分布式文件系统FastDFS架构剖析及配置优化

余 庆

2012年04月08日

DTCC2012

FastDFS概述

FastDFS是一款开源的轻量级分布式文件系统

- 纯C实现，支持Linux、FreeBSD等UNIX系统
- 类google FS，不是通用的文件系统，只能通过专有API访问，目前提供了C、Java和PHP API
- 为互联网应用量身定做，解决大容量文件存储问题，追求高性能和高扩展性
- FastDFS可以看做是基于文件的key value pair存储系统，称作分布式文件存储服务更为合适

FastDFS提供的功能

- upload：上传普通文件，包括主文件
- upload_appender：上传appender文件，后续可以对其进行append操作
- upload_slave：上传从文件
- download：下载文件
- delete：删除文件
- append：在已有文件后追加内容
- set_metadata：设置文件附加属性
- get_metadata：获取文件附加属性

FastDFS的特点

- 分组存储，灵活简洁
- 对等结构，不存在单点
- 文件ID由FastDFS生成，作为文件访问凭证。FastDFS不需要传统的name server
- 和流行的web server无缝衔接，FastDFS已提供apache和nginx扩展模块
- 大、中、小文件均可以很好支持，支持海量小文件存储
- 支持多块磁盘，支持单盘数据恢复
- 支持相同文件内容只保存一份，节省存储空间
- 存储服务器上可以保存文件附加属性
- 下载文件支持多线程方式，支持断点续传

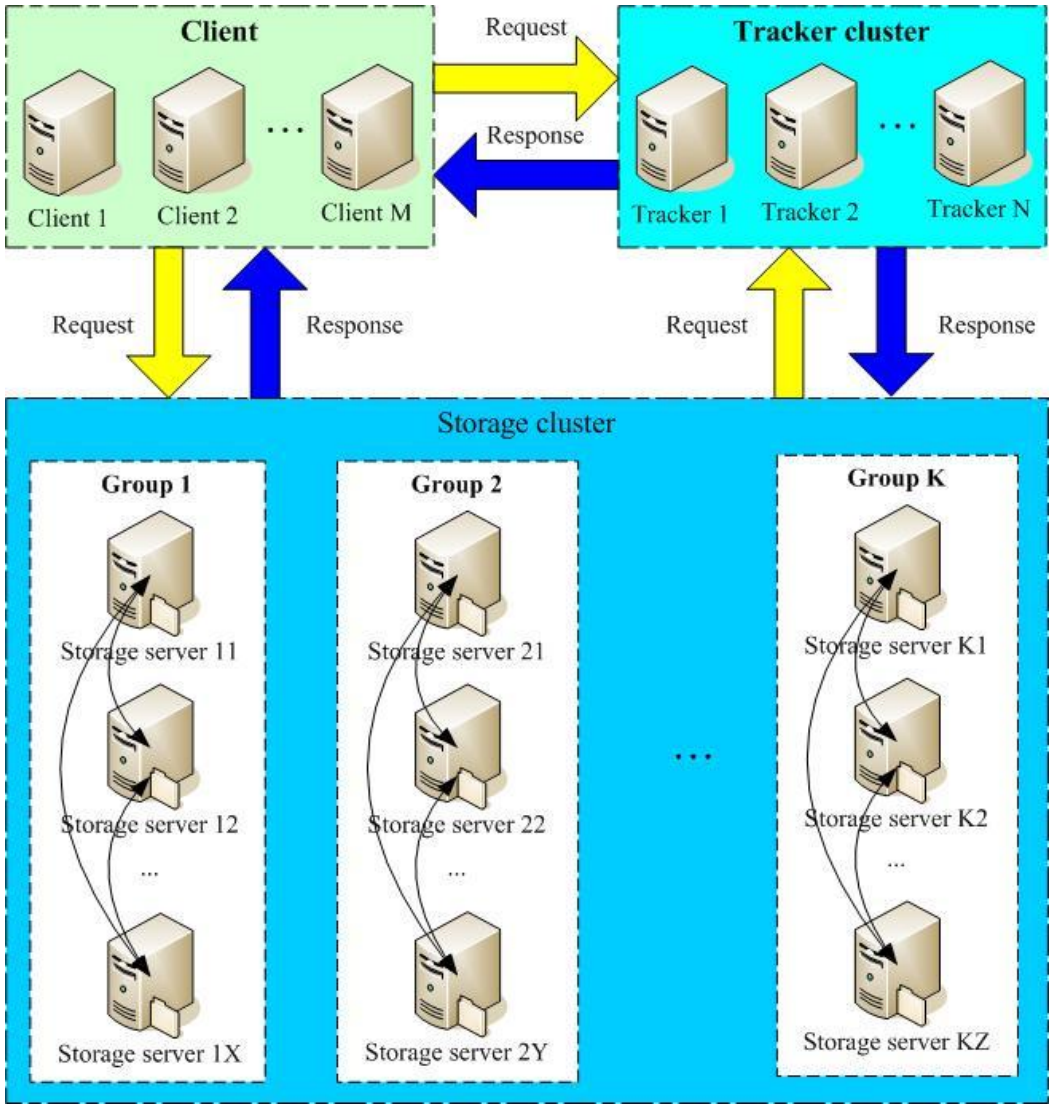
FastDFS发展历史

- 2008年4月项目启动，7月发布第一个版本V1.00，两年的时间内持续升级到V1.29。
- 2010年8月推出V2.00。V2.x最新版本是V2.13。
- 2011年6月推出V3.00。当前最新版本是V3.06。V3.x后续会一直进行维护和升级
- V1和V2系列后续不再维护和升级

FastDFS的版本演变

- V1.x：采用传统的一个请求一个线程服务的模式，系统资源消耗较大，支持的并发连接数在1K左右
- V2.x：采用libevent异步IO模型，磁盘读写采用专门的线程，比V1.x的工作模型更加先进和高效。支持的并发连接数可以达到10K
- V3.x：支持小文件合并存储，解决海量小文件的存储问题

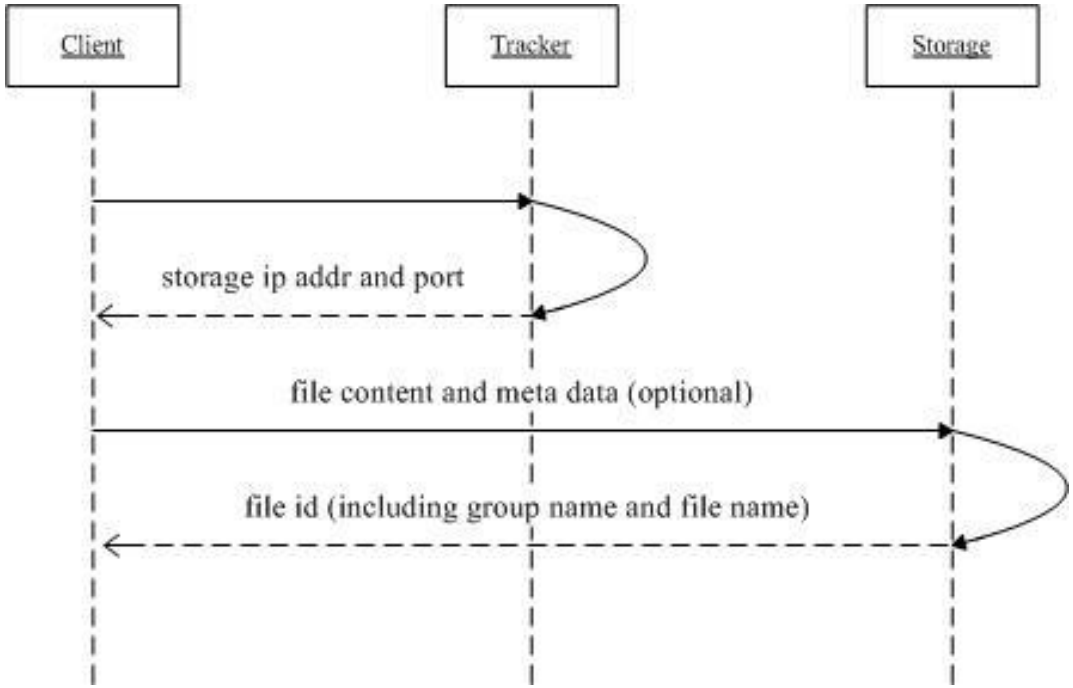
FastDFS架构示意图



FastDFS架构解读

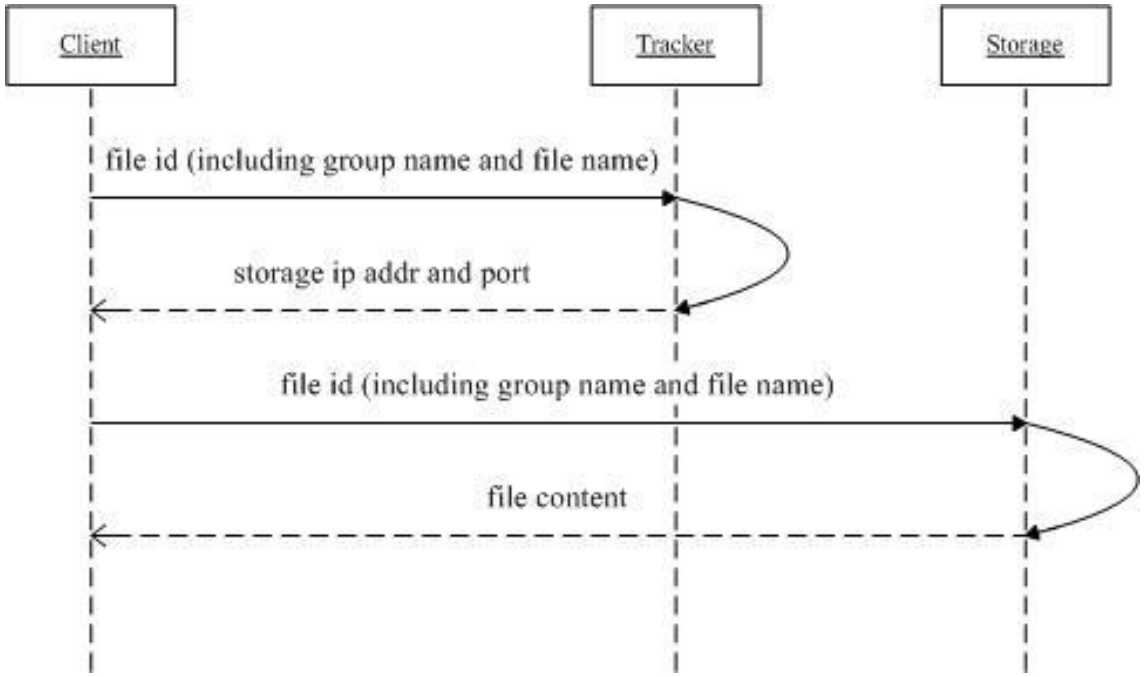
- 只有两个角色，tracker server和storage server，不需要存储文件索引信息
- 所有服务器都是对等的，不存在Master-Slave关系
- 存储服务器采用分组方式，同组内存储服务器上的文件完全相同（RAID 1）
- 不同组的storage server之间不会相互通信
- 由storage server主动向tracker server报告状态信息，tracker server之间通常不会相互通信

FastDFS上传文件流程图



1. client询问tracker上传到的storage；
2. tracker返回一台可用的storage；
3. client直接和storage通信完成文件上传，storage返回文件ID。

FastDFS下载文件流程图



1. client询问tracker可以下载指定文件的storage，参数为文件ID（组名和文件名）；
2. tracker返回一台可用的storage；
3. client直接和storage通信完成文件下载。

FastDFS如何做到无索引服务器？

- 上传文件时，文件ID由storage server生成并返回给client
- 文件ID包含了组名和文件名，storage server可以直接根据该文件名定位到文件
- 一个文件ID示例：

文件名中包含的信息

- 采用Base64编码，包含的字段包括：
  - 源storage server IP地址
  - 文件创建时间
  - 文件大小
  - 文件CRC32校验码
  - 随机数

合并存储文件ID格式

- 在文件名后面，多了16字节
- 采用Base64编码，包含的字段包括：
  - 存放到的trunk file ID
  - 文件偏移量
  - 占用的空间大小

FastDFS同步机制

- 采用binlog文件记录文件上传、删除等操作，根据binlog进行文件同步
- binlog中只记录文件名，不记录文件内容
- 增量同步方式，记录已同步的位置到.mark文件中
- 同组内的storage server之间是对等的，文件上传、删除等操作可以在任意一台storage server上进行
- 文件同步只在同组内的storage server之间进行，采用push方式，即源头服务器同步给目标服务器

FastDFS如何解决同步延迟问题？

- storage生成的文件名中，包含源头storage IP地址和文件创建时间戳
- 源头storage定时向tracker报告同步情况，包括向目标服务器同步到的文件时间戳
- tracker收到storage的同步报告后，找出该组内每台storage被同步到的时间戳（取最小值），作为storage属性保存到内存中

下载文件选择storage的方法

- client询问tracker有哪些storage可以下载指定文件时，tracker返回满足如下四个条件之一的storage：
  - (当前时间 -文件创建时间戳) > 同步延迟阀值（如一天）
  - 文件创建时间戳 < Storage被同步到的时间戳
  - 文件创建时间戳 == Storage被同步到的时间戳且(当前时间 -文件创建时间戳) > 文件同步最大时间（如5分钟）
  - 该文件上传到的源头storage

以HTTP方式下载文件

- FastDFS分组存储方式，为HTTP方式下载提供了便利
- FastDFS支持HTTP方式下载文件，不建议使用内置web server，推荐使用外部web server，如果apache或nginx
- 因为需要解决文件同步延迟的问题，因此在apache和nginx上需要使用FastDFS扩展模块。尤其是V3.x引入小文件合并存储后，必须使用扩展模块来读取文件

FastDFS扩展模块要点

- 使用扩展模块来解决文件同步延迟问题
- 在每台storage server上部署web server，直接对外提供HTTP服务
- tracker server上不需要部署web server
- 如果请求文件在当前storage上不存在，通过文件ID反解出源storage，直接请求源storage
- 目前已提供apache和nginx扩展模块
- FastDFS扩展模块不依赖于FastDFS server，可以独立存在！

FastDFS扩展模块特性

- 仅支持HTTP HEAD和GET
- 支持token方式的防盗链（缺省是关闭的）
  - ts：生成token的时间（unix时间戳）
  - token：32位的token字符串（md5签名）
- 支持指定保存的缺省文件名，URL参数名为filename
- 支持断点续传

推荐的FastDFS部署方案

- 文件上传和删除等操作：使用FastDFS client API，目前提供了C、PHP extension和Java的client API
- 文件下载采用HTTP方式：使用nginx或者apache扩展模块，不推荐使用FastDFS内置的web server
- 不要做RAID，直接挂载单盘，每个硬盘作为一个mount point

最大并发连接数设置

- 参数名：max_connections
- 缺省值：256
- 说明：FastDFS采用预先分配好buffer队列的做法，分配的内存大小为：max_connections * buff_size，因此配置的连接数越大，消耗的内存越多。不建议配置得过大，以避免无谓的内存开销。

工作线程数设置

- 参数名： work_threads
- 缺省值：4
- 说明：为了避免CPU上下文切换的开销，以及不必要的资源消耗，不建议将本参数设置得过大。为了发挥出多个CPU的效能，系统中的线程数总和，应等于CPU总数。
- 对于tracker server，公式为：

  work_threads + 1 = CPU数
- 对于storage，公式为：

  work_threads + 1 + (disk_reader_threads + disk_writer_threads) * store_path_count = CPU数

storage目录数设置

- 参数名：subdir_count_per_path
- 缺省值：256
- 说明：FastDFS采用二级目录的做法，目录会在FastDFS初始化时自动创建。存储海量小文件，打开了trunk存储方式的情况下，建议将本参数适当改小，比如设置为32，此时存放文件的目录数为 32 * 32 = 1024。假如trunk文件大小采用缺省值64MB，磁盘空间为2TB，那么每个目录下存放的trunk文件数均值为：2TB / (1024 * 64MB) = 32个

storage磁盘读写线程设置

- disk_rw_separated：磁盘读写是否分离
- disk_reader_threads：单个磁盘读线程数
- disk_writer_threads：单个磁盘写线程数
- 如果磁盘读写混合，单个磁盘读写线程数为读线程数和写线程数之后
- 对于单盘挂载方式，磁盘读写线程分别设置为1即可
- 如果磁盘做了RAID，那么需要酌情加大读写线程数，这样才能最大程度地发挥磁盘性能

storage同步延迟相关设置

- sync_binlog_buff_interval：将binlog buffer写入磁盘的时间间隔，取值大于0，缺省值为60s
- sync_wait_msec：如果没有需要同步的文件，对binlog进行轮询的时间间隔，取值大于0，缺省值为100ms
- sync_interval：同步完一个文件后，休眠的毫秒数，缺省值为0
- 为了缩短文件同步时间，可以将上述3个参数适当调小即可

FastDFS使用现状

- 国内几十家公司在使用
- 规模最大的一家：集群中存储的group数有200个，存储容量达到3PB，文件数接近1亿个。Group持续增长中。。。

已知使用FastDFS的用户

- 某大型网盘（公司名对方要求保密）
- UC（http://www.uc.cn/）
- 支付宝（http://www.alipay.com/）
- 京东商城（http://www.360buy.com/）
- 淘淘搜（http://www.taotaosou.com/）
- 飞信（http://feixin.10086.cn/）
- 赶集网（http://www.ganji.com/）
- 淘米网（http://www.61.com/）
- 迅雷（http://www.xunlei.com/）
- 蚂蜂窝（http://www.mafengwo.cn/）
- 丫丫网（http://www.iyaya.com/）
- 虹网（http://3g.ahong.com/）
- 5173（http://www.5173.com/）
- 华夏原创网（http://www.yuanchuang.com/）
- 华师京城教育云平台（http://www.hsjdy.com.cn/）
- 视友网（http://www.cuctv.com/）
- 。。。

FastDFS相关网址

- FastDFS论坛：http://www.csource.org/
- google code项目地址：http://code.google.com/p/fastdfs/
- QQ技术交流群：164684842

Q & A

谢谢大家！