CoolCeph系统设计

袁冬 2017年11月

# 3 数据分布

## 3.1 数据分布

为了解决对象副本间的数据同步问题，系统在对象和DataNode之间引入一个中间层，称之为PG（Placement Group）。PG表示一个副本组，即处于同一个PG中的对象，其数据分布是一致的。

对象先映射到PG，然后PG再映射到DataNode，如下图所示：



PG可以理解为一个逻辑概念，即对象的集合。同时，PG在系统实现中通常对应一个本地引擎。PG的多个实例，即其多个副本，是完全对等的，互相之间通过MultiPAXOS协议保证一致性。

### 3.1.1 对象映射到PG

系统当前的PG数目是相对静态的，通常可以认为是不变地，例如256个，因此对象映射到PG的过程采用直接HASH，即

pg\_id = hash(oid) % pg\_num

由于对象数目远多于PG的数目，因此可以认为对象是均匀的分布在所有PG中的。

### 3.1.2 PG映射到DataNode

PG到DataNode的映射通过集群视图完成，集群视图可以理解为一个查找表，其中包含了每一个PG对应的DataNode的信息。

集群视图的相关信息参考3.2节。

### 3.1.3 PG的数目

PG的数目通常可以认为是固定的，仅当集群节点大幅度增加，以至于无法满足负载均衡要求时，才需要扩展PG的数目。

PG数目的扩展操作采用PG分裂的方式进行， 即原来的一个PG.X将分裂为两个子PG：PG.X1和PG.X2。PG分裂由于涉及PG之间的数据迁移，对集群的IO能力是有损的，因此需要尽量避免。

为了满足负载均衡和故障恢复的要求，通常要求每个磁盘分配256个PG，每台物理机12块磁盘，则为3072个PG。实际生产中需预留一部分buffer，则建议每个物理机4096个PG分配。当集群扩容导致每台物理机的PG数目小于4096时，建议进行PG的分裂操作。

另一方面，为了避免频繁的PG扩容导致集群服务能力的损失，建议采用翻倍PG数目策略，即

（1）集群初始化时指定的PG数目 = 物理机 \* 4096 \* 2；

（2）每次PG分裂时，对每一个PG都执行分裂操作，使PG数目翻倍。

这样可以保证只有当集群容量翻倍时，才需要执行PG分裂操作，也有助于更好的负载均衡。

## 3.2 集群视图

系统的集群视图描述了PG到DataNode之间的映射关系，即集群的数据分布。

当且仅当集群的数据分布（特别PG到DataNode之间的映射关系）发生变化时，集群的视图才会发生变化。反之，集群内任意数据分布的变化，包括集群扩容、节点故障、PG分裂、从新负载均衡等导致的各类数据重新分布的场景，都通过集群视图的更新完成。

集群视图采用版本（epoch）的组织方式，每一个版本的视图表示了当前版本下，集群所期望的数据分布状态，即目标状态。

集群视图不管理数据变更的过程，数据变更的过程由DataNode根据其所持有的数据的当前状态和集群视图给出的目标状态计算得出。

### 3.2.1 集群视图的内容

集群视图中包括以下主要内容：

* 集群视图版本号（epoch）：单调递增
* DataNode信息：包括每个DataNode的IP:Port、容量、状态等
* PG信息：包括每个PG的ID、数据分布（所在DataNode）等

其中DataNode的状态包括：

* Wait：DataNode已经启动，但是尚未加入到视图中，不提供服务
* Active：DataNode已经启动并且加入到Hash环中，正常服务状态
* Expired：DataNode被认定为超时，但还未进行数据分布的变更

### 3.2.2 集群视图的生成

集群视图的生成由NameNode负责，NameNode会根据集群内的PG数目、DataNode的状态和容量等信息，计算出每一个PG的实例所在的DataNode。

需要注意的时，PG实例对应的DataNode是有序的。

具体生成算法可以是基于容量的负载均衡，也可以是一致性HASH。(暂未确定)

### 3.2.3 集群视图的变更

集群视图的作用是用于表示数据分布状态，因此需要重新分布数据的场景会导致集群视图的变更，包括以下可能的场景：

* DataNode故障
* DataNode加入/集群扩容
* PG分裂
* 重新负载均衡

当集群视图需要变更时，由NameNode根据目标状态的PG和DataNode信息，计算出PG的数据分布，从而得到完整的集群视图，并使得集群的epcoh +1。最后，NameNode将新生成1的集群视图分发给集群内的所有DataNode和连接到集群的SDK。

集群视图分发过程中，由于集群将短暂的处于视图不一致的状态，因此集群的IO将受到影响，具体影响大小视视图分发速度和集群规模而定。后续将通过AB切换的方法尽可能的降低该影响。