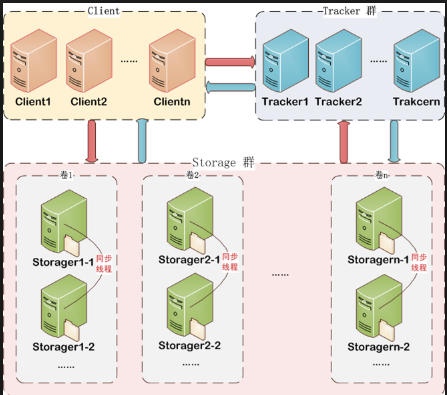
## FastDFS分布式文件系统

FastDFS是一个开源的轻量级分布式文件系统，由跟踪服务器(Tracker Server)、存储服务器(Storage Server)和客户端(Client)组成，主要 解决了海量数据存储问题，特别适合以中小文件(4KB---500MB)的在线服务, 应用于web网站服务存储图片

## FastDFS体系结构



Tracker server 跟踪服务器

Tracker是FastDFS的协调者，类似于调度，起负载均衡作用。负责管理所有的storage server和group，每个storage server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。

Storage server 存储服务器

Storage server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）

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个子目录，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。

## FastDFS的选举规则

### 1、后端多个组的选举规则

当tracker接收到upload的请求时，会为该文件分配一个可以存储该文件的group；选择group的规则

a、round robin

b、指定某一个具体的组

c、选择剩余存储空间多的group优先

### 2、选举同一个组中的某台storage server

当选定group后，tracker会在group内选择一个storage server给客户端；选择storage server的规则如下

a、round robin

b、按ip地址排序，小的优先

c、按优先级排序

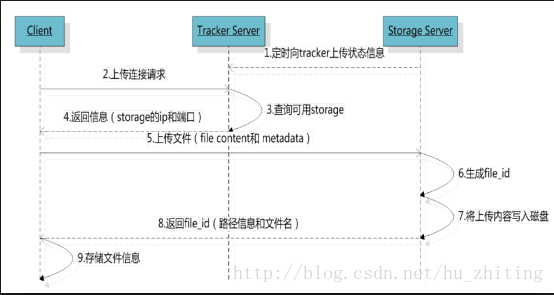
### 3、选举某台storage server的存储目录

当选择好storage server后，客户端向storage发送写请求，storage将会为文件分配一个数据存储目录；选择存储目录的规则如下：

a、round robin

b、剩余存储空间最多的优先

## FastDFS存储文件的流程

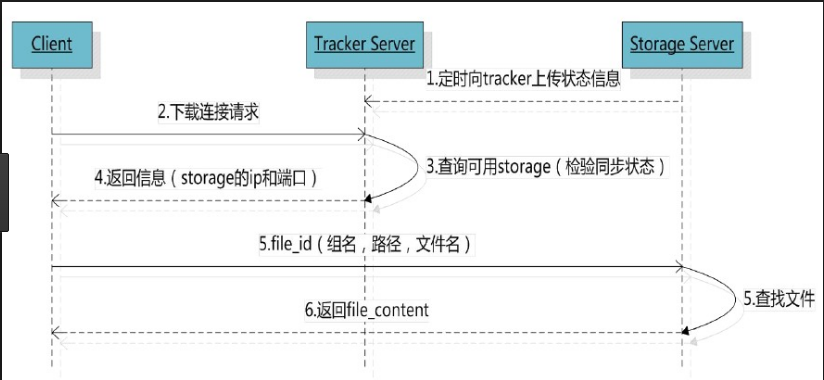


file id文件id格式

组名/存储目录映射名称/一级目录名称/二级目录名称/随机数

group01/M00/00/11/XXXXXXXXXXXXXXXX

## FastDFS读取文件的流程



## 部署FastDFS

环境描述

node01 192.168.122.101 Tracker Server

node02 192.168.122.102 Tracker Server

node03 192.168.122.103 Storage Server

node04 192.168.122.104 Storage Server

一、在所有节点安装fastdfs软件

安装libfastcommon库

[root@node01 ~]# git clone https://github.com/happyfish100/libfastcommon.git

[root@node01 ~]# cd libfastcommon/

[root@node01 libfastcommon]# ./make.sh

[root@node01 libfastcommon]# ./make.sh install

安装fastdfs

[root@node01 ~]# cd fastdfs/

[root@node01 fastdfs]# ./make.sh

[root@node01 fastdfs]# ./make.sh install

[root@node01 ~]# ls /etc/fdfs/

client.conf.sample storage.conf.sample storage\_ids.conf.sample tracker.conf.sample

[root@node01 ~]# ls /etc/init.d/

fdfs\_storaged fdfs\_trackerd functions netconsole network README

[root@node01 ~]#

二、配置Tracker Server

[root@node01 ~]# cd /etc/fdfs/

[root@node01 fdfs]# cp tracker.conf.sample tracker.conf

[root@node01 fdfs]# vim /etc/fdfs/tracker.conf

bind\_addr=192.168.122.101

base\_path=/data/tracker

[root@node01 fdfs]# mkdir -p /data/tracker

启动tracker服务

[root@node01 ~]# /etc/init.d/fdfs\_trackerd start

[root@node01 ~]# ss -antp | grep fdfs

LISTEN 0 128 192.168.122.101:22122 \*:\* users:(("fdfs\_trackerd",pid=2089,fd=5))

[root@node01 ~]# ps aux | grep fdfs

root 2089 0.0 0.1 79988 1664 ? Sl 14:07 0:00 /usr/bin/fdfs\_trackerd /etc/fdfs/tracker.conf

root 2104 0.0 0.0 112664 972 pts/0 S+ 14:07 0:00 grep --color=auto fdfs

[root@node01 ~]#

[root@node01 ~]# ls /data/tracker/

data logs

[root@node01 ~]# ls /data/tracker/logs/

trackerd.log

[root@node01 ~]# ls /data/tracker/data/

fdfs\_trackerd.pid storage\_changelog.dat

同样的方法配置node02 tracker server

三、配置Storage Server

[root@node03 ~]# cd /etc/fdfs/

[root@node03 fdfs]# cp storage.conf.sample storage.conf

[root@node03 ~]# vim /etc/fdfs/storage.conf

group\_name=group01

bind\_addr=192.168.122.103

base\_path=/data/storage

store\_path\_count=1

store\_path0=/fdfs/data

tracker\_server=192.168.122.101:22122

tracker\_server=192.168.122.102:22122

[root@node03 ~]# mkdir -p /data/storage

[root@node03 ~]# mkdir -p /fdfs/data

[root@node03 logs]# ps aux | grep fdfs

root 1444 1.0 6.5 82668 66836 ? Sl 14:23 0:00 /usr/bin/fdfs\_storaged /etc/fdfs/storage.conf

root 1487 0.0 0.0 112664 968 pts/0 S+ 14:25 0:00 grep --color=auto fdfs

[root@node03 logs]# ss -antp | grep fdfs

LISTEN 0 128 192.168.122.103:23000 \*:\* users:(("fdfs\_storaged",pid=1444,fd=5))

ESTAB 0 0 192.168.122.103:60922 192.168.122.102:22122 users:(("fdfs\_storaged",pid=1444,fd=21))

ESTAB 0 0 192.168.122.103:59021 192.168.122.101:22122 users:(("fdfs\_storaged",pid=1444,fd=20))

[root@node03 ~]# ls /data/storage/

data logs

[root@node03 ~]# ls /fdfs/data/data/

00 09 12 1B 24 2D 36 3F 48 51 5A 63 6C 75 7E 87 90 99 A2 AB B4 BD C6 CF D8 E1 EA F3 FC

01 0A 13 1C 25 2E 37 40 49 52 5B 64 6D 76 7F 88 91 9A A3 AC B5 BE C7 D0 D9 E2 EB F4 FD

02 0B 14 1D 26 2F 38 41 4A 53 5C 65 6E 77 80 89 92 9B A4 AD B6 BF C8 D1 DA E3 EC F5 FE

03 0C 15 1E 27 30 39 42 4B 54 5D 66 6F 78 81 8A 93 9C A5 AE B7 C0 C9 D2 DB E4 ED F6 FF

04 0D 16 1F 28 31 3A 43 4C 55 5E 67 70 79 82 8B 94 9D A6 AF B8 C1 CA D3 DC E5 EE F7

05 0E 17 20 29 32 3B 44 4D 56 5F 68 71 7A 83 8C 95 9E A7 B0 B9 C2 CB D4 DD E6 EF F8

06 0F 18 21 2A 33 3C 45 4E 57 60 69 72 7B 84 8D 96 9F A8 B1 BA C3 CC D5 DE E7 F0 F9

07 10 19 22 2B 34 3D 46 4F 58 61 6A 73 7C 85 8E 97 A0 A9 B2 BB C4 CD D6 DF E8 F1 FA

08 11 1A 23 2C 35 3E 47 50 59 62 6B 74 7D 86 8F 98 A1 AA B3 BC C5 CE D7 E0 E9 F2 FB

同样的方法配置node04 storage server

四、配置客户端测试数据读写

将node01配置为客户端作为测试

[root@node01 ~]# cd /etc/fdfs/

[root@node01 fdfs]# cp client.conf.sample client.conf

[root@node01 fdfs]# vim /etc/fdfs/client.conf

base\_path=/data/client

tracker\_server=192.168.122.101:22122

tracker\_server=192.168.122.102:22122

[root@node01 ~]# mkdir -p /data/client

上传文件

[root@node01 ~]# echo "hello fdfs" > /tmp/1.txt

[root@node01 ~]# fdfs\_upload\_file /etc/fdfs/client.conf /tmp/1.txt

group01/M00/00/00/wKh6Z1nN6q2AUUU9AAAAC\_UDvNc161.txt

查看文件信息

[root@node01 ~]# fdfs\_file\_info /etc/fdfs/client.conf group01/M00/00/00/wKh6Z1nN6q2AUUU9AAAAC\_UDvNc161.txt

source storage id: 0

source ip address: 192.168.122.103

file create timestamp: 2017-09-29 14:39:41

file size: 11

file crc32: 4110662871 (0xF503BCD7)

下载文件

[root@node01 ~]# fdfs\_download\_file /etc/fdfs/client.conf group01/M00/00/00/wKh6Z1nN6q2AUUU9AAAAC\_UDvNc161.txt

删除文件

[root@node01 ~]# fdfs\_delete\_file /etc/fdfs/client.conf group01/M00/00/00/wKh6Z1nN6q2AUUU9AAAAC\_UDvNc161.txt

## 整合fastdfs + nginx

目的：

1、通过URL地址的方式访问文件内容

2、

FastDFS 通过 Tracker 服务器，将文件放在 Storage 服务器存储，但是同组存储服务器之间需要进入文件复制流程，有同步延迟的问题。假设 Tracker 服务器将文件上传到了 192.168.122.101，上传成功后文件 ID已经返回给客户端。此时 FastDFS 存储集群机制会将这个文件同步到同组存储 192.168.122.102，在文件还没有复制完成的情况下，客户端如果用这个文件 ID 在 192.168.122.102上取文件，就会出现文件无法访问的错误。而 fastdfs-nginx-module 可以重定向文件连接到源服务器（192.168.122.101）上取文件，避免客户端由于复制延迟导致的文件无法访问错误。

### 在所有storage节点上安装nginx及fastdfs-nginx-module模块

1、安装nginx

[root@node04 ~]# yum install -y pcre-devel openssl-devel

[root@node04 ~]# unzip master.zip

[root@node04 ~]# mkdir /app

[root@node04 ~]# tar xf nginx-1.11.10.tar.gz

[root@node04 ~]# cd nginx-1.11.10/

[root@node04 nginx-1.11.10]# ./configure --prefix=/app/nginx --add-module=/root/fastdfs-nginx-module-master/src/

[root@node04 nginx-1.11.10]# make && make install

2、复制fastdfs-nginx-module中的配置文件到/etc/fdfs目录

[root@node04 ~]# cd fastdfs-nginx-module-master/src/

[root@node04 src]# cp mod\_fastdfs.conf /etc/fdfs/

[root@node04 ~]# vim /etc/fdfs/mod\_fastdfs.conf

tracker\_server=192.168.122.101:22122

tracker\_server=192.168.122.102:22122

group\_name=group01

url\_have\_group\_name = true

store\_path\_count=1

store\_path0=/fdfs/data

group\_count = 0

[group1]

group\_name=group01

storage\_server\_port=23000

store\_path\_count=1

store\_path0=/fdfs/data

2、复制fdfs源码目录中的关于http的配置文件

[root@node04 ~]# cd fastdfs/conf/

[root@node04 conf]# cp mime.types http.conf /etc/fdfs/

3、创建数据存放目录的软链接

[root@node04 ~]# ln -s /fdfs/data/data/ /fdfs/data/M00

4、编辑nginx配置文件, 并启动

[root@node04 ~]# vim /app/nginx/conf/nginx.conf

server {

listen 8888;

server\_name localhost;

location ~ /group[0-9][0-9]/M00 {

ngx\_fastdfs\_module;

}

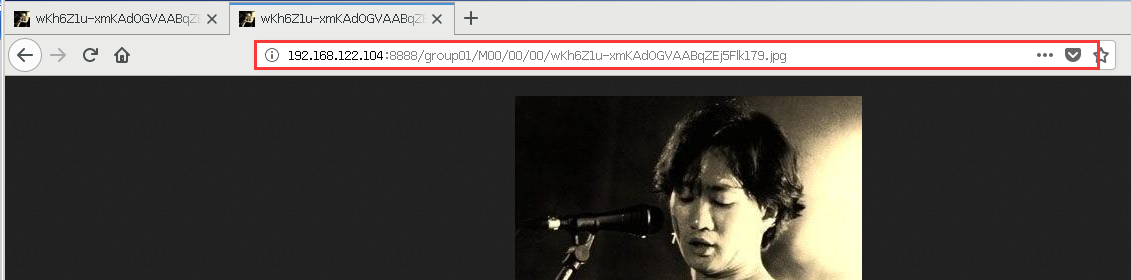
....

}

[root@node04 ~]# /app/nginx/sbin/nginx

ngx\_http\_fastdfs\_set pid=13331

通过浏览器访问文件内容



### 在Tracker节点上安装nginx

主要为了提供 http 访问的反向代理、负载均衡以及缓存服务。

1、安装nginx

[root@node02 ~]# mkdir /app

[root@node02 ~]# tar xf nginx-1.11.10.tar.gz

[root@node02 ~]# cd nginx-1.11.10/

[root@node02 nginx-1.11.10]# ./configure --prefix=/app/nginx

[root@node02 nginx-1.11.10]# make && make install

2、编辑nginx配置文件实现负载均衡

[root@node02 ~]# vim /app/nginx/conf/nginx.conf

upstream fdfs\_group01 {

server 192.168.122.103:8888 weight=1 max\_fails=2 fail\_timeout=2;

server 192.168.122.104:8888 weight=1 max\_fails=2 fail\_timeout=2;

}

server {

listen 8000;

server\_name localhost;

location /group01/M00 {

proxy\_pass http://fdfs\_group01;

}

...

}

[root@node02 ~]# /app/nginx/sbin/nginx

[root@node02 ~]# ss -antp | grep nginx

LISTEN 0 128 \*:8000 \*:\* users:(("nginx",pid=13943,fd=6),("nginx",pid=13942,fd=6))

3、测试通过tracker访问文件内容



https://mp.weixin.qq.com/s?\_\_biz=MzI0MDQ4MTM5NQ==&mid=2247487212&idx=1&sn=86761ca96f15ef23e5a8d4106714c767&chksm=e91b6bf0de6ce2e65d966fa781d8d570ca8a4bf2ebdae591b5b8a4402ca692513aec6655e328&mpshare=1&scene=23&srcid=1011x6dzqsVeU2iJTHFN7IW1#rd