OpenNebula.org

OpenNebula 4.6 Quickstart CentOS 6 and Xen

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 CentOS 6 as the operating system and Xen 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 infrastructure.

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.

> **Warning:** In order to avoid problems, we recommend to disable SELinux in all the nodes, **Frontend** and **Nodes**:
>
> ```
> # vi /etc/sysconfig/selinux
> ...
> SELINUX=disabled
> ...
>
> # setenforce 0
> # getenforce
> Permissive
> ```

PACKAGE LAYOUT

- opennebula-server: OpenNebula Daemons

- opennebula: OpenNebula CLI commands

- opennebula-sunstone: OpenNebula's web GUI

- opennebula-java: OpenNebula Java API

- opennebula-node-kvm: Installs dependencies required by OpenNebula in the nodes

- opennebula-gate: Send information from Virtual Machines to OpenNebula

- opennebula-flow: Manage OpenNebula Services

- opennebula-context: Package for OpenNebula Guests

Additionally `opennebula-common` and `opennebula-ruby` exist but they're intended to be used as dependencies. `opennebula-occi`, which is RESTful service to manage the cloud, is included in the `opennebula-sunstone` package.

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

Enable the EPEL repo:

```
# yum install http://dl.fedoraproject.org/pub/epel/6/i386/epel-release-6-8.noarch.rpm
```

Add the OpenNebula repository:

```
# cat << EOT > /etc/yum.repos.d/opennebula.repo
[opennebula]
name=opennebula
baseurl=http://downloads.opennebula.org/repo/CentOS/6/stable/x86_64
enabled=1
gpgcheck=0
EOT
```

2.2 1.2. Install the required packages

A complete install of OpenNebula will have at least both `opennebula-server` and `opennebula-sunstone` package:

```
# yum install opennebula-server opennebula-sunstone
```

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 can start the services:

```
# service opennebula start
# service opennebula-sunstone start
```

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 rpcbind restart
# service nfs 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.

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

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

STEP 2. INSTALLATION IN THE NODES

> **Warning:** The process to install Xen might change in the future. Please refer to the CentOS documenation on Xen4 CentOS6 QuickStart if any of the following steps do not work.

3.1 2.1. Install the repo

Add the CentOS Xen repo:

```
# yum install centos-release-xen
```

Add the OpenNebula repository:

```
# cat << EOT > /etc/yum.repos.d/opennebula.repo
[opennebula]
name=opennebula
baseurl=http://downloads.opennebula.org/repo/CentOS/6/stable/x86_64
enabled=1
gpgcheck=0
EOT
```

3.2 2.2. Install the required packages

```
# yum install opennebula-common xen nfs-utils ruby
```

Enable the Xen kernel by doing:

```
# /usr/bin/grub-bootxen.sh
```

Disable `xend` since it is a deprecated interface:

```
# chkconfig xend off
```

Now you must **reboot** the system in order to start with a Xen kernel.

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.

To do so, substitute `/etc/sysconfig/network-scripts/ifcfg-eth0` with:

```
DEVICE=eth0
BOOTPROTO=none
NM_CONTROLLED=no
ONBOOT=yes
TYPE=Ethernet
BRIDGE=br0
```

And add a new `/etc/sysconfig/network-scripts/ifcfg-br0` file.

If you were using DHCP for your `eth0` interface, use this template:

```
DEVICE=br0
TYPE=Bridge
ONBOOT=yes
BOOTPROTO=dhcp
NM_CONTROLLED=no
```

If you were using a static IP address use this other template:

```
DEVICE=br0
TYPE=Bridge
IPADDR=<YOUR_IPADDRESS>
NETMASK=<YOUR_NETMASK>
ONBOOT=yes
BOOTPROTO=static
NM_CONTROLLED=no
```

After these changes, restart the network:

```
# service network 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.

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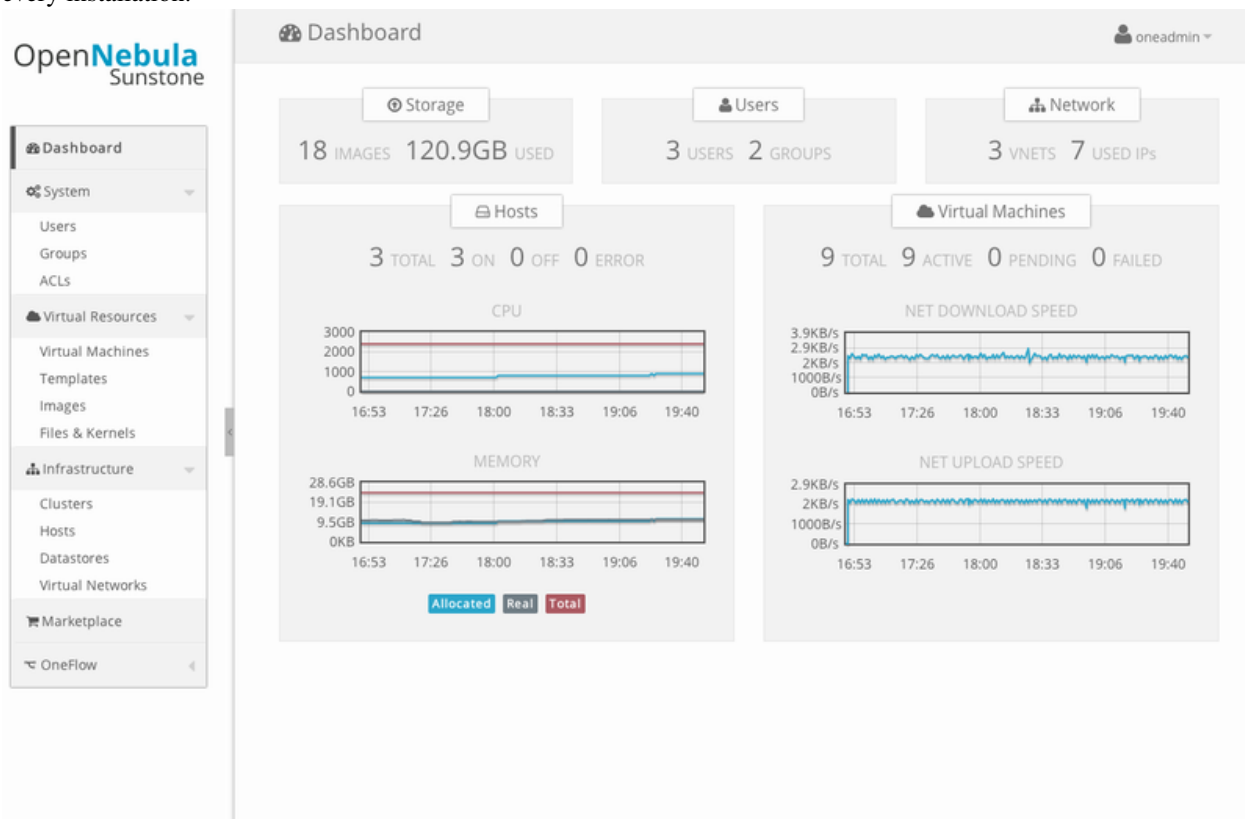
To interact with OpenNebula, you have to do it from the `oneadmin` account in the frontend. We will assume all the following commands are performed from that account. To login as `oneadmin` execute `su - oneadmin`.

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 xen -v xen -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

    --path "http://appliances.c12g.com/CentOS-6.5/centos6.5.qcow2.gz" \
$ oneimage create --name "CentOS-6.5_x86_64" \
    --path "http://172.16.77.1/vm-images/CentOS65.qcow2" \
    --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`.

We must specify the desired bootloader to the template we just created. To do so execute the following command:

```
$ EDITOR=vi onetemplate update CentOS-6.5
```

Add a new line to the OS section of the template that specifies the bootloader:

```
OS=[
  BOOTLOADER = "pygrub",
  ARCH="x86_64" ]
```

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*