**[ 2CEIT603: CLOUD COMPUTING]**

Practical: 7

**AIM- Configuration of Private cloud (OpenNebulla).**

**Submitted by:** 21012021001\_Adeshara Brijesh



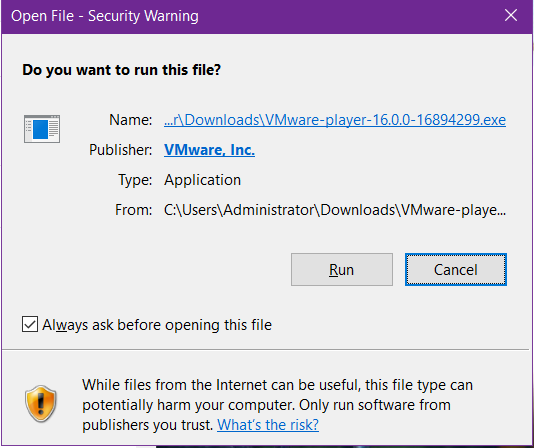
**Department of Computer Engineering/Information Technology**

### Installation of Linux.

* Download ISO image file of Ubuntu 18.04 or more.
* <https://releases.ubuntu.com/18.04/>

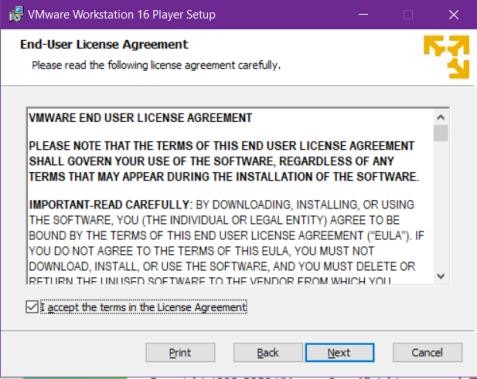
### Installation of VMware.

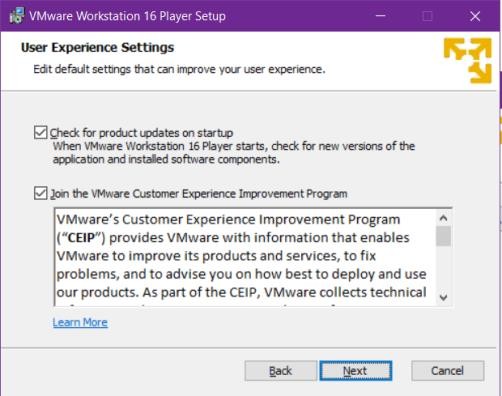
* Step 1: Download VMware software according system requirement.
* Step 2: Run file by double clicking on exe file.



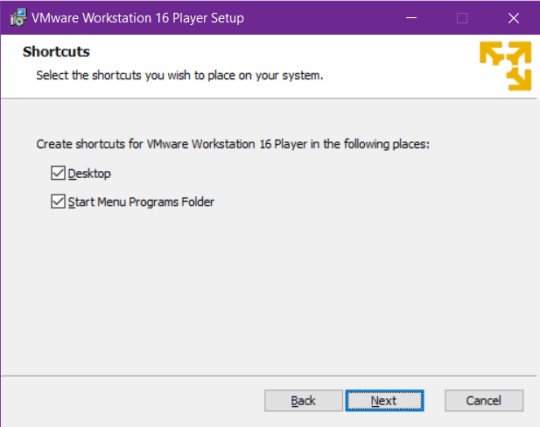
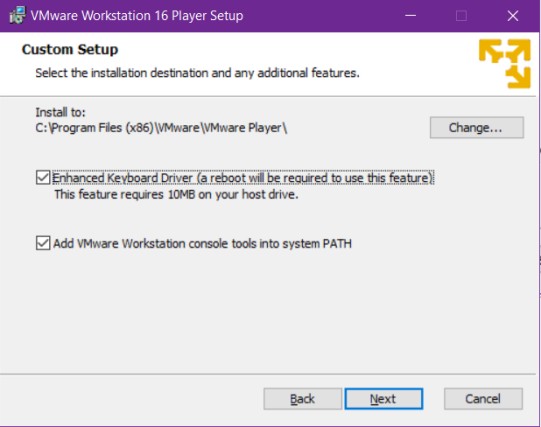
* Step 3: Click on run and setup all environment you needed.

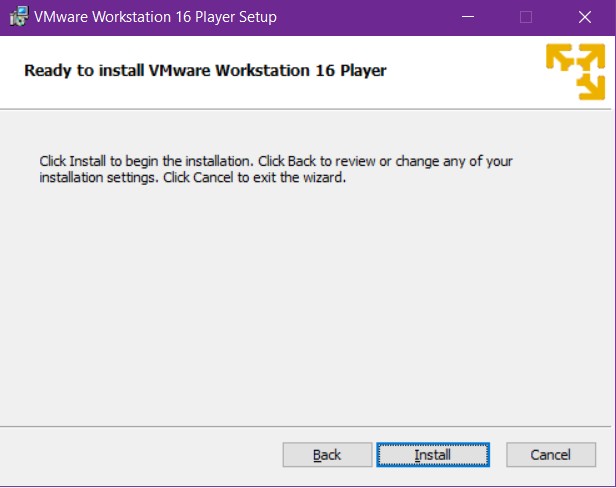


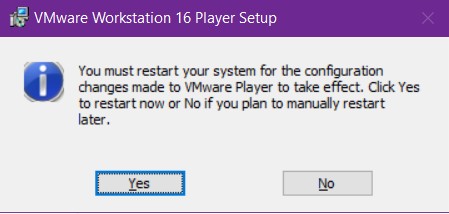




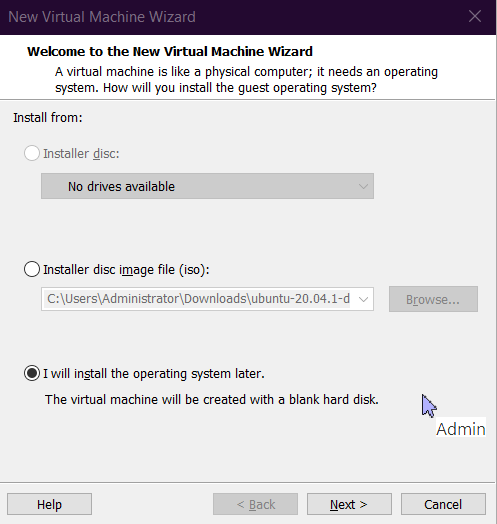
* Step 4:Once you setup all the things click on install

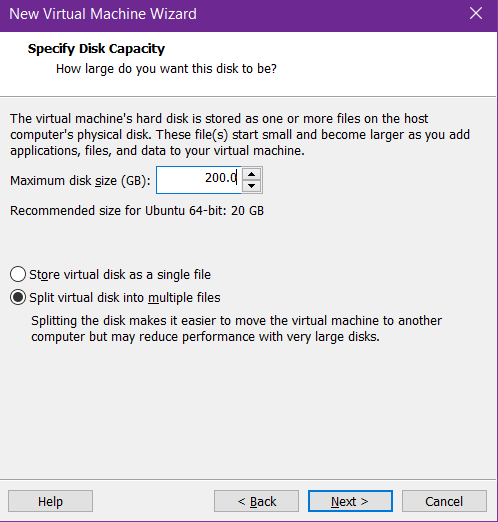
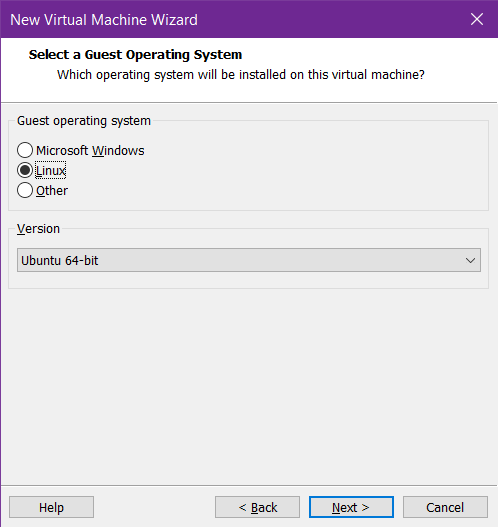




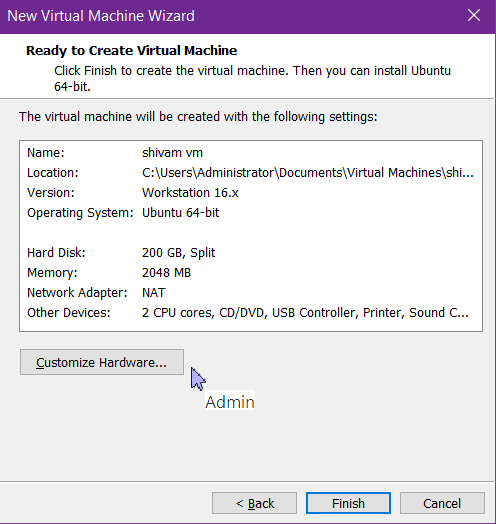
* Step 5: Now, it will install VMware in your system. After, successful installation it will show pop- up to restart your system to setting up all requirement.
* Now, you are ready to create virtual machine on your computer

### Create Virtual Machine

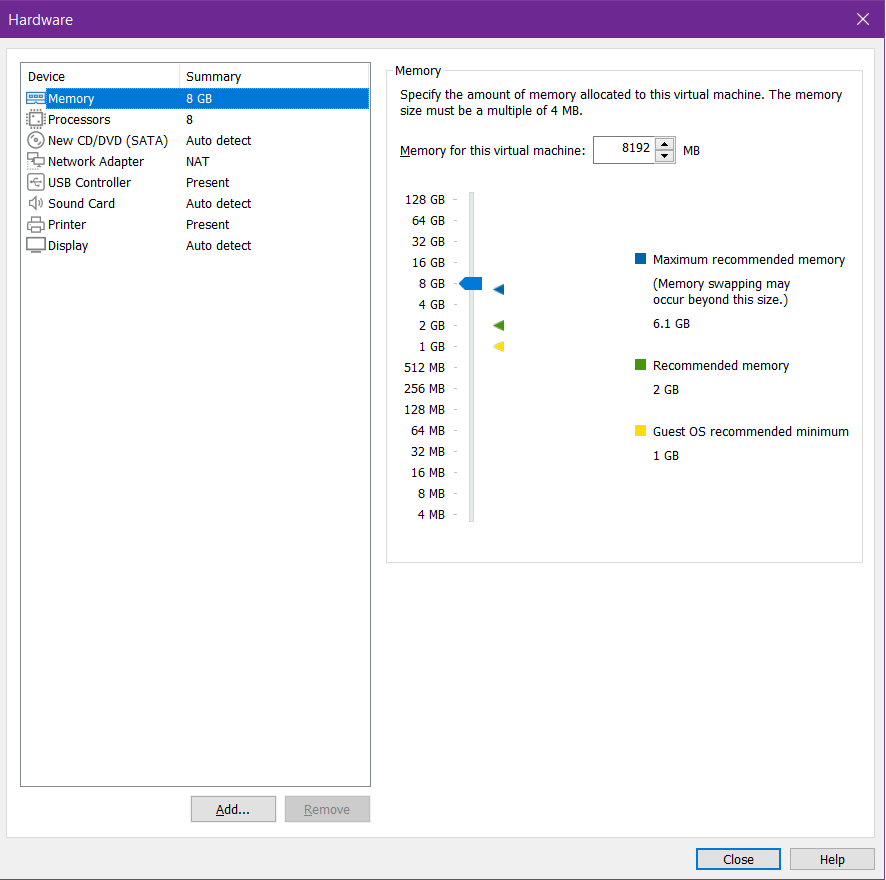
* Step 1: open VMware and click on create virtual device.
* Step 2: select option based on your requirement. Here, we will install OS in future step
* 
* Step 3: Select OS for which you want to create your VM. Here, we want Linux so we will select Linux(Ubuntu).

** **

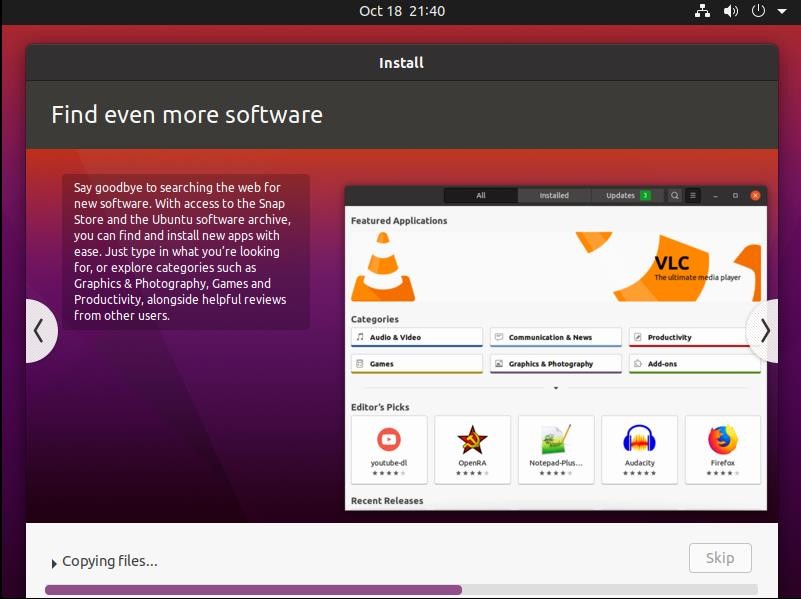
* Step 5: Set-up all your hardware requirement for your VM.



* Step 6: You can go with default hardware and can customize by clicking customize hardware

****

* Step 7:At last select ISO file you have download from internet to set up OS and create VM
* Step 8: Now, it will install or create your VM

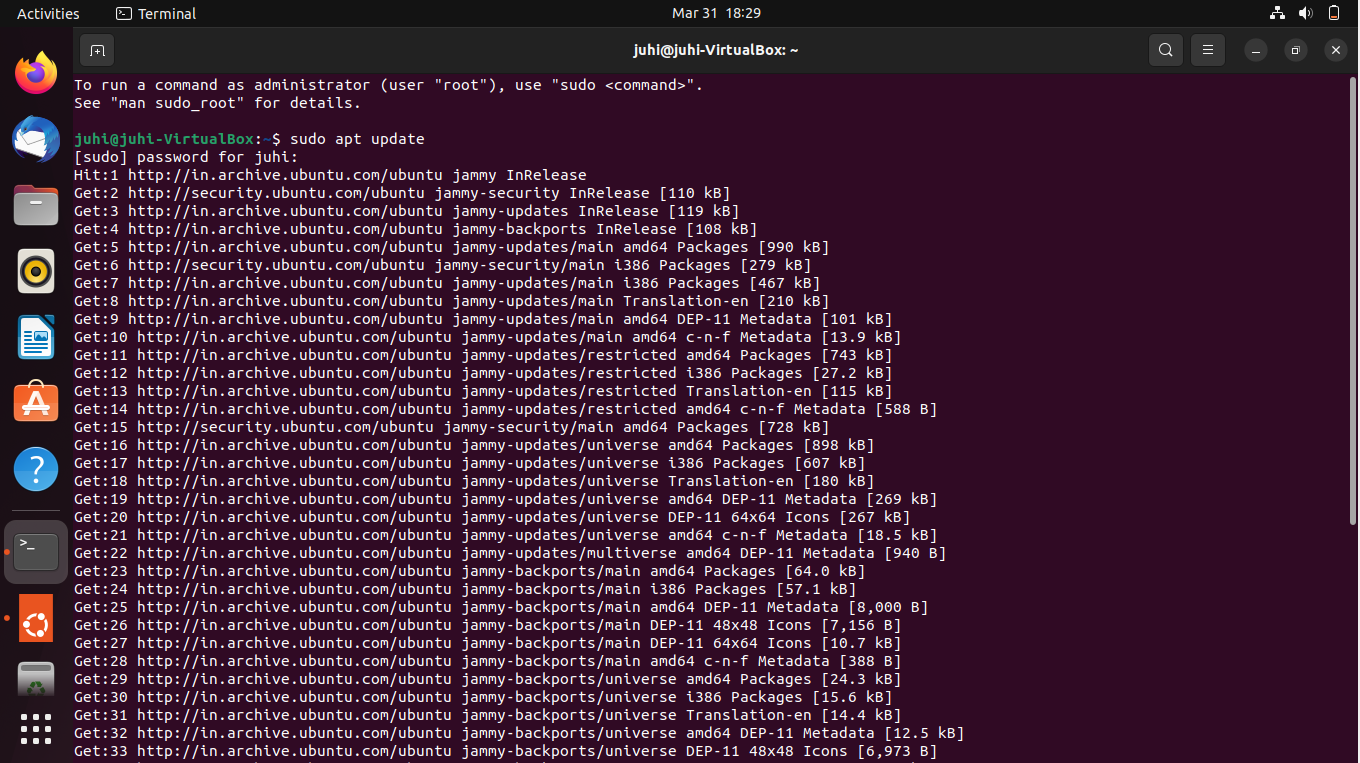


### Step 1: Install MariaDB Database Server

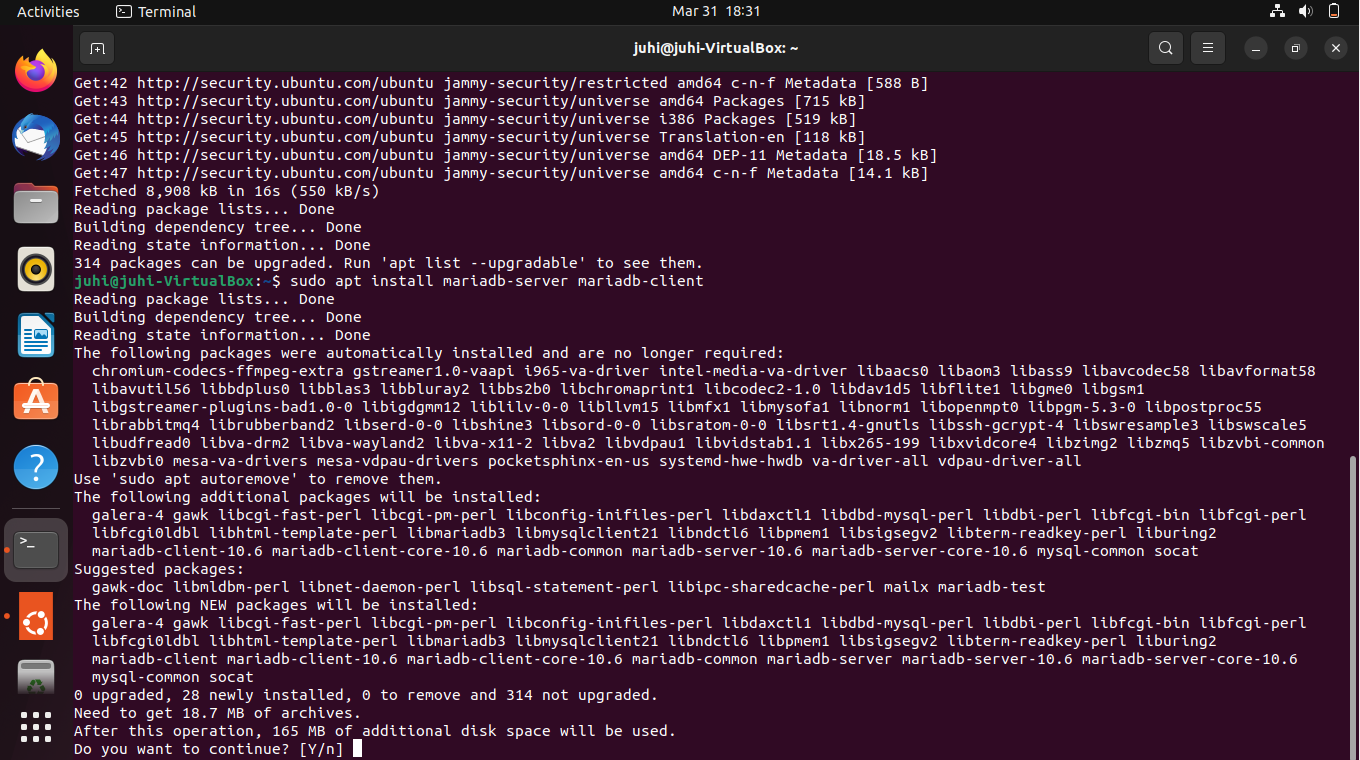
OpenNebula also needs a database server to store its content… and MariaDB database server is a great place to start when looking at open source database servers to use with OpenNebula.

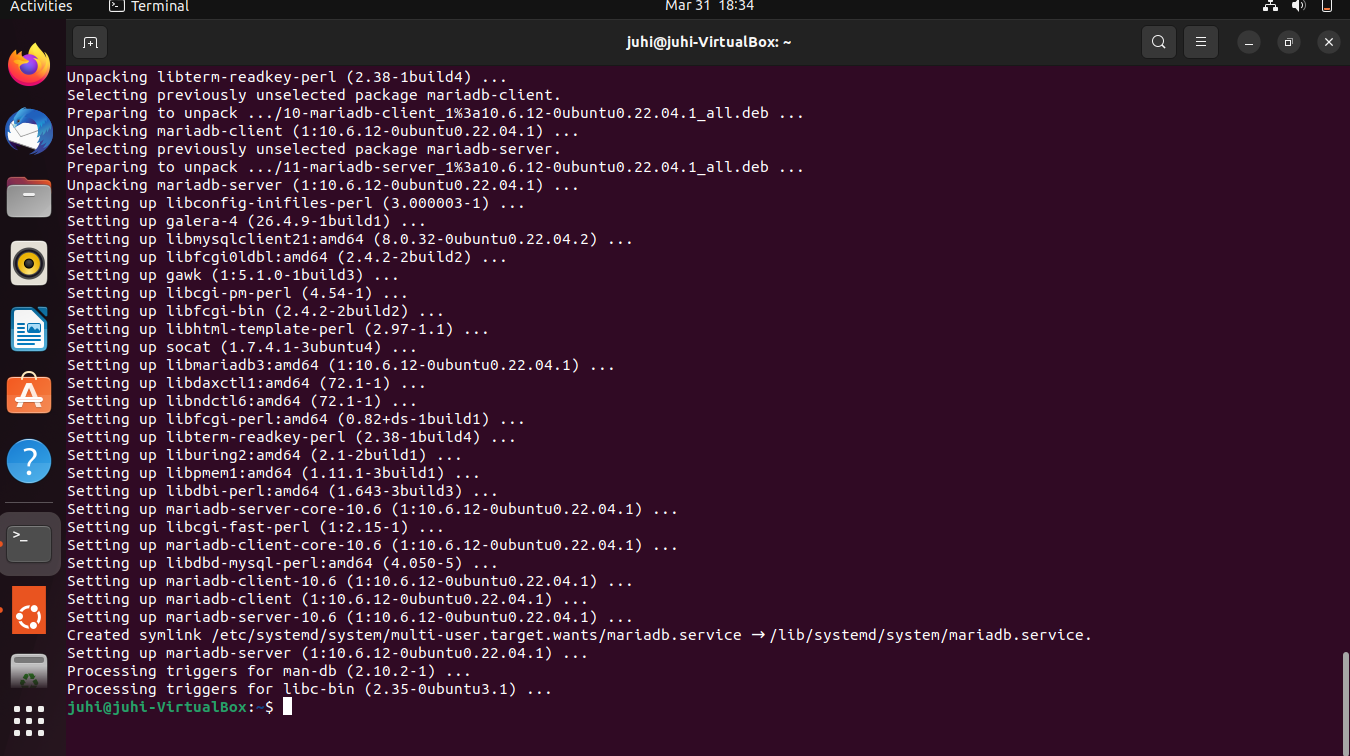
To install MariaDB run the commands below:

* sudo apt update



* sudo apt install mariadb-server mariadb-client

****

****

After installing MariaDB, the commands below can be used to stop, start and enable MariaDB service to always start up when the server boots..

* sudo systemctl stop mariadb.service
* sudo systemctl start mariadb.service
* sudo systemctl enable mariadb.service

After that, run the commands below to secure MariaDB server by creating a root password and disallowing remote root access.

sudo mysql\_secure\_installation

When prompted, answer the questions below by following the guide. Enter current password for root (enter for none): Just press the Enter Set root password? [Y/n]: Y

New password: Enter password

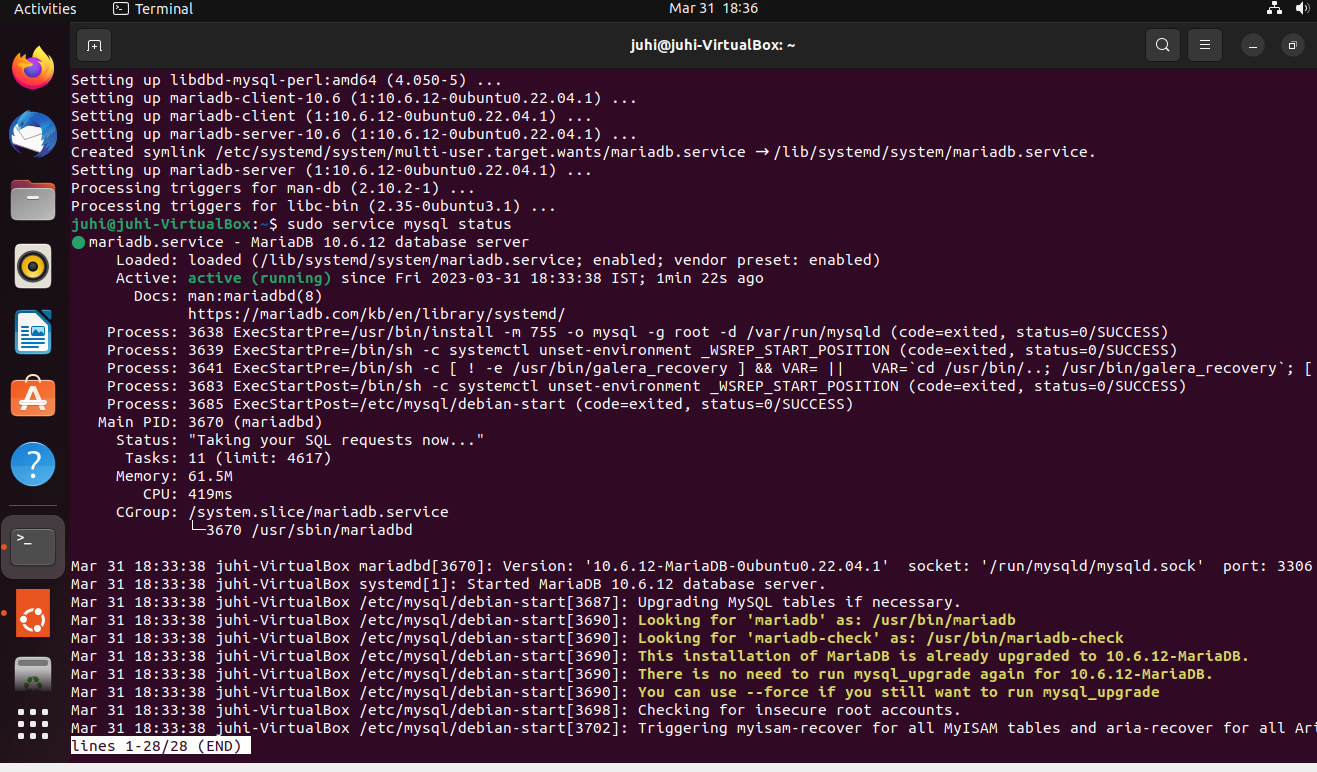
Re-enter new password: Repeat password Remove anonymous users? [Y/n]: Y Disallow root login remotely? [Y/n]: Y

Remove test database and access to it? [Y/n]: Y Reload privilege tables now? [Y/n]: Y

Restart MariaDB server

To test if MariaDB is installed and working, run the commands below:

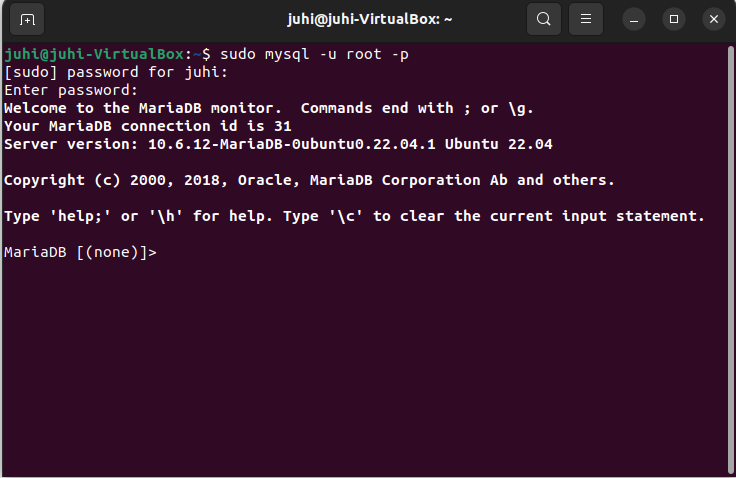
* sudo systemctl status mariadb

****

### Step 2: Create OpenNebula Database

Now that you’ve install all the packages that are required, continue below to start configuring the servers. First create a blank database for OpenNebula to use.

To do that, run the commands below to logon to MariaDB. When prompted for a password, type the root password you created above.

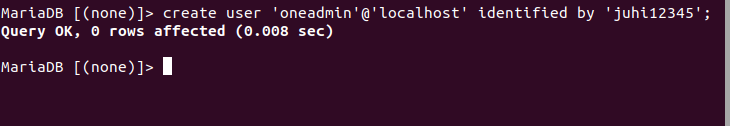
* sudo mysql -u root –p
* 

Then create a database called opennebula

* CREATE DATABASE opennebula;

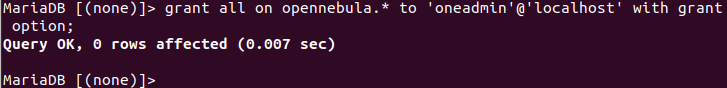
Create a database user called opennebulauser with new password

* CREATE USER 'opennebulauser'@'localhost' IDENTIFIED BY 'new\_password\_here';

****

Next, grant the user full access to the cakephpuser database.

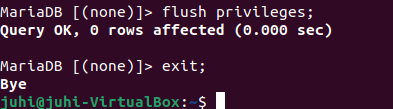
✑ GRANT ALL ON opennebula.\* TO 'opennebulauser'@'localhost' WITH GRANT OPTION;

****

Finally, save your changes and exit.

✑ FLUSH PRIVILEGES;

✑ EXIT;

****

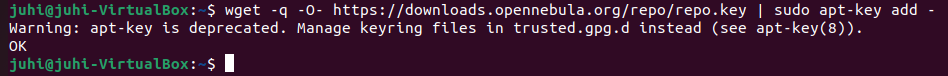
Now that MariaDB server is installed and a database created, now go and install OpenNebula.

### Step 3: Install OpenNebula

By default, OpenNebula isn’t available via Ubuntu default repositories. If you’d like to install it in Ubuntu, you’ll have to install it from its repository.

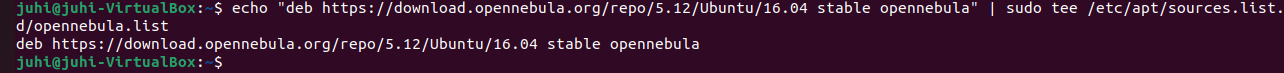
First, run the commands below to add its repository key to Ubuntu

✑ wget -q -O- https://downloads.opennebula.org/repo/repo.key | sudo apt-key add –



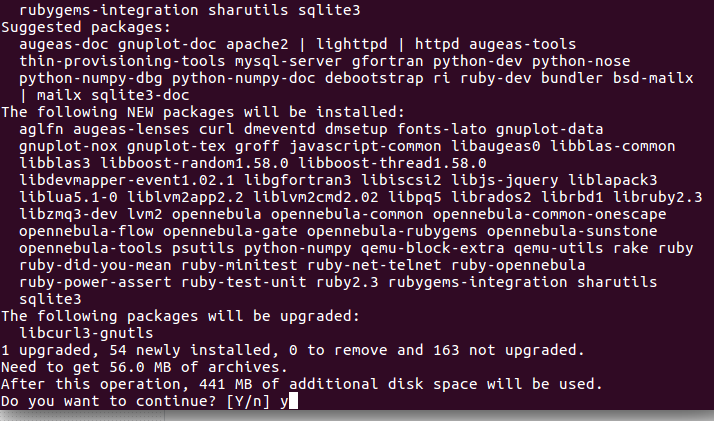
After adding the repository’s key, run the commands below to add its repository file.

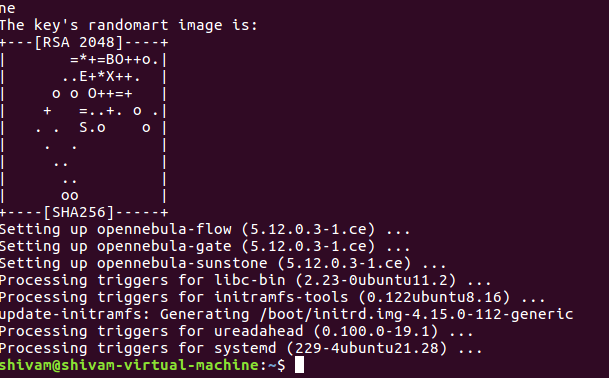
✑ echo "deb https://downloads.opennebula.org/repo/5.12/Ubuntu/18.04 stable opennebula" | sudo tee /etc/apt/sources.list.d/opennebula.list



Now that the repository’s key and file are added, run the commands below to install OpenNebula.

✑ sudo apt update

✑ sudo apt install opennebula opennebula-sunstone opennebula-gate opennebula-flow



When you’re done installing OpenNebula, run the commands below to install Ruby libraries and other required packages to support OpenNebula.

✑ sudo /usr/share/one/install\_gems

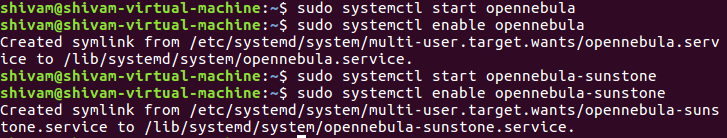
That should install OpenNebula on Ubuntu. After installing, the commands below can be used to start and enable OpenNebula services.

✑ sudo systemctl start opennebula

✑ sudo systemctl enable opennebula

✑ sudo systemctl start opennebula-sunstone

✑ sudo systemctl enable opennebula-sunstone

****

To support MariaDB database server, run the commands below to open its configuration file

✑ sudo nano /etc/one/oned.conf

Then edit the highlighted lines with database name, user and password created above and save the file. SCRIPTS\_REMOTE\_DIR=/var/tmp/one

PORT = 2633

LISTEN\_ADDRESS = "0.0.0.0"

#DB = [ BACKEND = "sqlite" ]

Sample configuration for MySQL

DB = [ BACKEND = "mysql",

SERVER = "localhost",

PORT = 0,

USER = "opennebulauser",

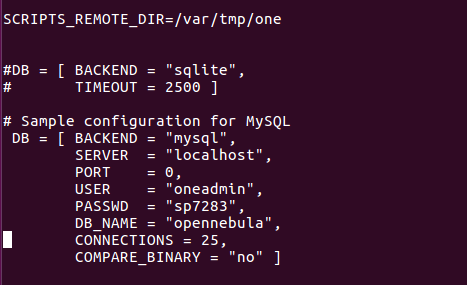
PASSWD = "database\_password\_here",

DB\_NAME = "opennebula",

CONNECTIONS = 50 ]

VNC\_PORTS = [ START = 5900

RESERVED = "6800, 6801, 6810:6820, 9869"]

****

Save (ctrl+o) the file and exit (ctrl+x).

To access OpenNebula portal, you’ll need a credential. By default, its default username and password is stored in the credential below:

✑ sudo nano /var/lib/one/.one/one\_auth



Edit it and replace the username and password that suits you.

After that, run the commands below to check the status of OpenNebula services.

# ✑ sudo systemctl status opennebula

# You should get similar lines as below:

That shows the service is running.

Another Way for installation of open nebula :- Install opennebula-4.14.2-sandbox.vdi File.

To install the vdi file :- <https://drive.google.com/file/d/1HvFDzcFrgbTmRQQzjJ4qjxbl_pRKPlG9/view?usp=drive_link>

### Step 4: Access OpenNebula Portal

Now that OpenNebula is installed, open your web browser ad browse to the server hostname or IP address followed by port 9869

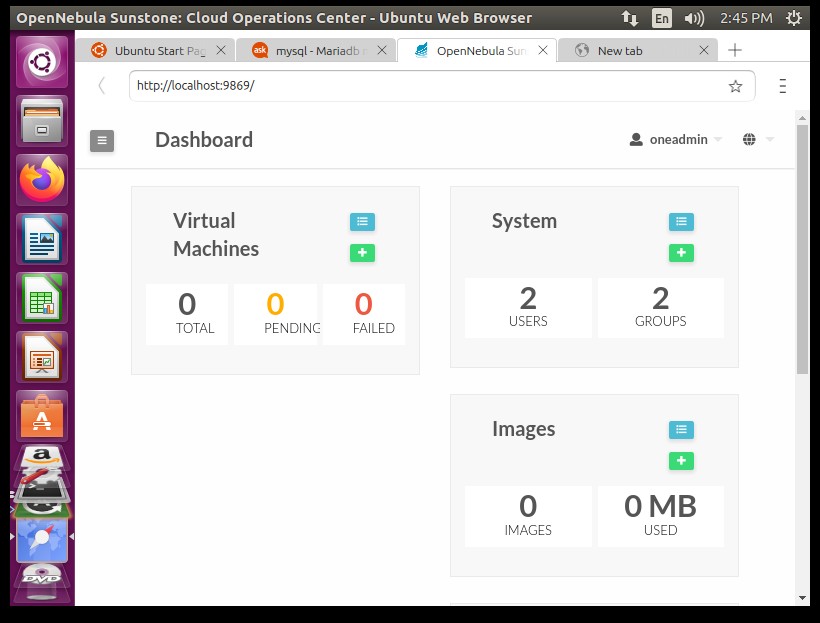
http://localhost:9869

# 

Login with the credential found in the file above.

# 

That should redirect you to OpenNebula dashboard.



That’s it! Your platform is ready, but you’ll need to add VMware or KVM nodes to manage. Setup KVM Node

### Step 5: Configure KVM node

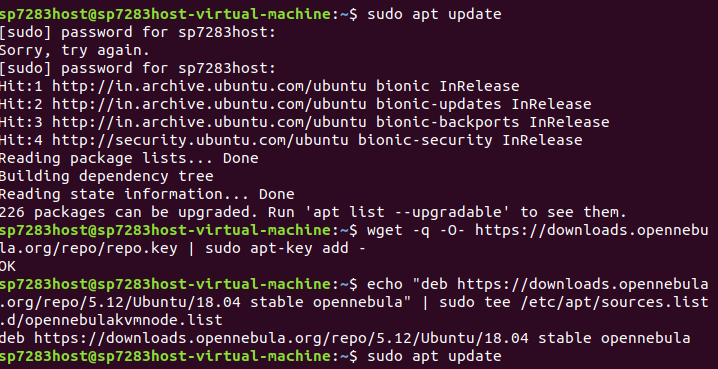
Now, that OpenNebula platform is installed and ready, you’ll need to add nodes to manage. To add KVM nodes, simply run the commands below node you want to manage via OpenNebula.

wget -q -O- https://downloads.opennebula.org/repo/repo.key | sudo apt-key add - Then add its repository

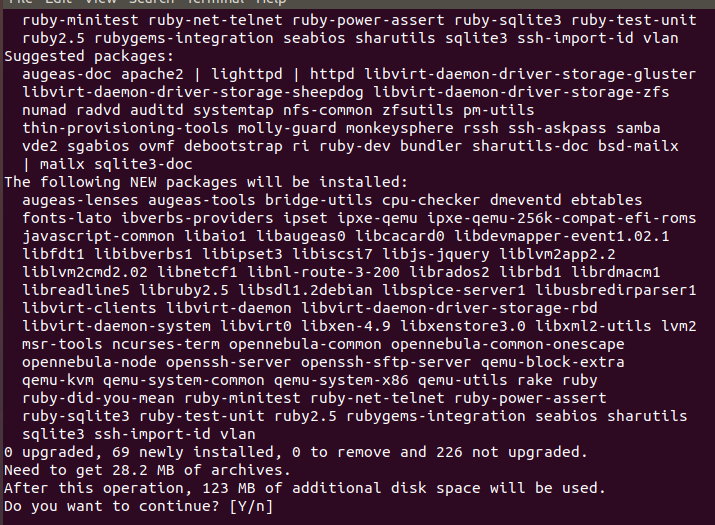
✑ echo "deb https://downloads.opennebula.org/repo/5.12/Ubuntu/18.04 stable opennebula" | sudo tee /etc/apt/sources.list.d/opennebulakvmnode.list

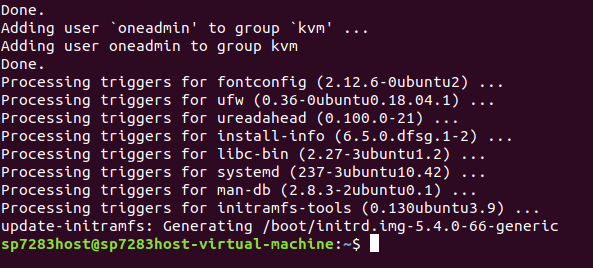
Finally, run the commands below to install the node package

✑ sudo apt update



sudo apt-get install opennebula-nod





After installing the node package, open its configuration file to allow OpenNebula to manage it.

✑ sudo nano /etc/libvirt/libvirtd.conf

Then edits these lines and save.

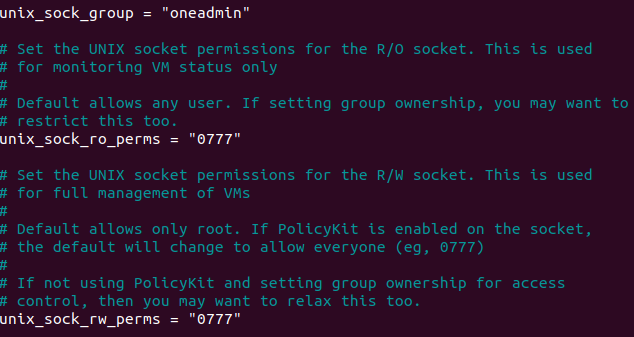
This is restricted to'root' by default.

unix\_sock\_group = "oneadmin"

If not using PolicyKit and setting group ownership for access control,

then you may want to relax this too.

unix\_sock\_rw\_perms = "0777"



Save the file and exit.

Restart the client services

✑ sudo systemctl restart libvirtd

✑ sudo systemctl restart libvirt-bin

Communication is done via SSH. So you’ll want install SSH and enable password-less SSH authentication between the OpenNebula host and the KVM node.

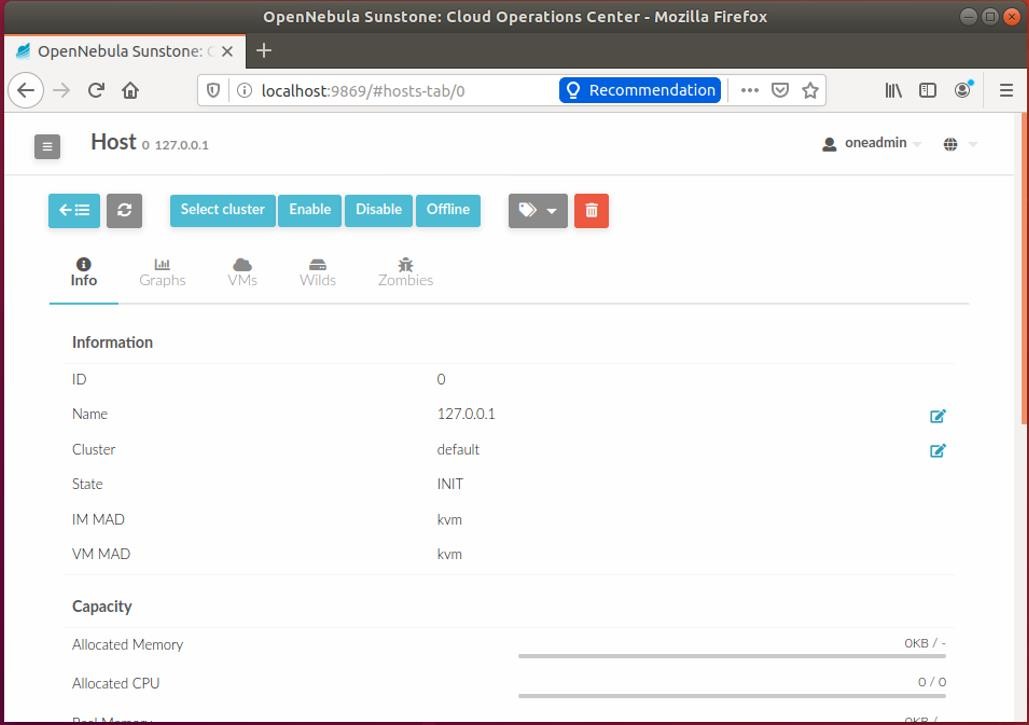
After creating a password authentication, copy the server SSH key to the client known\_hosts file at.

* /var/lib/one/.ssh/known\_hosts

All keys in there will be trusted.

To manage a KVM host, go to OpenNebula web interface, open Infrastructure >> Hosts and Click on the + button.

Add a new node to manage.



**Step 6: Configure Passwordless SSH**

OpenNebula Front-end connects to the hypervisor Hosts using SSH. You must distribute the public key of oneadmin user from all machines to the file /var/lib/one/.ssh/authorized\_keys in all the machines. There are many methods to achieve the distribution of the SSH keys, ultimately the administrator should choose a method (the recommendation is to use a configuration management system). In this guide we are going to manually scp the SSH keys.

When the package was installed in the Front-end, an SSH key was generated and

the authorized\_keys populated. We will sync the id\_rsa, id\_rsa.pub and authorized\_keys from the Front-end to the nodes. Additionally we need to create a known\_hosts file and sync it as well to the

nodes.

To create the known\_hosts file, we have to execute this command as user oneadmin in the Front-end with all the node names as parameters:

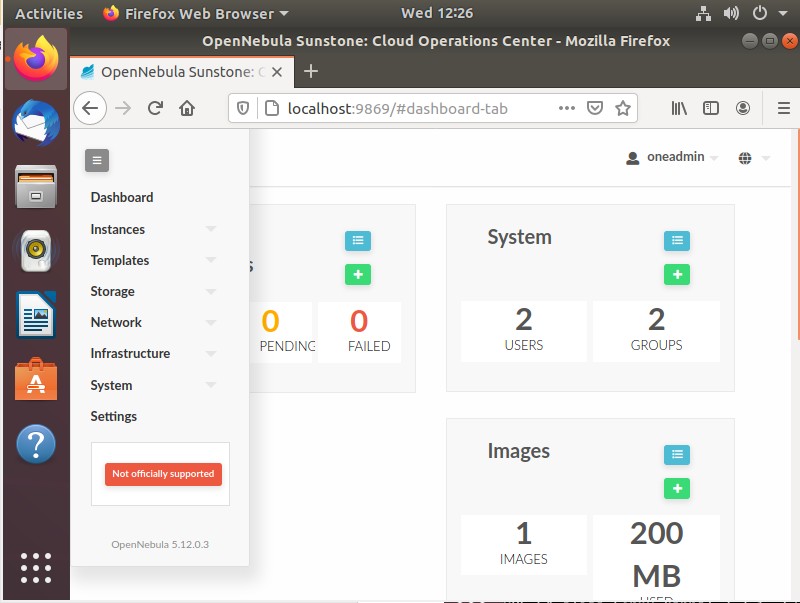
* ssh-keyscan <node1> <node2> <node3> ... >> /var/lib/one/.ssh/known\_hosts

Now we need to copy the directory /var/lib/one/.ssh to all the nodes. The easiest way is to set a temporary password to oneadmin in all the hosts and copy the directory from the Front-end:

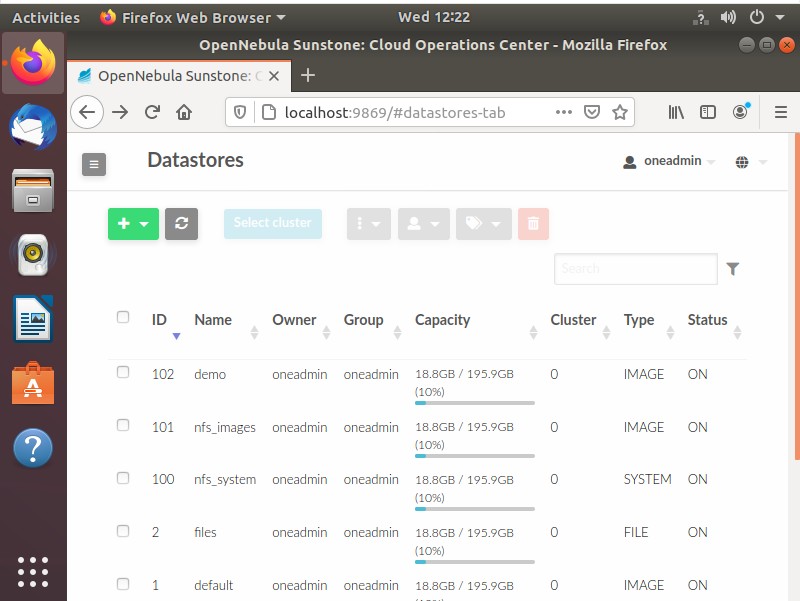
* scp -rp /var/lib/one/.ssh <node1>:/var/lib/one/

You should verify that connecting from the Front-end, as user oneadmin, to the nodes, and from the nodes to the Front-end, does not ask password:

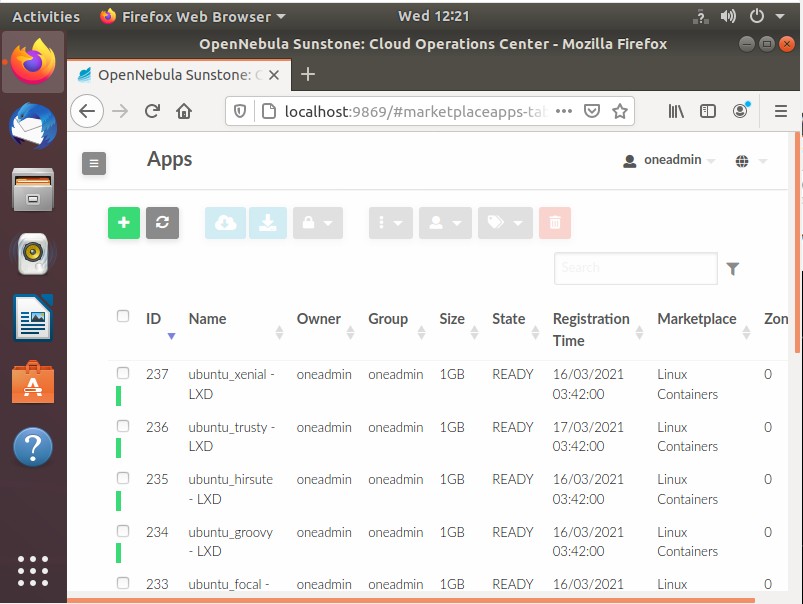
* ssh <node1>
* ssh <frontend>
* exit
* On dashboard you will see the menu bar from that you can see your datastores from your Storage menu bar.



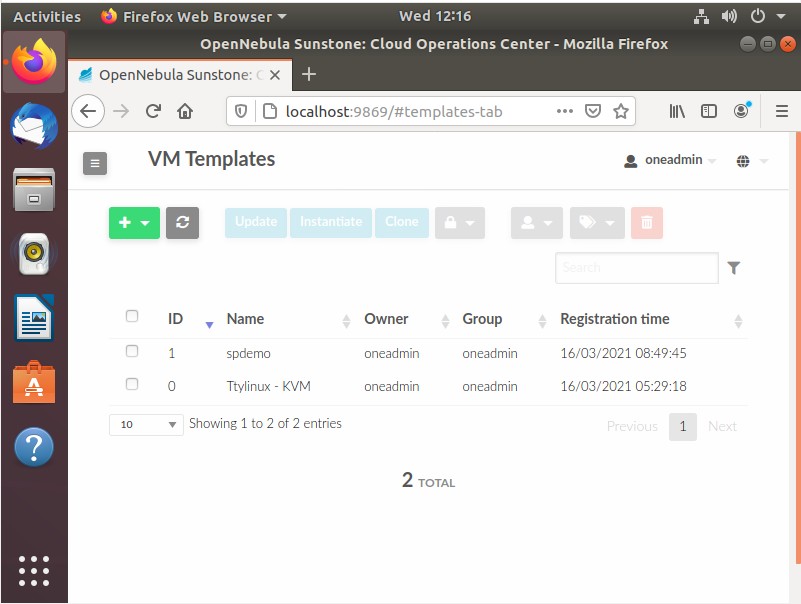
* You must have at least one datastore that can store your data you can create one from CLI as well as from Sunstone.



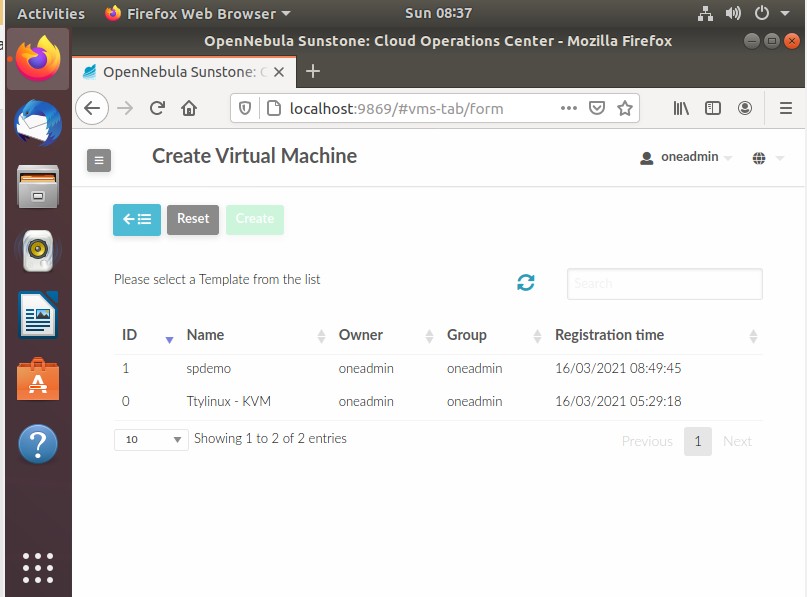
* Now, we have to add the apps to be run on our VMs. You can download the image or add to your private cloud by selecting one.



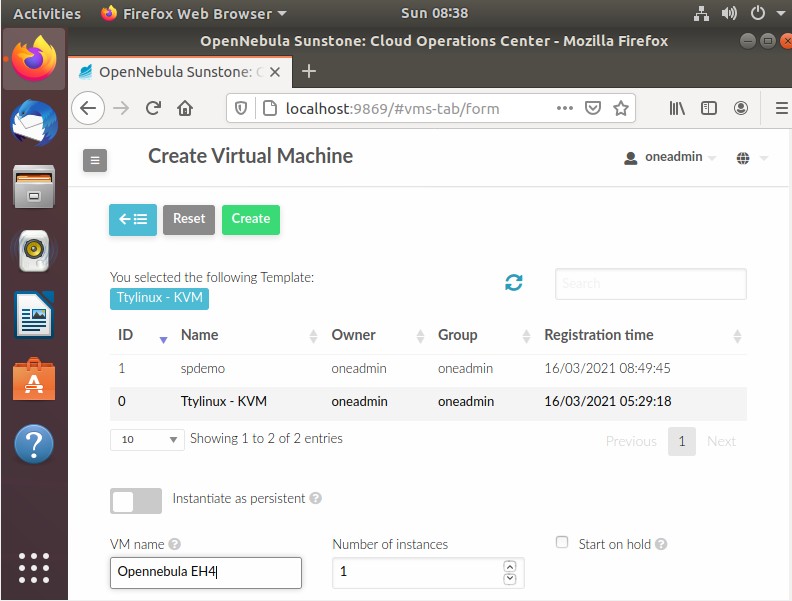
* In VM Templates you can see what apps you have added in your cloud and else you may create your custom templates from the same.



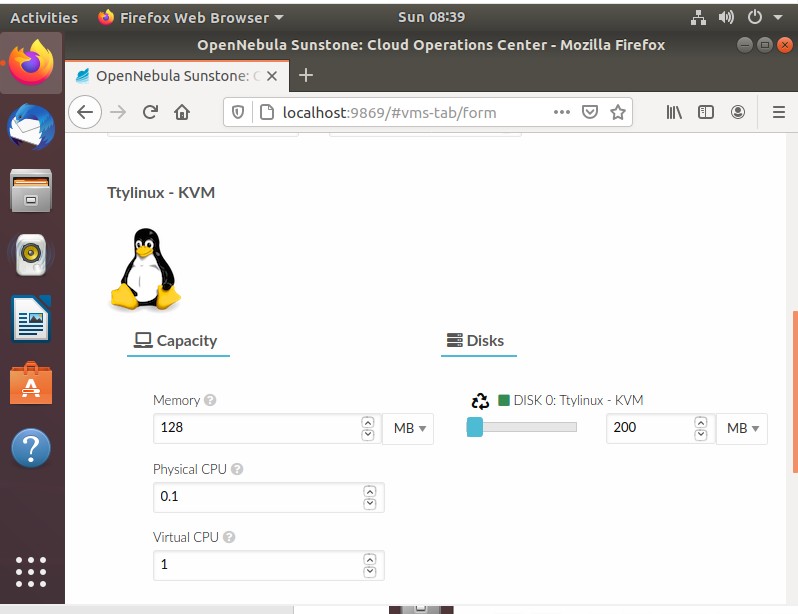
* We are done with all the thing. Now, we are good to create VM. Click on create VM.



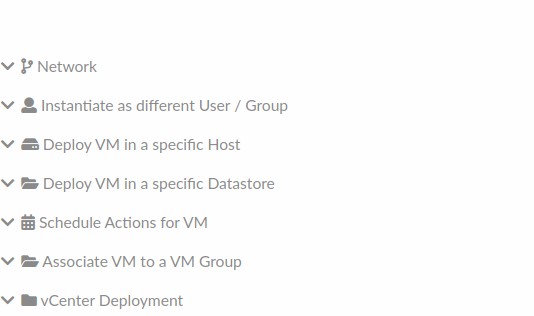
* Once, you click on create VM you have to select the app for which you wish to create VM.



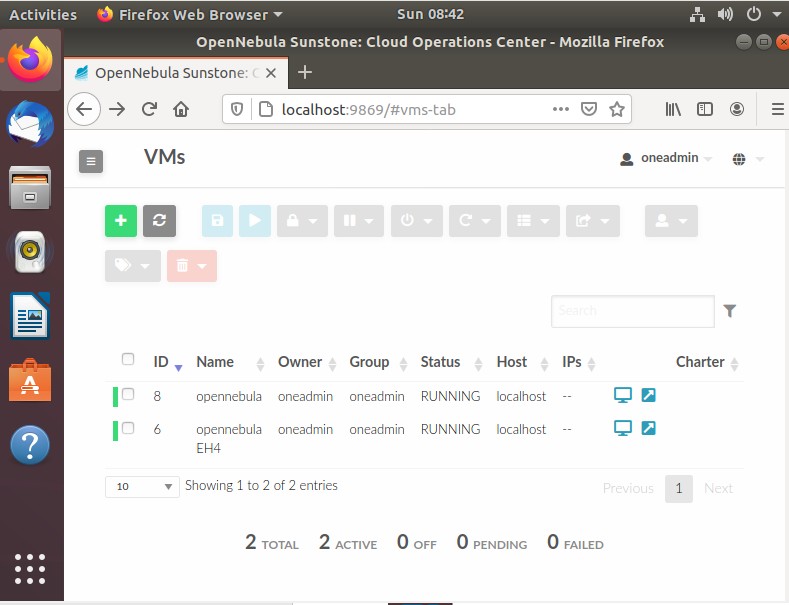
* Fill the other details. Remember Physical CPU value must be float and Virtual CPU value must be integer.



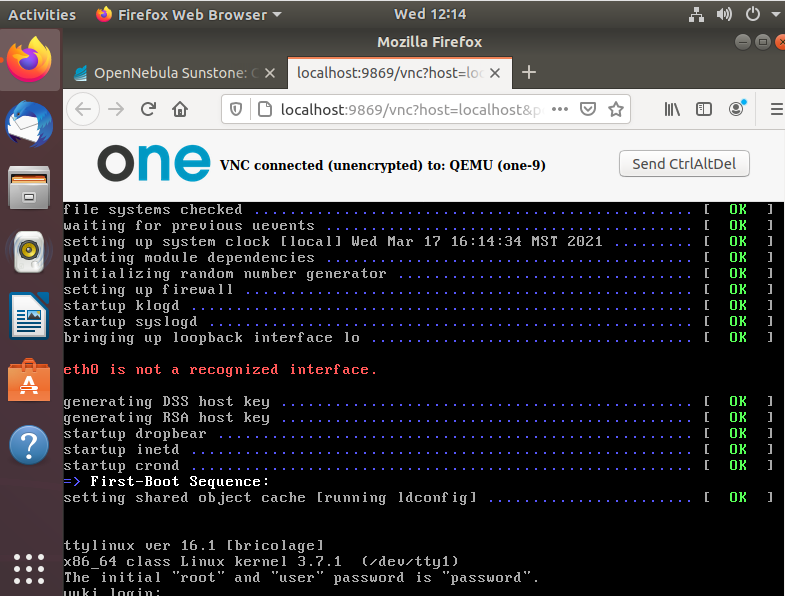
* From the same page you have many option you can select and change like on which datastore you wish to store your data, user, group and many more…



* Click on create now you must have this screen and all your machine must be on Running State.



* When you run machine your machine must be started on browser.



### Conclusion:

Here, using open-nebula we have created our private cloud and implemented IaaS service on which we can run machine.