Ceph 集群搭建和使用

Name：曲中岭

Email:zlinggu@126.com
Q Q :441869115

第一章 搭建

1.1 准备

1.1.1 选择版本

版本发行说明:
https://ceph.com/category/releases/
其中有些是 LTS 版本，有些是 bug 修复版本，建议使用有 LTS 标记的版本，我这里选择 v13.2.5

版本发行简略信息:
http://docs.ceph.com/docs/master/releases/schedule/

1.1.2 服务器准备

admin 节点：

1.2 创建集群

admin 上操作，如下，可同时设置多个 mon 节点。

```
mkdir ceph-cluster && cd ceph-cluster
ceph new ceph-1
ceph new ceph-1 ceph-e ceph-3
```

执行完后，会出现以下三个文件:
ceph.conf
ceph.log
ceph.mon.keyring

如果有需要，请修改 ceph.conf 文件，比如网络部分、IPV6 支持等。

1.3 安装相关包

```
ceph install ceph-1 ceph-2 ceph-3
```

这个命令会在各个节点执行

```
yum install –y ceph-radosgw
```

1.4 MON 节点

1.4.1 部署 mon

部署 mon 就是集群初始化的过程

```
ceph-deploy mon create-initial
```

执行完成这一步之后，会生产如下文件

- ceph.client.admin.keyring
- ceph.bootstrap-mgr.keyring
- ceph.bootstrap-osd.keyring
- ceph.bootstrap-mds.keyring
- ceph.bootstrap-rgw.keyring
- ceph.bootstrap-rbd.keyring
- ceph.bootstrap-rbd-mirror.keyring

1.4.2 删除 mon

必要时可以删除 mon

```
ceph-deploy mon destroy ceph-1
```

1.5 密钥管理

收集密钥，在继续下一步之前，必须收集密钥

```
ceph-deploy gatherkeys {monitor-host}
```

1.6 MGR 节点（选做）

MGR 是新有的功能，旧版本没有。提供了一个 web 展示界面，供查询、管理、监控等功能。

1.6.1 创建 mgr

```
ceph-deploy mgr create node2 node3
```

多个 MGR 将处于主备模式，只有一个为主节点。
执行完这一步之后，将会在 ceph.conf 中添加有关 MGR 的相关配置

1.6.2 删除 mgr

未做测试

1.7 MDS 节点（选做）

如果要使用 ceph 的文件存储功能，才需要安装。MDS 节点又叫做元数据服务器。

1.7.1 创建 mds

```
ceph-deploy mds create ceph-1
```

执行完这一步之后，自动更新 ceph.conf 文件，添加有关 MDS 的信息。ceph-s 表示节点名

1.7.2 删除 mds

暂时不支持

1.8 RGW 节点（选做）

如果要使用 ceph 的对象存储功能，才需要安装

1.8.1 创建 rgw

```
ceph-deploy rgw create {gateway-node}
```

执行完这一步之后，将会在 ceph.conf 中添加有关 RGW 的配置。

1.8.2 删除 rgw

未做测试

1.9 配置信息分发

1.9.1 首次分发

经过以上步骤之后,相关配置信息已经写道 ceph.conf 文件中,可使用如下命令,把 ceph.conf 和 ceph.client.admin.keyring 文件分发到指定的节点的/etc/ceph 路径下面。

```
ceph-deploy admin ceph-1 ceph-2 ceph-3
```

1.9.2 更新配置

当配置信息更新后，比如新增了 mgr 节点，ceph.conf 会更新，可使用如下命令进行更新推送

```
ceph-deploy config push {host-name [host-name]...}
```

1.9.3 收集配置

可能由于某种情况，admin 节点的配置信息丢失了，可使用如下命令将远程的配置信息收到到本地。

```
ceph-deploy config pull {host-name [host-name]...}
```

经测试，是收集了 ceph.conf 文件到当前目录

1.10 OSD 节点

既是添加 OSD 的过程，也是扩容 OSD 的过程。

1.10.1 添加 osd

列出磁盘

```
ceph-deploy disk list slave03
```

清空磁盘数据，官网中的文档的写法错了

```
ceph-deploy disk zap slave03 /dev/sdc
```

```
[root@master ceph-cluster]# ceph-deploy disk zap slave03 /dev/sdc
[ceph_deploy.conf][DEBUG ] found configuration file at: /root/.cephdeploy.conf
[ceph_deploy.cli][INFO  ] Invoked (2.0.1): /usr/bin/ceph-deploy disk zap slave03 /dev/sdc
[ceph_deploy.cli][INFO  ] ceph-deploy options:
[ceph_deploy.cli][INFO  ]  username                      : None
[ceph_deploy.cli][INFO  ]  verbose                       : False
[ceph_deploy.cli][INFO  ]  debug                         : False
[ceph_deploy.cli][INFO  ]  overwrite_conf                : False
[ceph_deploy.cli][INFO  ]  subcommand                    : zap
[ceph_deploy.cli][INFO  ]  quiet                         : False
[ceph_deploy.cli][INFO  ]  cd_conf                       : <ceph_deploy.conf.cephdeploy.Conf instance at
[ceph_deploy.cli][INFO  ]  cluster                       : ceph
[ceph_deploy.cli][INFO  ]  host                          : slave03
[ceph_deploy.cli][INFO  ]  func                          : <function disk at 0x7f7c216c48c0>
[ceph_deploy.cli][INFO  ]  ceph_conf                     : None
[ceph_deploy.cli][INFO  ]  default_release               : False
[ceph_deploy.cli][INFO  ]  disk                          : ['/dev/sdc']
[ceph_deploy.osd][DEBUG ] zapping /dev/sdc on slave03
[slave03][DEBUG ] connected to host: slave03
[slave03][DEBUG ] detect platform information from remote host
[slave03][DEBUG ] detect machine type
[slave03][DEBUG ] find the location of an executable
[ceph_deploy.osd][INFO  ] Distro info: CentOS Linux 7.5.1804 Core
[slave03][DEBUG ] zeroing last few blocks of device
[slave03][DEBUG ] find the location of an executable
[slave03][INFO  ] Running command: /usr/sbin/ceph-volume lvm zap /dev/sdc
[slave03][DEBUG ] --> Zapping: /dev/sdc
[slave03][DEBUG ] --> --destroy was not specified, but zapping a whole device will remove the partition t
[slave03][DEBUG ] Running command: /usr/sbin/wipefs --all /dev/sdc
[slave03][DEBUG ] Running command: /bin/dd if=/dev/zero of=/dev/sdc bs=1M count=10
[slave03][DEBUG ]  stderr: 10+0 records in
[slave03][DEBUG ] 10+0 records out
[slave03][DEBUG ] 10485760 bytes (10 MB) copied
[slave03][DEBUG ]  stderr: , 0.0578632 s, 181 MB/s
[slave03][DEBUG ] --> Zapping successful for: <Raw Device: /dev/sdc>
[root@master ceph-cluster]#
[root@master ceph-cluster]#
```

这一步需要注意，目标磁盘不能被创建 lvm，如果有，请先清空 lvm 信息。

创建 osd

```
ceph-deploy osd create --data /dev/sdc slave03
```

```
[root@master ceph-cluster]#   ceph-deploy osd create --data /dev/sdc slave03
[ceph_deploy.conf][DEBUG ] found configuration file at: /root/.cephdeploy.conf
[ceph_deploy.cli][INFO  ] Invoked (2.0.1): /usr/bin/ceph-deploy osd create --data /dev/sdc slave03
[ceph_deploy.cli][INFO  ] ceph-deploy options:
[ceph_deploy.cli][INFO  ]  verbose                       : False
[ceph_deploy.cli][INFO  ]  bluestore                     : None
[ceph_deploy.cli][INFO  ]  cd_conf                       : <ceph_deploy.conf.cephdeploy.Conf instance a
[ceph_deploy.cli][INFO  ]  cluster                       : ceph
[ceph_deploy.cli][INFO  ]  fs_type                       : xfs
[ceph_deploy.cli][INFO  ]  block_wal                     : None
[ceph_deploy.cli][INFO  ]  default_release               : False
```

创建后，会自动再目标磁盘创建 lvm。ceph 使用 lvm，实现磁盘扩容等功能。

创建后，可如下命令查看是否创建成功

```
ceph osd tree
```

1.10.2 观察

如下图，我添加了 10 个 OSD，osd30-39，使用率比较少，因为还在再平衡中

```
[root@master ~]#
[root@master ~]# ceph osd df
ID CLASS WEIGHT  REWEIGHT SIZE    USE     AVAIL   %USE VAR  PGS
30   hdd 3.82010  1.00000 3.8 TiB 251 GiB 3.6 TiB 6.42 0.90  56
31   hdd 3.82010  1.00000 3.8 TiB 258 GiB 3.6 TiB 6.60 0.92  61
32   hdd 3.82010  1.00000 3.8 TiB 256 GiB 3.6 TiB 6.54 0.92  60
33   hdd 3.82010  1.00000 3.8 TiB 251 GiB 3.6 TiB 6.41 0.90  56
34   hdd 3.82010  1.00000 3.8 TiB 261 GiB 3.6 TiB 6.68 0.93  64
35   hdd 3.82010  1.00000 3.8 TiB 251 GiB 3.6 TiB 6.41 0.90  50
36   hdd 3.82010  1.00000 3.8 TiB 251 GiB 3.6 TiB 6.41 0.90  58
37   hdd 3.82010  1.00000 3.8 TiB 230 GiB 3.6 TiB 5.87 0.82  35
38   hdd 3.82010  1.00000 3.8 TiB 227 GiB 3.6 TiB 5.81 0.81  35
39   hdd 3.82010  1.00000 3.8 TiB 238 GiB 3.6 TiB 6.10 0.85  44
 0   hdd 3.82010  1.00000 3.8 TiB 285 GiB 3.5 TiB 7.29 1.02  88
 1   hdd 3.82010  1.00000 3.8 TiB 275 GiB 3.6 TiB 7.04 0.98  81
 2   hdd 3.82010  1.00000 3.8 TiB 286 GiB 3.5 TiB 7.32 1.02  88
 3   hdd 3.82010  1.00000 3.8 TiB 289 GiB 3.5 TiB 7.38 1.03  89
 4   hdd 3.82010  1.00000 3.8 TiB 303 GiB 3.5 TiB 7.74 1.08 100
 5   hdd 3.82010  1.00000 3.8 TiB 285 GiB 3.5 TiB 7.29 1.02  86
 6   hdd 3.82010  1.00000 3.8 TiB 281 GiB 3.5 TiB 7.19 1.01  89
 7   hdd 3.82010  1.00000 3.8 TiB 290 GiB 3.5 TiB 7.42 1.04  89
 8   hdd 3.82010  1.00000 3.8 TiB 288 GiB 3.5 TiB 7.36 1.03  89
 9   hdd 3.82010  1.00000 3.8 TiB 306 GiB 3.5 TiB 7.82 1.09 107
10   hdd 3.82010  1.00000 3.8 TiB 294 GiB 3.5 TiB 7.52 1.05  92
11   hdd 3.82010  1.00000 3.8 TiB 297 GiB 3.5 TiB 7.60 1.06  97
12   hdd 3.82010  1.00000 3.8 TiB 297 GiB 3.5 TiB 7.60 1.06  99
13   hdd 3.82010  1.00000 3.8 TiB 284 GiB 3.5 TiB 7.27 1.02  88
14   hdd 3.82010  1.00000 3.8 TiB 311 GiB 3.5 TiB 7.95 1.11 109
15   hdd 3.82010  1.00000 3.8 TiB 297 GiB 3.5 TiB 7.60 1.06 100
16   hdd 3.82010  1.00000 3.8 TiB 284 GiB 3.5 TiB 7.27 1.02  87
17   hdd 3.82010  1.00000 3.8 TiB 284 GiB 3.5 TiB 7.27 1.02  84
18   hdd 3.82010  1.00000 3.8 TiB 291 GiB 3.5 TiB 7.43 1.04  91
19   hdd 3.82010  1.00000 3.8 TiB 274 GiB 3.6 TiB 7.02 0.98  78
20   hdd 3.82010  1.00000 3.8 TiB 283 GiB 3.5 TiB 7.24 1.01  85
21   hdd 3.82010  1.00000 3.8 TiB 299 GiB 3.5 TiB 7.64 1.07  98
22   hdd 3.82010  1.00000 3.8 TiB 296 GiB 3.5 TiB 7.57 1.06  96
23   hdd 3.82010  1.00000 3.8 TiB 284 GiB 3.5 TiB 7.26 1.02  88
24   hdd 3.82010  1.00000 3.8 TiB 283 GiB 3.5 TiB 7.22 1.01  87
25   hdd 3.82010  1.00000 3.8 TiB 306 GiB 3.5 TiB 7.83 1.10 104
26   hdd 3.82010  1.00000 3.8 TiB 279 GiB 3.5 TiB 7.13 1.00  81
27   hdd 3.82010  1.00000 3.8 TiB 298 GiB 3.5 TiB 7.63 1.07  98
28   hdd 3.82010  1.00000 3.8 TiB 298 GiB 3.5 TiB 7.62 1.07  96
29   hdd 3.82010  1.00000 3.8 TiB 281 GiB 3.5 TiB 7.19 1.01  81
                    TOTAL 153 TiB  11 TiB 142 TiB 7.15
MIN/MAX VAR: 0.81/1.11  STDDEV: 0.54
[root@master ~]#
```

使用 ceph -s 可以看到其状态，处于再平衡中。时间的长短，和当前已有的数据量以及新增的 osd 数量有关

```
[root@master ~]#
[root@master ~]# ceph health detail
HEALTH_WARN 310947/1919178 objects misplaced (16.202%)
OBJECT_MISPLACED 310947/1919178 objects misplaced (16.202%)
[root@master ~]#
[root@master ~]#
[root@master ~]# ceph -s
  cluster:
    id:     11b96eff-dc3f-45db-8b94-986e7e84ae2e
    health: HEALTH_WARN
            310131/1919190 objects misplaced (16.159%)

  services:
    mon: 3 daemons, quorum master,slave01,slave02
    mgr: master(active), standbys: slave02, slave01
    mds: myfs-1/1/1 up  {0=slave02=up:active}, 2 up:standby
    osd: 40 osds: 40 up, 40 in; 281 remapped pgs

  data:
    pools:   3 pools, 1088 pgs
    objects: 639.7 k objects, 1.3 TiB
    usage:   11 TiB used, 142 TiB / 153 TiB avail
    pgs:     310131/1919190 objects misplaced (16.159%)
             807 active+clean
             274 active+remapped+backfill_wait
             7   active+remapped+backfilling

  io:
    client:   279 KiB/s wr, 0 op/s rd, 21 op/s wr
    recovery: 97 MiB/s, 46 objects/s

[root@master ~]#
```

1.10.3 删除 osd

删除 osd 需要依次进行以下步骤

1）标记为移除，系统会进行数据迁移

ceph osd out osd.30

```
[root@master ceph-cluster]# ceph osd out osd.30
marked out osd.30.
```

可以使用

```
ceph -w
```

命令查看迁移的情况

2）停止 osd 服务

到 osd 的节点上，停止该服务

```
systemctl stop ceph-osd@30
```

```
[root@slave03 ~]#
[root@slave03 ~]# systemctl status ceph-osd@30
● ceph-osd@30.service - Ceph object storage daemon osd.30
   Loaded: loaded (/usr/lib/systemd/system/ceph-osd@.service; ena
   Active: active (running) since Wed 2019-04-17 20:14:48 CST; 22
  Process: 31223 ExecStartPre=/usr/lib/ceph/ceph-osd-prestart.sh
 Main PID: 31228 (ceph-osd)
   CGroup: /system.slice/system-ceph\x2dosd.slice/ceph-osd@30.ser
           └─31228 /usr/bin/ceph-osd -f --cluster ceph --id 30 --

Apr 17 20:14:48 slave03 systemd[1]: Starting Ceph object storage
Apr 17 20:14:48 slave03 systemd[1]: Started Ceph object storage d
Apr 17 20:14:48 slave03 ceph-osd[31228]: starting osd.30 at - osd
Apr 17 20:14:48 slave03 ceph-osd[31228]: 2019-04-17 20:14:48.820
Apr 17 20:14:50 slave03 ceph-osd[31228]: 2019-04-17 20:14:50.553
Apr 17 20:14:50 slave03 ceph-osd[31228]: 2019-04-17 20:14:50.554
[root@slave03 ~]#
[root@slave03 ~]#
[root@slave03 ~]# systemctl stop ceph-osd@30
[root@slave03 ~]#
```

停止之后，状态变为 down

```
29   hdd   3.63820          osd.29        up  1.00000 1.00000
-9         3.63820      host slave03
30   hdd   3.63820          osd.30      down        0 1.00000
[root@master ceph-cluster]#
```

3）从 crush 中移除节点

```
ceph osd purge osd.30 --yes-i-really-mean-it
```

```
[root@master ceph-cluster]#
[root@master ceph-cluster]# ceph osd purge osd.30 --yes-i-really-mean-it
purged osd.30
[root@master ceph-cluster]#
```

执行完这一步之后，使用 ceph osd tree 就看不到这个 osd 了

```
28   hdd   3.63820          osd.28        up  1.00000 1.00000
29   hdd   3.63820          osd.29        up  1.00000 1.00000
-9                0      host slave03
[root@master ceph-cluster]#
```

而且会删除对应的 osd 密钥

4）更新 ceph.conf

如果 ceph.con 中还保留了这个 osd 的相关信息，也需要一并删除，并更新过的配置文件到集群中的其他节点

1.11 清除环境

在某些情况下需要，比如安装出错，重新安装其他集群等，需要清除数据

ceph-deploy purgedata {ceph-node} [{ceph-node}]

注意清除数据前需要目标机器先卸载 ceph 相关包，否则报错

```
[root@master ceph-cluster]#
[root@master ceph-cluster]# ceph-deploy purgedata slave03
[ceph_deploy.conf][DEBUG ] found configuration file at: /root/.cephdeploy.conf
[ceph_deploy.cli][INFO  ] Invoked (2.0.1): /usr/bin/ceph-deploy purgedata slave03
[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  ]  quiet                         : False
[ceph_deploy.cli][INFO  ]  cd_conf                       : <ceph_deploy.conf.cephdeploy.Conf instanc
[ceph_deploy.cli][INFO  ]  cluster                       : ceph
[ceph_deploy.cli][INFO  ]  host                          : ['slave03']
[ceph_deploy.cli][INFO  ]  func                          : <function purgedata at 0x7f7802f688c0>
[ceph_deploy.cli][INFO  ]  ceph_conf                     : None
[ceph_deploy.cli][INFO  ]  default_release               : False
[ceph_deploy.install][DEBUG ] Purging data from cluster ceph hosts slave03
[slave03][DEBUG ] connected to host: slave03
[slave03][DEBUG ] detect platform information from remote host
[slave03][DEBUG ] detect machine type
[slave03][DEBUG ] find the location of an executable
[ceph_deploy.install][ERROR ] Ceph is still installed on: ['slave03']
[ceph_deploy][ERROR ] RuntimeError: refusing to purge data while Ceph is still installed

[root@master ceph-cluster]#
```

也可以使用如下命令，会将安装的 ceph 包和数据一并删除

ceph-deploy purge {ceph-node} [{ceph-node}]

清空后数据和 ceph 相关的只有如下包

```
[root@slave03 ~]#
[root@slave03 ~]# rpm -qa|grep ceph
python-cephfs-13.2.5-0.el7.x86_64
libcephfs2-13.2.5-0.el7.x86_64
[root@slave03 ~]#
[root@slave03 ~]#
[root@slave03 ~]#
```

数据只有下面这些，/etc/ceph 也没了

```
[root@slave03 ~]#
[root@slave03 ~]# ls /var/lib/ceph/ -R
/var/lib/ceph/:
bootstrap-osd  osd

/var/lib/ceph/bootstrap-osd:
ceph.keyring

/var/lib/ceph/osd:
ceph-30

/var/lib/ceph/osd/ceph-30:
activate.monmap  block  bluefs  ceph_fsid  fsid  keyring  kv_backend  magic  mkfs_done  osd_key  ready  type  whoami
[root@slave03 ~]#
[root@slave03 ~]#
[root@slave03 ~]#
```

第二章 使用

2.1 认证

2.1.1 启用认证

使用 ceph-deploy 安装 ceph，默认就是启用认证的，写在 ceph.conf 中。

```
auth cluster required = cephx
```

启用后，ceph 集群内部各个进程之间通信需要认证

```
auth service required = cephx
```

启用后，集群内部各程序需要客户端进行认证

```
auth client required = cephx
```

启用后，客户端需要集群各程序进行认证

如果不启用，将其配置为 none 即可，这在排错时有用。

2.1.2 密钥相关

http://docs.ceph.com/docs/master/rados/configuration/auth-config-ref/

各模块密钥保存位置：/var/lib/ceph/{type}/{id}/keyring

2.1.3 查看用户列表

包括所有用户和权限的详细信息

```
ceph auth list
```

查看单个用户的详细信息

```
ceph auth get client.admin
```

2.1.4 将密钥导出为文件

```
ceph auth get client.admin -o /etc/ceph/ceph.client.admin.keyring
```

2.1.5 修改用户能力

```
ceph-authtool /etc/ceph/ceph.keyring -n client.ringo --cap osd 'allow rwx' --cap mon 'allow rwx'
```

2.1.6 删除用户

```
ceph auth del osd.0
```

2.1.7 生成管理员密钥

```
ceph auth get-or-create client.admin mon 'allow *' mds 'allow *' mgr 'allow *' osd 'allow *' -o ./ceph.client.admin.keyring
```

部署工具已经生成了此密钥，无需重复生成

2.1.8 生成 mon 用户

```
ceph-authtool --create-keyring /tmp/ceph.mon.keyring --gen-key -n mon. --cap mon 'allow *'
```

部署工具已经生成了此密钥，无需重复生成

2.1.9 生成 mgr 用户

```
ceph auth get-or-create mgr.{$id} mon 'allow profile mgr' mds 'allow *' osd 'allow *' -o /var/lib/ceph/mgr/ceph-{$id}/keyring
```

部署工具已经生成了此密钥，无需重复生成

2.1.10 生成 osd 用户

```
ceph auth get-or-create osd.{$id} mon 'allow rwx' osd 'allow *' -o /var/lib/ceph/osd/ceph-{$id}/keyring
```

部署工具已经生成了此密钥，无需重复生成

2.1.11 生成 mds 用户

```
ceph auth get-or-create mds.{$id} mon 'allow rwx' osd 'allow *' mds 'allow *' mgr 'allow profile mds' -o /var/lib/ceph/mds/ceph-{$id}/keyring
```

部署工具已经生成了此密钥，无需重复生成

2.1.12 生成 rdb-client 用户

```
ceph auth get-or-create client.client mon 'allow r' osd 'allow rwx' -o ./ceph.client.client.keyring
```

用于 osd 的 rwx 权限，所以可以用于 rbd 客户端使用

注意 client.client 就是名字，会写到文件中。客户端引用时要匹配

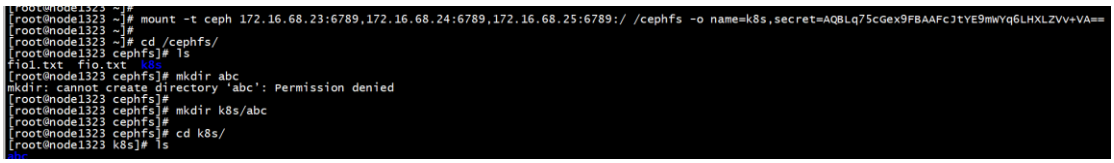
2.1.13 生成 rdb-cephfs 用户

```
ceph auth get-or-create client.cephfs mon 'allow r' mds 'allow r,allow rw path=/' osd 'allow rw pool=cephfs_data' -o ./ceph.client.cephfs.keyring
```

也可用如下命令，生产只对某个目录有权限

```
ceph auth get-or-create client.k8s mon 'allow r' mds 'allow r,allow rw path=/k8s' osd 'allow rw pool=cephfs_data'
```

这样设置后，虽然可以挂到根下，也可以查看，但只能在/k8s 里面进行读写删等，效果如下图，当然也可以不挂到根下



ceph auth get-or-create client.xmc mon 'allow r' mds 'allow r,allow rw path=/xmc' osd 'allow rw pool=cephfs_data' -o client.xmc.secret
ceph auth get-or-create client.mis mon 'allow r' mds 'allow r,allow rw path=/mis' osd 'allow rw pool=cephfs_data' -o client.mis.secret
ceph auth get-or-create client.x2 mon 'allow r' mds 'allow r,allow rw path=/x2' osd 'allow rw pool=cephfs_data' -o client.x2.secret

2.2 测试块存储

2.2.1 安装 ceph

方法 1:
类似于 ansible，做好主机信任，在管理节点上，使用如下命令，控制在远程 client 上安装 ceph 相关软件包
ceph-deploy install ceph-client-1
执行后，client 机器将自动安装如下软件包

```
[root@master ~]#
[root@master ~]# rpm -qa|grep ceph
python-cephfs-13.2.5-0.el7.x86_64
ceph-selinux-13.2.5-0.el7.x86_64
ceph-radosgw-13.2.5-0.el7.x86_64
libcephfs2-13.2.5-0.el7.x86_64
ceph-base-13.2.5-0.el7.x86_64
ceph-osd-13.2.5-0.el7.x86_64
ceph-13.2.5-0.el7.x86_64
ceph-common-13.2.5-0.el7.x86_64
ceph-mds-13.2.5-0.el7.x86_64
ceph-mgr-13.2.5-0.el7.x86_64
ceph-release-1-1.el7.noarch
ceph-mon-13.2.5-0.el7.x86_64
[root@master ~]#
```

并创建配置文件目录

```
[root@master ~]#
[root@master ~]# ls /etc/ceph/ -la
total 16
drwxr-xr-x   2 root root   20 Apr 16 16:58 .
drwxr-xr-x. 95 root root 8192 Apr 16 16:58 ..
-rw-r--r--   1 root root   92 Mar 13 01:42 rbdmap
[root@master ~]#
[root@master ~]# cat /etc/ceph/rbdmap
# RbdDevice             Parameters
#poolname/imagename     id=client,keyring=/etc/ceph/ceph.client.keyring
[root@master ~]#
```

方法 2:

配置 ceph 的 yum 源，其中 mimic 表示版本，应该选择 LTS 版本

```
yum install https://mirrors.aliyun.com/ceph/rpm-mimic/el7/noarch/ceph-release-1-1.el7.noarch.rpm
```

我这里选择阿里云的仓库，也可以使用其他仓库，看自己喜欢。

有些依赖需要 epel 源，也进行配置下:

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

然后使用如下命令安装

```
yum install ceph -y
```

2.2.2 生成客户端密钥

管理服务器上，执行如下命令

```
eph auth get-or-create client.rbd mon 'allow r' osd 'allow rwx' -o ./ceph.client.rbd.keyring
```

注意名字必须是*.*格式，否则报错

```
[root@master ~]# ceph auth get-or-create clientx mon 'allow r' osd 'allow rwx' -o ./ceph.client.keyring
Error EINVAL: bad entity name
[root@master ~]#
[root@master ~]#
```

提醒：一定不要使用默认的 ceph.client.admin.keyring，权限太大了

2.2.3 传递密钥和配置

在管理服务器上，将生成的 ceph.client.rbd.keyring 和 ceph.conf 文件复制到 client 的/etc/ceph 目录下，并保证前者有读权限

```
chmod +r /etc/ceph/ceph.client.rbd.keyring
```

注意这里，不要使用如下命令，因为会将 client.admin 分发到 client 节点，权限太大了，需要自己生成 client 的密钥。

```
ceph-deploy admin ceph-client
```

2.2.4 创建 pool

如果已经有了，可以不执行这一步。

admin node 节点上执行：

```
ceph osd pool create mypool 64
rbd pool init mypool
```

init 初始化之后，这里才显示为 rbd 类型，否则显示为空。

Name	Type	Applications
cephfs_data	replicated	cephfs
cephfs_metadata	replicated	cephfs
mypool	replicated	rbd
rbd	replicated	rbd

0 selected / 4 total

2.2.5 创建块设备

admin 权限的节点上，创建块设备 image

格式：

rbd create foo --size 4096 --image-feature layering [-m {mon-IP}] [-k /path/to/ceph.client.admin.keyring] [-p {pool-name}]

-m -k 等都可以省了，-p 如果省略，使用默认值 rbd

默认单位是 M，也可以使用 G、T 等单位

```
rbd create node1323 --size 200G --image-feature layering
rbd create foo2 --size 40G --image-feature layering -p mypool
```

比如使用如下语句创建

```
rbd create foo --size 4096 --image-feature layering -p mypool
```

rbd create mypool/foo1 --size 4096 --image-feature layering

```
[root@localhost ~]#
[root@localhost ~]# rbd create foo --size 4096 --image-feature layering -p mypool
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# rbd create mypool/foo1 --size 4096 --image-feature layering
[root@localhost ~]#
[root@localhost ~]#
```

查看已经创建的设备

rbd ls -l rbd

rbd ls -l

```
[root@master ~]# rbd ls -l
NAME         SIZE PARENT FMT PROT LOCK
node1121 200 GiB            2
node1122 200 GiB            2
node1123 200 GiB            2
node1221 200 GiB            2
node1222 200 GiB            2
node1223 200 GiB            2
node1321 200 GiB            2
node1322 200 GiB            2
node1323 200 GiB            2
[root@master ~]#
```

创建映射设备之后如果查看 pool 中的对象，可以发现类似如下内容

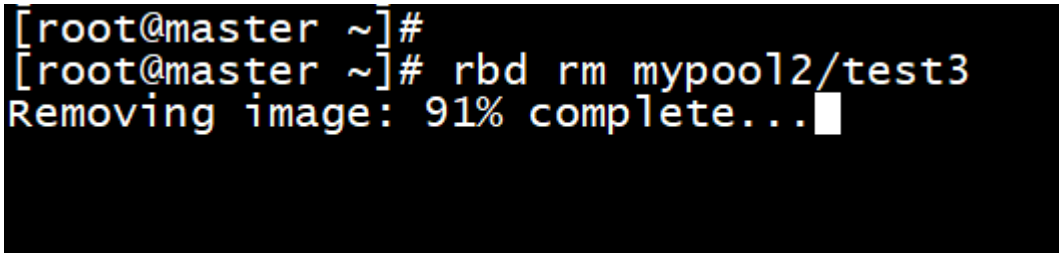
```
rbd_data.af1c6b8b4567.0000000000001428
[root@localhost ~]# rados -p mypool ls|grep id
rbd_id.foo2
rbd_id.foo1
rbd_id.foo
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]#
```

对于不同的使用场景，可使用优化配置，例如设置 objece-size=4K，对小文件的存储会更好。
rbd create mypool2/test3 --size 4096 --object-size 4K --image-feature layering

系统默认清空下，strping-unit 参数和 object-size 参数相同，所以只需要修改后者即可。

Size	4GiB
Objects	1.0486M
Object size	4KiB
Features	layering
Provisioned	N/A
Total provisioned	N/A
Striping unit	4KiB
Striping count	1
Parent	-
Block name prefix	rbd_data.1e7cb6b8b4567
Order	12

另外观察到一个现象，如果设置了 object-size 为 4k 后，删除 image 会非常慢



更多优化参数:
http://docs.ceph.org.cn/man/8/rbd/#id7

2.2.6 映射块设备

在 ceph-client 节点上，映射块设备
语法: rbd map foo --name client.admin [-m {mon-IP}] [-k /path/to/ceph.client.admin.keyring] [-p {pool-name}]
--name 设置 name，这个 name 就是 keyring 文件里面的
-m 指定 mon 节点，已经在 ceph.conf 中了，可以不用指定
-k 设置 keyring 的路径，默认查找位置:

- /etc/ceph/ceph.{name}.keyring
- /etc/ceph/ceph.keyring
- /etc/ceph/keyring
- /etc/ceph/keyring.bin

-p pool 的名字，默认是 rbd

```
rbd map mypool2/test --name client.rbd

rbd map node1323 --name client.rbd
```

```
[root@node1323 ceph]#
[root@node1323 ceph]# rbd map node1323 --name client.rbd
/dev/rbd0
[root@node1323 ceph]#
[root@node1323 ceph]#
[root@node1323 ceph]# cat /etc/ceph/ceph.client.rbd.keyring
[client.rbd]
        key = AQD6rb5cfIy5GRAA4hi7z7kN8B2t5KG4oOqKwQ==
[root@node1323 ceph]#
[root@node1323 ceph]#
```

创建完成的块设备，在/dev/rbd/POOL_NAME/RBD_NAME 都有对应的链接文件存在，如下，路径中的第二个 rbd 表示 pool 的名字

```
[root@node1323 ceph]#
[root@node1323 ceph]# ll /dev/rbd/rbd/node1323
lrwxrwxrwx 1 root root 10 Apr 23 14:26 /dev/rbd/rbd/node1323 -> ../../rbd0
[root@node1323 ceph]#
[root@node1323 ceph]# ll /dev/rbd0
brw-rw---- 1 root disk 252, 0 Apr 23 14:26 /dev/rbd0
[root@node1323 ceph]#
[root@node1323 ceph]#
```

使用 fdisk -l 也可以看到，类似于我们使用 fdisk 对磁盘做分区，所以下一步可以直接进行格式化操作

```
[root@node1323 ceph]# fdisk -l /dev/rbd0

Disk /dev/rbd0: 214.7 GB, 214748364800 bytes, 419430400 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 4194304 bytes / 4194304 bytes

[root@node1323 ceph]#
```

映射时如果出现以下错误，这是因为 image 支持的一些特性，OS kernel 并不支持，所以要在上一步创建 image 时使用参数 --image-feature layering

```
[root@localhost ~]#
[root@localhost ~]# rbd info mypool/foo17
rbd: error opening image foo17: (2) No such file or directory
[root@localhost ~]#
[root@localhost ~]# rbd info mypool/foo17
rbd image 'foo17':
        size 4 GiB in 1024 objects
        order 22 (4 MiB objects)
        id: afed6b8b4567
        block_name_prefix: rbd_data.afed6b8b4567
        format: 2
        features: layering, exclusive-lock, object-map, fast-diff, deep-flatten
        op_features:
        flags:
        create_timestamp: Wed Apr 17 13:54:29 2019
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# rbd map mypool/foo17
rbd: sysfs write failed
RBD image feature set mismatch. You can disable features unsupported by the kernel with "rbd feature disable mypool/foo17 object-map fast-diff deep-flatten".
In some cases useful info is found in syslog - try "dmesg | tail".
rbd: map failed: (6) No such device or address
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]#
```

2.2.7 查看已经映射的块设备

```
rbd showmapped
```

```
[root@localhost ~]# rbd showmapped
id pool   image snap device
0  mypool foo   -    /dev/rbd0
1  mypool foo1  -    /dev/rbd1
2  mypool foo2  -    /dev/rbd2
3  mypool foo14 -    /dev/rbd3
4  mypool foo15 -    /dev/rbd4
5  mypool foo16 -    /dev/rbd5
[root@localhost ~]#
```

2.2.8 删除映射

```
rbd unmap /dev/rbd0
```

```
[root@localhost ~]# rbd showmapped
id pool   image snap device
0  mypool foo   -    /dev/rbd0
1  mypool foo1  -    /dev/rbd1
2  mypool foo2  -    /dev/rbd2
3  mypool foo14 -    /dev/rbd3
4  mypool foo15 -    /dev/rbd4
5  mypool foo16 -    /dev/rbd5
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# rbd unmap /dev/rbd0
rbd: sysfs write failed
rbd: unmap failed: (16) Device or resource busy
[root@localhost ~]#
[root@localhost ~]# umount /rbd0/
[root@localhost ~]#
[root@localhost ~]# rbd unmap /dev/rbd0
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# rbd showmapped
id pool   image snap device
1  mypool foo1  -    /dev/rbd1
2  mypool foo2  -    /dev/rbd2
3  mypool foo14 -    /dev/rbd3
4  mypool foo15 -    /dev/rbd4
5  mypool foo16 -    /dev/rbd5
[root@localhost ~]#
```

已经挂载的设备不允许取消，必须先卸载

2.2.9 格式化和挂载

格式化时，有两种方法如下图

```
mkfs.xfs /dev/rbd0
```

```
[root@localhost ~]# mkfs.xfs /dev/rbd0
meta-data=/dev/rbd0              isize=512    agcount=8, agsize=131072 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=0, sparse=0
data     =                       bsize=4096   blocks=1048576, 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@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# mkfs.xfs /dev/rbd/mypool/foo1
meta-data=/dev/rbd/mypool/foo1   isize=512    agcount=8, agsize=131072 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=0, sparse=0
data     =                       bsize=4096   blocks=1048576, 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@localhost ~]#
[root@localhost ~]# mkfs.xfs /dev/rbd/mypool/foo2
meta-data=/dev/rbd/mypool/foo2   isize=512    agcount=16, agsize=655360 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=0, sparse=0
data     =                       bsize=4096   blocks=10485760, imaxpct=25
         =                       sunit=1024   swidth=1024 blks
naming   =version 2              bsize=4096   ascii-ci=0 ftype=1
log      =internal log           bsize=4096   blocks=5120, version=2
         =                       sectsz=512   sunit=8 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
[root@localhost ~]#
```

磁盘挂载，自动识别文件系统

mount /dev/rbd1 /rbd1

```
[root@localhost ~]#
[root@localhost ~]# mkdir /rbd0
[root@localhost ~]# mkdir /rbd1
[root@localhost ~]# mkdir /rbd2
[root@localhost ~]#
[root@localhost ~]# mount /dev/rbd0 /rbd0
[root@localhost ~]#
[root@localhost ~]# mount /dev/rbd1 /rbd1
[root@localhost ~]#
```

然后就可以使用/rbd1、 /rbd2 等设备了，如下图，就是挂在后的 rbd 设备，大小 200G。

```
[root@node1323 ceph]#
[root@node1323 ceph]# mount /dev/rbd0 /data/
[root@node1323 ceph]# df -h
Filesystem                                              Size  Used Avail Use% Mounted on
/dev/vda3                                               142G  2.1G  140G   2% /
devtmpfs                                                8.1G     0  8.1G   0% /dev
tmpfs                                                   8.1G     0  8.1G   0% /dev/shm
tmpfs                                                   8.1G  8.8M  8.1G   1% /run
tmpfs                                                   8.1G     0  8.1G   0% /sys/fs/cgroup
/dev/vda1                                               197M  115M   82M  59% /boot
tmpfs                                                   8.1G   12K  8.1G   1% /var/lib/kubelet/pc
tmpfs                                                   8.1G   12K  8.1G   1% /var/lib/kubelet/pc
overlay                                                 142G  2.1G  140G   2% /var/lib/docker/ove
overlay                                                 142G  2.1G  140G   2% /var/lib/docker/ove
shm                                                      64M     0   64M   0% /var/lib/docker/cor
shm                                                      64M     0   64M   0% /var/lib/docker/cor
overlay                                                 142G  2.1G  140G   2% /var/lib/docker/ove
overlay                                                 142G  2.1G  140G   2% /var/lib/docker/ove
tmpfs                                                   1.7G     0  1.7G   0% /run/user/0
172.16.68.23:6789,172.16.68.24:6789,172.16.68.25:6789:/  35T  1.1G   35T   1% /cephfs
/dev/rbd0                                               200G   33M  200G   1% /data
[root@node1323 ceph]#
[root@node1323 ceph]#
```

2.2.10 查看谁挂载了 image

```
rbd status rbd/node1323
```

2.2.11 开机自动挂载

1） 如下图，修改/etc/ceph/rbdmap 内容，添加如下一行

```
node1323            id=rbd,keyring=/etc/ceph/ceph.client.rbd.keyring
```

```
anaconda-ks.cfg
[root@node1323 ~]# vim /etc/ceph/rbdmap
# RbdDevice             Parameters
#poolname/imagename     id=client,keyring=/etc/ceph/ceph.client.keyring
node1323        id=rbd,keyring=/etc/ceph/ceph.client.rbd.keyring
~
~
~
```

注意：id 段是如下文件中的内容，不可乱写，否则失败

```
[root@node1323 ~]# cat /etc/ceph/ceph.client.rbd.keyring
[client.rbd]
        key = AQD6rb5cfIy5GRAA4hi7z7kN8B2t5KG4oOqKwQ==
[root@node1323 ~]#
[root@node1323 ~]#
```

keyring 字段是密钥地址

2） 修改/etc/fstab 内容，添加如下一行

```
/dev/rbd/rbd/mongo1350                  /data                   xfs
noauto          0 0
```

```
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
UUID=7a1c9c78-29ba-414f-bd69-ad7d49f321b3 /                       xfs     defaults        0 0
UUID=b51f2951-5ead-499c-99a8-3bdb795b1779 /boot                   xfs     defaults        0 0
#UUID=0O0d832f-0f89-45e7-9c0b-1f179259df61 swap                    swap    defaults        0 0
/dev/rbd/rbd/node1323 /data xfs noauto 0 0
~
~
~
~
```

注意，一定要加 noauto，防止过早挂载，因为这个需要 rdbmap.service 服务调用脚本创建 rbd 设备后才能挂载的，不能太早挂载

3）设置 rbdmap.service 服务开机自启

```
systemctl enable rbdmap.service
```

```
[root@node1323 ceph]#
[root@node1323 ceph]# systemctl list-unit-files|grep rbd
rbdmap.service                              disabled
[root@node1323 ceph]#
[root@node1323 ceph]#
[root@node1323 ceph]#
[root@node1323 ceph]# systemctl enable rbdmap.service
Created symlink from /etc/systemd/system/multi-user.target.wants/rbdmap.service to /usr/lib/systemd/system/rbdmap.service.
[root@node1323 ceph]#
[root@node1323 ceph]# systemctl list-unit-files|grep rbd
rbdmap.service                              enabled
[root@node1323 ceph]#
[root@node1323 ceph]#
[root@node1323 ceph]# rpm -qf /usr/lib/systemd/system/rbdmap.service
ceph-common-13.2.5-0.el7.x86_64
```

该服务由 ceph-common 这个包提供。

参考文档:
http://docs.ceph.com/docs/master/man/8/rbdmap/

2.2.12 快照和回滚

a) 如下图已经创建了两个文件夹

```
[root@node1323 data]#
[root@node1323 data]# pwd
/data
[root@node1323 data]#
[root@node1323 data]# df -h /data/
Filesystem      Size  Used Avail Use% Mounted on
/dev/rbd0       200G   33M  200G   1% /data
[root@node1323 data]#
[root@node1323 data]# ls
a.txt  b.txt
[root@node1323 data]#
```

b) 此时创建快照

```
rbd snap create --snap mysnap rbd/node1323
```

```
[root@master ~]# rbd snap create --snap mysnap rbd/node1323
[root@master ~]#
```

c) 查看创建的快照

```
rbd snap list rbd/node1323
```

```
[root@master ~]# rbd snap list rbd/node1323
SNAPID NAME       SIZE TIMESTAMP
     4 mysnap 200 GiB Tue Apr 23 15:29:21 2019
[root@master ~]#
[root@master ~]#
[root@master ~]# rbd snap list node1323
SNAPID NAME       SIZE TIMESTAMP
     4 mysnap 200 GiB Tue Apr 23 15:29:21 2019
[root@master ~]#
[root@master ~]#
[root@master ~]# rbd snap list node1323 -p rbd
SNAPID NAME       SIZE TIMESTAMP
     4 mysnap 200 GiB Tue Apr 23 15:29:21 2019
[root@master ~]#
```

也可以删除快照

```
rbd snap rm rbd/node1323@mysnap
```

```
[root@master ~]# rbd snap rm rbd/node1323@mysnap
Removing snap: 100% complete...done.
[root@master ~]#
[root@master ~]#
[root@master ~]# rbd snap list rbd/node1323
[root@master ~]#
```

d) 此时再创建一个文件

```
[root@node1323 data]# touch d.txt
[root@node1323 data]#
[root@node1323 data]#
[root@node1323 data]# pwd
/data
[root@node1323 data]#
[root@node1323 data]# ls
a.txt  b.txt  d.txt
[root@node1323 data]#
```

e) 卸载文件系统

一定要先卸载，后续再挂载，否则不生效。

```
umount /data
rbd unmap /dev/rbd0
```

f) 回滚

```
rbd snap rollback rbd/node1323@mysnap
```

```
[root@master ~]# rbd snap rollback rbd/node1323@mysnap
Rolling back to snapshot: 100% complete...done.
[root@master ~]#
[root@master ~]#
```

注意语法

rbd snap rollback POOL_NAME/IMAGE_NAME@SNAP_NAME

g) 重新挂载和 map

发现恢复到之前的状态

```
[root@node1323 ~]#
[root@node1323 ~]#
[root@node1323 ~]# rbd unmap /dev/rbd0
[root@node1323 ~]#
[root@node1323 ~]# rbd map node1323 --name client.rbd
/dev/rbd0
[root@node1323 ~]# mount /dev/rbd0 /data
[root@node1323 ~]# ls /data/
a.txt  b.txt
[root@node1323 ~]#
[root@node1323 ~]#
```

2.2.13 快照和克隆

2.2.14 块设备扩容

扩容块设备大小

```
rbd resize --size 81920 mypool/foo2
```

调整文件系统大小

```
resize2fs /dev/rbd2
```

```
/dev/rbd0       4.0G   33M  4.0G   1% /rbd0
/dev/rbd2        40G   49M   38G   1% /rbd2
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# rbd resize --size 81920 mypool/foo2
Resizing image: 100% complete...done.
[root@localhost ~]#
[root@localhost ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda3       214G  3.2G  211G   2% /
devtmpfs        126G     0  126G   0% /dev
tmpfs           126G     0  126G   0% /dev/shm
tmpfs           126G   11M  126G   1% /run
tmpfs           126G     0  126G   0% /sys/fs/cgroup
/dev/sda1      1014M  175M  840M  18% /boot
tmpfs            26G     0   26G   0% /run/user/0
/dev/rbd0       4.0G   33M  4.0G   1% /rbd0
/dev/rbd2        40G   49M   38G   1% /rbd2
[root@localhost ~]#
[root@localhost ~]# resize2fs /dev/rbd2
resize2fs 1.42.9 (28-Dec-2013)
Filesystem at /dev/rbd2 is mounted on /rbd2; on-line resizing required
old_desc_blocks = 5, new_desc_blocks = 10
The filesystem on /dev/rbd2 is now 20971520 blocks long.

[root@localhost ~]#
[root@localhost ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda3       214G  3.2G  211G   2% /
devtmpfs        126G     0  126G   0% /dev
tmpfs           126G     0  126G   0% /dev/shm
tmpfs           126G   11M  126G   1% /run
tmpfs           126G     0  126G   0% /sys/fs/cgroup
/dev/sda1      1014M  175M  840M  18% /boot
tmpfs            26G     0   26G   0% /run/user/0
/dev/rbd0       4.0G   33M  4.0G   1% /rbd0
/dev/rbd2        79G   56M   75G   1% /rbd2
[root@localhost ~]#
[root@localhost ~]# ls /dev/rbd2
/dev/rbd2
[root@localhost ~]# ls /rbd2/
a.txt  lost+found
[root@localhost ~]#
```

resize2fs 命令用于，调整 ext2\ext3\ext4 文件系统的大小，它可以放大或者缩小没有挂载的文件系统的大小。如果文件系统已经挂载，它可以扩大文件系统的大小，前提是内核支持在线调整大小。

如果是 xfs 文件系统，请使用 xfs_growfs 命令

```
xfs_growfs -d /dev/rbd0
```

```
/dev/rbd2        79G   56M   75G   1% /rbd2
/dev/rbd1       7.8G   18M  7.4G   1% /rbd1
/dev/rbd0       8.0G   33M  8.0G   1% /rbd0
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# rbd resize --size 9192 mypool/foo
Resizing image: 100% complete...done.
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# resize2fs /dev/rbd0
resize2fs 1.42.9 (28-Dec-2013)
resize2fs: Bad magic number in super-block while trying to open /dev/rbd0
Couldn't find valid filesystem superblock.
[root@localhost ~]#
[root@localhost ~]# xfs_growfs /dev/rbd0
meta-data=/dev/rbd0              isize=512    agcount=8, agsize=262144 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=0 spinodes=0
data     =                       bsize=4096   blocks=2097152, imaxpct=25
         =                       sunit=1024   swidth=1024 blks
naming   =version 2              bsize=4096   ascii-ci=0 ftype=1
log      =internal               bsize=4096   blocks=2560, version=2
         =                       sectsz=512   sunit=8 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
data blocks changed from 2097152 to 2353152
[root@localhost ~]#
[root@localhost ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda3       214G  3.2G  211G   2% /
devtmpfs        126G     0  126G   0% /dev
tmpfs           126G     0  126G   0% /dev/shm
tmpfs           126G   11M  126G   1% /run
tmpfs           126G     0  126G   0% /sys/fs/cgroup
/dev/sda1      1014M  175M  840M  18% /boot
tmpfs            26G     0   26G   0% /run/user/0
/dev/rbd2        79G   56M   75G   1% /rbd2
/dev/rbd1       7.8G   18M  7.4G   1% /rbd1
/dev/rbd0       9.0G   33M  9.0G   1% /rbd0
[root@localhost ~]#
```

2.2.15　多处挂载测试

先挂载 A，并写入了文件。再挂载 B，可看到 A 写入的，此时 A 的新写入的，B 需要重新挂载可以看到。

```
[root@master ~]# cat /mnt/a.txt
abc
[root@master ~]#
[root@master ~]# umount /mnt
[root@master ~]#
[root@master ~]# mount /dev/rbd0 /mnt
[root@master ~]#
[root@master ~]# ls /mnt/
a.txt  b.txt
[root@master ~]#
```

当在 B 写入时，此时文件系统部分损坏，无论在 A 或则 B，都无法看到 B 写入的内容。后续 A 写入的内容，参考上一步进行查看

```
[root@localhost ~]#
[root@localhost ~]# ls /rbd0/
ls: cannot access /rbd0/c.txt: No such file or directory
a.txt  b.txt  c.txt
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# echo cccc > /rbd0/d.txt
[root@localhost ~]#
[root@localhost ~]# ls /rbd0/
ls: cannot access /rbd0/c.txt: No such file or directory
a.txt  b.txt  c.txt  d.txt
[root@localhost ~]#
```

如果出现文件系统损坏，可使用以下方法进行修复

umount /rbd0/

xfs_repair /dev/rbd0

```
[root@localhost ~]# cat /rbd0/c.txt
cat: /rbd0/c.txt: No such file or directory
[root@localhost ~]# cat /rbd0/d.txt
cccc
[root@localhost ~]#
[root@localhost ~]# xfs_repair /dev/rbd0
xfs_repair: /dev/rbd0 contains a mounted filesystem
xfs_repair: /dev/rbd0 contains a mounted and writable filesystem

fatal error -- couldn't initialize XFS library
[root@localhost ~]#
[root@localhost ~]# umount /rbd0/
[root@localhost ~]#
[root@localhost ~]# xfs_repair /dev/rbd0
Phase 1 - find and verify superblock...
Phase 2 - using internal log
        - zero log...
        - scan filesystem freespace and inode maps...
sb_ifree 58, counted 57
sb_fdblocks 2094537, counted 2094536
        - found root inode chunk
Phase 3 - for each AG...
        - scan and clear agi unlinked lists...
        - process known inodes and perform inode discovery...
        - agno = 0
imap claims a free inode 8197 is in use, correcting imap and clearing inod
cleared inode 8197
```

修复后

```
[root@localhost ~]# mount /dev/rbd0 /rbd0/
[root@localhost ~]# ls /rbd0/
a.txt  b.txt  d.txt
[root@localhost ~]#
[root@localhost ~]# cat /rbd0/a.txt
abc
[root@localhost ~]# cat /rbd0/b.txt
lfdjslfd
[root@localhost ~]# cat /rbd0/d.txt
cccc
[root@localhost ~]#
```

2.3 测试文件存储

2.3.1 启用 MDS

要使用 CephFS，必须部署 mds 节点，参考其他章节

2.3.2 创建 pool

CephFS 需要创建两个 pool，分别用于存放 data 和 metadata

语法：

```
$ ceph osd pool create cephfs_data <pg_num>
$ ceph osd pool create cephfs_metadata <pg_num>
```

例如：

```
ceph osd pool create cephfs_data 256
ceph osd pool create cephfs_metadata 256
```

2.3.3 创建文件系统

语法：

```
ceph fs new <fs_name> <metadata> <data>
```

例如：

```
ceph fs new cephfs cephfs_metadata cephfs_data
```

查看创建的结果：

```
ceph fs ls
ceph mds stat
ceph -s
```

```
[root@master ~]# ceph fs ls
name: cephfs, metadata pool: cephfs_metadata, data pools: [cephfs_data ]
[root@master ~]#
[root@master ~]#
```

```
[root@master ~]#
[root@master ~]# ceph mds stat
cephfs-1/1/1 up  {0=master=up:active}
[root@master ~]#
[root@master ~]# ceph -s
  cluster:
    id:     11b96eff-dc3f-45db-8b94-986e7e84ae2e
    health: HEALTH_OK

  services:
    mon: 3 daemons, quorum master,slave01,slave02
    mgr: master(active)
    mds: cephfs-1/1/1 up  {0=master=up:active}
    osd: 30 osds: 30 up, 30 in

  data:
    pools:   5 pools, 1600 pgs
    objects: 304  objects, 1.0 GiB
    usage:   46 GiB used, 109 TiB / 109 TiB avail
    pgs:     1600 active+clean

[root@master ~]#
```

2.3.4 创建用户

创建 cephfs-client 使用的用户，参考相关章节。
建议重新创建密钥，不用直接使用 client-admin 密钥。

2.3.5 挂载 cephfs—方法 1

Linux 内核 2.6 之后，开始支持 ceph 文件系统，可直接使用

也可使用如下命令查看是否已经支持，经验证，centos6.9 ,内核 2.6.32-696.el6.x86_64，并未支持。

```
modprobe -c|grep  ceph
```

- **直接使用密钥挂载**

客户端不安装任何 ceph 相关的包，因为内核支持。

```
mount  -t  ceph  172.16.68.23:6789,172.16.68.24:6789,172.16.68.25:6789:/  /cephfs  -o
name=k8s,secret=AQBLq75cGex9FBAAFcJtYE9mWYq6LHXLZVv+VA==
```

如果使用标准端口 6789，端口也可以省略

```
mount  -t  ceph  172.16.68.23:6789,172.16.68.24:6789,172.16.68.25:6789:/  /cephfs  -o
name=k8s,secret=AQBLq75cGex9FBAAFcJtYE9mWYq6LHXLZVv+VA==
```

注意 name 和密钥需要和上一步一致。

也可以挂载子目录，生产中建议挂载子目录

```
mount -t ceph 172.16.68.23,172.16.68.24,172.16.68.25:/k8s /cephfs -o name=k8s,secret=AQBLq75cGex9FBAAFcJtYE9mWYq6LHXLZVv+VA==
```

挂载后，可以看到如下挂载信息。

```
[root@node1223 ~]#
[root@node1223 ~]# df -h  /cephfs/
Filesystem                                 Size  Used Avail Use% Mounted on
172.16.68.23,172.16.68.24,172.16.68.25:/k8s   35T   19G   35T   1% /cephfs
[root@node1223 ~]#
[root@node1223 ~]#
[root@node1223 ~]#
```

- **使用密钥文件挂载**

如果使用密钥文件挂载，即 secretfile 选项，需要安装依赖包。

安装 ceph 源

```
yum install https://mirrors.aliyun.com/ceph/rpm-mimic/el7/noarch/ceph-release-1-1.el7.noarch.rpm
```

安装 epel 源

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

安装 ceph-common

```
yum install ceph-common
```

安装完成之后，可以使用 secretfile 选项

```
mount -t ceph 172.16.68.23:6789,172.16.68.24:6789,172.16.68.25:6789:/ /cephfs -o name=cephfs,secretfile=/etc/ceph/client.cephfs.keyring
```

文件中只存密钥即可。

- **开机自动挂载，/etc/fstab 中添加如下内容**

```
172.16.68.23,172.16.68.24,172.16.68.25:/ /cephfs ceph name=cephfs,secret=AQDfhbxc2mnrARAAENHjyLnTvJx+56RkJCU1Iw==,_netdev,noatime 0 0
```

```
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
UUID=ad7704c8-b9e0-484d-9745-26795a994e07 /                       xfs     defaults        0 0
UUID=94a5d398-357d-411d-94f5-85666039e8b8 /boot                   xfs     defaults        0 0
172.16.68.25:6789,172.16.68.24:6789,172.16.68.25:6789:/ /cephfs ceph name=cephfs,secret=AQDfhbxc2mnrAR
AAENHjyLnTvJx+56RkJCU1lw==,_netdev,noatime 0 0
#UUID=496ec6d3-72cf-405d-9538-df31d1e7ce7a swap                    swap    defaults        0 0
~
```

或者使用密钥文件

```
172.16.68.23,172.16.68.24,172.16.68.25:/                    /cephfs                 ceph
name=cephfs,secretfile=/etc/ceph/client.cephfs.secret,_netdev,noatime 0 0
```

```
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
UUID=2e8e01cc-a0ca-4ad9-9718-e6a65c7749e0 /                       xfs     defaults        0 0
UUID=2516aa92-532c-4c6b-b760-8c24d7e5f821 /boot                   xfs     defaults        0 0
UUID=2c629893-fe63-47bd-ae19-749c6a2809fd swap                    swap    defaults        0 0
172.16.68.23,172.16.68.24,172.16.68.25:/ /cephfs ceph name=cephfs,secretfile=/etc/ceph/client.cephfs.secret,_netdev,noatime 0 0
~
~
~
~
```

2.3.6 挂载 cephfs-方法 2

2.3.7 查看谁挂载了 cephfs

```
ceph tell mds.0 client ls|grep client
```

```
[root@master ceph-cluster]#
[root@master ceph-cluster]# ceph tell mds.0 client ls
2019-04-24 16:58:04.772 7fea21ffb700  0 client.127051 ms_handle_reset on 172.16.68.25:6800/642884290
2019-04-24 16:58:04.790 7fea22ffd700  0 client.127172 ms_handle_reset on 172.16.68.25:6800/642884290
[
    {
        "id": 126541,
        "num_leases": 0,
        "num_caps": 9,
        "state": "open",
        "request_load_avg": 0,
        "uptime": 6406.058212,
        "replay_requests": 0,
        "completed_requests": 1,
        "reconnecting": false,
        "inst": "client.126541 172.16.13.23:0/318140826",
        "client_metadata": {
            "features": "0000000000000ff",
            "entity_id": "k8s",
            "hostname": "node1323",
            "kernel_version": "3.10.0-957.el7.x86_64",
            "root": "/"
        }
    }
]
[root@master ceph-cluster]# ceph tell mds.0 client ls|grep client
2019-04-24 16:58:17.258 7f032f7f6700  0 client.127063 ms_handle_reset on 172.16.68.25:6800/642884290
2019-04-24 16:58:17.276 7f03307f8700  0 client.127066 ms_handle_reset on 172.16.68.25:6800/642884290
        "inst": "client.126541 172.16.13.23:0/318140826",
        "client_metadata": {
[root@master ceph-cluster]#
[root@master ceph-cluster]#
```

2.3.8 k8s 使用 cephfs

pv 的大小可以动态改变，如果变小，不能小于对应的 pvc 大小，如下图：

```
[root@master1120 pv_pvc]# kubectl get pv
NAME     CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM        STORAGECLASS   REASON   AGE
mis-pv   15Gi       RWX            Retain           Bound    mis/mypvc                            172m
x2-pv    6Gi        RWX            Retain           Bound    x2/mypvc                             173m
xmc-pv   5Gi        RWX            Retain           Bound    xmc/mypvc                            173m
[root@master1120 pv_pvc]#
[root@master1120 pv_pvc]# kubectl apply -f mis-pv.yml
persistentvolume/mis-pv configured
[root@master1120 pv_pvc]#
[root@master1120 pv_pvc]# kubectl get pv
NAME     CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM        STORAGECLASS   REASON   AGE
mis-pv   10Gi       RWX            Retain           Bound    mis/mypvc                            172m
x2-pv    6Gi        RWX            Retain           Bound    x2/mypvc                             173m
xmc-pv   5Gi        RWX            Retain           Bound    xmc/mypvc                            173m
[root@master1120 pv_pvc]#
```

pvc 不能缩小

```
[root@master1120 pv_pvc]# ls
mis-pvc.yml  mis-pv.yml  nginx-pod.yaml  x2-pvc.yml  x2-pv.yml  xmc-pvc.yml  xmc-pv.yml
[root@master1120 pv_pvc]#
[root@master1120 pv_pvc]# kubectl get pvc -n x2
NAME    STATUS   VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   AGE
mypvc   Bound    x2-pv    6Gi        RWX                           166m
[root@master1120 pv_pvc]#
[root@master1120 pv_pvc]# kubectl apply -f x2-pvc.yml
The PersistentVolumeClaim "mypvc" is invalid: spec.resources.requests.storage: Forbidden: field can not be less than previous value
[root@master1120 pv_pvc]#
[root@master1120 pv_pvc]#
```

pvc 能否增大，取决于方式，只有动态供给的 pvc、且存储类支持动态变化的才能增大

```
[root@master1120 pv_pvc]# kubectl get pvc -n mis
NAME    STATUS   VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   AGE
mypvc   Bound    mis-pv   5Gi        RWX                           132m
[root@master1120 pv_pvc]#
[root@master1120 pv_pvc]# kubectl apply -f mis-pvc.yml
Error from server (Forbidden): error when applying patch:
{"metadata":{"annotations":{"kubectl.kubernetes.io/last-applied-configuration":"{\"apiVersion\":\"v1\",\"kind\":\"PersistentVolumeClaim\",\"metadata\":{\"annotations\":{},\"name
mespace\":\"mis\"},\"spec\":{\"accessModes\":[\"ReadWriteMany\"],\"resources\":{\"requests\":{\"storage\":\"6Gi\"}},\"volumeName\":\"mis-pv\"}}\n"}},"spec":{"resources":{"reques
G"}}}}
to:
Resource: "/v1, Resource=persistentvolumeclaims", GroupVersionKind: "/v1, Kind=PersistentVolumeClaim"
Name: "mypvc", Namespace: "mis"
Object: &{map["apiVersion":"v1" "metadata":map["annotations":map["kubectl.kubernetes.io/last-applied-configuration":"{\"apiVersion\":\"v1\",\"kind\":\"PersistentVolumeClaim\",\"
otations\":{},\"name\":\"mypvc\",\"namespace\":\"mis\"},\"spec\":{\"accessModes\":[\"ReadWriteMany\"],\"resources\":{\"requests\":{\"storage\":\"5Gi\"}},\"volumeName\":\"mis-pv\
etes.io/bind-completed":"yes"] "finalizers":["kubernetes.io/pvc-protection"] "name":"mypvc" "namespace":"mis" "selfLink":"/api/v1/namespaces/mis/persistentvolumeclaims/mypvc" "u
8-11e9-9608-525400a49cb6" "resourceVersion":"370428" "creationTimestamp":"2019-05-06T04:36:12Z"] "spec":map["volumeMode":"Filesystem" "dataSource":<nil> "accessModes":["ReadWrit
s":map["requests":map["storage":"5Gi"]] "volumeName":"mis-pv"] "status":map["accessModes":["ReadWriteMany"] "capacity":map["storage":"5Gi"] "phase":"Bound"] "kind":"PersistentVo
for: "mis-pvc.yml": persistentvolumeclaims "mypvc" is forbidden: only dynamically provisioned pvc can be resized and the storageclass that provisions the pvc must support resize
[root@master1120 pv_pvc]#
```

2.4 测试 pool（官网测试）

创建测试 pool

```
ceph osd pool create pool2 8
```

准备测试文件

```
echo abc > pooltest.txt
```

把文件存入对象 my-object 中，指定使用的 pool

```
rados put my-object pooltest.txt --pool=pool2
```

查看是否存储了该对象

```
rados -p pool2 ls
```

确定对象的位置

```
ceph osd map pool2 my-boject
```

```
[root@localhost rbd2]#
[root@localhost rbd2]# ceph osd map mypool foo2
osdmap e2001 pool 'mypool' (9) object 'foo2' -> pg 9.327131b3 (9.33) -> up ([17,0,27], p17) acting ([17,0,27], p17)
[root@localhost rbd2]#
[root@localhost rbd2]# ceph osd map mypool foo1
osdmap e2001 pool 'mypool' (9) object 'foo1' -> pg 9.be9754b3 (9.33) -> up ([17,0,27], p17) acting ([17,0,27], p17)
[root@localhost rbd2]# ceph osd map mypool foo
osdmap e2001 pool 'mypool' (9) object 'foo' -> pg 9.7fc1f406 (9.6) -> up ([16,24,8], p16) acting ([16,24,8], p16)
[root@localhost rbd2]#
[root@localhost rbd2]#
[root@localhost rbd2]#
```

测试完成，删除测试对象

```
rados rm my-object --pool=pool2
```

删除 pool (强制删除，会删除 pool 下面的所有数据)

```
ceph osd pool rm pool2 pool2 --yes-i-really-really-mean-it
```

第三章 集群管理

3.1 服务管理

集群中的节点会有以下服务：已经加入的模块，是开机自启

```
[root@master ceph-cluster]#
[root@master ceph-cluster]# systemctl list-unit-files|grep ceph
ceph-disk@.service                            static
ceph-mds@.service                             enabled
ceph-mgr@.service                             enabled
ceph-mon@.service                             enabled
ceph-osd@.service                             enabled-runtime
ceph-radosgw@.service                         disabled
ceph-volume@.service                          enabled
ceph-mds.target                               enabled
ceph-mgr.target                               enabled
ceph-mon.target                               enabled
ceph-osd.target                               enabled
ceph-radosgw.target                           enabled
ceph.target                                   enabled
[root@master ceph-cluster]#
```

ceph-mds: 开启文件存储时才使用，非必须

ceph-mon: 监视器，必须

ceph-osd：osd，必须的

ceph-radosgw: 使用对象存储的才使用，非必须

ceph-mgr: 使用 web 管理时才开启，非必须

ceph.target： 一键管理所有，可选，默认是启用的

ceph 的 rbd 客户端会有以下服务，否使非开机自启

```
[root@localhost ~]# systemctl list-unit-files|grep ceph
ceph-disk@.service                            static
ceph-mds@.service                             disabled
ceph-mgr@.service                             disabled
ceph-mon@.service                             disabled
ceph-osd@.service                             disabled
ceph-radosgw@.service                         disabled
ceph-volume@.service                          disabled
ceph-mds.target                               enabled
ceph-mgr.target                               enabled
ceph-mon.target                               enabled
ceph-osd.target                               enabled
ceph-radosgw.target                           enabled
ceph.target                                   enabled
[root@localhost ~]#
```

3.2 配置

3.2.1 查看配置：

```
ceph daemon {daemon-type}.{id} config show | less
```

daemon-type 可以是 osd、mgr、mon、mds 等

id 就是后面的 0 、1、 2 ， a、b、c 等

例如：

```
ceph daemon osd.0 config show | less
```

3.2.2 推送配置文件

使用如下命令

```
ceph-deploy admin ceph-client-1
```

如果更新了配置文件，需要更新远程的配置文件，使用如下命令

```
ceph-deploy config push {host-name [host-name] ...}
```

3.3 排错

3.2.3 时钟偏移

使用 ceph -s 查看到有如下错误

health: HEALTH_ERR

Module 'dashboard' has failed: IOError("Port 8080 not free on '192.168.68.23'",)

clock skew detected on mon.slave01, mon.slave02

```
[root@master ceph-cluster]# ceph -s
  cluster:
    id:     11b96eff-dc3f-45db-8b94-986e7e84ae2e
    health: HEALTH_ERR
            Module 'dashboard' has failed: IOError("Port 8080 not free on '192.168.68.23'",)
            clock skew detected on mon.slave01, mon.slave02

  services:
    mon: 3 daemons, quorum master,slave01,slave02
    mgr: slave02(active), standbys: slave01, master
    mds: cephfs-1/1/1 up  {0=master=up:active}
    osd: 30 osds: 30 up, 30 in

  data:
    pools:   5 pools, 1600 pgs
    objects: 22  objects, 3.2 KiB
    usage:   32 GiB used, 109 TiB / 109 TiB avail
    pgs:     1600 active+clean

[root@master ceph-cluster]#
[root@master ceph-cluster]# date
Wed Apr 17 14:58:27 CST 2019
```

产生了时钟偏移，从以下两点着手排查

1、 monitor 节点的 ntp 服务是否正常

2、 monitor 的时钟偏差是否设置的过小

处理方法：停止 mon 节点的 ntp 服务，同步时间

```
ntpdate    202.112.29.82
```

重启 mon 服务

```
systemctl restart ceph-mon.target
```

3.2.4 dashboard 错误

重启后这个错误没了，但还有另一个错误

```
[root@master ceph-cluster]#
[root@master ceph-cluster]# ceph -s

  cluster:
    id:     11b96eff-dc3f-45db-8b94-986e7e84ae2e
    health: HEALTH_ERR
            Module 'dashboard' has failed: IOError("Port 8080 not free on '192.168.68.23'",)

  services:
    mon: 3 daemons, quorum master,slave01,slave02
    mgr: slave02(active), standbys: slave01, master
    mds: cephfs-1/1/1 up  {0=master=up:active}
    osd: 30 osds: 30 up, 30 in

  data:
    pools:   5 pools, 1600 pgs
    objects: 22  objects, 3.2 KiB
    usage:   32 GiB used, 109 TiB / 109 TiB avail
    pgs:     1600 active+clean

[root@master ceph-cluster]#
[root@master ceph-cluster]# ceph mgr module disable dashboard
[root@master ceph-cluster]#
[root@master ceph-cluster]#
[root@master ceph-cluster]# ceph -s
  cluster:
    id:     11b96eff-dc3f-45db-8b94-986e7e84ae2e
    health: HEALTH_OK

  services:
    mon: 3 daemons, quorum master,slave01,slave02
    mgr: slave02(active), standbys: master, slave01
    mds: cephfs-1/1/1 up  {0=master=up:active}
    osd: 30 osds: 30 up, 30 in

  data:
    pools:   5 pools, 1600 pgs
    objects: 22  objects, 3.2 KiB
    usage:   32 GiB used, 109 TiB / 109 TiB avail
    pgs:     1600 active+clean

[root@master ceph-cluster]#
```

执行如下卸载后再加载，即可恢复

```
ceph config-key set mgr/dashboard/server_addr 0.0.0.0

ceph config set mgr mgr/dashboard/server_addr 0.0.0.0

ceph mgr module disable dashboard

ceph mgr module enable dashboard
```

第四章 ceph shell

2.1 常用操作

查看状态

```
ceph -s
```

查看实时运行状态

```
ceph -w
```

查看健康状况

```
ceph health
```

查看健康状态的细节

```
ceph health detail
```

查看空间使用

```
ceph df
```

查看 osd 空间占用

```
ceph osd df
```

查看 osd 分布

```
ceph osd tree
```

其他常用命令参考:
https://blog.csdn.net/zhao897426182/article/details/78837422

2.2 pool 操作

pool 是 ceph 存储数据时的逻辑分区，它起到 namespace 的作用。其他分布式存储系统，比如 Mogilefs、Couchbase、Swift 都有 pool 的概念，只是叫法不同。每个 pool 包含一定数量的 PG，PG 里的对象被映射到不同的 OSD 上，因此 pool 是分布到整个集群的。

除了隔离数据，我们也可以分别对不同的 POOL 设置不同的优化策略，比如副本数、数据清洗次数、数据块及对象大小等。

2.1.1 创建 pool

64 表示 64 个 pg_num

```
ceph osd pool create mypool 64
```

设置 pool 副本数量，默认 3（可能和 node 数量有关，待验证）

```
ceph osd pool set mypool size 2
```

```
[root@master ~]#
[root@master ~]# ceph osd map pool2 my-boject
osdmap e1987 pool 'pool2' (7) object 'my-boject' -> pg 7.ba23d3d0 (7.10) -> up ([3,28,18], p3) acting ([3,28,18], p3)
[root@master ~]#
[root@master ~]#
[root@master ~]# ceph osd pool set pool2 size 2
set pool 7 size to 2
[root@master ~]#
[root@master ~]# ceph osd map pool2 my-boject
osdmap e1989 pool 'pool2' (7) object 'my-boject' -> pg 7.ba23d3d0 (7.10) -> up ([3,28], p3) acting ([3,28], p3)
[root@master ~]#
[root@master ~]#
```

通常在创建 pool 之前，需要覆盖默认的 pg_num，官方推荐：

- 若少于 5 个 OSD， 设置 pg_num 为 128。
- 5~10 个 OSD，设置 pg_num 为 512。
- 10~50 个 OSD，设置 pg_num 为 4096。
- 超过 50 个 OSD，可以参考 pgcalc 计算。

2.1.2 修改 pool 的类型

```
ceph osd pool application enable mypool2 rbd
```

当前常见池使用类型有三种
CephFS uses the application name cephfs
RBD uses the application name rbd
RGW uses the application name rgw

2.1.3 查看 pool

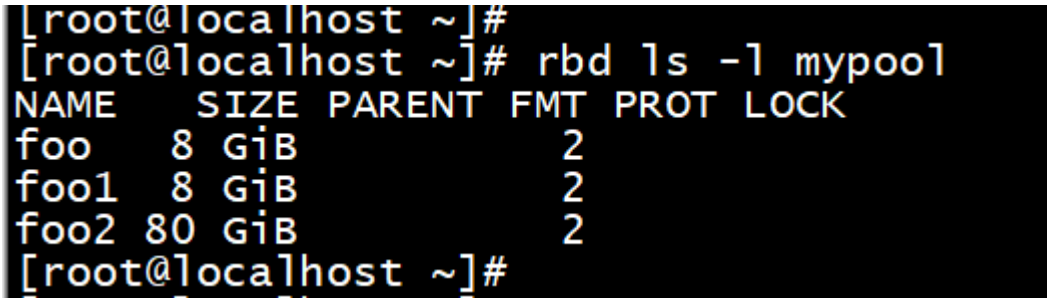
```
rados lspools
ceph osd lspools
ceph osd dump |grep pool
```

2.1.4 查看 pool 中的对象

```
rados -p pool2 ls
```

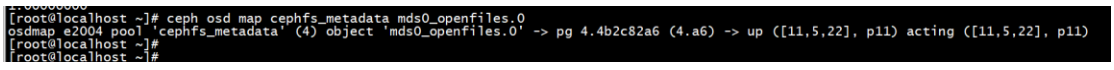
2.1.5 查看 pool 中的 images

```
rbd ls -l mypool
```



2.1.6 查看对象分布

```
ceph osd map cephfs_metadata mds0_openfiles.0
```



2.1.7 删除 pool

```
ceph osd pool rm mypool
```

即使删除了 pool，pool 中的 image 还是存在的。

2.1.8 设置 pool 限额

如下配置最大 100 个对象，容量最大是 100G

```
ceph osd pool set-quota mypool max_objects 100
ceph osd pool set-quota mypool max_bytes $((100 * 1024 * 1024 * 1024))
```

2.1.9 重命名 pool

```
ceph osd pool rename mypool pool2
```

2.1.10 查看 pool 状态

```
rados df
```

2.1.11 创建快照

```
ceph osd pool mksnap pool2 pool2_snap
```

2.1.12 查看快照

```
ceph osd dump |grep pool
```

2.1.13 删除快照

```
ceph osd pool rmsnap pool2 pool2_snap
```

2.3 image 操作

2.3.1 创建 images

```
rbd create mypool/foo1 --size 4096 --image-feature layering
```

--object-size *B/K/M 指定 object 大小，默认是 4M，可取 4kB-32MB,必须是 2 的整数次幂。

提醒：设置该参数为 4K 或者 8K，可以改善小文件（4K）的读写性能。

2.3.2 查看 image

```
rbd ls -l mypool
rbd list mypool
```

2.3.3 查看 image 详细信息

```
rbd info mypool/foo
rbd info foo1 -p mypool
```

2.3.4 查看 image 包含哪些 obj

根据 info 出来的 block_name_prefix 进行查询

```
rbd info node1323
```

```
[root@slave01 ~]# rbd info node1323
rbd image 'node1323':
        size 200 GiB in 51200 objects
        order 22 (4 MiB objects)
        id: 1ec936b8b4567
        block_name_prefix: rbd_data.1ec936b8b4567
        format: 2
        features: layering
        op_features:
        flags:
        create_timestamp: Tue Apr 23 14:23:48 2019
[root@slave01 ~]#
```

```
rados -p rbd ls | grep rbd_data.1ec936b8b4567
```

```
[root@slave01 ~]# rados -p rbd ls |grep rbd_data.1ec936b8b4567|more
rbd_data.1ec936b8b4567.000000000000013c
rbd_data.1ec936b8b4567.0000000000000c80
rbd_data.1ec936b8b4567.00000000000000f4
rbd_data.1ec936b8b4567.0000000000000099
rbd_data.1ec936b8b4567.00000000000001b2
rbd_data.1ec936b8b4567.000000000000004d
rbd_data.1ec936b8b4567.00000000000000b8
rbd_data.1ec936b8b4567.000000000000013e
rbd_data.1ec936b8b4567.0000000000000035
rbd_data.1ec936b8b4567.0000000000000052
```

2.3.5 查看 image 已使用空间大小

方法 1)

```
rbd info node1323
rados -p rbd ls | grep rbd_data.1ec936b8b4567|wc -l
```

```
[root@slave01 ~]# rbd info node1323
rbd image 'node1323':
        size 200 GiB in 51200 objects
        order 22 (4 MiB objects)
        id: 1ec936b8b4567
        block_name_prefix: rbd_data.1ec936b8b4567
        format: 2
        features: layering
        op_features:
        flags:
        create_timestamp: Tue Apr 23 14:23:48 2019
[root@slave01 ~]#
[root@slave01 ~]#
[root@slave01 ~]# rados -p rbd ls | grep rbd_data.1ec936b8b4567|wc -l
556
[root@slave01 ~]#
[root@slave01 ~]#
[root@slave01 ~]#
```

发现包含 556 个块

改 image 已使用 2081m，约等于 556 个 obj*4m

方法二:
挂载后直接查看大小

```
[root@node1323 mnt]# df -m /mnt/
Filesystem     1M-blocks  Used Available Use% Mounted on
/dev/rbd0         204700  2081    202620   2% /mnt
[root@node1323 mnt]#
```

方法三:
使用如下命令，计算出是 2152M

```
rbd diff node1323| awk '{ SUM += $2 } END { print SUM/1024/1024 " MB" }'
```

```
[root@slave01 ~]# rbd diff node1323| awk '{ SUM += $2 } END { print SUM/1024/1024 " MB" }'
2152.44 MB
[root@slave01 ~]#
[root@slave01 ~]#
```

以上三种方法得出的结果类似

2.3.6 查看 obj 的真实位置

```
ceph osd map rbd rbd_data.1ec936b8b4567.00000000000000ef
```

```
[root@slave01 ~]#
[root@slave01 ~]# ceph osd map rbd rbd_data.1ec936b8b4567.00000000000000ef
osdmap e3356 pool 'rbd' (6) object 'rbd_data.1ec936b8b4567.00000000000000ef' -> pg 6.4dc98218 (6.218) -> up ([15,21,5], p15) acting ([15,21,5], p15)
[root@slave01 ~]#
[root@slave01 ~]#
[root@slave01 ~]#
```

如图表示，rbd_data.1ec936b8b4567.00000000000000ef 这个对象位于 6.218 这个 pg 上，而且处于 osd.15、osd.21、osd.5 这三个 osd 上，其中 osd.15 是主，另外两个是副本。

2.3.7 删除 image

```
rbd rm mypool/foo
```

无法删除已经映射的 image，必须先卸载、取消映射，再删除

```
[root@localhost ~]#
[root@localhost ~]# rbd showmapped
id pool   image snap device
1  mypool foo1  -    /dev/rbd1
2  mypool foo2  -    /dev/rbd2
3  mypool foo14 -    /dev/rbd3
4  mypool foo15 -    /dev/rbd4
5  mypool foo16 -    /dev/rbd5
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# rbd rm mypool/foo1
2019-04-17 14:13:02.291 7f3889a61700 -1 librbd::image::RemoveRequ
Removing image: 0% complete...failed.
rbd: error: image still has watchers
This means the image is still open or the client using it crashed
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# rbd unmap mypool/foo1
rbd: sysfs write failed
rbd: unmap failed: (16) Device or resource busy
[root@localhost ~]#
[root@localhost ~]# umount /rbd1/
[root@localhost ~]#
[root@localhost ~]# rbd unmap mypool/foo1
[root@localhost ~]#
[root@localhost ~]# rbd rm mypool/foo1
Removing image: 100% complete...done.
[root@localhost ~]#
[root@localhost ~]#
```

2.4 排错

2.5.1 副本数不够

报错如下:

```
[root@master ceph-cluster]# ceph -s

  cluster:
    id:     11b96eff-dc3f-45db-8b94-986e7e84ae2e
    health: HEALTH_WARN
            Reduced data availability: 7 pgs inactive
            Degraded data redundancy: 7 pgs undersized

  services:
    mon: 3 daemons, quorum master,slave01,slave02
    mgr: master(active), standbys: slave02, slave01
    mds: myfs-1/1/1 up  {0=slave02=up:active}, 2 up:standby
    osd: 30 osds: 30 up, 30 in; 7 remapped pgs

  data:
    pools:   3 pools, 1088 pgs
    objects: 4.23 k objects, 16 GiB
    usage:   5.5 TiB used, 109 TiB / 115 TiB avail
    pgs:     0.643% pgs not active
             1081 active+clean
             7    activating+undersized+remapped

  io:
    client:   36 KiB/s rd, 44 op/s rd, 0 op/s wr
```

查看错误的详细信息:

ceph health detail

```
[root@master osd]#
[root@master osd]# ceph health detail
HEALTH_WARN Reduced data availability: 7 pgs inactive; Degraded data redundancy: 7 pgs undersized
PG_AVAILABILITY Reduced data availability: 7 pgs inactive
    pg 6.a8 is stuck inactive for 1658.339939, current state activating+undersized+remapped, last acting [14,23]
    pg 6.194 is stuck inactive for 1658.325703, current state activating+undersized+remapped, last acting [13,29]
    pg 6.1c9 is stuck inactive for 1658.350127, current state activating+undersized+remapped, last acting [14,20]
    pg 6.280 is stuck inactive for 1613.501272, current state activating+undersized+remapped, last acting [24,10]
    pg 6.2ba is stuck inactive for 1658.329735, current state activating+undersized+remapped, last acting [15,23]
    pg 6.34d is stuck inactive for 1658.337745, current state activating+undersized+remapped, last acting [14,26]
    pg 6.376 is stuck inactive for 1658.324012, current state activating+undersized+remapped, last acting [14,27]
PG_DEGRADED Degraded data redundancy: 7 pgs undersized
    pg 6.a8 is stuck undersized for 1656.340093, current state activating+undersized+remapped, last acting [14,23]
    pg 6.194 is stuck undersized for 1656.329171, current state activating+undersized+remapped, last acting [13,29]
    pg 6.1c9 is stuck undersized for 1613.324218, current state activating+undersized+remapped, last acting [14,20]
    pg 6.280 is stuck undersized for 1612.312691, current state activating+undersized+remapped, last acting [24,10]
    pg 6.2ba is stuck undersized for 1656.332990, current state activating+undersized+remapped, last acting [15,23]
    pg 6.34d is stuck undersized for 1656.345586, current state activating+undersized+remapped, last acting [14,26]
    pg 6.376 is stuck undersized for 1656.343917, current state activating+undersized+remapped, last acting [14,27]
[root@master osd]#
[root@master osd]#
[root@master osd]#
```

逐个重启对应的 osd，重启一个，观察，再重启另一个。

```
[root@slave01 ~]#
[root@slave01 ~]# systemctl restart ceph-osd@14
[root@slave01 ~]# systemctl restart ceph-osd@10
[root@slave01 ~]# systemctl restart ceph-osd@13
[root@slave01 ~]# systemctl restart ceph-osd@15
[root@slave01 ~]#
```

重启后，恢复正常

第五章 ssd 加速

5.1 ssd 在 ceph 中一般有几种使用方式

http://stor.51cto.com/art/201711/559120.htm

5.2 选型

我这里选择使用 bluestore+wal+db 的形式。

5.2.1 准备 ssd 分区

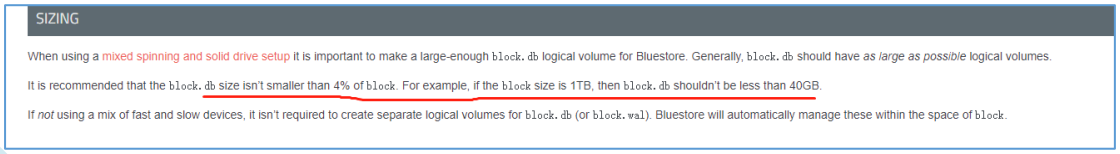
我这里每台服务器有 10*4T 的 SATA 盘，分别是/dev/sdc--/dev/sdl，/devsda 是系统盘，/dev/sdb 是一个 4T 的 ssd 盘，将 sdb 用作缓存盘，准备 20 个分区，如下图

```
Disk /dev/sdb: 3840.2 GB, 3840204079104 bytes, 7500398592 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 262144 bytes / 262144 bytes
Disk label type: gpt
Disk identifier: E18F99D4-7172-45DA-A6C6-469050868388


#         Start          End    Size  Type            Name
 1          4096      3905535    1.9G  Microsoft basic wal-1
 2       3911680      7813119    1.9G  Microsoft basic wal-2
 3       7815168     11718655    1.9G  Microsoft basic wal-3
 4      11720704     15624191    1.9G  Microsoft basic wal-4
 5      15626240     19531775    1.9G  Microsoft basic wal-5
 6      19533824     23437311    1.9G  Microsoft basic wal-6
 7      23439360     27342847    1.9G  Microsoft basic wal-7
 8      27344896     31250431    1.9G  Microsoft basic wal-8
 9      31252480     35155967    1.9G  Microsoft basic wal-9
10      35160064     39063551    1.9G  Microsoft basic wal-10
11      39063552    429688831  186.3G  Microsoft basic db-1
12     429690880    820314111  186.3G  Microsoft basic db-2
13     820316160   1210939391  186.3G  Microsoft basic db-3
14    1210941440   1601564671  186.3G  Microsoft basic db-4
15    1601566720   1992189951  186.3G  Microsoft basic db-5
16    1992192000   2382815231  186.3G  Microsoft basic db-6
17    2382817280   2773440511  186.3G  Microsoft basic db-7
18    2773442560   3164065791  186.3G  Microsoft basic db-8
19    3164067840   3554691071  186.3G  Microsoft basic db-9
20    3554693120   3945316351  186.3G  Microsoft basic db-10
```

分区命令使用 parted /dev/sdb
分区中 1-10 是 2G，用作 wal
11-20 是 200G，用作 db。

根据官网介绍，db 容量应该大于 osd 的 4%
http://docs.ceph.com/docs/mimic/rados/configuration/bluestore-config-ref/?tdsourcetag=s_pctim_aiomsg



SIZING

When using a mixed spinning and solid drive setup it is important to make a large-enough `block.db` logical volume for Bluestore. Generally, `block.db` should have *as large as possible* logical volumes.

It is recommended that the `block.db` size isn't smaller than 4% of `block`. For example, if the `block` size is 1TB, then `block.db` shouldn't be less than 40GB.

If *not* using a mix of fast and slow devices, it isn't required to create separate logical volumes for `block.db` (or `block.wal`). Bluestore will automatically manage these within the space of `block`.

wal 大小，我这里设置了 2G，不需要太大。

parted 分区命令，使用方法参考其他地方。

5.2.2 添加 bulestore 的 osd

使用以下命令可以获取帮助

ceph-deplay osd --help

```
[root@master ceph-cluster]#
[root@master ceph-cluster]# ceph-deploy osd --help
usage: ceph-deploy osd [-h] {list,create} ...

Create OSDs from a data disk on a remote host:

    ceph-deploy osd create {node} --data /path/to/device

For bluestore, optional devices can be used::

    ceph-deploy osd create {node} --data /path/to/data --block-db /path/to/db-device
    ceph-deploy osd create {node} --data /path/to/data --block-wal /path/to/wal-device
    ceph-deploy osd create {node} --data /path/to/data --block-db /path/to/db-device --block-wal /path/to/wal-device

For filestore, the journal must be specified, as well as the objectstore::

    ceph-deploy osd create {node} --filestore --data /path/to/data --journal /path/to/journal

For data devices, it can be an existing logical volume in the format of:
vg/lv, or a device. For other OSD components like wal, db, and journal, it
can be logical volume (in vg/lv format) or it must be a GPT partition.

positional arguments:
  {list,create}
    list         List OSD info from remote host(s)
    create       Create new Ceph OSD daemon by preparing and activating a
                 device

optional arguments:
  -h, --help     show this help message and exit
```

传递配置到远程机器

```
ceph-deploy admin ceph-1 ceph-2 ceph-3
```

在 admin 节点上，切换到工作目录

```
ceph-deploy osd create master --bluestore --data /dev/sdl --block-db /dev/sdb20 --block-wal /dev/sdb10
```

其他盘区对应方式如下

osd-device	db-device	wal-device
/dev/sdc	/dev/sdb11	/dev/sdb1
/dev/sdd	/dev/sdb12	/dev/sdb2
/dev/sde	/dev/sdb13	/dev/sdb3
/dev/sdf	/dev/sdb14	/dev/sdb4
/dev/sdg	/dev/sdb15	/dev/sdb5
/dev/sdh	/dev/sdb16	/dev/sdb6
/dev/sdi	/dev/sdb17	/dev/sdb7
/dev/sdj	/dev/sdb18	/dev/sdb8

/dev/sdk	/dev/sdb19	/dev/sdb9
/dev/sdl	/dev/sdb20	/dev/sdb10

```
ceph-deploy osd create ceph6826 --bluestore --data /dev/sdc --block-db /dev/sdb11 --block-wal /dev/sdb1
ceph-deploy osd create ceph6826 --bluestore --data /dev/sdd --block-db /dev/sdb12 --block-wal /dev/sdb2
ceph-deploy osd create ceph6826 --bluestore --data /dev/sde --block-db /dev/sdb13 --block-wal /dev/sdb3
ceph-deploy osd create ceph6826 --bluestore --data /dev/sdf --block-db /dev/sdb14 --block-wal /dev/sdb4
ceph-deploy osd create ceph6826 --bluestore --data /dev/sdg --block-db /dev/sdb15 --block-wal /dev/sdb5
ceph-deploy osd create ceph6826 --bluestore --data /dev/sdh --block-db /dev/sdb16 --block-wal /dev/sdb6
ceph-deploy osd create ceph6826 --bluestore --data /dev/sdi --block-db /dev/sdb17 --block-wal /dev/sdb7
ceph-deploy osd create ceph6826 --bluestore --data /dev/sdj --block-db /dev/sdb18 --block-wal /dev/sdb8
ceph-deploy osd create ceph6826 --bluestore --data /dev/sdk --block-db /dev/sdb19 --block-wal /dev/sdb9
ceph-deploy osd create ceph6826 --bluestore --data /dev/sdl --block-db /dev/sdb20 --block-wal /dev/sdb10
```