OpenNebula.org

OpenNebula 4.6 Quickstart Ubuntu 12.04 and KVM

Release 4.6

OpenNebula Project

May 09, 2014

The purpose of this guide is to provide users with step by step guide to install OpenNebula using Ubuntu 14.04 as the operating system and KVM as the hypervisor.

After following this guide, users will have a working OpenNebula with graphical interface (Sunstone), at least one hypervisor (host) and a running virtual machines. This is useful at the time of setting up pilot clouds, to quickly test new features and as base deployment to build a large informations°structure.

Throughout the installation there are two separate roles: **Frontend** and **Nodes**. The Frontend server will execute the OpenNebula services, and the Nodes will be used to execute virtual machines. Please not that **it is possible** to follow this guide with just one host combining both the Frontend and Nodes roles in a single server. However it is recommended execute virtual machines in hosts with virtualization extensions. To test if your host supports virtualization extensions, please run:

```
grep -E 'svm|vmx' /proc/cpuinfo
```

If you don't get any output you probably don't have virtualization extensions supported/enabled in your server.

ONE

PACKAGE LAYOUT

- **opennebula-common**: Provides the user and common files
- **libopennebula-ruby**: All ruby libraries
- **opennebula-node**: Prepares a node as an opennebula-node
- **opennebula-sunstone**: OpenNebula Sunstone Web Interface
- **opennebula-tools**: Command Line interface
- **opennebula-gate**: Gate server that enables communication between VMs and OpenNebula
- **opennebula-flow**: Manages services and elasticity
- **opennebula**: OpenNebula Daemon

TWO

STEP 1. INSTALLATION IN THE FRONTEND

> **Warning:** Commands prefixed by # are meant to be run as `root`. Commands prefixed by $ must be run as `oneadmin`.

2.1 1.1. Install the repo

Add the OpenNebula repository:

```
# wget -q -O- http://downloads.opennebula.org/repo/Ubuntu/repo.key | apt-key add -
# echo "deb http://downloads.opennebula.org/repo/Ubuntu/14.04 stable opennebula" \
    > /etc/apt/sources.list.d/opennebula.list
```

2.2 1.2. Install the required packages

```
# apt-get update
# apt-get install opennebula opennebula-sunstone nfs-kernel-server
```

2.3 1.3. Configure and Start the services

There are two main processes that must be started, the main OpenNebula daemon: `oned`, and the graphical user interface: `sunstone`.

`Sunstone` listens only in the loopback interface by default for security reasons. To change it edit `/etc/one/sunstone-server.conf` and change `:host:  127.0.0.1` to `:host:  0.0.0.0`.

Now we must restart the Sunstone:

```
# /etc/init.d/opennebula-sunstone restart
```

2.4 1.4. Configure NFS

> **Warning:** Skip this section if you are using a single server for both the frontend and worker node roles.

Export `/var/lib/one/` from the frontend to the worker nodes. To do so add the following to the `/etc/exports` file in the frontend:

```
/var/lib/one/ *(rw,sync,no_subtree_check,root_squash)
```

Refresh the NFS exports by doing:

```
# service nfs-kernel-server restart
```

2.5 1.5. Configure SSH Public Key

OpenNebula will need to SSH passwordlessly from any node (including the frontend) to any other node.

To do so run the following commands:

```
# su - oneadmin
$ cp ~/.ssh/id_rsa.pub ~/.ssh/authorized_keys
```

Add the following snippet to `~/.ssh/config` so it doesn't prompt to add the keys to the `known_hosts` file:

```
$ cat << EOT > ~/.ssh/config
Host *
    StrictHostKeyChecking no
    UserKnownHostsFile /dev/null
EOT
$ chmod 600 ~/.ssh/config
```

STEP 2. INSTALLATION IN THE NODES

3.1 2.1. Install the repo

Add the OpenNebula repository:

```
# wget -q -O- http://downloads.opennebula.org/repo/Ubuntu/repo.key | apt-key add -
# echo "deb http://downloads.opennebula.org/repo/Ubuntu/14.04 stable opennebula" > \
    /etc/apt/sources.list.d/opennebula.list
```

3.2 2.2. Install the required packages

```
# apt-get update
# apt-get install opennebula-node nfs-common bridge-utils
```

3.3 2.3. Configure the Network

> **Warning:** Backup all the files that are modified in this section before making changes to them.

You will need to have your main interface, typically `eth0`, connected to a bridge. The name of the bridge should be the same in all nodes.

If you were using DHCP for your `eth0` interface, replace `/etc/network/interfaces` with:

```
auto lo
iface lo inet loopback

auto br0
iface br0 inet dhcp
        bridge_ports eth0
        bridge_fd 9
        bridge_hello 2
        bridge_maxage 12
        bridge_stp off
```

If you were using a static IP addresses instead, use this other template:

```
auto lo
iface lo inet loopback
```

```
auto br0
iface br0 inet static
        address 192.168.0.10
        network 192.168.0.0
        netmask 255.255.255.0
        broadcast 192.168.0.255
        gateway 192.168.0.1
        bridge_ports eth0
        bridge_fd 9
        bridge_hello 2
        bridge_maxage 12
        bridge_stp off
```

After these changes, restart the network:

```
# /etc/init.d/networking restart
```

3.4 2.4. Configure NFS

> **Warning:** Skip this section if you are using a single server for both the frontend and worker node roles.

Mount the datastores export. Add the following to your `/etc/fstab`:

```
192.168.1.1:/var/lib/one/  /var/lib/one/  nfs   soft,intr,rsize=8192,wsize=8192,noauto
```

> **Warning:** Replace `192.168.1.1` with the IP of the frontend.

Mount the NFS share:

```
# mount /var/lib/one/
```

If the above command fails or hangs, it could be a firewall issue.

3.5 2.5. Configure Qemu

The `oneadmin` user must be able to manage libvirt as root:

```
# cat << EOT > /etc/libvirt/qemu.conf
user  = "oneadmin"
group = "oneadmin"
dynamic_ownership = 0
EOT
```
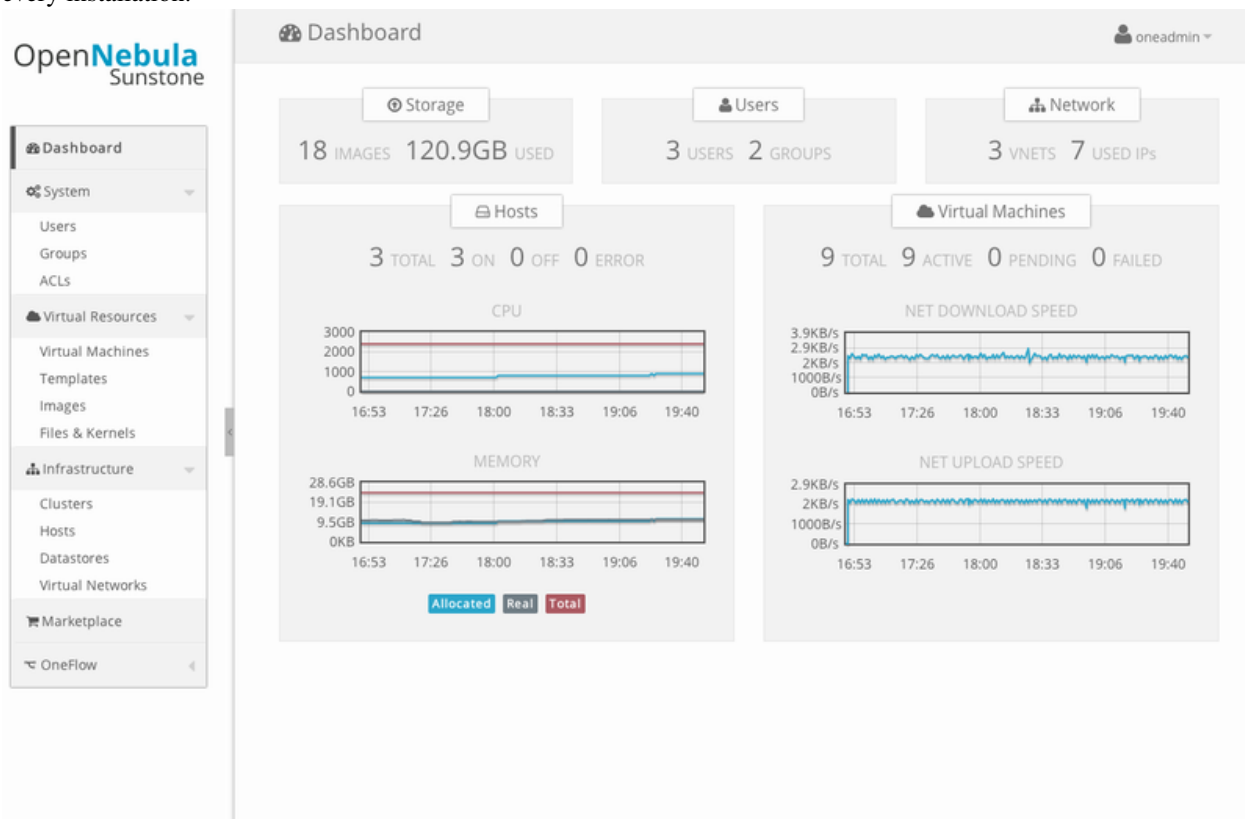
Restart libvirt to capture these changes:

```
# service libvirt-bin restart
```

STEP 3. BASIC USAGE

> **Warning:** All the operations in this section can be done using Sunstone instead of the command line. Point your browser to: `http://frontend:9869`.

The default password for the `oneadmin` user can be found in `~/.one/one_auth` which is randomly generated on every installation.

To interact with OpenNebula, you have to do it from the `oneadmin` account in the frontend. We will assume all the following commands are performed from that account. To login as `oneadmin` execute `su - oneadmin`.

4.1 3.1. Adding a Host

To start running VMs, you should first register a worker node for OpenNebula.

Issue this command for each one of your nodes. Replace `localhost` with your node's hostname.

```
$ onehost create localhost -i kvm -v kvm -n dummy
```

Run `onehost list` until it's set to on. If it fails you probably have something wrong in your ssh configuration. Take a look at `/var/log/one/oned.log`.

4.2 3.2. Adding virtual resources

Once it's working you need to create a network, an image and a virtual machine template.

To create networks, we need to create first a network template file `mynetwork.one` that contains:

```
NAME = "private"
TYPE = FIXED

BRIDGE = br0

LEASES = [ IP=192.168.0.100 ]
LEASES = [ IP=192.168.0.101 ]
LEASES = [ IP=192.168.0.102 ]
```

> **Warning:** Replace the leases with free IPs in your host's network. You can add any number of leases.

Now we can move ahead and create the resources in OpenNebula:

```
$ onevnet create mynetwork.one

$ oneimage create --name "CentOS-6.5_x86_64" \
    --path "http://appliances.c12g.com/CentOS-6.5/centos6.5.qcow2.gz" \
    --driver qcow2 \
    --datastore default

$ onetemplate create --name "CentOS-6.5" --cpu 1 --vcpu 1 --memory 512 \
    --arch x86_64 --disk "CentOS-6.5_x86_64" --nic "private" --vnc \
    --ssh
```

You will need to wait until the image is ready to be used. Monitor its state by running `oneimage list`.

In order to dynamically add ssh keys to Virtual Machines we must add our ssh key to the user template, by editing the user template:

```
$ EDITOR=vi oneuser update oneadmin
```

Add a new line like the following to the template:

```
SSH_PUBLIC_KEY="ssh-dss AAAAB3NzaC1kc3MAAACBANBWTQmm4Gt..."
```

Substitute the value above with the output of `cat ~/.ssh/id_dsa.pub`.

4.3 3.3. Running a Virtual Machine

To run a Virtual Machine, you will need to instantiate a template:

```
$ onetemplate instantiate "CentOS-6.5" --name "My Scratch VM"
```

Execute `onevm list` and watch the virtual machine going from PENDING to PROLOG to RUNNING. If the vm fails, check the reason in the log: `/var/log/one/<VM_ID>/vm.log`.

FIVE

FURTHER INFORMATION

- *Planning the Installation*
- *Installing the Software*
- FAQs. Good for troubleshooting
- *Main Documentation*