1 环境说明

1、环境说明：

a：生产环境尽量要选配置高一些的服务器，如配置：双 CPU 2.4GHz 24cores*2/64GB*8/600GB SAS*2 或者更高主频的 CPU，网络至少是 10Gb/s，主要还是根据自己实际业务场景去选型。

b：实验环境规划及说明

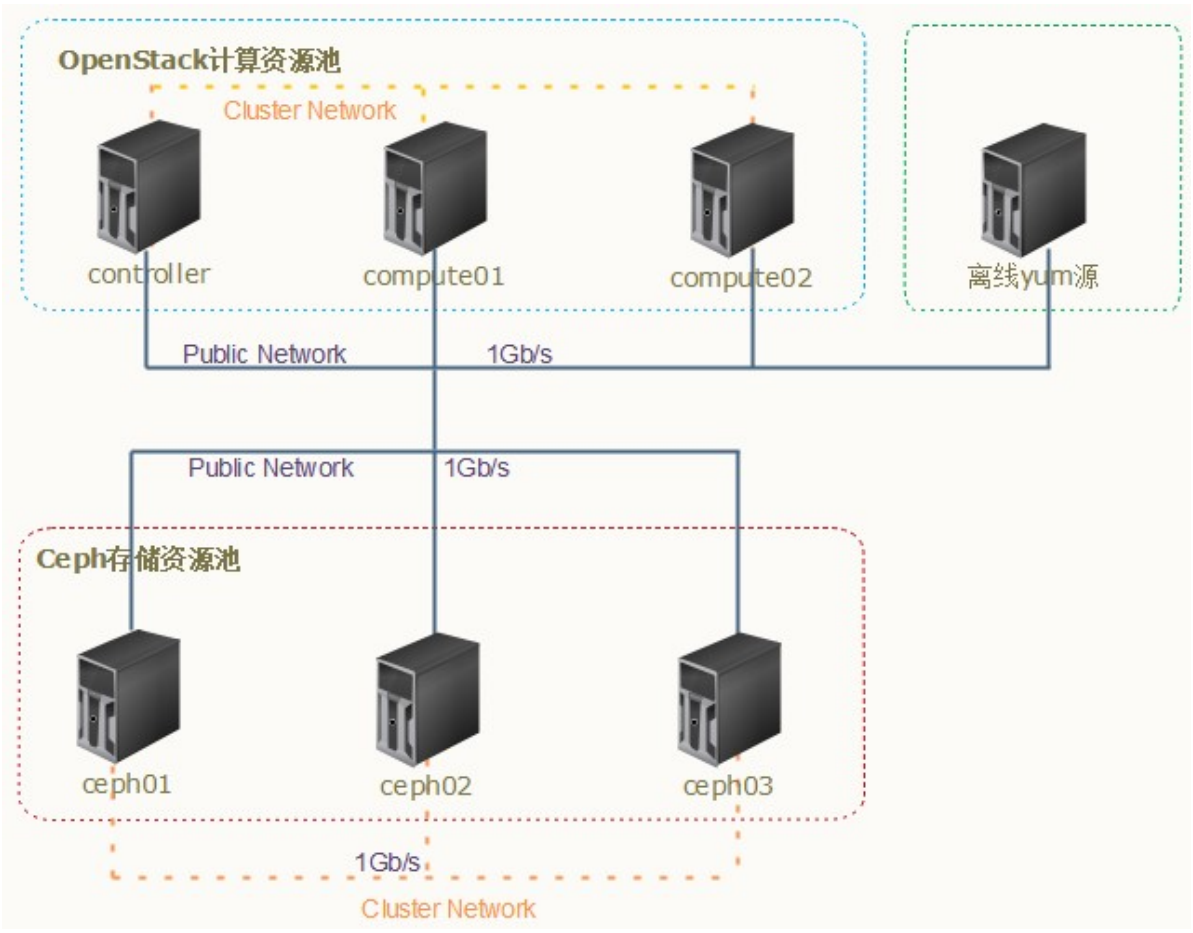
宿主机硬件配置：i7-10750H / 32GB / TOSHIBA 500G SSD / WD 1TB SSD

宿主机操作系统：Windows 10 64 位专业版 / 版本号：21H2

宿主机使用软件：VMware® Workstation 16 Pro

Ceph 版本： 14.2.22（Nautilus）

组网方式：



Ceph 集群用到的虚拟机规划见下表：

主机名	用途	磁盘/大小	操作系统ISO	最小内存	swap分区	ens33 Provider IP(公共网络)	ens37 vxlan 隧道IP(租户网络)	ens38 集群网络	备注
template	虚拟机模板	sda/300G	CentOS-7-x86_64-DVD-2009.iso (CentOS-7.9-x64)	8GB	实验环境内存不足，需分配swap(8GB)，减少OOM，生产环境建议只在计算节点上分配swap,且不宜过大，避免影响虚机性能。	192.168.59.251/24			仅用作虚拟机克隆用
yum01	本地yum源仓库	sda/300G		2GB		192.168.59.250/24			
controller	openstack 控制节点	sda/300G		8GB		192.168.59.20/24	10.168.59.20/24	20.168.59.20/24	镜像制作服务器
compute01	openstack 计算节点/存储节点	sda/300G、sdb/50G		6GB		192.168.59.31/24	10.168.59.31/24	20.168.59.21/24	
compute02	openstack 计算节点/存储节点	sda/300G、sdb/50G		6GB		192.168.59.32/24	10.168.59.32/24	20.168.59.22/24	
ceph01	ceph存储节点	sda/300G、sdb/100G、sdc/100G、sdd/100G		4GB		192.168.59.11/24		20.168.59.11/24	ceph作为cinder的后端存储。sdb/sdc/sdd用作osd
ceph02	ceph存储节点	sda/300G、sdb/100G、sdc/100G、sdd/100G		4GB		192.168.59.12/24		20.168.59.12/24	
ceph03	ceph存储节点	sda/300G、sdb/100G、sdc/100G、sdd/100G		4GB		192.168.59.13/24		20.168.59.13/24	

在 ceph01 节点上执行

设置主机名

```
# hostnamectl set-hostname ceph01
```

修改网卡 ens33 的 IP

```
# sed -i 's/59.251/59.11/g' /etc/sysconfig/network-scripts/ifcfg-ens33
```

配置用于集群网络的网卡 ens37 的 IP

```
# cat <<EOF > /etc/sysconfig/network-scripts/ifcfg-ens37
TYPE=Ethernet
BOOTPROTO=static
DEFROUTE=yes
IPV6INIT=no
NAME=ens37
DEVICE=ens37
ONBOOT=yes
IPADDR=20.168.59.11
NETMASK=255.255.255.0
EOF
```

systemctl restart network

在 ceph02 节点上执行
设置主机名
hostnamectl set-hostname ceph02
修改网卡 ens33 的 IP
sed -i 's/59.251/59.12/g' /etc/sysconfig/network-scripts/ifcfg-ens33

配置用于集群网络的网卡 ens37 的 IP
# cat <<EOF > /etc/sysconfig/network-scripts/ifcfg-ens37
TYPE=Ethernet
BOOTPROTO=static
DEFROUTE=yes
IPV6INIT=no
NAME=ens37
DEVICE=ens37
ONBOOT=yes
IPADDR=20.168.59.12
NETMASK=255.255.255.0
EOF

systemctl restart network

在 ceph03 节点上执行
设置主机名
hostnamectl set-hostname ceph03
修改网卡 ens33 的 IP
sed -i 's/59.251/59.13/g' /etc/sysconfig/network-scripts/ifcfg-ens33

# cat <<EOF > /etc/sysconfig/network-scripts/ifcfg-ens37
TYPE=Ethernet
BOOTPROTO=static
DEFROUTE=yes
IPV6INIT=no
NAME=ens37
DEVICE=ens37
ONBOOT=yes
IPADDR=20.168.59.13
NETMASK=255.255.255.0
EOF

systemctl restart network

ceph01-ceph03 各主机 hosts 设置如下
cat /etc/hosts
openstack
192.168.59.20 controller
192.168.59.31 compute01
192.168.59.32 compute02

cpeh-public
192.168.59.11 ceph01
192.168.59.12 ceph02
192.168.59.13 ceph03

ceph-cluster
20.168.59.11 ceph01-cluster
20.168.59.12 ceph02-cluster
20.168.59.13 ceph03-cluster

2 安装 ceph

2.1 系统初始化参数检查及设置

在 ceph 所有节点（ceph01、ceph02、ceph03）上检查执行

再次检查 hosts
cat /etc/hosts

openstack
192.168.59.20 controller
192.168.59.31 compute01
192.168.59.32 compute02

cpeh-public
192.168.59.11 ceph01
192.168.59.12 ceph02
192.168.59.13 ceph03

ceph-cluster
20.168.59.11 ceph01-cluster
20.168.59.12 ceph02-cluster
20.168.59.13 ceph03-cluster

yum
192.168.59.250 yum01

检查网络
ifconfig
关闭 NetworkMnanager
systemctl stop NetworkManager
systemctl disable NetworkManager

关闭防火墙：
systemctl stop firewalld
systemctl disable firewalld

关闭 selinux
sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config

SELINUX=disabled

设置 /etc/security/limits.conf
End of file
* hard nofile 655360
* soft nofile 655360
* hard nproc 655360
* soft nproc 655360
* soft core 655360
* hard core 655360
设置 /etc/security/limits.d/20-nproc.conf

* soft nproc unlimited
root soft nproc unlimited

2.2 NTP 客户端配置

在 ceph 所有节点（ceph01、ceph02、ceph03）上执行

a、统一时区，在所有节点上执行
ln -sf /usr/share/zoneinfo/Asia/Shanghai /etc/localtime

b、安装 chrony，默认已安装
yum -y install chrony

然后，修改配置/etc/chrony.conf 中服务器为 controller，即
#server 0.centos.pool.ntp.org iburst
#server 1.centos.pool.ntp.org iburst
#server 2.centos.pool.ntp.org iburst
#server 3.centos.pool.ntp.org iburst
server controller iburst

设置开机自启动
systemctl enable chronyd
systemctl restart chronyd

验证时间同步：
chronyc sources

2.3 配置 SSH 免密登录

1、在 ceph01 节点生成公钥，然后发放到所有的主机/客户机节点上，即包括 ceph02、ceph03、controler、compute01、compute02。

ssh-keygen -t rsa
/*
说明：按回车键，即默认设置完成
*/
发放公钥到所有主机上，输入目标主机密码完成。
ssh-copy-id ceph01
ssh-copy-id ceph02
ssh-copy-id ceph03
ssh-copy-id controller
ssh-copy-id compute01
ssh-copy-id compute02

验证从 ceph01 访问所有主机是否可以免密登录
ssh ceph01
ssh ceph02
ssh ceph03
ssh controller
ssh compute01
ssh compute02

2.4 安装 Ceph 软件包

集成过程中，需要在 OpenStack 的 controller 节点和 compute 节点安装 Ceph 软件包，作为 Ceph 客户端。

如果已经制作离线 yum 源，则直接安装即可；如果要使用外部互联网 yum 源，需要设置 ceph 源（此处列出阿里云源），配置如下
vim /etc/yum.repos.d/ceph.repo
[ceph]
name=Ceph packages for $basearch
baseurl=http://mirrors.aliyun.com/ceph/rpm-nautilus/el7/x86_64
enabled=1
gpgcheck=0

[ceph-noarch]
name=Ceph noarch packages
baseurl=http://mirrors.aliyun.com/ceph/rpm-nautilus/el7/noarch
enabled=1
gpgcheck=0

[ceph-source]
name=Ceph source packages

```
baseurl=http://mirrors.aliyun.com/ceph/rpm-nautilus/el7/SRPMS
enabled=1
gpgcheck=0
```

在 controller、compute01、compute02 安装 ceph 软件包

```
# yum -y install ceph ceph-radosgw
```

在 ceph01 上这装 ceph-deploy

```
# yum -y install ceph-deploy ceph ceph-mds ceph-radosgw
```

```
/*
说明：ceph-deploy 是 ceph 软件定义存储系统的一部分，用来方便地配置和管理 Ceph 存储集群。
*/
```

在 ceph02、ceph03 上这装 ceph

```
# yum -y install ceph ceph-mds ceph-radosgw
```

2.5 初始化配置 mon

```
# 新建 Ceph 集群，并生成集群配置文件和密钥文件（在 ceph01 上执行）
# mkdir /etc/ceph-cluster && cd /etc/ceph-cluster
# ceph-deploy new ceph01 ceph02 ceph03
```

修改/etc/ceph-cluster/ceph.conf，继续添加如下内容：

```
[global]
...
auth_allow_insecure_global_id_reclaim = false
public_network = 192.168.59.0/24
cluster_network = 20.168.59.0/24
```

初始化 mon 节点（在 ceph01 上执行）

```
# cd /etc/ceph-cluster
# ceph-deploy mon create-initial
```

将 keyring 同步到各节点，以便其它节点可以执行 ceph 集群管理命令（在 ceph01 上执行）

```
# ceph-deploy --overwrite-conf admin ceph01 ceph02 ceph03 controller compute01 compute02
```

验证

```
# ceph -s

cluster:
id: 8dd1d22b-6a47-44f2-909f-0390b46d4b05
health: HEALTH_OK
services:
mon: 3 daemons, quorum ceph01,ceph02,ceph03 (age 32m)
mgr: no daemons active
osd: 0 osds: 0 up, 0 in
data:
pools: 0 pools, 0 pgs
objects: 0 objects, 0 B
usage: 0 B used, 0 B / 0 B avail
pgs:
```

```
/*
说明：如果提示：clock skew detected on mon.xxxx，则需要修改配置
mon_clock_drift_allowed = 1.0
*/
```

2.6 初始化配置 mgr

在 ceph01 节点上执行
cd /etc/ceph-cluster
ceph-deploy mgr create ceph01 ceph02 ceph03

ceph-mgr 进程是主备模式，同一时刻只有一个节点工作，其他节点处于 standby

验证 MGR 是否部署成功
ceph -s
cluster:
id: 8dd1d22b-6a47-44f2-909f-0390b46d4b05
health: HEALTH_WARN
OSD count 0 < osd_pool_default_size 3
services:
mon: 3 daemons, quorum ceph01,ceph02,ceph03 (age 34m)
mgr: ceph01(active, since 28s), standbys: ceph02, ceph03
osd: 0 osds: 0 up, 0 in
data:
pools: 0 pools, 0 pgs
objects: 0 objects, 0 B
usage: 0 B used, 0 B / 0 B avail
pgs:

2.7 初始化配置 osd

划分用于 OSD 的日志分区和元数据分区，此处实验环境，故设置较小，按照规划每个节点有 3 个 OSD。
只需要在 ceph01 节点上完成如下命令执行即可
lvcreate -L 5G -n lvwal01 vg00
lvcreate -L 20G -n lvbdb01 vg00

lvcreate -L 5G -n lvwal02 vg00
lvcreate -L 20G -n lvbdb02 vg00

lvcreate -L 5G -n lvwal03 vg00
lvcreate -L 20G -n lvbdb03 vg00

ceph-deploy osd create ceph01 --data /dev/sdb --block-wal vg00/lvwal01 --block-db vg00/lvbdb01 --bluestore
ceph-deploy osd create ceph02 --data /dev/sdb --block-wal vg00/lvwal01 --block-db vg00/lvbdb01 --bluestore
ceph-deploy osd create ceph03 --data /dev/sdb --block-wal vg00/lvwal01 --block-db vg00/lvbdb01 --bluestore

ceph-deploy osd create ceph01 --data /dev/sdc --block-wal vg00/lvwal02 --block-db vg00/lvbdb02 --bluestore
ceph-deploy osd create ceph02 --data /dev/sdc --block-wal vg00/lvwal02 --block-db vg00/lvbdb02 --bluestore
ceph-deploy osd create ceph03 --data /dev/sdc --block-wal vg00/lvwal02 --block-db vg00/lvbdb02 --bluestore

ceph-deploy osd create ceph01 --data /dev/sdd --block-wal vg00/lvwal03 --block-db vg00/lvbdb03 --bluestore
ceph-deploy osd create ceph02 --data /dev/sdd --block-wal vg00/lvwal03 --block-db vg00/lvbdb03 --bluestore
ceph-deploy osd create ceph03 --data /dev/sdd --block-wal vg00/lvwal03 --block-db vg00/lvbdb03 --bluestore

/*
生产环境 Wal 和 db 分区大小建议
按 4TB 一个 OSD，1TB 需要 40GB 的空间存放 DB，4TB 需要大于 160GB 的 SSD 存放 DB
SSD 空间足够时，单个 WAL 分区分 60GB、单个 DB 分区分 180GB。
如果 SSD 空间不够时，WAL 分区分 40GB、单个 DB 分区分 60GB
https://access.redhat.com/documentation/en-us/red_hat_ceph_storage/4/html/administration_guide/osd-bluestore
*/

2.8 初始化 osd 异常处理

初始化 osd 失败时，按照删除 osd 逻辑进行处理（初始化异常时，仅需要从执行 g-h 两步）
a、先确认好要删除的 osd，比如此处要删除 osd.0

[root@ceph01 ceph-cluster]# ceph osd tree
ID CLASS WEIGHT TYPE NAME STATUS REWEIGHT PRI-AFF
-1 0 root default
-3 0 host ceph01
0 0 osd.0 down 0 1.00000

b、将 osd.0 踢出集群
[root@ceph01 ceph-cluster]# ceph osd out 0

c、待集群重新平衡后，停止对应的 osd 进程，需要在目标 osd 主机上执行
systemctl stop ceph-osd@0

d、从 crush 中移除，然后再用 ceph osd tree 查看，会发现 osd.0 不在 osd tree 中了
[root@ceph01 ceph-cluster]# ceph osd crush remove osd.0
[root@ceph01 ceph-cluster]# ceph osd tree

e、删除节点以及节点认证信息
ceph osd rm osd.0
ceph auth del osd.0

f、此时删除成功但是原来的数据和日志目录还在，也就是说数据还在
lsblk
删除对应分区(如果是 lvm，需要 lvremove 命令删除 vg)
需要将 lvm 状态改为 inactive
lvchange -an /dev/ceph-a4a91457-db04-4afc-a938-b195a0c4a0c7/osd-block-8c82a7e1-5205-4472-a0f1-77f0d44485e0
使用 lvremove 移除对应 lv
lvremove /dev/ceph-a4a91457-db04-4afc-a938-b195a0c4a0c7/osd-block-8c82a7e1-5205-4472-a0f1-77f0d44485e0

g、清除磁盘分区信息
ceph-deploy disk zap ceph01 /dev/sdc
h、重新初始化 osd
ceph-deploy osd create ceph01 --data /dev/sdc --block-wal vg00/lvwal02 --block-db vg00/lvbdb02 --bluestore

3 OpenStack 集成 Ceph

3.1 控制节点和计算节点安装 ceph

a、检查 ceph01 节点生成的公钥是否已经发放到 openstack 所有控制节点和计算节点上
ssh controller
ssh compute01
ssh compute02
如果能从 ceph01 免密登录 openstack 的控制节点和计算节点，则表明公钥已经发放，否则再次执行以下命令
ssh-copy-id controller
ssh-copy-id compute01
ssh-copy-id compute02

b、检查 OpenStack 的控制节点和计算节点是否已经安装 Ceph 软件包。作为 Ceph 客户端，需要在 controller、compute01、compute02 上执行以下步骤
在 controller、compute01、compute02 安装 ceph 软件包
yum -y install ceph ceph-radosgw

/*
说明：如果提示没有相关软件包，则检查离线 yum 源配置，此处列出阿里云源，在公网环境下可直接复制使用，配置如下
vim /etc/yum.repos.d/ceph.repo
[ceph]
name=Ceph packages for $basearch
baseurl=http://mirrors.aliyun.com/ceph/rpm-nautilus/el7/x86_64
enabled=1
gpgcheck=0

```
[ceph-noarch]
name=Ceph noarch packages
baseurl=http://mirrors.aliyun.com/ceph/rpm-nautilus/el7/noarch
enabled=1
gpgcheck=0

[ceph-source]
name=Ceph source packages
baseurl=http://mirrors.aliyun.com/ceph/rpm-nautilus/el7/SRPMS
enabled=1
gpgcheck=0

*/
```

3.2 创建 OpenStack 所需的存储池

在 ceph01 上创建所需的存储池，根据实际 OSD 数量修改 PG 数目

```
# ceph osd pool create volumes 128
# ceph osd pool create images 32
# ceph osd pool create backups 32
# ceph osd pool create vms 128
```

查看创建的存储池

```
# ceph osd pool ls
```

在 ceph01 同步配置文件，将 ceph01 上的配置文件同步到 controller 和 computer01、computer02 节点

```
cd /etc/ceph-cluster
ceph-deploy --overwrite-conf admin ceph01 controller compute01 compute02
```

3.3 创建 openstack 各组件需要的 ceph 用户并授权

在 ceph01 上为 cinder、glance、cinder-backup 用户创建密钥，允许其访问 Ceph 存储池

创建用户 client.cinder，对 volumes 存储池有 rwx 权限，对 vms 存储池有 rwx 权限，对 images 池有 rx 权限

```
# ceph auth get-or-create client.cinder mon 'allow r' osd 'allow class-read object_prefix rbd_children, allow rwx pool=volumes, allow rwx pool=vms, allow rx pool=images'
```

创建用户 client.glance，对 images 存储池有 rwx 权限

```
# ceph auth get-or-create client.glance mon 'allow r' osd 'allow class-read object_prefix rbd_children, allow rwx pool=images'
```

创建用户 client.cinder-backup，对 backups 存储池有 rwx 权限

```
# ceph auth get-or-create client.cinder-backup mon 'profile rbd' osd 'profile rbd pool=backups'
```

将 glance 的 keyring 保存到 controller（glance 服务所在节点）上

```
# ceph auth get-or-create client.glance | ssh controller tee /etc/ceph/ceph.client.glance.keyring
# ssh controller chown glance:glance /etc/ceph/ceph.client.glance.keyring
```

将 cinder 的 keyring 保存到（控制节点、计算节点、存储节点 服务所在节点）上

```
# ceph auth get-or-create client.cinder | ssh controller tee /etc/ceph/ceph.client.cinder.keyring #  还有 computer0{1..n}上
# ssh controller chown cinder:cinder /etc/ceph/ceph.client.cinder.keyring #  还有 computer0{1..n}上
```

```
/*
说明：节点多，可以使用 for 循环完成
for h in controller compute01 compute02
do
ceph auth get-or-create client.cinder | ssh $h tee /etc/ceph/ceph.client.cinder.keyring
ssh $h chown cinder:cinder /etc/ceph/ceph.client.cinder.keyring
done
*/
```

将 cinder-backup 的 keyring 保存到 （cinder-backup 服务所在节点，此处是 controller/computer01/computer02）上

```
# ceph auth get-or-create client.cinder-backup | ssh controller tee /etc/ceph/ceph.client.cinder-backup.keyring #  还有 computer0{1..n}上
```

```
# ssh controller chown cinder:cinder /etc/ceph/ceph.client.cinder-backup.keyring #  还有 computer0{1..n}上
#  查看
# ceph auth get-key client.cinder | ssh controller tee client.cinder.key

/*
说明：节点多，可以使用 for 循环完成
for h in controller compute01 compute02
do
ceph auth get-or-create client.cinder-backup | ssh $h tee /etc/ceph/ceph.client.cinder-backup.keyring
ssh $h chown cinder:cinder /etc/ceph/ceph.client.cinder-backup.keyring
ceph auth get-key client.cinder | ssh $h tee client.cinder.key
done

不要直接用 admin 的 key，因为不同用户要读到该密钥文件，要修改属组和属主，否则没有权限（当然可以将 ceph.client.admin.keyring 文件改为 775
允许 cinder/glance 用户读取，但不推荐）
cat /etc/ceph/ceph.client.admin.keyring
[client.admin]
key = AQCyTYZheH+KKhAAw227TN1qho/8OMhGTyL+UA==
caps mds = "allow *"
caps mgr = "allow *"
caps mon = "allow *"
caps osd = "allow *"
*/
```

3.4 添加密钥到 libvirt

在计算节点（compute01、compute02）上向 libvirt 添加秘钥，先在计算节点 compute01 上操作，生成密钥。

```
# UUID=$(uuidgen)
# cd /etc/ceph
# cat > secret.xml <<EOF

<secret ephemeral='no' private='no'>
<uuid>${UUID}</uuid>
<usage type='ceph'>
<name>client.cinder secret</name>
</usage>
</secret>
EOF

# cat secret.xml
<secret ephemeral='no' private='no'>
<uuid>aa22a048-147f-44d1-8571-8c394d924299</uuid>
<usage type='ceph'>
<name>client.cinder secret</name>
</usage>
</secret>

添加密钥到 libvirt
# virsh secret-define --file secret.xml
Secret aa22a048-147f-44d1-8571-8c394d924299 created
# virsh secret-set-value --secret ${UUID} --base64 $(cat /etc/ceph/ceph.client.cinder.keyring | grep key | awk -F ' ' '{ print $3 }')
Secret value set

/*
说明：保存此处生成的 UUID 的值，后面 Cinder 以及 Nova 的配置中需要用到，本示例中的 UUID 为：585e1823-adc0-4c25-b368-3ec81300fac2
如果添加错误，需要删除，则执行如下命令
# virsh secret-undefine aa22a048-147f-44d1-8571-8c394d924299
如果控制节点也当计算点用，则也要添加密钥
*/

查看添加后的密钥
# virsh secret-list
UUID Usage
--------------------------------------------------------------------------------
aa22a048-147f-44d1-8571-8c394d924299 ceph client.cinder secret
```

virsh secret-get-value aa22a048-147f-44d1-8571-8c394d924299
AQB5L4dhbMx1LRAAnsY7CEmS+oOQfD3ie7y5/g==

再在其他(compute)计算节点上操作：
1、将 secret.xml 拷贝到其他计算节点
scp secret.xml controller:/etc/ceph/
cp secret.xml compute02:/etc/ceph/
在 controller 节点上操作（因为 controller 也当计算节点在用）
UUID=da0a5738-671d-4070-a4c5-b6c88586e22c
virsh secret-define --file secret.xml
virsh secret-set-value --secret ${UUID} --base64 $(cat /etc/ceph/ceph.client.cinder.keyring | grep key | awk -F ' ' '{ print $3 }')

在 compute02 节点上操作
UUID=da0a5738-671d-4070-a4c5-b6c88586e22c
virsh secret-define --file secret.xml
virsh secret-set-value --secret ${UUID} --base64 $(cat /etc/ceph/ceph.client.cinder.keyring | grep key | awk -F ' ' '{ print $3 }')

3.5 配置 Glance 集成 Ceph 作为后端存储并验证

在控制节点（controller）上修改 Glance 的配置文件。

vi /etc/glance/glance-api.conf

在[DEFAULT]部分添加如下内容
[DEFAULT]
show_image_direct_url = True

在[glance_store]部分添加如下内容，需要注释原来的 stores 和 default_store
[glance_store]
#stores = file,http
#default_store = file
filesystem_store_datadir = /var/lib/glance/images/
stores = rbd,file,http
default_store = rbd
rbd_store_pool = images
rbd_store_user = glance
# rbd_store_user = admin # 直接用 admin，当在 ceph 中没有创建 glance 用户时，同时还要修改/etc/ceph/ceph.client.admin.keyring 文件权限为 775
rbd_store_ceph_conf = /etc/ceph/ceph.conf
# 镜像存储在 ceph 中会被分割成多个对象，单个对象大小，单位是 MB，该值最好是 2 的幂次方，默认是 8，chunk_size 不能太大，当镜像小于 chunk_size 时会上传失败。
rbd_store_chunk_size = 8

在[paste_deploy]部分添加如下内容，避免 images 缓存到/var/lib/glance/image-cache 目录下
[paste_deploy]
flavor = keystone

在控制节点（controller）重启 glance-api 服务。
systemctl restart openstack-glance-api.service

验证镜像上传
source /etc/keystone/admin-openrc.sh
openstack image create "cirros-0.4.0-x86_64" --file cirros-0.4.0-x86_64-disk.img --disk-format qcow2 --container-format bare --public

rbd ls images

```
[root@controller ceph]# rbd ls images
4b2f54ee-4021-43d5-ac9b-3190cb76c722
```

openstack image list

```
[root@controller ceph]# openstack image list
+--------------------------------------+---------------------+--------+
| ID                                   | Name                | Status |
+--------------------------------------+---------------------+--------+
| 4b2f54ee-4021-43d5-ac9b-3190cb76c722 | cirros-0.4.0-x86_64 | active |
+--------------------------------------+---------------------+--------+
```

rbd info images/4b2f54ee-4021-43d5-ac9b-3190cb76c722

```
[root@controller ~]# rbd info images/4b2f54ee-4021-43d5-ac9b-3190cb76c722
rbd image '4b2f54ee-4021-43d5-ac9b-3190cb76c722':
        size 12 MiB in 2 objects
        order 23 (8 MiB objects)
        snapshot_count: 1
        id: 124b5a8084a4
        block_name_prefix: rbd_data.124b5a8084a4
        format: 2
        features: layering, exclusive-lock, object-map, fast-diff, deep-flatten
        op_features:
        flags:
        create_timestamp: Mon Jan  3 19:13:58 2022
        access_timestamp: Mon Jan  3 19:37:52 2022
        modify_timestamp: Mon Jan  3 19:13:58 2022
```

3.6 配置 Cinder 集成 Ceph 作为后端存储并验证

a、在 Cinder 控制节点（controller）上，修改配置文件/etc/cinder/cinder.conf
在[DEFAULT]部分，将默认 volume 设置为 ceph
[DEFAULT]
#default_volume_type = hdd
default_volume_type = ceph

修改 default_volume_type 会重新往 cinder 库中 volume_types 表中增加一条名为 ceph 的记录
在控制节点重启 cinder-api 和 cinder-scheduler 服务
systemctl restart openstack-cinder-api.service openstack-cinder-scheduler.service

b、在 Cinder 存储节点（compute01、compute02）上，修改配置文件/etc/cinder/cinder.conf
vi /etc/cinder/cinder.conf
在[DEFAULT]部分，注释已经有的 enabled_backends
[DEFAULT]
#enabled_backends = lvm
enabled_backends = ceph,lvm

在文件末尾添加[ceph]部分
[ceph]
volume_driver = cinder.volume.drivers.rbd.RBDDriver
volume_backend_name = ceph
rbd_pool = volumes
rbd_ceph_conf = /etc/ceph/ceph.conf
# rbd snapshot 在底层会快速复制一个元信息表，但不会产生实际的数据拷贝，当从 snapshot 创建新卷时，用户可能会期望不要依赖原来的 snapshot，这个选项开启会在创建新卷时对原来的 snapshot 数据进行拷贝来生成一个不依赖于源 snapshot 的卷，默认为 false
rbd_flatten_volume_from_snapshot = false
# 克隆卷的最大嵌套（层）数，设置为 0 表示禁用克隆，降低这个值不会影响克隆深度超过新值的已有卷
rbd_max_clone_depth = 5
rbd_store_chunk_size = 4
rados_connect_timeout = -1
glance_api_version = 2
rbd_user = cinder
# 注意 uuid 后面不要复制空格
rbd_secret_uuid = aa22a048-147f-44d1-8571-8c394d924299

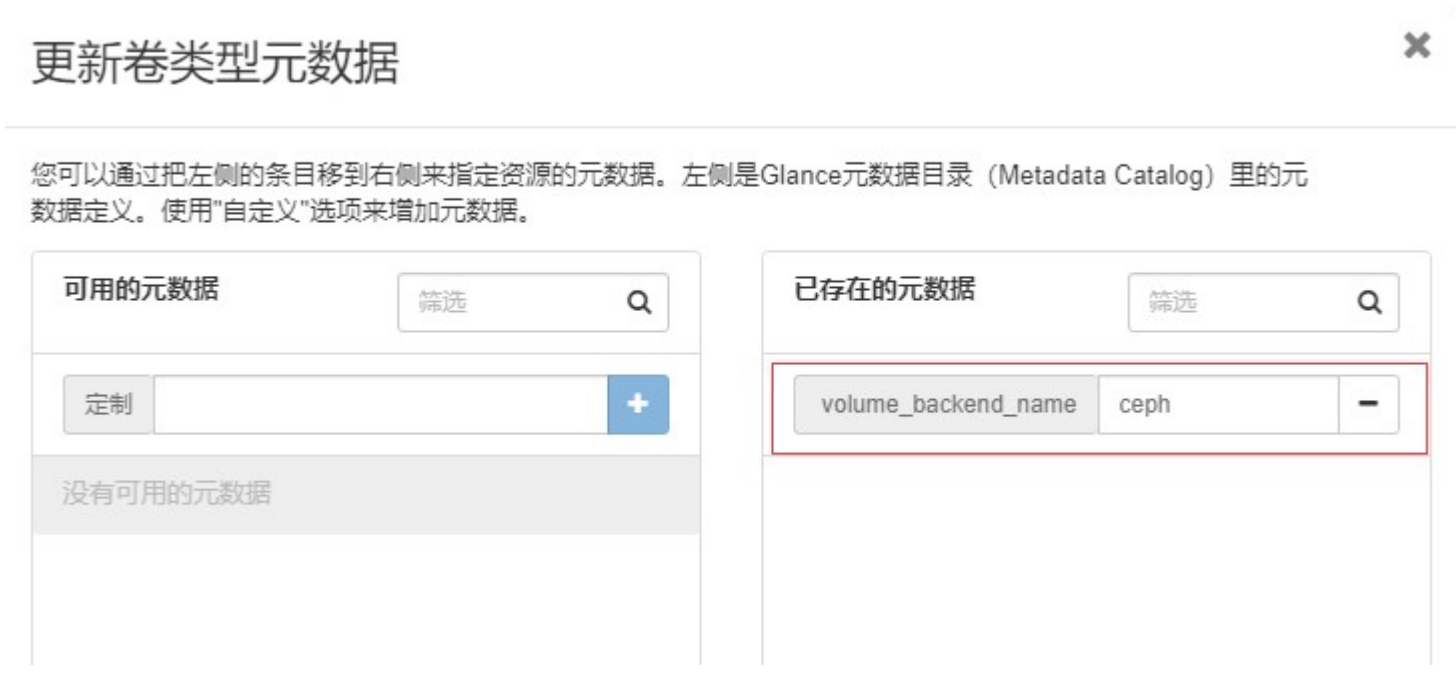
```
/*
说明：此处 rbd_secret_uuid 的值即为配置 Ceph 环境保存的 UUID 值
*/
```

在 Cinder 节点（compute01、compute02）上，重启 cinder-volume 服务
systemctl restart openstack-cinder-volume.service

c、创建实例验证
i、先创建一个 ceph 的卷类型

新增卷类型

ii、为 ceph 卷类型增加一个元数据 volume_backend_name=ceph



iii、创建实例，看是否使用了 ceph 创建卷。

3.7 配置 cinder_backup 集成 Ceph 作为后端存储并验证

cinder-backup 用于将卷（volume）备份到其他存储系统上，目前支持的备份存储系统有 swift、ceph 以及 IBM Tivoli Storage Manager(TSM)，默认为 Swift，但需要配置 swift 组件

a、在存储节点(compute01、compute02）上修改配置文件/etc/cinder/cinder.conf

在[DEFAULT]部分，增加如下内容
[DEFAULT]

backup_driver = cinder.backup.drivers.ceph.CephBackupDriver
backup_ceph_conf = /etc/ceph/ceph.conf
backup_ceph_user = cinder-backup
backup_ceph_chunk_size = 4194304
backup_ceph_pool = backups
backup_ceph_stripe_unit = 0
backup_ceph_stripe_count = 0

restore_discard_excess_bytes = true

说明：在配置参数 back_driver 时需要注释掉其他 backup_driver 的配置。
另外，在 dashboard 配置（/etc/openstack-dashboard/local_settings）中的 OPENSTACK_CINDER_FEATURES 中增加如下配置

OPENSTACK_CINDER_FEATURES = {
'enable_backup': True,
}

启动 cinder backup 服务进程并设置成开机自启动
systemctl enable openstack-cinder-backup.service
systemctl restart openstack-cinder-backup.service

重启 httpd 服务进程。
systemctl restart httpd

3.8 配置 Nova 集成 Ceph

在 Nova 计算节点（compute01、compute02）上，修改配置文件/etc/nova/nova.conf
vim /etc/nova/nova.conf

修改并新增以下内容：

[DEFAULT]
# 支持热迁移
live_migration_flag="VIR_MIGRATE_UNDEFINE_SOURCE,VIR_MIGRATE_PEER2PEER,VIR_MIGRATE_LIVE,VIR_MIGRATE_PERSIST_DEST,VIR_MIGRATE_TUNNELLED"
[libvirt]
# 虚拟机模拟 openstack 可能需要将 virt_type 设置为 qemu，否则创建虚拟机后，一直停在 GRUB Loading stage2
# virt_type = qemu
inject_partition=-2
virt_type = kvm
images_type = rbd
images_rbd_pool = vms
images_rbd_ceph_conf = /etc/ceph/ceph.conf
disk_cachemodes="network=writeback"
rbd_user = cinder
# 此处 uuid 与 cinder.conf 是同一个
rbd_secret_uuid = aa22a048-147f-44d1-8571-8c394d924299

在 Nova 计算节点（controller/computer）上，重启 nova-compute 服务。
systemctl restart openstack-nova-compute.service
说明：nova 集成 ceph，允许从 ceph 卷上启动实例。当 cinder 挂载或者卸载块设备时，libvirt 进程需要有访问 ceph 集群的权限 。