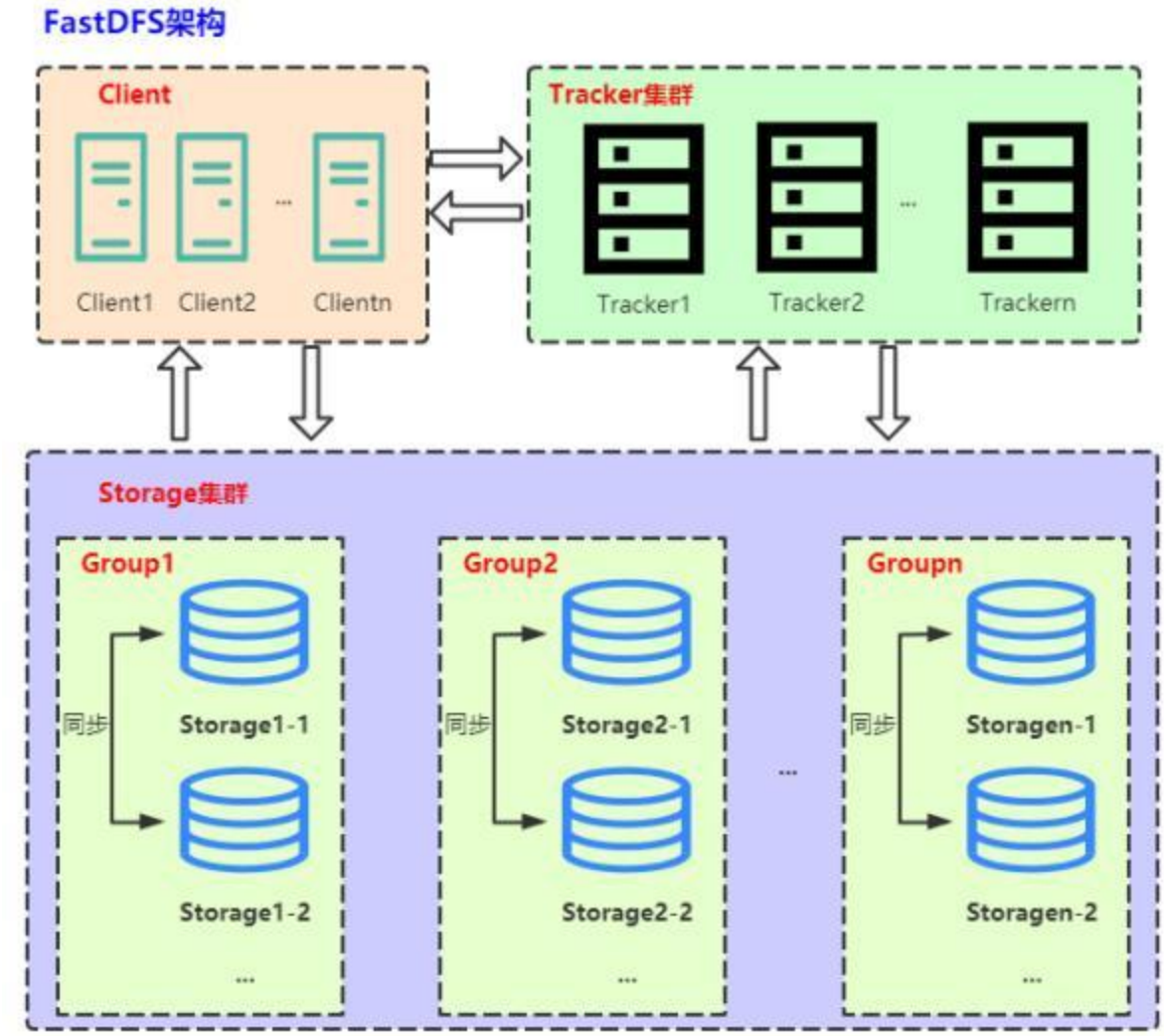
1 FastDFS架构分析

FastDFS是由国人余庆所开发，其项目地址： https://github.com/happyfish100

FastDFS主要的功能包括：文件存储，同步和访问，设计基于高可用和负载均衡。 FastDFS非常适用于基于文件服务的站点，。

FastDFS由**跟踪服务器（tracker server）**、**存储服务器（storage server）**和**客户端（client）**三个部分组成，主要解决海量数据存储问题，特别适合以中小文件（建议范围： 4KB < file_size < 500MB）为载体的在线服务，例如图片分享和视频分享网站。

FastDFS架构如下所示：



FastDFS服务有三个角色:跟踪服务器(tracker server)、存储服务器(storage server)和客户端(client)

1.1 Tracker server

Tracker是FastDFS的协调者，负责管理所有的storage server和group，每个storage在启动后会连接Tracker，告知自己所属的group等信息，并保持周期性的心跳， tracker根据storage的心跳信息，建立group==>[storage server list]的映射表。

Tracker需要管理的元信息很少，会全部存储在内存中；另外tracker上的元信息都是由storage汇报的信息生成的，本身不需要持久化任何数据，这样使得tracker非常容易扩展， 直接增加tracker机器即可扩展为tracker cluster来服务， cluster里每个tracker之间是完全对等的，所有的tracker都接受stroage的心跳信息，生成元数据信息来提供读写服务。

1.2 Storage server

Storage server （后简称storage）以组（卷， group或volume）为单位组织， 一个group内包含多台storage机器，数据互为备份，存储空间以group内容量最小的storage为准，所以建议group内的多个storage尽量配置相同，以免造成存储空间的浪费。

以group为单位组织存储能方便的进行应用隔离、负载均衡、副本数定制（group内storage server数量即为该group的副本数），比如将不同应用数据存到不同的group就能隔离应用数据，同时还可根据应用的访问特性来将应用分配到不同的group来做负载均衡；缺点是group的容量受单机存储容量的限制，同时当group内有机器坏掉时，数据恢复只能依赖group内地其他机器，使得恢复时间会很长。

group内每个storage的存储依赖于本地文件系统， storage可配置多个数据存储目录，比如有10块磁盘，分别挂载在/data/disk1-/data/disk10，则可将这10个目录都配置为storage的数据存储目录。

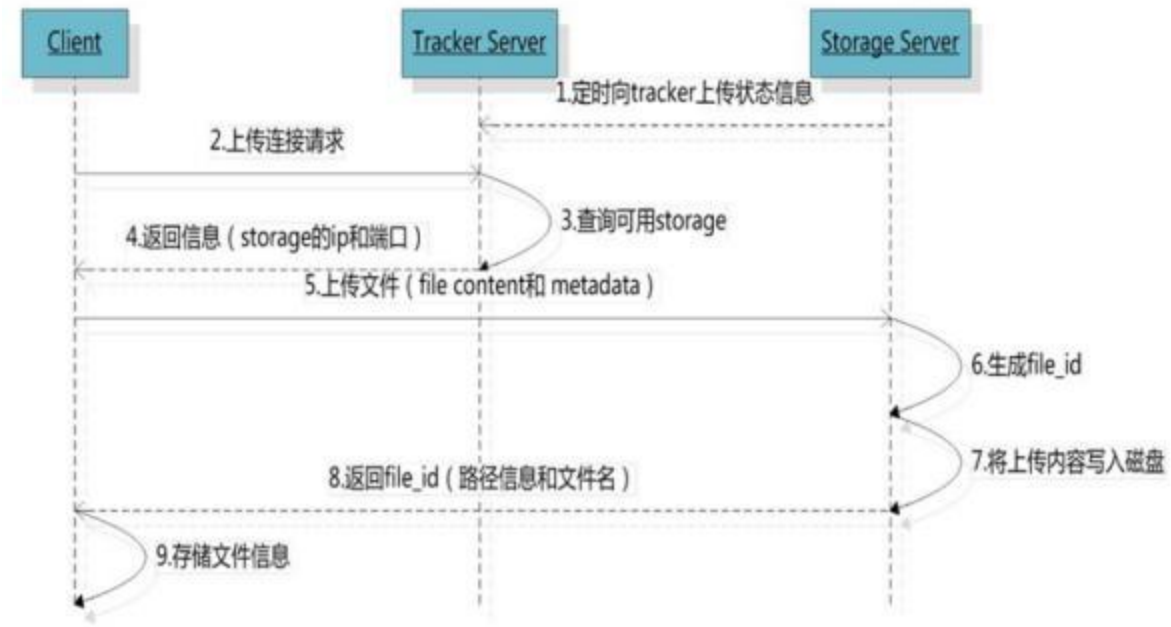
storage接受到写文件请求时， 会根据配置好的规则，选择其中一个存储目录来存储文件。为了避免单个目录下的文件数太多，在storage第一次启动时， 会在每个数据存储目录里创建2级子目录， 每级256个，总共65536个文件，新写的文件会以hash的方式被路由到其中某个子目录下，然后将文件数据直接作为一个本地文件存储到该目录中。

1.3 Client

FastDFS向使用者提供基本文件访问接口，比如monitor、 upload、download、append、delete等，以客户端库的方式提供给用户使用。

2 FastDFS各功能逻辑分析

2.1 upload file原理



选择tracker server

当集群中不止一个tracker server时，由于tracker之间是完全对等的关系，客户端在upload文件时可以任意选择一个trakcer。

选择存储的group

当tracker接收到upload file的请求时，会为该文件分配一个可以存储该文件的group，支持如下选择group的规则：

1. Round robin，所有的group间轮询
2. Specified group，指定某一个确定的group
3. Load balance，选择最大剩余空 间的组上传文件

选择storage server

当选定group后， tracker会在group内选择一个storage server给客户端，支持如下选择storage的规则：

1. Round robin，在group内的所有storage间轮询
2. First server ordered by ip，按ip排序
3. First server ordered by priority，按优先级排序（优先级在storage上配置）

选择storage path

当分配好storage server后，客户端将向storage发送写文件请求， storage将会为文件分配一个数据存储目录，支持如下规则：

1. Round robin，多个存储目录间轮询
2. 剩余存储空间最多的优先

生成Fileid

选定存储目录之后，storage会为文件生一个Fileid，由： storage server ip、文件创建时间、文件大小、文件crc32和一个随机数拼接而成，然后将这个二进制串进行base64编码，转换为可打印的字符串。

选择两级目录

当选定存储目录之后， storage会为文件分配一个fileid，每个存储目录下有两级256*256的子目录，storage会按文件fileid进行两次hash（猜测），路由到其中一个子目录，然后将文件以fileid为文件名存储到该子目录下。

生成文件名

当文件存储到某个子目录后，即认为该文件存储成功，接下来会为该文件生成一个文件名，文件名由：group、存储目录、两级子目录、fileid、文件后缀名（由客户端指定，主要用于区分文件类型）拼接而成。

文件名规则：

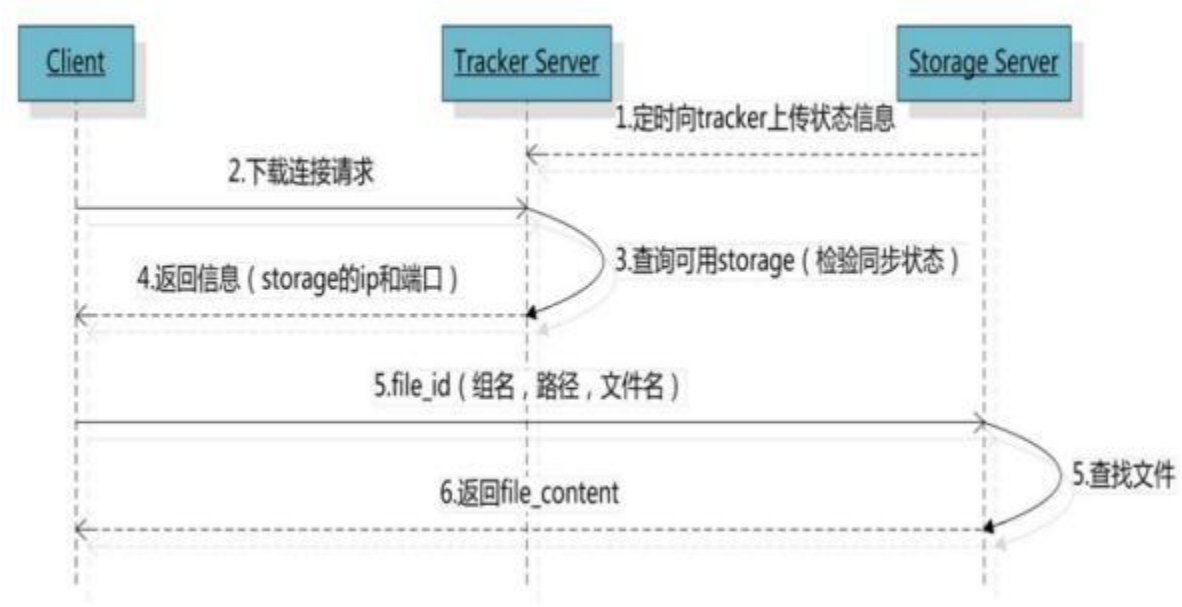
- storage_id（ip的数值型）源storage server ID或IP地址
- timestamp（文件创建时间戳）
- file_size（若原始值为32位则前面加入一个随机值填充，最终为64位）
- crc32 （文件内容的检验码）

随机数 （引入随机数的目的是防止生成重名文件）

```
eBuDxWCb2qmAQ89yAAAAKeR1iIo162
| 4bytes  | 4bytes     | 8bytes     |4bytes  | 2bytes  |
| ip      | timestamp  | file_size  |crc32   | 校验值  |
```

2.2 download file逻辑

客户端upload file成功后，会拿到一个storage生成的文件名，接下来客户端根据这个文件名即可访问到该文件。



跟upload file一样，在download file时客户端可以选择任意tracker server。

tracker发送download请求给某个tracker，必须带上文件名信息， tracke从文件名中解析出文件的group、大小、创建时间等信息，然后为该请求选择一个storage用来服务读请求。由于group内的文件同步时在后台异步进行的，所以有可能出现在读到时候，文件还没有同步到某些storage server上，为了尽量避免访问到这样的storage，tracker按照如下规则选择group内可读的storage。

1. 该文件上传到的源头storage -源头storage只要存活着，肯定包含这个文件，源头的地址被编码在文件名中。
2. 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）-文件创建后，认为经过最大同步时间后，肯定已经同步到其他storage了。
3. 文件创建时间戳 < storage被同步到的时间戳。 - 同步时间戳之前的文件确定已经同步了
4. (当前时间-文件创建时间戳）> 同步延迟阀值（如一天）。 - 经过同步延迟阈值时间，认为文件肯定已经同步了。

2.3 HTTP下载逻辑

FastDFS自带的http服务已经弃用，需要通过nginx + fastdfs-nginx-module的方式去实现下载。

2.4 同步机制

同一组内storage server之间是对等的，文件上传、下载、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，即源服务器同步给目标服务器；