7_Cluster03ceph 概述部署 cephe 集群 ceph 块存储

一 存储的分类:

1.1 DAS 直连存储: direct attached storage

硬盘和主机用数据线直接连接,连接方式有 ide sata usb SAS(SAS 用于服务器)

1.2 NAS 网络附加存储: network attached storage

硬盘和主机通过网络连接,连接方式有 nfs samba ftp,共享文件系统(格式化后的块设备),不需要格式化,直接挂载使用

1.3 SAN 存储区域网络:storage area network

连接方式有 iscsi,共享一个裸磁盘(块设备),要使用需要先格式化

1.4 NAS 和 SAN 缺点:一个目录下只能挂载一个 SAN 或 NAS;没有备份;

1.5 SDS 软件定义存储: software define storage,分布式存储

数据存储在若干个主机上,通过网络连接,常用连接方式见 2.2

二 分布式文件系统

2.1 概念:

分布式文件系统(Distributed File System)指文件系统管理的物理存储资源不一定直接连接在本地节点上,而是通过计算机网络与节点相连

分布式文件系统的设计基于客户机/服务器模式

2.2 常用分布式文件系统

Lustre Hadoop FastDFS **Ceph** GlusterFS

三 Ceph

3.1 Ceph 是 1 个分布式文件系统

具有高扩展\高可用\高性能的特点

ceph 可以提供对象存储\块存储\文件系统存储

ceph 可以提供 PB\EB 级的存储空间

ceph 是存储行业的一大发展趋势,越来受到市场的认可

帮助文档:http://docs.ceph.org/start/info

3.2 ceph 组件(一个组件 1 个软件包)

3.2.1 OSDs:存储设备(软件包 ceph-osd)

3.2.2 Monitors:集群监控组件(软件包 ceph-mon)

3.2.3 RadosGateway(RGW):对象存储网关(软件包 ceph-radosgw)

3.2.4 MDSs: 存放文件系统的元数据(对象存储\块存储不需要该组件)(软件包 ceph-mds)

3.2.5 Client: ceph 客户端

3.3 ceph 默认 3 副本:1 份数据同时在随机的 3 台主机上存储 3 份,若其中 1 台主机损坏,在其他别的 1 台主机上再存储 1 份,始终保持 3 份副本

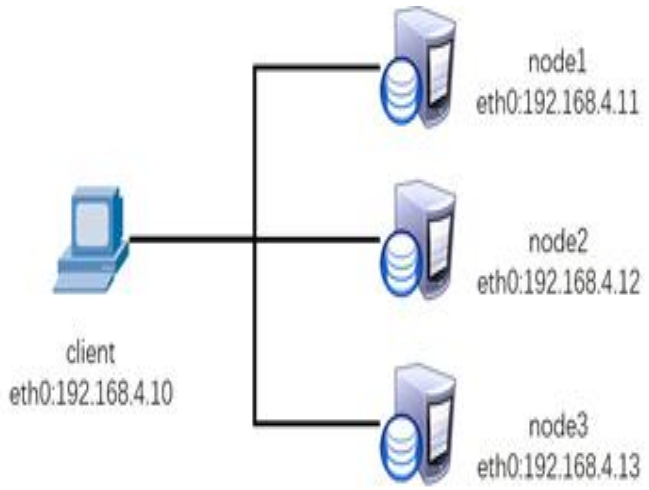
四 案例：ceph 实验环境

4.1 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

4.2 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。



主机名称	值
client	eth0:192.168.4.10/24
node1	eth0:192.168.4.11/24
node2	eth0:192.168.4.12/24
node3	eth0:192.168.4.13/24

图-1　　表一1 主机名称及对应 IP 地址表

所有主机的主机名及对应的 IP 地址如表-1 所示。

注意：所有主机基本系统光盘的 YUM 源必须提前配置好。

4.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：安装前准备

1）物理机为所有节点配置 yum 源服务器

提示：ceph10.iso 在/linux-soft/02 目录。

```
room9pc01 ~]# mkdir  /var/ftp/ceph

room9pc01 ~]# mount ceph10.iso /var/ftp/ceph/
```

2）配置无密码连接(包括自己远程自己也不需要密码)，在 node1 操作。

```
node1 ~]# ssh-keygen   -f /root/.ssh/id_rsa    -N ''
```

```
node1 ~]# for i in 10  11  12  13
do
      ssh-copy-id  192.168.4.$i
done
```

3）修改/etc/hosts 并同步到所有主机

警告：/etc/hosts 解析的域名必须与本机主机名一致！！！！

```
node1 ~]# cat /etc/hosts
... ...
192.168.4.10    client
192.168.4.11    node1
192.168.4.12    node2
192.168.4.13    node3
#警告：/etc/hosts 解析的域名必须与本机主机名一致！！！！
node1 ~]# for i in client node1  node2  node3
do
scp  /etc/hosts   $i:/etc/
done
```

4）修改所有节点都需要配置 YUM 源，并同步到所有主机

```
node1 ~]# cat /etc/yum.repos.d/ceph.repo
[mon]
```

```
name=mon
baseurl=ftp://192.168.4.254/ceph/MON
gpgcheck=0
[osd]
name=osd
baseurl=ftp://192.168.4.254/ceph/OSD
gpgcheck=0
[tools]
name=tools
baseurl=ftp://192.168.4.254/ceph/Tools
gpgcheck=0
node1 ~]# yum repolist  #验证YUM源软件数量
源标识      源名称      状态
Dvd         redhat     9,911
Mon         mon        41
Osd         osd        28
Tools       tools      33
repolist: 10,013
node1 ~]# for i in  client  node1  node2  node3
do
```

```
scp  /etc/yum.repos.d/ceph.repo   $i:/etc/yum.repos.d/
done
```

5）所有节点主机与真实主机的 NTP 服务器同步时间。

提示：默认真实物理机已经配置为 NTP 服务器。

```
node1 ~]# vim /etc/chrony.conf
server 192.168.4.254   iburst
node1 ~]# for i in client  node1  node2  node3
do
    scp /etc/chrony.conf $i:/etc/
    ssh  $i  "systemctl restart chronyd"
done
```

步骤三：准备存储磁盘

物理机上为每个虚拟机准备 3 块 20G 磁盘（node1 2 3 添加）。

```
room9pc01 ~]# virt-manager
```

五 案例：部署 ceph 集群

5.1 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

5.2 步骤

步骤一：安装部署软件 **ceph-deploy**

1）在 **node1** 安装部署工具，学习工具的语法格式。

```
node1 ~]#  yum -y install ceph-deploy(该软件用户部署配置 ceph)
node1 ~]#  ceph-deploy  --help
node1 ~]#  ceph-deploy mon --help
```

2）创建目录

```
node1 ~]#  mkdir ceph-cluster
node1 ~]#  cd ceph-cluster/
```

步骤二：部署 **Ceph** 集群

1）创建 **Ceph** 集群配置,在 **ceph-cluster** 目录下生成 **Ceph** 配置文件。

在 ceph.conf 配置文件中定义 **monitor** 主机。

```
node1 ceph-cluster]# ceph-deploy new node1 node2 node3
```

2）给所有节点安装 **ceph** 相关软件包。

```
node1 ceph-cluster]# :for i in node1 node2 node3
do
    ssh    $i   "yum   -y   install   ceph-mon   ceph-osd   ceph-mds
ceph-radosgw"
done
```

3）初始化所有节点的 **mon** 服务，也就是启动 **mon** 服务（主机名解析必须对）。

```
node1 ceph-cluster]# ceph-deploy mon create-initial
```

常见错误及解决方法（非必要操作，有错误可以参考）：

如果提示如下错误信息：

```
[node1][ERROR ] admin_socket: exception getting command
descriptions: [Error 2] No such file or directory
```

解决方案如下（在 node1 操作）：

先检查自己的命令是否是在 ceph-cluster 目录下执行的！！！！如果确认是在该目录下执行的 create-initial 命令，依然报错，可以使用如下方式修复。

```
node1 ceph-cluster]# vim ceph.conf      #文件最后追加以下内容
public_network = 192.168.4.0/24
```

修改后重新推送配置文件：

```
node1 ceph-cluster]# ceph-deploy --overwrite-conf config push
node1 node2 node3
```

步骤三：创建 OSD

备注：vdb1 和 vdb2 给 vdc 和 vdd 做缓存磁盘。实际生产中为固态磁盘

```
node1 ceph-cluster]# for i in node1 node2 node3
do
    ssh $i "parted /dev/vdb mklabel gpt"
    ssh $i "parted /dev/vdb mkpart primary 1 50%"
    ssh $i "parted /dev/vdb mkpart primary 50% 100%"
 done
```

2）磁盘分区后的默认权限无法让 ceph 软件对其进行读写操作，需要修改权限。

node1、node2、node3 都需要操作，这里以 node1 为例。

```
node1 ceph-cluster]# chown  ceph.ceph  /dev/vdb1
node1 ceph-cluster]# chown  ceph.ceph  /dev/vdb2
#上面的权限修改为临时操作，重启计算机后，权限会再次被重置。
#我们还需要将规则写到配置文件实现永久有效。临时和永久都需要修改.
#规则：如果设备名称为/dev/vdb1 则设备文件的所有者和所属组都设置为 ceph。
#规则：如果设备名称为/dev/vdb2 则设备文件的所有者和所属组都设置为 ceph。
node1 ceph-cluster]# vim /etc/udev/rules.d/70-vdb.rules
ENV{DEVNAME}=="/dev/vdb1",OWNER="ceph",GROUP="ceph"
ENV{DEVNAME}=="/dev/vdb2",OWNER="ceph",GROUP="ceph"
```

3）初始化清空磁盘数据（仅 node1 操作即可）。

```
node1 ceph-cluster]# ceph-deploy disk  zap  node1:vdc  node1:vdd
node1 ceph-cluster]# ceph-deploy disk  zap  node2:vdc  node2:vdd
node1 ceph-cluster]# ceph-deploy disk  zap  node3:vdc  node3:vdd
```

4）创建 OSD 存储空间（在 node1 操作,将 vdb1\2 划作 vdc vdd 缓存）

重要：很多同学在这里会出错！将主机名、设备名称输入错误！！！

```
node1 ceph-cluster]# ceph-deploy osd create \
node1:vdc:/dev/vdb1 node1:vdd:/dev/vdb2
#创建 osd 存储设备，vdc 为集群提供存储空间，vdb1 提供 JOURNAL 缓存，
```

#每个存储设备对应一个缓存设备，缓存需要 SSD，不需要很大

```
node1 ceph-cluster]# ceph-deploy osd create \
 node2:vdc:/dev/vdb1 node2:vdd:/dev/vdb2
node1 ceph-cluster]# ceph-deploy osd create \
 node3:vdc:/dev/vdb1 node3:vdd:/dev/vdb2
```

常见错误及解决方法（非必须操作）。

使用 osd create 创建 OSD 存储空间时，如提示下面的错误提示：

```
[ceph_deploy][ERROR ] RuntimeError: bootstrap-osd keyring not
found; run 'gatherkeys'
```

可以使用如下命令修复文件，重新配置 ceph 的密钥文件：

```
node1 ceph-cluster]#  ceph-deploy gatherkeys node1 node2 node3
```

步骤四：验证测试

1）查看集群状态。

```
node1 ~]#  ceph  -s;ceph osd tree;
```

2）常见错误（非必须操作）。

如果查看状态包含如下信息：

```
health: HEALTH_WARN
       clock skew detected on  node2, node3…
```

clock skew 表示时间不同步，解决办法：请先将所有主机的时间都使用 NTP 时间同步！！！

Ceph 要求所有主机时差不能超过 0.05s，否则就会提示 WARN。

如果状态还是失败，可以尝试执行如下命令，重启 ceph 服务：

```
node1 ~]#  systemctl restart ceph\*.service ceph\*.target
```

六 案例：创建 Ceph 块存储

共享池:(默认自带创建,查看命令:ceph osd lspools,见步骤 1)

共享镜像 1　共享镜像 2　共享镜像 3

上例的 120G 的 osd 不能给 client 直接使用,需要先创建共享池,定义共享池内的镜像时,容量可以随意写,但共享池内的共享镜像的实际使用量之和的上限为 120G.

6.1 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

6.2 步骤

步骤一：创建镜像

1）查看存储池。

```
node1 ~]# ceph osd lspools
0 rbd,
```

2）创建镜像、查看镜像

```
node1 ~]# rbd create demo-image --image-feature  layering --size
10G
```

```
node1 ~]# rbd create rbd/image  --image-feature  layering --size
10G
#这里的 demo-image 和 image 为创建的镜像名称，可以为任意字符。
#--image-feature 参数指定我们创建的镜像的功能，layering 是开启 COW 功能。
#提示：ceph 镜像支持很多功能，但很多是操作系统不支持的，我们只开启 layering。
node1 ~]# rbd list
node1 ~]# rbd info demo-image
rbd image 'demo-image':
    size 10240 MB in 2560 objects
    order 22 (4096 kB objects)
    block_name_prefix: rbd_data.d3aa2ae8944a
    format: 2
    features: layering
```

步骤二：动态调整

1）缩小容量

```
node1 ~]# rbd resize --size 7G image --allow-shrink
node1 ~]# rbd info image
```

2）扩容容量

```
node1 ~]# rbd resize --size 15G image
node1 ~]# rbd info image
```

步骤三：通过 KRBD 访问

1）客户端通过 KRBD 访问

#客户端需要安装 ceph-common 软件包

#拷贝配置文件（否则不知道集群在哪）

#拷贝连接密钥（否则无连接权限）

```
client ~]# yum -y  install ceph-common

client ~]# scp 192.168.4.11:/etc/ceph/ceph.conf  /etc/ceph/

client ~]# scp 192.168.4.11:/etc/ceph/ceph.client.admin.keyring
/etc/ceph/  #复制连接密钥

client ~]# rbd map image   #rbd map 共享镜像

client ~]#  lsblk

client ~]# rbd showmapped

id pool image snap device

0  rbd  image -    /dev/rbd0

rdb showmapped  #查看对应关系
```

2) 客户端格式化、挂载分区

```
client ~]# mkfs.xfs /dev/rbd0

client ~]# mount /dev/rbd0 /mnt/

client ~]# echo "test" > /mnt/test.txt
```

步骤四：创建镜像快照

1) 查看镜像快照（默认所有镜像都没有快照）。

node1 ~]# **rbd snap ls** image #rbd snap ls 镜像名

2) 给镜像创建快照。

node1 ~]# **rbd snap create** image **--snap** image-snap1

#为 image 镜像创建快照，快照名称为 image-snap1

node1 ~]# rbd snap ls image

SNAPID NAME SIZE

4 image-snap1 15360 MB

3) 删除客户端写入的测试文件

client ~]# rm -rf /mnt/test.txt #客户端删除文件

client ~]# umount /mnt #客户端 umount 镜像

4) 还原快照

node1 ~]# **rbd snap rollback** image **--snap** image-snap1

client ~]# mount /dev/rbd0 /mnt/ #客户端重新挂载分区并查看

client ~]# ls /mnt

步骤五：创建快照克隆

1）克隆快照(克隆的是快照,不是镜像,克隆出来的是镜像,可直接挂载使用)

先对快照进行保护,再 clone 快照

node1 ~]# **rbd snap [un]protect image** --snap image-snap1

node1 ~]# **rbd snap rm image** --snap image-snap1 #会失败

```
node1 ~]#  rbd clone image --snap image-snap1(被克隆的快照)
image-clone(克隆出来的镜像) --image-feature layering
#使用 image 的快照 image-snap1 克隆一个新的镜像,命名为 image-clone
```

2）查看克隆镜像与父镜像快照的关系

```
node1 ~]#  rbd info image-clone
rbd image 'image-clone':
    size 15360 MB in 3840 objects
    order 22 (4096 kB objects)
    block_name_prefix: rbd_data.d3f53d1b58ba
    format: 2
    features: layering
    flags:
    parent: rbd/image@image-snap1
#克隆镜像很多数据都来自于快照链
#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
耗时！！！
node1 ~]#  rbd flatten image-clone  #将克隆镜像与父快照解除关联
node1 ~]#  rbd info image-clone
rbd image 'image-clone':
    size 15360 MB in 3840 objects
```

```
    order 22 (4096 kB objects)
    block_name_prefix: rbd_data.d3f53d1b58ba
    format: 2
    features: layering
    flags:
```

#注意，父快照信息没了！

```
node1 ~]#rbd snap unprotect image --snap image-snap1
```

#取消快照保护

```
node1 ~]# rbd snap rm image --snap image-snap1  #可以删除快照
```

解除关联-->取消保护-->删除

步骤六：其他操作

1） 客户端撤销磁盘映射

```
client ~]# umount /mnt
client ~]# rbd showmapped
id pool image       snap device
0  rbd  image       -    /dev/rbd0
```

#语法格式:

```
client ~]# rbd unmap /dev/rbd0
```