**ceph概述**

基础知识

什么是分布式文件系统

分布式文件系统（Distributed File System）是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

常用的分布式文件系统

Lustre

Hadoop

FastDFS

Ceps

GlusterFS

什么是ceph

ceph是一个分布式文件系统，可以直接往里头加存储空间

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级别的存储空间（PV->TB->GB）

1024G\*1024G=1048576G

软件定义存储（software Defined Storage）作为存储行业的一大发展趋势，已经越来越受到市场的认可

ceph组件

OSDs

存储设备。真正存储数据的组件。一般来说，每块参与存储的磁盘都需要一个OSD进程

monitors

集群监控组件。MON通过保存一系列集群状态map来监视集群状态，所以需要多台;另外，mon需要是基数，如果出现分歧，则采用投票机制，少数服从多数

MDSs

存放文件系统的元数据（对象存储和块存储不需要该组件），只有cephfs需要它。元数据：metadata，存储数据的数据。比如一本书内容是数据，那么书的作者、出版社、出版时间之类的信息就是元数据

RADOS：可靠自主分布式对象存储。它是ceph存储的基础，保证一切都以对象形式存储。

RBD：RADOS块设备，提供块存储

cephfs：提供文件系统级别的存储

RGW：RADOS网关，提供对象存储

client

ceph客户端

CEPH应用

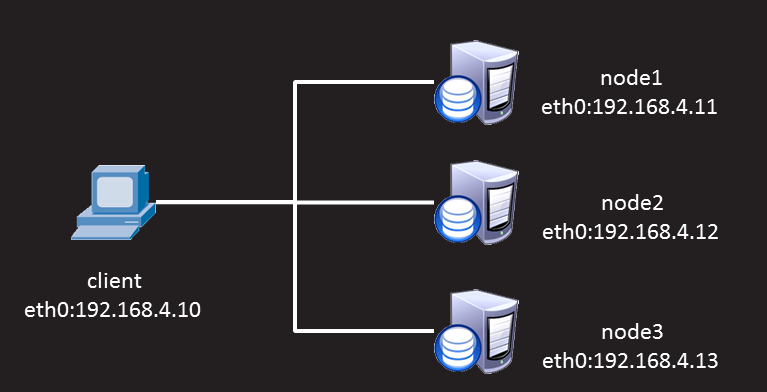
1. 块存储：使用最多的一种方式
2. cephFs：了解，不建议在生产环境中使用，因为还不成熟
3. 对象存储：了解，使用亚马逊s3

实验环境准备

实验拓扑图

1台客户端虚拟机

3台存储集群虚拟机



配置yum

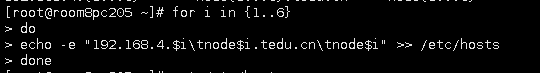
物理机创建Yum源服务器

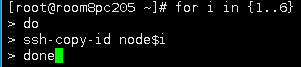
yum -y install vsftpd

mkdir /var/ftp/ceph

mount -o loop rhcs2.0-rhosp9-20161113-86\_64.iso /var/ftp/ceph

systemctl restart vsftpd











虚拟机调用yum源（以其中node1为例，并配置为管理节点）

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

[mon]

name=mon

baseurl=ftp://192.168.4.254/ceph/rhcs2.0-rhosp9-20161113-86\_64.iso/MON

enable=1

gpgcheck=0

[osd]

name=osd

baseurl=ftp://192.168.4.254/ceph/rhcs2.0-rhosp9-20161113-86\_64.iso/OSD

enable=1

gpgcheck=0

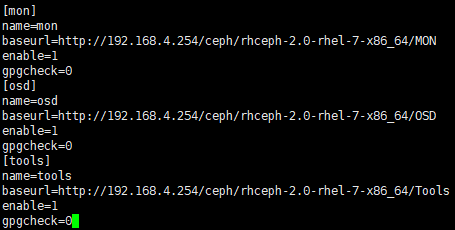
[tools]

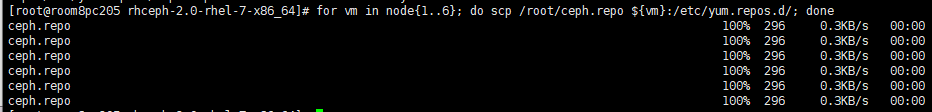
name=tools

baseurl=ftp://192.168.4.254/ceph/rhcs2.0-rhosp9-20161113-86\_64.iso/Tools

enable=1

gpgcheck=0





配置SSH无密码连接

修改主机名

vim /etc/hosts

192.168.4.10 client

192.168.4.11 node1

192.168.4.12 node2

192.168.4.13 node3

for i in 10 11 12 13

>do

>scp /etc/hosts 192.168.4.$i:/etc

>done





非交互生成密钥对

ssh-keygen -f /root/.ssh/id\_rsa -N ‘ ‘



发布密钥到各个主机（包括自己）

for i in 10 11 12 13

>do

>ssh-copy-id 192.168.4.$i

>done



NTP时间同步（配置node6为时间服务器）

客户端创建NTP服务器

yum -y install chrony



cat /etc/chrony.conf

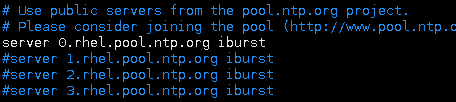
server0.centos.pool.ntp.org iburst

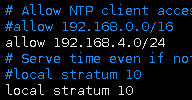
allow 192.168.4.0/24

local stratum 10

systemctl restart chronyd



注释3个



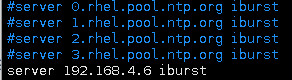


其他所有主机与其同步时间（以node1为例）

cat /etc/chrony.conf

server 192.168.4.10 iburst

systemctl restart chronyd







准备存储磁盘

物理机上为每个虚拟机创建3个磁盘（node1-node3各添加三个10GB磁盘）

cd /var/lib/libvirt/images

qemu-img create -f qcow2 node1-vdb.vol 10G

qemu-img create -f qcow2 node1-vdb.vol 10G

qemu-img create -f qcow2 node1-vdb.vol 10G

qemu-img create -f qcow2 node2-vdb.vol 10G

qemu-img create -f qcow2 node2-vdb.vol 10G

qemu-img create -f qcow2 node2-vdb.vol 10G

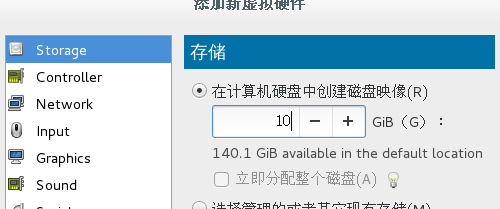
qemu-img create -f qcow2 node3-vdb.vol 10G

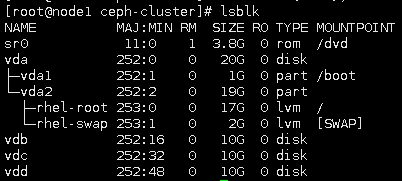
qemu-img create -f qcow2 node3-vdb.vol 10G

qemu-img create -f qcow2 node3-vdb.vol 10G

在图形环境中为虚拟机添加磁盘

virt-manager





**部署ceph集群**

准备部署环境

安装部署软件

使用node1作为部署主机

yum -y install ceph-deploy



ceph-deploy命令与子命令都支持--help查看帮助

ceph-deploy --help

创建目录

为部署工具创建目录，存放密钥与配置文件

mkdir ceph-cluster

cd ceph-cluster



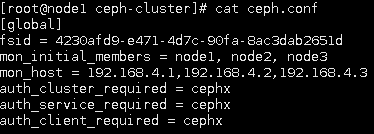
部署存储集群

创建ceph集群配置（所有节点都为mon）

ceph-deploy new node1 node2 node3







给所有节点都安装ceph软件包

ceph-deploy install node1 node2 node3



初始化所有节点的mon服务（主机名解析必须对）

ceph-deploy mon create-initial



这里没有指定主机，是因为第一步创建的配置文件中已经有了，所以要求主机名解析必须对，否则连接不到对应的主机

如果有报错：

[ERROR]admin\_socket:expection getting command descriptions:[Errno2]no such file or deirectory

解决方案：

vim ceph.conf最下面加入public\_network=192.168.0.0/24

再执行以下命令：ceph-deploy --overwrite-conf config push node1 node2 node3

创建OSD

把node1-3的vdb作为日志盘。ext/xfs都是日志文件系统，一个分区分成日志区和数据区。为了更好的性能，vdb专门作为vdc和vdd的日志盘（下面以node1为例）

parted /dev/vdb mklabel gpt

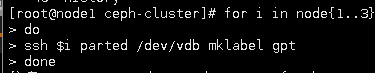
parted /dev/vdb mkpart primary 1M 50%

parted /dev/vdb mkpart primary 50% 100%

chown ceph.ceph /dev/vdb1

chown ceph.ceph /dev/vdb2

这两个分区用来做存储服务器的日志journal盘









创建udev规则（否则重启后属主和属组会改变），使得vdb1和vdb2重启后，属主属组仍然是ceph

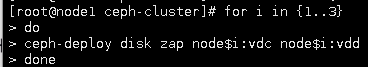


初始化清空磁盘数据（仅node1操作即可）

ceph-deploy disk zap node1:vdc node1:vdd

ceph-deploy disk zap node2:vdc node2:vdd

ceph-deploy disk zap node3:vdc node3:vdd

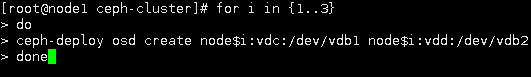


创建OSD存储空间（仅node1操作即可）

ceph-deploy osd node1:vdc:/dev/vdb1 node1:vdd:/dev/vdb2

ceph-deploy osd node2:vdc:/dev/vdb1 node2:vdd:/dev/vdb2

ceph-deploy osd node3:vdc:/dev/vdb1 node3:vdd:/dev/vdb2

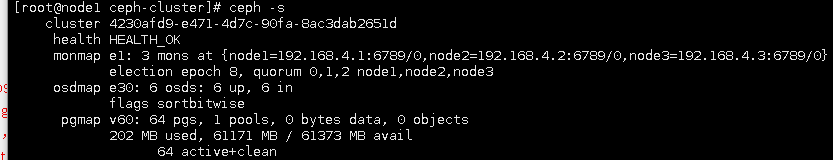


创建osd存储设备，vdc为集群提供存储空间，vdb1提供JOURNAL日志，一个存储设备对应一个日志设备，日志需要SSD，不需要很大

验证

查看集群状态

ceph -s



ok即可

可能出现的错误

osd create创建OSD存储空间，如提示run ‘gatherkeys’

ceph-deploy gatherkeys node1 node2 node3

ceph -s查看状态，如果失败

systemctl restart ceph\\*.service ceph\\*.target

在所有节点，或仅在失败的节点重启服务

重启后状态为error，连接超时

systemctl restart ceph\\*

**ceph块存储（在创建时，如果创建的镜像容量比真实大小小的话，在使用满之后需要扩容操作，如果创建的镜像比真实容量大，则只能使用真实容量的大小）**

概述

什么是块存储

单机块设备

光盘

磁盘

分布式块存储

ceph

cinder

ceph块设备也叫做RADOS块设备

RADOS block device ：RBD

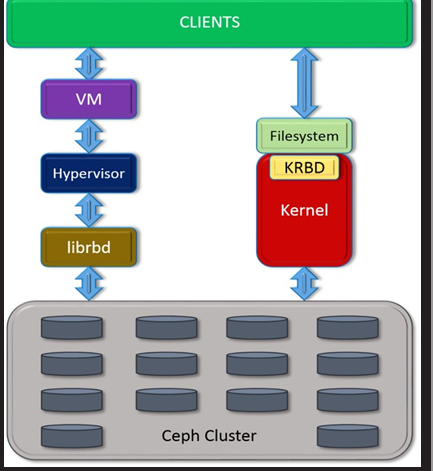
RBD驱动已经很好的集成在了linux内核中

RBD提供了企业功能，如快照、COW克隆等等

RBD还支持缓存，从而能够大大提高性能

linux内核可以直接访问ceph块存储

KVM可用借助于librbd访问



块存储集群

创建镜像

查看存储池（默认有一个rdb池）

ceph osd lspools

可以查看0号镜像池，名字为rbd

创建镜像、查看镜像

创建名为demo-image的镜像，大小为10GB

rbd create demo-image --image-feature layering --size 10G



创建第2个镜像，名为image，指定它位于rbd池中

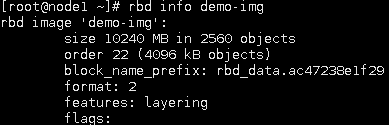
rbd create rbd/image --image-feature layering --size 10G



rbd list



rbd info demo-image

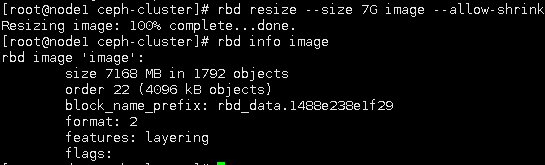


动态调整大小

缩小容量

rbd resize --size 7G image --allow-shrink

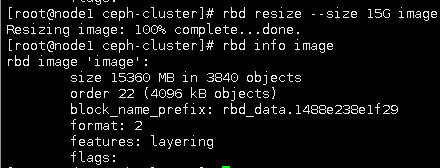
rbd info image



扩大容量

rbd resize --size 15G image

rbd info image



集群内通过KRDB访问

将镜像映射为本地磁盘

rbd map demo-image

lsblk

接下来，进行格式化

mkfs.xfs /dev/rdb0

mount /dev/rdb0 /mnt

客户端通过KRDB访问（将node6作为客户端，使用ceph创建的镜像作为存储设备）

客户端需要安装ceph-common软件包

yum -y install ceph-common



拷贝配置文件（否则不知道集群在哪里，该配置文件里面记录了ceph集群访问的方式和地址）

scp 192.168.4.11:/etc/ceph/ceph.conf /etc/ceph



拷贝连接密钥（否则无连接权限）

scp 192.168.4.11:/etc/ceph/ceph.client.admin.keyring /etc/ceph



ceph.client.admin.keyring是client.admin用户的密钥文件，client是类型

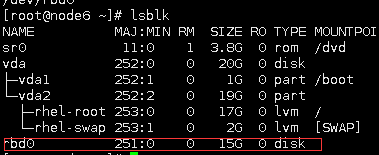
映射镜像到本地磁盘

rbd map image



/dev/rbd0是映射出来的盘

lsblk



rbd showmapped

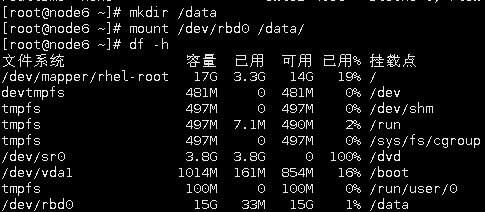


客户端格式化、挂载分区

mkfs.xfs /dev/rbd0



mount /dev/rbd0 /mnt



echo “a” > /mnt/test.txt



创建镜像快照

查看镜像快照

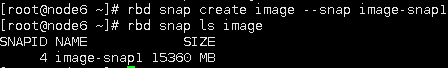
rbd snap ls image



创建镜像快照

rbd snap create image --snap image-snap1

rbd snap ls image

给image创建image-snap1的快照

注意：快照使用COW技术，对大数据快照速度会很快

使用快照恢复数据

删除客户端写入的测试文件

rm -rf /mnt/test.txt



卸载挂载

umount /mnt



还原快照

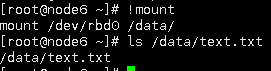
rbd snap rollback image --snap image-snap1



客户端重新挂载分区

mount /dev/rbd0 /mnt

ls /mnt



快照克隆（克隆出来的是镜像，也就是说一个快照可以创建多个镜像）

如果想从快照恢复一个新的镜像，则可以使用克隆

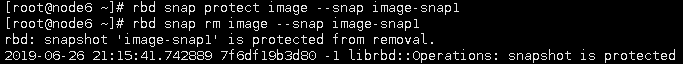
注意，克隆前，需要对快照进行<保护>操作

被保护的快照无法删除，取消保护（unprotect）

1.克隆快照，首先要先将快照保护起来，防止误删除之类的操作

rbd snap protect image --snap image-snap1

rbd snap rm image --snap image-snap1//会失败



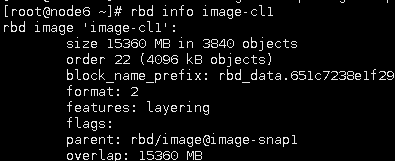
2.克隆image-snap1快照，克隆名称是image-cl1

rbd clone image --snap image-snap1 image-cl1 --image-feature layering



使用image的快照image-snap1克隆一个新的image-cl1镜像

3.查看信息（克隆镜像与父镜像快照关系）



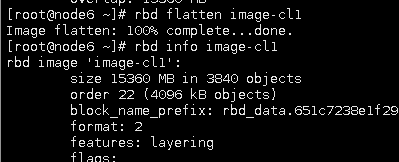
4.合并

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

rbd flatten image-clone

rbd info image-clone



客户端撤销磁盘映射

umount挂载点

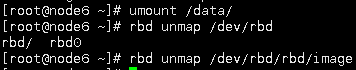
umount /mnt

取消RBD磁盘映射

rbd showmapped

rbd unmap /dev/rbd/{poolname}/{imagename}

rbd unmap /dev/rbd/rbd/image



删除快照与镜像

删除快照（确保快照未被保护）

rbd snap rm image --snap image-snap





删除镜像

rbd lst

rbd rm image

