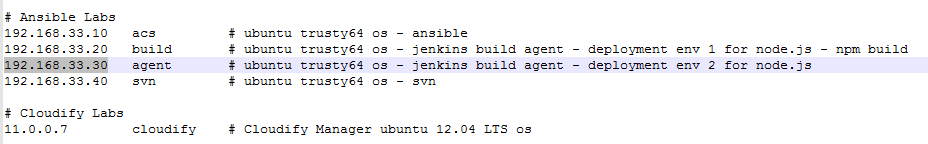
Following softwares are to be downloaded from the internet. Download location may be obtained by a simple Google search.

|  |  |
| --- | --- |
| **Software\*** | **Used For** |
| Oracle Virtual Box (latest) MSI | VM Container |
| Vagrant (latest) MSI | Provisioning VMs through script |
| Oracle Java 8 SDK 64 bit for Windows MSI | Development and runtimes for java based applications |
| Oracle Java 8 SDK 64 bit for Linux tar.gz |
| Git client (latest) for Windows | Source control and openSSH provider on Windows |
| PuTTY | Open terminal through SSH |
| Apache Tomcat 8 | Demo application deployment |
| Apache maven 3 | Demo java application build tool |
|  |  |

\*Software stack can also be downloaded from my cloud drive, please send me (avikdeb@gmail.com) a request mail if you need any.

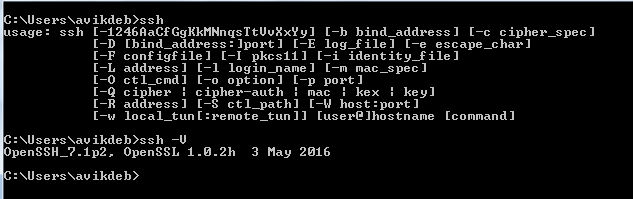
## Some useful tips

1. Install the software stack in the following sequence on the developer workstation:
2. Install Git client for Windows
3. Install Oracle Virtual Box
4. Install vagrant
5. Test VMs by spinning one VM defined in vagrantfile
6. Install JDK 8 on both Windows and Linux VMs
7. Install MySQL Community edition
8. Install Apache Tomcat 8
9. Install maven
10. Install Tortoise SVN client
11. TBD
12. Create a folder C:\vagrantlab where all your VMs, vagrant configuration files shall be kept
13. Define meaningful and intuitive hostname to your VMs and follow some IP naming convention in your etc/hosts file



1. Install Git client first. Keep the default choices as is.
2. Check in your path, if Git is located. Then check by running ssh in the command prompt.

|  |
| --- |
| C:\Program Files\Git\cmd;C:\Program Files\Git\mingw64\bin;C:\Program Files\Git\usr\bin; |



# Oracle Virtual Box Installation and Configuration

Download the latest version of the installer from VirtualBox site and follow the on-screen instructions during installation. Keep the defaults as when prompted.

This is a pre-requisite for Vagrant. For this lab we will use Vagrant to set-up our VMs. No extra configuration is required at this point, it will be done through vagrant scripts.

However, Virtual Box may be used independently to install any VM, provided the OS image (iso or vbox) is available.

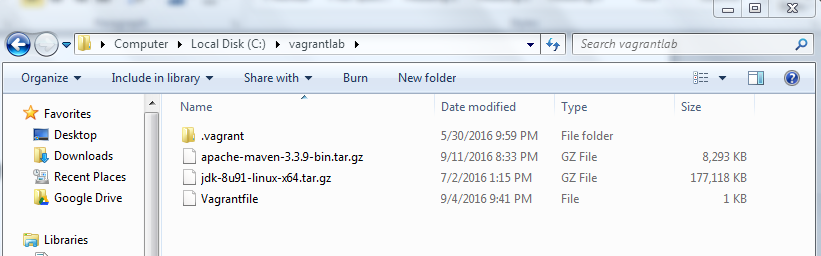
# Vagrant Installation and Configuration

Download vagrant installer from Vagrant site. Reboot your machine after installation, if required.

We will use Vagrant for provisioning our VMs in this lab. The VMs will be hosted inside virtual box but the properties etc. shall all be given through vagrant scripts / commands. This makes the server provisioning as a simple, automated process.

Follow the steps below to provision the Linux servers (Ubuntu Trusty 64-bit server without GUI):

1. Create a dedicated folder C:\vagrantlab where all your VMs config settings, vagrant files, shared folders (between host and gues VM) will be kept.
2. cd C:\vagrantlab and run command: vagrant init
3. This init command will do the initial setups automatically and creates the Vagrantfile in C:\vagrantlab. This vagrantfile holds all the server provisioning instructions. It is written in a DSL based on Ruby.



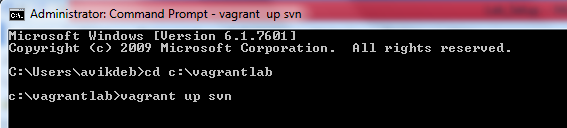
1. Open the generated Vagrant file using your favourite editor and replace the whole content with the one given in the box below:

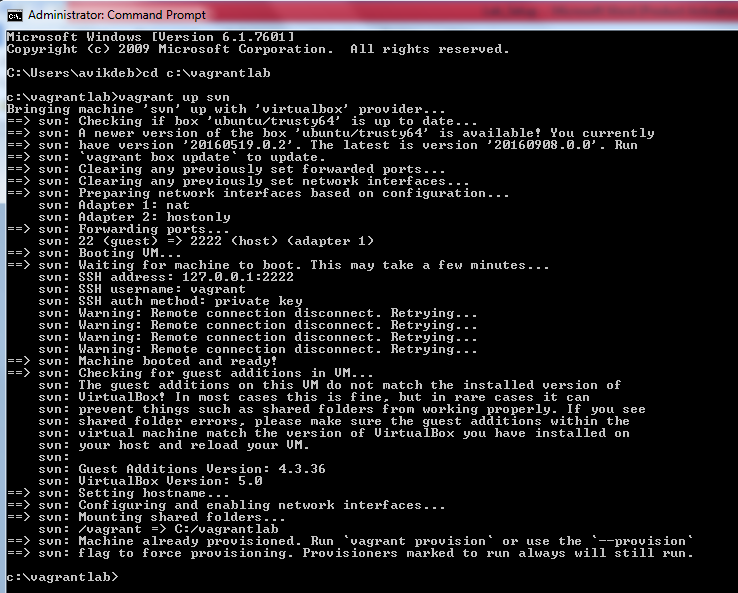
|  |
| --- |
| # -\*- mode: ruby -\*-  # vi: set ft=ruby :  # All Vagrant configuration is done below. The "2" in Vagrant.configure  # configures the configuration version (we support older styles for  # backwards compatibility). Please don't change it unless you know what  # you're doing.  Vagrant.configure(2) do |config|    config.vm.define "acs" do |acs|  acs.vm.box = "ubuntu/trusty64"  acs.vm.hostname = "acs"  acs.vm.network "private\_network", ip: "192.168.33.10"  end    config.vm.define "build" do |build|  build.vm.box = "ubuntu/trusty64"  build.vm.hostname = "build"  build.vm.network "private\_network", ip: "192.168.33.20"  end    config.vm.define "agent" do |agent|  agent.vm.box = "ubuntu/trusty64"  agent.vm.hostname = "agent"  agent.vm.network "private\_network", ip: "192.168.33.30"  end    config.vm.define "svn" do |svn|  svn.vm.box = "ubuntu/trusty64"  svn.vm.hostname = "svn"  svn.vm.network "private\_network", ip: "192.168.33.40"  end    end |

**Note:**

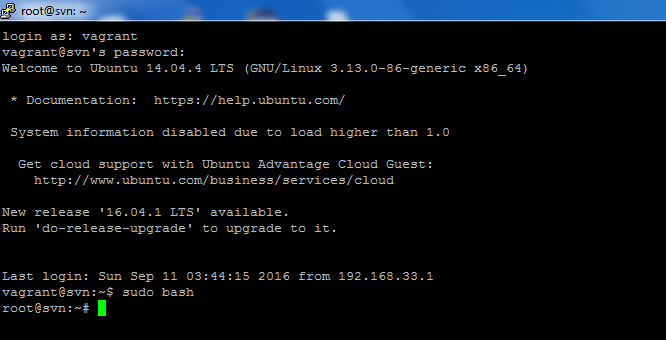
We have defined four VMs named acs, build, agent and svn – in config.vm.defined section within do – end loop. Hostname and network IPs are given. You can create as many VM as you with using this. Save the file.

1. Now open command prompt and go to C\vagrantlab folder or the folder where you have run the vagrant init command.
2. Issue command: vagrant up svn (or simple vagrant up svn. To spin up all VMs at ).This will bring up server named

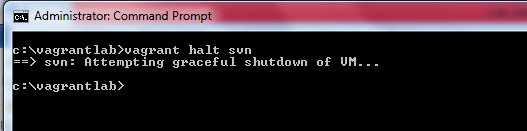




1. The VM has default username: vagrant and password: vagrant. Open PuTTY and run a new session for svn machine at IP: 192.168.33.40. Use vagrant/vagrant to log in. This user also has sudo



1. To shut-down any running VM, issue command: vagrant halt svn – this will bring down the server named svn. Only ‘vagrant halt’ command will shut-down all the VMs one after another



1. To remove any VM, issue command: vagrant destroy svn (or vagrant destroy to remove all VMs. Mere deletion will not erase the VM completely
2. You may create as many VMs as you want this way

# Lab Exercises

1. Write the Vagrantfile and set up few more servers
2. Setup a server with CenOS 7 as the operating system

(Hint: The list of images that vagrant can pull can be found in: <https://atlas.hashicorp.com/boxes/search>

1. How to remove all VM images from VirtualBox? Remember, vagrant destroy do not physically remove the image from VirtualBox, it just removes it from vagrant configuration.
2. How to install a VM behind firewall / in a controlled environment when there is no internet connection?