ONEStor 模块详解之MDS

1

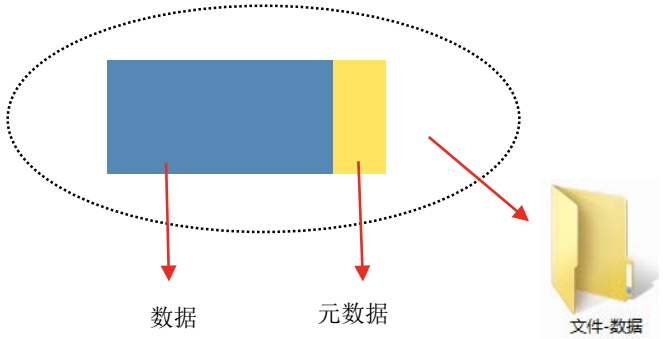
PART 01

第一部分

MDS简介

数据的简介

数据：所有能输入到计算机并被计算机程序处理的符号的介质的总称（文档、图片、视频、表格等）。
元数据：**中介数据**，为描述数据的数据（data about data），主要是描述数据属性（property）的信息，用来支持如指示存储位置、历史数据、资源查找、文件记录等功能。



如果要对一个文件进行读写操作，需要先找到这个文件的存放位置，即先读取文件的元数据。文件的不断更新，对应的元数据也随之改变。

元数据管理

集中式管理和分布式管理

集中式管理：在系统中有一个节点专门司职元数据管理，所有元数据都存储在该节点的存储设备上。所有客户端对文件的请求前，都要先对该元数据管理器请求元数据。

分布式管理：将元数据存放在系统的任意节点并且能动态的迁移。对元数据管理的职责也分布到各个不同的节点上。大多数集群文件系统都采用集中式的元数据管理。

集中式管理实现简单，一致性维护容易。缺点是有单一失效点问题（整个系统的工作流不会因为一个单点的失败而停止整个工作），若该服务器出现故障，整个系统将无法正常工作。而且，当对元数据的操作过于频繁时，集中的元数据管理成为整个系统的性能瓶颈

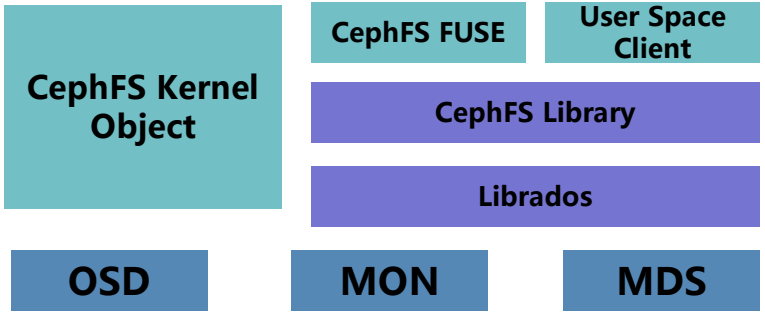
MDS

MDS： Metadata Server 元数据服务器

在CephFS中负责管理元数据，其本质上只是一个**守护进程**。MDS进程本身并不具备持久化存储的功能，而只能依靠**内存**临时记录部分元数据。cephFS的元数据持久化保存在OSD上，用所谓的**元数据池**进行统一管理。在MDS进行主备切换时，原active MDS的内存数据会全部丢失，原standby MDS需要从元数据池读取信息，并在自己内存中重建所有必要的元数据。

元数据的访问占整个文件系统访问的30%~70%左右。所以在文件系统中，MDS的性能与文件系统的性能强相关。

Ceph 文件系统架构



OSD、MON和MDS组成RADOS层，存储数据和元数据；
Kernel Object、FUSE、SpaceClient为客户端接口；
Librados：本地C语言库，提供API支持，允许应用程序直接或者并行访问集群
Library：客户端用它直接与osd交互，提供数据交换接口

文件系统的基本数据类型

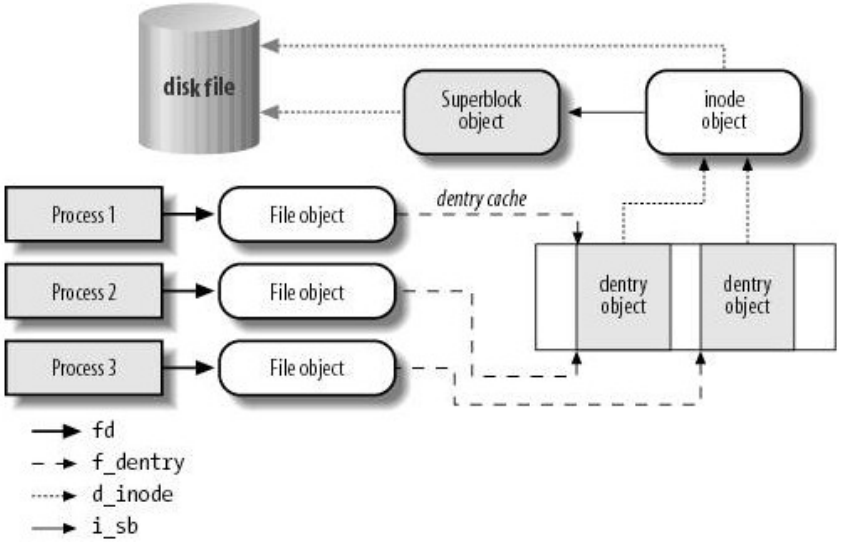
1、SuperBlock：定义文件系统的类型、大小、状态和其他信息。

2、Inode：记录数据块在存储介质上的位置和分布，以及文件对象属性（权限、属性组、数据块信息、时间戳等），不包括文件名和文件内容本身，Inode结构大小固定。

3、Dentry：记录文件在目录树中的位置信息，组成文件系统的目录树，包含了文件名、父目录、子目录、文件的Inode号等信息，连接不同文件对应的inode，包含文件名、文件inode等信息，是连接目录到文件之间的关键。

4、File：文件操作句柄，表示一个打开的文件

数据类型关系



1、每个进程尝试去打开文件，都会建立一个file；同一个进程多次打开同一个文件，也会得到多个file

2、多个file结构可以对应同一个dentry结构。

3、多个dentry对应一个inode

MDS的备份策略

冷备

备份的mds只起到一个进程备份的作用，并不备份元数据。主备进程保持心跳关系，一旦主的mds挂了，备份mds replay元数据到缓存，需要消耗一点时间。

热备

除了进程备份，元数据缓存还时刻与主mds保持同步，当 active mds挂掉后，热备的mds直接变成主mds，并且没有replay的操作，元数据缓存大小和主mds保持一致。

Ceph集群冗余方式

1、主备MDS（一主多备）
2、多主MDS

2

PART 02

第二部分

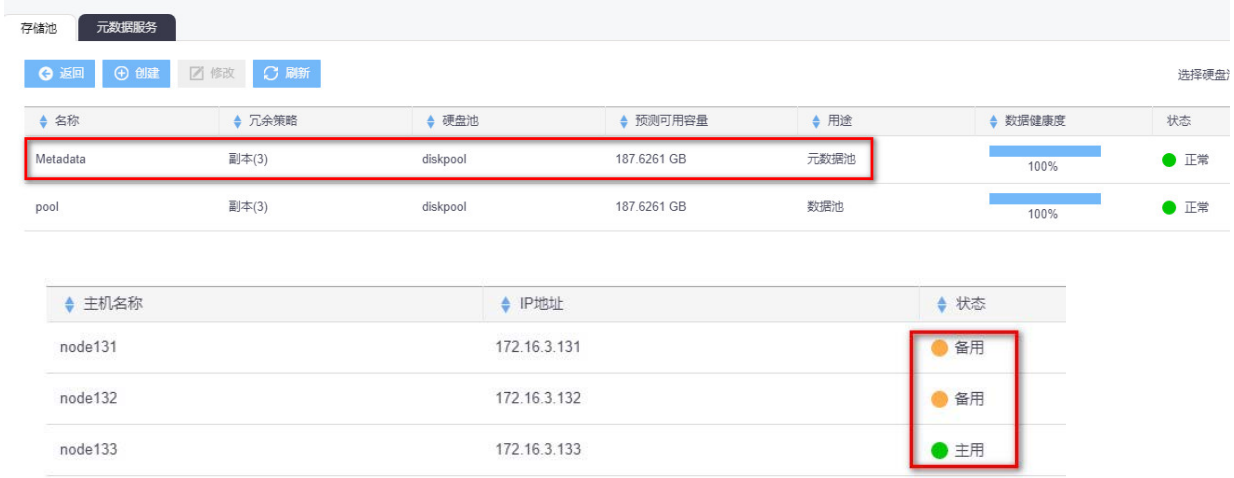
ONEStor中的MDS

ONEStor中的MDS

若服务器有SSD，则建议SSD盘作为元数据盘，服务类型选择文件存储-元数据池，文件存储-数据池选择HDD，采用元数据池分离部署的方式。

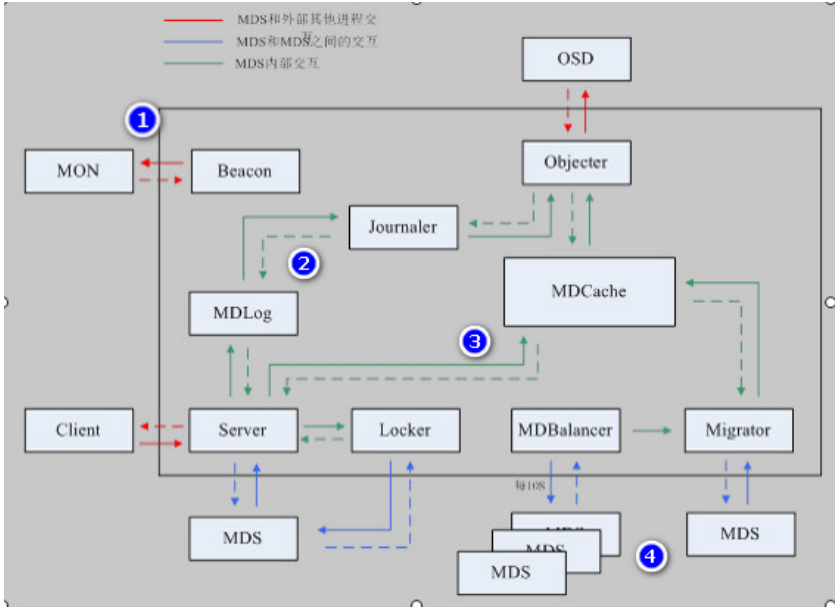
- 元数据池和数据池的冗余策略请按照规划配置；
- 配置向导默认配置所有存储节点为*MDS*节点；
- 元数据池没有纠删码策略，最小为3副本

ONEStor中的MDS



文件系统搭建完毕之后，元数据服务器为多主多备，主备mds的个数取决于集群节点的个数（前提条件：默认集群内所有存储节点都为mds节点）；例如：3节点是1主2备；4节点是2主2备；5节点是2主3备，由系统默认判断生成

MDS与其余组件的交互



Beacon：处理beacon相关逻辑，负责与mon与MDS的状态更新
Objecter：操作RADOS，Object
Journaler：记录元数据操作日志
MDLog：记录文件系统日志
MDcache：MDS内存数据，包含Cinode、Cdir、Cdentry等
Server：处理来自client的大部分文件操作请求
Locker：处理来自client的与锁或权限相关的请求
MDBalancer：多MDS的负载均衡处理
Migrator：子树迁移处理

MDS各种状态简介

1、active：MDS正常运行状态

2、standby：灾备状态，用来接替主挂掉的情况

3、boot：mds在启动期间被广播到monitor

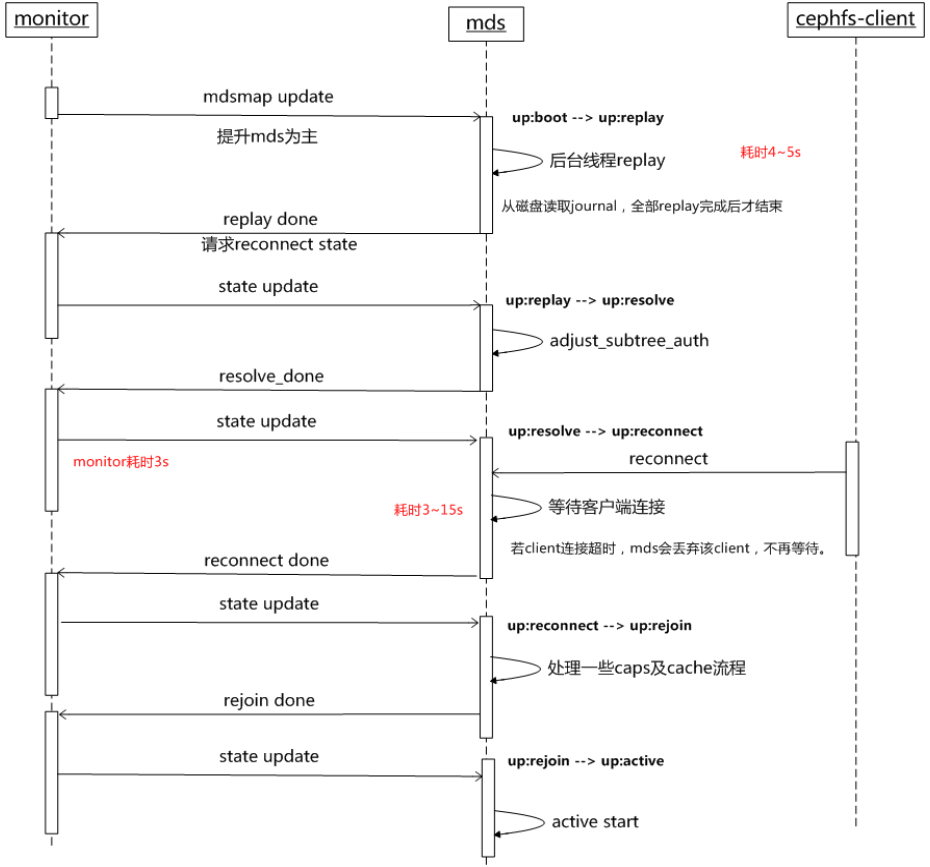
4、replay：日志恢复阶段，将日志内容读入内存后，在内存中进行回放

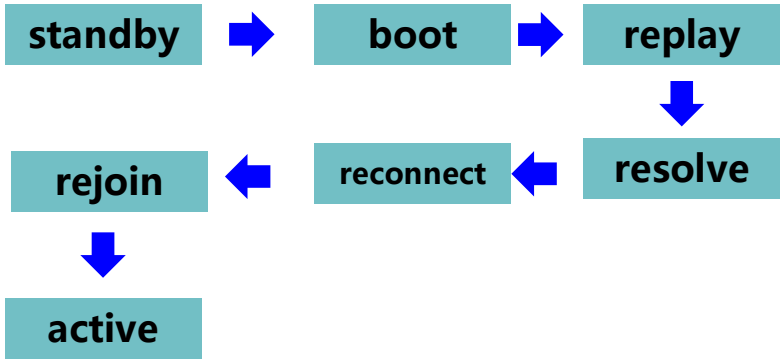
5、resovle：用于解决跨多个mds出现权威元数据分歧场景

6、reconnect：恢复的mds需要与之前的客户端重新建立连接，查询之前客户端发布的文件句柄，重新在mds的缓存中创建一致性功能和锁状态

7、rejoin：将客户端的inode加载到mds cache中

主备MDS切换流程


monitor
mds
cephfs-client
mdsmap update
提升mds为主
up:boot --> up:replay
后台线程replay
耗时4~5s
从磁盘读取journal，全部replay完成后才结束
replay done
请求reconnect state
state update
up:replay --> up:resolve
adjust_subtree_auth
resolve_done
state update
monitor耗时3s
up:resolve --> up:reconnect
reconnect
耗时3~15s
等待客户端连接
若client连接超时，mds会丢弃该client，不再等待。
reconnect done
state update
up:reconnect --> up:rejoin
处理一些caps及cache流程
rejoin done
state update
up:rejoin --> up:active
active start


standby
boot
replay
resolve
reconnect
rejoin
active

MDS状态变化日志打印

```
2018-11-28 13:52:02.376509 7f5fcc3ff700  INFO mds.1.13 handle_mds_map i am now mds.1.13
2018-11-28 13:52:02.376512 7f5fcc3ff700  INFO mds.1.13 handle_mds_map state change up:boot --> up:replay
2018-11-28 13:52:02.376524 7f5fcc3ff700  INFO mds.1.13 replay_start
2018-11-28 13:52:02.376529 7f5fcc3ff700  INFO mds.1.13  recovery set is 0
2018-11-28 13:52:02.376534 7f5fcc3ff700  INFO mds.1.13  waiting for osdmap 243 (which blacklists prior instance)
2018-11-28 13:52:02.445223 7f5fc17ff700 WARNING mds.1.cache creating system inode with ino:0x101
2018-11-28 13:52:02.445403 7f5fc17ff700 WARNING mds.1.cache creating system inode with ino:0x1
2018-11-28 13:52:28.222059 7f5fc27f9700 WARNING  mds1:WormClock::decode_worm_clock_info 216 decode_worm_clock_info Worm clock is not set
2018-11-28 13:52:28.232961 7f5fc27f9700 WARNING mds.1.13 decode_quota_info decode _quota_set success
2018-11-28 13:52:28.232970 7f5fc27f9700  INFO mds.1.13 replay_done
2018-11-28 13:52:28.232973 7f5fc27f9700  INFO mds.1.13 making mds journal writeable
2018-11-28 13:52:28.233185 7f5fc27f9700  INFO mds.1.13  recovery set is 0
2018-11-28 13:52:28.363169 7f5fcc3ff700  INFO mds.1.13 handle_mds_map i am now mds.1.13
2018-11-28 13:52:28.363173 7f5fcc3ff700  INFO mds.1.13 handle_mds_map state change up:replay --> up:resolve
2018-11-28 13:52:28.363187 7f5fcc3ff700  INFO mds.1.13 resolve_start
2018-11-28 13:52:28.363189 7f5fcc3ff700  INFO mds.1.13 reopen_log
2018-11-28 13:52:28.363253 7f5fcc3ff700  INFO mds.1.13 now recovery set is 0
2018-11-28 13:52:28.363261 7f5fcc3ff700  INFO mds.1.13  recovery set is 0
2018-11-28 13:52:28.363862 7f5fd0bfe700 WARNING -- 172.110.6.21:6824/3557770486 >> 172.110.6.24:6824/1991823167 conn(0x7f5fcf40e000 :6824 s=STATE_ACCEPTING_WAIT_CONNECT_MSG_AUTH pgs=0 cs=0 l=0).handle_connect_msg accept connect_seq 0 vs existing csq=0 existing_state=STATE_CONNECTING_WAIT_CONNECT_REPLY
2018-11-28 13:52:28.374716 7f5fcc3ff700  INFO mds.1.13 resolve_done
2018-11-28 13:52:29.366844 7f5fcc3ff700  INFO mds.1.13 handle_mds_map i am now mds.1.13
2018-11-28 13:52:29.366847 7f5fcc3ff700  INFO mds.1.13 handle_mds_map state change up:resolve --> up:reconnect
2018-11-28 13:52:29.366857 7f5fcc3ff700  INFO mds.1.13 reconnect_start
2018-11-28 13:52:29.366873 7f5fcc3ff700  INFO mds.1.server reconnect_clients -- 4 sessions
2018-11-28 13:52:29.367434 7f5fcc3ff700 WARNING log_channel(cluster) log [DBG] : reconnect by client.11600 172.110.6.21:0/2785747455 after 0.000505
2018-11-28 13:52:29.367636 7f5fcc3ff700 WARNING log_channel(cluster) log [DBG] : reconnect by client.10718 172.110.6.20:0/1471566747 after 0.000743
2018-11-28 13:52:29.367821 7f5fcc3ff700 WARNING log_channel(cluster) log [DBG] : reconnect by client.70379 172.110.6.20:0/1991406755 after 0.000920
2018-11-28 13:52:29.387506 7f5fcc3ff700 WARNING log_channel(cluster) log [DBG] : reconnect by client.12434 172.110.6.22:0/589559399 after 0.020610
2018-11-28 13:52:29.388562 7f5fcc3ff700  INFO mds.1.13 reconnect_done
2018-11-28 13:52:30.370529 7f5fcc3ff700  INFO mds.1.13 handle_mds_map i am now mds.1.13
2018-11-28 13:52:30.370538 7f5fcc3ff700  INFO mds.1.13 handle_mds_map state change up:reconnect --> up:rejoin
2018-11-28 13:52:30.370553 7f5fcc3ff700  INFO mds.1.13 rejoin_start
2018-11-28 13:52:30.371952 7f5fcc3ff700  INFO mds.1.13 now recovery set is 0
2018-11-28 13:52:30.371961 7f5fcc3ff700  INFO mds.1.13 rejoin_joint_start
```

```
2018-11-28 13:52:30.371962 7f5fcc3ff700  INFO mds.1.cache rejoin_send_rejoins begin, resend_bit: 0, rejoin_gather: (0), rejoin_ack_gather: (), rejoin_sent: (), rejoin_ack_sent: ().
2018-11-28 13:52:30.373847 7f5fcc3ff700  INFO mds.1.cache rejoin_send_rejoins,  rejoin message sent to mds:0
2018-11-28 13:52:30.373851 7f5fcc3ff700  INFO mds.1.cache rejoin_send_rejoins done, resend_bit: 0, rejoin_gather: (0), rejoin_ack_gather: (0), rejoin_sent: (0), rejoin_ack_sent: (), for_resend_rejoin: ().
2018-11-28 13:52:30.373896 7f5fcc3ff700  INFO mds.1.cache handle_cache_rejoin_strong begin, receive rejoin_strong from: 0, rejoin_gather: (0), rejoin_ack_gather: (0,1).
2018-11-28 13:52:30.374845 7f5fcc3ff700  INFO mds.1.cache handle_cache_rejoin_strong done, no need to gather rejoin and rejoin ack, rejoin_gather: (), rejoin_ack_gather: (0,1)
2018-11-28 13:52:30.378484 7f5fcc3ff700  INFO mds.1.cache handle_cache_rejoin_ack form mds.0
2018-11-28 13:52:30.379388 7f5fcc3ff700  INFO mds.1.cache handle_cache_rejoin_ack done, still need to gather rejoin or rejoin ack, rejoin_gather: (), rejoin_ack_gather (1), rejoin_sent: (0), rejoin_ack_sent: ().
2018-11-28 13:52:30.717028 7f5fc27f9700  INFO mds.1.cache rejoin_send_acks,  recovery_set: 0, rejoin_ack_sent: , need_resend_rejoin_ack:
2018-11-28 13:52:30.719953 7f5fc27f9700  INFO mds.1.cache rejoin_send_acks,  rejoin ack message sent to mds:0
2018-11-28 13:52:30.720003 7f5fc27f9700  INFO mds.1.13 rejoin_done
2018-11-28 13:52:30.720071 7f5fc27f9700  INFO mds.1.cache rejoin_gather_finish done, after rejoin_send_acks, rejoin_gather: (), rejoin_ack_gather: (), rejoin_sent: (0), rejoin_ack_sent: (0).
2018-11-28 13:52:31.243895 7f5fc93fe700  INFO mds.1.13 may_need_resend_rejoins, now start to check rejoin.
2018-11-28 13:52:31.374476 7f5fcc3ff700  INFO mds.1.13 handle_mds_map i am now mds.1.13
2018-11-28 13:52:31.374480 7f5fcc3ff700  INFO mds.1.13 handle_mds_map state change up:rejoin --> up:active
2018-11-28 13:52:31.374490 7f5fcc3ff700  INFO mds.1.13 recovery_done -- successful recovery!
2018-11-28 13:52:31.374671 7f5fcc3ff700  INFO mds.1.13 active_start
2018-11-28 13:52:31.511262 7f5fcc3ff700 WARNING mds.1.cache add_ino_to_worm_list totally add 0 files to worm list
2018-11-28 13:52:31.511298 7f5fcc3ff700  INFO mds.1.13 cluster recovered.
```

文件系统数据分区方法

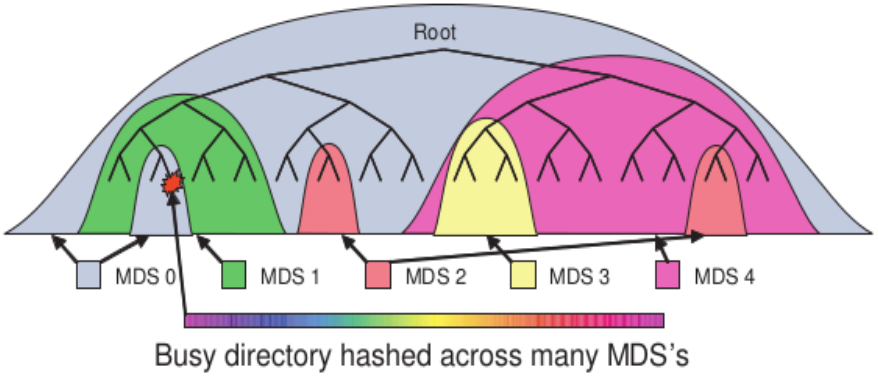
目的：实现文件系统元数据、数据负载均衡

1、静态子树分区
通过手工分区方式，将数据直接分配到某个服务节点上，出现负载不均衡时，由管理员手动重新进行分配。
缺点：只适应于数据位置固定的场景，不适合动态扩展

2、hash计算分区法
通过hash计算来分配数据存储位置。适合数据分布均衡、且需要应用各种异常的场景。
缺点：不适合数据分布固定、环境变化频率很高的场景，不适合动态扩展（元数据访问频率不同，热点信息容易频繁变化）

3、动态子树分区
通过实时监控集群节点负载，动态调整子树分布于不同的节点，适合各种异常场景，特别适用于少量元数据迁移场景。
缺点：不太适合数据大量迁移场景，大量数据迁移会导致性能较低。

动态子树分区

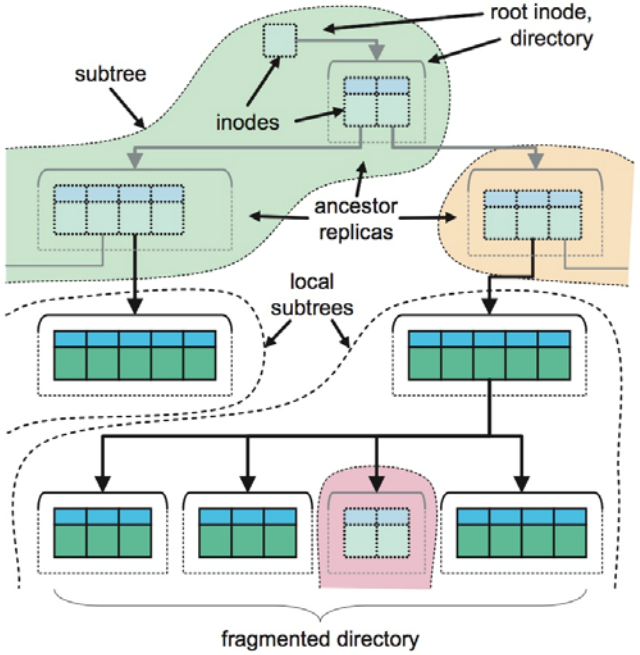


- 初始状态由mds.0管理所有元数据
- 基于当前的工作负载来动态映射目录层级子树到其他元数据服务器
- 每个目录变为热点时会在多个节点产生副本
- 客户端将缓存“目录-mds”映射关系，直接与相关mds通讯

目录不是负载均衡的最小单位，目录分片才是

动态子树分区

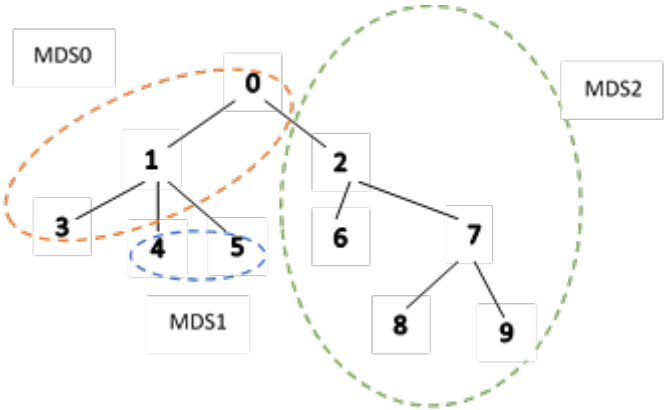
- 元数据被划分为由目录片段为单位的子树。
- 每个MDS复制本地管理的子树的父元数据。
- 大的目录将被分成多个片段，然后可以形成嵌套子树。
- 每个MDS只知道自己的缓存中的元数据的权限。
- 客户端会缓存子树边界，方便元数据请求。



动态子树迁移

目的：实现文件系统数据的负载均衡
实时监控集群节点的负载，动态调整子树分布于不同节点，实现负载均衡

子树迁移：初始状态由mds.0管理所有元数据，如果当前mds的负载过重，就会有动态子树迁移将当前mds所管理的一些子树（目录树信息）迁移到其他的mds上，这样可以分担当前mds的负载。Ceph使用动态子树迁移实现横向扩展。适用于**少量**元数据迁移的场景



3

PART 03

第三部分

常见MDS问题分析

MDS状态震荡

故障现场：MDS状态一直在反复震荡，停止/主用/备用等反复切换，在Handy界面上手动启动或停止MDS，提示连接失败
日志：/var/log/ceph/ceph-mds.mds*.log

ceph-mds.mds*.log日志里面打印mds状态从standby/reconnect/active等反复切换

原因排查：可能是网络震荡导致mds状态异常，可以查看messages、ceph_net、THEMIS日志，看是否有网络相关打印，MDS通过存储前端网与mon维持心跳，通信异常导致mds状态异常

```
Line 5808: Oct 13 04:34:21 cvknode10 kernel: [   30.218580] cnic: QLogic cnic Driver v2.5.20w (Feb 22, 2018)
Line 5808: Oct 13 04:34:21 cvknode10 kernel: [   30.218580] cnic: QLogic cnic Driver v2.5.20w (Feb 22, 2018)
Line 5842: Oct 13 04:47:00 cvknode10 kernel: [  789.168600] i40e 0000:3d:00.1 eth7: NIC Link is Down
Line 5844: Oct 13 04:47:07 cvknode10 kernel: [  796.104706] i40e 0000:3d:00.1 eth7: NIC Link is Up, 1000 Mbps Full Duplex, Flow Control: None
Line 6189: Oct 13 06:28:36 cvknode10 kernel: [ 6885.364507] ixgbe 0000:b0:00.1 eth5: NIC Link is Down
Line 6191: Oct 13 06:28:52 cvknode10 kernel: [ 6901.358371] bnx2x 0000:af:00.0 eth2: NIC Link is Down
Line 6193: Oct 13 06:30:50 cvknode10 kernel: [ 7019.781502] bnx2x 0000:af:00.0 eth2: NIC Link is Up, 10000 Mbps full duplex, Flow control: ON - receive & transmit
Line 6196: Oct 13 06:30:58 cvknode10 kernel: [ 7026.973522] ixgbe 0000:b0:00.1 eth5: NIC Link is Up 10 Gbps, Flow Control: RX/TX
Line 6201: Oct 13 06:32:08 cvknode10 kernel: [ 7097.872703] bnx2x 0000:86:00.1 eth1: NIC Link is Down
Line 6202: Oct 13 06:32:09 cvknode10 kernel: [ 7097.940691] ixgbe 0000:b0:00.0 eth4: NIC Link is Down
Line 6205: Oct 13 06:32:31 cvknode10 kernel: [ 7120.044462] bnx2x 0000:86:00.1 eth1: NIC Link is Up, 10000 Mbps full duplex, Flow control: ON - receive & transmit
Line 6206: Oct 13 06:32:31 cvknode10 kernel: [ 7120.191545] ixgbe 0000:b0:00.0 eth4: NIC Link is Up 10 Gbps, Flow Control: RX/TX
```

IP冲突引起MDS状态震荡

故障现场：IP冲突引起MDS状态一直在反复震荡，停止/主用/备用等反复切换，在Handy界面上手动启动或停止MDS，提示连接失败
日志：/var/log/ceph/ceph-mds.mds*.log

解决办法：修复IP冲突环境，在受IP冲突的MDS节点执行ip neigh flush dev *name*（*name*是冲突IP对应的网口）

子树迁移

故障现场：业务压力较大的情况下客户侧感知业务有卡顿，但是集群健康度OK

日志：/var/log/ceph/ceph-mds.mds*.log

cat /ceph-mds.mds*.log | grep “===“

```
2018-11-28 12:15:46.466275 7f39813ff700  INFO mds.0.server handle_client_mkdir balancer->smart_pin: -1 smart_pin_depths [3] ino: 0x100000003eb path vdb.1_1.dir new1->inode.depth 3 map to rank: 1
2018-11-28 12:15:51.278325 7f39813ff700 WARNING mds.0.bal   mds.0 mdsload<[3.34692,1374.58 2752.5]/[3.34692,1374.58 2752.5], req 0, hr 0, qlen 0, cpu 1.47, subtree 2> = 2752.5 ~ 2752.5
2018-11-28 12:15:51.278346 7f39813ff700 WARNING mds.0.bal   mds.1 mdsload<[0,0 0]/[0,0 0], req 0, hr 0, qlen 0, cpu 0.69, subtree 2> = 0 ~ 0
2018-11-28 12:15:52.052251 7f39813ff700  INFO mds.0.migrator   ===>  sending MExportDirDiscover on [dir 0x100000003eb /data/share/vdb.1_1.dir/ ] to mds.1
2018-11-28 12:15:52.056514 7f39813ff700  INFO mds.0.migrator   ===>  handle_export_discover_ack from mds.1 on [dir 0x100000003eb /data/share/vdb.1_1.dir/ ]
2018-11-28 12:15:52.224116 7f39813ff700  INFO mds.0.migrator   ===>  export_dir couldn't acquire all needed locks, failing. [dir 0x100000003eb /data/share/vdb.1_1.dir/ ]
2018-11-28 12:15:52.224122 7f39813ff700  INFO mds.0.migrator export_try_cancel [dir 0x100000003eb /data/share/vdb.1_1.dir/ ]
2018-11-28 12:15:52.224125 7f39813ff700  INFO mds.0.migrator export state=freezing : canceling freeze
2018-11-28 12:15:56.274557 7f39813ff700  INFO mds.0.migrator   ===>  sending MExportDirDiscover on [dir 0x100000003eb /data/share/vdb.1_1.dir/ ] to mds.1
2018-11-28 12:15:56.275664 7f39813ff700  INFO mds.0.migrator   ===>  handle_export_discover_ack from mds.1 on [dir 0x100000003eb /data/share/vdb.1_1.dir/ ]
2018-11-28 12:15:56.278494 7f397a3fe700  INFO mds.0.migrator   ===>  sending MExportDirPrep on [dir 0x100000003eb /data/share/vdb.1_1.dir/ ] to mds.1
2018-11-28 12:15:56.280027 7f39813ff700  INFO mds.0.migrator   ===>  handle_export_prep_ack from mds.1 on [dir 0x100000003eb /data/share/vdb.1_1.dir/ ]
2018-11-28 12:15:56.364249 7f3977bf9700  INFO mds.0.migrator   ===>  sending MExportDir on [dir 0x100000003eb /data/share/vdb.1_1.dir/ ] to mds.1
2018-11-28 12:15:56.499790 7f39813ff700  INFO mds.0.migrator   ===>  handle_export_ack from mds.1 on [dir 0x100000003eb /data/share/vdb.1_1.dir/ ]
2018-11-28 12:15:56.503847 7f3977bf9700  INFO mds.0.migrator   ===>  sending MExportDirFinish (last==true) on [dir 0x100000003eb /data/share/vdb.1_1.dir/ ] to mds.1
2018-11-28 12:16:01.272910 7f39813ff700 WARNING mds.0.bal   mds.0 mdsload<[232.474,0 232.474]/[232.474,0 232.474], req 0, hr 0, qlen 0, cpu 1.48, subtree 3> = 232.474 ~ 232.474
2018-11-28 12:16:01.272926 7f39813ff700 WARNING mds.0.bal   mds.1 mdsload<[0,572.533 1145.07]/[0,572.533 1145.07], req 0, hr 0, qlen 0, cpu 0.59, subtree 3> = 1145.07 ~ 1145.07
```

方法：将目录pin住，不让子树进行频繁迁移

Thanks !

新华三集团
www.h3c.com