CoolCeph系统设计

袁冬 2017年11月

# 2 系统架构

## 2.1 系统架构



系统以lib的形式提供SDK给业务端，业务端需要将改lib编译连接到自己的程序中，然后调用SDK的接口向集群发起IO请求。

系统有两个核心的组件：DataNode和NameNode。其中DataNode即为实际的存储数据服务器；NameNode则负责DataNode的管理，包括DataNode的加入、离开、负载均衡等。

通常情况下，SDK在启动的时候会与NameNode进行通讯，获取当前集群中所有DataNode信息，并缓存到本地，此后的所有请求都直接发往DataNode。只有当集群的DataNode发生变化时，例如扩容新的DataNode，SDK才需要再次与NameNode进行通讯，以便获取新的DataNode集群信息。

DataNode管理其所在服务器上的磁盘，并处理SDK发起的数据读写请求。DataNode也需要从NameNode获取到视图，以便确认自身仍然作为一个被认可的节点提供服务。DataNode在故障恢复等场景下只会相互之间进行通讯，以便完成数据缺失副本的重建等操作。

NameNode采用Active-Standy的高可用方式，实现中依赖于外部的Zookeeper集群（后续可以可能考虑去除对ZK的依赖）。

## 2.2 功能模块

（1）SDK

SDK以库的方式（cceph.h, cceph.a, cceph.so）提供给业务端。业务端需要将其编译链接进业务程序的内部。

当需要访问集群时，需要调用SDK的初始化方法对其进行初始化，然后即可对集群发起IO请求。当不在需要访问集群时，需要调用销毁方法SDK所占用的相关资源。

（2）DataNode

DataNode是提供实际处理客户端发送的IO请求的节点。每一台物理机通常运行一个DataNode服务，管理该机器上所有的磁盘资源。

集群中的对象根据数据分布算法均匀的分散到集群中所有的DataNode上，当SDK发起的请求达到DataNode的时候，DataNode会首先判断请求的对象是否属于当前DataNode负责，如果是，则负责该请求的处理，并在处理完成后，向SDK返回ACK；如果不是，则拒绝该请求。

当对集群的DataNode进行扩容或者集群中DataNode由于故障等原因退出集群时，DataNode需要根据变更后的数据分布情况进行数据恢复操作，以便完成数据副本的重建，并保证对象的分布符合集群数据分布算法给出的预期结果。

（3）NameNode

DataNode负责集群内DataNode的管理，即根据DataNode的扩容和移除，更新集群的视图以及对应的数据分布情况。

SDK和DataNode在需要集群视图的场景，例如新加入集群或者视图更新的情况下，都需要跟NameNode进行通讯，已获取最新的集群视图。

NameNode采用Active-Standy的方式实现高可用，支持一主多从。当前Active故障的情况下，通常需要10~30秒的时间完成Standy的切换操作。

由于NameNode只负责DataNode的管理，因此NameNode不可用的情况下，只会影响DataNode的加入和离开，并不影响数据的IO操作。

（4）Zookeeper

外部的ZK集群，主要用户实现NameNode的高可用，还用于保存集群视图等少量的元数据信息。实际生产中，通常包括三个或者五个独立的Zookeeper服务器，部署不同的TOR之下。Zookeeper可以和NameNode混布。