lightFS 系统设计说明书

易剑 2013/6/8

目录

修订记录

修改人 （格式：中文全名）	修改日期 （格式：2013/6/8）	修改记录 （格式：要求描述增修减了什么内容）
易剑	2013/6/8	创建

1 前言

1.1什么是 lightFS？

lightFS 是一个设计理念完全不同于 GFS/HDFS 的轻量级分布式容灾文件系统。它完美的利用了 CAP 原理，通过牺牲一致性，实现了超高的可用性和可靠性。

1.2文档目的

编写本文的目的，是为进一步将脑海中的思路沉淀下来，以便做进一步优化。同时，用以指导其他开发人员去实现 lightFS。

1.3参考资料

《lightFS-牺牲强一致性获得超高可用性和可靠性的磁盘级轻量分布式文件系统.ppt》

1.4技术交流

1.4.1 技术博客

http://aquester.cublog.cn

1.4.2 开源项目

http://code.google.com/p/mooon，lightFS 将作为 mooon 的一个子项目存在。

1.4.3 技术论坛

http://bbs.hadoopor.com

2 应用场景

- 无超大文件，主要几十兆或几百兆字节的文件
- 不需要强一致性
- 觉得 HDFS 等太重

3 设计目标

- 文件自动容灾，对外透明
- 超高的可用性和可靠性，近完美的 A 和 P（相对 CAP 原理）
- 最终的一致性
- 可容忍一半的节点故障
- 兼容 sshd，可通过 scp 上传和下载文件

4 设计指标

4.1 集群大小

目标集群为 100~1000 个节点，最佳性能在 100 个节点左右。

4.2 存储容量

假设平均文件路径为 1K，则 1000 个文件为 1M，100 万文件为 1G，10 万文件为 100M，设计支持容量为 10 万个文件，文件最大为 4G。

4.3 设计约束

- 不支持文件分块
- 单个文件最大不能超过 2G
- 不提供强一致性，但保证最终一致性

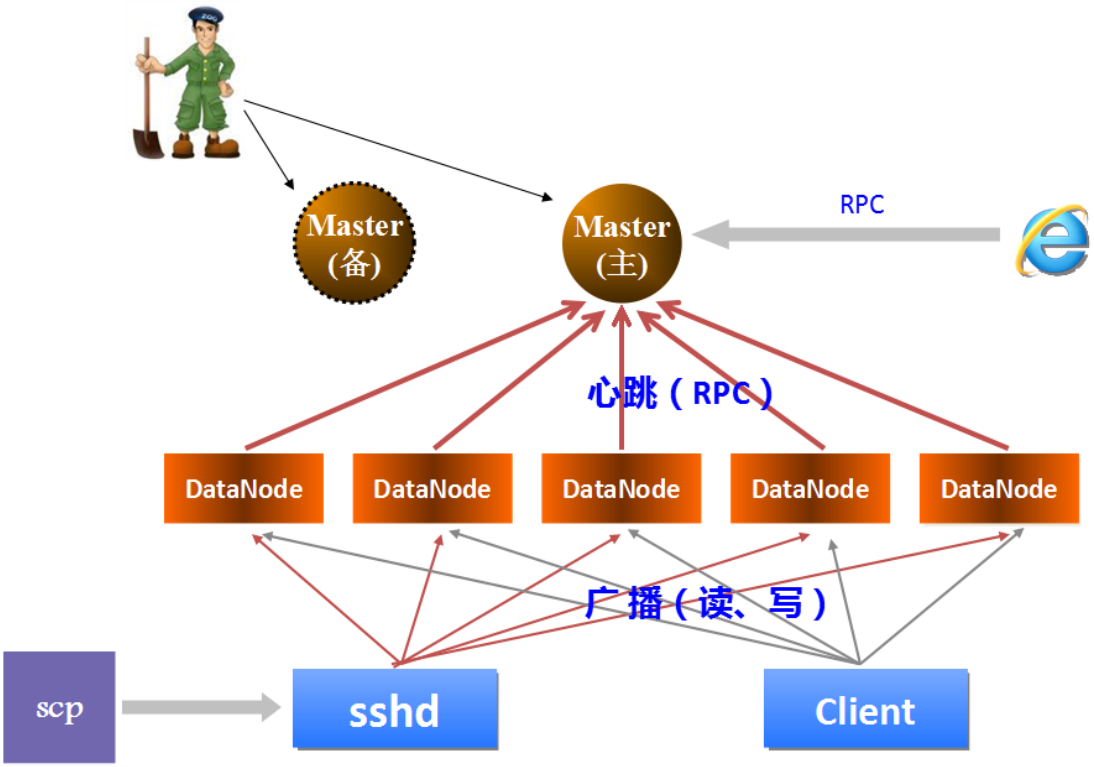
5 接口设计

采用和 POSIX 兼容接口设计：
int open(const char *pathname, int flags, mode_t mode);
int open(const char *pathname, int flags);

6 命令行工具

命令行工具的存在，是为便于直接操作存在于 lightFS 上的整个文件或多个文件。

7 总体结构



备注：lightFS 采用的是弱主架构。

7.1 lightFS-Master

主控节点，主要行驶如下功能：

1) 监控 DataNode，异常时告警
2) 汇总元数据，提供浏览接口
3) 补齐副本数
4) 删除脏数据
5) 提供整个集群运营统计接口

6) 采用 RPC 和 DataNode 通讯

7.2 lightFS-DataNode

数据节点，用来存放文件的节点，主要功能为：

1) 存储文件，并提供读写文件接口
2) 维护自身的文件元数据
3) 能够根据文件扫描自动重建元数据
4) 元数据按磁盘分开管理
5) 磁盘级容灾
6) 接受 Master 指令，能够从另一 DataNode 拉取副本
7) 支持 100~1000 的并发量

7.3 lightFS-Client

客户端，可以为用户程序，也可以为 lightFS 提供的客户端工具：

1) 发起写文件和读文件广播
2) 多写方式实现多副本（缺点：效率低）

7.4 lightFS-Web

管理前台，提供如下功能：

1) 浏览 lightFS 运营状态
2) 浏览目录树
3) 执行对文件的删除、改名等操作

7.4.1 运营指标

1) 每小时/天/周/月，新增/删除/修改的文件个数和大小
2) 各 DataNode 节点的系统资源、负载情况

7.5 网络通信

1) lightFS-Master 和 lightFS-Web 间采用 thrift RPC 通信
2) lightFS-Master 和 lightFS-DataNode 间采用 thrift RPC 通信
3) lightFS-Client 和 lightFS-DataNode 采用 thrift RPC 和 socket 两种方式，其中 socket 方式用于读写文件

8 lightFS-Master 结构

9 lightFS-DataNode 结构

10 lightFS-Client 结构

11目录树设计

11.1设计要求

- 占用内存小，浪费控制在 10%以内
- 提供一个基目录，可以自动扫描后生成
- 能够快速打包成字节流，以方便从 DataNode 传到 Master
- 能够合并，比如在 Master 需要将各节点的目录树合并成一颗大的

12 附 1：技术决策

12.1网络通讯

lightFS 计划基于 mooon 实现，同时采大量使用开源库，如 thrift RPC 来简化实现。

读写文件究竟是采用 RPC 还是直接的 socket 了？

RPC 直接上会简化逻辑，但如何处理网络事件了？比如 DataNode 数据收不过来，Client 什么时候才可以继续发送。

12.2是否支持文件分块

lightFS 的定位究竟是什么，根据它的定位来决定是否需要分块。基于 lightFS 的应用场景和设计目标，最终决定放弃对文件分块，以简化实现。

12.3多副本写入方式

写入多副本有两种常见选择：一是由 Client 直接写入所有副本；二是 Client 只直接写入一个副本，其它副本的写入交给 DataNode 完成：

	Client 直接写入所有副本	**Client 只直接写入一个副本**
优点	1) 简单明了 2) DataNode 的逻辑相对简单	1) Client 实现简单 2) Client 和 DataNode 跨 IDC 时，写操

		作性能影响小 3) Client 和 DataNode 可以分布不同的 IDC
缺点	1) Client 写操作压力较大 2) Client 和 DataNode 跨 IDC 时，写操作性能下降和副本数约成正比，有异常时更为严重 3) 应用局限于 Client 和 DataNode 部署在同一个 IDC	1) DataNode 的逻辑相对复杂

12.4DataNode 和 Master 通信方向

是 DataNode 主动连接 Master，还是 Master 主动连接 DataNode？

	DataNode 主动连接 Master	**Master 主动连接 DataNode**
优点	1) 实现简单，DataNode 只需要一个专用的 Agent，而 Master 为通用 Server	1) DataNode 上不用配置 Master 的 IP 2) 主备 Master 切换，DataNode 不用关心
缺点	1) 主备 Master 切换时，DataNode 连接的 Master 的 IP 需要变更	1) DataNode 需要为 Master 开端口 2) Master 需连接多个 Server

12.5Client 向 DataNode 写数据策略

Client 可以采用广播方式取所有 DataNode 信息，然后选择向最轻的几个 DataNode 写数据；也可以采用随机的方式选择 DataNode：

	广播方式	**随机选择**
优点	1) 完全保证均衡	1) 性能相对较好 2) 实现非常简单 3) 加入新的空闲节点时，不需要特别处理，即可保证数据不会倾向于新节点
缺点	1) 性能相对较差，因为需要保证一定量的节点响应后才可以选择可写的节点 2) 实现相对复杂 3) 新节点加入时，必须先由 Master 发起均衡，否则数据会倾向于新节点	1) 可能会存在不均衡

13 附 2：思考

13.1如何支持文件分块

13.2如何支持快速检索

13.3如何支持写时压缩

13.4如何支持上万节点的集群

增加 proxy。

14 附 3：开发计划

预计召集 3+1 人，其中后台开发 3 人，前台 1 人，7 个工作日左右完成 demo 版本的开发，约 30 个工作日提供一个稳定版本（不包含 ssh 兼容特性）。

14.1人员名单

目前确定会参与开的人员名单如下：

姓名	前后台	负责模块	简介
易剑 eyjian@qq.com eyjian@gmail.com	后台	系统设计、项目协调和 DataNode 模块的实现	2002 年本科毕业，擅长 C/C++程序开发，热衷于软件设计，对分布式系统和大规模数据系统具有较大兴趣
曹彪 caobiao@gmail.com	后台	Client 模块的实现	
丁峰峰 haofefe@163.com	前台	lightFS-Web 的实现	擅长.net 和 js，自创了基于 HTML5 的画流程图组件 jmgraph

15 附 3：授权协议

lightFS 将采用 Apache 协议，毫无保留地对外开放，供学习、研究和直接使用，不附带任何约束条件。