# **[用FastDFS一步步搭建文件管理系统](https://www.cnblogs.com/chiangchou/p/fastdfs.html)**

# **一、FastDFS介绍**

FastDFS开源地址：[https://github.com/happyfish100](https://github.com/happyfish100" \t "https://www.cnblogs.com/chiangchou/p/_blank)

参考：[分布式文件系统FastDFS设计原理](http://blog.chinaunix.net/uid-20196318-id-4058561.html" \t "https://www.cnblogs.com/chiangchou/p/_blank)

参考：[FastDFS分布式文件系统](http://www.cnblogs.com/Leo_wl/p/6731647.html" \t "https://www.cnblogs.com/chiangchou/p/_blank)

个人封装的FastDFS Java API：[https://github.com/bojiangzhou/lyyzoo-fastdfs-java](https://github.com/bojiangzhou/lyyzoo-fastdfs-java" \t "https://www.cnblogs.com/chiangchou/p/_blank)

### **1、简介**

FastDFS 是一个开源的高性能分布式文件系统（DFS）。 它的主要功能包括：文件存储，文件同步和文件访问，以及高容量和负载平衡。主要解决了海量数据存储问题，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。

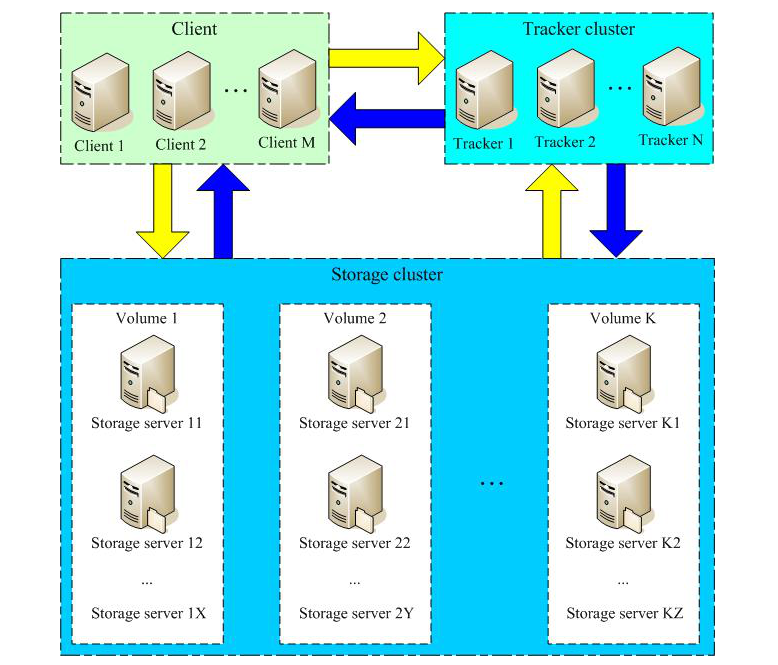
FastDFS 系统有三个角色：跟踪服务器(Tracker Server)、存储服务器(Storage Server)和客户端(Client)。

**Tracker Server**：跟踪服务器，主要做调度工作，起到均衡的作用；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 group 等信息，并保持周期性心跳。

**Storage Server**：存储服务器，主要提供容量和备份服务；以 group 为单位，每个 group 内可以有多台 storage server，数据互为备份。

**Client**：客户端，上传下载数据的服务器，也就是我们自己的项目所部署在的服务器。

## **二、安装FastDFS环境**



### ****2、FastDFS的存储策略****

为了支持大容量，存储节点（服务器）采用了分卷（或分组）的组织方式。存储系统由一个或多个卷组成，卷与卷之间的文件是相互独立的，所有卷的文件容量累加就是整个存储系统中的文件容量。一个卷可以由一台或多台存储服务器组成，一个卷下的存储服务器中的文件都是相同的，卷中的多台存储服务器起到了冗余备份和负载均衡的作用。

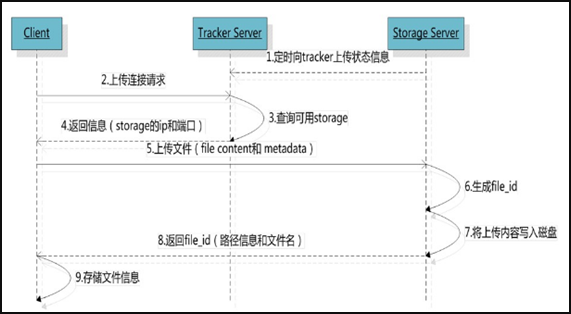
在卷中增加服务器时，同步已有的文件由系统自动完成，同步完成后，系统自动将新增服务器切换到线上提供服务。当存储空间不足或即将耗尽时，可以动态添加卷。只需要增加一台或多台服务器，并将它们配置为一个新的卷，这样就扩大了存储系统的容量。

### ****3、FastDFS的上传过程****

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

Storage Server会定期的向Tracker Server发送自己的存储信息。当Tracker Server Cluster中的Tracker Server不止一个时，各个Tracker之间的关系是对等的，所以客户端上传时可以选择任意一个Tracker。

当Tracker收到客户端上传文件的请求时，会为该文件分配一个可以存储文件的group，当选定了group后就要决定给客户端分配group中的哪一个storage server。当分配好storage server后，客户端向storage发送写文件请求，storage将会为文件分配一个数据存储目录。然后为文件分配一个fileid，最后根据以上的信息生成文件名存储文件。



### ****4、FastDFS的文件同步****

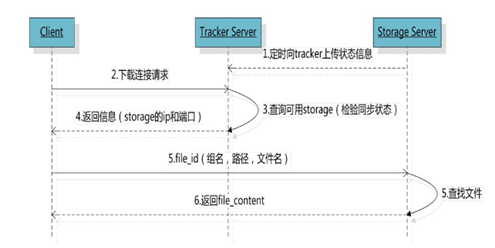
写文件时，客户端将文件写至group内一个storage server即认为写文件成功，storage server写完文件后，会由后台线程将文件同步至同group内其他的storage server。

每个storage写文件后，同时会写一份binlog，binlog里不包含文件数据，只包含文件名等元信息，这份binlog用于后台同步，storage会记录向group内其他storage同步的进度，以便重启后能接上次的进度继续同步；进度以时间戳的方式进行记录，所以最好能保证集群内所有server的时钟保持同步。

storage的同步进度会作为元数据的一部分汇报到tracker上，tracke在选择读storage的时候会以同步进度作为参考。

### ****5、FastDFS的文件下载****

客户端uploadfile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downloadfile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。

## **二、安装FastDFS环境**

### **0、前言**

操作环境：CentOS7 X64，以下操作都是单机环境。

我把所有的安装包下载到/software/下，解压到当前目录。

先做一件事，修改hosts，将文件服务器的ip与域名映射(单机TrackerServer环境)，因为后面很多配置里面都需要去配置服务器地址，ip变了，就只需要修改hosts即可。

# vim /etc/hosts

增加如下一行，这是我的IP

192.168.224.11 server1.com

如果要本机访问虚拟机，在C:\Windows\System32\drivers\etc\hosts中同样增加一行

### **编译环境**

### **yum install git gcc gcc-c++ make automake autoconf libtool pcre pcre-devel zlib zlib-devel openssl-devel wget vim -y**

### **1、下载安装 libfastcommon**

libfastcommon是从 FastDFS 和 FastDHT 中提取出来的公共 C 函数库，基础环境，安装即可

1. 下载libfastcommon

wget https://github.com/happyfish100/libfastcommon/archive/V1.0.7.tar.gz

1. 解压

tar -zxvf V1.0.7.tar.gz

cd libfastcommon-1.0.7

3,编译、安装

./make.sh

./make.sh install

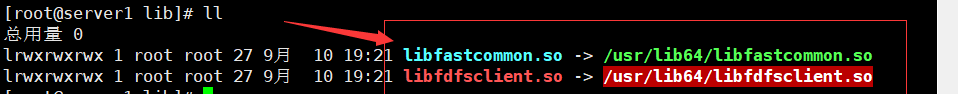
④ libfastcommon.so 安装到了/usr/lib64/libfastcommon.so，但是FastDFS主程序设置的lib目录是/usr/local/lib，所以需要创建软链接。

ln -s /usr/lib64/libfastcommon.so /usr/local/lib/libfastcommon.so

ln -s /usr/lib64/libfastcommon.so /usr/lib/libfastcommon.so

ln -s /usr/lib64/libfdfsclient.so /usr/local/lib/libfdfsclient.so

ln -s /usr/lib64/libfdfsclient.so /usr/lib/libfdfsclient.so



cd /software 目录

### **2、下载安装FastDFS**

① 下载FastDFS

wget https://github.com/happyfish100/fastdfs/archive/V5.05.tar.gz

② 解压

# tar -zxvf V5.05.tar.gz

# cd fastdfs-5.05

③ 编译、安装

# ./make.sh

# ./make.sh install

④ 默认安装方式安装后的相应文件与目录  
　　A、服务脚本：

/etc/init.d/fdfs\_storaged

/etc/init.d/fdfs\_tracker

　　B、配置文件（这三个是作者给的样例配置文件） :

/etc/fdfs/client.conf.sample

/etc/fdfs/storage.conf.sample

/etc/fdfs/tracker.conf.sample

1. 命令工具在 /usr/bin/ 目录下



⑤ FastDFS 服务脚本设置的 bin 目录是 /usr/local/bin， 但实际命令安装在 /usr/bin/ 下。可以把/usr/bin/下的服务脚本文件软连接到/usr/local/bin 下，解决问题。

　　两种方式：

　　》 一是修改FastDFS 服务脚本中相应的命令路径，也就是把 /etc/init.d/fdfs\_storaged 和 /etc/init.d/fdfs\_tracker 两个脚本中的 /usr/local/bin 修改成 /usr/bin。

 　　　　# vim fdfs\_trackerd  
　　　　使用查找替换命令进统一修改:%s+/usr/local/bin+/usr/bin  
　　　　# vim fdfs\_storaged  
　　　　使用查找替换命令进统一修改:%s+/usr/local/bin+/usr/bin

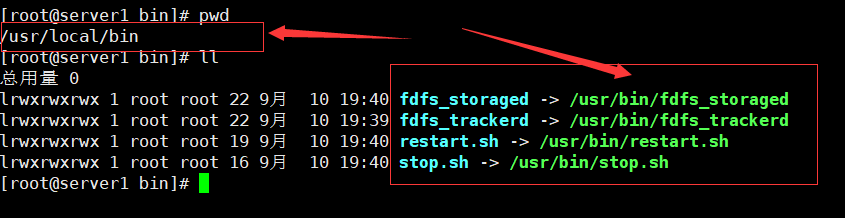
》 二是建立 /usr/bin 到 /usr/local/bin 的软链接，我是用这种方式。

ln -s /usr/bin/fdfs\_trackerd /usr/local/bin

ln -s /usr/bin/fdfs\_storaged /usr/local/bin

ln -s /usr/bin/stop.sh /usr/local/bin

ln -s /usr/bin/restart.sh /usr/local/bin



### **3、配置FastDFS跟踪器(Tracker)**

① 进入 /etc/fdfs，复制 FastDFS 跟踪器样例配置文件 tracker.conf.sample，并重命名为 tracker.conf。

# cd /etc/fdfs

# cp tracker.conf.sample tracker.conf

# vim tracker.conf

1. 编辑tracker.conf ，标红的需要修改下，其它的默认即可。

配置文件是否不生效，false 为生效

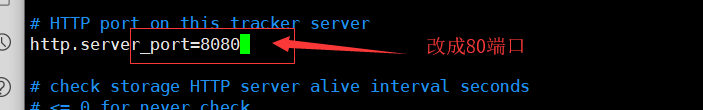
disabled=false

# 提供服务的端口

port=22122

# Tracker 数据和日志目录地址(根目录必须存在,子目录会自动创建)base\_path=/data/fastdfs/tracker

# HTTP 服务端口http.server\_port=80



1. 创建tracker基础数据目录，即base\_path对应的目录

mkdir -p /data/fastdfs/tracker

1. 防火墙中打开跟踪端口（默认的22122）

添加如下端口行：

iptables -A INPUT -m state --state NEW -p tcp -m tcp --dport 22122 -j ACCEPT启动Tracker

初次成功启动，会在 /data/fdfsdfs/tracker/ (配置的base\_path)下创建 data、logs 两个目录。

可以用这种方式启动

# /etc/init.d/fdfs\_trackerd start

也可以用这种方式启动，前提是上面创建了软链接，后面都用这种方式

# service fdfs\_trackerd start

查看 FastDFS Tracker 是否已成功启动 ，22122端口正在被监听，则算是Tracker服务安装成功

netstat -putnel|grep fdfs



关闭Tracker命令：

# service fdfs\_trackerd stop

⑥ 设置Tracker开机启动

# chkconfig fdfs\_trackerd on  
  
或者：  
# vim /etc/rc.d/rc.local  
加入配置：  
/etc/init.d/fdfs\_trackerd star

⑦ tracker server 目录及文件结构

Tracker服务启动成功后，会在base\_path下创建data、logs两个目录。目录结构如下：

${base\_path}

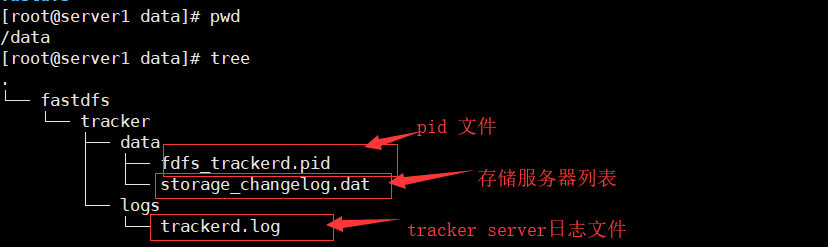
|\_\_data

| |\_\_storage\_groups.dat：存储分组信息

| |\_\_storage\_servers.dat：存储服务器列表

|\_\_logs

| |\_\_trackerd.log： tracker server 日志文件



### **4、配置 FastDFS 存储 (Storage) 多机版，把上面安装的1-2的步骤在Storaga服务器做一遍**

① 进入 /etc/fdfs 目录，复制 FastDFS 存储器样例配置文件 storage.conf.sample，并重命名为 storage.conf

cd /etc/fdfs

cp storage.conf.sample storage.conf  
vim storage.conf

② 编辑storage.conf

标红的需要修改，其它的默认即可。

配置文件是否不生效，false 为生效

disabled=false

# 指定此 storage server 所在 组(卷)

group\_name=group1

# storage server 服务端口

port=23000

# 心跳间隔时间，单位为秒 (这里是指主动向 tracker server 发送心跳)

heart\_beat\_interval=30

# Storage 数据和日志目录地址(根目录必须存在，子目录会自动生成)base\_path=/dada/fastdfs/storage

# 存放文件时 storage server 支持多个路径。这里配置存放文件的基路径数目，通常只配一个目录。

store\_path\_count=1

# 逐一配置 store\_path\_count 个路径，索引号基于 0。

# 如果不配置 store\_path0，那它就和 base\_path 对应的路径一样。

：109 行 store\_path0=/data/fastdfs/file

# FastDFS 存储文件时，采用了两级目录。这里配置存放文件的目录个数。

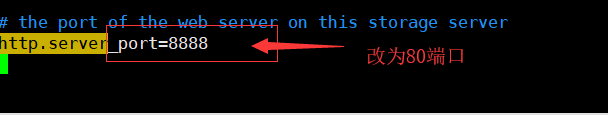
# 如果本参数只为 N（如： 256），那么 storage server 在初次运行时，会在 store\_path 下自动创建 N \* N 个存放文件的子目录。

subdir\_count\_per\_path=256

# tracker\_server 的列表 ，会主动连接 tracker\_server 可以写ip地址 ：118行

# 有多个 tracker server 时，每个 tracker server 写一行tracker\_server=server1.com:22122  
  
# 允许系统同步的时间段 (默认是全天) 。一般用于避免高峰同步产生一些问题而设定。  
sync\_start\_time=00:00  
sync\_end\_time=23:59

# 访问端口  
http.server\_port=80



③ 创建Storage基础数据目录，对应base\_path目录

mkdir -p /data/fastdfs/storage

# 这是配置的store\_path0路径

# mkdir -p /data/fastdfs/file

④ 防火墙中打开存储器端口（默认的 23000）

添加如下端口行：

iptables -A INPUT -m state --state NEW -p tcp -m tcp --dport 23000 -j ACCEPT

多机版需要添加iptables -A INPUT -p tcp -m multiport --sports 22122 -j ACCEPT

⑤ 启动 Storage

启动Storage前确保Tracker是启动的。初次启动成功，会在 /ljzsg/fastdfs/storage 目录下创建 data、 logs 两个目录。

可以用这种方式启动

# /etc/init.d/fdfs\_storaged start

也可以用这种方式，后面都用这种

# service fdfs\_storaged start

查看 Storage 是否成功启动，23000 端口正在被监听，就算 Storage 启动成功。

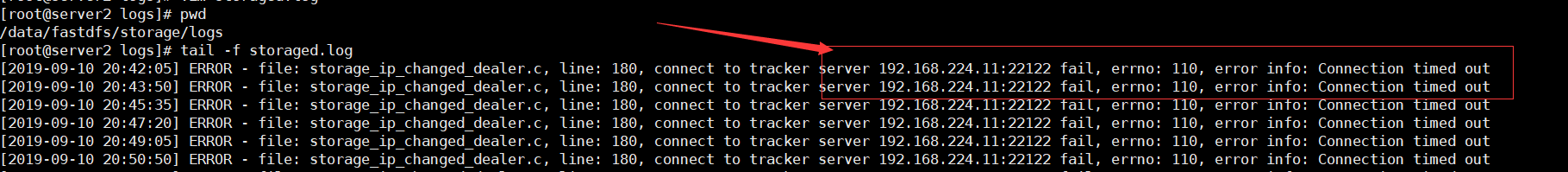
# netstat -unltp|grep fdfs



关闭Storage命令：

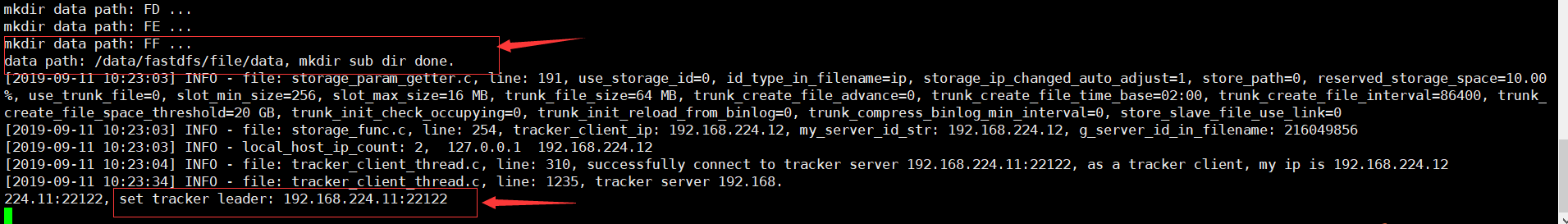
service fdfs\_storaged stop

如果启动失败，看下storage的日志，发现tracker 节点22122防火墙连不上。去设置下,让来源22122端口的都通过

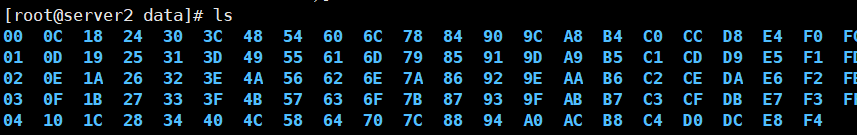


[root@server2 ~]# iptables -A INPUT -p tcp -m multiport --sports 22122 -j ACCEPT

然后就可以查看日志，看到如下信息，tail -f /data/fastdfs/storage/logs/storaged.log

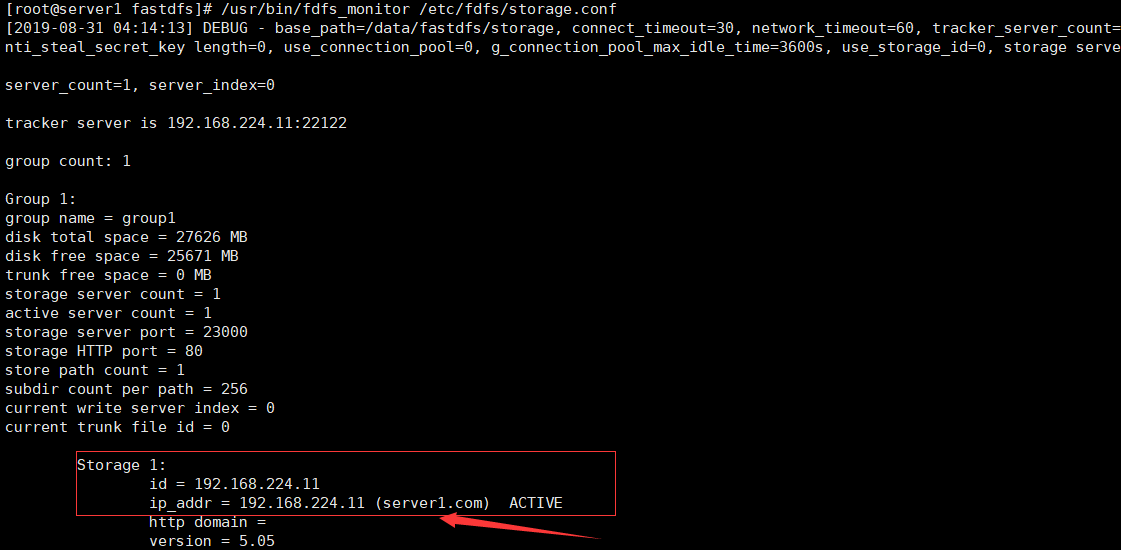


这是会发现存储数据目录已经自动创建了，但是目录里面目前没有数据，



查看Storage和Tracker是否在通信：

/usr/bin/fdfs\_monitor /etc/fdfs/storage.conf



⑥ 设置 Storage 开机启动

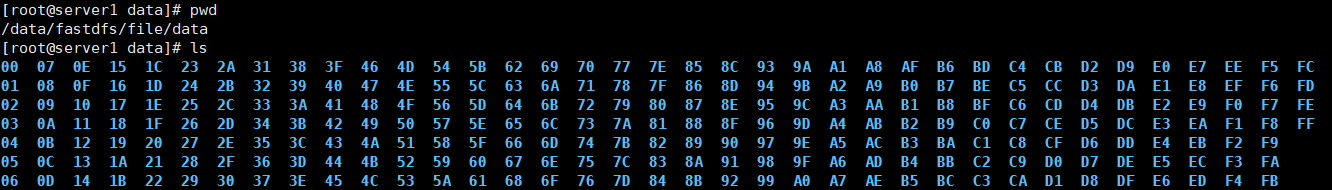
# chkconfig fdfs\_storaged on

或者：  
# vim /etc/rc.d/rc.local  
加入配置：  
/etc/init.d/fdfs\_storaged start

⑦ Storage 目录

同 Tracker，Storage 启动成功后，在base\_path 下创建了data、logs目录，记录着 Storage Server 的信息。

在 store\_path0 目录下，创建了N\*N个子目录：



### **文件上传测试**

### 由于目前还没有搭建完集群，因此我们暂且在tracker的一台设备上（我们就选择192.168.224.11这台设备）使用client来上传图片进行测试。

① 修改 Tracker 服务器中的客户端配置文件

# cd /etc/fdfs

# cp client.conf.sample client.conf

# vim client.conf

# Client 的数据和日志目录base\_path=/data/fastdfs/client

# Tracker端口tracker\_server=server1.com:22122

创建目录

mkdir -p /data/fastdfs/client

② 上传测试

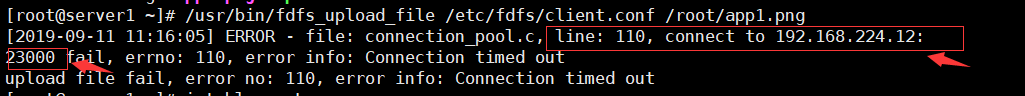
 在linux内部执行如下命令上传 app1.png 图片

[root@server1 ~]# /usr/bin/fdfs\_upload\_file /etc/fdfs/client.conf /root/app1.png

下面我们来上传一张图片，我把/root/目录下一张app1.png图片上传，使用的命令：/usr/bin/fdfs\_upload\_file  /etc/fdfs/client.conf /root/app1.png，可以看到这条命令由3部分组成，第一部分是/usr/bin/fdfs\_upload\_file，意思是指定要进行上传文件操作，第二部分是/etc/fdfs/client.conf，意思是指定上传操作使用的配置文件，这个配置文件就是我们上面刚配置过的client.conf文件，第三部分是/root/app1.png，意思是指定要上传哪个目录下的哪个文件。按回车执行上传命令后，会返回一个串：group1/M00/00/00/wKjgDF14eNKAb0NiAADQmXv4gHY384.png，其中group1表示这张图片被保存在了哪个组当中，M00代表磁盘目录，如果电脑只有一个磁盘那就只有M00， 如果有多个磁盘，那就M01、M02...等等。00/00代表磁盘上的两级目录，每级目录下是从00到FF共256个文件夹，两级就是256\*256个。wKjgDF14eNKAb0NiAADQmXv4gHY384.png表示被存储到storage上的app1.png被重命名的名字，这样做的目的是为了防止图片名字重复。

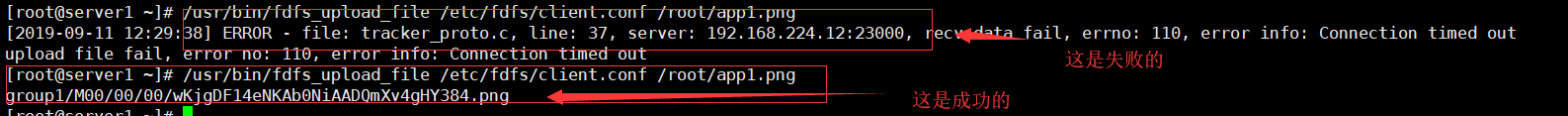
上传成功后返回文件ID号：group1/M00/00/00/wKjgDF14eNKAb0NiAADQmXv4gHY384.png

如果报错误，说明防火墙导致连接失败，

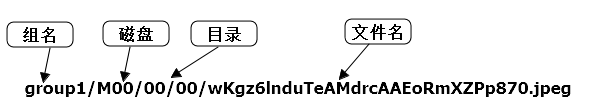


配置storage服务器的防火墙规则就可以了

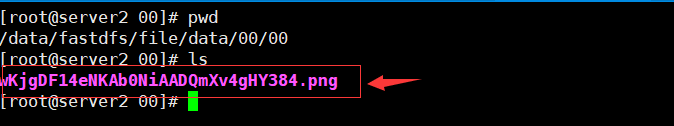
[root@server2 storage]# iptables -A INPUT -p tcp -m multiport --dports 23000 -j ACCEPT



返回的文件ID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



上传的文件就保存在 storage存储服务器的 /data/fastdfs/file/data/00/00/下



## **三、安装Nginx**

上面将文件上传成功了，但我们无法下载。因此安装Nginx作为服务器以支持Http方式访问文件。同时，后面安装FastDFS的Nginx模块也需要Nginx环境。

Nginx只需要安装到StorageServer所在的服务器即可，用于访问文件。我这里由于是单机，TrackerServer和StorageServer在一台服务器上。多机就把nginx 安装到storageserver服务器上。

### **1、安装nginx所需环境**

yum install -y gcc-c++ pcre pcre-devel zlib zlib-devel openssl openssl-devel

① gcc 安装

# yum install -y gcc-c++

② PCRE pcre-devel 安装

# yum install -y pcre pcre-devel

③ zlib 安装

# yum install -y zlib zlib-devel

④ OpenSSL 安装

# yum install -y openssl openssl-devel

cd /software

### **2、安装Nginx**

① 下载nginx

wget -c https://nginx.org/download/nginx-1.12.1.tar.gz

② 解压

# tar -zxvf nginx-1.12.1.tar.gz

# cd nginx-1.12.1

③ 使用默认配置

./configure

make

make install

1. 启动nginx

cd /usr/local/nginx/sbin/

# ./nginx

其它命令

# ./nginx -s stop

# ./nginx -s quit

# ./nginx -s reload

或者设置软连接

ln -s /usr/local/nginx/sbin/\* /usr/local/sbin/ #制作nginx的软链接

然后直接 nginx 启动 nginx -s stop 停止 nginx -s quit 也是退出

nginx -s reload 重新加载

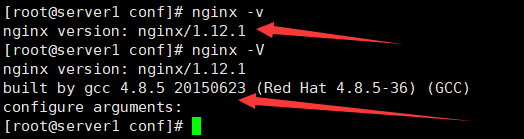
⑥ 设置开机启动

# vim /etc/rc.local

添加一行：

/usr/local/nginx/sbin/nginx  
  
# 设置执行权限  
# chmod 755 /etc/rc.local

1. 查看nginx的版本及模块



⑧ 防火墙中打开Nginx端口（默认的 80）

添加后就能在本机使用80端口访问了。

### **3、访问文件**

简单的测试访问文件

① 修改nginx.conf

[IMG_256](https://www.cnblogs.com/chiangchou/p/javascript:void(0);)

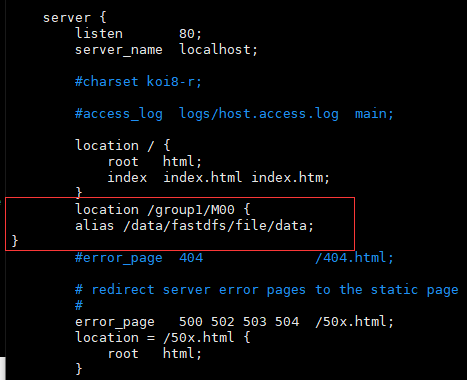
# vim /usr/local/nginx/conf/nginx.conf

#### 添加如下行，将 /group1/M00 映射到 /data/fastdfs/file/data

location /group1/M00 {

alias /data/fastdfs/file/data;

}



# 重启nginx  
nginx -s reload

② 在浏览器访问之前上传的图片、成功。

[http://192.168.224.12:80/group1/M00/00/00/wKjgDF14eNKAb0NiAADQmXv4gHY384.png](http://192.168.224.11:80/group1/M00/00/00/wKjgC11piVmARkD7AAEhr9cPOwk512.png)



## **四、FastDFS 配置 Nginx 模块**

### **1、安装配置Nginx模块**

① fastdfs-nginx-module 模块说明

　　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行文件复制， 有同步延迟的问题。

　　假设 Tracker 服务器将文件上传到了 192.168.224.12，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 192.168.224.13，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 192.168.224.13 上取文件,就会出现文件无法访问的错误。

而 fastdfs-nginx-module 可以重定向文件链接到源服务器取文件，避免客户端由于复制延迟导致的文件无法访问错误。

cd /software

安装fastdfs-nginx的模块

git clone https://github.com/happyfish100/fastdfs-nginx-module.git --depth 1 这个模块和nginx-1.12会有冲突，所以下载下面的模块

cp /software/fastdfs-nginx-module/src/mod\_fastdfs.conf /etc/fdfs/

③ 配置Nginx

在nginx中添加模块

[IMG_256](https://www.cnblogs.com/chiangchou/p/javascript:void(0);)

# 这里为啥这么长一串呢，因为最新版的master与当前nginx有些版本问题。

# wget https://github.com/happyfish100/fastdfs-nginx-module/archive/5e5f3566bbfa57418b5506aaefbe107a42c9fcb1.zip

# 解压

# unzip 5e5f3566bbfa57418b5506aaefbe107a42c9fcb1.zip

# 重命名

# mv fastdfs-nginx-module-5e5f3566bbfa57418b5506aaefbe107a42c9fcb1 fastdfs-nginx-module-master

[IMG_257](https://www.cnblogs.com/chiangchou/p/javascript:void(0);)

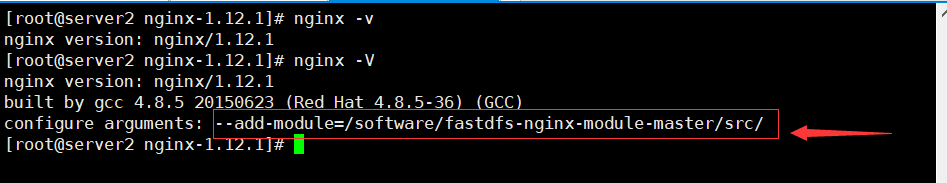
# 先停掉nginx服务  
nginx -s stop  
  
进入解压包目录 # cd /software/nginx-1.12.1/ #

添加模块 # ./configure --add-module=/software/fastdfs-nginx-module-master/src 重新编译、安装 # make && make install

 查看Nginx的模块

# /usr/local/nginx/sbin/nginx -V

有下面这个就说明添加模块成功



⑤ 复制 fastdfs-nginx-module 源码中的配置文件到/etc/fdfs 目录， 并修改

# cd /software/fastdfs-nginx-module-master/src  
  
# cp mod\_fastdfs.conf /etc/fdfs/

vim /etc/fdfs/mod\_fastdfs.conf

# 连接超时时间  
connect\_timeout=10

# Tracker Server

tracker\_server=server1.com:22122  
# StorageServer 默认端口

storage\_server\_port=23000

# 如果文件ID的uri中包含/group\*\*，则要设置为true

url\_have\_group\_name = true

# Storage 配置的store\_path0路径，必须和storage.conf中的一致

store\_path0=/data/fastdfs/file

[IMG_256](https://www.cnblogs.com/chiangchou/p/javascript:void(0);)⑥ 复制 FastDFS 的部分配置文件到/etc/fdfs 目录

# cd /software/fastdfs-5.05/conf/

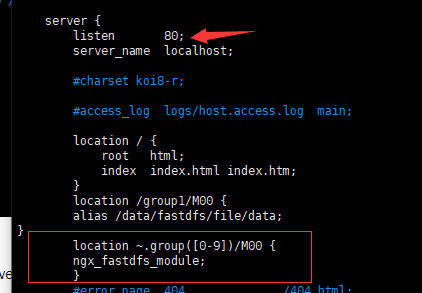
# cp anti-steal.jpg http.conf mime.types /etc/fdfs/

⑦ 配置nginx，修改nginx.conf

# vim /usr/local/nginx/conf/nginx.conf

修改配置，其它的默认

在80端口下添加fastdfs-nginx模块



注意：

　　listen 80 端口值是要与 /etc/fdfs/storage.conf 中的 http.server\_port=80 (前面改成80了)相对应。如果改成其它端口，则需要统一，同时在防火墙中打开该端口。

　　location 的配置，如果有多个group则配置location ~/group([0-9])/M00 ，没有则不用配group。

⑧ 在/data/fastdfs/file 文件存储目录下创建软连接，将其链接到实际存放数据的目录，这一步可以省略。

ln -s /data/fastdfs/file/data/ /data/fastdfs/file/data/M00

⑨ 启动nginx

# /usr/local/nginx/sbin/nginx

nginx

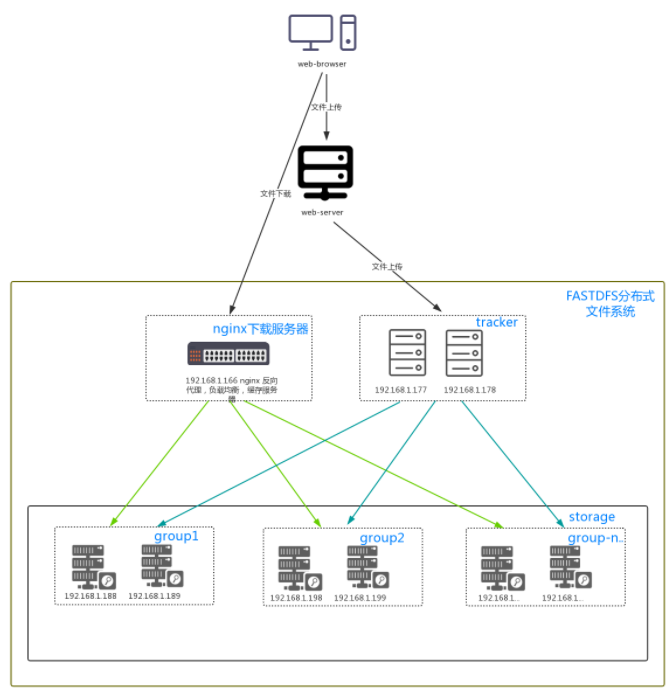
打印处如下就算配置成功



⑩ 在地址栏访问。

能下载文件就算安装成功。注意和第三点中直接使用nginx路由访问不同的是，这里配置 fastdfs-nginx-module 模块，可以重定向文件链接到源服务器取文件。

[http://192.168.224.12/group1/M00/00/00/wKjgDF14eNKAb0NiAADQmXv4gHY384.png](http://192.168.224.11/group1/M00/00/00/wKjgC11piVmARkD7AAEhr9cPOwk512.png)

最终部署结构图(盗的图)：可以按照下面的结构搭建环境。

## **五、Java客户端**

前面文件系统平台搭建好了，现在就要写客户端代码在系统中实现上传下载，这里只是简单的测试代码。

### **1、首先需要搭建 FastDFS 客户端Java开发环境**

① 项目中使用maven进行依赖管理，可以在pom.xml中引入如下依赖即可：

### **六、权限控制**

前面使用nginx支持http方式访问文件，但所有人都能直接访问这个文件服务器了，所以做一下权限控制。

FastDFS的权限控制是在服务端开启token验证，客户端根据文件名、当前unix时间戳、秘钥获取token，在地址中带上token参数即可通过http方式访问文件。

修改http.conf

# vim /etc/fdfs/http.conf

设置为true表示开启token验证http.anti\_steal.check\_token=true  
  
设置token失效的时间单位为秒(s)  
http.anti\_steal.token\_ttl=1800

密钥，跟客户端配置文件的fastdfs.http\_secret\_key保持一致

http.anti\_steal.secret\_key=FASTDFS1234567890

如果token检查失败，返回的页面

http.anti\_steal.token\_check\_fail=/ljzsg/fastdfs/page/403.html

记得重启服务。

② 配置客户端

客户端只需要设置如下两个参数即可，两边的密钥保持一致。

token 防盗链功能

fastdfs.http\_anti\_steal\_token=true

# 密钥

fastdfs.http\_secret\_key=FASTDFS1234567890

③ 客户端生成token

访问文件需要带上生成的token以及unix时间戳，所以返回的token是token和时间戳的拼接。

之后，将token拼接在地址后即可访问：<http://192.168.224.11/group1/M00/00/00/wKjgC11piVmARkD7AAEhr9cPOwk512.png>

?token=078d370098b03e9020b82c829c205e1f&ts=1508141521

/\*\*

\* 获取访问服务器的token，拼接到地址后面

\*

\* @param filepath 文件路径 group1/M00/00/00/wKgzgFnkTPyAIAUGAAEoRmXZPp876.jpeg

\* @param httpSecretKey 密钥

\* @return 返回token，如： token=078d370098b03e9020b82c829c205e1f&ts=1508141521

\*/

public static String getToken(String filepath, String httpSecretKey){

// unix seconds

int ts = (int) Instant.now().getEpochSecond();

// token

String token = "null";

try {

token = ProtoCommon.getToken(getFilename(filepath), ts, httpSecretKey);

} catch (UnsupportedEncodingException e) {

e.printStackTrace();

} catch (NoSuchAlgorithmException e) {

e.printStackTrace();

} catch (MyException e) {

e.printStackTrace();

}

StringBuilder sb = new StringBuilder();

sb.append("token=").append(token);

sb.append("&ts=").append(ts);

return sb.toString();

}

④ 注意事项

如果生成的token验证无法通过，请进行如下两项检查：  
　　A. 确认调用token生成函数(ProtoCommon.getToken)，传递的文件ID中没有包含group name。传递的文件ID格式形如：M00/00/00/wKgzgFnkTPyAIAUGAAEoRmXZPp876.jpeg

　　B. 确认服务器时间基本是一致的，注意服务器时间不能相差太多，不要相差到分钟级别。

⑤ 对比下发现，如果系统文件隐私性较高，可以直接通过fastdfs-client提供的API去访问即可，不用再配置Nginx走http访问。配置Nginx的主要目的是为了快速访问服务器的文件(如图片)，如果还要加权限验证，则需要客户端生成token，其实已经没有多大意义。

关键是，这里我没找到FastDFS如何对部分资源加token验证，部分开放。

OK，以上就是单机中使用FastDFS搭建文件系统并上传下载的过程。

完！！！