Ex.No:1 Date:	Install Virtualbox/VMware Workstation with different flavors of Linux or windows OS on top of windows7 or 8.

AIM:

To Install Virtualbox/VMware Workstation with different flavors of linux or windows OS on top of windows7 or 8.

PROCEDURE TO INSTALL:

Step 1- Download Link

Link for downloading the software is https://www.vmware.com/products/workstation-pro/workstation-pro-evaluation.html. Download the software for windows.

Step 2- Download the installer file

It should probably be in the download folder by default, if you have not changed the settings in your browser. File name should be something like VMware-workstation-full-15.5.1-15018445.exe. This file name can change depending on the version of the software currently available for download. But for now, till the next version is available, they will all be VMware Workstation 15 Pro.

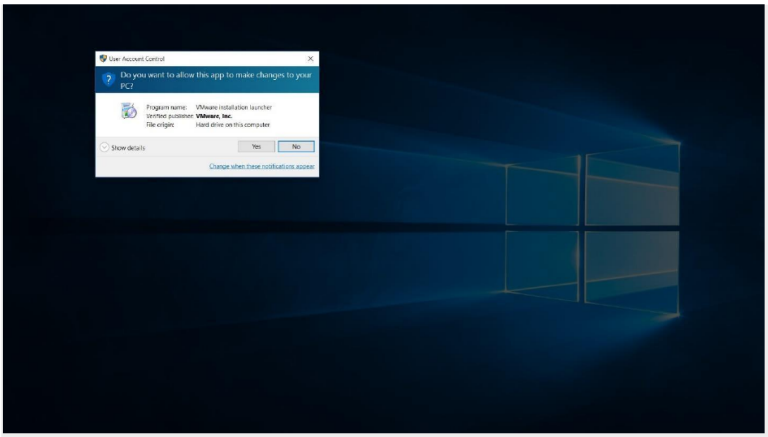
Step 3- Locate the downloaded installer file

For demonstration purpose, I have placed the downloaded installer on my desktop. Find the installer on your system and double click to launch the application.

VMware workstation 15 pro for windows 10 installer file screenshot.

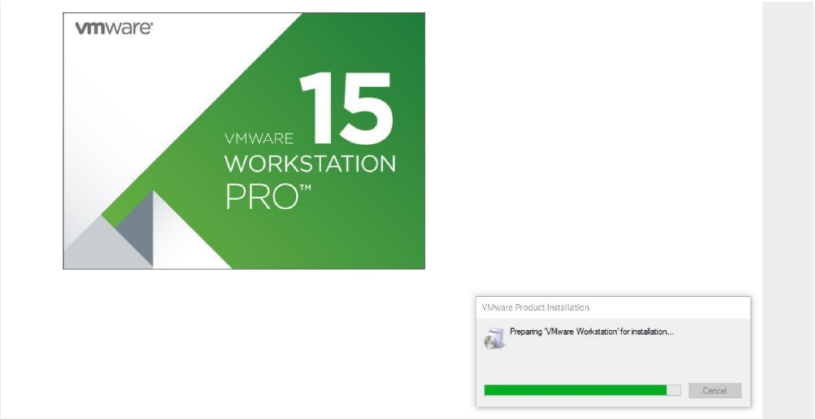
Step 4- User Access Control (UAC) Warning

Now you should see User Access Control (UAC) dialog box. Click yes to continue.



VMware Workstation 12 Pro installer windows 10 UAC screenshot

Initial Splash screen will appear. Wait for the process to complete.



VMware Workstation 15 Installation Splash Screen

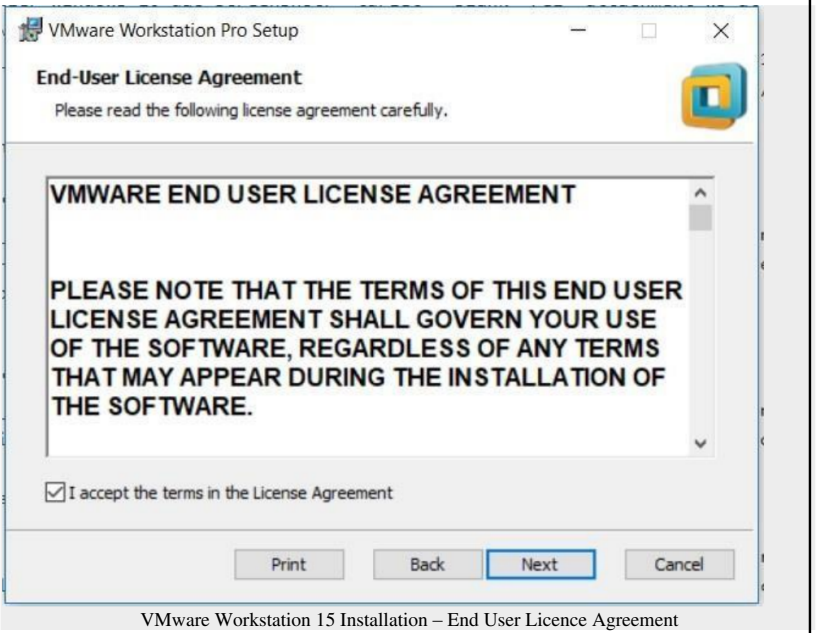
Step 5- VMware Workstation Setup wizard

Now you will see VMware Workstation setup wizard dialog box. Click next to continue.

VMware Workstation 15 Installation – Setup Wizard

Step 6- End User Licence Agreement

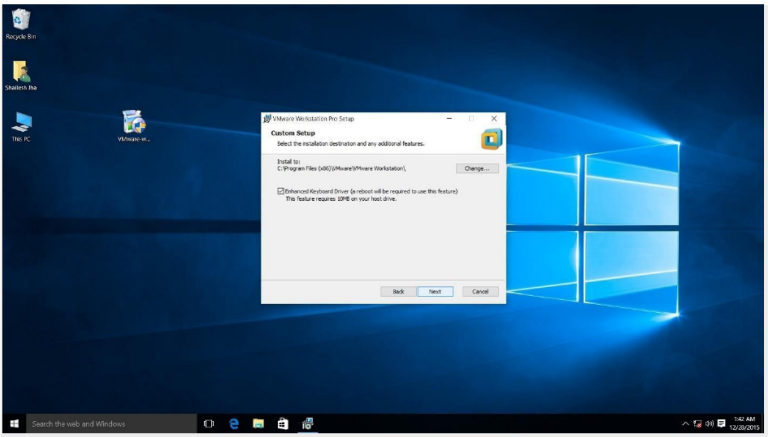
This time you should see End User Licence Agreement dialog box. Check “I accept the terms in the Licence Agreement” box and press next to continue.



VMware Workstation 15 Installation – End User Licence Agreement

Step 7- Custom Setup options

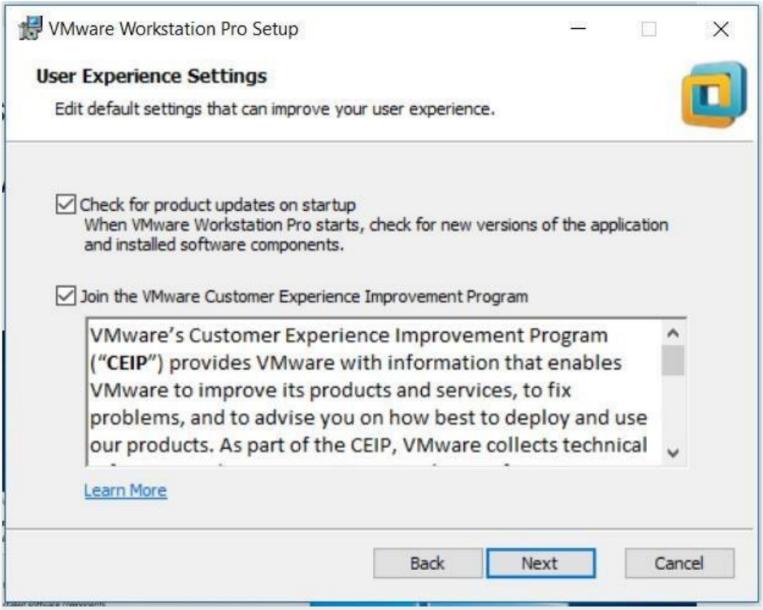
Select the folder in which you would like to install the application. There is no harm in leaving the defaults as it is. Also select Enhanced Keyboard Driver check box.



VMware Workstation 15 Pro installation – select installation folder

Step 8- User Experience Settings

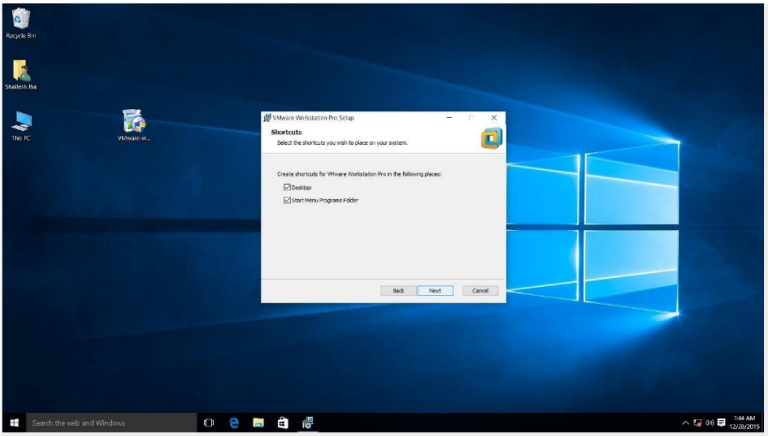
Next you are asked to select “Check for Updates” and “Help improve VMware Workstation Pro”. Do asyou wish. I normally leave it to defaults that are unchecked.



VMware Workstation 15 Installation – User Experience Settings

Step 9- Application Shortcuts preference

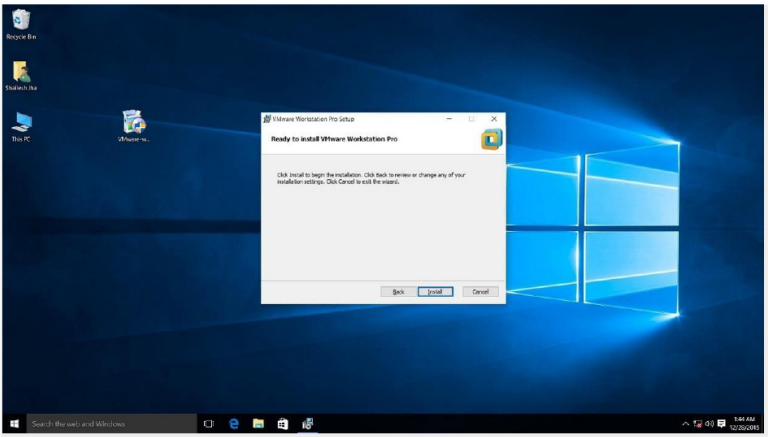
Next step is to select the place you want the shortcut icons to be placed on your system to launch the application. Please select both the options, desktop and start menu and click next.



VMware workstation 15 pro installation shortcut selection checkbox screenshot.

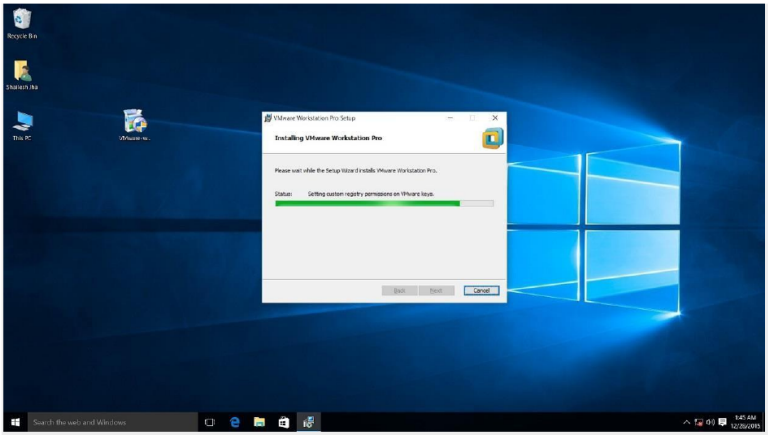
Step 10- Installation begins

Now you see the begin installation dialog box. Click install to start the installation process.



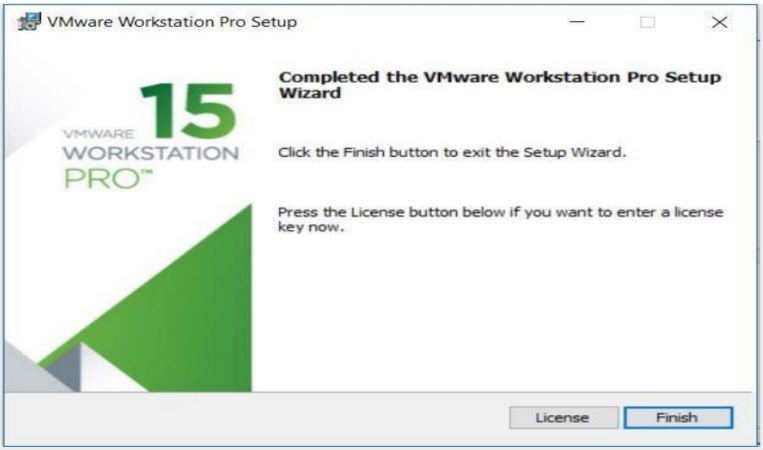
Screenshot for VMware Workstation 15 pro installation begin confirmation dialog box on windows 10.

Below screenshot shows Installation in progress. Wait for this to complete.



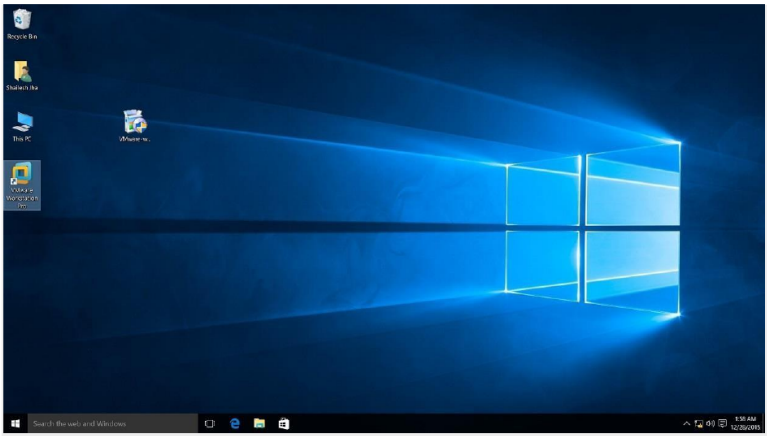
Screenshot for VMware Workstation 15 pro installation process.

At the end you will see installation complete dialog box. Click finish and you are done with the installation process. You may be asked to restart your computer. Click on Yes to restart.



Step 11- Launch VMware Workstation

After the installation completes, you should see VMware Workstation icon on the desktop. Double click on it to launch the application.



Screenshot for VMware Workstation 15 Pro icon on windows 10 desktop.

RESULT:

Thus,the Virtual box/ VMWare Workstation with different flavors of Linux or Windows OS on top of windows7 was installed successfully.

Ex.No:2	Install a C compiler in the virtual machine created using virtual box and execute Simple Programs
Date:	

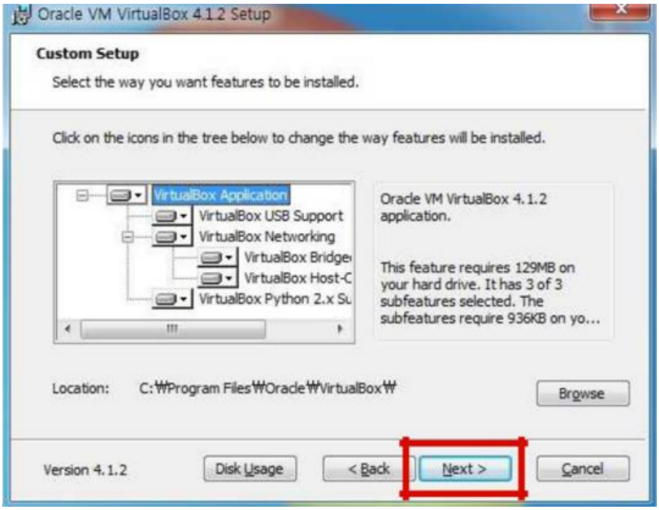
AIM:

To install virtual machine using c programming language and virtual box.

PROCEDURE:

InstallVirtualBox

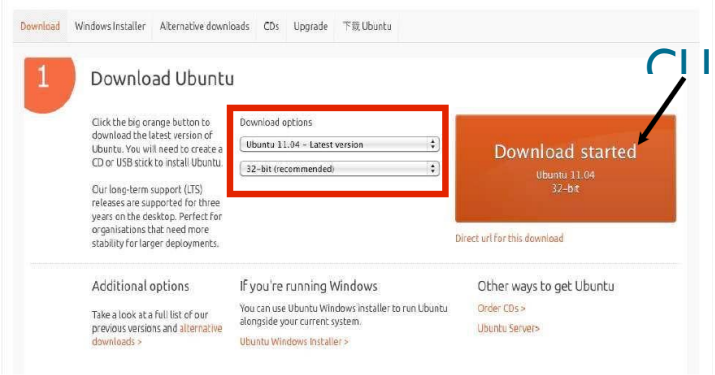
1. Visit http://www.virtualbox.org/wiki/downloads
2. Download VirtualBox platform packages for your OS
3. Open the Installation Package by double clicking



Download Linux

Step 1: Visit the pagehttp://www.ubuntu.com/download/ubuntu/download

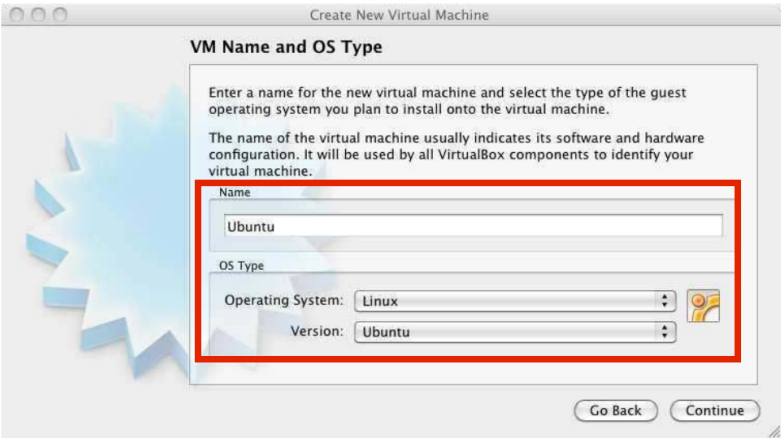
Step 2: Choose the Latest version of Ubuntu and32-bit and click “StartDownload”



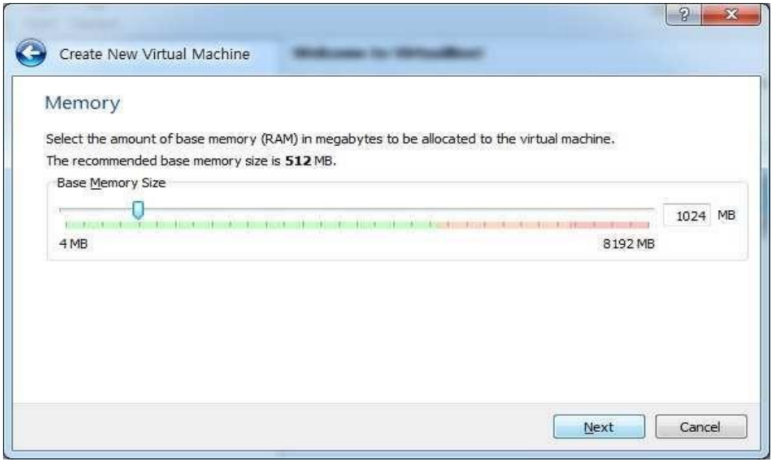
Step 3: Run VirtualBox by double-clicking theicon

Step4: Click“New” button on the top leftcorner

Step 5:
Click “Continue” on the pop-upwindowTypeVMname,select“Linux”fortheOSandchoose “Ubuntu” for theversion.



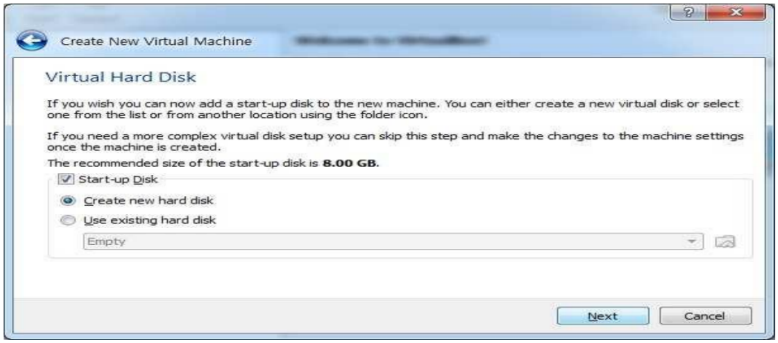
Step 6: Choose the amount of memory to allocate (I suggest choosing between 5I2 MB to I024MB)
Step 7:Click Continue orNe



Step 8: Choose create a new virtual harddisk

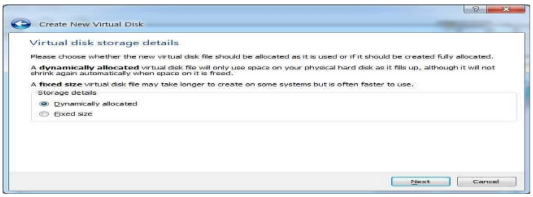
Step 9: Click Continue orNext

Choose VDI (VirtualBox DiskImage)



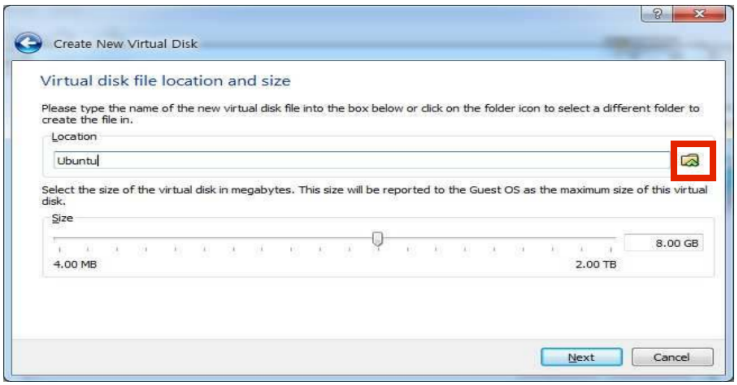
Choose “DynamicallyAllocated” click continue.

This way,the size of yourVirtual Hard Disk will grow as you use.

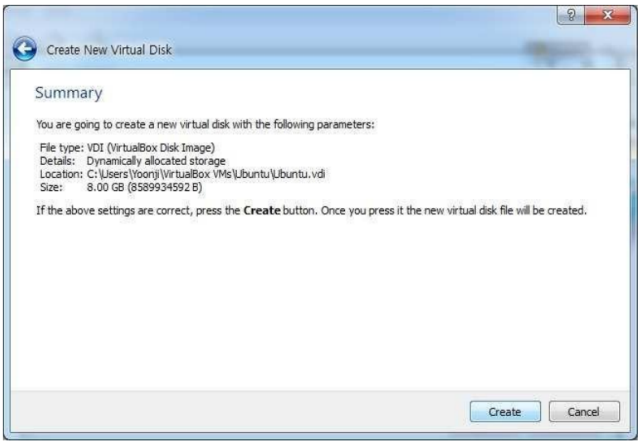


Click the folder icon and choose the ubuntu iso file you downloaded.

Select the size of the Virtual Disk (I recommend choosing 8 GB) and click continue



I4. ClickCreate



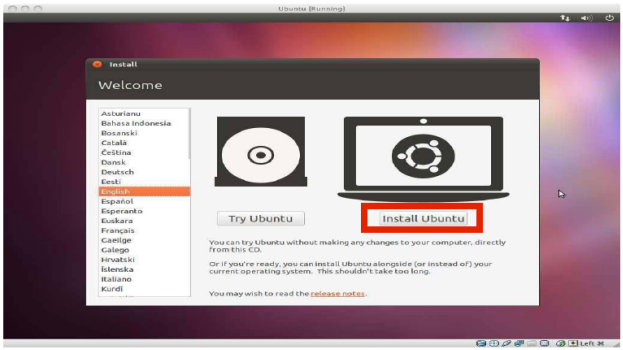
Running Linux

1.Choose Ubuntu from left column and click Star

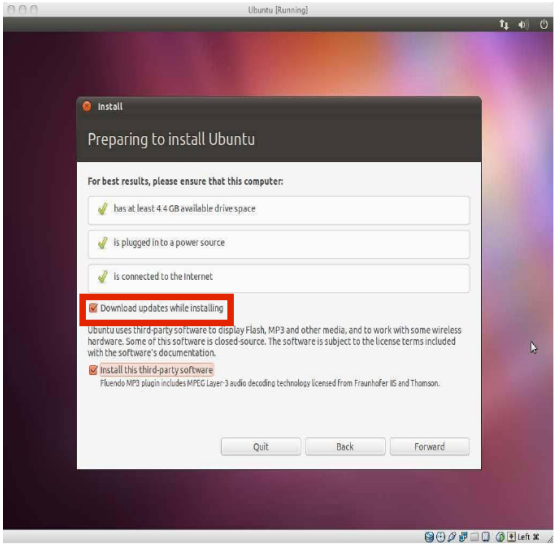
2.Click continue on pop-upwindow

3.Click the folder icon and choose the ubuntu iso file you downloaded and click continue and start

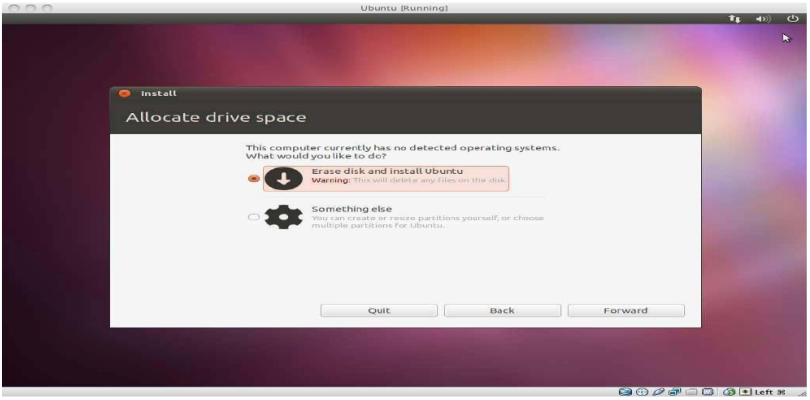
3.Click InstallUbunt



4.Check “Download updates” and clickForward

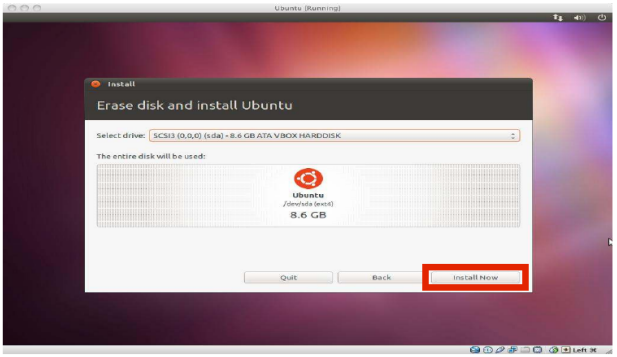


5.Choose “Erase disk and install Ubuntu” and click Forward (Don’t worry, it won’t wipe yourcomputer)



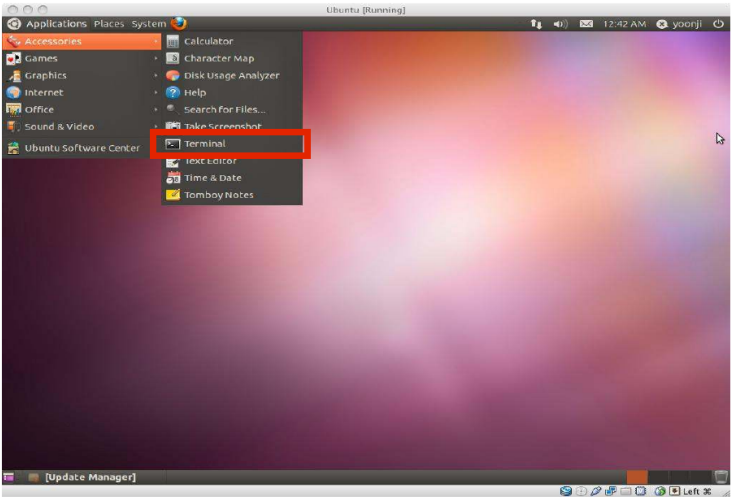
6.Click “Install Now” and wait. Maybe grab asnack.

7.When finished, click Restart and pressEnter.



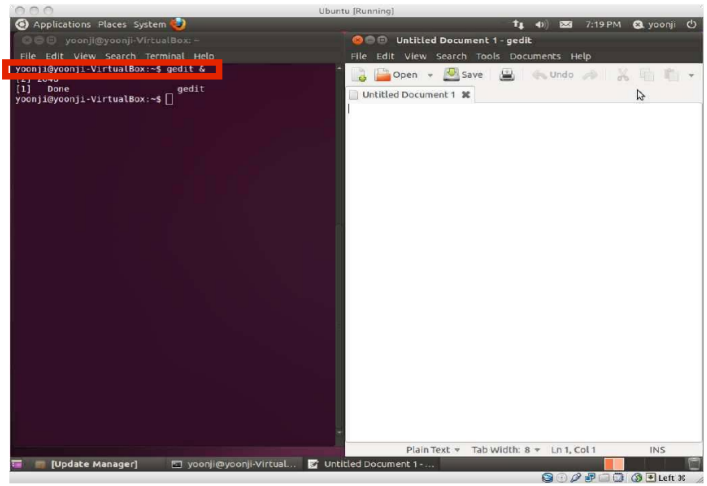
C Programming on Linux

I. OpenTerminal(Applications-Accessories-Terminal)



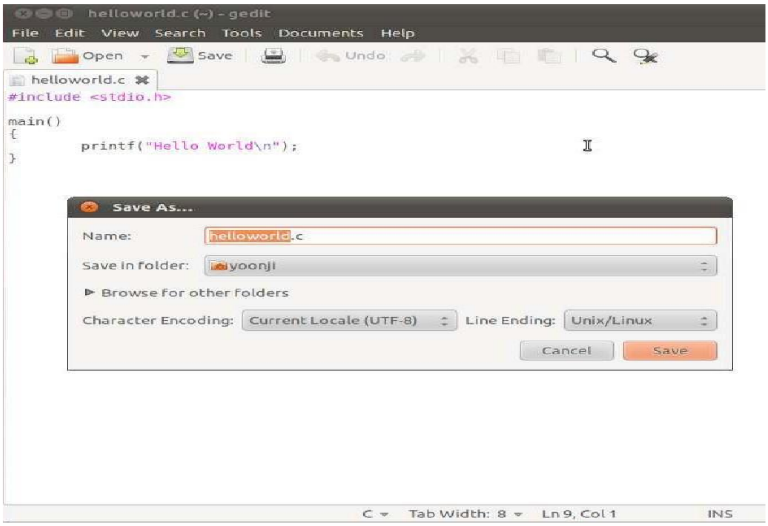
2.Open geditby typing “gedit&” onterminal

(You can also use any other Text Editor application)



3.Typethe following on gedit (or any other texteditor)

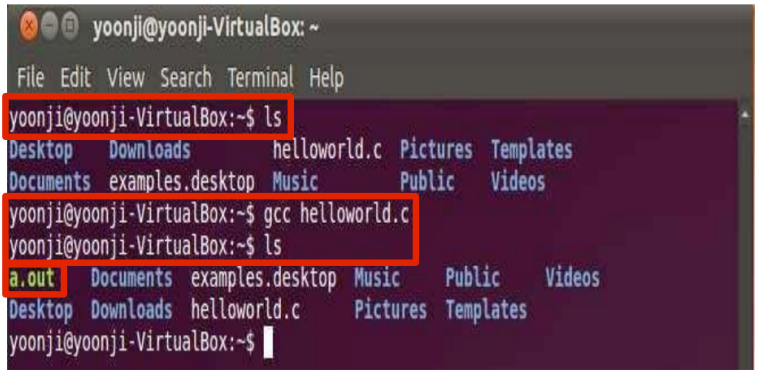
```
#include<stdio.h>main()
{
printf("Hello World\n");
}
```



5.Type“ls” on Terminal to see all files under currentfolder

6. Confirm that “helloworld.c” is in the currentdirectory. Ifnot,typecd DIRECTORY_PATHto go to the directory that has“helloworld.c”

7.Type“gcchelloworld.c” to compile, and type “ls” to confirm that a new executable file “a.out” iscreated

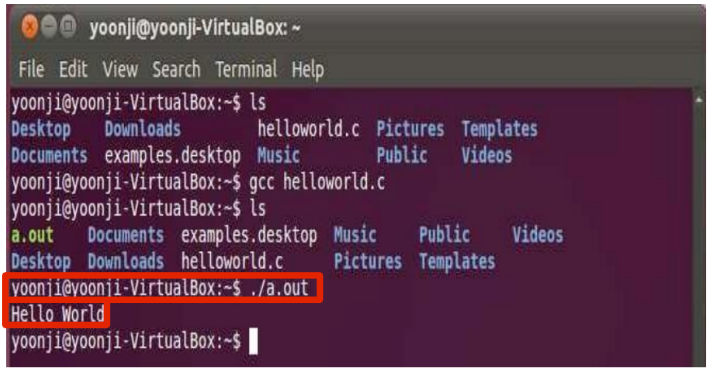


C Programming on Linux

8. Type "./a.out" on Terminal to run the program

9. If you see "HelloWorld" on the next line, you just successfully ran your first C program!

10. Try other codes from "A Shotgun Introduction to C" on professor Edwards's webpage.
You can also find many C programing guides online. (justgoogleit!)
Enjoy:)



RESULT:

Thus, the virtual machine using c programming language and virtual box was created and executed successfully.

Ex.No.3	Installing Google App Engine. Create hello world app and other simple web applications using python/java.
Date:	

AIM:

To install the google app, create hello world app and other simple web applications using python.

The App Engine SDK allows you to run Google App Engine Applications on your local computer. It simulates the run--tme environment of the Google App Engine infrastructure.

PROCEDURE:

Pre--Requisites: Python 2.5.4

If you don't already have Python 2.5.4 installed in your computer, download and Install Python 2.5.4 from:http://www.python.org/download/releases/2.5.4/

Download and Install

You can download the Google App Engine SDK by going to:

http://code.google.com/appengine/downloads.html

and download the appropriate installpackage.

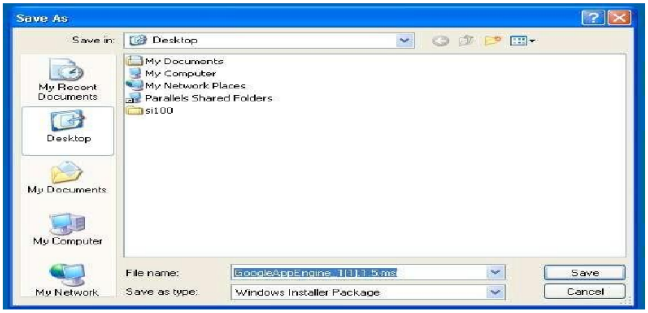
Download the Google App Engine SDK

Before downloading, please read the Terms that govern your use of the App Engine SDK.

Please note: The App Engine SDK is under **active development**, please keep this in mind as you explore its capabilities. See the SDK Release Notes for the information on the most recent changes to the App Engine SDK. If you discover any issues, please feel free to notify us via our Issue Tracker.

Platform	Version	Package	Size	SHA1 Checksum
Windows	1.1.5 - 10/03/08	GoogleAppEngine_1.1.5.msi	2.5 MB	e974312b4aefc0b3873ff0d93eb4c525d5e88c30
Mac OS X	1.1.5 - 10/03/08	GoogleAppEngineLauncher-1.1.5.dmg	3.6 MB	f62208ac01c1b3e39796e58100d5f1b2f052d3e7
Linux/Other Platforms	1.1.5 - 10/03/08	google_appengine_1.1.5.zip	2.6 MB	cbb9ce817bdabf1c4f181d9544864e55ee253de1

Download the Windows installer – the simplest thing is to download it to your Desktop or another folder that youremember.



Double Click on the **GoogleApplicationEngine**installer.

Click through the installation wizard, and it should install the App Engine. If you do not have Python 2.5,
it will install Python 2.5 as well.

Once the install is complete you can discard the downloaded installer

Making your FirstApplication

Now you need to create a simple application. We could use the "+" option to have the launcher make us an application – but instead we will do it by hand to get a better sense of what is goingon.

Make a folder for your Google App Engine applications. I am going to make the Folder on my Desktop called "**apps**" – the path to this folder is:

C:\Documents and Settings\csev\Desktop\apps

Andthenmakeasub- folderinwithin**apps**called"**ae-01-trivial**"–thepathtothis folder would be:

C:\ Documents and Settings \csev\Desktop\apps\ae--01--trivial

Using a text editor such as JEdit (www.jedit.org), create a file called **app.yaml**in the **ae--01--trivial** folder with the following contents:

application: ae-01-trivial version: 1
runtime: python api_version: 1
handlers:
- url: /.*
script: index.py

Note: Please do not copy and paste these lines into your text editor – you might end up with strange characters – simply type them into your editor.

Then create a file in the ae--01--trivial folder called index.py with three lines in it:

print 'Content-Type:text/plain' print ' '
print 'Hello there Chuck'

RESULT:

Thus, the installation of Google app Engine was installed and verified successfully.

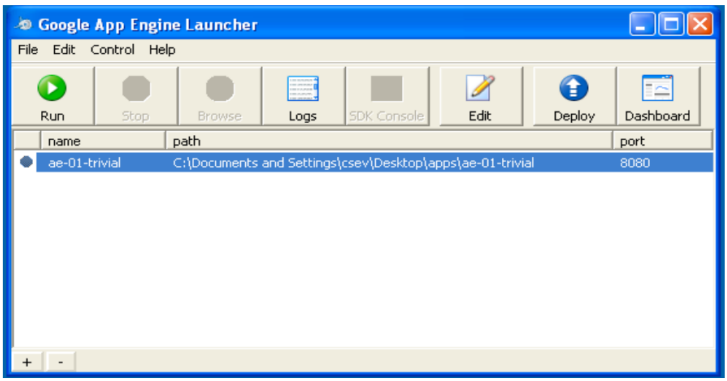
EX.NO: 4 Date:	Use GAE launcher to launch the web applications.

AIM:
To use GAE launcher to launch web application.

PROCEDURE:

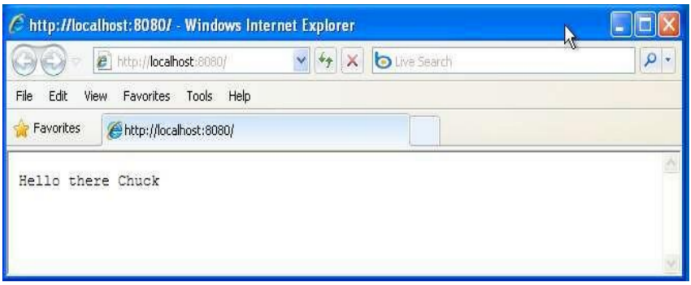
Start the **GoogleAppEngineLauncher**program that can be found under **Applications**.

Use the **File --> Add Existing Application** command and navigate into the **apps**directory and select the **ae--01--trivial** folder. Once you have added the application, select it so that you can control the application using the launcher.



Once you have selected your application and press **Run**. After a few moments your application will start and the launcher will show a little green icon next to your application. Then press **Browse** to open a browser pointing at your application which is running at **http://localhost:8080/**

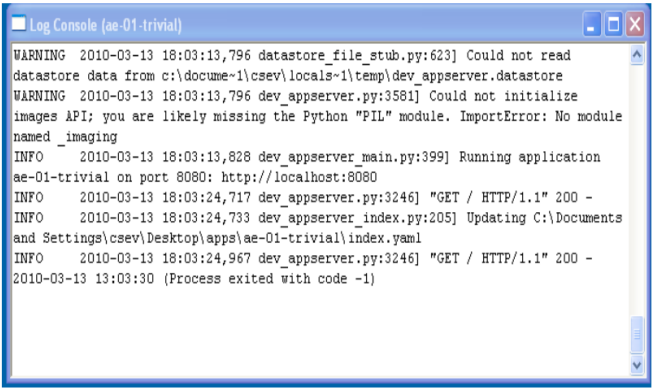
Paste **http://localhost:8080** into your browser and you should see your application as follows:



Just for fun, edit the **index.py** to change the name "Chuck" to your own name and press Refresh in the browser to verify your updates.

Watching the Log

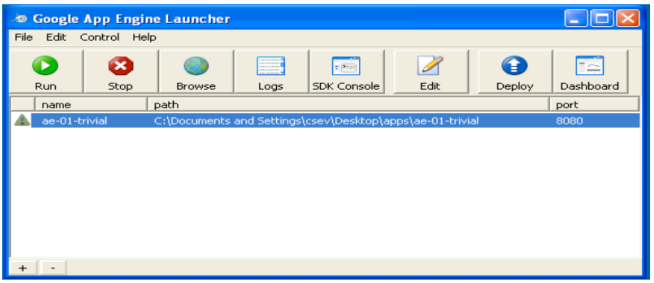
You can watch the internal log of the actions that the web server is performing when you are interacting with your application in the browser. Select your application in the Launcher and press the **Logs** button to bring up a log window:



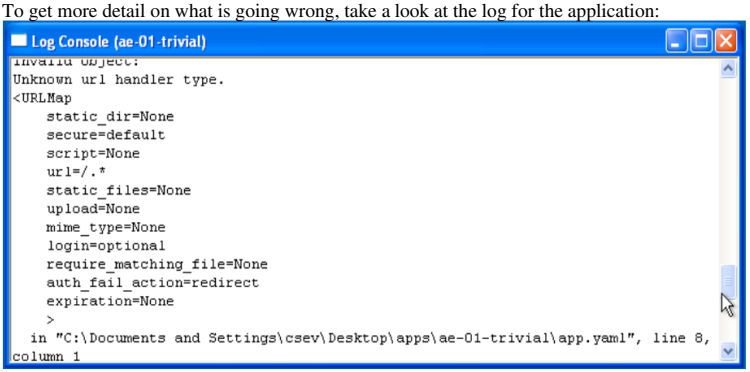
Each time you press **Refresh** in your browser – you can see it retrieving the output with a **GET** request.

Dealing With Errors

With two files to edit, there are two general categories of errors that you may encounter. If you make a mistake on the **app.yaml**file, the App Engine will not start and your launcher will show a yellow icon near your application:

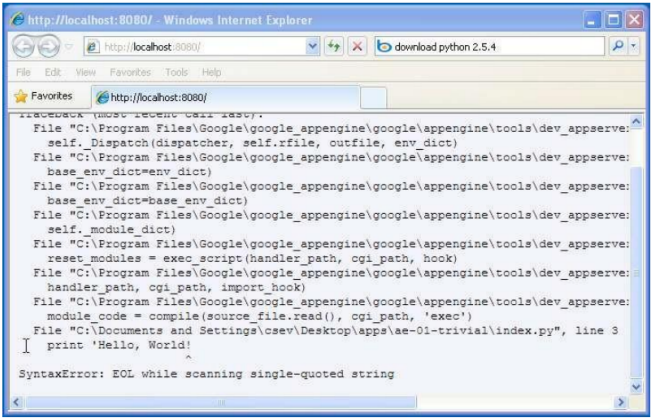


To get more detail on what is going wrong, take a look at the log for the application:



Inthisinstance–themistakeismis-indentingthelastlineinthe**app.yaml**(line8).

If you make a syntax error in the **index.py** file, a Python trace back error will appear in your browser.



The error you need to see is likely to be the last few lines of the output – in this case I made a Python syntax error on line one of our one- line application.

Reference: http://en.wikipedia.org/wiki/Stack_trace

When you make a mistake in the **app.yaml**file – you must the fix the mistake and attempt to start the application again.

If you make a mistake in a file like i**ndex.py**, you can simply fix the file and press refresh in your browser – there is no need to restart the server.

Shutting Down the Server

To shut down the server, use the Launcher, select your application and press the **Stop** button.

RESULT:

Thus, the installation of GAE launcher was launched and verified successfully.

Ex.No:5 Date:	**Simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present in CloudSim.**

AIM:

To simulate a cloud scenario using cloudsim and run a scheduling algorithm into it.

PROCEDURE:

CloudSim in Eclipse

CloudSim is written in Java. The knowledge you need to use CloudSim is basic Java programming and some basics about cloud computing. Knowledge of programming IDEs such as Eclipse or NetBeans is also helpful. It is a library and, hence, CloudSim does not have to be installed. Normally, you can unpack the downloaded package in any directory, add it to the Java classpath and it is ready to be used. Please verify whether Java is available on your system.

To use CloudSim in Eclipse:

1.Download CloudSiminstallablefiles from *https://code.google.com/p/cloudsim/downloads/list and unzip*

2.OpenEclipse

3.Create a new Java Project: File ->New

4.Import an unpacked CloudSim project into the new JavaProject

5.The first step is to initialise the CloudSim package by initialising the CloudSimlibrary,as follows:

```
CloudSim.init(num_user, calendar, trace_flag)
```

6.Data centres are the resource providers in CloudSim; hence, creation of data centres is a second step. To create Datacenter, you need the DatacenterCharacteristics object that stores the properties of a data centre such as architecture, OS, list of machines, allocation policy that covers the time or spaceshared, the time zone and itsprice:

```
Datacenter datacenter9883 = new Datacenter(name, characteristics, new VmAllocationPolicySimple
(hostList),s
```

7.The third step is to create abroker:

```
DatacenterBroker broker = createBroker();
```

8.The fourth step is to create one virtual machine unique ID of the VM, userId ID of theVM's owner, mips, number Of Pes amount of CPUs, amount of RAM, amount of bandwidth, amount of storage, virtual machine monitor, and cloudletScheduler policy forcloudlets:

```
Vmvm = new Vm(vmid, brokerId, mips, pesNumber, ram, bw, size, vmm, new CloudletSchedulerTimeShared())
```

9.Submit the VM list to thebroker:

```
broker.submitVmList(vmlist)
```

10.Create a cloudlet with length, file size, output size, and utilisationmodel:

Cloudlet cloudlet = new Cloudlet(id, length, pesNumber, fileSize, outputSize, utilizationModel,
11. Submit the cloudlet list to thebroker:
broker.submitCloudletList(cloudletList)

12.Start thesimulation:

CloudSim.startSimulation()
Sample Output from the Existing Example:

Starting CloudSimExample1... Initialising...

Starting CloudSim version 3.0 Datacenter_0 is starting...

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>null

Broker is starting... Entities started.

: Broker: Cloud Resource List received with 1resource(s) 0.0: Broker: Trying to Create VM #0 inDatacenter_0

: Broker: VM #0 has been created in Datacenter #2, Host#0

0.1: Broker: Sending cloudlet 0 to VM #0

400.1: Broker: Cloudlet 0 received

: Broker: All Cloudlets executed.Finishing... 400.1: Broker: Destroying VM#0

Broker is shutting down... Simulation: No more future events

CloudInformationService: Notify all CloudSim entities for shutting down. Datacenter_0 is shutting down...

Broker is shutting down... Simulationcompleted.

Simulation completed.

========== OUTPUT ==========

Cloudlet ID STATUS Data center ID VM ID Time Start Time Finish Time 0 SUCCESS 20

400 0.1 400.1

*****Datacenter:Datacenter_0***** Userid Debt

3 35.6

CloudSimExample1 finished!

RESULT:

Thus, the simulation of a cloud scenario using cloudsim and run a scheduling algorithm into it was created and verified successfully.

Ex.No:6	Transfer the files from one virtual machine to another virtual machine
Date :	

AIM:
To transfer a file from one VM to another VM by using a virtualbox.

PROCEDURE:

1.You can copy few (or more) lines with copy & paste mechanism.
2.For this you need to share clipboard between host OS and guest OS, installing Guest Addition on both the virtual machines (probably setting bidirectional and restarting them). You copy from guest OS in the clipboard that is shared with the host OS.
Then you paste from the host OS to the second guest OS.
3.You can enable drag and drop too with the same method (Click on the machine, settings, general, advanced, drag and drop: set to bidirectional)
4.You can have common Shared Folders on both virtual machines and use one of the directory shared as buffer to copy.
5.Installing Guest Additions you have the possibility to set Shared Folders too. As you put a file in a shared folder from host OS or from guest OS, is immediately visible to the other. (Keep in mind that can arise some problems for date/time of the files when there are different clock settings on thedifferent virtual machines).
6.If you use the same folder shared on more machines you can exchange files directly copying them in this folder.
7.You can use usual method to copy files between 2 different computer with client-server application. (e.g. scp with sshd active for linux, winscp... you can get some info about SSH servers e.g.here)
You need an active server (sshd) on the receiving machine and a client on the sending machine. Of course to have the authorization setted (via password or, better, via an automatic authentication method).
Note: many Linux/Ubuntu distribution install sshd by default: you can see if it is running with pgrepsshd from a shell. You can install with sudo apt-get install openssh-server.
8.You can mount part of the file system of a virtual machine via NFS or SSHFS on the other, or you can share file and directory with Samba.
You may find interesting the article Sharing files between guest and host without VirtualBox shared folders with detailed step by step instructions.
Each virtual machine has its own operative system running on and acts as a physical machine.
Each virtual machine is an instance of a program owned by an user in the hosting operative system and should undergo the restrictions of the user in the hostingOS.

RESULT:

Thus, the transferring of a file from one VM to another VM by using a VirtualBox was transferred and verified successfully.

EX.NO:7 Date:	**Install Hadoop single node cluster and run simple applications like wordcount.**

AIM:

To install Hadoop single node cluster and simple application like wordcount**.**

PROCEDURE:

Install Hadoop

Step 1: Click hereto download the Java 8 Package. Save this file in your home

directory.

Step 2: Extract the Java Tar File.
Command: tar -xvf jdk-8u101-linux-i586.tar.gz

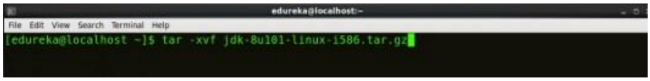


Fig: Hadoop Installation – Extracting Java Files

Step 3: Download the Hadoop 2.7.3 Package.

Command:wget,https://archive.apache.org/dist/hadoop/core/hadoop-2.7.3/hadoop- 2.7.3.tar.gz

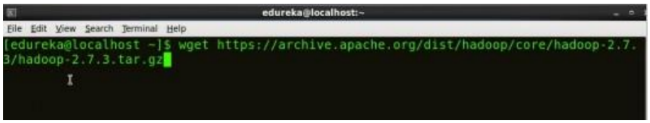


Fig: Hadoop Installation – Downloading Hadoop

Step 4: Extract the Hadoop tar File.

Command: tar -xvf hadoop-2.7.3.tar.gz

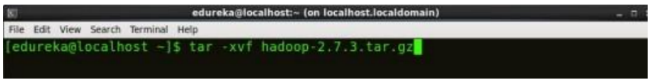


Fig: Hadoop Installation – Extracting Hadoop Files

Step 5: Add the Hadoop and Java paths in the bash file (.bashrc). Open. bashrc file.

Now, add Hadoop and Java Path as shown below.

Command: vi .bashrc

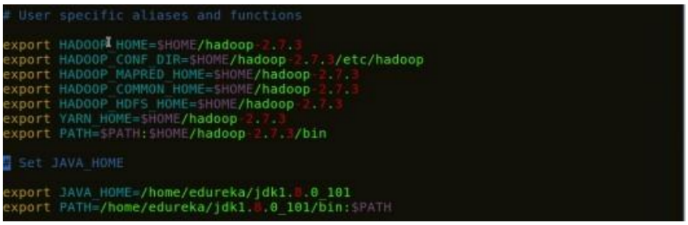


Fig: Hadoop Installation – Setting Environment Variable

Then, save the bash file and close it.

For applying all these changes to the current Terminal, execute the source command.

Command: source .bashrc

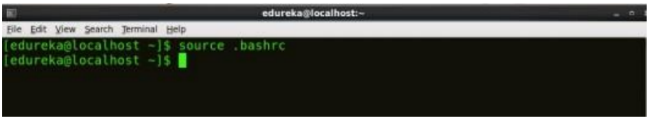


Fig: Hadoop Installation – Refreshing environment variables

To make sure that Java and Hadoop have been properly installed on your system and can be accessed through the Terminal, execute the java -version and hadoop version commands.

Command: java -version

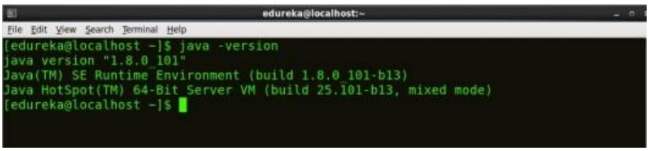


Fig: Hadoop Installation – Checking Java Version

Command: hadoop version

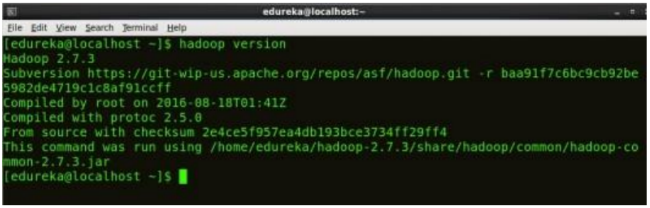


Fig: Hadoop Installation – Checking Hadoop Version

Step 6: Edit the Hadoop Configuration files.

Command: cd hadoop-2.7.3/etc/hadoop/

Command: ls

All the Hadoop configuration files are located in hadoop-2.7.3/etc/hadoop directory as you can see in the snapshot below:

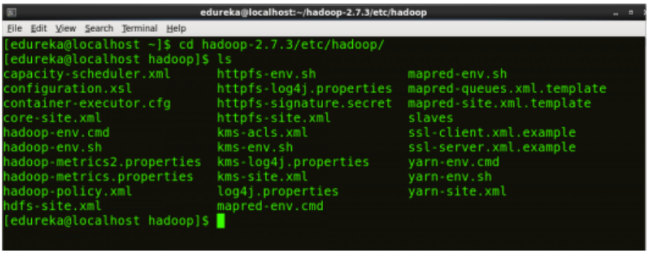


Fig: Hadoop Installation – Hadoop Configuration Files

Step 7: Open core-site.xml and edit the property mentioned below inside configuration tag:

core-site.xml informs Hadoop daemon where NameNode runs in the cluster. It contains configuration settings of Hadoop core such as I/O settings that are common to HDFS & MapReduce.

Command: vi core-site.xml

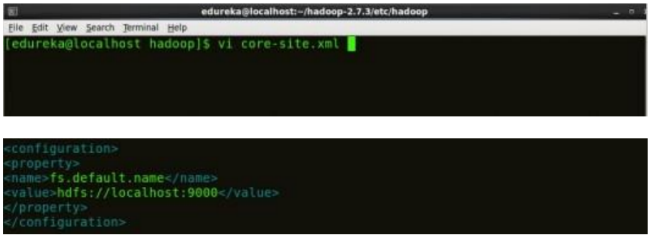


Fig: Hadoop Installation – Configuring core-site.xml

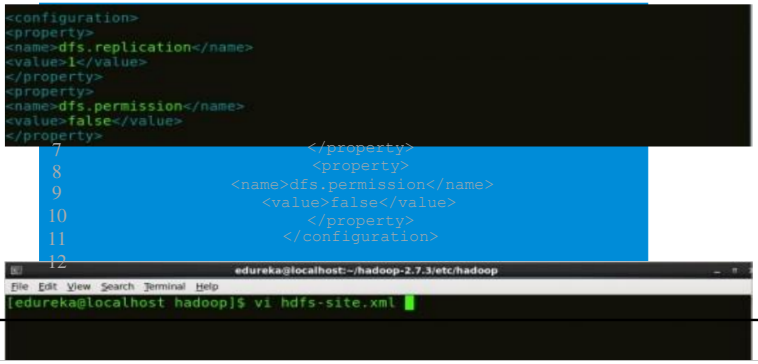
```
<?xmlversion="1.0"encoding="UTF-8"?>
<?xml-stylesheettype="text/xsl"href="configuration.xsl"?>
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://localhost:9000</value>
</property>
</configuration>
```

Step 8: Edithdfs-site.xmland edit the property mentioned below inside configuration tag:

hdfs-site.xml contains configuration settings of HDFS daemons (i.e.NameNode, DataNode, Secondary NameNode). It also includes the replication factor and block size of HDFS.

Command: vi hdfs-site.xml

Fig: Hadoop Installation – Configuring hdfs-site.xml



Step 9: Edit the mapred-site.xml file and edit the property mentionedbelow inside configuration tag:

mapred-site.xml contains configuration settings of MapReduce application like number of JVM that can run in parallel, the size of the mapper and the reducer process, CPU cores available for a process,etc.

In some cases, mapred-site.xml file is not available. So, we have to create the mapred- site.xml file using mapred-site.xml template.

Command: cp mapred-site.xml.template mapred-site.xml

Command: vi mapred-site.xml.

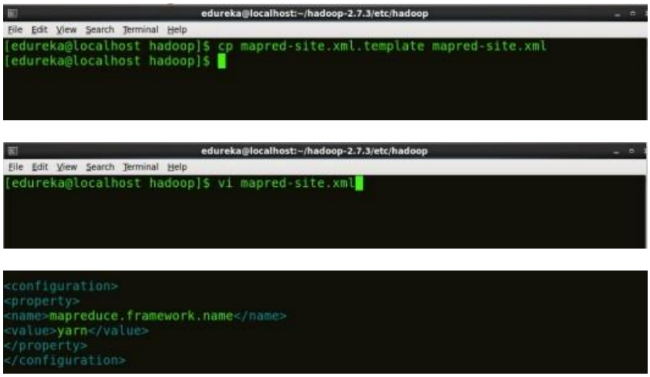


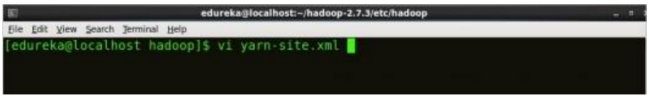
Fig: Hadoop Installation – Configuring mapred-site.xml

```
<?xmlversion="1.0"encoding="UTF-8"?>
<?xml-stylesheettype="text/xsl"href="configuration.xsl"?>
<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
</configuration>
```

Step 10: Edit yarn-site.xml and edit the property mentioned below inside configuration tag:

yarn-site.xml contains configuration settings of ResourceManager and NodeManager like application memory management size, the operation needed on program & algorithm, etc.

Command: vi yarn-site.xml



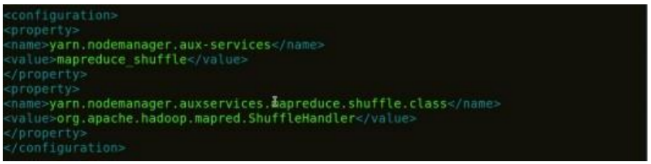


Fig: Hadoop Installation – Configuring yarn-site.xml

```
<?xmlversion="1.0">
<configuration>
<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
<property>
<name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</
name>
<value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
</configuration>
```

Step 11: Edit hadoop-env.sh and add the Java Path as mentioned below:

hadoop-env.sh contains the environment variables that are used in the script to run Hadoop like Java home path, etc.

Command: vi hadoop–env.sh

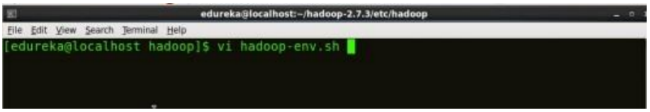


Fig: Hadoop Installation – Configuring hadoop-env.sh Step 12:

Go to Hadoop home directory and format the NameNode. Command: cd

Command: cd hadoop-2.7.3

Command: bin/hadoopnamenode -format

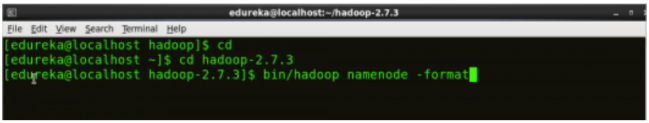


Fig: Hadoop Installation – Formatting NameNode

This formats the HDFS via NameNode. This command is only executed for the first time.

Formatting the file system means initializing the directory specified by the dfs.name.dir variable.

Never format, up and running Hadoop filesystem. You will lose all your data stored in theHDFS.

Step 13: Once the NameNode is formatted, go to hadoop-2.7.3/sbin directory and start all the daemons.

Command: cd hadoop-2.7.3/sbin

Either you can start all daemons with a single command or do it individually.

Command: ./start-all.sh

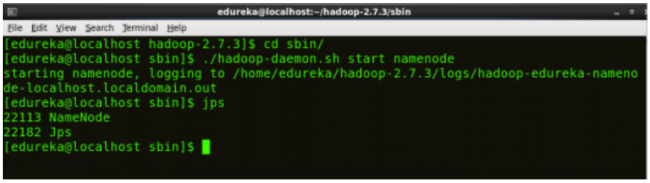
The above command is a combination of start-dfs.sh, start-yarn.sh &mr-jobhistory-daemon.sh

Or you can run all the services individually as below:

Start NameNode:

The NameNode is the centerpiece of an HDFS file system. It keeps the directory tree of all files stored in the HDFS and tracks all the file stored across the cluster.

Command: ./hadoop-daemon.sh start namenode



StartDataNode:

ig: Hadoop Installation

On startup, a DataNode connects to the Namenode and it responds to the requests from the Namenode for differentoperations.

Command: ./hadoop-daemon.sh start datanode

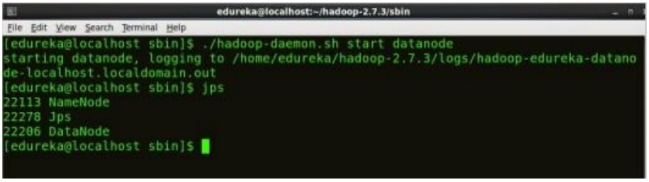


Fig: Hadoop Installation – Starting DataNode

Start ResourceManager:

ResourceManager is the master that arbitrates all the available cluster resources and thus helps in managing the distributed applications running on the YARN system. Its work is to manage each NodeManagers and the each application's ApplicationMaster.

Command: ./yarn-daemon.sh start resourcemanager

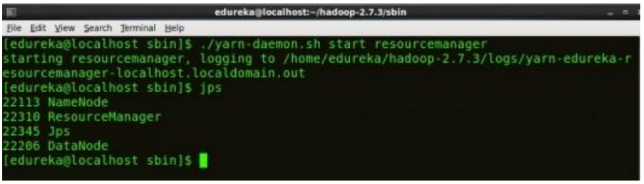


Fig: Hadoop Installation – Starting ResourceManager

Start NodeManager:

The NodeManager in each machine framework is the agent which is responsible for managing containers, monitoring their resource usage and reporting the same to the ResourceManager.

Command: ./yarn-daemon.sh start nodemanager

Fig: Hadoop Installation – Starting NodeManager

Start JobHistoryServer:

JobHistoryServer is responsible for servicing all job history related requests from client.

Command: ./mr-jobhistory-daemon.sh start historyserver

Step 14: To check that all the Hadoop services are up and running, run the below command.

Command: jps

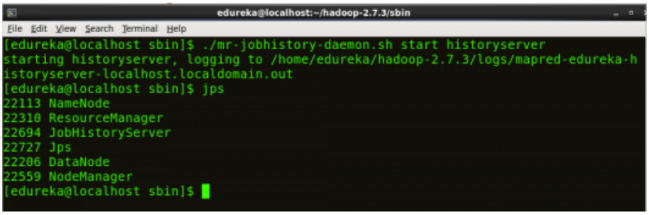


Fig: Hadoop Installation – Checking Daemons

Step 15: Now open the Mozilla browser and go to localhost:50070/dfshealth.html to check the NameNode interface.

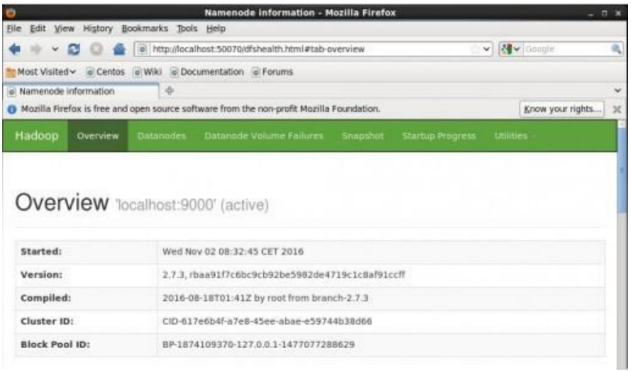


Fig: Hadoop Installation – Starting WebUI

Congratulations, you have successfully installed a single node Hadoop cluster

RESULT:

Thus, the installation of Hadoop single node cluster and simple application like wordcount was installed and verified successfully.

Ex No: 8 Date:	Creating and Executing Your First Container Using Docker

AIM

To create and execute your first container using docker.

PROCEDURE

To access a docker client, either on localhost, use a terminal from Theia - Cloud IDE at https://labs.cognitiveclass.ai/tools/theiadocker or be using Play with Docker for example.

Run docker -h,

```
$ docker -h
Flag shorthand -h has been deprecated, please use --help

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

...
Management Commands:
  builder     Manage builds
  config      Manage Docker configs
  container   Manage containers
  engine      Manage the docker engine
  image       Manage images
  network     Manage networks
  node        Manage Swarm nodes
  plugin      Manage plugins
  secret      Manage Docker secrets
  service     Manage services
  stack       Manage Docker stacks
  swarm       Manage Swarm
  system      Manage Docker
  trust       Manage trust on Docker images
  volume      Manage volumes
```

The Docker command line can be used to manage several features of the Docker Engine. In this lab, we will mainly focus on the container command.

If podman is installed, you can run the alternative command for comparison.

sudo podman -h

You can additionally review the version of your Docker installation,

docker version

Client:
 Version: 19.03.6
 ...

Server: Docker Engine - Community
 Engine
  Version: 19.03.5
  ...

You note that Docker installs both a Client and a Server: Docker Engine. For instance, if you run the same command for podman, you will see only a CLI version, because podman runs daemonless and relies on an OCI compliant container runtime (runc, crun, runv etc) to interface with the OS to create the running containers.

sudo podman version --events-backend=none
Version: 2.1.1
API Version: 2.0.0
Go Version: go1.15.2
Built: Thu Jan 1 00:00:00 1970
OS/Arch: linux/amd64

Step 1: Run your first container

We are going to use the Docker CLI to run our first container.

1. Open a terminal on your local computer
2. Run docker container run -t ubuntu top

   Use the docker container run command to run a container with the ubuntu image using the top command. The -t flags allocate a pseudo-TTY which we need for the top to work correctly.

   $ docker container run -it ubuntu top
   Unable to find image 'ubuntu:latest' locally
   latest: Pulling from library/ubuntu
   aafe6b5e13de: Pull complete
   0a2b43a72660: Pull complete
   18bdd1e546d2: Pull complete
   8198342c3e05: Pull complete
   f56970a44fd4: Pull complete
   Digest:
   sha256:f3a61450ae43896c4332bda5e78b453f4a93179045f20c8181043b26b5e790
   28
   Status: Downloaded newer image for ubuntu:latest

   The docker run command will result first in a docker pull to download the ubuntu image onto your host. Once it is downloaded, it will start the container. The output for the running container should look like this:

   top - 20:32:46 up 3 days, 17:40, 0 users, load average: 0.00, 0.01, 0.00
   Tasks: 1 total, 1 running, 0 sleeping, 0 stopped, 0 zombie

KiB Swap: 1048572 total, 1048572 free,   0 used. 1548356 avail Mem

PID USER   PR NI   VIRT   RES   SHR S %CPU %MEM   TIME+ COMMAND
  1 root   20   0   36636   3072   2640 R   0.3 0.2   0:00.04 top

top is a linux utility that prints the processes on a system and orders them by resource consumption. Notice that there is only a single process in this output: it is the top process itself. We don't see other processes from our host in this list because of the PID namespace isolation.

Containers use linux namespaces to provide isolation of system resources from other containers or the host. The PID namespace provides isolation for process IDs. If you run top while inside the container, you will notice that it shows the processes within the PID namespace of the container, which is much different than what you can see if you ran top on the host.

Even though we are using the ubuntu image, it is important to note that our container does not have its own kernel. Its uses the kernel of the host and the ubuntu image is used only to provide the file system and tools available on an ubuntu system.

Inspect the container with docker container exec

The docker container exec command is a way to "enter" a running container's namespaces with a new process.

Open a new terminal. On cognitiveclass.ai, select Terminal > New Terminal.

Using play-with-docker.com, to open a new terminal connected to node1, click "Add New Instance" on the lefthand side, then ssh from node2 into node1 using the IP that is listed by 'node1 '. For example:

```
[node2] (local) root@192.168.0.17 ~
$ ssh 192.168.0.18
[node1] (local) root@192.168.0.18 ~
$
```

In the new terminal, use the docker container ls command to get the ID of the running container you just created.

```
$ docker container ls
CONTAINER ID        IMAGE               COMMAND             CREATED
STATUS              PORTS               NAMES
b3ad2a23fab3        ubuntu              "top"               29 minutes ago      Up 29 minutes
goofy_nobel
```

Then use that id to run bash inside that container using the docker container exec command. Since we are using bash and want to interact with this container from our terminal, use -it flags to run using interactive mode while allocating a psuedo-terminal.

```
$ docker container exec -it b3ad2a23fab3 bash
root@b3ad2a23fab3:/#
```

And Voila! We just used the docker container exec command to "enter" our container's namespaces with our bash process. Using docker container exec with bash is a common pattern to inspect a docker container.

Notice the change in the prefix of your terminal. e.g. root@b3ad2a23fab3:/. This is an indication that we are running bash "inside" of our container.

%Cpu(s): 0.0 us, 0.1 sy, 0.0 ni, 99.9 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st

Note: This is not the same as ssh'ing into a separate host or a VM. We don't need an ssh server to connect with a bash process. Remember that containers use kernel-level features to achieve isolation and that containers run on top of the kernel. Our container is just a group of processes running in isolation on the same host, and we can use docker container exec to enter that isolation with the bash process. After running docker container exec, the group of processes running in isolation (i.e. our container) include top and bash.

From the same termina, run ps -ef to inspect the running processes.

```
root@b3ad2a23fab3:/# ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 20:34 ?        00:00:00 top
root        17     0  0 21:06 ?        00:00:00 bash
root        27    17  0 21:14 ?        00:00:00 ps -ef
```

You should see only the top process, bash process and our ps process.

For comparison, exit the container, and run ps -ef or top on the host. These commands will work on linux or mac. For windows, you can inspect the running processes using tasklist.

```
root@b3ad2a23fab3:/# exit
exit
$ ps -ef
# Lots of processes!
```

Technical Deep Dive PID is just one of the linux namespaces that provides containers with isolation to system resources. Other linux namespaces include: - MNT - Mount and unmount directories without affecting other namespaces - NET - Containers have their own network stack - IPC - Isolated interprocess communication mechanisms such as message queues. - User - Isolated view of users on the system - UTC - Set hostname and domain name per container

These namespaces together provide the isolation for containers that allow them to run together securely and without conflict with other containers running on the same system. Next, we will demonstrate different uses of containers. and the benefit of isolation as we run multiple containers on the same host.

Note: Namespaces are a feature of the **linux** kernel. But Docker allows you to run containers on Windows and Mac... how does that work? The secret is that embedded in the Docker product or Docker engine is a linux subsystem. Docker open-sourced this linux subsystem to a new project: LinuxKit. Being able to run containers on many different platforms is one advantage of using the Docker tooling with containers.

In addition to running linux containers on Windows using a linux subsystem, native Windows containers are now possible due the creation of container.

primitives on the Windows OS. Native Windows containers can be run on Windows 10 or Windows Server 2016 or newer.

Note: if you run this exercise in a containerized terminal and execute the ps -ef command in the terminal, e.g. in https://labs.cognitiveclass.ai, you will still see a limited set of processes after exiting the exec command. You can try to run the ps -ef command in a terminal on your local machine to see all processes.

1. Clean up the container running the top processes by typing: <ctrl>-c, list all containers and remove the containers by their id.
2. docker ps -a

3. docker rm <CONTAINER ID>

RESULT

Ex No: 9	Run a Container from Docker Hub
Date:	

AIM

To create and run a container from the Docker Hub.

PROCEDURE

The following section contains step-by-step instructions on how to get started with Docker Hub.

Step 1: Sign up for a Docker account

Start by creating a Docker IDopen_in_new.

A Docker ID grants you access to Docker Hub repositories and lets you explore available images from the community and verified publishers. You also need a Docker ID to share images on Docker Hub.

Step 2: Create your first repository

To create a repository:

1. Sign in to Docker Hubopen_in_new.
2. Select Create a Repository on the Docker Hub welcome page.
3. Name it <your-username>/my-private-repo.
4. Set the visibility to Private.
5. Select Create.

You've created your first repository.

Step 3: Download and install Docker Desktop

You need to download Docker Desktop to build, push, and pull container images.

1. Download and install Docker Desktop.
2. Sign in to Docker Desktop using the Docker ID you created in step one.

Step 4: Pull and run a container image from Docker Hub

In your terminal, run docker pull hello-world to pull the image from Docker Hub. You should see output similar to:

1. $ docker pull hello-world
2. Using default tag: latest
3. latest: Pulling from library/hello-world
4. 2db29710123e: Pull complete

5. Digest: sha256:7d246653d0511db2a6b2e0436cfd0e52ac8c066000264b3ce63331ac66dca625
6. Status: Downloaded newer image for hello-world:latest
7. docker.io/library/hello-world:latest
8. Run docker run hello-world to run the image locally. You should see output similar to:
9. $ docker run hello-world
10. Hello from Docker!
11. This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:

1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub. (amd64)
3. The Docker daemon created a new container from that image which runs the executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

Step 5: Build and push a container image to Docker Hub from your computer

1. Start by creating a Dockerfile to specify your application as shown below:
2. *# syntax=docker/dockerfile:1*
3. FROM busybox

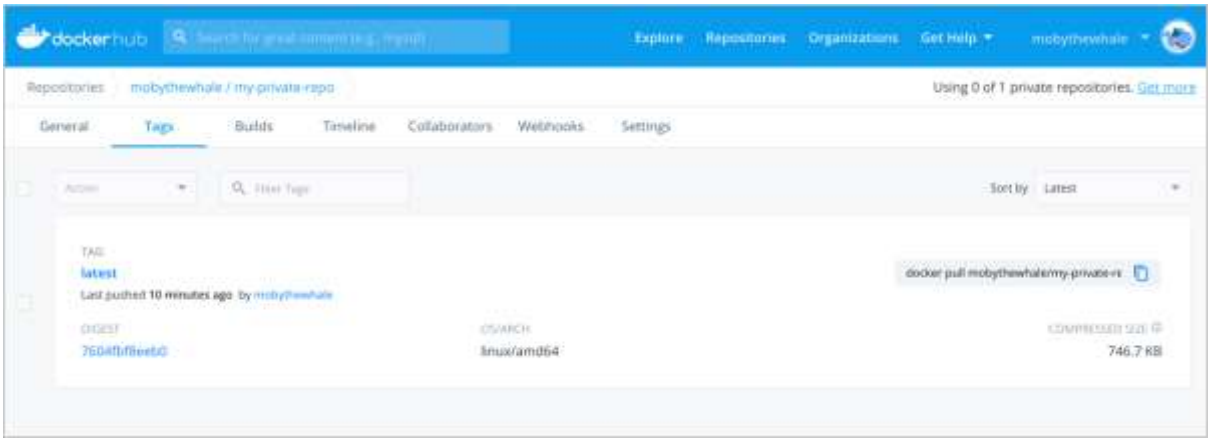
CMD echo "Hello world! This is my first Docker image."

4. Run docker build -t <your_username>/my-private-repo . to build your Docker image.
5. Run docker run <your_username>/my-private-repo to test your Docker image locally.
6. Run docker push <your_username>/my-private-repo to push your Docker image to Docker Hub. You should see output similar to:

```
cat > Dockerfile <<EOF
FROM busybox
CMD echo "Hello world! This is my first Docker image."
EOF
docker build -t mobythewhale/my-private-repo .
[+] Building 1.2s (5/5) FINISHED
 => [internal] load build definition from Dockerfile                    0.0s
 => => transferring dockerfile: 110B                                    0.0s
 => [internal] load .dockerignore                                       0.0s
 => => transferring context: 2B                                         0.0s
 => [internal] load metadata for docker.io/library/busybox:latest       1.2s
 => CACHED [1/1] FROM docker.io/library/busybox@sha256:a9286defaba7b3a519  0.0s
 => exporting to image                                                  0.0s
 => => exporting layers                                                 0.0s
 => => writing image sha256:dcdb1fd928bfb257bfc0122ea47accd911a3a386ce618  0.0s
 => => naming to docker.io/mobythewhale/my-private-repo                 0.0s
docker run mobythewhale/my-private-repo
Hello world! This is my first Docker image.
docker push mobythewhale/my-private-repo
The push refers to repository [docker.io/mobythewhale/my-private-repo]
d2421964bad1: Layer already exists
latest: digest: sha256:7604fbf8eeb03d866fd005fa95cdbb802274bf9fa51f7dafba6658294
efa9baa size: 526
```

You must be signed in to Docker Hub through Docker Desktop or the command line, and you must also name your images correctly, as per the above steps.

1. Your repository in Docker Hub should now display a new latest tag under Tags:



RESULT

Ex.No:10	Online Openstack Demo Version
Date :	

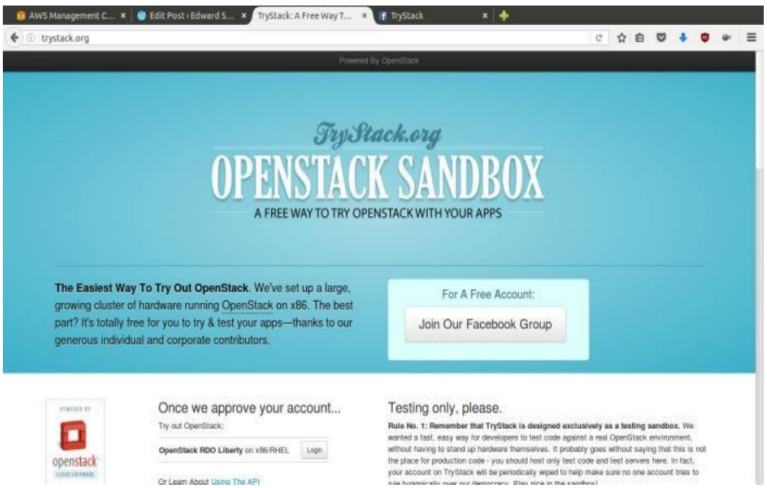
AIM:

To study how to launch the installation and configuration of open stack private cloud.

PROCEDURE:

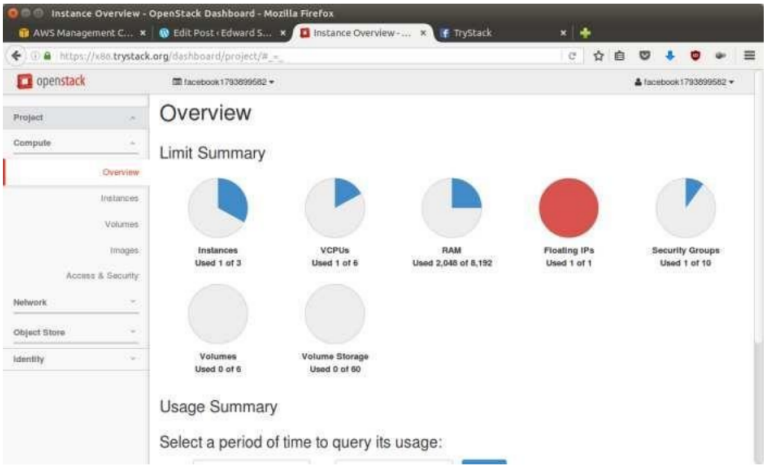
OpenStack is an open-source software cloud computing platform. OpenStack is primarily used for deploying an infrastructure as a service (IaaS) solution like Amazon Web Service (AWS). In other words, you can make your own AWS by using OpenStack. If you want to try out OpenStack, TryStackis the easiest and free way to do it.

In order to try OpenStack in TryStack, you must register yourself by joining TryStackFacebookGroup. The acceptance of group needs a couple days because it's approved manually. After you have been accepted in the TryStack Group, you can log in TryStack.



TryStack.org Homepage

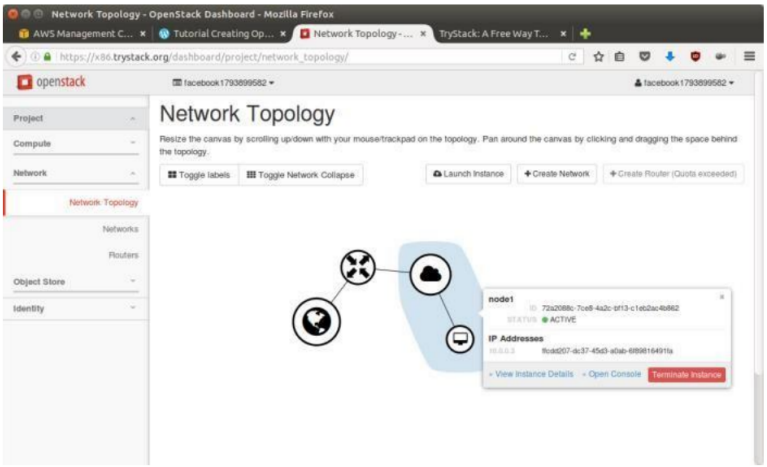
If you already join to the Facebook Group and login to the dashboard. After youlog in to the TryStack, you will see the Compute Dashboard like:



OpenStack Compute Dashboard

Overview: What we will do?

In this post, I will show you how to run an OpenStack instance. The instance will be accessible through the internet (have a public IP address). The final topology will like:



Network topology

As you see from the image above, the instance will be connected to a local network and the local network will be connected to internet.

Step 1: Create Network

The network in here is our own local network. So, your instances will be not mixed up with the others. You can imagine this as your own LAN (Local Area Network) in the cloud.

1. Go to **Network > Networks** and then click **CreateNetwork**.
2. In **Network** tab, fill **Network Name** for example internal and then click**Next**.
3. In **Subnet**tab,
   1. Fill **Network Address** with appropriate CIDR, for example 192.168.1.0/24. Use private network CIDR block as the bestpractice.
   2. Select **IP Version** with appropriate IP version, in this caseIPv4.
   3. Click**Next**.
4. In **Subnet Details** tab, fill **DNS Name Servers** with 8.8.8.8 (Google DNS) and then click **Create**.

Step 2: Create Instance

Now, we will create an instance. The instance is a virtual machine in the cloud, like AWS EC2. You need the instance to connect to the network that we just created in the previous step.

1. Go to **Compute > Instances** and then click **LaunchInstance**.
2. In **Details**tab,
   Fill **Instance Name**, for example Ubuntu1.
   Select **Flavor**, for examplem1.medium.
   Fill **Instance Count** with**1**.
   Select **Instance Boot Source** with **Boot fromImage**.
   Select **Image Name** with **Ubuntu 14.04 amd64 (243.7 MB)** if you want install Ubuntu 14.04 in your virtual machine.
3. In **Access &Security**tab,
   Click [**+**] button of **Key Pair** to import key pair. This key pair is a public and private keythat we will use to connect to the instance from ourmachine.
   In **Import Key Pair**dialog,
   Fill **Key Pair Name** with your machine name (for exampleEdward-Key).
   Fill **Public Key** with your **SSH public key** (usually is in ~/.ssh/id_rsa.pub). See description in Import Key Pair dialog box for more information. If you are using Windows, you can use **Puttygen** to generate keypair.
   Click **Import keypair**.
   In **Security Groups**, mark/check**default**.
   In **Networking**tab,
   In **Selected Networks**, select network that have been created in Step 1, for example internal.
   Click **Launch**.

1. If you want to create multiple instances, you can repeat step 1-5. I created one more instance
2. with instance name Ubuntu2.

Step 3: Create Router

I guess you already know what router is. In the step 1, we created our network, but it is isolated. It doesn't connect to the internet. To make our network has an internet connection, we need a router that running as the gateway to the internet.

1. Go to **Network > Routers** and then click **CreateRouter**.
2. Fill **Router Name** for example router1 and then click **Createrouter**.
3. Click on your **router name link**, for example router1, **Router Details**page.
4. Click **Set Gateway** button in upperright:
    - Select **External networks** with**external**.
    - Then**OK**.
    - Click **Add Interface**button.
    - Select **Subnet** with the network that you have been created in Step 1.
    - Click **Addinterface**.
5. Go to **Network > Network Topology**. You will see the network topology. In the example, there are two network, i.e. external and internal, those are bridged by a router. Thereareinstances those are joined to internal network.

Step 4: Configure Floating IP Address

Floating IP address is public IP address. It makes your instance is accessible from the internet. When you launch your instance, the instance will have a private network IP, but no public IP. In OpenStack, the public IPs is collected in a pool and managed by admin (in our case is TryStack). You need to request a public (floating) IP address to be assigned to your instance.

1. Go to **Compute >Instance**.
2. In one of your instances, click **More > Associate FloatingIP**.
3. In **IP Address**, click Plus[**+**].
4. Select **Pool** to **external** and then click **AllocateIP**.
5. Click**Associate**.
6. Now you will get a public IP, e.g. 8.21.28.120, for yourinstance.

Step 5: Configure Access & Security

OpenStack has a feature like a firewall. It can whitelist/blacklist your in/out connection. It is called *Security Group*.

1. Go to **Compute > Access & Security** and then open **Security Groups**tab.
2. In **default** row, click **ManageRules**.

3. Click **Add Rule**, choose **ALL ICMP** rule to enable ping into your instance, and then click**Add**.
4. Click **Add Rule**, choose **HTTP** rule to open HTTP port (port 80), and then click**Add**.
5. Click **Add Rule**, choose **SSH** rule to open SSH port (port 22), and then click**Add**.
6. You can open other ports by creating newrules.

Step 6: SSH to Your Instance

Now, you can SSH your instances to the floating IP address that you got in the step 4. If you are using Ubuntu image, the SSH user will be ubuntu.

RESULT:

Thus, the installation and configuration of open stack private cloud was studied.