**Ceph分布式存储安装部署文档**

# 安装基础环境

yum install -y <https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm>

vim /etc/yum.repos.d/ceph.repo

[ceph-noarch]

name=Ceph noarch packages

baseurl=https://download.ceph.com/rpm-jewel/el7/noarch

enabled=1

gpgcheck=1

type=rpm-md

gpgkey=https://download.ceph.com/keys/release.asc

[ceph-noarch]

name=Ceph noarch packages

baseurl=http://mirrors.163.com/ceph/rpm-jewel/el7

enabled=1

gpgcheck=1

type=rpm-md

gpgkey=http://mirrors.163.com/ceph/keys/release.asc

[Ceph]

name=Ceph noarch packages

baseurl=http://mirrors.163.com/ceph/rpm-jewel/el7

enabled=1

gpgcheck=1

type=rpm-md

gpgkey=http://mirrors.163.com/ceph/keys/release.asc

[Ceph-noarch]

name=Ceph noarch packages

baseurl=http://mirrors.163.com/ceph/rpm-jewel/el7/noarch

enabled=1

gpgcheck=1

type=rpm-md

gpgkey=http://mirrors.163.com/ceph/keys/release.asc

[ceph-source]

name=Ceph source packages

baseurl=http://mirrors.163.com/ceph/rpm-jewel/el7/SRPMS

enabled=1

gpgcheck=1

type=rpm-md

gpgkey=http://mirrors.163.com/ceph/keys/release.asc

yum update && sudo yum install ceph-deploy

# 安装NTP服务

确保ceph的monitor没有时间差，避免一些问题的出现

[root@compute-1 ~]# yum install ntp ntpdate ntp-doc -y

确保每个节点都安装了ssh服务

yum install openssh-server

# 管理节点添加hosts

编辑/etc/hosts 添加hostname到ceph-admin节点上：

[root@compute-1 ~]# cat /etc/hosts

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

10.12.15.220 compute-1

10.12.15.223 compute-2

10.12.15.221 compute-3

[root@compute-1 ~]#

# SSH互信

命令为：ssh-keygen，连续按enter键；

输出内容如下：

Generating public/private key pair.

Enter file in which to save the key (/ceph-admin/.ssh/id\_rsa):

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /ceph-admin/.ssh/id\_rsa.

Your public key has been saved in /ceph-admin/.ssh/id\_rsa.pub.

复制密钥到其他节点上

ssh-copy-id root@compute-1

ssh-copy-id root@compute-2

ssh-copy-id root@compute-3

# 修改~/.ssh/config

确保ceph-deploy管理其他节点时，避免每次输入密码；

[root@compute-1 ~]# cat ~/.ssh/config

Host compute-1

Hostname compute-1

User root

Host compute-2

Hostname compute-2

User root

Host compute-3

Hostname compute-3

User root

[root@compute-1 ~]#

# TTY

运行： visudo

找到这个 Defaults requiretty

改为：Defaults:ceph !requiretty 或者 注释掉

# 关闭SELINUX

[root@compute-1 ~]# setenforce 0

[root@compute-1 ~]# vim /etc/sysconfig/selinux

[root@compute-1 ~]# cat /etc/sysconfig/selinux

# This file controls the state of SELinux on the system.

# SELINUX= can take one of these three values:

# enforcing - SELinux security policy is enforced.

# permissive - SELinux prints warnings instead of enforcing.

# disabled - No SELinux policy is loaded.

SELINUX=disabled

# SELINUXTYPE= can take one of three two values:

# targeted - Targeted processes are protected,

# minimum - Modification of targeted policy. Only selected processes are protected.

# mls - Multi Level Security protection.

SELINUXTYPE=targeted

[root@compute-1 ~]#

# 清理ceph集群

比如安装ceph集群时，遇到问题时，可以使用以下命令来推到ceph集群，然后重新做集群。

ceph-deploy purge compute1 compute2 compute3

ceph-deploy purgedata compute1 compute2 compute3

ceph-deploy forgetkeys

清理完所有数据后，为了保证重新安装ceph集群，需要重启每个节点。

# 创建集群

创建集群目录

[root@compute-1 ~]# **mkdir my-cluster**

[root@compute-1 ~]# ll

total 16

-rw-------. 1 root root 6913 Mar 1 02:45 anaconda-ks.cfg

drwxr-xr-x. 2 root root 6 May 17 08:04 my-cluster

-rw-------. 1 root root 6580 Mar 1 02:45 original-ks.cfg

[root@compute-1 ~]# cd my-cluster/

[root@compute-1 my-cluster]# ll

total 0

[root@compute-1 my-cluster]# **ceph-deploy new compute1 compute2 compute3**

[ceph\_deploy.conf][DEBUG ] found configuration file at: /root/.cephdeploy.conf

[ceph\_deploy.cli][INFO ] Invoked (1.5.37): /usr/bin/ceph-deploy new compute-1

[ceph\_deploy.cli][INFO ] ceph-deploy options:

[ceph\_deploy.cli][INFO ] username : None

[ceph\_deploy.cli][INFO ] func : <function new at 0x1687410>

[ceph\_deploy.cli][INFO ] verbose : False

[ceph\_deploy.cli][INFO ] overwrite\_conf : False

[ceph\_deploy.cli][INFO ] quiet : False

[ceph\_deploy.cli][INFO ] cd\_conf : <ceph\_deploy.conf.cephdeploy.Conf instance at 0x16f15a8>

[ceph\_deploy.cli][INFO ] cluster : ceph

[ceph\_deploy.cli][INFO ] ssh\_copykey : True

[ceph\_deploy.cli][INFO ] mon : ['compute-1']

[ceph\_deploy.cli][INFO ] public\_network : None

[ceph\_deploy.cli][INFO ] ceph\_conf : None

[ceph\_deploy.cli][INFO ] cluster\_network : None

[ceph\_deploy.cli][INFO ] default\_release : False

[ceph\_deploy.cli][INFO ] fsid : None

[ceph\_deploy.new][DEBUG ] Creating new cluster named ceph

[ceph\_deploy.new][INFO ] making sure passwordless SSH succeeds

[compute-1][DEBUG ] connected to host: compute-1

[compute-1][DEBUG ] detect platform information from remote host

[compute-1][DEBUG ] detect machine type

[compute-1][DEBUG ] find the location of an executable

[compute-1][INFO ] Running command: /usr/sbin/ip link show

[compute-1][INFO ] Running command: /usr/sbin/ip addr show

[compute-1][DEBUG ] IP addresses found: [u'10.12.15.220']

[ceph\_deploy.new][DEBUG ] Resolving host compute-1

[ceph\_deploy.new][DEBUG ] Monitor compute-1 at 10.12.15.220

[ceph\_deploy.new][DEBUG ] Monitor initial members are ['compute-1']

[ceph\_deploy.new][DEBUG ] Monitor addrs are ['10.12.15.220']

[ceph\_deploy.new][DEBUG ] Creating a random mon key...

[ceph\_deploy.new][DEBUG ] Writing monitor keyring to ceph.mon.keyring...

[ceph\_deploy.new][DEBUG ] Writing initial config to ceph.conf...

[root@compute-1 my-cluster]#

[root@compute-1 my-cluster]# ll

total 12

-rw-r--r--. 1 root root 199 May 17 08:05 ceph.conf

-rw-r--r--. 1 root root 2957 May 17 08:05 ceph-deploy-ceph.log

-rw-------. 1 root root 73 May 17 08:05 ceph.mon.keyring

[root@compute-1 my-cluster]# vim ceph

ceph.conf ceph-deploy-ceph.log ceph.mon.keyring

[root@compute-1 my-cluster]# vim ceph.conf ceph

编辑当前文件夹中的ceph.conf 中，并添加下一行：

默认数字从3改为2，因此ceph只有两个OSD，就可以达到active+clean状态，

[root@compute-1 my-cluster]# cat ceph.conf

[global]

fsid = 84b6ce28-6eef-4342-b180-102f627e0061

mon\_initial\_members = compute-1,compute-2,compute-3

mon\_host = 10.12.15.220,

auth\_cluster\_required = cephx

auth\_service\_required = cephx

auth\_client\_required = cephx

**osd pool default size = 2**

**public network = 10.0.1.0/24**

**cluster network = 10.0.16.0/24**

**mon clock drift allowed = 2**

**mon clock drift warn backoff = 30**

**osd journal size = 20480**

# 国内源部署Ceph

这里以安装最新版本的Jewel为例，由于Jewel版本中已经不提供el6的镜像源，所以只能使用CentOS 7以上版本进行安装。我们并不需要在repos里增加相应的源，只需要设置环境变量，即可让ceph-deploy使用国内源，具体过程如下：

CentOS:

export CEPH\_DEPLOY\_REPO\_URL=http://mirrors.163.com/ceph/rpm-jewel/el7

export CEPH\_DEPLOY\_GPG\_URL=http://mirrors.163.com/ceph/keys/release.asc

export CEPH\_DEPLOY\_REPO\_URL=http://mirrors.plcloud.com/ceph/rpm-jewel/el7

export CEPH\_DEPLOY\_GPG\_URL=http://mirrors.plcloud.com/ceph/keys/release.asc

Ubuntu:

export CEPH\_DEPLOY\_REPO\_URL=http://mirrors.163.com/ceph/debian-jewel

export CEPH\_DEPLOY\_GPG\_URL=http://mirrors.163.com/ceph/keys/release.asc

由于网络方面的原因，Ceph的部署经常受到干扰，通常为了加速部署，基本上大家都是将Ceph的源同步到本地进行安装。根据Ceph中国社区的统计，当前已经有国内的网站定期将Ceph安装源同步，极大的方便了我们的测试。本文就是介绍如何使用国内源，加速ceph-deploy部署Ceph集群。

关于国内源

根据Ceph中国社区的统计，国内已经有四家网站开始同步Ceph源，分别是：

网易镜像源http://mirrors.163.com/ceph

阿里镜像源http://mirrors.aliyun.com/ceph

中科大镜像源http://mirrors.ustc.edu.cn/ceph

宝德镜像源 http://mirrors.plcloud.com/ceph

**国内源分析**

以163为例，是以天为单位向回同步Ceph源，完全可以满足大多数场景的需求，同步的源也非常全，包含了calamari，debian和rpm的全部源，最近几个版本的源也能从中找到。

**安装指定版本的Ceph**

这里以安装最新版本的Jewel为例，由于Jewel版本中已经不提供el6的镜像源，所以只能使用CentOS 7以上版本进行安装。我们并不需要在repos里增加相应的源，只需要设置环境变量，即可让ceph-deploy使用国内源，具体过程如下：

export CEPH\_DEPLOY\_REPO\_URL=http://mirrors.163.com/ceph/rpm-jewel/el7

export CEPH\_DEPLOY\_GPG\_URL=http://mirrors.163.com/ceph/keys/release.asc

之后的过程就没有任何区别了：

# Create monitor node

ceph-deploy new node1 node2 node3

# Software Installation

ceph-deploy install deploy node1 node2 node3

# Gather keys

ceph-deploy mon create-initial

# Ceph deploy parepare and activate

ceph-deploy osd prepare node1:/dev/sdb node2:/dev/sdb node3:/dev/sdb

ceph-deploy osd activate node1:/var/lib/ceph/osd/ceph-0 node2:/var/lib/ceph/osd/ceph-1 node3:/var/lib/ceph/osd/ceph-2

# Make 3 copies by default

echo "osd pool default size = 3" | tee -a $HOME/ceph.conf

# Copy admin keys and configuration files

ceph-deploy --overwrite-conf admin deploy node1 node2 node3

这样就可以很快速的使用国内源创建出Ceph集群，希望能对大家日常的使用提供便捷。

# 安装ceph

在管理节点上，使用ceph-deploy，在每个节点上安装部署ceph。

ceph-deploy install {ceph-node}[{ceph-node} ...]

ceph-deploy install compute1 compute2 compute3

然后初始化ceph monitor，后生成一下必要文件

[root@compute1 cluster]# ceph-deploy mon create-initial

[root@compute1 cluster]# ll

总用量 1200

-rw------- 1 root root 113 6月 2 08:37 ceph.bootstrap-mds.keyring

-rw------- 1 root root 113 6月 2 08:37 ceph.bootstrap-osd.keyring

-rw------- 1 root root 113 6月 2 08:37 ceph.bootstrap-rgw.keyring

-rw------- 1 root root 129 6月 2 08:37 ceph.client.admin.keyring

-rw-r--r-- 1 root root 414 6月 2 08:26 ceph.conf

-rw-r--r-- 1 root root 1466 6月 2 08:22 ceph.confbk

-rw-r--r-- 1 root root 1195977 6月 2 08:37 ceph-deploy-ceph.log

-rw------- 1 root root 73 6月 2 08:25 ceph.mon.keyring

-rw-r--r-- 1 root root 143 5月 25 15:57 repo.sh

[root@compute1 cluster]#

# 准备并激活ceph osd

## 方案一：

每个磁盘分为ceph的journal和osd两个分区：

根据预先准备好的磁盘划分

ceph-deploy osd prepare compute1:/dev/sde compute1:/dev/sdf compute2:/dev/sde compute2:/dev/sdf compute3:/dev/sde compute3:/dev/sdf

准备完osd磁盘以后，查看磁盘划分状态：

[root@compute1 cluster]# ceph-disk list

/dev/dm-0 swap, swap

/dev/dm-1 other, xfs, mounted on /

/dev/dm-2 other, xfs, mounted on /home

/dev/sda other, xfs

/dev/sdb other, ext4

/dev/sdc other, ext4

/dev/sdd other, ext4

/dev/sde :

/dev/sde2 ceph journal, for /dev/sde1

/dev/sde1 ceph data, active, cluster ceph, osd.0, journal /dev/sde2

/dev/sdf :

/dev/sdf2 ceph journal, for /dev/sdf1

/dev/sdf1 ceph data, active, cluster ceph, osd.1, journal /dev/sdf2

/dev/sdg :

/dev/sdg2 other, LVM2\_member

/dev/sdg1 other, xfs, mounted on /boot

/dev/sr0 other, unknown

[root@compute1 cluster]#

激活数据盘：磁盘的第一个分区是数据盘

ceph-deploy osd activate compute1:/dev/sde1 compute1:/dev/sdf1 compute2:/dev/sde1 compute2:/dev/sdf1 compute3:/dev/sde1 compute3:/dev/sdf1

ceph-deploy osd prepare compute1:/dev/sda compute1:/dev/sdb compute1:/dev/sdc compute1:/dev/sdd

ceph-deploy osd prepare compute2:/dev/sda compute2:/dev/sdb compute2:/dev/sdc compute2:/dev/sdd

ceph-deploy osd prepare compute3:/dev/sda compute3:/dev/sdb compute3:/dev/sdc compute3:/dev/sdd

ceph-deploy osd activate compute1:/dev/sda1 compute1:/dev/sdb1 compute1:/dev/sdc1 compute1:/dev/sdd1

ceph-deploy osd activate compute2:/dev/sda1 compute2:/dev/sdb1 compute2:/dev/sdc1 compute2:/dev/sdd1

ceph-deploy osd activate compute3:/dev/sda1 compute3:/dev/sdb1 compute3:/dev/sdc1 compute3:/dev/sdd1

## 方案二：

SSD作为osd的journal盘，SATA作为osd数据盘，一个SSD journal盘对应4个OSD SATA盘

**注意: ceph-deploy osd create 不管用**

ceph-deploy osd prepare compute1:sda:/dev/sde compute1:sdb:/dev/sde compute1:sdc:/dev/sde compute1:sdd:/dev/sde

ceph-deploy osd prepare compute2:sda:/dev/sde compute2:sdb:/dev/sde compute2:sdc:/dev/sde compute2:sdd:/dev/sde

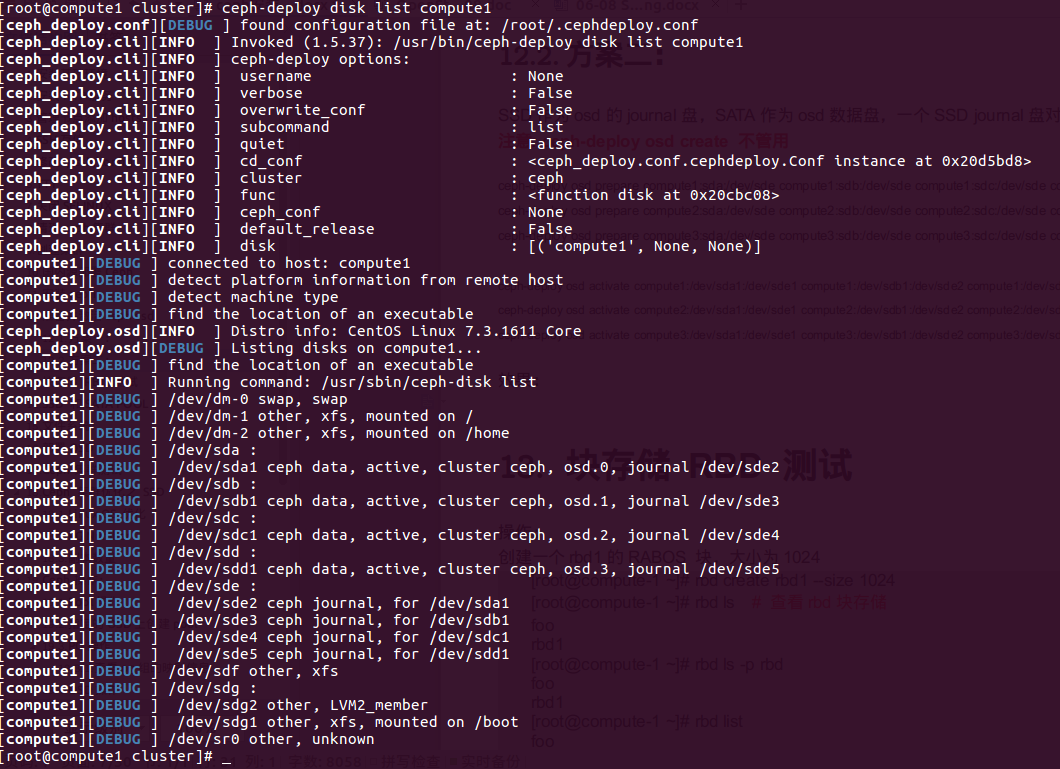
ceph-deploy osd prepare compute3:sda:/dev/sde compute3:sdb:/dev/sde compute3:sdc:/dev/sde compute3:sdd:/dev/sde

ceph-deploy osd activate compute1:/dev/sda1:/dev/sde1 compute1:/dev/sdb1:/dev/sde2 compute1:/dev/sdc1:/dev/sde3 compute1:/dev/sdd1:/dev/sde4

ceph-deploy osd activate compute2:/dev/sda1:/dev/sde1 compute2:/dev/sdb1:/dev/sde2 compute2:/dev/sdc1:/dev/sde3 compute2:/dev/sdd1:/dev/sde4

ceph-deploy osd activate compute3:/dev/sda1:/dev/sde1 compute3:/dev/sdb1:/dev/sde2 compute3:/dev/sdc1:/dev/sde3 compute3:/dev/sdd1:/dev/sde4

效果：



# CephFS文件系统

**安装mds:**

[root@ceph-1 cluster]# pwd

/home/cluster

[root@ceph-1 cluster]# ceph-deploy mds create ceph-1

**查看状态：**

[root@ceph-1 cluster]# systemctl status ceph-mds@ceph-1

**创建pool**

[root@ceph-1 cluster]# ceph osd pool create cephfs\_data 100

[root@ceph-1 cluster]# ceph osd pool create cephfs\_metadata 100

[root@ceph-1 cluster]# ceph osd lspools **# 查看创建的pool**

1 rbd,2 .rgw.root,3 default.rgw.control,4 default.rgw.data.root,5 default.rgw.gc,6 default.rgw.log,7 default.rgw.users.uid,8 **cephfs\_data**,9 **cephfs\_metadata,**

[root@ceph-1 cluster]#

**安装ceph-fuse**

[root@ceph-1 cluster]# yum install ceph-fuse -y

[root@ceph-1 cluster]# mkdir /mnt/mycephfs

**挂载文件系统：**

[root@ceph-1 cluster]# **ceph-fuse -m 10.12.16.43:6789 /mnt/mycephfs/**

ceph-fuse[11734]: starting ceph client

2017-09-13 08:48:04.056698 7fde1460dec0 -1 init, newargv = 0x7fde1f894840 newargc=11

ceph-fuse[11734]: starting fuse

Aborted (core dumped)

[root@ceph-1 cluster]# **df -lh**

Filesystem Size Used Avail Use% Mounted on

/dev/vda1 40G 6.3G 34G 16% /

devtmpfs 1.9G 0 1.9G 0% /dev

tmpfs 1.9G 0 1.9G 0% /dev/shm

tmpfs 1.9G 17M 1.9G 1% /run

tmpfs 1.9G 0 1.9G 0% /sys/fs/cgroup

/dev/vdb1 15G 37M 15G 1% /var/lib/ceph/osd/ceph-0

/dev/vdc1 15G 41M 15G 1% /var/lib/ceph/osd/ceph-1

tmpfs 380M 0 380M 0% /run/user/0

ceph-fuse 90G 236M 90G 1% /mnt/mycephfs

[root@ceph-1 cluster]# **cd /mnt/mycephfs/**

[root@ceph-1 mycephfs]# touch xxx.txt

[root@ceph-1 mycephfs]# ll

total 0

-rw-r--r-- 1 root root 0 Sep 13 08:48 xxx.txt

[root@ceph-1 mycephfs]#

**查看cephfs 文件系统：**

[root@ceph-1 cluster]# **ceph fs get cephfs**

Filesystem '**cephfs**' (1)

fs\_name cephfs

epoch 5

flags 0

created 2017-09-13 08:38:16.263390

modified 2017-09-13 08:38:16.263390

tableserver 0

root 0

session\_timeout 60

session\_autoclose 300

max\_file\_size 1099511627776

last\_failure 0

last\_failure\_osd\_epoch 0

compat compat={},rocompat={},incompat={1=base v0.20,2=client writeable ranges,3=default file layouts on dirs,4=dir inode in separate object,5=mds uses versioned encoding,6=dirfrag is stored in omap,8=file layout v2}

max\_mds 1

in 0

up {0=14126}

failed

damaged

stopped

data\_pools 8

metadata\_pool 9

inline\_data disabled

14126: 10.12.16.43:6808/2416388813 'ceph-1' mds.0.4 up:active seq 132

[root@ceph-1 cluster]#

**删除cephfs文件系统：**

[root@ceph-1 cluster]# **umount /mnt/mycephfs**

[root@ceph-1 cluster]# **ceph fs rm cephfs**

Error EINVAL: all MDS daemons must be inactive before removing filesystem

[root@ceph-1 cluster]# **systemctl stop ceph-mds@ceph-1**

[root@ceph-1 cluster]# **ceph fs rm cephfs --yes-i-really-mean-it**

[root@ceph-1 cluster]# **ceph fs ls**

No filesystems enabled

[root@ceph-1 cluster]#

[root@ceph-1 cluster]# **rados lspools**

rbd

.rgw.root

default.rgw.control

default.rgw.data.root

default.rgw.gc

default.rgw.log

default.rgw.users.uid

cephfs\_data

cephfs\_metadata

[root@ceph-1 cluster]# **ceph osd pool rm cephfs\_data cephfs\_data --yes-i-really-really-mean-it**

pool 'cephfs\_data' removed

[root@ceph-1 cluster]# **ceph osd pool rm cephfs\_metadata cephfs\_metadata --yes-i-really-really-mean-it**

pool 'cephfs\_metadata' removed

[root@ceph-1 cluster]# **rados lspools**

rbd

.rgw.root

default.rgw.control

default.rgw.data.root

default.rgw.gc

default.rgw.log

default.rgw.users.uid

[root@ceph-1 cluster]#

# Radosgw对象存储

从管理节点安装radosgw包

[root@ceph-1 cluster]# ceph-deploy install --rgw ceph-1

创建radosgw的进程

[root@ceph-1 cluster]# ceph-deploy rge create ceph-1

配置文件添加：

[client.radosgw.ceph-1]

# host = ceph-1

#keyring = /etc/ceph/keyring.radosgw.gateway

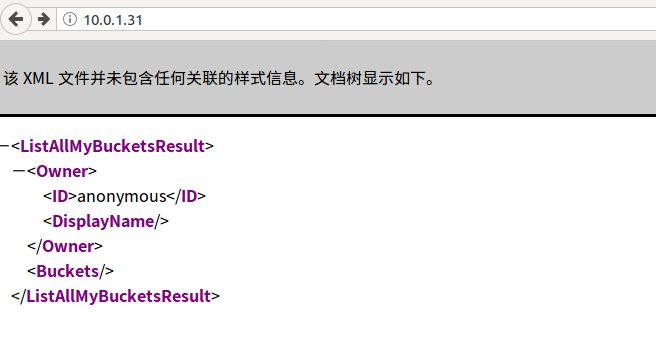
#log file = /var/log/radosgw.log

rgw\_frontends = "civetweb port=80" // 默认端口为7480

然后重启服务：

[root@ceph-1 cluster]# systemctl restart ceph-radosgw@rgw.ceph-1

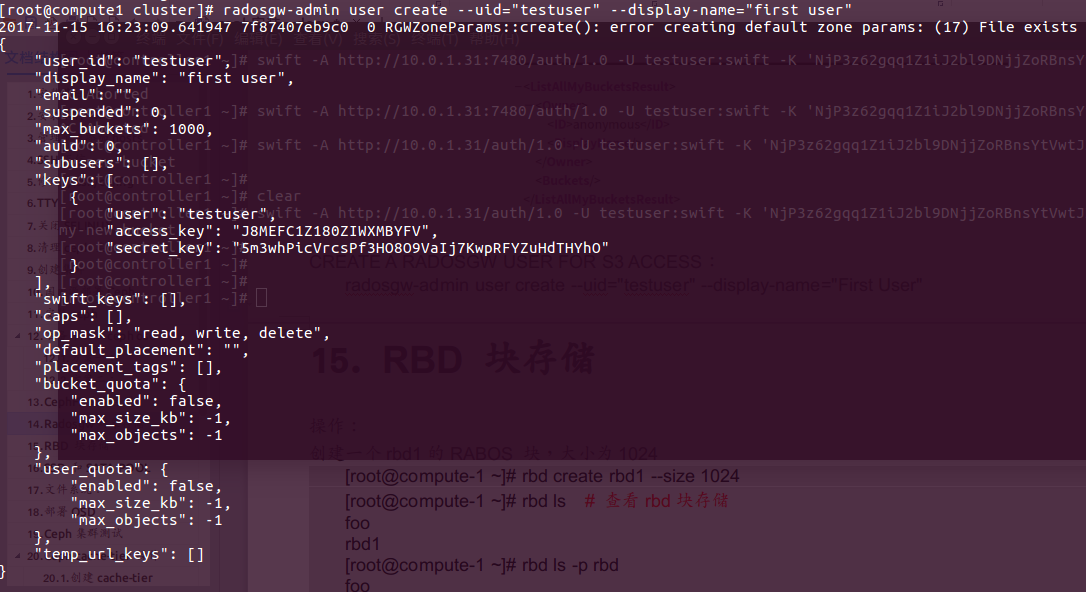
浏览器访问：



CREATE A RADOSGW USER FOR S3 ACCESS：

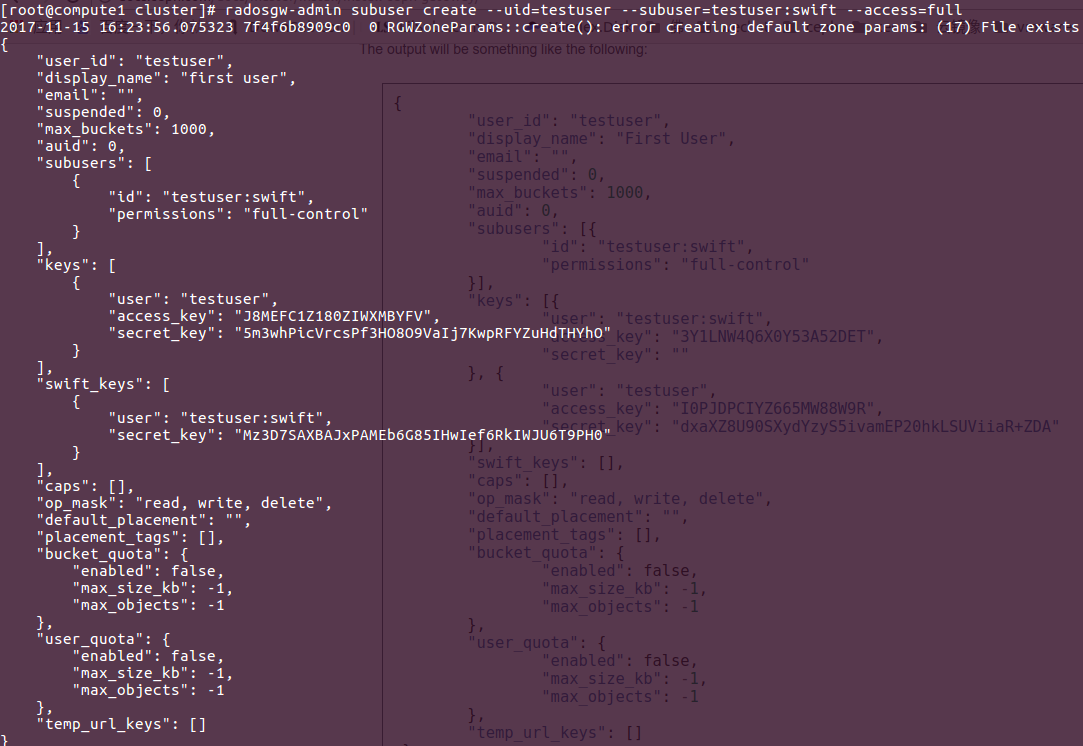
radosgw-admin user create --uid="testuser" --display-name="First User"

**The values of keys->access\_key and keys->secret\_key are needed for access validation.**



CREATE A SWIFT USER

**radosgw-admin subuser create --uid=testuser --subuser=testuser:swift --access=full**



Create the secret key:

radosgw-admin key create --subuser=testuser:swift --key-type=swift --gen-secret



**测试：**

[root@compute1 cluster]# **sudo yum install python-boto**

[root@compute1 cluster]# **cat s3test.py**

import boto.s3.connection

access\_key = **'J8MEFC1Z180ZIWXMBYFV'**

secret\_key = **'5m3whPicVrcsPf3HO8O9VaIj7KwpRFYZuHdTHYhO'**

conn = boto.connect\_s3(

aws\_access\_key\_id=access\_key,

aws\_secret\_access\_key=secret\_key,

host=**'compute1'**, port=**80**,

is\_secure=False, calling\_format=boto.s3.connection.OrdinaryCallingFormat(),

)

bucket = conn.create\_bucket('my-new-bucket')

for bucket in conn.get\_all\_buckets():

print "{name} {created}".format(

name=bucket.name,

created=bucket.creation\_date,

)

[root@compute1 cluster]#

运行并查看结果：

[root@compute1 cluster]# python s3test.py

my-new-bucket 2017-11-15T08:28:42.454Z

[root@compute1 cluster]#

在controller节点运行，测试swift的访问： ***因为配置文件中，访问端口已经设置为80.***

[root@controller1 ~]# swift -A http://**10.0.1.31**/auth/1.0 -U testuser:swift \

-K '*NjP3z62gqq1Z1iJ2bl9DNjjZoRBnsYtVwtJZtpcy*' list

**my-new-bucket**

[root@controller1 ~]#

# RBD 块存储

操作：

创建一个rbd1的RABOS 块，大小为1024

[root@compute-1 ~]# rbd create rbd1 --size 1024

[root@compute-1 ~]# rbd ls # 查看rbd块存储

foo

rbd1

[root@compute-1 ~]# rbd ls -p rbd

foo

rbd1

[root@compute-1 ~]# rbd list

foo

rbd1

[root@compute-1 ~]#

[root@compute-1 ~]# rbd --image rbd1 info # 查看rbd1的相关信息

rbd image 'rbd1':

size 1024 MB in 256 objects

order 22 (4096 kB objects)

block\_name\_prefix: rbd\_data.430732ae8944a

format: 2

features: layering, exclusive-lock, object-map, fast-diff, deep-flatten

flags:

[root@compute-1 ~]#

# 删除创建的块存储

[root@compute-1 ~]# rbd remove rbd1

Removing image: 100% complete...done.

使用这个块存储，必须现在做mapping，但是做mapping时，注意一下内容：

|  |
| --- |
| 测试信息如下：  #新建rbd 块：  rbd create rbd/test2 --size 10G --image-format 2  #查看块信息：  rbd ls  #将rbd 块映射到主机：  rbd map rbd/test2  #报错内容如下  rbd: sysfs write failed  RBD image feature set mismatch. You can disable features unsupported by the kernel with "rbd feature disable".  In some cases useful info is found in syslog - try "dmesg | tail" or so.  rbd: map failed: (6) No such device or address  故障排查：  rbd 块ceph 支持两种格式：  rbd create rbd/test2 --size 10G --image-format 2  --image-format *format-id*  选择用哪个对象布局，默认为 1 。  format 1 - 新建 rbd 映像时使用最初的格式。此格式兼容所有版本的 librbd 和内核模块，但是不支持较新的功能，像克隆。  format 2 - 使用第二版 rbd 格式， librbd 和 3.11 版以上内核模块才支持（除非是分拆的模块）。此格式增加了克隆支持，使得扩展更容易，还允许以后增加新功能。  为使用rbd 块新特性，使用格式2，在map 时发生以上报错：  查找官网相关资料，找到信息如下：  笔者安装的是jewel 版本，新建rbd块指定格式2，默认格式2的rbd 块支持如下特性，默认全部开启；  layering: 支持分层  striping: 支持条带化 v2  exclusive-lock: 支持独占锁  object-map: 支持对象映射（依赖 exclusive-lock ）  fast-diff: 快速计算差异（依赖 object-map ）  deep-flatten: 支持快照扁平化操作  journaling: 支持记录 IO 操作（依赖独占锁）  笔者使用系统为centos7.2 ，内核版本 3.10.0-327.18.2.el7.x86\_64,根据报错内容提示可知，服务器系统内核版本，不支持有些格式2 的新特性导致。可以使用  --image-feature   选项指定使用特性，不用全部开启。我们的需求仅需要使用快照等特性，开启layering即可，配置方式如下：  rbd create rbd/test1 --size 10G --image-format 2 --image-feature  layering  rbd ls  rbd map rbd/test1   #可以正常映射；  #经测试，内核版本 3.10，仅支持此特性（layering），其它特性需要使用更高版本内核，或者从新编译内核加载特性模块才行。 |

## 创建rbd块,并做mapping

[root@compute-1 ~]# ceph osd lspools

0, rbd

[root@compute-1 ~]# ceph osd pool create baater 300 300

[root@compute-1 ~]# rbd create baater/lily --size 100G --image-format **2** --image-feature **layering**

[root@compute1 ceph-disk0]# ceph osd lspools # 查看pool，默认有个rbd 的pool，实例在默认pool中

0 rbd, 1 baater ,

[root@compute-1 ~]# rbd info --image baater/lily

rbd image lily :

size 10240 MB in 2560 objects

order 22 (4096 kB objects)

block\_name\_prefix: rbd\_data.10dd238e1f29

format: 2

features: layering

flags:

[root@compute-1 ~]#

[root@compute-1 ~]# rbd ls baater

lily

[root@compute-1 ~]# rbd map --image baater/lily

/dev/rbd0

[root@compute-1 ~]# rbd showmapped

id pool image snap device

0 baater lily - /dev/rbd0

[root@compute-1 ~]#

[root@compute-1 ~]# **rbd unmap /dev/rbd0 # 删除**

[root@compute-1 ~]# fdisk -l /dev/rbd0

磁盘 /dev/rbd0：10.7 GB, 10737418240 字节，20971520 个扇区

Units = 扇区 of 1 \* 512 = 512 bytes

扇区大小(逻辑/物理)：512 字节 / 512 字节

I/O 大小(最小/最佳)：4194304 字节 / 4194304 字节

[root@compute-1 ~]#

[root@compute-1 ~]# mkfs.xfs /dev/rbd0

meta-data=/dev/rbd0 isize=512 agcount=17, agsize=162816 blks

= sectsz=512 attr=2, projid32bit=1

= crc=1 finobt=0, sparse=0

data = bsize=4096 blocks=2621440, imaxpct=25

= sunit=1024 swidth=1024 blks

naming =version 2 bsize=4096 ascii-ci=0 ftype=1

log =internal log bsize=4096 blocks=2560, version=2

= sectsz=512 sunit=8 blks, lazy-count=1

realtime =none extsz=4096 blocks=0, rtextents=0

[root@compute-1 ~]# mkdir /mnt/ceph-disk0

[root@compute-1 ~]# mount /dev/rbd0 /mnt/ceph-disk0

[root@compute-1 ~]# df -lh

文件系统 容量 已用 可用 已用% 挂载点

/dev/vda1 100G 1.7G 99G 2% /

devtmpfs 901M 0 901M 0% /dev

tmpfs 920M 0 920M 0% /dev/shm

tmpfs 920M 17M 904M 2% /run

tmpfs 920M 0 920M 0% /sys/fs/cgroup

tmpfs 184M 0 184M 0% /run/user/0

/dev/vdb1 5.0G 51M 5.0G 1% /var/lib/ceph/osd/ceph-0

/dev/vdc1 5.0G 35M 5.0G 1% /var/lib/ceph/osd/ceph-1

/dev/rbd0 10G 33M 10G 1% /mnt/ceph-disk0

[root@compute-1 ~]#

# SSD 和SATA POOL

[root@compute1 ~]# ceph osd lspools

0, rbd

[root@compute1 ~]# ceph osd pool create ssd-pool 300 300

[root@compute1 ~]# ceph osd pool create sata-pool 300 300

[root@compute1 ~]# ceph osd lspools

0 rbd,3 ssd-pool,4 sata-pool,

[root@compute1 ~]# rbd create ssd-pool/ssd --size 100G --image-format 2 --image-feature layering

[root@compute1 ~]# ceph osd lspools

0 rbd,3 ssd-pool,4 sata-pool,

[root@compute1 ~]# rbd ls ssd-pool

ssd

[root@compute1 ~]# rbd ls sata-pool

[root@compute1 ~]# rbd create sata-pool/sata --size 100G --image-format 2 --image-feature layering

[root@compute1 ~]# rbd ls sata-pool

sata

[root@compute1 ~]#

[root@compute1 ~]# rbd showmapped

[root@compute1 ~]# rbd map ssd

rbd: sysfs write failed

In some cases useful info is found in syslog - try "dmesg | tail" or so.

rbd: map failed: (2) No such file or directory

[root@compute1 ~]# rbd map ssd-pool/ssd

/dev/rbd0

[root@compute1 ~]# rbd showmapped

id pool image snap device

0 ssd-pool ssd - /dev/rbd0

[root@compute1 ~]# rbd map sata-pool/sata

/dev/rbd1

[root@compute1 ~]# rbd showmapped

id pool image snap device

0 ssd-pool ssd - /dev/rbd0

1 sata-pool sata - /dev/rbd1

[root@compute1 ~]#

[root@compute1 ~]# fdisk -l /dev/rbd0

[root@compute1 ~]# mkfs.xfs /dev/rbd0

meta-data=/dev/rbd0 isize=512 agcount=17, agsize=1637376 blks

= sectsz=512 attr=2, projid32bit=1

= crc=1 finobt=0, sparse=0

data = bsize=4096 blocks=26214400, imaxpct=25

= sunit=1024 swidth=1024 blks

naming =version 2 bsize=4096 ascii-ci=0 ftype=1

log =internal log bsize=4096 blocks=12800, version=2

= sectsz=512 sunit=8 blks, lazy-count=1

realtime =none extsz=4096 blocks=0, rtextents=0

[root@compute1 ~]# mkfs.xfs /dev/rbd1

meta-data=/dev/rbd1 isize=512 agcount=17, agsize=1637376 blks

= sectsz=512 attr=2, projid32bit=1

= crc=1 finobt=0, sparse=0

data = bsize=4096 blocks=26214400, imaxpct=25

= sunit=1024 swidth=1024 blks

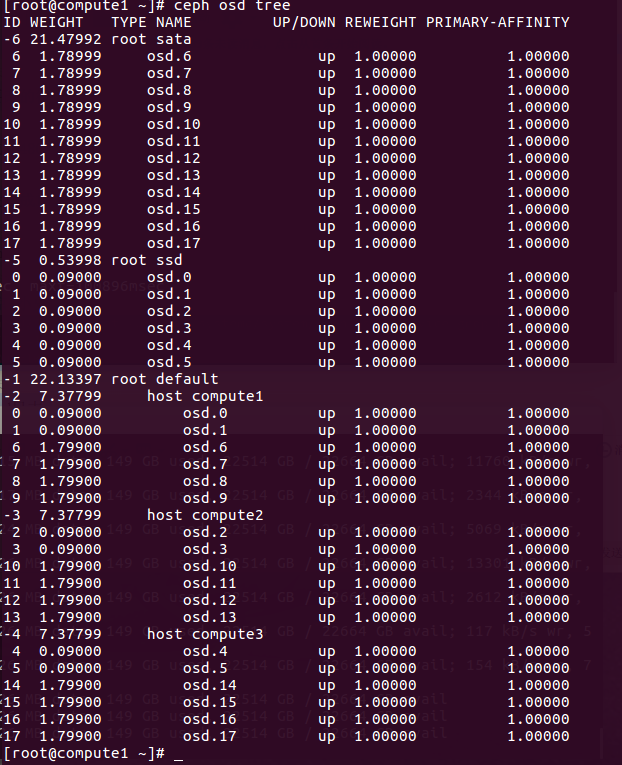
naming =version 2 bsize=4096 ascii-ci=0 ftype=1

log =internal log bsize=4096 blocks=12800, version=2

= sectsz=512 sunit=8 blks, lazy-count=1

realtime =none extsz=4096 blocks=0, rtextents=0

[root@compute1 ~]#



# 文件系统：

Ceph 的 OSD 依赖于底层文件系统的稳定性和性能。

当前，我们推荐部署生产系统时使用 xfs 文件系统；推荐用 btrfs 做测试、开发和其他不太要紧的部署。我们相信，长远来看 btrfs 适合 Ceph 的功能需求和发展方向，但是 xfs 和 ext4 能提供当前部署所必需的稳定性。 btrfs 开发在迅速推进，所以它的用户应该有能力经常更新到最新内核发布，而且能跟踪严重缺陷的修正进度。

OSD 守护进程有赖于底层文件系统的扩展属性（ XATTR ）存储各种内部对象状态和元数据。底层文件系统必须能为 XATTR 提供足够容量， btrfs 没有限制随文件的 xattr 元数据总量； xfs 的限制相对大（ 64KB ），多数部署都不会有瓶颈； ext4 的则太小而不可用。

使用 ext4 文件系统时，一定要把下面的配置放于 ceph.conf 配置文件的 [osd] 段下；用 btrfs 和 xfs 时可以选填。

filestore xattr use omap = true

# 部署OSD

部署拓扑图

按照要求开始部署OSD：

先查看节点上的磁盘状态，并使用**fdisk -l** 查看哪些盘是SSD和SATA盘。红色部分用于部署OSD

[root@compute1 cluster]# ceph-deploy disk list compute1

[ceph\_deploy.conf][DEBUG ] found configuration file at: /root/.cephdeploy.conf

[ceph\_deploy.cli][INFO ] Invoked (1.5.37): /usr/bin/ceph-deploy disk list compute1

[ceph\_deploy.cli][INFO ] ceph-deploy options:

[ceph\_deploy.cli][INFO ] username : None

[ceph\_deploy.cli][INFO ] verbose : False

[ceph\_deploy.cli][INFO ] overwrite\_conf : False

[ceph\_deploy.cli][INFO ] subcommand : list

[ceph\_deploy.cli][INFO ] quiet : False

[ceph\_deploy.cli][INFO ] cd\_conf : <ceph\_deploy.conf.cephdeploy.Conf instance at 0x1e0abd8>

[ceph\_deploy.cli][INFO ] cluster : ceph

[ceph\_deploy.cli][INFO ] func : <function disk at 0x1e00c08>

[ceph\_deploy.cli][INFO ] ceph\_conf : None

[ceph\_deploy.cli][INFO ] default\_release : False

[ceph\_deploy.cli][INFO ] disk : [('compute1', None, None)]

[compute1][DEBUG ] connected to host: compute1

[compute1][DEBUG ] detect platform information from remote host

[compute1][DEBUG ] detect machine type

[compute1][DEBUG ] find the location of an executable

[ceph\_deploy.osd][INFO ] Distro info: CentOS Linux 7.3.1611 Core

[ceph\_deploy.osd][DEBUG ] Listing disks on compute1...

[compute1][DEBUG ] find the location of an executable

[compute1][INFO ] Running command: /usr/sbin/ceph-disk list

[compute1][DEBUG ] /dev/dm-0 swap, swap

[compute1][DEBUG ] /dev/dm-1 other, xfs, mounted on /

[compute1][DEBUG ] /dev/dm-2 other, xfs, mounted on /home

[compute1][DEBUG ] /dev/sda other, unknown

[compute1][DEBUG ] /dev/sdb other, unknown

[compute1][DEBUG ] /dev/sdc other, unknown

[compute1][DEBUG ] /dev/sdd other, unknown

[compute1][DEBUG ] /dev/sde other, unknown

[compute1][DEBUG ] /dev/sdf other, unknown

[compute1][DEBUG ] /dev/sdg :

[compute1][DEBUG ] /dev/sdg2 other, LVM2\_member

[compute1][DEBUG ] /dev/sdg1 other, xfs, mounted on /boot

[compute1][DEBUG ] /dev/sr0 other, unknown

[root@compute1 cluster]#

确定哪个是SSD后，开始部署OSD

[root@compute1 cluster]# ceph-deploy osd prepare

compute1:/dev/sdf compute2:/dev/sdf compute3:/dev/sdf

[root@compute1 cluster]# ceph-deploy disk list compute1

[ceph\_deploy.conf][DEBUG ] found configuration file at: /root/.cephdeploy.conf

[ceph\_deploy.cli][INFO ] Invoked (1.5.37): /usr/bin/ceph-deploy disk list compute1

[ceph\_deploy.cli][INFO ] ceph-deploy options:

[ceph\_deploy.cli][INFO ] username : None

[ceph\_deploy.cli][INFO ] verbose : False

[ceph\_deploy.cli][INFO ] overwrite\_conf : False

[ceph\_deploy.cli][INFO ] subcommand : list

[ceph\_deploy.cli][INFO ] quiet : False

[ceph\_deploy.cli][INFO ] cd\_conf : <ceph\_deploy.conf.cephdeploy.Conf instance at 0x26febd8>

[ceph\_deploy.cli][INFO ] cluster : ceph

[ceph\_deploy.cli][INFO ] func : <function disk at 0x26f4c08>

[ceph\_deploy.cli][INFO ] ceph\_conf : None

[ceph\_deploy.cli][INFO ] default\_release : False

[ceph\_deploy.cli][INFO ] disk : [('compute1', None, None)]

[compute1][DEBUG ] connected to host: compute1

[compute1][DEBUG ] detect platform information from remote host

[compute1][DEBUG ] detect machine type

[compute1][DEBUG ] find the location of an executable

[ceph\_deploy.osd][INFO ] Distro info: CentOS Linux 7.3.1611 Core

[ceph\_deploy.osd][DEBUG ] Listing disks on compute1...

[compute1][DEBUG ] find the location of an executable

[compute1][INFO ] Running command: /usr/sbin/ceph-disk list

[compute1][DEBUG ] /dev/dm-0 swap, swap

[compute1][DEBUG ] /dev/dm-1 other, xfs, mounted on /

[compute1][DEBUG ] /dev/dm-2 other, xfs, mounted on /home

[compute1][DEBUG ] /dev/sda other, unknown

[compute1][DEBUG ] /dev/sdb other, unknown

[compute1][DEBUG ] /dev/sdc other, unknown

[compute1][DEBUG ] /dev/sdd other, unknown

[compute1][DEBUG ] /dev/sde other, unknown

[compute1][DEBUG ] /dev/sdf :

[compute1][DEBUG ] /dev/sdf2 ceph journal, for /dev/sdf1 # 日志盘

[compute1][DEBUG ] /dev/sdf1 ceph data, active, cluster ceph, osd.0, journal /dev/sdf2 # 数据盘

[compute1][DEBUG ] /dev/sdg :

[compute1][DEBUG ] /dev/sdg2 other, LVM2\_member

[compute1][DEBUG ] /dev/sdg1 other, xfs, mounted on /boot

[compute1][DEBUG ] /dev/sr0 other, unknown

[root@compute1 cluster]# ceph-deploy osd activate # 激活数据盘

compute1:/dev/sdf1 compute2:/dev/sdf1 compute3:/dev/sdf1

[root@compute1 cluster]#

激活后，查看ceph集群状态：

[root@compute1 cluster]# ceph osd tree

ID WEIGHT TYPE NAME UP/DOWN REWEIGHT PRIMARY-AFFINITY

-1 0.31256 root default

-2 0.10419 host compute1

0 0.10419 osd.0 up 1.00000 1.00000

-3 0.10419 host compute2

1 0.10419 osd.1 up 1.00000 1.00000

-4 0.10419 host compute3

2 0.10419 osd.2 up 1.00000 1.00000

[root@compute1 cluster]# ceph status

cluster 1dff8233-2e51-43a1-8577-9b6c8af94f4a

health HEALTH\_OK

monmap e3: 3 mons at

{compute1=10.0.1.31:6789/0,compute2=10.0.1.32:6789/0,compute3=10.0.1.33:6789/0}

election epoch 14, quorum 0,1,2 compute1,compute2,compute3

osdmap e15: 3 osds: 3 up, 3 in

flags sortbitwise,require\_jewel\_osds

pgmap v29: 64 pgs, 1 pools, 0 bytes data, 0 objects

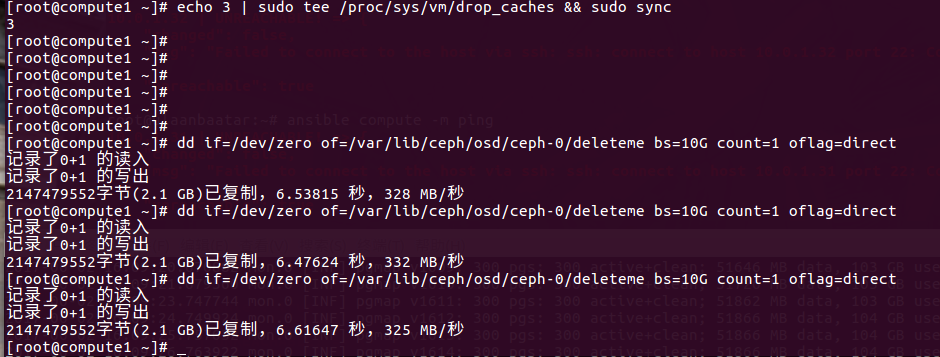
100 MB used, 320 GB / 320 GB avail

64 active+clean

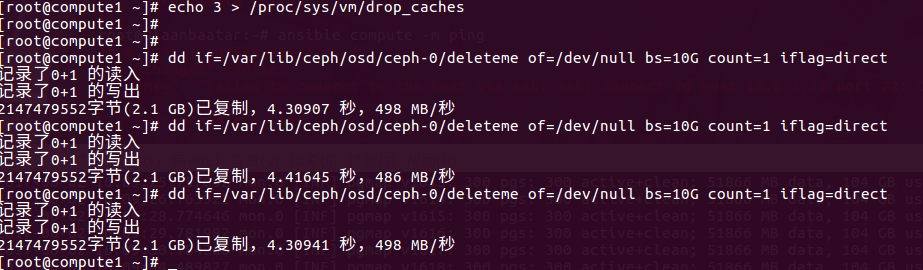
[root@compute1 cluster]#

# Ceph集群测试

写：



读



# Ceph cache-tier 操作

## 创建cache-tier

## 删除cache-tier

REMOVING A WRITEBACK CACHE

删除cache-mode为writeback 模式的：

[root@compute1 ~]# ceph osd tier cache-mode ssd-cache-pool forward --yes-i-really-mean-it

set cache-mode for pool 'ssd-cache-pool' to forward

[root@compute1 ~]#

# Ceph 集群优化之 SSD

## Linux OS 优化

Kernel pid max

[root@compute1 ~]# cat /proc/sys/kernel/pid\_max

32768

[root@compute1 ~]# echo 4194303 > /proc/sys/kernel/pid\_max

[root@compute1 ~]# cat /proc/sys/kernel/pid\_max

4194303

[root@compute1 ~]#

关口的MTU设置为9000

[root@compute1 ~]# ifconfig p6p2

p6p2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> **mtu 1500**

inet 10.0.16.31 netmask 255.255.255.0 broadcast 10.0.16.255

inet6 fe80::a236:9fff:fe27:f122 prefixlen 64 scopeid 0x20<link>

ether a0:36:9f:27:f1:22 txqueuelen 1000 (Ethernet)

RX packets 6043 bytes 1024056 (1000.0 KiB)

RX errors 0 dropped 0 overruns 0 frame 0

TX packets 6054 bytes 1010268 (986.5 KiB)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

[root@compute1 ~]# **ifconfig p6p2 mtu 9000**

[root@compute1 ~]# ifconfig p6p2

p6p2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> **mtu 9000**

inet 10.0.16.31 netmask 255.255.255.0 broadcast 10.0.16.255

inet6 fe80::a236:9fff:fe27:f122 prefixlen 64 scopeid 0x20<link>

ether a0:36:9f:27:f1:22 txqueuelen 1000 (Ethernet)

RX packets 7184 bytes 1147944 (1.0 MiB)

RX errors 0 dropped 0 overruns 0 frame 0

TX packets 7174 bytes 1133408 (1.0 MiB)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

# 永久设置

[root@compute1 ~]# echo "MTU=9000" | tee -a /etc/sysconfig/network-scripts/ifcfg-p6p2

## PG Number

查看某个pool的pg和pgp的个数：

[root@compute1 cluster]# ceph osd pool get rbd pg\_num

pg\_num: 300

[root@compute1 cluster]# ceph osd pool get rbd pgp\_num

pgp\_num: 300

[root@compute1 cluster]#

PG和PGP数量一定要根据OSD的数量进行调整，计算公式如下，但是最后算出的结果一定要接近或者等于一个2的指数。

Total PGs = (Total\_number\_of\_OSD \* 100) / max\_replication\_count

例如15个OSD，副本数为3的情况下，根据公式计算的结果应该为500，最接近512，所以需要设定该pool(volumes)的pg\_num和pgp\_num都为512.

ceph osd pool set volumes pg\_num 512

ceph osd pool set volumes pgp\_num 512

## 副本数量

获取对象的副本数量：

[root@compute1 cluster]# ceph osd dump | grep 'replicated size'

pool 0 'rbd' **replicated size 2** min\_size 1 crush\_ruleset 0 object\_hash rjenkins pg\_num 200 pgp\_num 200 last\_change 79 flags hashpspool stripe\_width 0

[root@compute1 cluster]#

[root@compute1 cluster]# ceph osd pool get rbd size

size: 2

[root@compute1 cluster]# ceph osd pool set rbd size 3 # 修改副本书为3

# Ceph操作：

## Pool操作

格式为： ceph osd pool create {pool-name} {pg-num} [{pgp-num}]

[root@compute1 cluster]# ceph osd pool create baater 300 300

[root@compute1 cluster]# ceph osd lspools # 查看已创建的pool

0 rbd,2 baater,

[root@compute1 cluster]# ceph osd pool get baater pg\_num

pg\_num: 300

[root@compute1 cluster]# ceph osd pool get baater pgp\_num

pgp\_num: 300

## 编辑Crushmap

**Get a CRUSH Map：**

[root@compute1 cluster]# **ceph osd getcrushmap -o crushmap-compiled**

got crush map from osdmap epoch 2108

[root@compute1 cluster]# **crushtool -d crushmap-compiled -o crushmap-decompiled**

[root@compute1 cluster]# ll

-rw-r--r-- 1 root root 736 2月 12 16:11 crushmap-compiled

-rw-r--r-- 1 root root 1508 2月 12 16:12 crushmap-decompiled

[root@compute1 cluster]#

**编辑新的crushmap布局**

[root@compute1 cluster]# **crushtool -c crushmap-decompiled -o crushmap-new**

[root@compute1 cluster]# ll

-rw-r--r-- 1 root root 736 2月 12 16:11 crushmap-compiled

-rw-r--r-- 1 root root 1736 2月 12 16:16 crushmap-decompiled

-rw-r--r-- 1 root root 883 2月 12 16:17 crushmap-new

[root@compute1 cluster]# **ceph osd setcrushmap -i crushmap-new**

set crush map

[root@compute1 cluster]# ceph osd tree

ID WEIGHT TYPE NAME UP/DOWN REWEIGHT PRIMARY-AFFINITY

-1 4.00000 root default

-2 1.00000 host compute1

0 1.00000 osd.0 up 1.00000 1.00000

1 1.00000 osd.1 up 1.00000 1.00000

2 1.00000 osd.2 up 1.00000 1.00000

3 1.00000 osd.3 up 1.00000 1.00000

-3 1.00000 host compute2

4 1.00000 osd.4 up 1.00000 1.00000

5 1.00000 osd.5 up 1.00000 1.00000

6 1.00000 osd.6 up 1.00000 1.00000

7 1.00000 osd.7 up 1.00000 1.00000

-4 1.00000 host compute3

8 1.00000 osd.8 up 1.00000 1.00000

10 1.00000 osd.10 up 1.00000 1.00000

11 1.00000 osd.11 up 1.00000 1.00000

**-5 1.00000 host compute5**

**9 2.00000 osd.9 up 1.00000 1.00000**

**12 2.00000 osd.12 up 1.00000 1.00000**

**13 2.00000 osd.13 up 1.00000 1.00000**

**14 2.00000 osd.14 up 1.00000 1.00000**

[root@compute1 cluster]#

## 指定的OSD上创建pool

先获取ceph集群的CRUSHmap：

[root@compute1 cluster]# ceph osd getcrushmap -o crushmapdump

got crush map from osdmap epoch 1594

[root@compute1 cluster]# crushtool -d crushmapdump -o crushmapdump-decompiled

[root@compute1 cluster]# ls -lh crushmapdump-decompiled

-rw-r--r-- 1 root root 1.8K 6月 5 12:00 crushmapdump-decompiled

[root@compute1 cluster]#

存放一下对象，并查看保存对象到指定的osd上：

[root@compute1 ~]# rados -p ssd-pool put dummpy\_obj1 /etc/hosts

[root@compute1 ~]# rados -p sata-pool put dummpy\_obj1 /etc/hosts

[root@compute1 ~]# rados -p ssd-pool ls

dummpy\_obj1

[root@compute1 ~]# rados -p ssd-pool rm dummpy\_obj1 # 删除对象

[root@compute1 ~]# rados -p ssd-pool ls

[root@compute1 ~]#

[root@compute1 ~]# rados -p sata-pool ls

dummpy\_obj1

[root@compute1 ~]# ceph osd map ssd-pool dummpy\_obj1

osdmap e1604 pool 'ssd-pool' (3) object 'dummpy\_obj1' -> pg 3.565b968a (3.8a) -> up ([1,5], p1) acting ([1,5], p1)

[root@compute1 ~]# ceph osd map sata-pool dummpy\_obj1

osdmap e1604 pool 'sata-pool' (4) object 'dummpy\_obj1' -> pg 4.565b968a (4.8a) -> up ([14,15], p14) acting ([14,15], p14)

## PG操作

### 查看pg组的映射信息

[root@compute1 cluster]# **ceph pg dump**

dumped all in format plain

version 795

stamp 2017-05-31 11:49:07.837122

last\_osdmap\_epoch 93

last\_pg\_scan 93

full\_ratio 0.95

nearfull\_ratio 0.85

pg\_stat objects mip degr misp unf bytes log disklog state state\_stamp v reported up up\_primary acting acting\_primary last\_scrub scrub\_stamp last\_deep\_scrub deep\_scrub\_stamp

0.c7 13 0 0 0 0 54525952 3048 3048 active+clean 2017-05-31 09:33:49.952490 93'8049 93:4999 [5,2] 5 [5,2] 5 0'0 2017-05-31 08:42:35.390042 0'0 2017-05-27 17:35:00.949804

0.c6 6 0 0 0 0 25165824 3059 3059 active+clean 2017-05-31 09:32:45.912542 93'4059 93:3171 [1,5] 1 [1,5] 1 0'0 2017-05-31 08:41:27.187428 0'0 2017-05-31 08:41:27.187428

查看保存的对象的对应的pg名字

[root@compute1 ceph-disk0]# ceph osd map rbd test

osdmap e526 pool 'rbd' (0) object 'test' -> pg 0.40e8aab5 (0.b5) -> up ([2,0], p2) acting ([2,0], p2)

## OSD操作

### 启动osd进程

要想在节点上启动一种服务时，在该节点上运行一下命令：

sudo systemctl start ceph-osd.target

sudo systemctl start ceph-mon.target

### 查看集群中每个osd的使用信息

[root@compute1 ~]# ceph osd df

ID WEIGHT REWEIGHT SIZE USE AVAIL %USE VAR PGS

0 0.10419 1.00000 106G 4483M 102G 4.10 1.22 62

1 0.10419 1.00000 106G 2981M 103G 2.73 0.81 62

2 0.10419 1.00000 106G 3690M 103G 3.38 1.01 72

3 0.10419 1.00000 106G 3318M 103G 3.04 0.90 62

4 0.10419 1.00000 106G 3223M 103G 2.95 0.88 64

5 0.10419 1.00000 106G 4330M 102G 3.96 1.18 78

TOTAL 640G 22027M 618G 3.36

MIN/MAX VAR: 0.81/1.22 STDDEV: 0.51

[root@compute1 ~]#

### 删除osd：

~~Sometimes removing OSD, if not done properly can result in double rebalancing. The best practice to remove an OSD involves changing the crush weight to 0.0 as first step.~~

~~So in the end, this will give you:~~

~~$ ceph osd crush reweight osd.<ID> 0.0~~

~~Then you wait for rebalance to be completed. Eventually completely remove the OSD:~~

~~$ ceph osd out <ID>~~

~~$ service ceph stop osd.<ID>~~

~~$ ceph osd crush remove osd.<ID>~~

~~$ ceph auth del osd.<ID>~~

~~$ ceph osd rm <ID>~~

首先set out osd，并ceph集群开始数据回复

首先停止制定的osd服务进程

# systemctl stop ceph-osd@9

$ ceph osd crush reweight osd.<ID> 0.0

# ceph osd out osd.9

同时观察ceph 集群状态

# ceph -w

**Ceph 集群状状态为ok时，**标记为out的osd还在运行中所以stop

# ceph osd down id

Osd的状态为down，并查看结果

目前，osd不再集群中，所以从CRUSH Map中remove

# ceph osd crush remove osd.9

Remove osd authentication keys

# ceph auth del osd.9

最后，删除osd，并官擦ceph集群状态

# ceph osd rm osd.9

假如osd.9是在一个节点上，的那么ceph 的集群状态微健康时，可以删除掉这个节点。

# ceph osd crush remove *ceph-node4*

### 查看某个osd的配置信息：

ceph daemon osd.0 config show | less

### 在指定的OSD上创建ceph pool

**Ceph cookbook 第196页**

### 查看OSD映射信息

[root@compute1 cluster]# **ceph osd dump**

epoch 93

fsid 59624c1d-304a-4a3c-84c7-aeab454e17e2

created 2017-05-27 17:35:00.941747

modified 2017-05-31 10:16:52.710723

flags sortbitwise,require\_jewel\_osds

pool 0 'rbd' replicated size 2 min\_size 1 crush\_ruleset 0 object\_hash rjenkins pg\_num 200 pgp\_num 200 last\_change 93 flags hashpspool stripe\_width 0

removed\_snaps [1~3]

max\_osd 6

osd.0 up in weight 1 up\_from 52 up\_thru 90 down\_at 50 last\_clean\_interval [35,51) 10.0.1.31:6800/3623 10.0.16.31:6804/1003623 10.0.16.31:6805/1003623 10.0.1.31:6804/1003623 exists,up 433c82a1-cf4f-487a-ae09-33668d98c202

osd.1 up in weight 1 up\_from 54 up\_thru 90 down\_at 50 last\_clean\_interval [35,51) 10.0.1.31:6802/3804 10.0.16.31:6806/1003804 10.0.16.31:6807/1003804 10.0.1.31:6805/1003804 exists,up 3e26082c-9199-48ca-9a69-c67940d9601e

osd.2 up in weight 1 up\_from 89 up\_thru 89 down\_at 87 last\_clean\_interval [42,87) 10.0.1.32:6802/4047 10.0.16.32:6804/1004047 10.0.16.32:6805/1004047 10.0.1.32:6804/1004047 exists,up 73190e4a-79c8-460c-a236-ab1283c1fe97

osd.3 up in weight 1 up\_from 90 up\_thru 90 down\_at 87 last\_clean\_interval [41,88) 10.0.1.32:6800/3867 10.0.16.32:6802/1003867 10.0.16.32:6806/1003867 10.0.1.32:6803/1003867 exists,up 44963706-e26a-4959-9120-2f6c9a0adac1

osd.4 up in weight 1 up\_from 84 up\_thru 90 down\_at 81 last\_clean\_interval [74,83) 10.0.1.33:6802/8059 10.0.16.33:6800/1008059 10.0.16.33:6806/1008059 10.0.1.33:6801/1008059 exists,up 36987fee-8f52-4df5-98a4-d9867f5db8c9

osd.5 up in weight 1 up\_from 83 up\_thru 90 down\_at 81 last\_clean\_interval [74,82) 10.0.1.33:6800/8063 10.0.16.33:6804/1008063 10.0.16.33:6805/1008063 10.0.1.33:6804/1008063 exists,up b27e00cc-6940-4783-8a15-ac1ef0e9626c

[root@compute1 cluster]#

## MON操作

### 添加新的monitor

**注意：关掉防火墙！！！**

[root@compute1 cluster]# ceph-deploy mon add compute3

[root@compute1 cluster]# cat ceph.conf #手动添加第三个mon节点

[global]

fsid = 1dff8233-2e51-43a1-8577-9b6c8af94f4a

mon\_initial\_members = compute1,compute2,compute3

mon\_host = 10.0.1.31,10.0.1.32,10.0.1.33

auth\_cluster\_required = cephx

auth\_service\_required = cephx

auth\_client\_required = cephx

public\_network = 10.0.1.0/24

osd pool default size = 2

mon-clock-drift-allowed = 2

[root@compute1 cluster]#

[root@compute1 cluster]# ceph quorum\_status --format json-pretty # 查看仲裁状态

{

"election\_epoch": 6,

"quorum": [

0,

1,

2

],

"quorum\_names": [

"compute1",

"compute2",

"compute3"

],

"quorum\_leader\_name": "compute1",

"monmap": {

"epoch": 3,

"fsid": "1dff8233-2e51-43a1-8577-9b6c8af94f4a",

"modified": "2017-05-26 09:33:30.138366",

"created": "2017-05-26 09:10:38.567603",

"mons": [

{

"rank": 0,

"name": "compute1",

"addr": "10.0.1.31:6789\/0"

},

{

"rank": 1,

"name": "compute2",

"addr": "10.0.1.32:6789\/0"

},

{

"rank": 2,

"name": "compute3",

"addr": "10.0.1.33:6789\/0"

}

]

}

}

[root@compute1 cluster]#

### 删除monitor

首先查看monitor状态

# ceph mon stat

### 重启mon和osd

# systemctl restart ceph-mon@compute1

# systemctl restart

# 三副本读操作流程

本文详细描述Jewel版本的Ceph集群在三副本的情况下如何处理客户端读请求的。

一、主OSD读处理流程

OSD::ms\_fast\_dispatch()

|\_\_OSD::dispatch\_session\_waiting()

|\_\_OSD::dispatch\_op\_fast()

|\_\_OSD::handle\_op()

|\_\_OSD::get\_pg\_or\_queue\_for\_pg() 找到OpRequest中对应的PG和Pool信息

|\_\_OSD::enqueue\_op()

|\_\_PG::queue\_op()

|\_\_OSD::ShardedThreadPool::ShardedWQ::queue() 将PG和Op一起放入队列中

OSD::ShardedOpWQ::\_process() 负责处理OSD::ShardedThreadPool::ShardedWQ队列中的Op

|\_\_PGQueueable::RunVis::operator()(const OpRequestRef &op)

|\_\_OSD::dequeue\_op()

|\_\_ReplicatedPG::do\_request()

|\_\_检查当前PG是否处于flush或peering状态，若是则将op放入waiting\_for\_peered队列中等待PG变成可用状态

|\_\_检查当前PG是否处于Active状态，若不是则将op放入waiting\_for\_active队列中

|\_\_检查当前PG是否处于REPLAY状态，若是则将op放入waiting\_for\_active队列中

|\_\_ReplicatedPG::do\_op()

|\_\_RepliatedPG::do\_pg\_op() 对于请求中包含对PG的操作CEPH\_OSD\_RMW\_FLAG\_PGOP

|\_\_根据op请求创建hobject\_t类对象(head)

|\_\_检查对象名字长度/对象locator key长度/对象locator名称空间长度是否大于osd\_max\_object\_name\_len

|\_\_通过FileStore检查object的head是否有效

|\_\_检查op请求地址是否在OSDMap的blacklist中

|\_\_对于写请求，检查写请求的数据大小是否大于osd\_max\_write\_size值

|\_\_对于op请求的head目前不可读，则将op放入waiting\_for\_unreadable\_object队列中且调用maybe\_kick\_recovery()函数尝试启动recovery

|\_\_ReplicatedPG::is\_degraded\_or\_backfilling\_object() 检查当前op请求的head是否处于recovery或backfill状态

|\_\_ReplicatedPG::wait\_for\_degraded\_object() 将当前op请求的head放入waiting\_for\_degraded\_object队列中

|\_\_检查head是否在objects\_blocked\_on\_degraded\_snap队列中，若是则将当前op请求的head放入waiting\_for\_degraded\_object队列中

|\_\_检查head是否在objects\_blocked\_on\_snap\_promotion队列中，若是则将当前op请求的head放入waiting\_for\_blocked\_object队列中

|\_\_检查head是否在objects\_blocked\_on\_cache\_full队列中，若是则将当前op请求的head放入waiting\_for\_cache\_not\_full队列中

|\_\_检查head的snapdir是否不可读，若是则将head的snapdir放入waiting\_for\_unreadable\_object队列中且调用maybe\_kick\_recovery()函数尝试启动recovery

|\_\_检查head的snapdir是否处于recovery或backfill状态，若是则将head的snapdir放入waiting\_for\_degraded\_object队列中

|\_\_对于op写请求已经在PGLog中，则若已经完成了写操作则直接给客户端返回MOSDOpReply消息且设置CEPH\_OSD\_FLAG\_ACK，否则将op放入到waiting\_for\_ack或waiting\_for\_ondisk队列中

|\_\_ReplicatedPG::find\_object\_context() 得到object context信息

|\_\_检查object context是否处于io blocked状态，若是则将op请求放入到waiting\_for\_blocked\_object或waiting\_for\_degraded\_object队列中

|\_\_ReplicatedPG::execute\_ctx()

|\_\_ReplicatedPG::prepare\_transaction()

|\_\_创建MOSDOpReply消息实例

|\_\_调用ReplicatePG::complete\_read\_ctx() 将读操作的结果返回给client端

# Ceph-lazy 工具

**HOW TO INSTALL**

Usage is pretty straightforward just ensure you have all dependencies installed, clone the git repo

git clone https://github.com/gcharot/ceph-lazy.git

Copy the ceph-lazy to a PATH directory and ensure the proper permissions are set.

cp ceph-lazy/ceph-lazy /usr/local/sbin/

chown root:root /usr/local/sbin/ceph-lazy

chmod u+x /usr/local/sbin/ceph-lazy

If you want to enable bash completion you can copy the completion config file.

cp ceph-lazy/bash\_completion.d/ceph-lazy /etc/bash\_completion.d/

**HOW TO USE**

Simply execute ceph-lazy with no parameter or use the -h option to get the list of options and commands. Using -d as first parameter enable verbose mode (printed on stderr). General syntax usage is :

Usage : ceph-lazy [-d | -h] [command] [parameters]

报错解决方法：

[root@compute1 ~]# ceph-lazy osd-most-used

ERROR: jq cannot be found... Aborting.

[root@compute1 ~]#

[root@compute1 ~]#

[root@compute1 ~]# yum install jq -y

# 网卡流量监控

监控linux服务器的每个网卡的network

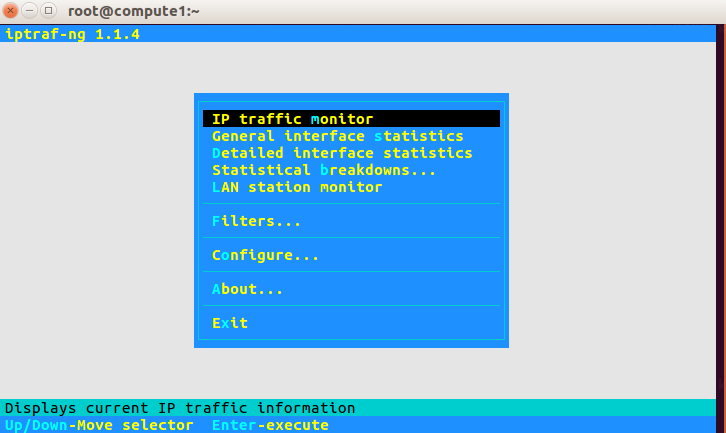
安装：

# yum install iptraf -y

运行：

# iptraf-ng

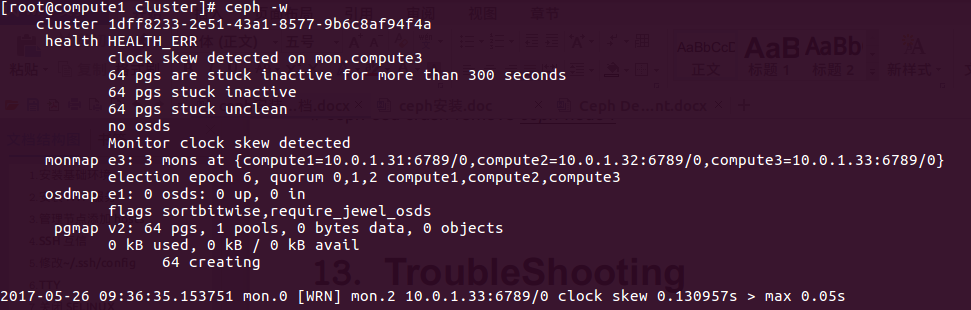
进去后看见管理界面，并选择要查看的网卡



# TroubleShooting

## Monitor之间有时间差

Monitor clock skew detected



添加完以后，分发到每个节点，然后在管理节点重启monitor

[root@compute1 cluster]# cat ceph.conf

[global]

fsid = 1dff8233-2e51-43a1-8577-9b6c8af94f4a

mon\_initial\_members = compute1,compute2,compute3

mon\_host = 10.0.1.31,10.0.1.32,10.0.1.33

auth\_cluster\_required = cephx

auth\_service\_required = cephx

auth\_client\_required = cephx

public\_network = 10.0.1.0/24

osd pool default size = 2

mon clock drift allowed = 2

mon clock drift warn backoff = 30

[root@compute1 cluster]# ceph-deploy --overwrite-conf admin compute1 compute2 compute3

[root@compute1 cluster]# systemctl restart ceph-mon@compute1

## 调整pg数量

查看集群，只有一个默认的pool

# sudo ceph osd lspools

0 rbd,

查看rbd pool 的 pgs

# sudo ceph osd pool get rbd pg\_num

pg\_num: 64 #默认为64

pgs为64，因为是2副本的配置，所以当有8个osd的时候，每个osd上均分了64/8 \*2=16个pgs,也就是出现了如上的错误 小于最小配置30个

解决办法：修改默认pool rbd的pgs

$ sudo **ceph osd pool set rbd pg\_num 128**

set pool 0 pg\_num to 128

$ sudo ceph -s

cluster 257faba1-f259-4164-a0f9-1726bd70b05a

health HEALTH\_WARN

64 pgs stuck inactive

64 pgs stuck unclean

pool rbd pg\_num 128 > pgp\_num 64

monmap e1: 1 mons at {bdc217=192.168.13.217:6789/0}

election epoch 2, quorum 0 bdc217

osdmap e52: 8 osds: 8 up, 8 in

flags sortbitwise

pgmap v121: 128 pgs, 1 pools, 0 bytes data, 0 objects

715 MB used, 27550 GB / 29025 GB avail

64 active+clean

64 creating

发现需要把pgp\_num也一并修改，默认两个pg\_num和pgp\_num一样大小均为64，此处也将两个的值都设为128

$ sudo ceph osd pool set rbd pgp\_num 128

set pool 0 pgp\_num to 128

最后查看集群状态，显示为OK，错误解决：

## 服务器重启后，OSD为down

首先查看所有的服务：

[root@compute3 ~]# systemctl status ceph\\*.service ceph\\*.target

● ceph-osd@5.service - Ceph object storage daemon

Loaded: loaded (/usr/lib/systemd/system/ceph-osd@.service; disabled; vendor preset: disabled)

Active: failed (Result: start-limit) since 四 2017-06-01 09:29:54 CST; 49min ago

Process: 11377 ExecStart=/usr/bin/ceph-osd -f --cluster ${CLUSTER} --id %i --setuser ceph --setgroup ceph (code=exited, status=1/FAILURE)

Process: 11335 ExecStartPre=/usr/lib/ceph/ceph-osd-prestart.sh --cluster ${CLUSTER} --id %i (code=exited, status=0/SUCCESS)

Main PID: 11377 (code=exited, status=1/FAILURE)

6月 01 09:29:34 compute3 systemd[1]: Unit ceph-osd@5.service entered failed state.

6月 01 09:29:34 compute3 systemd[1]: ceph-osd@5.service failed.

6月 01 09:29:54 compute3 systemd[1]: ceph-osd@5.service holdoff time over, scheduling restart.

6月 01 09:29:54 compute3 systemd[1]: start request repeated too quickly for ceph-osd@5.service

6月 01 09:29:54 compute3 systemd[1]: Failed to start Ceph object storage daemon.

6月 01 09:29:54 compute3 systemd[1]: Unit ceph-osd@5.service entered failed state.

6月 01 09:29:54 compute3 systemd[1]: ceph-osd@5.service failed.

● ceph-osd.target - ceph target allowing to start/stop all ceph-osd@.service instances at once

Loaded: loaded (/usr/lib/systemd/system/ceph-osd.target; enabled; vendor preset: enabled)

Active: active since 三 2017-05-31 18:11:48 CST; 16h ago

● ceph-disk@dev-sdf2.service - Ceph disk activation: /dev/sdf2

Loaded: loaded (/usr/lib/systemd/system/ceph-disk@.service; static; vendor preset: disabled)

Active: failed (Result: exit-code) since 三 2017-05-31 18:11:51 CST; 16h ago

Process: 1157 ExecStart=/bin/sh -c timeout 120 flock /var/lock/ceph-disk-$(basename %f) /usr/sbin/ceph-disk --verbose --log-stdout trigger --sync %f (code=exited, status=1/FAILURE)

Main PID: 1157 (code=exited, status=1/FAILURE)

5月 31 18:11:51 compute3 sh[1157]: main(sys.argv[1:])

5月 31 18:11:51 compute3 sh[1157]: File "/usr/lib/python2.7/site-packages/ceph\_disk/main.py", line 4998, in main

5月 31 18:11:51 compute3 sh[1157]: args.func(args)

5月 31 18:11:51 compute3 sh[1157]: File "/usr/lib/python2.7/site-packages/ceph\_disk/main.py", line 4435, in main\_trigger

5月 31 18:11:51 compute3 sh[1157]: raise Error('return code ' + str(ret))

5月 31 18:11:51 compute3 sh[1157]: ceph\_disk.main.Error: Error: return code 1

5月 31 18:11:51 compute3 systemd[1]: ceph-disk@dev-sdf2.service: main process exited, code=exited, status=1/FAILURE

5月 31 18:11:51 compute3 systemd[1]: Failed to start Ceph disk activation: /dev/sdf2.

5月 31 18:11:51 compute3 systemd[1]: Unit ceph-disk@dev-sdf2.service entered failed state.

5月 31 18:11:51 compute3 systemd[1]: ceph-disk@dev-sdf2.service failed.

● ceph-mon@compute3.service - Ceph cluster monitor daemon

Loaded: loaded (/usr/lib/systemd/system/ceph-mon@.service; enabled; vendor preset: disabled)

Active: active (running) since 四 2017-06-01 10:04:48 CST; 14min ago

Main PID: 12772 (ceph-mon)

CGroup: /system.slice/system-ceph\x2dmon.slice/ceph-mon@compute3.service

└─12772 /usr/bin/ceph-mon -f --cluster ceph --id compute3 --setuser ceph --setgroup ceph

6月 01 10:04:48 compute3 systemd[1]: Started Ceph cluster monitor daemon.

6月 01 10:04:48 compute3 systemd[1]: Starting Ceph cluster monitor daemon...

6月 01 10:04:48 compute3 ceph-mon[12772]: starting mon.compute3 rank 2 at 10.0.1.33:6789/0 mon\_data /var/lib/ceph/mon/ceph-compute3 fs...454e17e2

● ceph-mds.target - ceph target allowing to start/stop all ceph-mds@.service instances at once

Loaded: loaded (/usr/lib/systemd/system/ceph-mds.target; enabled; vendor preset: enabled)

Active: active since 三 2017-05-31 18:11:48 CST; 16h ago

● ceph-disk@dev-sde1.service - Ceph disk activation: /dev/sde1

Loaded: loaded (/usr/lib/systemd/system/ceph-disk@.service; static; vendor preset: disabled)

Active: failed (Result: exit-code) since 三 2017-05-31 18:11:51 CST; 16h ago

Process: 1163 ExecStart=/bin/sh -c timeout 120 flock /var/lock/ceph-disk-$(basename %f) /usr/sbin/ceph-disk --verbose --log-stdout trigger --sync %f (code=exited, status=1/FAILURE)

Main PID: 1163 (code=exited, status=1/FAILURE)

5月 31 18:11:51 compute3 sh[1163]: main(sys.argv[1:])

5月 31 18:11:51 compute3 sh[1163]: File "/usr/lib/python2.7/site-packages/ceph\_disk/main.py", line 4998, in main

5月 31 18:11:51 compute3 sh[1163]: args.func(args)

5月 31 18:11:51 compute3 sh[1163]: File "/usr/lib/python2.7/site-packages/ceph\_disk/main.py", line 4435, in main\_trigger

5月 31 18:11:51 compute3 sh[1163]: raise Error('return code ' + str(ret))

5月 31 18:11:51 compute3 sh[1163]: ceph\_disk.main.Error: Error: return code 1

5月 31 18:11:51 compute3 systemd[1]: ceph-disk@dev-sde1.service: main process exited, code=exited, status=1/FAILURE

5月 31 18:11:51 compute3 systemd[1]: Failed to start Ceph disk activation: /dev/sde1.

5月 31 18:11:51 compute3 systemd[1]: Unit ceph-disk@dev-sde1.service entered failed state.

5月 31 18:11:51 compute3 systemd[1]: ceph-disk@dev-sde1.service failed.

● ceph.target - ceph target allowing to start/stop all ceph\*@.service instances at once

Loaded: loaded (/usr/lib/systemd/system/ceph.target; enabled; vendor preset: enabled)

Active: active since 三 2017-05-31 18:12:12 CST; 16h ago

5月 31 18:12:12 compute3 systemd[1]: Reached target ceph target allowing to start/stop all ceph\*@.service instances at once.

5月 31 18:12:12 compute3 systemd[1]: Starting ceph target allowing to start/stop all ceph\*@.service instances at once.

● ceph-disk@dev-sdf1.service - Ceph disk activation: /dev/sdf1

Loaded: loaded (/usr/lib/systemd/system/ceph-disk@.service; static; vendor preset: disabled)

Active: failed (Result: exit-code) since 三 2017-05-31 18:11:51 CST; 16h ago

Process: 1189 ExecStart=/bin/sh -c timeout 120 flock /var/lock/ceph-disk-$(basename %f) /usr/sbin/ceph-disk --verbose --log-stdout trigger --sync %f (code=exited, status=1/FAILURE)

Main PID: 1189 (code=exited, status=1/FAILURE)

5月 31 18:11:51 compute3 sh[1189]: main(sys.argv[1:])

5月 31 18:11:51 compute3 sh[1189]: File "/usr/lib/python2.7/site-packages/ceph\_disk/main.py", line 4998, in main

5月 31 18:11:51 compute3 sh[1189]: args.func(args)

5月 31 18:11:51 compute3 sh[1189]: File "/usr/lib/python2.7/site-packages/ceph\_disk/main.py", line 4435, in main\_trigger

5月 31 18:11:51 compute3 sh[1189]: raise Error('return code ' + str(ret))

5月 31 18:11:51 compute3 sh[1189]: ceph\_disk.main.Error: Error: return code 1

5月 31 18:11:51 compute3 systemd[1]: ceph-disk@dev-sdf1.service: main process exited, code=exited, status=1/FAILURE

5月 31 18:11:51 compute3 systemd[1]: Failed to start Ceph disk activation: /dev/sdf1.

5月 31 18:11:51 compute3 systemd[1]: Unit ceph-disk@dev-sdf1.service entered failed state.

5月 31 18:11:51 compute3 systemd[1]: ceph-disk@dev-sdf1.service failed.

● ceph-disk@dev-sde2.service - Ceph disk activation: /dev/sde2

Loaded: loaded (/usr/lib/systemd/system/ceph-disk@.service; static; vendor preset: disabled)

Active: failed (Result: exit-code) since 三 2017-05-31 18:11:51 CST; 16h ago

Process: 1167 ExecStart=/bin/sh -c timeout 120 flock /var/lock/ceph-disk-$(basename %f) /usr/sbin/ceph-disk --verbose --log-stdout trigger --sync %f (code=exited, status=1/FAILURE)

Main PID: 1167 (code=exited, status=1/FAILURE)

5月 31 18:11:51 compute3 sh[1167]: main(sys.argv[1:])

5月 31 18:11:51 compute3 sh[1167]: File "/usr/lib/python2.7/site-packages/ceph\_disk/main.py", line 4998, in main

5月 31 18:11:51 compute3 sh[1167]: args.func(args)

5月 31 18:11:51 compute3 sh[1167]: File "/usr/lib/python2.7/site-packages/ceph\_disk/main.py", line 4435, in main\_trigger

5月 31 18:11:51 compute3 sh[1167]: raise Error('return code ' + str(ret))

5月 31 18:11:51 compute3 sh[1167]: ceph\_disk.main.Error: Error: return code 1

5月 31 18:11:51 compute3 systemd[1]: ceph-disk@dev-sde2.service: main process exited, code=exited, status=1/FAILURE

5月 31 18:11:51 compute3 systemd[1]: Failed to start Ceph disk activation: /dev/sde2.

5月 31 18:11:51 compute3 systemd[1]: Unit ceph-disk@dev-sde2.service entered failed state.

5月 31 18:11:51 compute3 systemd[1]: ceph-disk@dev-sde2.service failed.

● ceph-radosgw.target - ceph target allowing to start/stop all ceph-radosgw@.service instances at once

Loaded: loaded (/usr/lib/systemd/system/ceph-radosgw.target; enabled; vendor preset: enabled)

Active: active since 三 2017-05-31 18:11:48 CST; 16h ago

● ceph-mon.target - ceph target allowing to start/stop all ceph-mon@.service instances at once

Loaded: loaded (/usr/lib/systemd/system/ceph-mon.target; enabled; vendor preset: enabled)

Active: active since 三 2017-05-31 18:12:12 CST; 16h ago

5月 31 18:12:12 compute3 systemd[1]: Reached target ceph target allowing to start/stop all ceph-mon@.service instances at once.

5月 31 18:12:12 compute3 systemd[1]: Starting ceph target allowing to start/stop all ceph-mon@.service instances at once.

Hint: Some lines were ellipsized, use -l to show in full.

[root@compute3 ~]#

从输出内容看，是ceph 的disk服务没有启动，运行一下命令来激活disk服务

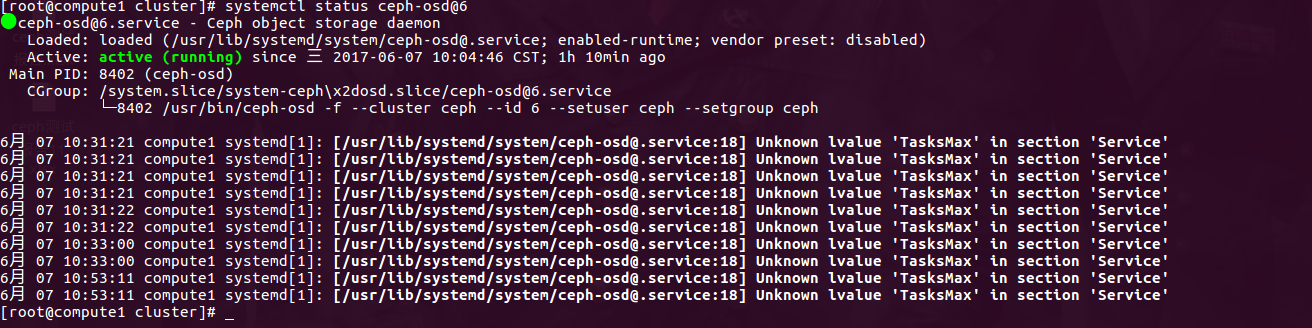
[root@compute3 ~]# **ceph-disk activate-all**

Created symlink from /run/systemd/system/ceph-osd.target.wants/ceph-osd@5.service to /usr/lib/systemd/system/ceph-osd@.service.

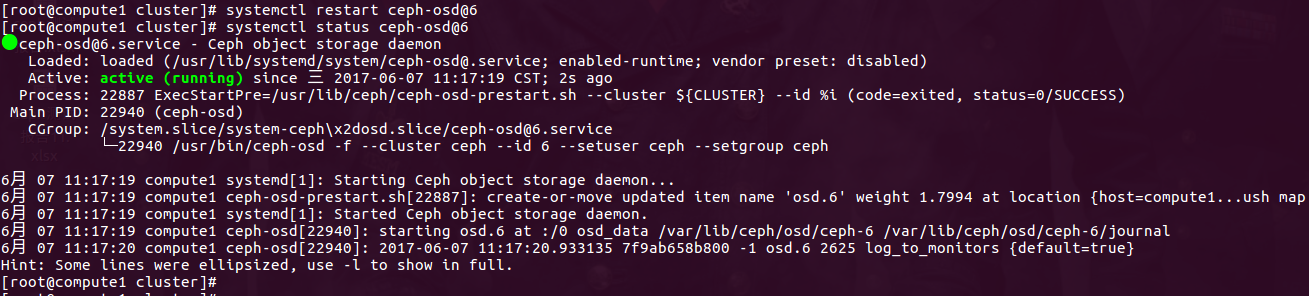
Created symlink from /run/systemd/system/ceph-osd.target.wants/ceph-osd@4.service to /usr/lib/systemd/system/ceph-osd@.service.

[root@compute3 ~]#

## 升级systemd



升级完systemd，然后重启就OK了



## 清理rbd pool中的object

Awesome little script, this helps me cleanup after a failed openstack deployment.

I changed it a little and added this

for ii in $(ceph osd pool ls) ; do

for i in $(rados -p $ii ls); do echo $i; rados -p $ii rm $i; done ;

done

## 修复ceph pg 问题：candidate had a read error

**<http://www.yangguanjun.com/2017/05/31/ceph-pg-inconsistent-error/>**

**<https://ceph.com/geen-categorie/ceph-manually-repair-object/>**

**1 pgs inconsistent, 1 scrub errors**

**candidate had a read error**

[root@compute3 ~]# **ceph -s**

cluster 369fc051-5f04-44b9-be69-d547439b65f7

health HEALTH\_ERR

1 pgs inconsistent

1 scrub errors

monmap e2: 3 mons at {compute1=10.0.0.31:6789/0,compute2=10.0.0.32:6789/0,compute3=10.0.0.33:6789/0}

election epoch 140, quorum 0,1,2 compute1,compute2,compute3

osdmap e1909: 12 osds: 11 up, 11 in

flags sortbitwise,require\_jewel\_osds

pgmap v7184899: 463 pgs, 4 pools, 1902 GB data, 429 kobjects

5687 GB used, 4328 GB / 10016 GB avail

462 active+clean

1 active+clean+inconsistent

client io 135 kB/s wr, 0 op/s rd, 14 op/s wr

[root@compute3 ~]# **ceph health detail**

HEALTH\_ERR 1 pgs inconsistent; 1 scrub errors

pg 4.72 is active+clean+inconsistent, acting [11,7,1]

1 scrub errors

[root@compute3 ~]# **grep -Hn 'ERR' /var/log/ceph/ceph-osd.11.log**

/var/log/ceph/ceph-osd.11.log:1061:2018-01-22 09:15:27.597949 7fd412a54700 -1 log\_channel(cluster) log [ERR] : 4.72 shard 11: soid 4:4fe6b6b4:::**rbd\_data.28ca3238e1f29.0000000000000763**:head candidate had a read error

/var/log/ceph/ceph-osd.11.log:1212:2018-01-22 09:19:43.985770 7fd41024f700 -1 log\_channel(cluster) log [ERR] : 4.72 shard 11: soid 4:4fe6b6b4:::rbd\_data.28ca3238e1f29.0000000000000763:head candidate had a read error

[root@compute3 ~]#

## 分析 & 解决

手动执行pg修复

ceph pg repair 14.16a

ceph pg deep-scrub 14.16a

结果：集群状态依旧HEALTH\_ERR

## 重启对应osd daemon

systemctl restart ceph-osd@<osdid>.service

结果：集群状态依旧HEALTH\_OK

[root@compute3 DIR\_7]# pwd

/var/lib/ceph/osd/ceph-11/current/4.72\_head/DIR\_2/DIR\_F/DIR\_7

[root@compute3 DIR\_7]# ll rbd\\udata.**28ca3238e1f29.0000000000000763**\_\_head\_2D6D67F2\_\_4

-rw-r--r-- 1 ceph ceph 4194304 1月 22 10:07 rbd\udata.28ca3238e1f29.0000000000000763\_\_head\_2D6D67F2\_\_4

#### 需要找到这个object：**28ca3238e1f29.0000000000000763**

### compute1节点

[root@compute1 DIR\_F]# pwd

/var/lib/ceph/osd/ceph-1/current/4.72\_head/DIR\_2/DIR\_F

[root@compute1 DIR\_F]#

[root@compute1 DIR\_F]# md5sum rbd\\udata.28ca3238e1f29.0000000000000763\_\_head\_2D6D67F2\_\_4

\b4b7e4019b202360feb6c95286b058e4 rbd\\udata.28ca3238e1f29.0000000000000763\_\_head\_2D6D67F2\_\_4

### compute2节点

[root@compute2 DIR\_7]# pwd

/var/lib/ceph/osd/ceph-7/current/4.72\_head/DIR\_2/DIR\_F/DIR\_7

[root@compute2 DIR\_7]# md5sum rbd\\udata.28ca3238e1f29.0000000000000763\_\_head\_2D6D67F2\_\_4

\aa04fdff117c2964c1c8c9ad4a053707 rbd\\udata.28ca3238e1f29.0000000000000763\_\_head\_2D6D67F2\_\_4

[root@compute2 DIR\_7]#

### compute3 节点

[root@compute3 DIR\_7]# pwd

/var/lib/ceph/osd/ceph-11/current/4.72\_head/DIR\_2/DIR\_F/DIR\_7

[root@compute3 DIR\_7]#

[root@compute3 DIR\_7]# md5sum rbd\\udata.28ca3238e1f29.0000000000000763\_\_head\_2D6D67F2\_\_4

\aa04fdff117c2964c1c8c9ad4a053707 rbd\\udata.28ca3238e1f29.0000000000000763\_\_head\_2D6D67F2\_\_4

[root@compute3 DIR\_7]#

## 维护 OSD

STOPPING W/OUT REBALANCING

Periodically, you may need to perform maintenance on a subset of your cluster, or resolve a problem that affects a failure domain (e.g., a rack). If you do not want CRUSH to automatically rebalance the cluster as you stop OSDs for maintenance, set the cluster to noout first:

ceph osd set noout

Once the cluster is set to noout, you can begin stopping the OSDs within the failure domain that requires maintenance work.

stop ceph-osd id={num}

Note Placement groups within the OSDs you stop will become degraded while you are addressing issues with within the failure domain.

Once you have completed your maintenance, restart the OSDs.

start ceph-osd id={num}

Finally, you must unset the cluster from noout.

ceph osd unset noout

## 删除ceph中的openstack的volume

### 查看snapshot

[root@compute1 ~]# rbd -p volume list|grep fc3112ca-ba73-4935-aca0-07a5a0186a58

**volume-fc3112ca-ba73-4935-aca0-07a5a0186a58**

[root@compute1 ~]#

[root@compute1 ~]# rbd snap ls volume/volume-fc3112ca-ba73-4935-aca0-07a5a0186a58

SNAPID NAME SIZE

28 snapshot-15471697-cc9c-4995-b179-e21b81a9915b 500 GB

### 删除snapshot

[root@compute1 ~]# rbd snap rm \

volume/volume-fc3112ca-ba73-4935-aca0-07a5a0186a58@snapshot-15471697-cc9c-4995-b179-e21b81a9915b

rbd: snapshot 'snapshot-15471697-cc9c-4995-b179-e21b81a9915b2018-02-02 11:37:03.961384 7f23a6548700 -1 librbd::Operations: snapshot is protected' is protected from removal.

[root@compute1 ~]#

### 需要unprotect：

[root@compute1 ~]# rbd snap unprotect \

volume/volume-fc3112ca-ba73-4935-aca0-07a5a0186a58@snapshot-15471697-cc9c-4995-b179-e21b81a9915b

2018-02-02 11:39:17.342994 7fbb7687c700 -1 librbd::SnapshotUnprotectRequest: **cannot unprotect: at least 1 child(ren) [249fbf1af92e7e] in pool 'volume'**

2018-02-02 11:39:17.344413 7fbb7687c700 -1 librbd::SnapshotUnprotectRequest: encountered error: (16) Device or resource busy

2018-02-02 11:39:17.344450 7fbb7687c700 -1 librbd::SnapshotUnprotectRequest: 0x55d8052bbc70 should\_complete\_error: ret\_val=-16

2018-02-02 11:39:17.347721 7fbb7687c700 -1 librbd::SnapshotUnprotectRequest: 0x55d8052bbc70 should\_complete\_error: ret\_val=-16

rbd: unprotecting snap failed: (16) Device or resource busy

[root@compute1 ~]#

### 删除snapshot的children

**查看某个snapshot有没有被clone过（使用children命令), 发现有children时，去要flatten。**

**注意：需要做flatten的是children，而不是原来的镜像**

[root@compute1 ~]# rbd **children** \

volume/volume-fc3112ca-ba73-4935-aca0-07a5a0186a58@snapshot-15471697-cc9c-4995-b179-e21b81a9915b

*volume/volume-29b9cdd8-f2f9-419a-a571-33401dc832aa*

[root@compute1 ~]#

*FLATTENING A CLONED IMAGE*

*Cloned images retain a reference to the parent snapshot. When you remove the reference from the child clone to the parent snapshot, you effectively “flatten” the image by copying the information from the snapshot to the clone. The time it takes to flatten a clone increases with the size of the snapshot. To delete a snapshot, you must flatten the child images first.*

[root@compute1 ~]# rbd flatten *volume/volume-29b9cdd8-f2f9-419a-a571-33401dc832aa*

Image flatten: 100% complete...

### 最后，可以直接删除

[root@compute1 ~]# rbd snap unprotect \

volume/volume-fc3112ca-ba73-4935-aca0-07a5a0186a58@snapshot-15471697-cc9c-4995-b179-e21b81a9915b

[root@compute1 ~]# rbd snap rm \

volume/volume-fc3112ca-ba73-4935-aca0-07a5a0186a58@snapshot-15471697-cc9c-4995-b179-e21b81a9915b

[root@compute1 ~]# rbd snap ls volume/volume-fc3112ca-ba73-4935-aca0-07a5a0186a58

[root@compute1 ~]#

## 扩张ceph集群方法

### 先设置ceph集群停止rebalancing