

NSD CLUSTER DAY04

1. [案例1：实验环境](#)
2. [案例2：部署ceph集群](#)
3. [案例3：创建Ceph块存储](#)

1 案例1：实验环境

1.1 问题

准备四台KVM虚拟机，其三台作为存储集群节点，一台安装为客户端，实现如下功能：

- 创建1台客户端虚拟机
- 创建3台存储集群虚拟机
- 配置主机名、IP地址、YUM源
- 修改所有主机的主机名
- 配置无密码SSH连接
- 配置NTP时间同步
- 创建虚拟机磁盘

1.2 方案

使用4台虚拟机，1台客户端、3台存储集群服务器，拓扑结构如图-1所示。



图-1

所有主机的主机名及对应的IP地址如表-1所示。

表 - 1 主机名称及对应IP地址表

主机名称	值
client	192.168.4.10/24
node1	192.168.4.11/24
node2	192.168.4.12/24
node3	192.168.4.13/24

1.3 步骤

实现此案例需要按照如下步骤进行。

[Top](#)

步骤一：安装前准备

1) 物理机为所有节点配置yum源，注意所有的虚拟主机均需要挂载安装光盘。

```
01. [ root@root9pc01 ~] # yum -y install vsftpd
02. [ root@root9pc01 ~] # mkdir /var/ftp/ceph
03. [ root@root9pc01 ~] # mount -o loop \
04. rhcs2.0-rhosp9-20161113-x86_64.iso /var/ftp/ceph
05. [ root@root9pc01 ~] # systemctl restart vsftpd
```

2) 修改所有节点yum配置 (以node1为例)

```
01. [ root@node1 ~] # cat /etc/yum.repos.d/ceph.repo
02. [mon]
03. name=mon
04. baseurl=ftp://192.168.4.254/ceph/rhceph-2.0-rhel-7-x86_64/MON
05. gpgcheck=0
06. [osd]
07. name=osd
08. baseurl=ftp://192.168.4.254/ceph/rhceph-2.0-rhel-7-x86_64/OSD
09. gpgcheck=0
10. [tools]
11. name=tools
12. baseurl=ftp://192.168.4.254/ceph/rhceph-2.0-rhel-7-x86_64/Tools
13. gpgcheck=0
```

3) 修改/etc/hosts并同步到所有主机。

```
01. [ root@node1 ~] # cat /etc/hosts
02. ... ..
03. 192.168.4.10    client
04. 192.168.4.11    node1
05. 192.168.4.12    node2
06. 192.168.4.13    node3
07. [ root@node1 ~] # for i in 10 11 12 13
08. > do
09. > scp /etc/hosts 192.168.2.$i:/etc/
10. > done
```

[Top](#)

3) 配置无密码连接。

```

01. [ root@node1 ~] # ssh-keygen -f /root/.ssh/id_rsa -N ""
02. [ root@node1 ~] # for i in 10 11 12 13
03. > do
04. > ssh-copy-id 192.168.4.$i
05. > done

```

步骤二：配置NTP时间同步

1) 创建NTP服务器。

```

01. [ root@client ~] # yum -y install chrony
02. [ root@client ~] # cat /etc/chrony.conf
03. server 0.centos.pool.ntp.org iburst
04. allow 192.168.4.0/24
05. local stratum 10
06. [ root@client ~] # systemctl restart chronyd

```

2) 其他所有节点与NTP服务器同步时间（以node1为例）。

```

01. [ root@node1 ~] # cat /etc/chrony.conf
02. server 192.168.4.10 iburst
03. [ root@node1 ~] # systemctl restart chronyd

```

步骤三：准备存储磁盘

1) 物理机上为每个虚拟机准备3块磁盘。（可以使用命令，也可以使用图形直接添加）

```

01. [ root@root9pc01 ~] # cd /var/lib/libvirt/images
02. [ root@root9pc01 ~] # qemu-img create -f qcow2 node1-vdb.vol 10G
03. [ root@root9pc01 ~] # qemu-img create -f qcow2 node1-vdc.vol 10G
04. [ root@root9pc01 ~] # qemu-img create -f qcow2 node1-vdd.vol 10G
05. [ root@root9pc01 ~] # qemu-img create -f qcow2 node2-vdb.vol 10G
06. [ root@root9pc01 ~] # qemu-img create -f qcow2 node2-vdc.vol 10G
07. [ root@root9pc01 ~] # qemu-img create -f qcow2 node2-vdd.vol 10G
08. [ root@root9pc01 ~] # qemu-img create -f qcow2 node3-vdb.vol 10G
09. [ root@root9pc01 ~] # qemu-img create -f qcow2 node3-vdc.vol 10G
10. [ root@root9pc01 ~] # qemu-img create -f qcow2 node3-vdd.vol 10G

```

[Top](#)

2) 使用virt-manager为虚拟机添加磁盘。

```
01. [root@root9pc01 ~] # virt-manager
```

2 案例2：部署ceph集群

2.1 问题

沿用练习一，部署Ceph集群服务器，实现以下目标：

- 安装部署工具ceph-deploy
- 创建ceph集群
- 准备日志磁盘分区
- 创建OSD存储空间
- 查看ceph状态，验证

2.2 步骤

实现此案例需要按照如下步骤进行。

步骤一：部署软件

1) 在node1安装部署工具，学习工具的语法格式。

```
01. [root@node1 ~] # yum -y install ceph-deploy
02. [root@node1 ~] # ceph-deploy --help
```

2) 创建目录

```
01. [root@node1 ~] # mkdir ceph-cluster
02. [root@node1 ~] # cd ceph-cluster/
```

步骤二：部署Ceph集群

1) 创建Ceph集群配置。

```
01. [root@node1 ceph-cluster] # ceph-deploy new node1 node2 node3
```

2) 给所有节点安装软件包。

[Top](#)

```
01 [root@node1 ceph-cluster]# ceph-deploy install node1 node2 node3
```

3) 初始化所有节点的mon服务（主机名解析必须对）

```
01 [root@node1 ceph-cluster]# ceph-deploy mon create-initial
```

步骤三：创建OSD

1) 准备磁盘分区

```
01 [root@node1 ~]# parted /dev/vdb mklabel gpt
02 [root@node1 ~]# parted /dev/vdb mkpart primary 1M 50%
03 [root@node1 ~]# parted /dev/vdb mkpart primary 50% 100%
04 [root@node1 ~]# chown ceph.ceph /dev/vdb1
05 [root@node1 ~]# chown ceph.ceph /dev/vdb2
06 //这两个分区用来做存储服务器的日志journal盘
```

2) 初始化清空磁盘数据（仅node1操作即可）

```
01 [root@node1 ~]# ceph-deploy disk zap node1:vdc node1:vdd
02 [root@node1 ~]# ceph-deploy disk zap node2:vdc node2:vdd
03 [root@node1 ~]# ceph-deploy disk zap node3:vdc node3:vdd
```

3) 创建OSD存储空间（仅node1操作即可）

```
01 [root@node1 ~]# ceph-deploy osd create node1:vdc:/dev/vdb1 node1:vdd:/dev/vdb2
02 //创建osd存储设备，vdc为集群提供存储空间，vdb1提供JOURNAL日志，
03 //一个存储设备对应一个日志设备，日志需要SSD，不需要很大
04 [root@node1 ~]# ceph-deploy osd create node2:vdc:/dev/vdb1 node2:vdd:/dev/vdb2
05 [root@node1 ~]# ceph-deploy osd create node3:vdc:/dev/vdb1 node3:vdd:/dev/vdb2
06
```

步骤四：验证测试

[Top](#)

1) 查看集群状态

```
01. [root@node1 ~]# ceph -s
```

3 案例3：创建Ceph块存储

3.1 问题

沿用练习一，使用Ceph集群的块存储功能，实现以下目标：

- 创建块存储镜像
- 客户端映射镜像
- 创建镜像快照
- 使用快照还原数据
- 使用快照克隆镜像
- 删除快照与镜像

3.2 步骤

实现此案例需要按照如下步骤进行。

步骤一：创建镜像

1) 查看存储池。

```
01. [root@node1 ~]# ceph osd lspools
02. 0 rbd,
```

2) 创建镜像、查看镜像

```
01. [root@node1 ~]# rbd create demo- image -- image- feature layering -- size 10G
02. [root@node1 ~]# rbd create rbd/image -- image- feature layering -- size 10G
03. [root@node1 ~]# rbd list
04. [root@node1 ~]# rbd info demo- image
05. rbd image 'demo- image':
06.     size 10240 MB in 2560 objects
07.     order 22 ( 4096 kB objects)
08.     block_name_prefix: rbd_data.d3aa2ae8944a
09.     format: 2
10.     features: layering
```

步骤二：动态调整

1) 缩小容量

[Top](#)

```
01. [ root@node1 ~] # rbd resize -- size 7G image -- allow- shrink
02. [ root@node1 ~] # rbd info image
```

2) 扩容容量

```
01. [ root@node1 ~] # rbd resize -- size 15G image
02. [ root@node1 ~] # rbd info image
```

步骤三：通过KRBD访问

1) 集群内将镜像映射为本地磁盘

```
01. [ root@node1 ~] # rbd map demo- image
02. /dev /rbd0
03. [ root@node1 ~] # lsblk
04. ... ...
05. rbd0      251:0    0  10G  0 disk
06. [ root@node1 ~] # mkfs.xfs /dev /rbd0
07. [ root@node1 ~] # mount /dev /rbd0 /mnt
```

2) 客户端通过KRBD访问

```
01. #客户端需要安装ceph- common软件包
02. #拷贝配置文件（否则不知道集群在哪）
03. #拷贝连接密钥（否则无连接权限）
04. [ root@client ~] # yum- y install ceph- common
05. [ root@client ~] # scp 192.168.4.11: /etc/ceph/ceph.conf /etc/ceph/
06. [ root@client ~] # scp 192.168.4.11: /etc/ceph/ceph.client.admin.key ring \
07. /etc/ceph/
08. [ root@client ~] # rbd map image
09. [ root@client ~] # lsblk
10. [ root@client ~] # rbd showmapped
11. id pool image snap device
12. 0 rbd image - /dev /rbd0
```

[Top](#)

3) 客户端格式化、挂载分区

```
01. [root@client ~] # mkfs.xfs /dev/rbd0
02. [root@client ~] # mount /dev/rbd0 /mnt/
03. [root@client ~] # echo "test" > /mnt/test.txt
```

步骤四：创建镜像快照

1) 查看镜像快照

```
01. [root@node1 ~] # rbd snap ls image
```

2) 创建镜像快照

```
01. [root@node1 ~] # rbd snap create image -- snap image- snap1
02. [root@node1 ~] # rbd snap ls image
03.  SNAPID NAME      SIZE
04.      4 image- snap1 15360 MB
```

3) 删除客户端写入的测试文件

```
01. [root@client ~] # rm -rf /mnt/test.txt
```

4) 还原快照

```
01. [root@node1 ~] # rbd snap rollback image -- snap image- snap1
02. #客户端重新挂载分区
03. [root@client ~] # umount /mnt
04. [root@client ~] # mount /dev/rbd0 /mnt/
05. [root@client ~] # ls /mnt
```

步骤四：创建快照克隆

1) 克隆快照

```
01. [root@node1 ~] # rbd snap protect image -- snap image- snap1
02. [root@node1 ~] # rbd snap rm image -- snap image- snap1 //会失败
```

[Top](#)


```
03. [root@node1 ~]# rbd clone \
04. image -- snap image- snap1 image- clone -- image- feature layering
05. //使用image的快照image- snap1克隆一个新的image- clone镜像
```

2) 查看克隆镜像与父镜像快照的关系

```
01. [root@node1 ~]# rbd info image- clone
02. rbd image 'image- clone':
03.   size 15360 MB in 3840 objects
04.   order 22 ( 4096 kB objects)
05.   block_name_prefix: rbd_data.d3f53d1b58ba
06.   format: 2
07.   features: layering
08.   flags:
09.   parent: rbd/image@image- snap1
10.   #克隆镜像很多数据都来自于快照链
11.   #如果希望克隆镜像可以独立工作，就需要将父快照中的数据，全部拷贝一份，但比较耗
12. [root@node1 ~]# rbd flatten image- clone
13. [root@node1 ~]# rbd info image- clone
14. rbd image 'image- clone':
15.   size 15360 MB in 3840 objects
16.   order 22 ( 4096 kB objects)
17.   block_name_prefix: rbd_data.d3f53d1b58ba
18.   format: 2
19.   features: layering
20.   flags:
21.   #注意，父快照信息没了！
```

步骤四：其他操作

1) 客户端撤销磁盘映射

```
01. [root@client ~]# umount /mnt
02. [root@client ~]# rbd showmapped
03. id pool image      snap device
04. 0 rbd image        - /dev/rbd0
05. //语法格式:
06. [root@client ~]# rbd unmap /dev/rbd/{ poolname} /{ imagename}
07. [root@client ~]# rbd unmap /dev/rbd/rbd/image
```

[Top](#)

2) 删除快照与镜像

01. [root@node1 ~] # rbd snap rm image -- snap image- snap
02. [root@node1 ~] # rbd list
03. [root@node1 ~] # rbd rm image