

集群与存储

NSD CLUSTER

DAY05

内容

上午	09:00 ~ 09:30	作业讲解和回顾
	09:30 ~ 10:20	块存储应用案例
	10:30 ~ 11:20	
	11:30 ~ 12:20	分布式文件系统
下午	14:00 ~ 14:50	
	15:00 ~ 15:50	对象存储
	16:10 ~ 17:00	
	17:10 ~ 18:00	总结和答疑



块存储应用案例

块存储应用案例

准备实验环境

创建磁盘镜像

Ceph认证账户

部署客户端环境

创建KVM虚拟机

创建初始化虚拟机

配置libvirt secret

虚拟机的XML配置文件

修改XML配置文件

准备实验环境

创建磁盘镜像

- 为虚拟机创建磁盘镜像

```
[root@node1 ~]# rbd create vm1-image --image-feature layering --size 10G
[root@node1 ~]# rbd create vm2-image --image-feature layering --size 10G
```

- 查看镜像

```
[root@node1 ~]# rbd list
[root@node1 ~]# rbd info vm1-image
[root@node1 ~]# qemu-img info rbd:rbd/vm1-image
image: rbd:rbd/vm1-image
file format: raw
virtual size: 10G (10737418240 bytes)
disk size: unavailable
```



Ceph认证账户

- Ceph默认开启用户认证，客户端需要账户才可以访问
 - 默认账户名称为client.admin，key是账户的密钥
 - 可以使用ceph auth添加新账户（案例我们使用默认账户）

```
[root@node1 ~]# cat /etc/ceph/ceph.conf //配置文件
```

```
[global]
```

```
mon_initial_members = node1, node2, node3
```

```
mon_host = 192.168.2.10,192.168.2.20,192.168.2.30
```

```
auth_cluster_required = cephx //开启认证
```

```
auth_service_required = cephx //开启认证
```

```
auth_client_required = cephx //开启认证
```

```
[root@node1 ~]# cat /etc/ceph/ceph.client.admin.keyring //账户文件
```

```
[client.admin]
```

```
key = AQBTSdRapUxBKRAANXtteNUyoEmQHveb75bISg==
```



部署客户端环境

- 注意：这里使用真实机当客户端！！！！
- 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）

```
[root@room9pc01 ~]# yum -y install ceph-common
[root@room9pc01 ~]# scp 192.168.4.11:/etc/ceph/ceph.conf /etc/ceph/
[root@room9pc01 ~]# scp 192.168.4.11:/etc/ceph/ceph.client.admin.keyring \
/etc/ceph/
```

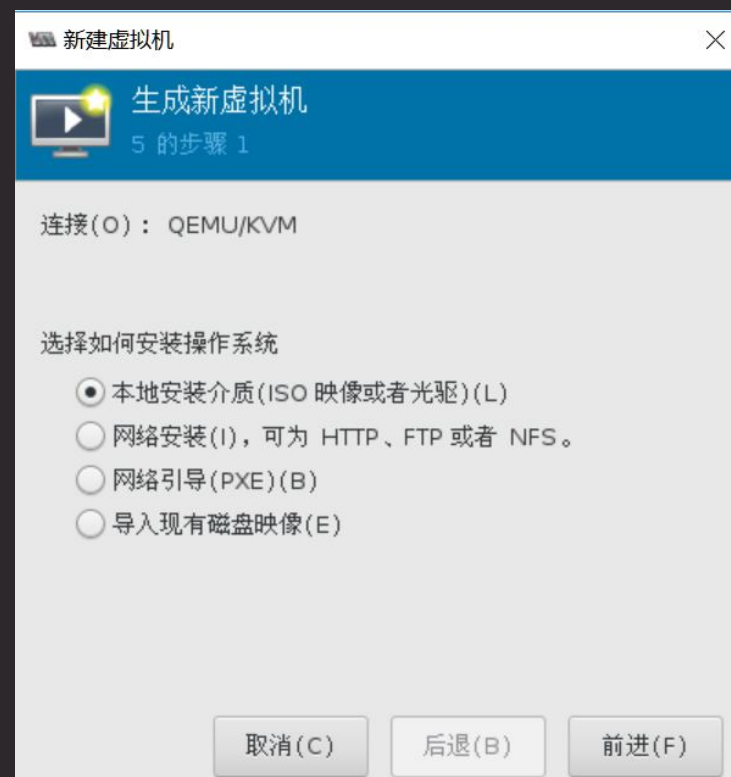
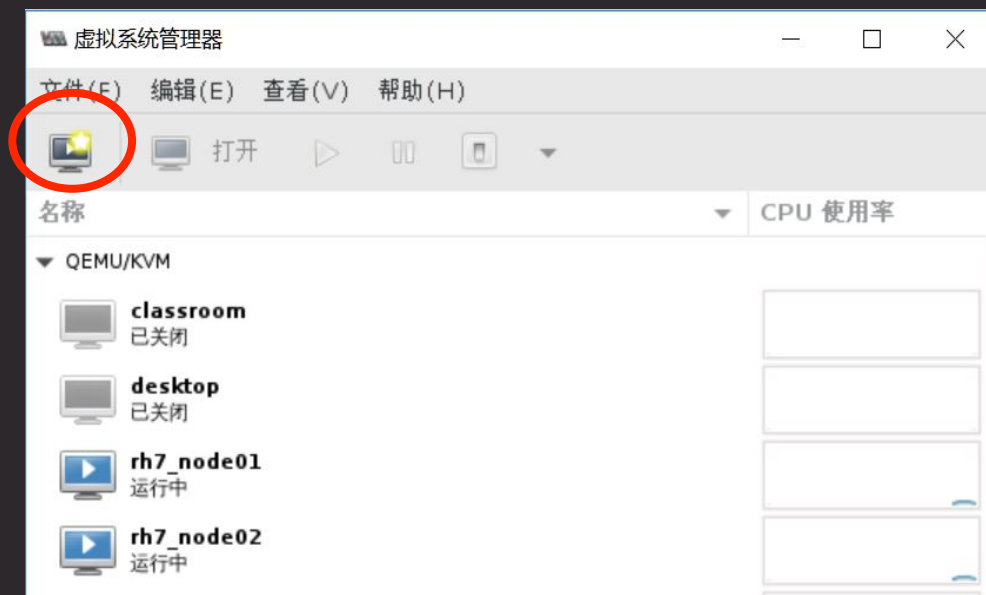


创建KVM虚拟机

创建初始化虚拟机

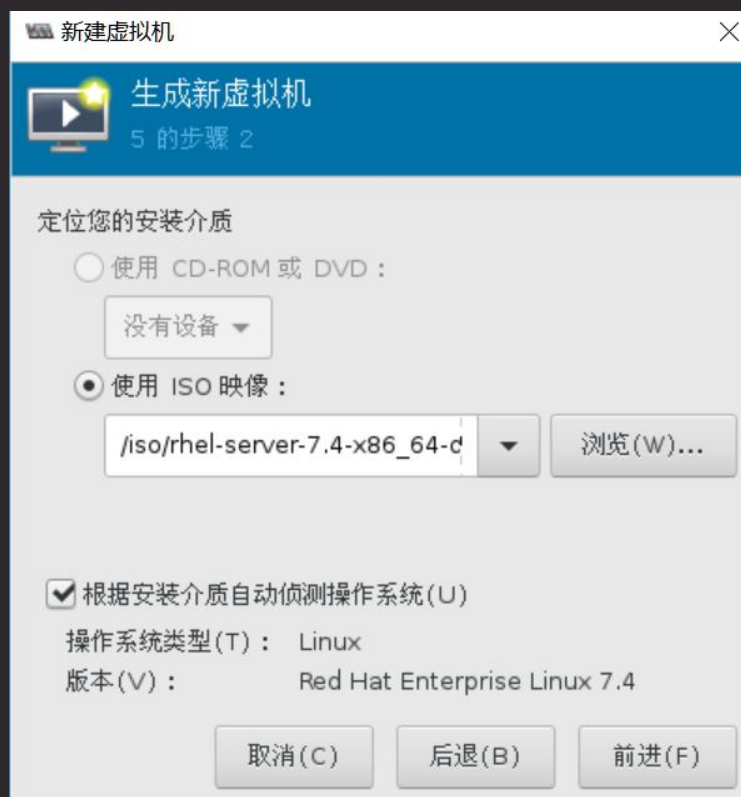
- 使用virt-manager创建2台普通的KVM虚拟机
 - 这里以1个虚拟机为例

`[root@room9pc01 ~]# virt-manager`



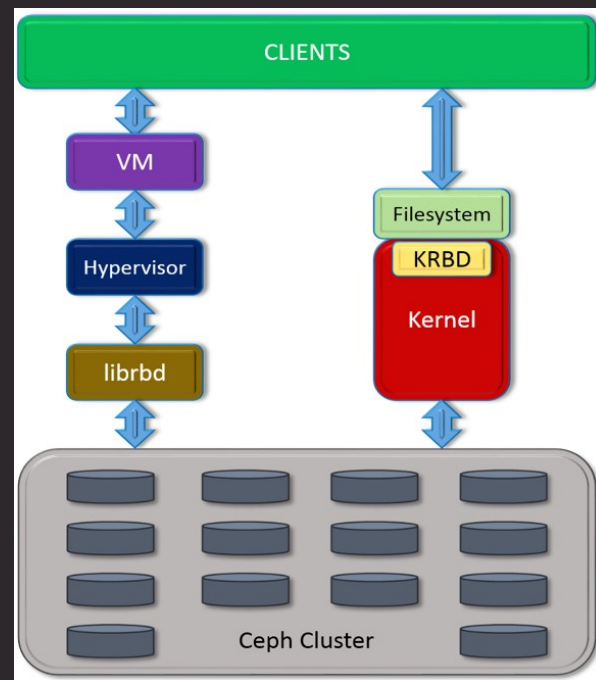
创建初始化虚拟机（续1）

- 创建虚拟机后，不着急启动虚拟机（关闭虚拟机）



配置libvirt secret

- KVM虚拟机需要使用librbd才可以访问ceph集群
- Librbd访问ceph又需要账户认证
- 所以这里，我们需要给libvirt设置账户信息



配置libvirt secret (续1)

- 编写账户信息文件 (真实机操作)

```
[root@room9pc01 ~]# vim secret.xml //新建临时文件，内容如下
<secret ephemeral='no' private='no'>
  <usage type='ceph'>
    <name>client.admin secret</name>
  </usage>
</secret>
```

- 使用XML配置文件创建secret

```
[root@room9pc01 ~]# virsh secret-define --file secret.xml
733f0fd1-e3d6-4c25-a69f-6681fc19802b
//随机的UUID，这个UUID对应的有账户信息
```



配置libvirt secret (续2)

- 编写账户信息文件 (真实机操作)

```
[root@room9pc01 ~]# ceph auth get-key client.admin
```

//获取client.admin的key , 或者直接查看密钥文件

```
[root@room9pc01 ~]# cat /etc/ceph/ceph.client.admin.keyring
```

- 设置secret , 添加账户的密钥

```
[root@room9pc01] virsh secret-set-value \
```

```
--secret 733f0fd1-e3d6-4c25-a69f-6681fc19802b \
```

```
--base64 AQBTSdRapUxBKRAANXtteNUyoEmQHveb75bISg
```

//这里secret后面是之前创建的secret的UUID

//base64后面是client.admin账户的密码

//现在secret中既有账户信息又有密钥信息



虚拟机的XML配置文件

- 每个虚拟机都会有一个XML配置文件，包括：
 - 虚拟机的名称、内存、CPU、磁盘、网卡等信息

```
[root@room9pc01 ~]# vim /etc/libvirt/qemu/vm1.xml
```

//修改前内容如下

```
<disk type='file' device='disk'>
    <driver name='qemu' type='qcow2'/>
    <source file='/var/lib/libvirt/images/vm1.qcow2'/>
    <target dev='vda' bus='virtio'/>
    <address type='pci' domain='0x0000' bus='0x00'
slot='0x07' function='0x0'/>
</disk>
```



修改XML配置文件

- 不推荐直接使用vim修改配置文件
- 推荐使用virsh edit修改配置文件

[root@room9pc01] virsh edit vm1 //vm1为虚拟机名称

```
<disk type='network' device='disk'>
  <driver name='qemu' type='raw' />
  <auth username='admin'>
    <secret type='ceph' uuid='733f0fd1-e3d6-4c25-a69f-6681fc19802b' />
  </auth>
  <source protocol='rbd' name='rbd/vm1'>
    <host name='192.168.4.11' port='6789' />
  </source>
  <target dev='vda' bus='virtio' />
  <address type='pci' domain='0x0000' bus='0x00' slot='0x07' function='0x0' />
</disk>
```



修改XML配置文件（续1）

- 关键词说明

```
<secret type='ceph' uuid='3b8b0c5c-bebc-4fc2-9137-0e3deb61dc8b' />
```

//这里的uuid就是secret的uuid，有client.admin账户和密钥信息

```
<source protocol='rbd' name='rbd/vm1'>
```

```
<host name='192.168.4.11' port='6789' />
```

```
</source>
```

//这里说明使用账户连接哪台ceph主机和端口，访问哪个池和镜像

```
<target dev='vda' bus='virtio' />
```

//这里说明，将获取的镜像，设置为虚拟机的vda磁盘

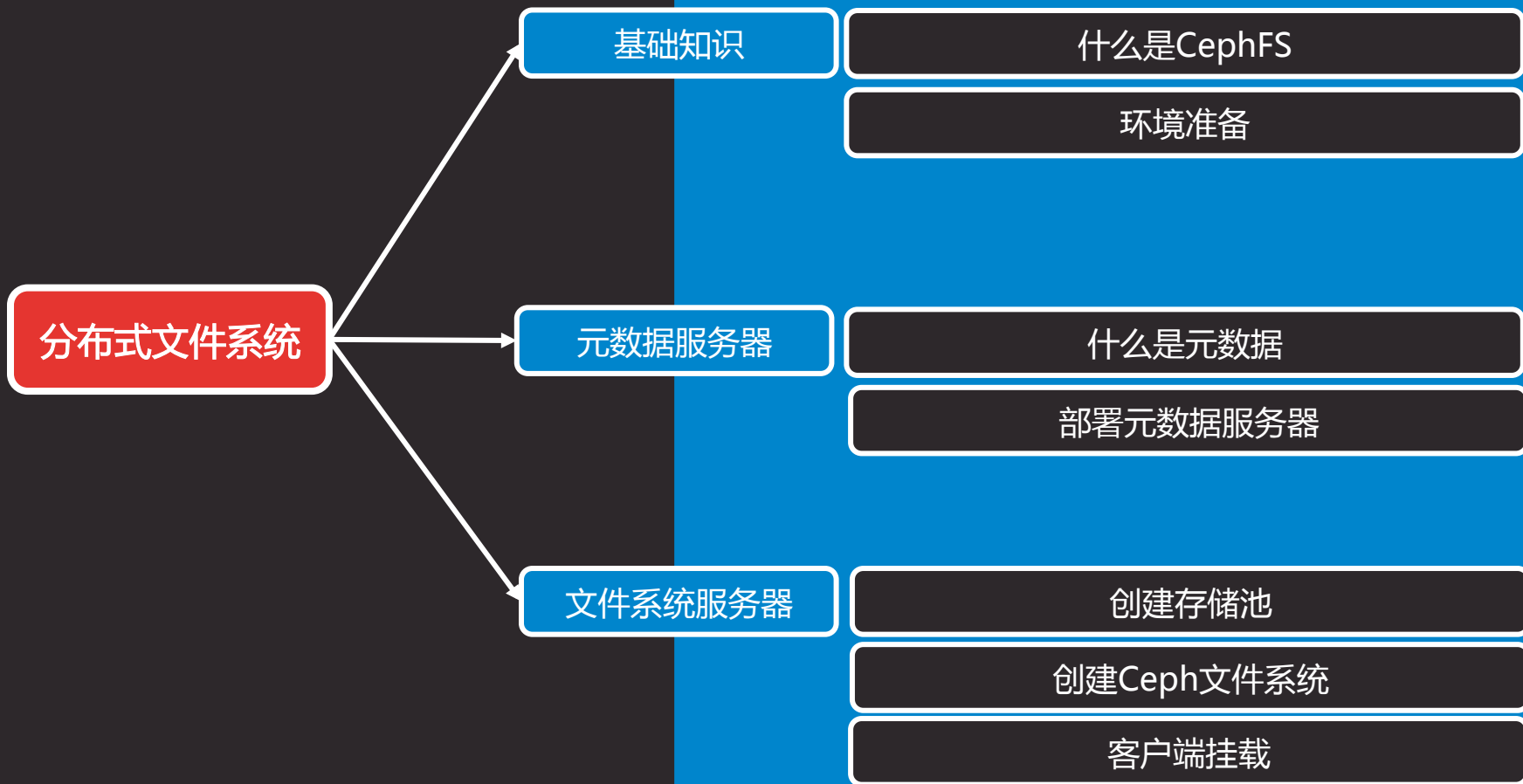


案例1：块存储应用案例

- Ceph创建块存储镜像
- 客户端安装部署ceph软件
- 客户端部署虚拟机
- 客户端创建secret
- 设置虚拟机配置文件，调用ceph存储



分布式文件系统



基础知识

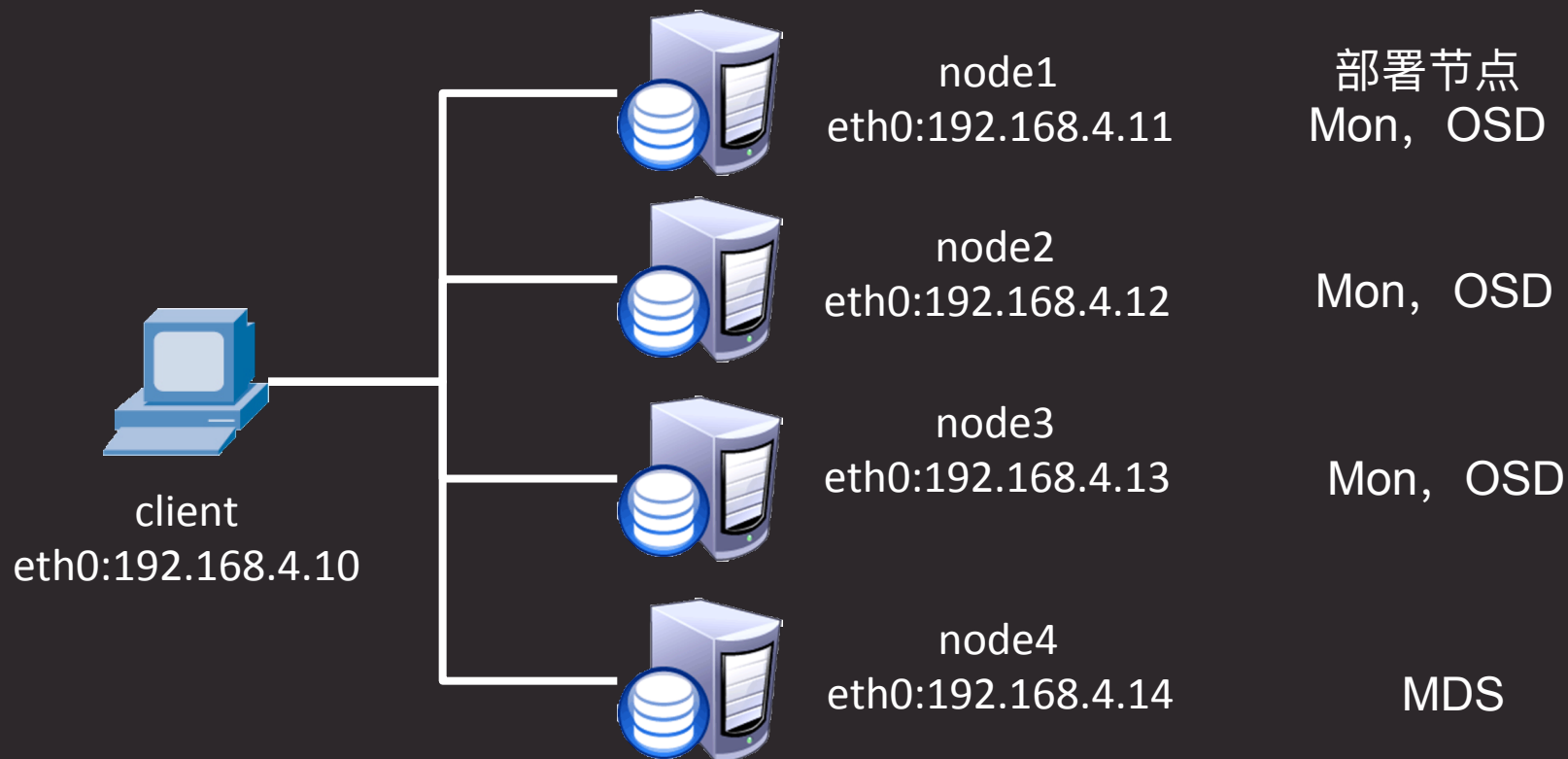
什么是CephFS

- 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
- CephFS使用Ceph集群提供与POSIX兼容的文件系统
- 允许Linux直接将Ceph存储mount到本地



环境准备

- 环境拓扑



环境准备（续1）

- 准备一台新的虚拟机，作为元数据服务器
- 要求如下：
 - IP地址:192.168.4.14
 - 主机名:node4
 - 配置yum源（包括rhel、ceph的源）
 - 与Client主机同步时间
 - node1允许无密码远程node4
 - 修改node1的/etc/hosts，并同步到所有node主机



元数据服务器

什么是元数据

- 元数据 (Metadata)
 - 任何文件系统中的数据分为数据和元数据。
 - 数据是指普通文件中的实际数据
 - 而元数据指用来描述一个文件的特征的系统数据
 - 比如：访问权限、文件拥有者以及文件数据块的分布信息(inode...)等
- 所以CephFS必须有MDSs节点



部署元数据服务器

- 登陆node4，安装ceph-mds软件包

```
[root@node4 ~]# yum -y install ceph-mds
```

- 登陆node1部署节点操作

```
[root@node1 ~]# cd /root/ceph-cluster
```

//该目录，是最早部署ceph集群时，创建的目录

```
[root@node1 ceph-cluster]# ceph-deploy mds create node4
```

//给nod4拷贝配置文件，启动mds服务

- 同步配置文件和key

```
[root@node1 ceph-cluster]# ceph-deploy admin node4
```



文件系统服务器

创建存储池

- 文件系统需要至少2个池
 - 一个池用于存储数据
 - 一个池用于存储元数据

```
[root@node4 ~]# ceph osd pool create cephfs_data 128  
//创建存储池，对应128个PG
```

```
[root@node4 ~]# ceph osd pool create cephfs_metadata 128  
//创建存储池，对应128个PG
```



创建Ceph文件系统

- 使用前面创建的池，创建文件系统

```
[root@node4 ~]# ceph mds stat           //查看mds状态
e2:, 1 up:standby
```

```
[root@node4 ~]# ceph fs new myfs1 cephfs_metadata cephfs_data
new fs with metadata pool 2 and data pool 1
//注意，现在medadata池，再写data池
//默认，只能创建1个文件系统，多余的会报错
```

```
[root@node4 ~]# ceph fs ls
name: myfs1, metadata pool: cephfs_metadata, data pools: [cephfs_data ]
```

```
[root@node4 ~]# ceph mds stat
e4: 1/1/1 up {0=node4=up:creating}
```



客户端挂载

- Linux内核支持ceph文件系统（不需要装软件）

```
[root@client ~]# mount -t ceph 192.168.4.11:6789:/ /mnt/cephfs/ \  
-o name=admin,secret=AQBTsdRapUxBKRAANXtteNUyoEmQHveb75bISg==
```

//注意:文件系统类型为ceph

//192.168.4.11为MON节点的IP（不是MDS节点）

//admin是用户名,secret是密钥

//密钥可以在/etc/ceph/ceph.client.admin.keyring中找到



案例2：Ceph文件系统

- 部署MDSs节点
- 创建Ceph文件系统
- 客户端挂载文件系统



Ceph对象存储

Ceph对象存储

概述

什么是对象存储

环境准备

对象存储

FastDFS系统结构

FastDFS上传文件过程

FastDFS下载文件过程

概述

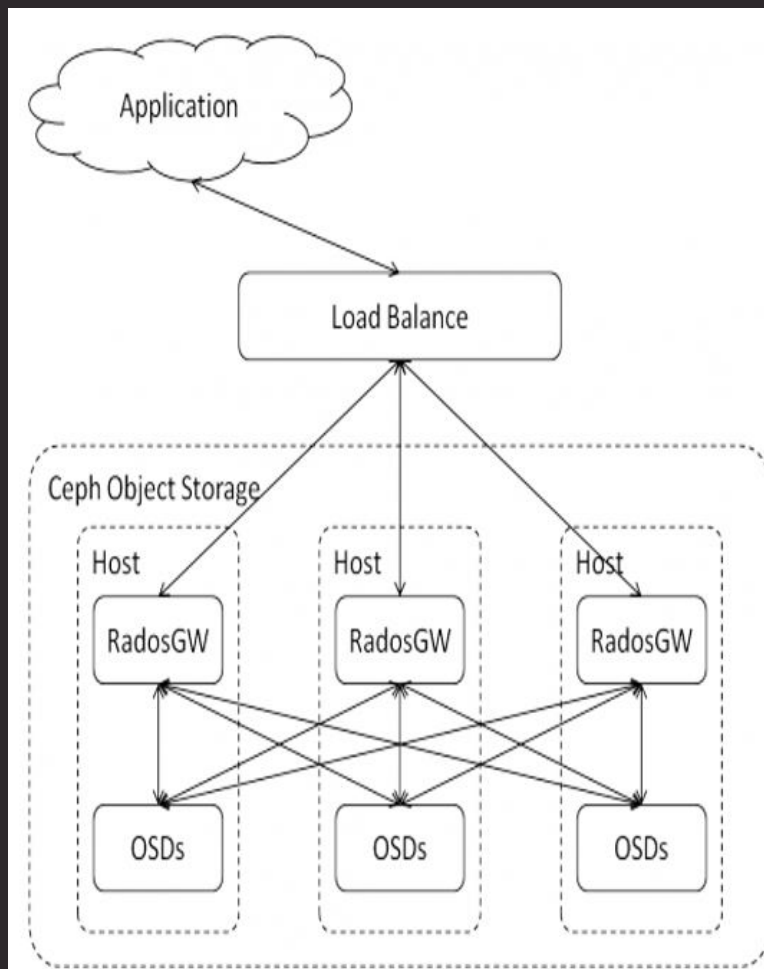
什么是对象存储

- 对象存储
 - 也就是键值存储，通过接口指令，也就是简单的GET、PUT、DEL和其他扩展，向存储服务上传下载数据
 - 对象存储中所有数据都被认为是一个对象，所以，任何数据都可以存入对象存储服务器，如图片、视频、音频等
- RGW全称是Rados Gateway
- RGW是Ceph对象存储网关，用于向客户端应用呈现存储界面，提供RESTful API访问接口



什么是对象存储

- RGW可以部署多台，拓扑如下：



环境准备

- 准备一台新的虚拟机，作为元数据服务器
- 要求如下：
 - IP地址:192.168.4.15
 - 主机名:node5
 - 配置yum源（包括rhel、ceph的源）
 - 与Client主机同步时间
 - node1允许无密码远程node5
 - 修改node1的/etc/hosts，并同步到所有node主机



对象存储

部署RGW软件包

- 用户需要通过RGW访问存储集群
 - 通过node1安装ceph-radosgw软件包

```
[root@node1 ~]# ceph-deploy install --rgw node5
```

- 同步配置文件与密钥到node5

```
[root@node1 ~]# cd /root/ceph-cluster
```

```
[root@node1 ~]# ceph-deploy admin node5
```



新建网关实例

- 启动一个rgw服务

```
[root@node1 ~]# ceph-deploy rgw create node5
```

- 登陆node5验证服务是否启动

```
[root@node5 ~]# ps aux | grep radosgw
```

```
ceph    4109  0.2  1.4 2289196 14972 ?        Ssl  22:53   0:00 /usr/bin/
radosgw -f --cluster ceph --name client.rgw.node4 --setuser ceph --
setgroup ceph
```

```
[root@node5 ~]# systemctl status ceph-radosgw@\*
```



修改服务端口

- 登陆node5，RGW默认服务端口为7480，修改为8081或80更方便客户端使用

```
[root@node5 ~]# vim /etc/ceph/ceph.conf
[client.rgw.node5]
host = node5
rgw_frontends = "civetweb port=8081 "
```

//node5为主机名

//civetweb是RGW内置的一个web服务

```
[root@node5 ~]# systemctl restart ceph-radosgw@\*
```



客户端测试

- 这里仅测试RGW是否正常工作
 - 上传、下载数据还需要调用API接口

```
[root@client ~]# curl 192.168.4.15:8081
```

```
<?xml version="1.0" encoding="UTF-8"?><ListAllMyBucketsResult  
xmlns="http://s3.amazonaws.com/doc/  
2006-03-01/"><Owner><ID>anonymous</ID><DisplayName></  
DisplayName></Owner><Buckets></Buckets></ListAllMyBucketsResult>
```



使用第三方软件访问

- 登陆node5 (RGW) 创建账户

```
[root@node5 ~]# radosgw-admin user create \
--uid="testuser" --display-name="First User"
```

```
... ..
```

```
"keys": [
  {
    "user": "testuser",
    "access_key": "5E42OEGb1M95Y49IBG7B",
    "secret_key": "i8YtM8cs7QDCK3rTRopb0TTPBFJvXdEryRbeLGK6"
  }
],
```

```
... ..
```

```
#
```

```
[root@node5 ~]# radosgw-admin user info --uid=testuser
//testuser为用户，key是账户访问密钥
```



使用第三方软件访问（续1）

- 客户端安装软件

```
[root@client ~]# yum install s3cmd-2.0.1-1.el7.noarch.rpm
```

- 配置软件

```
[root@client ~]# s3cmd --configure
Access Key: 5E42OEGB1M95Y49IBG7B
Secret Key: i8YtM8cs7QDCK3rTRopb0TTPBFJVXdEryRbeLGK6
S3 Endpoint [s3.amazonaws.com]: 192.168.4.15:8081
[% (bucket)s3.amazonaws.com]: %(bucket)s.192.168.4.15:8081
Use HTTPS protocol [Yes]: No
Test access with supplied credentials? [Y/n] Y
Save settings? [y/N] y
//注意，其他提示都默认回车
```



使用第三方软件访问（续2）

- 客户端测试

```
[root@client ~]# s3cmd ls
```

- 创建存储数据的bucket（类似于存储数据的目录）

```
[root@client ~]# s3cmd mb s3://my_bucket  
Bucket 's3://my_bucket/' created
```

```
[root@client ~]# s3cmd ls  
2018-05-09 08:14 s3://my_bucket
```

```
[root@client ~]# s3cmd put /var/log/messages s3://my_bucket/log/
```



使用第三方软件访问（续3）

- 客户端测试

```
[root@client ~]# s3cmd ls
```

```
2018-05-09 08:14 s3://my_bucket
```

```
[root@client ~]# s3cmd ls s3://my_bucket
```

```
DIR s3://my_bucket/log/
```

```
[root@client ~]# s3cmd ls s3://my_bucket/log/
```

```
2018-05-09 08:19 309034 s3://my_bucket/log/messages
```

- 测试下载功能

```
[root@client ~]# s3cmd get s3://my_bucket/log/messages /tmp/
```

- 测试删除功能

```
[root@client ~]# s3cmd del s3://my_bucket/log/messages
```



案例3：创建对象存储服务器

- 安装部署Rados Gateway
- 启动RGW服务
- 设置RGW的前端服务与端口
- 客户端测试



总结和答疑
