RGM COLLEGE OF ENGINEERING AND TECHNOLOGY
(AUTONOMOUS)
(ESTD-1995)
Accredited by NAAC of UGC, New Delhi with 'A' Grade
Nandyal - 518 501. Kurnool (Dist.) A.P.

(ESTD-1995)

INDEX

Sl. No.	Date	Name of the Experiment	Page No.	Marks	Remarks
1.	28/11/24	Install virtual box/ VM ware workstation with different flavours of Linux or windows os on top of windows 10 or 11	1-2	10	5/12/24
2.	5/12/24	Install c compiler in the virtual machine created using virtual box & execute simple programs.	3-5	10	12/12/24
3.	12/12/24	Install Google App Engine create hellowoorld and other simple web application using java	6-7	10	12/12/24
4.	19/12/24	use GAE Launcher to launch the web applic ations.	8-11	10	2/1/25
5.	2/1/25	Simulate a cloud scena rio using cloudsim & run a scheduling algo rithm that is not	12-18	10	23/1/25

RGM COLLEGE OF ENGINEERING AND TECHNOLOGY
(AUTONOMOUS)
(ESTD-1995)
Accredited by NAAC of UGC, New Delhi with 'A' Grade
Nandyal - 518 501. Kurnool (Dist.) A.P.

(ESTD-1995)

INDEX

Sl. No.	Date	Name of the Experiment	Page No.	Marks	Remarks
		present in cloudsim.			
6.	23/1/25	find a procedure to transfer the files from one virtual machine to another virtual machine	19	10	13/2/25
7.	13/2/25	find a procedure to launch VM using trystack.	20-23	10	20/2/25
8.	20/2/25	simulate to install hadoop single node cluster & run simple applications like a wordcount	24-29	10	13/3/25
9.	13/3/25	simulate a secure file sharing mechanisms using cloudsim.	30-35	10	13/3/25

Ex NO:01

Date: 28/11/24

1) Install virtual Box workstations with different flavours of Linux or windows OS on top of the windows 10 or 11.

Aim: To install virtual Box or VM ware workstation to run different flavours of Linux or windows on top of windows 10 or 11.

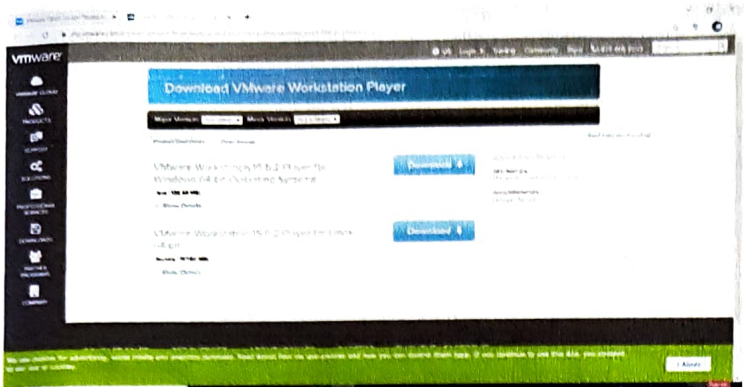
Description: Oracle VM virtual Box is a virtualization software that can be used in cloud computing to create and run multiple VM on a single device.
It can be installed on a variety of platforms, including a cloud instance, and is used by developers, IT Terms and solution providers.
VM ware is a cloud computing & virtualization company that help users convert physical hardware into virtual machines. VMs are isolated environments that can run programs, store data & connect to network, all while using virtual resources, instead of physical components.
VM ware is commercial software with advanced features, while virtual Box is a opensource with basic virtualization functionalities.

Prerequisites:
1. windows 10/11 installed on the host machine.
2. Downloaded virtual Box / VM ware software.

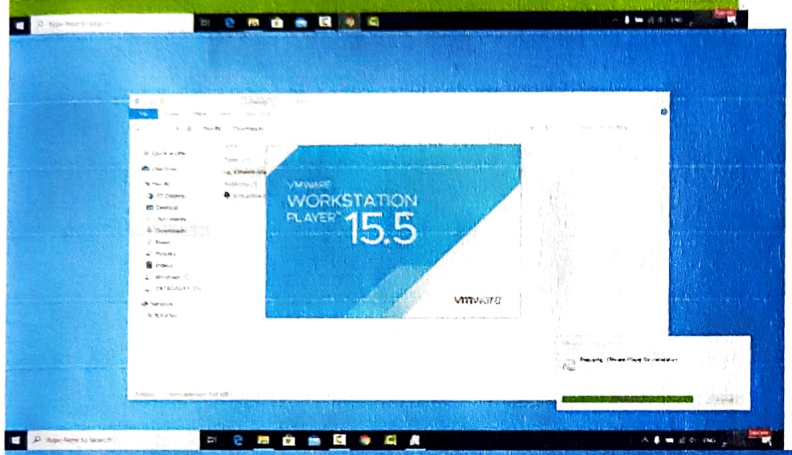
Steps:

1. Download : Download the latest version of the virtual Box from the official website.
2. Install : open the downloaded application follow steps in the windows installation.
3. create a virtual machine :
   - go to machine > New
   - Enter the following Info
   - Name
   - type
   - version
   - RAM
   - Hard disk setting
   - LINUX ISO setting
   - LINUX ISO file
   - OS info
   - path for virtual hard disks
   - N/w settings
   - processor settings
   - storage settings.
4. start the virtual machine
5. Install the OS.

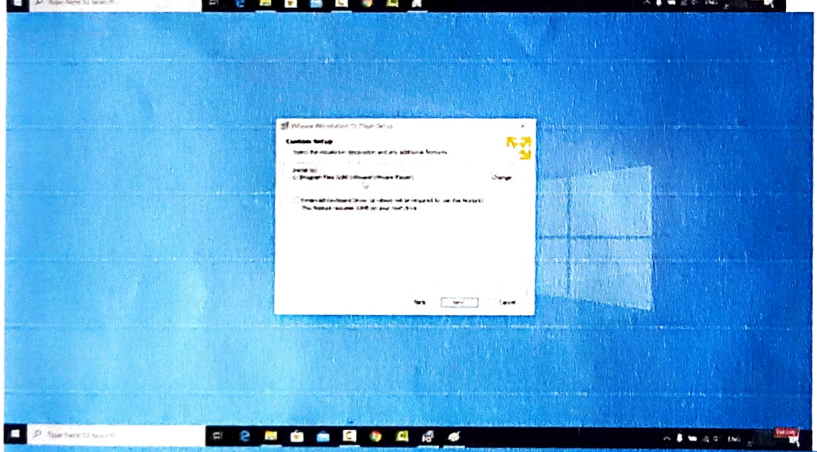
Result : Hence virtual Box or VM ware workstation to run different flavours of linux or window on top of window 10 installed successfully.



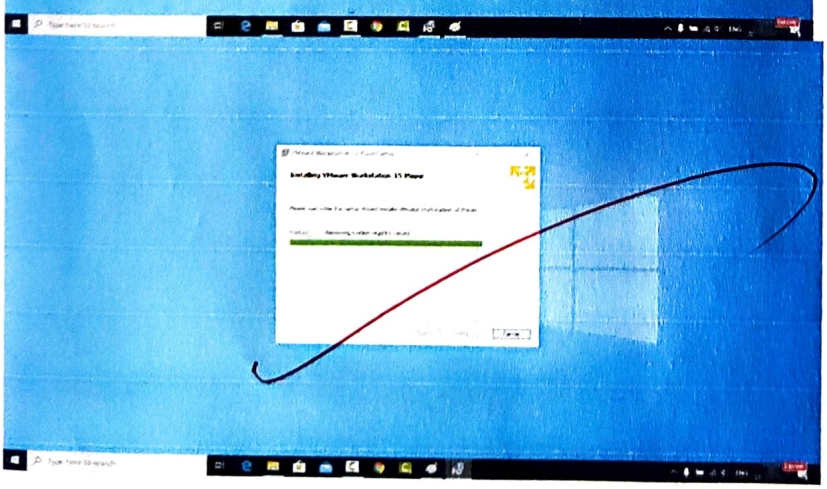
→ Download VMware workstation



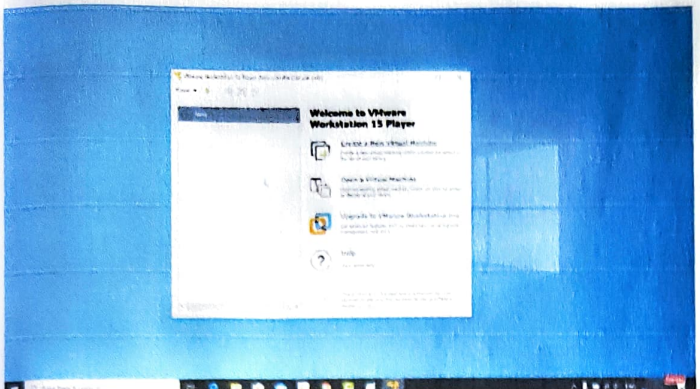
→ VMware workstation



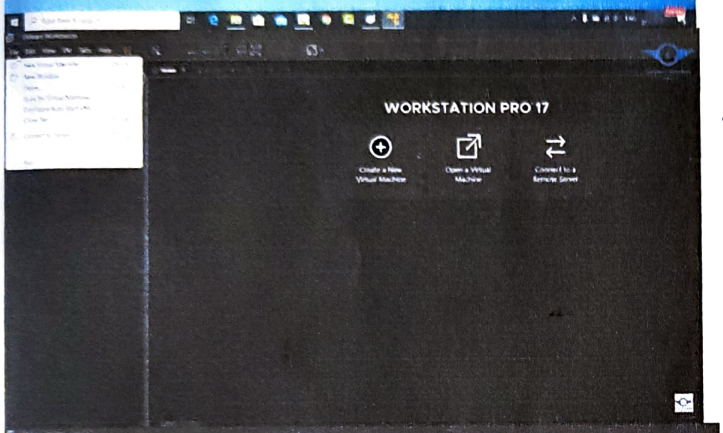
→ custom setup



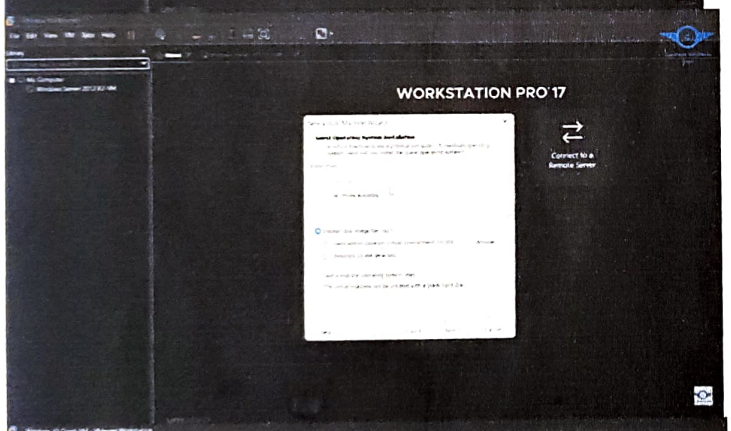
→ Installing VMware



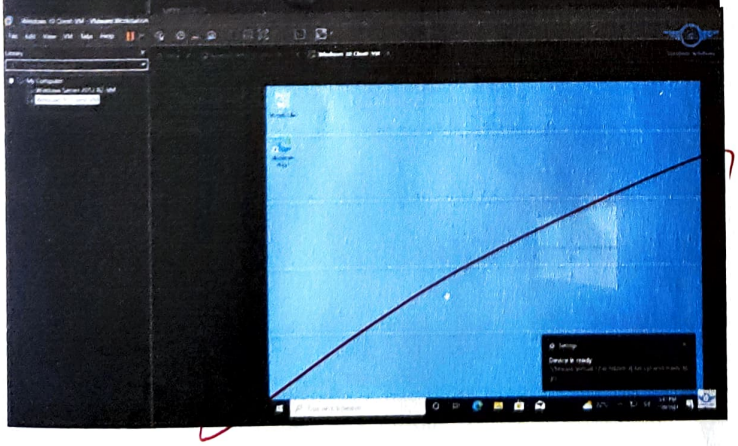
→ Welcome to VMware



→ VMware window



→ Guest OS Installation window



→ window IDE in VM ware

EXP NO: 2

Date: 5/12/24

Install a c compiler in VM created using virtual Box and to execute simple program.

Aim: To install a c compiler in the virtual machine created using virtual.

prereq sites:
1. virtual machine created in expt 1
2. Basic knowledge of c programming.

procedure:

1. Access the VM
   → start the VM
   → open terminal
2. Install gcc compiler.
   → for Linux-based systems, run the command
   sudo apt update
   sudo apt install gcc
3) verify and compile a simple program.
   create a file named hello.c

```
#include <stdio.h>
int main(){
print("Hello world");
return 0;
}
```

→ compile the program
   gcc hello.c

→ Run the program.

Experiment with Additional programs

4. write programs to perform addition, calculate factorial, or find the largest in an array.

1. Addition of two numbers :

```
#include <stdio.h>
int main(){
   int a,b;
   printf("Enter two numbers :");
   scanf("%d %d",&a,&b);
   printf("sum :%d\n",a+b);
   return 0;
}
```

output: Enter two numbers :10 20
sum :30

2. calculate factorial :

```
#include <stdio.h>
int main(){
   int n,i,fact =1;
   printf("Enter a number");
   scanf("%d",&n);
   for(i=1;i<=n;i++){
    fact *=i;
   }
   printf("factorial :%d\n",fact);
   return 0;
   }
```

output: Enter a number : 5
factorial :120

3. Largest number in an array:

```
#include <stdio.h>
int main() {
    int n, i, max;
    printf("Enter number of elements");
    scanf("%d", &n);
    int arr[n];
    printf("Enter numbers:");
    for(i=0; i<n; i++){
      scanf("%d", &arr[i]);
    }
    max = arr[0];
    for(i=1; i<n; i++){
        if(arr[i] > max){
            max=arr[i]
        }
    }
    printf("Largest: %d\n", max);
    return 0;
}
```

output: Enter number of elements: 3
Enter numbers: 10 20 30
Largest: 30

Result: Hence a c compiler in the virtual machine created using virtual box and to execute simple programs installed successfully.

12/12

Expno: 03

Date: 12|12|24

Install google App Engine. Create hello world app & other simple web applications using python or java.

Aim: To set up google App Engine (GAE) & deploy simple web applications using python/java.

Description: Google App Engine is mostly used to run web applications. these dynamic scales as demand change overtime because of google's vast computing infrastructure. Because if offers a secure execution environment in addition to a number of services App engine makes it easier to develop scalable & high performance web apps. Google's applications will scale up and down in response to shifting demand. It is a scalable runtime environment.

Pre-requisites:
→ python or java installed on the host machine.
→ google cloud platform account with billing enabled.

Procedure:
1. Install google cloud SDK
- Download SDK from google cloud website.
- Install SDK & initialize it:

```
gcloud init.
```

- create a simple web applications
- for python : create a file named main.py with following content :

python
copy code

```
from flask import flask
app = Flask(__name__)
@app.route('/')
def hello_world():
    return 'Hello, world!'
if __name__ == '__main__':
    app.run(host='127.0.0.1', port=8080, debug=True)
```

- create an app.yaml file for deployment

yaml
copy code

```
runtime : python39
entrypoint : python main.py
```

- Deploy the application
- Run the following commands :

```
gcloud app deploy
gcloud app browse.
```

Result : Hence Installation of google App Engine & hello world app & other simple web applications using python created successfully.

10/12

EXPNO:04

Date: 19/12/24

Use GAE Launcher to launch the web applications.

Aim: To deploy and launch simple web applications on google App Engine using the GAE Launcher.

Description: TO GAE Launcher is a graphical front-end that allows user to run & deploy applications developed for GAE. It is avoided, available for windows & is similar to the launcher for MacOS.

pre-requisites:
- → python or java development environment installed on your system.
- → google cloud SDK installed & configured.
- → A google cloud account with biling enabled.
- → Basic understanding of web development concepts.

step-by-step procedure

1. Install and set up google cloud SDK

1. Download and Install google cloud SDK: visit the google cloud SDK installation page & doonload the appropriate version for your OS. Follow the installation instructions specific to your OS.

2. Authenticate your google cloud Account

Run the following command in terminal or the command and prompt:

gcloud auth login

This will open a browser window for authentication. Log in using your google account.

Set the Active project: Select a google account project or create a new one using command:

gcloud projects create <PROJECT_ID>

gcloud config set project <PROJECT_ID>

2. Install google app engine launcher

1. Install GAE Launcher: Download the google App Engine launcher for your OS from official download page. follow installation wizard to complete the set up.

2. Verify Installation: Open the GAE launcher & ensure it displays the interface for managing & deploying projects.

3. Create a web Application:

1. set up Application Directory: create a folder named my app & navigate to it. Inside folder, create two files:

1. app.yaml: configuration file for GAE Application

2. main.py: python script for web applications.

3. write python App code:

In main.py, write a simple web application

python:

```
from flask import flask
app = Flask (__name__)
@app.route('/')
def home():
    return 'Hello, world! welcome to GAE!'
if __name__ ='__main__':
    app.run(host='127.0.0.1', port=8080,
                debug=True)
```

1. configure app.yaml:

Add the following content to define runtime environment for application:

```
yaml:
runtime : python39
entrypoint : python main.py
handlers:
 -url : /.*
 script : auto
```

2. Install Flask: Install flask in your development environment:

pip install flask.

3. Test the Application locally

1. Open the GAE launcher
2. Add the application folder to the GAE launcher by selecting file > Add existing APP.

3. Run the application locally using the Run button in the GAE launcher.

4. open your browser & navigate to
http://127.0.0.1:8080
• verify that the message "Hello, world! welcome to google App Engine!" is displayed.

5. Deploy the Application to GAE
1. open the GAE launcher
select the application in launcher
2. Deploy the Application
click on Deploy button
3. verify Deployment
After deployment is complete, open browser & navigate it.
https://<PROJECT_ID>.appspot.com
• Ensure web app is running live

6. Modify & Redeploy the App
1. Make changes to main.py
2. Re deploy using GAE Launcher
the changes will reflect on live app URL

Expected output (or)
Executed output: when app is accessed through browser, it should display:
CSS
copy code
Hello, world! welcome to google App Engine

Result: Hence web app by using GAE launcher deployed successfully

19/12/24

EXP NO 05

Date: 2/1/25

simulate a cloud scenario using cloudsim and run a scheduling algorithm that is not present in cloudsim.

Aim: To simulate a cloudsim scenario using cloudsim & implement a custom scheduling algorithm which is not present in cloudsim.

Description: cloudsim is a open-source framework which is used to simulate cloud computing infrastructure and services. It is developed by the clouds Lab organization is written entirely in java. It is used for modelling & simulating a cloud computing environment as a means for evaluating a hypothesis is prior to software development in order to reproduce tests & results.

prerequisites:

cloud sim installed

Basic knowledge of java.

Eclipse IDE or Intellij IDEA installed on the system.

cloudsim toolkit downloaded.

Basic understanding of cloud resource management & scheduling algorithms.

Tools Required:

cloudsim Toolkit: A framework for simulate

cloud computing scenario.

- Java Development kit: Ensure JDK is installed
- Eclipse IDE: IDE for java development.

Procedure:

- set up the cloudsim environment
- Define datacenter configuration & virtual machines.
- write & integrate a custom scheduling algorithm
- simulate the scenario & Analyze results.

Step1: setup cloudsim Environment

1. Download cloudsim

→ Download the cloudsim source code from it's the github repository.

2. Import cloudsim into Eclipse

→ open Eclipse & create a new java project.

→ Right-click the project > Build path > Add External Archives.

→ Add the cloudsim-<version>.jar file & its dependencies.

3. Verify setup:

- create sample main class & write a basic program to check the cloudsim environment.

```
import org.cloudbus.cloudsim.core.cloudsim;
public class cloudsimTest{
    public static void main(String[] args){
System.out.println("cloudsim Environment set
                    up successfully");
}
}
```

• Run the program to ensure there are no errors.

Step2 : Create cloud scenario

1. Initialize cloudsim

• start the cloudsim environment

cloudsim.init (numUsers, calendar, traceflag);

• numUsers : Numbers of users sharing the cloud infrastructure.

• calendar : A calendar instance.

• traceflag : Boolean to enable or disable event tracing.

2. Define a data center

. Create data center with hosts, each having a specific configuration.

```
Data center datacenter =CreateDataCenter("Data
        center :0");
Private <Host> hostList =new ArrayList<>();
 Int ram = 2048;
 long storage =10000000;
 int bw =10000;
 int numpes =4;
for(int i =0;i<2;i++) {
 hostList.add (new Host(i,
        new RamProvisionerSimple(ram);
        new BwProvisionerSimple(bw);
        storage,
    List.of(new Pe(0, newProvisionerSimple(1000))
    new VmSchedulerTimeShared(List of (new Pe(0,
```

```
new peprovisionersimple(1000))))));
}
return new Datacenter(name,hostlist,new VMAll
    ocationpolicysimple(hostlist),new LinkedList<>
                                    (),0);
}
```

3. Create virtual machines (VMs)

Define VMs to be hosted in the data center;

```
vm vm=new Vm(vmId, brokerId, mips, pesNumb-
        er, ram, bw, size, vmm, new CloudletSch-
        edulerTimeshared());
vmlist.add(vm);
```

4. Create cloudlets:

Create tasks (cloudlets) that will be executed on VMs

```
cloudlet cloudlet=new cloudlet(cloudletId, len
  gth, pesNumber, filesize, outputsize, utilizati
  onModel, utilizationModel, utilizationModel);
cloudlist.add(cloudlet);
```

Step 3: Implement a customer scheduling Algorithm

1. create a New scheduler class

Implement a class extends DatacenterBroker and overrides its submitCloudlets() method:

```
public class customerBroker extends Datacenter
Broker {
public customerBroker(String name) throws Exc
                                    eption {
```

```
    super(name);
  }
@override
protected void submitcloudlets(){
for(cloudlet cloudlet: getcloudletlist()){
vm bestvm =select vm for cloud (cloudlet);
bindcloudlet vm (cloudlet.get cloudletId());
bestvm.getId());
  }
}
private vm selectvm for cloudlet(cloudlet clou
dlet){
the least cloudlets assigned)
    vm bestvm = null;
   int minTasks = Integer.MAX_value;
  for(vm vm :getvmlist()){
   int tasks =getclou ist(vm).size();
   if(tasks <minTasks){
      minTasks =tasks;
       bestvm =vm;
    }
   }
   return bestvm;
  }
 }
```

2. Integrate the custom scheduler

- Replace the default broker with your custom broker:

```
CustomBroker broker = new customBroker("cus
                                tomBroker");
broker.submitvmList(vmList);
broker submit cloudletList(cloudletList);
```

Step 4: Run the simulation

1. Start the cloudsim simulation:

```
cloudsim.startsimulation();
```

2. Stop the simulation once all events are proce ssed

```
cloudsim.stopsimulation();
```

3. Collect and display results

```
List<Cloudlet> newList = broker.getcloudletRecei
                                vedList();
printCloudletResults(newList);
```

Step 5: Analyze Results

Measures metrics such as:

- Cloudlet Execution Time: Time taken for cloudlets to complete
- VM utilization: Percentage of resources utilized by VMs.

Expected output: cloudsim will simulate execution of cloudlets on VM based on the custom scheduling alg.

the results will show:

- Cloudlets ID's VM assignments, start times & execution times.

cloudletID	status	VMID	start time	finish time	execution time
0		0.1	2.3	2.2	
1		0.2	2.5	2.3	

Result: Hence cloud scenario using cloudsim is simulated & running a scheduling algorithm executed successfully.

23/1/25

EXP NO :06 | Date: 23/1/25

6. Find a procedure to transfer the files from one virtual machine to another virtual machine

Aim: To transfer the files from one virtual machine to another virtual machine.

Description: A virtual machine in cloud computing is a software-based computers that runs on top of another computer. VM's are a key part of cloud infrastructure because they allow users possible by virtualizations tools, which create a virtualized layer between the h/w & operating systems.

prerequisites:

1. Two running virtual machines.
2. Networking enabled b/w the VM's

procedure:

1. Enabled shared folders or Networks
   - configure a shared-folder in virtual Box
   - Alternatively, enable SSH b/w the VM's.
2. Transfer the files using SCP:
   - In VM1, run:
     scp file.txt user@VM2-IP:/path/to/destination.
3. verify file transfer
   - Access VM2 & check the destination folder.

Result: Hence files one VM to another VM transferred successfully.

EXP NO: 07 | Date: 13/02/25

7. Find a procedure to launch virtual machine using trystack.

Aim: To launch a virtual machine using Trystack, an online demo version of openstack & understand the basic operations of openstack.

Description: Trystack is a public, production openstack instance and the maintainers have learned a great deal about maintenance, upgrades, automation around running a large public cloud.

Prerequisites:

Access to a web browser with an internet connection.

A valid facebook account

Basic understanding of virtualization concepts.

Tools Required:

web browser: Latest version of google chrome, firefox or edge.

Trystack Account: Access trystack.

procedure to launch a virtual machine using trystack:

Step1: Access trystack.

1. open the trystack website

Navigate to the trystack website.

2. Login to trystack

- click on the login button, which redirects you to the facebook authentication page.
- Login using your facebook credentials.
- Once authenticated, you will be redirected to the to the openstack dashboard.

Step2: Explore the openstack Dashboard.

1. After logging in, you will see the openstack dashboard, also known as the Horizon Dashboard.
2. familiarize yourself with the main sections:
   - project: Access resources for your virtual m/c.
   - Admin: Manage overall resources.
   - Network: set up & manage network & subnets for VM's.

Step3: set up key pairs

key pairs are used for secure SSH access to the virtual machine.

1. Navigate to project > compute > keypairs.
2. click on create key pair
3. Enter a name for the key pair.
4. the private will be automatically downloaded to your local m/c as a .pem file.

Step 4 : Launch a Virtual Machine.

1. Go to project > compute > Instances.
2. click Launch instance to start the VM creation process.

Step 5 : configure VM settings

1. Detail Tab:
   - Instance Name : provide a name for your instance
   - Availability zone : Leave it as default.
2. Source code :
   - select the operating system image for the VM :
   - set the boot source to image.
   - choose an image from the list.
3. Flavour Tab :
   - select the flavour based on the required h/w configuration.
   - flavours define virtual CPU, RAM & storage size
4. Networks Tab :
   - Ensure a n/w is selected. trystack typically provides a fault network.
5. Key pair Tab :
   - select the key pair created earlier

Step 6 : Launch the Instance.

1. After configuring the settings, click Launch instance
2. the instance will appear the list under projects > compute > instances with status of building.

•Once the status changes to Active, your VM is ready.

Step7: Access the Virtual Machine.

1. Obtain the Instance's IP Address:

• In the instance list, note the following / floating IP address associated with the VM.

• If no floating IP is assigned, go to project > N/W > floating IP's to allocate one & associate it with the instance.

2. Access the VM via SSH:

•Open a terminal on your local machine

•Use the following command to connect to the VM

3. Replace <floating-ip-address> with actual IP add of Instance.

4. Verify Access:

Once connected, you can interact with the VM through terminal.

Result:

Hence virtual m/c using trystack launched succe ssfully.

20/2/25

EXP NO: 08 Date: 20/2/25

8. Install Hadoop single node cluster & run simple applications like word count.

Aim: To install Hadoop single node cluster & run simple applications like word count.

Description: Single Node cluster - It has one Data Node running and setting up all the Name Node, Data Node, resource Manager and Node Manager on a single Machine. This is used for studying and testing purposes. Multi-Node cluster Has more than one data One Node running each data Node is running on different machines.

Prerequisites:

A Linux-based operating system installed on the system or a virtual machine.

Basic understanding of java & Linux commands.

JDK version 8 or higher installed.

Tools Required

Hadoop

Java Development kit.

Step-by-step procedure

Step1: update & Install dependencies

1. update system packages:

Open a terminal & update the system:

sudo apt update && sudo apt upgrade -y

2. Install Java:

Install the Java Development kit

sudo apt install openjdk-11-jdk -y

3. Verify Java Installation:

check the installed java version:

Java -version

output: arduino
copy code
openjdk version "11.x.x"

Step 2: Download and Install Hadoop

1. Download Hadoop

Go to the Apache Hadoop website & download the latest stable version.

Alternatively, as we get to download Hadoop directly

wget https://downloads.apache.org/hadoop/common/hadoop-3.x.x/hadoop-3.x.x.tar.gz

2. Extract Hadoop

Extract the download file to /usr/local:

sudo tar -xvzf hadoop-3.x.x.tar.gz -C /usr/local/ rename the extracted folder for simplicity.

sudo mv /usr/local/hadoop-3.x.x /usr/local/hadoop.

3. set Hadoop environment variables:
Add the following lines to the end of your .bashrc file:

```
export HADOOP_HOME=/usr/local/hadoop
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export JAVA_HOME=/usr/lib/jvm-11-openjdk-amd64
```

Apply changes:

```
source ~/.bashrc
```

Step3: configure Hadoop for single-Node setup

1. edit hadoop-env.sh:
configure the java home directory in hadoop

```
nano $HADOOP_HOME/etc/hadoop/hadoop-env.sh
```

set the following line:

```
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
```

2. edit core-site.xml:
open & edit core-site configuration file

```
nano $HADOOP_HOME/etc/hadoop/core-site.xml
```

Add the following configuration

```
xml
copycode
<configuration>
 <property>
  <name>fs.defaultFS</name>
```

<value> hdfs://localhost:9000 </value>
</property>
</configuration>

3. Edit hdfs-site.xml:
Configure hdfs settings:
nano ${HADOOP_HOME}/etc/hadoop/hdfs-site.xml
Add the following
xml
<configuration>
<property>
<name> dfs.replication </name>
<value>1 </values>
</property>
<property>
<name>dfs.namenode.dir </name>
<value> file:///usr/local/hadoop/tmp/dfs/
name/value>
</property>
<property>
<name> dfs.datanode.data.dir </name>
<value> file:/// local/hadoop/tmp/dfs/data
</value>
</property>
</configuration>

4. format the namenode
format the HDFS Namenode
hdfs namenode -format

step4: start Hadoop services

start-dfs.sh

check the status of Namenode & DataNode using jps

Expected o/p includes:

Namenode

Datanode

Secondary namenode

2start YARN (optional)

start-yarn.sh

check status of resource manager & Node manager using.

step5: Run word count Application

1. create input data

create sample text file for the wordcount program.

echo "Hello Hadoop Hello world" > input.txt

2. upload the input file to HDFS:

copy the file the Hadoop distributed file sys.

hdfs dfs -mkdir /input

hdfs dfs -put input.txt /input.

3. Run word count program

Execute:

hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/

hadoop-mapreduce-examples-3.x.x.jar wordcount

input/output

4. view the output

check the results of the wordcount program

hdfs dfs -cat /output/part-r-000000

executed output:

copycode

Hadoop1

Hello 2

world1

Result: Hence Installation of Hadoop single node and cluster & running simple applications like word count execute successfully.

20.2.25

Date: 13/02/25

ExpNO:09

Aim: To Simulate a secure file-sharing mechanism in a cloud computing environment using the cloudsim framework.

prerequisites:

- Basic knowledge of Java programming.
- cloudsim toolkit downloaded & configured.
- Eclipse IDE or Intellij installed on your system.
- familiarity with cryptographic algorithms.

Tools Required:

* cloudsim Toolkit:- Download from cloudsim github.
* Java Development kit: Ensure JDK 8 higher is installed.
* Eclipse IDE: Java Development environment for coding and simulation.

Step-by-step procedure:

Step1: set up cloudsim.

1. Download cloudsim

* Download the latest version of the cloudsim toolkit
* Extract the downloaded file.

2. Import cloudsim into Eclipse:

* Open eclipse and create a new Java project.
* Add the cloudsim-<version>.jar file and other dependencies to the build path:

->Right - click the project > Buildpath > Add External JARS > select the JAR files.

3. verify the Environment.

* create a test program to ensure cloudsim is set up correctly:

```
java
copy code
import org.cloudbus.cloudsim.core.cloudsim;
public class Test cloudsim {
  public static void main(String[] args){
    System.out.printh("cloudsim Environment is ready");
  }
}
```

Step 2: Design the cloud Environment

1. Initialize cloudsim:

start the simulation Environment:

```
java
copy code
cloudsim.init(numusrs, calender, tracelog);
```

* numusers: NO. of users (e.g:, 1);
* calender: use null for the default calender.
* traceFlag: Boolean to enable/disable tracing.

2. Create a Data center:

Define a data center with hosts for VMs:

```
java
copy code
Datacenter datacenter = create Datacenter("Datacenter-0");
Private static Datacenter create Datacenter (String name){
  List<Host> hostlist = new Arraylist<>();
  int ram = 2048
```

```
long storage =100000;
int bw=10000;
int pesNumber =4;
for(int i=0; i<2;i++) {
 List<pe> peList =new ArrayList<>()
peList.add(new pe(0,new peprovisionersimple(1000)));
hostlist.add (new Host(i,new Ramprovisionersimple
(ram),new Bwprovisionersimple(bw),storage,peList,
new vmschedulerTimeshared(peList)));
}
return new Datacenter(name, hostlist,new VmAllo-
cationpolicysimple(hostlist),new LinkedList<>(),0);
}
```

3. create virtual Machines:

Define VM's to host cloudlets:

```
java
copy code
vm vm1 =new vm(0,brokerId,1000,1,512,1000,1000,"xen",
newcloudlet &schedulertimeshared());
vm vm2 =new vm(1,brokerId,1000,1,512,1000,10000,
"xen",new CloudletSchedulerTime shared());
List<vm> vmlist =new ArrayList<>();
vmlist.add(vm1);
vmlist.add(vm2);
```

4. create cloudlets (tasks);

Step 3: Implement secure file sharing

1. Encryption and Decryption Mechanism:

Simulate encryption before file sharing & decryption after reception.

java
copy code

```
public static string encrypt (string data, string
                                        key) {
  byte[] keyBytes = key.getBytes();
  byte[] dataBytes = data.getBytes();
  byte[] encrypted = new byte[dataBytes.length];
 for(int i=0; i<dataBytes.length; i++){
  encrypt[i] = (byte)(dataBytes[i] ^ keyBytes[i%.keyBy
                                        tes.length]);
 }
 return new string (encrypted);
}
public static string decrypt(string encryptedData,
   string key){
return encrypt(encryptedData, key);
}
```

2. Simulate secure file transfer:

Before executing a cloudlet, encrypt its data, simulate transfer, and decrypt it at the destination:

java
copy code

```
String filedata = "this is a secure file";
String key = "encryption key";
String encryptedData = encrypt (filedata, key);
System.out.println("Encrypted Data "+encryptedData);
String decryptedData = decrypt(encryptedData, key);
System.out.println("Decryptdata "+decryptedData);
```

Step4: Simulate and Execute

1. Submit VMs & cloudlets to broker:

create a broker to manage resource allocation:

java

```
DatacenterBroker broker = new DatacenterBroker("br");
broker.submitVmList(vmList);
broker.submitCloudletList(cloudletList);
```

2. Start Simulation:

Begin the cloudsim simulation

java

copycode

```
cloudsim.startsimulation();
List<Cloudlet> resultList = broker.getCloudletReceived
                                    List();
cloudsim.stopSimulation();
```

3. Print Results:

Display the cloudlet results & verify secure file sharing:

java

```
for (Cloudlet cloudlet : resultList) {
 System.out.println("Cloudlet "+cloudlet.getCloudSet
                                    Id()+"Execute)
```

5. Expected output:

* the simulation will display the encrypted data & decrypted data for secure.
* cloudlet results will show successful execution on assigned VMs.

Example output:

csharp

copy code

Encrypted Data: XrknKs8....

Decrypted Data: this is a secure file

cloudlet 0 executed on VM 0

cloudlet 1 executed on VM 1

Result: Hence simulation of a secure file sharing using cloudsim is executed successfully.

23/4/25