etcd: 基本介绍

李嘉睿

中间件团队

2020.9

主要内容

- 什么是 etcd
- 为什么使用 etcd
- etcd 一些实现细节
- 如何使用 etcd
- 一些仍需解决的问题

什么是 etcd

etcd 是一个强一致的分布式键值存储。

强一致

- 基于 raft 共识算法，单一 raft 备份组，无分片
- 线性读写: leader 分配所有备份节点的全局读写顺序，同时保证读总是能读到最新的值

数据模型：键值存储 + 版本

扁平的键空间模拟文件目录结构。

- 持久化: 键值对存储在持久化 b+ 树中，支持对键的范围查找
- 内存 b 树缓存指向键值对的指针
- MVCC: 每次数据版本更新添加数据增量部分

为什么使用 etcd

元数据存储

单一 raft 备份组，无分片。

- 无分片，水平扩展能力较差，支持数 GB 数据 (默认 2GB，最大 8GB)，因此海量数据存储需要使用 NewSQL 数据库。
- 无分片，无需分片备份组之间两阶段提交以及分布式锁的开销，性能更好。

分布式协调

开箱即用的分布式协调原语。

- 提供监视器、租约、leader 选举和分布式锁的支持。
- 简单易用，支持 Restful 接口，可以从命令行使用分布式协调服务。
- 使用 gRPC 框架，已经有多种语言的 API 支持或客户端实现。

为什么使用 etcd

本质上，etcd 和 zookeeper 解决了相同的问题。其比较如下：

比较

- 数据模型：etcd 使用键值对，支持范围查找，使用 role-based 访问控制。zk 使用树形 znode 结构，其内包含 ACL 访问控制列表。
- 并发原语：etcd 内置并发原语，zk 使用外部客户端库 curator。
- 读操作：zk 不支持线性化读，读操作可能读到过时数据。
- MVCC：etcd 支持 MVCC，zk 不支持。
- 监视器通知：etcd 支持范围键值的监视器。
- API 支持：etcd 支持 HTTP/JSON API，zk 不支持。
- RPC 框架：etcd 使用 grpc 框架，zk 使用自己定制的 rpc 协议。
- 存储限制：etcd 最多存储 8GB，zk 通常支持几百 MB。

如何使用 etcd

etcd 其它细节：客户端实现

grpc1.0

客户端为每个 endpoint 维护一个 TCP 连接，第一个成功建立连接的 endpoint 作为"pinned address". 多个 TCP 连接有利于更快的故障恢复，但是需要更多资源。

grpc1.7

首先尝试连接所有集群服务器，维护第一个成功连接的 TCP 连接。遇到错误时，由客户端的错误处理器 (error handler) 决定是否重连或者选择新的服务器地址。需要维护 endpoints 状态列表，其中不健康状态的判定是 false positive 的，即被标记为不健康的节点可能在之后恢复健康。

grpc1.23

客户端为每个 endpoint 维护一个 TCP 连接，通过轮转 (round robin) 负载均衡，通过 gRPC 链式拦截器实现重连. 仍未解决:网络分区情况下阻塞，缺少集群健康情况查询服务。