Curve

raft在Curve存储中的工程实践

D I G I T A L S A I L

陈威

Curve Maintaner

网易资深服务端开发工程师

Curve

DIGITAL SAIL

目录

01 **Curve介绍**

项目背景 | Curve架构 | 使用场景 | Curve社区

02 **raft和braft**

raft协议介绍 | braft介绍

03 **raft在Curve中的应用**

raft in Curve块存储 | raft in curve文件存储 | 配置变更

04 **Curve对raft的优化**

优化点1 | 优化点2

05 **Q&A**

答疑

项目背景

Curve

Curve是一个 **高性能、更稳定、易运维** 的 **云原生** 分布式存储系统，支持 **块存储** 和 **文件存储**

Curve块存储和文件存储均采用raft协议

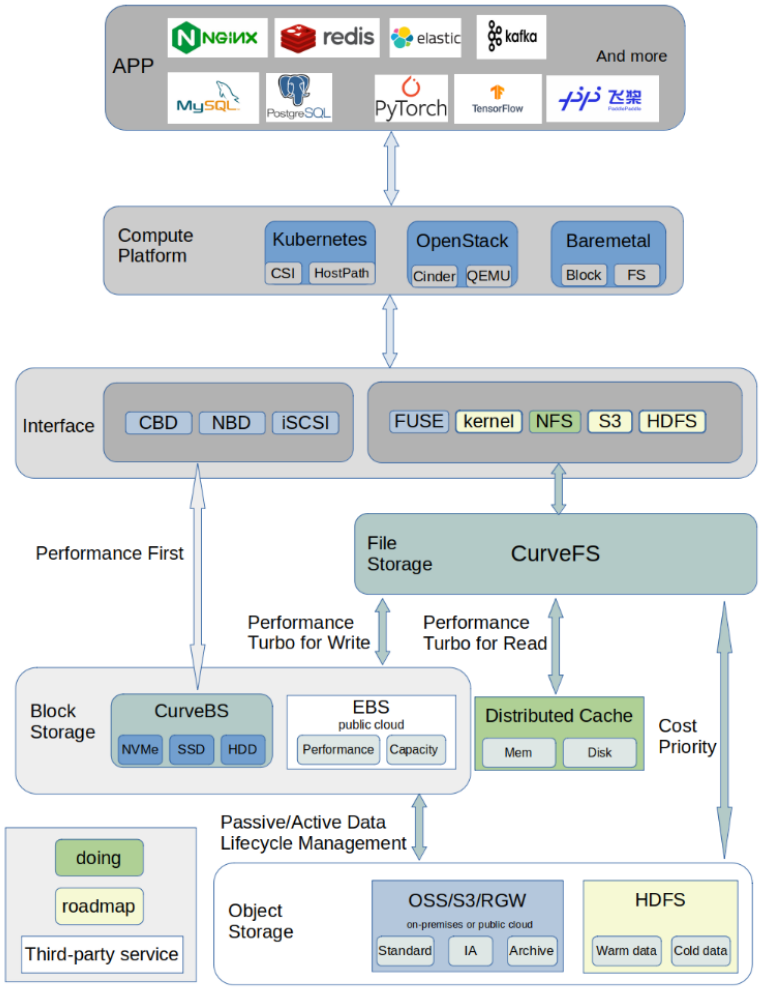
2018~2021 Curve块存储

- 基于Openstack构建云计算平台
- 底层存储使用Ceph块存储
- 稳定性挑战

2021~2022 Curve文件存储

- 算力平台kubernetes的迅速发展
- AI/大数据业务的快速增长
- 存储使用Ceph文件存储/HDFS
- 成本/性能挑战

整体架构



- 对接OpenStack平台为云主机提供高性能块存储服务
- 对接Kubernetes为其提供RWO、RWX等类型的持久化存储卷
- 对接PolarFS作为云原生数据库的高性能存储底座，完美支持云原生数据库的存算分离架构
- Curve作为云存储中间件使用S3兼容的对象存储作为数据存储引擎，为公有云用户提供高性价比的共享文件存储
- 支持在物理机上挂载使用块设备或FUSE文件系统

开源社区

Curve

目标

方法

影响力

社区运营

降本

生态共建　开源共建

获客

源码兜底　技术领先

网易游戏　有道 youdao

LOFTER　XFUSION

网易严选　创云融达 ORIGINALITY CLOUD

网易云信　万方电子 WANFANG ELECTRONIC

网易云音乐　SmartMore 思谋科技

用户

Contributors 36

+ 25 contributors

开发者

操作系统：OpenAnolis 龙蜥社区；OpenEuler

芯片：Kunpeng；Phytium 飞腾；HYGON 中科海光

数据库：PolarDB；MySQL；PostgreSQL；TDengine

云原生：openstack；kubernetes；ZStack；esage 易鸥捷

AI训练：TensorFlow；PyTorch；飞桨

大数据：elastic；kafka

KYLIN 麒麟；长江存储 YANGTZE MEMORY；PLiOPS

社区生态

Curve

DIGITAL SAIL

目录

01 **Curve介绍**

项目背景 | Curve架构 | 使用场景 | Curve社区

02 **raft和braft**

raft协议介绍 | braft介绍

03 **raft在Curve中的应用**

raft in Curve块存储 | raft in curve文件存储 | 配置变更

04 **Curve对raft的优化**

优化点1 | 优化点2

05 **Q&A**

答疑

RAFT协议简介

Curve

什么是raft

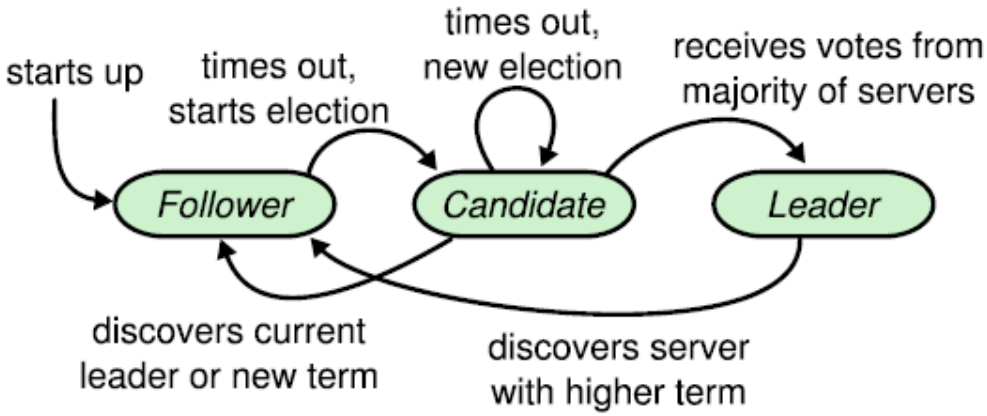
- raft 是一种新型易于理解的分布式一致性复制协议，由斯坦福大学的Diego Ongaro和John Ousterhout提出，《In Search of an Understandable Consensus Algorithm(Extended Version)》
- raft 是一种Leader-Based的Multi-Paxos变种，提供了更完整更清晰的协议描述，更容易理解和实现。
- raft可以解决分布式理论中的CP，即一致性和分区容忍性
- 大多数副本成功即可返回成功
- 速度取决于写的较快的大多数

RAFT协议简介

Curve

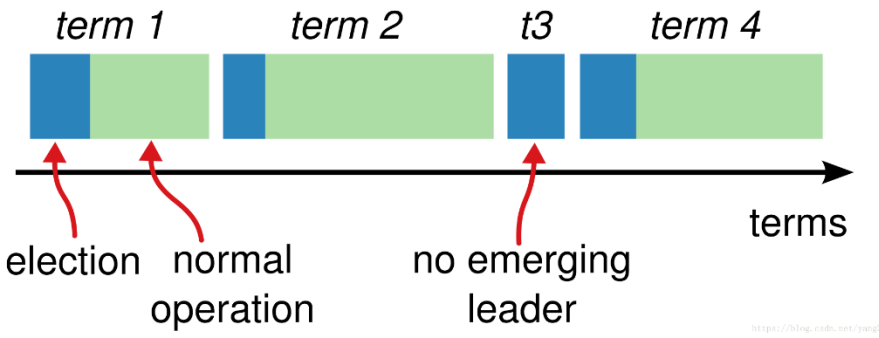
raft选举leader

- **Leader:** 负责从客户端接受日志，把日志复制到其他服务器，当保证安全性的时候告诉其他服务器应用日志条目到他们的状态机中。
- **Candidate:** 发起选举。获取大多数选票的候选人将成为领导者。
- **Follower:** 响应来自其他服务器的请求，如果接受不到消息，就变成候选人并发起一次选举。



raft任期

- 时间被划分成一个个的任期，每个任期开始都是一次选举。
- 选举成功，领导人会管理整个集群直到任期结束。
- 选举失败，这个任期就会没有领导人而结束。



RAFT协议简介

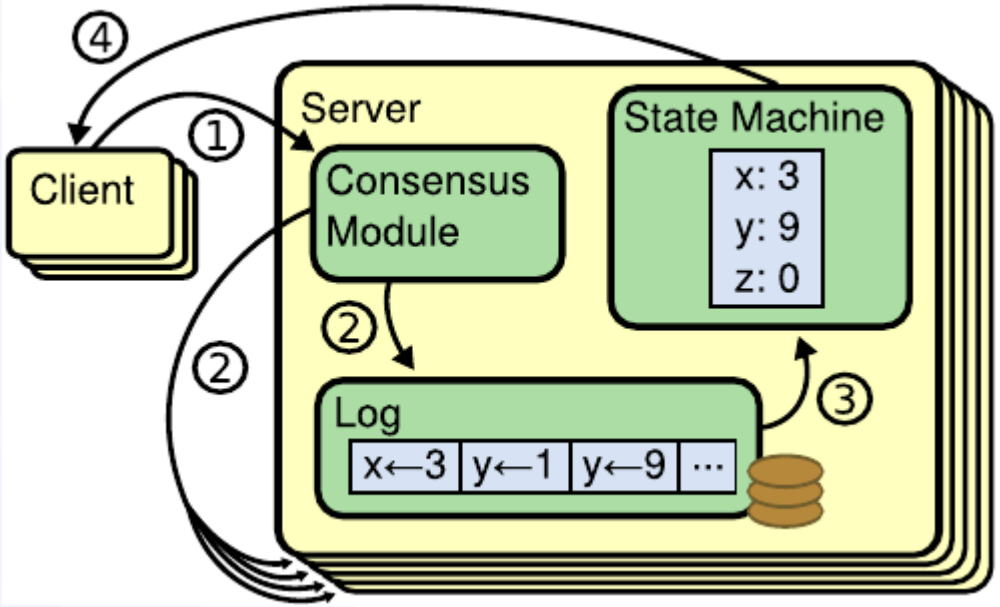
Curve

raft复制状态机

- 提供命令在多个节点之间有序复制和执行，当多个节点初始状态一致的时候，保证节点之间状态一致。

raft日志复制

1. leader收到客户端的请求。
2. leader把请求指令记录下来，写入日志，然后并行发给其他的服务器，让他们复制这条日志。
3. 当这条日志条目被安全的复制，leader会应用这条日志条目到它的状态机中。
4. 然后把执行的结果返回给客户端。



RAFT协议简介

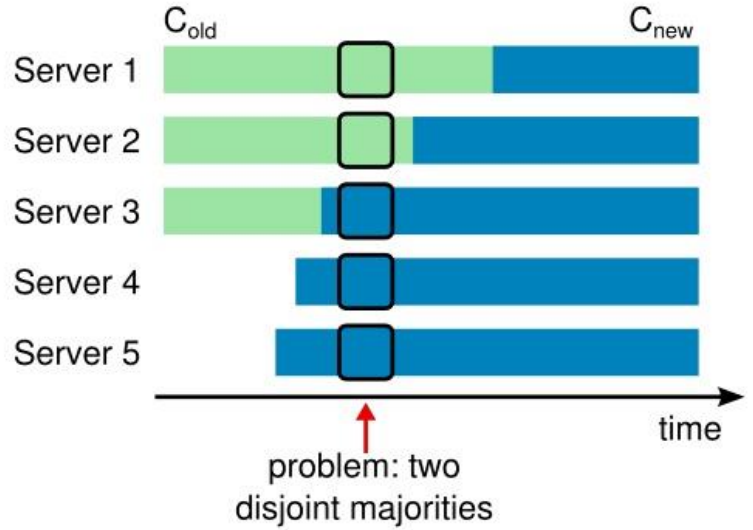
Curve

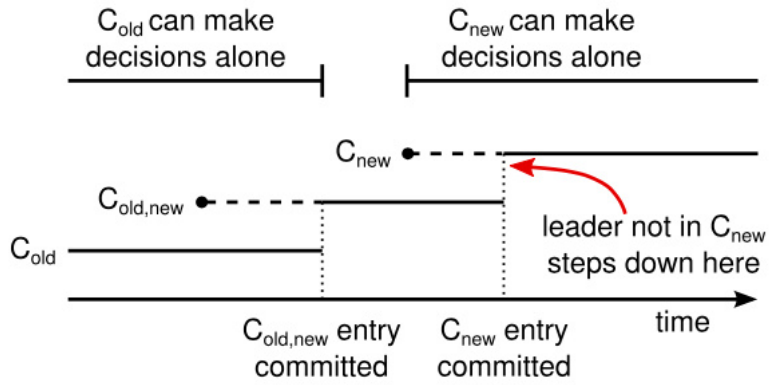
raft配置变更

- 配置：加入一致性算法的服务器集合。
- 集群的配置不可避免会发生变更，比如替换宕机的机器。

直接配置变更可能出现双主问题

- 共同一致（joint consensus）
- 集群先切换到一个过渡的配置(old + new)，一旦共同一致已经被提交，系统切换到新的配置(new)。



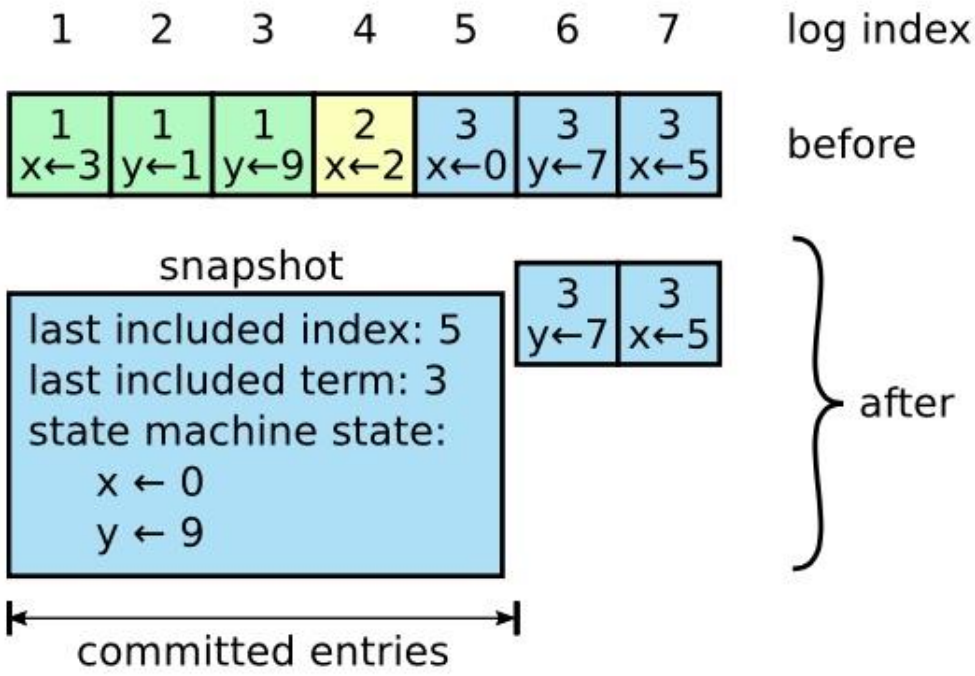


RAFT协议简介

Curve

日志压缩

- 日志会不断增长，占用空间
- 采用快照的方式压缩日志
- 在某个时间点，整个系统的状态都以快照的形式写入到稳定的持久化存储中
- 完成一次快照之后，删除时间点之前的所有日志和快照。



BRAFT简介

Curve

什么是braft

- raft协议提出之后，涌现出了非常多的实现，比如etcd，braft，tikv等。
- braft是raft的一个实现，实现了raft的一致性协议和复制状态机，而且提供了一种通用的基础库。基于braft，可以基于自己的业务逻辑构建自己的分布式系统。
- braft本身不提供server功能，需要业务自己实现状态机。

braft 接口

Node（一个raft实例）

```
int init(const NodeOptions& options);
void apply(const Task& task);
void add_peer(const PeerId& peer, Closure* done);
void remove_peer(const PeerId& peer, Closure* done);
void change_peers(const Configuration& new_peers,
Closure* done);
```

StateMachine

```
void on_apply(::raft::Iterator& iter);
void on_snapshot_save(SnapshotWriter* writer,
                      Closure* done);
int on_snapshot_load(SnapshotReader* reader);
void on_leader_start(int64_t term);
void on_leader_stop(const butil::Status& status);
void on_error(const Error& e);
```

Curve

DIGITAL SAIL

目录

01 **Curve介绍**

项目背景 | Curve架构 | 使用场景 | Curve社区

02 **raft和braft**

raft协议介绍 | braft介绍

03 **raft在Curve中的应用**

raft in Curve块存储 | raft in curve文件存储 | 配置变更

04 **Curve对raft的优化**

优化点1 | 优化点2

05 **Q&A**

答疑

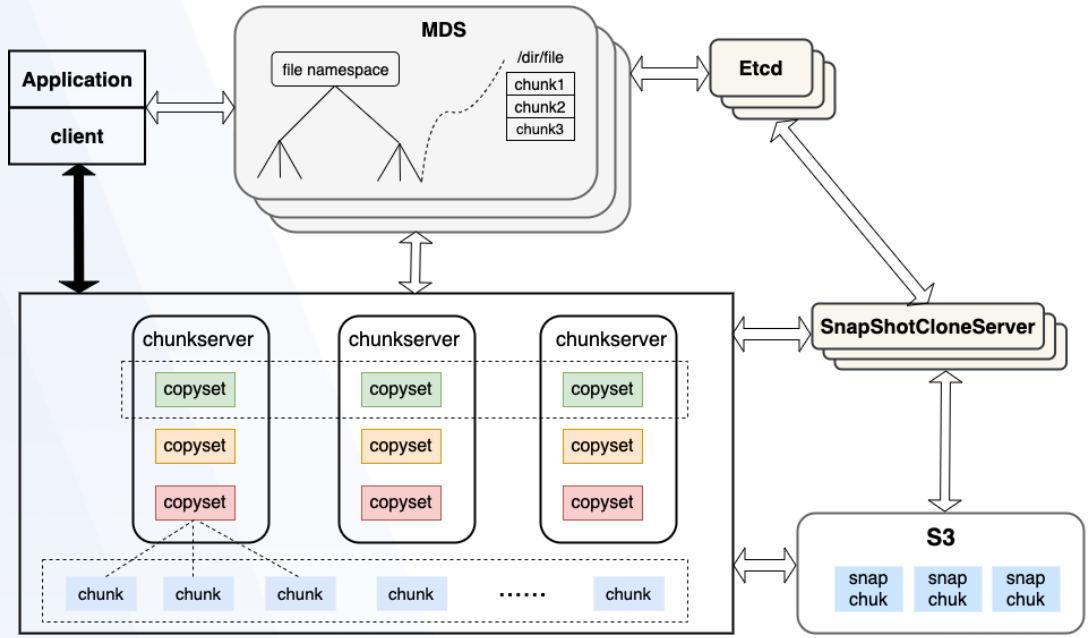
Curve块存储RAFT应用

Curve块存储

- 高性能、更稳定、易运维
- 支持NBD(network block device)、iscsi
- 支持RDMA和SPDK

Curve块存储架构

- client：接受用户请求。
- mds：保存元数据，包括topo信息、块设备信息、数据分布信息等，持久化到etcd中。
- **chunkserver**：采用**raft协议**3副本的方式保存块设备上的数据。
- snapshotCloneServer：卷的快照克隆服务，持久化到S3中。



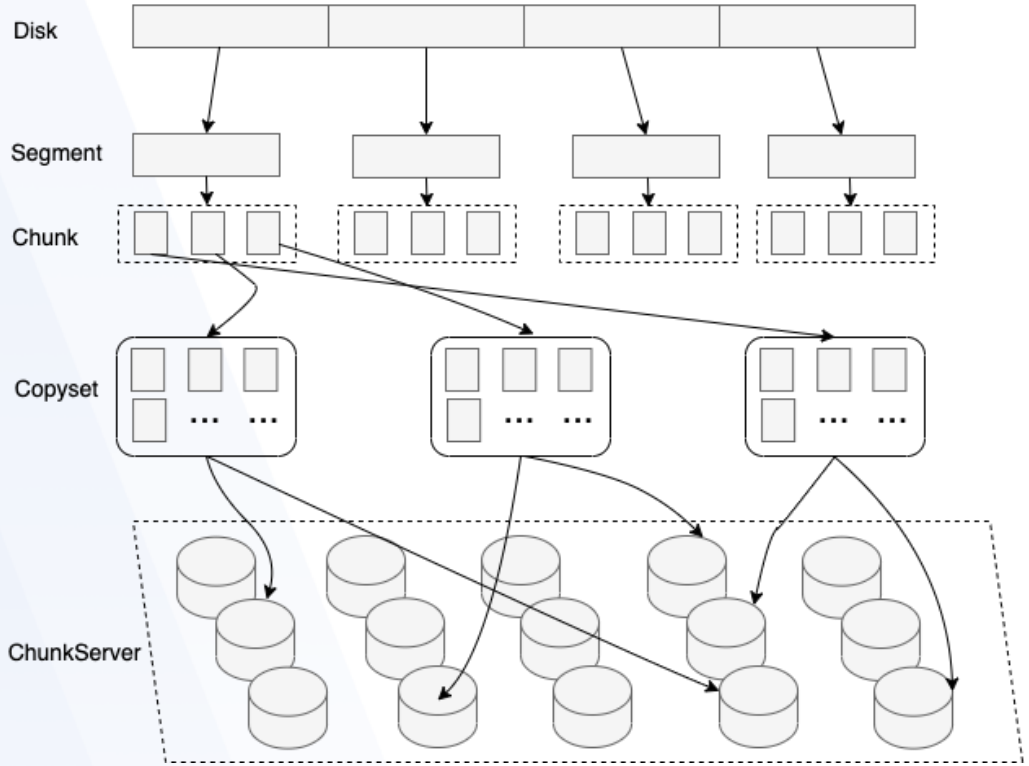
Curve块存储RAFT应用

数据分布

- disk -> segment -> chunk
- chunk -> copyset -> chunk in 3 chunkserver

raft复制组

- 每个raft实例用一个copyset管理，copyset是个逻辑概念。写入chunk的数据，由copyset对应的raft完成3副本的写入。
- multi-raft：copyset和chunkserver是多对多的关系
  - 每个copyset由3个chunkserver组成
  - 每个chunkserver可以服务多个copyset

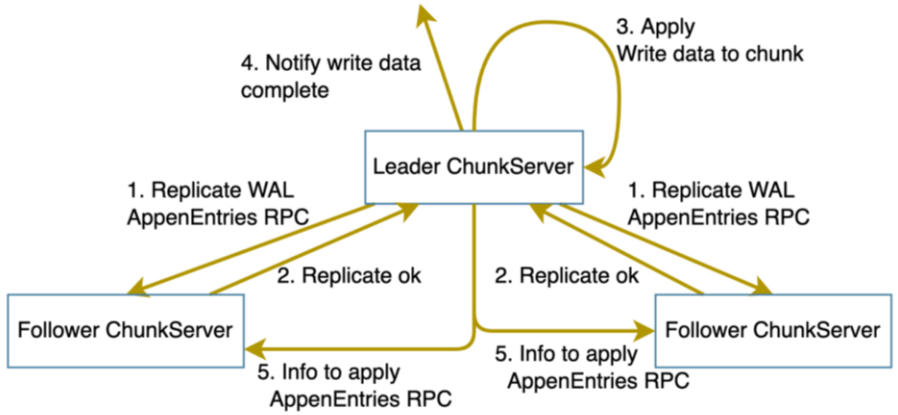


Curve块存储RAFT应用

请求处理流程

以写请求为例：

1. Client 发送写请求给 Leader ChunkServer。
2. ChunkServer 收到请求，将请求封装成一个 log entry，提交给 raft。
3. raft模块在本地持久化 entry 的同时发送 entry 给其他副本（ChunkServer）。
4. 本地持久化 log entry 成功，且另一个副本也写入 log entry 成功则 commit。
5. commit 后执行 apply，apply 把数据写入chunk。
6. 返回client写入成功。



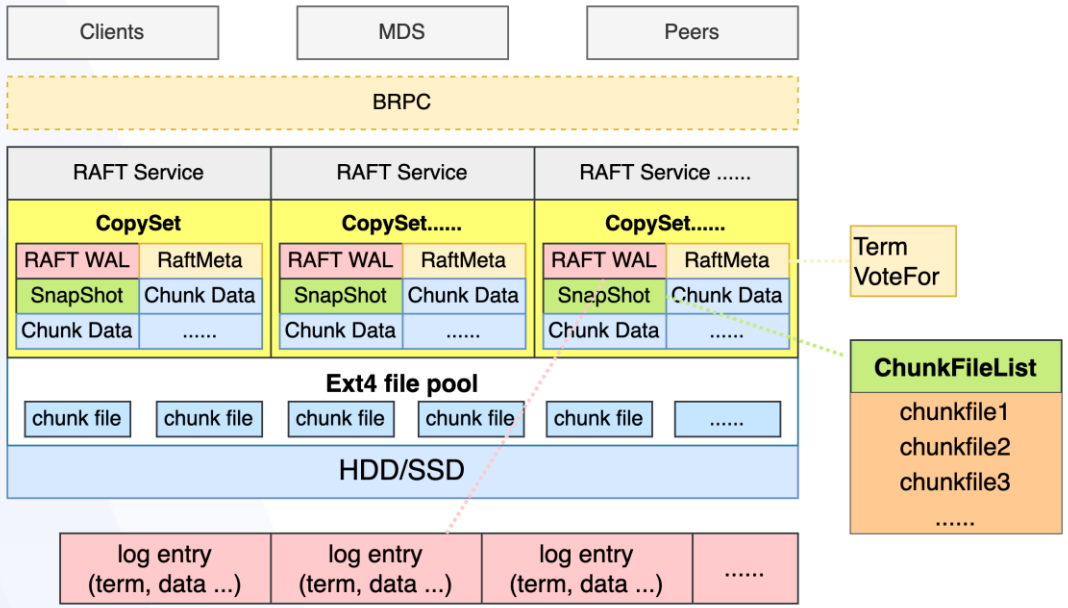
Curve块存储RAFT应用

raft apply

- 用户数据的写入最终转化为对chunk的写入。
- raft的apply，直接在对应的chunk上写入数据。

raft snapshot

- 由于chunk可以覆盖写，所以chunk的写入天然具有幂等性。
- 对chunk打快照不需要把chunk重新换个地方复制一遍，只需要记录下chunk文件的list。
- follower从快照恢复，只需要leader把最近一次快照涉及到chunk数据给到follower，follower再从上次快照后的日志重放即可。
- chunkserver服务重启，只需要加载快照，然后对chunk重放快照之后日志即可。



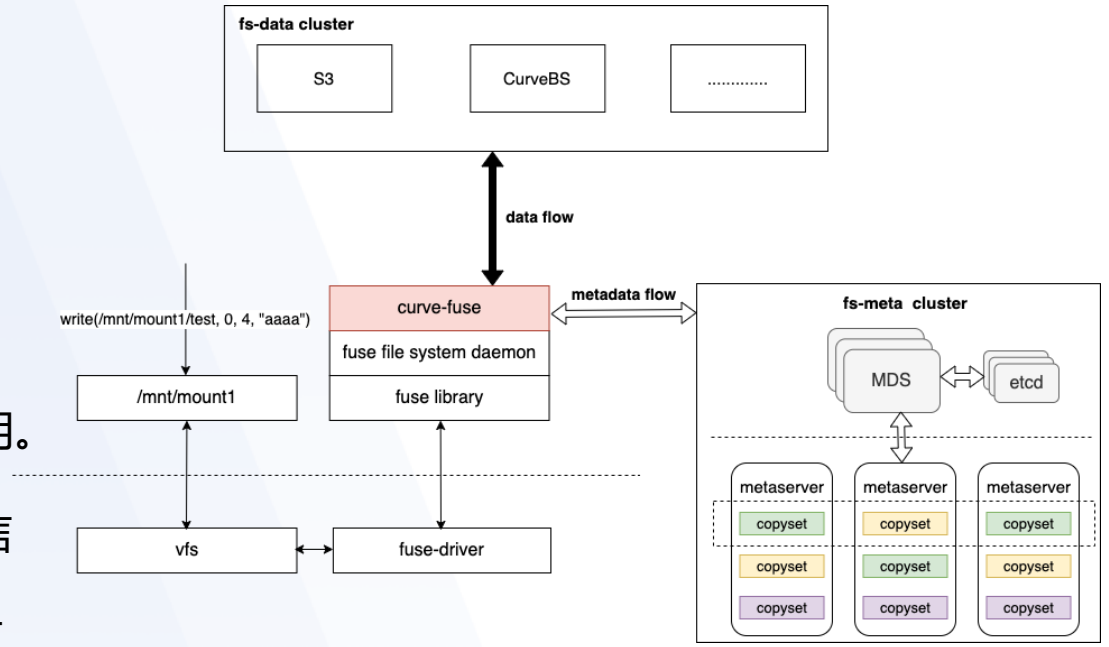
Curve文件存储RAFT应用

Curve文件存储

- 分布式文件系统
- 支持多挂载，提供close-to-open一致性
- 提供缓存加速，可使用内存、本地盘、云盘加速
- 存储后端可对接对象存储，降低成本
- 支持生命周期管理

Curve文件存储架构

- client：接受用户请求，采用fuse的方式挂载挂载使用。
- 元数据集群：mds 和 metaserver。
  - mds：保存元数据，包括topo信息、文件系统信息、元数据分布信息等，持久化到etcd中。
  - **metaserver**：采用**raft协议**3副本的方式保存文件文件的元数据，包括inode，dentry，文件的空间分配信息。
- 数据集群：采用外部存储，S3或者Curve块存储，保存写入文件的数据。



Curve文件存储RAFT应用

Curve文件系统与Curve块存储的实现区别

- Curve文件系统也是使用copyset管理。
- 写日志的方式与Curve块存储基本一致，实现细节略有差异。
- raft apply和raft snapshot的实现和Curve块存储不同。
- metaserver有两套存储引擎，基于memory和基于rocksdb。

基于memory的存储引擎

- 要求存储的元数据的大小不超过内存的大小
- raft apply的请求，数据都在内存，直接修改内存中的数据
- raft snapshot，为避免快照对正常操作的影响，利用操作系统的内存写时复制技术，fork一个进程创建完整的状态机的内存快照，后台遍历内存，把内存的数据持久化到本地磁盘

基于rocksdb的存储引擎

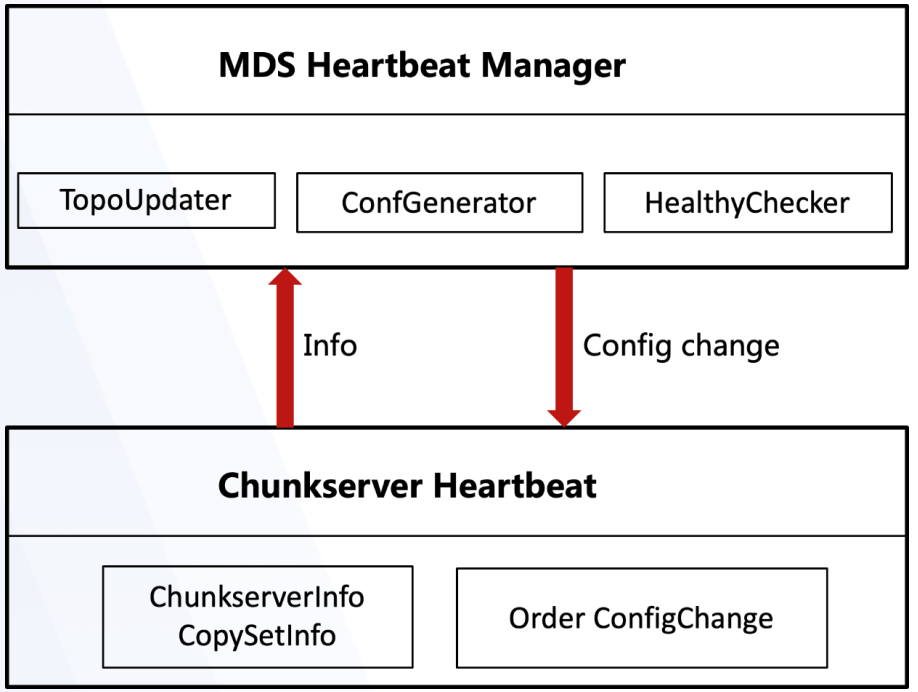
- 存储元数据量不受内存大小限制
- raft apply请求，数据保存在rocksdb，向rocksdb插入记录。
- raft snapshot，利用rocksdb的快照功能，对rocksdb打快照。

CURVE的RAFT配置变更

Curve块存储和文件存储的配置变更实现基本一致

心跳

- 通过心跳维持mds和chunkserver/metaserve的数据交互
- mds发起配置变更，copyset复制组执行
- 在curve自动容错和负载均衡时，需要进行raft配置变更。
- 自动容错保证常见异常（如坏盘、机器宕机）导致的数据丢失不依赖人工处理，可以自动修复。
- 负载均衡和资源均衡保证集群中的磁盘、cpu、内存等资源的利用率最大化。



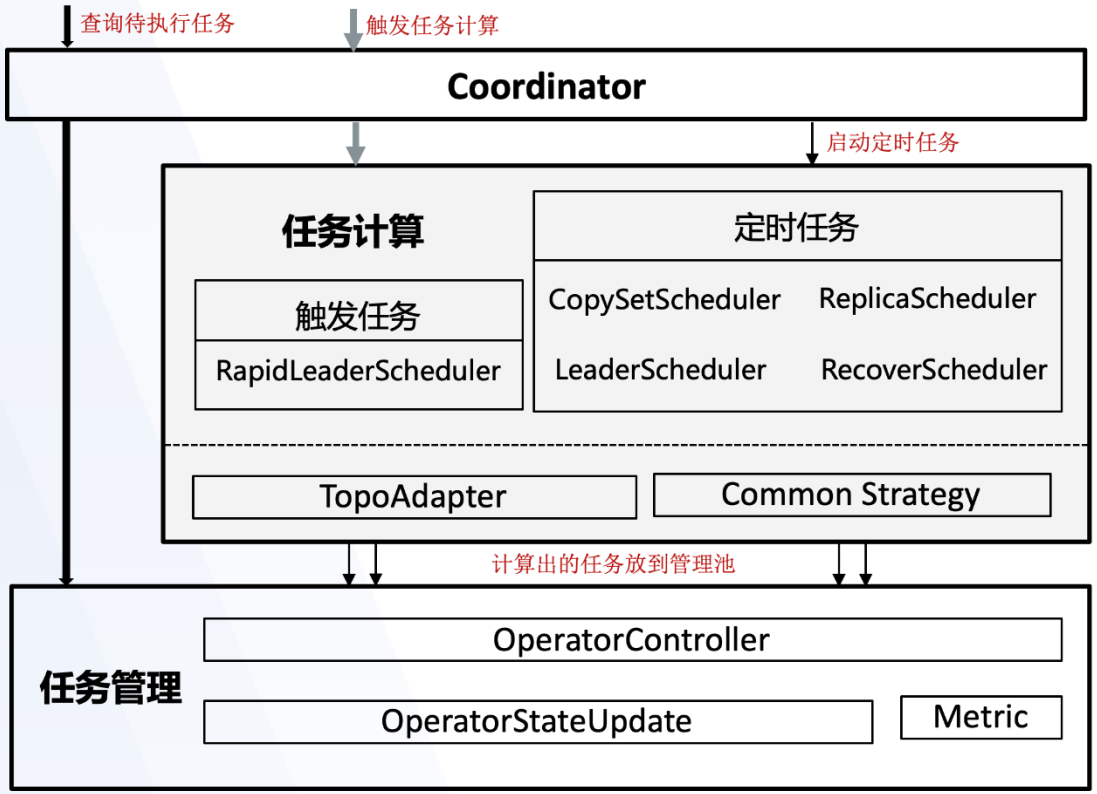
CURVE的RAFT配置变更

异常场景下配置变更

- ReplicaSchedule
- RecoverSchedule

均衡场景下配置变更

- CopySetSchedule
- LeaderSchedule
- RapidLeaderScheduler

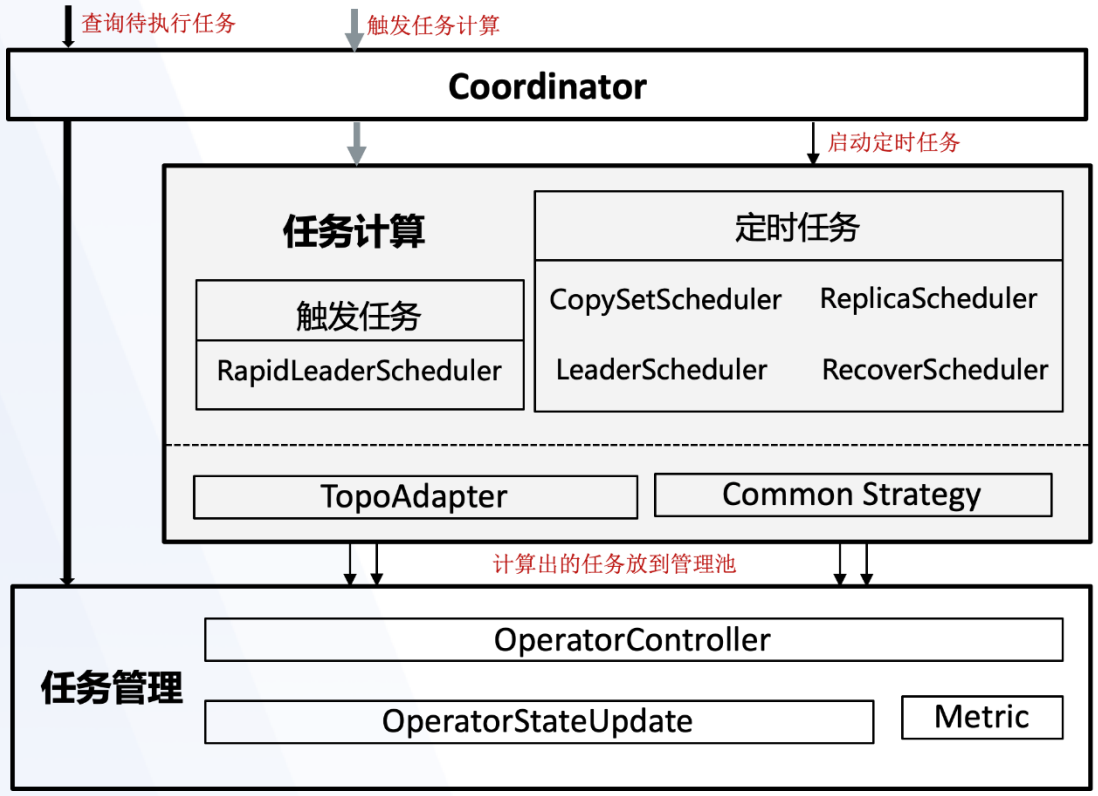


CURVE的RAFT配置变更

异常场景下配置变更

- **ReplicaSchedule**
  (a, b) -> (a, b, c) or
  (a, b, c, d) -> (a, b, c, d)
  保持raft的副本个数为指定值

- **RecoverSchedule**
  (a, ~~b~~, c) -> (a, c, d)
  替换不能提供服务的server



CURVE的RAFT配置变更

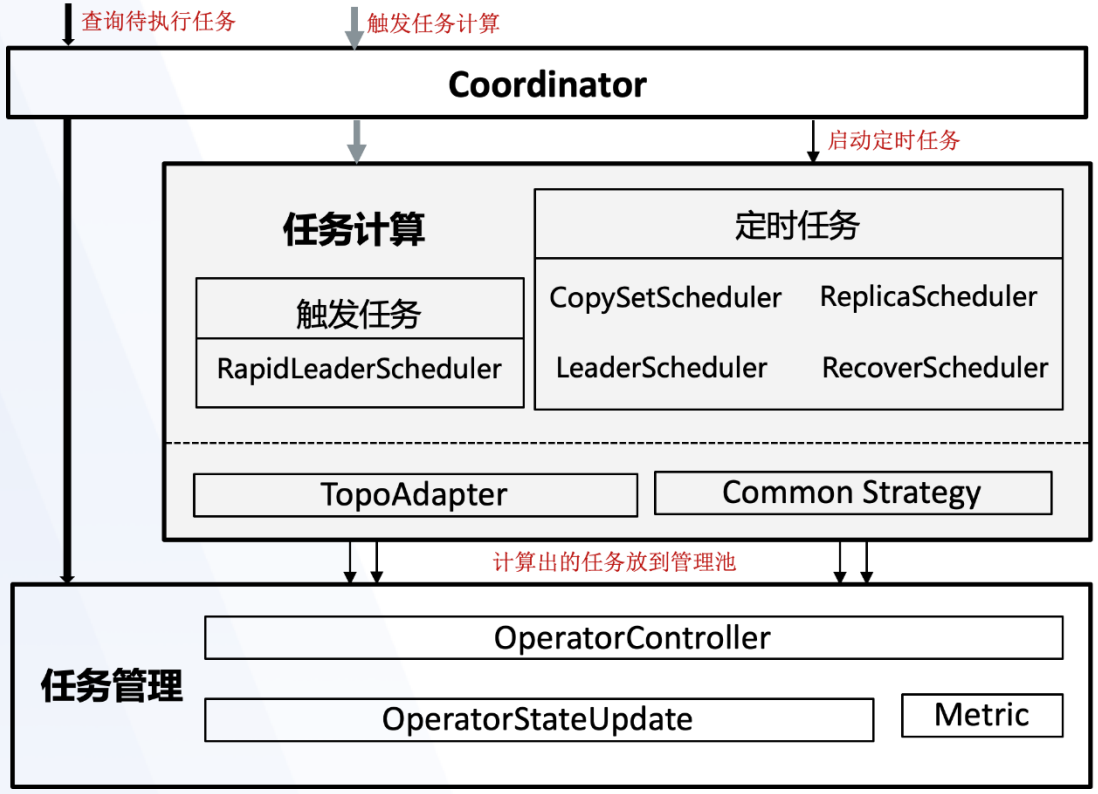
Curve

均衡场景下配置变更

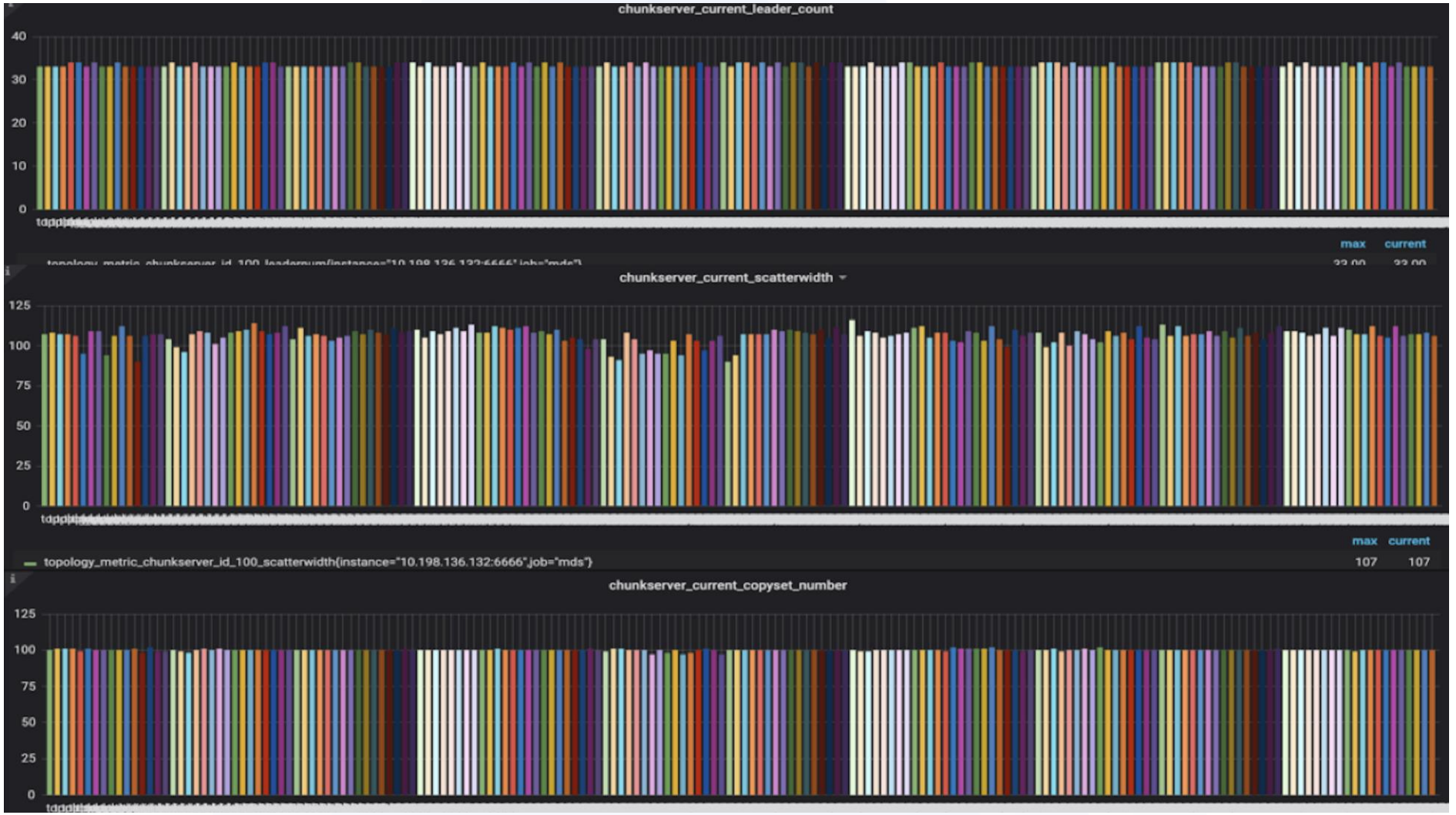
- **CopySetSchedule**
  (a, b, c) -> (a, b, d)
  每个chunkserver上的copyset个数尽量均衡

- **LeaderSchedule**
  (a(leader), b, c) -> (a, b(leader) , c)
  每个chunkserver上的leader copyset个数尽量均衡。

- **RapidLeaderScheduler**
  手动执行，快速leader均衡。
  用于新建集群、扩容集群、升级服务等场景



CURVE的均衡效果


chunkserver_current_leader_count
40
30
20
10
0
max current
chunkserver_current_scatterwidth
125
100
75
50
25
0
max current
topology_metric_chunkserver_id_100_scatterwidth{instance="10.198.136.132:6666",job="mds"}
107 107
chunkserver_current_copyset_number
125
100
75
50
25
0

Curve

DIGITAL SAIL

目录

01 **Curve介绍**

项目背景 | Curve架构 | 使用场景 | Curve社区

02 **raft和braft**

raft协议介绍 | braft介绍

03 **raft在Curve中的应用**

raft in Curve块存储 | raft in curve文件存储 | 配置变更

04 **Curve对raft的优化**

优化点1 | 优化点2

05 **Q&A**

答疑

Curve对RAFT的优化

Curve

优化点一：轻量级快照

问题背景：

raft的快照需要定期打快照，用来清理log。对于Curve块存储场景，系统状态就是Chunk当前的数据。
如果把所有chunk 拷贝一遍打快照，会出现两个问题：

1. 每次快照，空间上要多出1倍，空间浪费严重。
2. Curve块存储快照间隔默认30 分钟一次，每次快照会产生大量的数据拷贝，占用chunkserver的处理能力，对磁盘造成很大压力，影响正常IO。

解决思路：

chunk支持覆盖写，覆盖写天然是幂等的，写一次和写多次结果一致。
打快照只记录chunk文件的列表，不拷贝chunk的内容。从快照+日志加载数据时，下载的chunk文件不是打快照的状态，而是最新的状态，回放日志时，把数据重写一遍。

Curve对RAFT的优化

Curve

优化点二： chunkfile pool

问题背景：

Chunkserver使用基于ext4实现的本地文件系统，由于写操作存在较大的IO放大，因此在创建chunk文件时会调用fallocate为文件预分配固定大小的空间，但是即便fallocate以后，在写文件未写过的块时仍需要更改元数据，存在一定的IO放大。

解决思路：

直接使用覆盖写过一遍的文件。由于chunk大小固定，预先生成一批被写过的固定大小文件。创建chunk文件或快照文件时直接从预分配的文件池中获取进行重命名，删除chunk时再将文件重命名放到预分配池中，这个预分配池就是chunkfile pool。

进一步优化：

对chunk写0 -> nvme的write zero

Curve

DIGITAL SAIL

目录

01 **Curve介绍**

项目背景 | Curve架构 | 使用场景 | Curve社区

02 **raft和braft**

raft协议介绍 | braft介绍

03 **raft在Curve中的应用**

raft in Curve块存储 | raft in curve文件存储 | 配置变更

04 **Curve对raft的优化**

优化点1 | 优化点2

05 **Q&A**

答疑

Q&A

Curve

OpenCurve_bot

扫一扫上面的二维码图案，加我为朋友

- 主页：https://opencurve.io/
- 论坛：https://ask.opencurve.io/
- Github：https://github.com/opencurve/curve
- 公众号：OpenCurve
- 用户群：添加微信号OpenCurve_bot可邀请加群
- Slack：workspace cloud-native.slack.com，channel #project_curve