Virtual Storage Manager for Ceph

Version: 2.0.0.201

Source: 2015-09-20

Keywords: Ceph, Openstack, Virtual Storage Management

Supported Combo:

```
OS:         Ubuntu 14.04.2/CentOS 7
Ceph:       Firefly/Giant*/Hammer
OpenStack:  Havana/Icehouse/Juno

(Other combos might also be working, but we didn't try yet.)
```

Preparation

Before you get ready to install VSM, you should prepare your environment. ***VSM CANNOT manage Ceph Cluster not created by it***. The sections here are helpful for understanding the deployment concepts.

Note: For a Ceph cluster created and managed by VSM you need to prepare at least three storage nodes plus a VSM controller node. VSM requires a minimum of three Ceph storage nodes (physical or virtual) before it will create a Ceph cluster.

Roles

There are two roles for the nodes (servers) on your VSM created Ceph cluster.

Controller Node

The controller node is used to run mariadb, rabbitmq, web ui services for the VSM cluster.

Agent Node (a.k.a Storage Node)

The agent node is used to run the vsm-agent service which manages the Ceph and physical storage resources. These nodes are the Ceph storage and monitor nodes.

Network

There are three kinds of networks defined in VSM, and the three networks can all be the same network or separate networks or subnets. VSM does not support split subnets - e.g. two or more different subnets that together make up the management network, or the ceph public network. or the ceph public network.

Management Network

Management Network is used to manage the VSM cluster, and interchanges VSM mangement data between vsm controller and agents.

Ceph Public Network

Ceph Public Network is used to serve IO operations between ceph nodes and clients.

Ceph Cluster Network

Ceph Cluster Network is used to interchange data between ceph nodes like Monitors and OSDs for replication and rebalancing.

Recommendations

- Controller node should have connectivity to:

```
Management Network
```

- Agent Node should have connectivity to:

```
Management Network
Ceph Public Network
Ceph Cluster Network
```

Sample 1

- **Controller node** contains the networks listed below:

```
192.168.123.0/24
```

- **Storage node** contains networks below:

```
192.168.123.0/24
192.168.124.0/24
192.168.125.0/24
```

Then we may assign these networks as below:

```
>     Management network: 192.168.123.0/24
>     Ceph public netwok: 192.168.124.0/24
>     Ceph cluster network: 192.168.125.0/24
```

The configuration for VSM in the `cluster.manifest` file should be:

```
>     [management_addr]
>     192.168.123.0/24
>
>     [ceph_public_addr]
>     192.168.124.0/24
>
>     [ceph_cluster_addr]
>     192.168.125.0/24
```

Refer cluster.manifest for details.

Sample 2

But how about when all the nodes just have two NICs. Such as a controller node and storage node having:

```
>     192.168.123.0/24
>     192.168.124.0/24
```

We can assign these two networks as below:

```
>     Management network: 192.168.123.0/24
>     Ceph public network: 192.168.124.0/24
>     Ceph cluster network: 192.168.123.0/24
```

The configuration for VSM in `cluster.manifest` file would then be:

```
>     [management_addr]
>     192.168.123.0/24
>
>     [ceph_public_addr]
>     192.168.124.0/24
>
>     [ceph_cluster_addr]
>     192.168.123.0/24
```

Sample 3

It's quite common to have just one NIC in demo environment, then all nodes just have:

```
>   192.168.123.0/24
```

We may assign this network as below:

```
>   Management network: 192.168.123.0/24
>   Ceph public network: 192.168.123.0/24
>   Ceph cluster network: 192.168.123.0/24
```

So all of the three VSM networks use the same subnet, The configurations in `cluster.manifest` file would then be:

```
>     [management_addr]
>     192.168.123.0/24
>
>     [ceph_public_addr]
>     192.168.123.0/24
>
>     [ceph_cluster_addr]
>     192.168.123.0/24
```

Deployment

Deployment involves building the four required Ceph cluster nodes, configuring them for Ceph/VSM deployment, and then deploying VSM (which also deploys Ceph).

Pre-Flight Configuration

Some pre-flight configuration steps are required before you attempt to deploy your newly built tarball:

1. Create four Ubuntu 14.04 virtual machines. One of them will be the VSM controller, the other three will be storage nodes in the cluster. VSM requires a minimum of three storage nodes and one controller. There are many configurations you could use, but this is the simplest that is still fully functional. Since the controller and storage nodes are nearly identical to each other, we'll just specify and install the controller node VM and then clone it for a storage node. We'll then add storage devices to the storage node and clone that one twice more for the other two storage nodes, as follows:

    1. Choose a user that will be the ceph/vsm user - something like cephuser.
    2. Ensure ntp is configured and refers to a good time source (this is pretty much automatic with Ubuntu).
    3. Ensure OpenSSH server software is installed:
2. Once the OS installation process has completed and the system has rebooted, login to the controller as cephuser and update and upgrade the system to ensure that the very latest Ubuntu software is installed (download this script):

    Update System

    ```
    $ sudo apt-get update
    ...
    $ sudo apt-get upgrade
    ...
    $ sudo apt-get dist-upgrade
    ...
    ```

    Answer 'Y' to all the prompts (or just press 'enter' - most of them default to 'Y' anyway).

3. Edit the /etc/hosts file and remove any secondary localhost addresses (e.g., 127.0.1.1) and ensure the primary localhost address (127.0.0.1) does not refer to the actual host name, but rather only to "localhost". This is necessary (as explained in the VSM deployment wiki page in item 3 of the Automatic Deployment section) because VSM deployment will sync the /etc/hosts file to the cluster storage nodes - you don't want system A referring to localhost via system B's host name.

4. Make the cephuser a super user with respect to sudo (download this script): Make cephuser a Super User

```
$ echo "cephuser ALL=(ALL) NOPASSWD:ALL" | sudo tee /etc/sudoers.d/cephuser
$ sudo chmod 0440 /etc/sudoers.d/cephuser
```

VERY IMPORTANT: These commands add an effective line to the sudoers file that allows cephuser to do anything in sudo without a password. This is necessary because often the VSM deployment script performs remote operations using commands prefixed with ssh cephuser@remote-node sudo ... which will fail if sudo requires a password, as there's no way to front sudo's interactive password prompt at the controller end of the command.

5. Shut down the controller and clone it for the first storage node.
6. Edit the VM settings for the clone and add two additional virtual hard drives (/dev/sdb and /dev/sdc); these will be the storage node's data store. Ceph likes to use the xfs file system with a separate journal. The journal drive can be smaller than the data drive. As per xfs documentation, the size of the journal drive depends on how you intend to use the storage space on the data drive but for this experiment a few GB is sufficient for journaling.
7. Boot up the first storage node and rename it - on Ubuntu, host rename can be done with the following command:

   Ubuntu Host Rename

   ```
   $ sudo hostname vsm-store1
   Logout and reboot to allow the DNS server to pick up the new name.
   ```

8. Login again as cephuser and run the following commands to prepare the /dev/sdb and /dev/sdc devices for Ceph use as a storage device (download this script):

   Partition /dev/sdb for XFS

   ```
   $ sudo parted /dev/sdb -- mklabel gpt
   [sudo] password for cephuser: ******
   Information: You may need to update /etc/fstab.

   $ sudo parted -a optimal /dev/sdb -- mkpart xfs 1MB 100%
   Information: You may need to update /etc/fstab.

   $ sudo parted /dev/sdc -- mklabel gpt
   Information: You may need to update /etc/fstab.

   $ sudo parted -a optimal /dev/sdc -- mkpart primary 1MB 100%
   Information: You may need to update /etc/fstab.
   ```

   This formats the /dev/sdb device and adds an XFS file system, and then formats the /dev/sdc device in preparation for use as an xfs journal.

9. Logout and shut down the first storage node and clone it twice more to create the remaining two storage nodes.
10. Power these system on one at a time and change the host names of each so they're unique (I named mine, vsm-controller, vsm-node1, vsm-node2, and vsm-node3, for instance). NOTE: This can be tricky in our dynamic DHCP environment where DNS is populated by querying hosts as they request addresses.
11. Power them on and edit the /etc/hosts files on each of the four VMs; append entries mapping each of the node names to their DHCP IP addresses: Additional Entries on Each Node's /etc/hosts File

    ```
    ...
    10.50.33.75 vsm-controller
    10.50.33.76 vsm-node1
    10.50.33.77 vsm-node2
    10.50.33.78 vsm-node3
    ```

    This step is only necessary to ensure a stable name-to-IP mapping among the nodes. Without doing this in a dynamic DHCP setting like ours, you could find the DNS server is not quick enough picking up the host names and mapping them to their DHCP provided IP addresses in the DNS database. The end result is that deployment tends to fail because DNS can't find the host (yet). This step is, of course, not needed if you're using static IP addresses.

12. On each of the four systems, create an ssh key for the cephuser account (don't set any passwords on the key), then copy the ssh identity on each of the four nodes to the other three. For instance, on the controller node:

    Create an SSH Key

```
cephuser@vsm-controller:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/cephuser/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/cephuser/.ssh/id_rsa.
Your public key has been saved in /home/cephuser/.ssh/id_rsa.pub.
The key fingerprint is:
ee:4d:85:19:69:26:0b:06:55:b5:f4:c6:7a:43:e2:2a cephuser@vsm->
controller
The key's randomart image is:
+--[ RSA 2048]----+
|    ......o      |
|     .   . =     |
|      o . B =    |
|     . . * O     |
|        S = +    |
|       . . o .   |
|      E o .      |
|       o o       |
|        . .      |
+-----------------+

cephuser@vsm-controller:~$ ssh-copy-id vsm-node1
The authenticity of host 'vsm-node1 (10.50.33.75)' can't be established.
ECDSA key fingerprint is b6:29:c3:eb:3c:01:09:68:2b:bc:ab:29:f3:3c:15:58.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
cephuser@vsm-node1's password: ******
Number of key(s) added: 1
Now try logging into the machine, with:   "ssh 'vsm-node1'"
and check to make sure that only the key(s) you wanted were added.
cephuser@vsm-controller:~$ ssh-copy-id vsm-node2
...
cephuser@vsm-controller:~$ ssh-copy-id vsm-node3
...
Do the same on each of the other nodes; this will allow the deployment process to ssh from any node to any node without c
```

13. On the controller node, set a root password and su into the root account, then copy the .ssh directory from /home/cephuser/.ssh to /root/.ssh so you can deploy as root (via sudo) and still have password-less access to the cephuser accounts on each of the other nodes:

```
Copy /home/cephuser/.ssh to /root/.ssh
cephuser@vsm-controller:~$ sudo passwd root
[sudo] password for cephuser:
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
cephuser@vsm-controller:~$ su -
Password: ******
root@vsm-controller:~# cp -r /home/cephuser/.ssh .ssh
```

14. At this point, it might be a good idea to take a VMware snapshot of these four systems so you have a clean starting point if you wish to restart from scratch.

15. Copy the built tarball from your build system to the controller node's cephuser account home directory.

16. Extract the tarball (it will create its own root directory). It should expand to a directory that looks something like this: The Expanded Release Tarball

```
.
├── CHANGELOG.md
├── CHANGELOG.pdf
├── debs.lst
├── get_pass.sh
├── INSTALL.md
├── INSTALL.pdf
├── installrc
├── install.sh
├── LICENSE
├── manifest
│   ├── cluster.manifest.sample
│   └── server.manifest.sample
├── NOTICE
├── README.md
├── RELEASE
├── uninstall.sh
├── VERSION
└── vsmrepo
    ├── Packages.gz
    ├── python-vsmclient_2.0.0-149_amd64.deb
    ├── vsm_2.0.0-149_amd64.deb
    ├── vsm-dashboard_2.0.0-149_amd64.deb
    └── vsm-deploy_2.0.0-149_amd64.deb
```

Automatic Deployment

Starting with VSM 1.1, an automatic deployment tool is provided which can simplify the deployment. This tool is still in development, so your feedback and JIRA reports of any problems are very welcome.

This section will describe how to use the tool to conduct automation.

1. Firstly, a VSM binary release package should be acquired. It may be downloaded from binary repository, or built from source (see Build VSM). Then unpack the release package, the folder structure looks as following:

   ```
   .
   ├── CHANGELOG
   ├── installrc
   ├── INSTALL.md
   ├── install.sh
   ├── uninstall.sh
   ├── LICENSE
   ├── manifest
   │   ├── cluster.manifest.sample
   │   └── server.manifest.sample
   ├── NOTICE
   ├── README
   └── vsmrepo
       ├── python-vsmclient_2.0.0-123_amd64.deb
       ├── Packages.gz
       ├── vsm_2.0.0-123_amd64.deb
       ├── vsm-dashboard-2.0.0-123_amd64.deb
       └── vsm-deploy-2.0.0-123_amd64.deb
   ```

2. Changing the *installrc* file, set the *AGENTADDRESSLIST* and the *CONTROLLER_ADDRESS*, the ip addresses in *AGENTADDRESSLIST* is delimitered by space, and all ip addresses are used in management subnet. e.g.:

   ```
   AGENT_ADDRESS_LIST="192.168.123.21 192.168.123.22 192.168.123.23"
   CONTROLLER_ADDRESS="192.168.123.10"
   ```

   It's OK to use host name instead of ip addresses here.

3. VSM will sync /etc/hosts file from the controller node, make sure your controller node's /etc/hosts file follows below rules:

> Lines with `localhost`, `127.0.0.1` and `::1` should not contains the actual hostname.

4. Under the *manifest* folder, you should create the folders named by the management ip of the controller and storage nodes, and then the structure looks as follows:

```
.
├── 192.168.123.10
├── 192.168.123.21
├── 192.168.123.22
├── 192.168.123.23
├── cluster.manifest.sample
└── server.manifest.sample
```

5. Copy the *cluster.manifest.sample* to the folder named by the management ip of controller node, then change the filename to *cluster.manifest* and edit it as required, refer cluster.manifest for details.

6. Copy the *server.manifest.sample* to the folders named by the management ip of storage nodes, then change the filename to *server.manifest* and edit it as required, refer server.manifest for details.

7. Finally, the manifest folder structure looks as follows:

```
.
├── 192.168.123.10
│   └── cluster.manifest
├── 192.168.123.21
│   └── server.manifest
├── 192.168.123.22
│   └── server.manifest
├── 192.168.123.23
│   └── server.manifest
├── cluster.manifest.sample
└── server.manifest.sample
```

8. If you want to upgrade vsm binary packages only, one approach is to build release package separately (see Build Packages). The generated binary packages will be in *vsmrepo* folder after unpack the release package, then you can execute below command to install binary package:

```
dpkg -i <package>
```

9. Now we are ready to start the automatic procedure by executing this command line:

```
./install.sh -u ubuntu -v <version>
```

   where *version* is the vsm version like 1.1, 2.0.

10. If execution is blocked at any point, please try to enter "y" and move ahead.

11. If all goes well, you can then login to the VSM Web UI.

VSM Web UI

1. Access https://vsm controller IP/dashboard/vsm.(for example *https://192.168.123.10/dashboard/vsm*)

2. User name: admin, and password can be obtained from: */etc/vsmdeploy/deployrc* in the ADMIN_PASSWORD field:

   > ./get_pass.sh

3. Then you can switch to the `Cluster Management` item, then `Create Cluster` panel, and push the create cluster button to create a ceph cluster. At this point please refer to the VSM Manual, which is located at `https://01.org/virtual-storage-manager`

Uninstall

There are a few cases where you may expect to uninstall VSM, e.g, you expect to reinstall it with different configurations, you feel VSM doesn't

work as you expected. You could take below steps to do the removal:

1. Go to the VSM folder where you start the installation procedure.
2. Make sure the `installrc` file is there, and the ip addresses for controller node and agent nodes are correctly set. Normally, if you correctly installed VSM, you should have already correctly set the file.
3. Execute below command:

```
./uninstall.sh
```

Reference

Build Packages

There are two approaches to get a VSM release package, a direct way is to download release package from github, or you can build release package from source code as following:

```
>    ./buildvsm.sh -v <version>
```

where *version* is the vsm version like 1.1, 2.0. A release package named like *2.0.0-123.tar.gz* will be generated in *release* folder if all execute well.

cluster.manifest

The cluster.manifest file is under manifest// folder, the three subnets must be modified according to Ceph cluster network topology.

subnets

1. Modify the three IP addresses according to your environment. `management_addr` is used by VSM to communicate with different services, such as using rabbitmq to transfer messages, rpc.call/rpc.cast etc.`ceph_public_addr` is a public (front-side) network address. `ceph_cluster_addr` is a cluster (back-side) network address.

Also, make sure the netmask is correctly set. In this sample, *netmask*=24 is fine, but with AWS instances, normally, *netmask*=16 are required.

```
[management_addr]
192.168.123.0/24

[ceph_public_addr]
192.168.124.0/24

[ceph_cluster_addr]
192.168.125.0/24
```

Here is a complete list of all settings for cluster.manifest:

- [**storage_class**]

  In this section, you can put you planned storage class name. One line for one class name, only names with numbers, alphabetic and underscore can be used for class name.

- [**storage_group**]

  In this section, you can put your storage group definition, in the below format, here [] is not needed. Only numbers, alphabetic and underscore can be used for any of them.

```
[storage group name]  [user friendly storage group name] [storage class name]
```

- [**cluster**]

  In this section, you can put your cluster name. Only numbers, alphabetic and underscore can be used.

- [**file_system**]

You can use the file system which ceph can support here. The default value is xfs

- [**zone**]

  In this section, you can add zone name under the section.

  - format:

```
[zone-name]
```

  - comments:

    1. Only numbers, alphabetic and underscore can be used for zone name.
    2. By default, this section is disabled, in this case, a default zone called *zone_one* will be used.

  - example:

```
zone1
```

- [**management_addr**]
- [**ceph_public_addr**]
- [**ceph_cluster_addr**]

  Those 3 sections will define the three subnets.

- [**settings**]

  In the section, you can set values for these settings for ceph and VSM.

```
storage_group_near_full_threshold  65
storage_group_full_threshold  85
ceph_near_full_threshold  75
ceph_full_threshold  90
pg_count_factor 100
heartbeat_interval 5
osd_heartbeat_interval 10
osd_heartbeat_grace 10
disk_near_full_threshold 75
disk_full_threshold 90
osd_pool_default_size 3
```

- [**ec_profiles**]

  In this section, you can define some erasure coded pool profile before you create the cluster.

  - format:

```
profile-name] [path-to-plugin] [plugin-name] [pg_num value] [json format key/value]
```

  - comments:

    1. the key/value strings should not have spaces.

  - example:

```
default_profile  /usr/lib64/ceph/erasure-code  jerasure  3  {"k":2,"m":1,"technique":"reed_sol_van"}
```

- [**cache*tier*defaults**]

  The default settings value for create cache tier in the web UI. You can also change them while you create cache tier for pools.

```
ct_hit_set_count 1
ct_hit_set_period_s 3600
ct_target_max_mem_mb 1000000
ct_target_dirty_ratio 0.4
```

```
ct_target_full_ratio 0.8
ct_target_max_objects 1000000
ct_target_min_flush_age_m 10
ct_target_min_evict_age_m 20
```

server.manifest

The server.manifest file is under manifest// folder, below settings must be modified based on your environment.

- [**vsm_controller_ip**]

  Here `vsm_controller_ip` is the VSM controller's IP address under `management_addr` subnet.

  - example:

```
[vsm_controller_ip]
192.168.123.10
```

- [**role**]

  Delete one if you don't want this server act as this role. The default is that server will act as storage node and monitor at the same time.

  - example:

```
[role]
storage
monitor
```

- [**auth_key**]

  Replace the content with the key you get from controller by running the agent-token command on the controller.

  DON'T MODIFY IT, the automatic deployment tool will fill this section.

- **OSD definition under each storage group**

  The storage you use for your Ceph cluster must have previously been provisioned by you with a label and a partition.

  For example:

```
parted /dev/sdb -- mklabel gpt
parted -a optimal /dev/sdb -- mkpart xfs 1MB 100%
```

  Enter your primary and associated journal storage information in the server.manifest, remeber to fill them in right storage group.

  For example, change the lines below:

```
[10krpm_sas]
#format [sas_device]  [journal_device]
%osd-by-path-1%   %journal-by-path-1%
%osd-by-path-2%   %journal-by-path-2%
%osd-by-path-3%   %journal-by-path-3%
```

  to be:

```
[10krpm_sas]
#format [sas_device]  [journal_device]
/dev/sdb1 /dev/sdc1
/dev/sdd1 /dev/sdc2
/dev/sde1 /dev/sdf
```

  Then delete the redundant lines with %osd-by-path%, if you have fewer disks.

  We recommend though that you use disk-by-path instead for the disk paths. Use the command below to find the true by-path:

```
ls -al /dev/disk/by-path/* | grep `disk-path` | awk '{print $9,$11}'
```

For example:

```
$> ls -al /dev/disk/by-path/* | grep sdb | awk '{print $9,$11}'
/dev/disk/by-path/pci-0000:00:0c.0-virtio-pci-virtio3 ../../sdb
```

Then replace the /dev/sdb with `/dev/disk/by-path/pci-0000:00:0c.0-virtio-pci-virtio3` in `/etc/manifest/server.manifest` file. Do this also for all the other disks listed in this file.

Warning: It may cause an error when you add a disk without by-path. So, If you can not find the by-path for a normal disk, you should not use it. Or if you use it to create the cluster, and the create cluster fails, please delete it from the `/etc/manifest/server.manifest` file.

After that the disk list appears like this, here the storage group name `10krpm_sas` should have already defined in `[storage_group]` section in `cluster.manifest`.

```
[10krpm_sas]
#format [sas_device]  [journal_device]
/dev/disk/by-path/pci-0000:00:0c.0-virtio-pci-virtio3   /dev/disk/by-path/pci-0000:00:0d.0-virtio-pci-virtio4
/dev/disk/by-path/pci-0000:00:0e.0-virtio-pci-virtio5   /dev/disk/by-path/pci-0000:00:0f.0-virtio-pci-virtio6
/dev/disk/by-path/pci-0000:00:10.0-virtio-pci-virtio7   /dev/disk/by-path/pci-0000:00:11.0-virtio-pci-virtio8
```

Storage Groups

If you have several kinds of storage media like 10krpm SAS drives & solid state drives, and you want these disks organized into different storage groups in VSM, then you may follow the operations below. Otherwise, you may skip this step and just put all the disks into the `[10krpm_sas]` section.

You may want to add disks into other sections in the `/etc/manifest/server.manifest` file after the `[10krpm_sas]` section. Take `[ssd]` as an example:

- Add storage class in `/etc/manifest/cluster.manifest` in controller node.

  [storage*class] ssd # add this line 10krpm*sas

- Add storage group in `/etc/manifest/cluster.manifest` on the controller node.

  [storage*group] high*performance*test HighPerformanceSSD*test ssd

- Add disks under `/etc/manifest/server.manifest` on the storage node(s) which have SSD, such as:

```
[ssd]
/dev/disk/by-path/pci-0000:00:0c.0-virtio-pci-virtio9   /dev/disk/by-path/pci-0000:00:0d.0-virtio-pci-virtio1
/dev/disk/by-path/pci-0000:00:0e.0-virtio-pci-virtio11   /dev/disk/by-path/pci-0000:00:0f.0-virtio-pci-virtio14
/dev/disk/by-path/pci-0000:00:10.0-virtio-pci-virtio23  /dev/disk/by-path/pci-0000:00:11.0-virtio-pci-virtio10
```

Trouble shooting

If you encountered any issues, below files may give you hints:

```
> /var/log/vsm/*.log
> /var/log/httpd/*.log
> /var/log/syslog
```

If you want to check that the vsm agent started correctly, you can look at the files in */var/log/vsm* on each storage node. The vsm.physical.log should have no errors and end with the line:

```
INFO [vsm.openstack.common.rpc.common] Connected to AMQP server on <vsm-controller-IP-address>:5673
```

and you should see a similar message at the start of vsm.agent.log and ending with:

```
INFO [vsm.agent.manager] agent/manager.py update ceph.conf from db. OVER
```

Likewise, you should see no errors in the three log files in /var/log/vsm on the controller node.

Frequently Asked Questions

Q: Executing "agent-token" is hung.

```
A: Please check http proxy setting to make sure no http_proxy variable is set in the enviornment.
```

Q: "An error occurred authenticating. Please try again later." appears on the controller web ui after fresh installation.

```
A: Firstly, please make sure the right password is entered, the password can be obtained from /etc/vsmdeploy/deployrc in "ADMIN_PA
```

Q: keyring error on cluster creation.

```
A: The root cause is that the vsm controller has already updated a new token, but it is not applied on all agents.
```

Q: Negative update time is showing on RBD list page.

```
A: Before creating the ceph cluster, please make sure all ceph nodes are time synchronized via NTP.
```

Q: vsm-agent process causes one disk to be saturated with i/o load.

```
A: A known case causes i/o saturation if multiple OSDs are defined on the same physical device, which is normally used in the demo
```

Q: Can't replace node if ceph cluster contains only 3 nodes.

```
A: This is an expected safeguard. A 3 node cluster minimum is needed to meet availability requirements.
```

Q: Can I define the ceph version I want to install?

```
A:
VSM can work with different Ceph releases like Firefly, Giant and Hammer. By default, it will use the ceph version provided by OS (
If user expects to use some specific ceph version, he/she needs add the new ceph repo into system repository.

For ubuntu, user could create /etc/apt/sources.list.d/ceph.list to override OS distro's defaults before installation. For example,
    >
    > sudo rm /etc/apt/sources.list.d/ceph.list
    > echo deb http://ceph.com/debian-hammer/ $(lsb_release -sc) main | sudo tee /etc/apt/sources.list.d/ceph.list


For CentOS, creating /etc/yum.repos.d/ceph.repo at first, then filling the ceph.repo shown below:
    ###################
    [ceph]
    name=Ceph packages for $basearch
    baseurl=http://ceph.com/rpm-hammer/el6/$basearch
    enabled=1
    priority=2
    gpgcheck=1
    type=rpm-md
    gpgkey=https://ceph.com/git/?p=ceph.git;a=blob_plain;f=keys/release.asc

    [ceph-noarch]
    name=Ceph noarch packages
    baseurl=http://ceph.com/rpm-hammer/el6/noarch
    enabled=1
    priority=2
    gpgcheck=1
    type=rpm-md
    gpgkey=https://ceph.com/git/?p=ceph.git;a=blob_plain;f=keys/release.asc
```

```
    [ceph-source]
    name=Ceph source packages
    baseurl=http://ceph.com/rpm-hammer/el6/SRPMS
    enabled=0
    priority=2
    gpgcheck=1
    type=rpm-md
    gpgkey=https://ceph.com/git/?p=ceph.git;a=blob_plain;f=keys/release.asc
    ###############

In 2.0, VSM also provides ceph upgrade feature from Web UI under "Cluster Management" menu.
```

Q: Can VSM be installed on ubuntu desktop version?

```
A: We don't well test it on ubuntu desktop, but we know this kind of case, one common issue to be encountered is:
    Error: "Activation of org.freedesktop.secrets timed out"

The workaround is to execute below command to temporary disable gnome-keyring service.
    mv /usr/share/dbus-1/services/org.freedesktop.secrets.service /usr/share/dbus-1/services/org.freedesktop.secrets.service.bak
```

Q: Why I can't upgrade from Firefly to Giant?

```
A: There are a few missing packages in Giant repository like python-rados, librados2-devel, python-cephfs, python-rbd, librbd1-dev
```