集群搭建

环境说明

服务器：（centos7）

192.168.3.5 tracker-group1

192.168.3.13 tracker-group2

192.168.3.47 storage-group1-1

192.168.3.49 storage-group1-2

192.168.3.50 storage-group2-1

192.168.3.51 storage-group2-2

安装包：

- `libfastcommon-1.0.39.tar.gz`： 公共库安装包，github下载地址：https://github.com/happyfish100/libfastcommon/releases
- `fastdfs-5.11.tar.gz`：fastdfs安装包，github下载地址：https://github.com/happyfish100/fastdfs/releases
- `fastdfs-nginx-module-1.20.tar.gz`：fastdfs的nginx模块，github下载地址：https://github.com/happyfish100/fastdfs-nginx-module/releases
- `nginx-1.16.0.tar.gz`：nginx 安装包，下载地址：http://nginx.org/en/download.html
- `ngx_cache_purge-2.3.tar.gz`: nginx purge模块，添加到踪器中的nginx配置， 下载地址：https://github.com/FRiCKLE/ngx_cache_purge/releases

1、安装前的环境准备（六台机器）

注意：所有节点需要安装

（1）安装gcc、pcre、zlib等

FastDFS是C写的，安装环境必须支持C编译。

```
yum -y install zlib zlib-devel pcre pcre-devel gcc gcc-c++ openssl openssl-devel libevent libevent-devel perl unzip net-tools wget
```

（2）新建目录，并上传安装包到该目录

```
mkdir -p /usr/local/fast
```

新建`fast`目录后，xshell上传安装包到目录下。（安装完成后可以删除安装包，避免占用空间）

2、安装libfastcommon（六台机器）

（1）进入 `fast` 目录

```
cd /usr/local/fast
```

（2）解压 libfastcommon-1.0.39.tar.gz

```
tar -zxvf libfastcommon-1.0.39.tar.gz
```

（3）进入解压的目录

```
cd libfastcommon-1.0.39
```

（4）编译和安装

编译：

```
./make.sh
```

如果编译没问题，则安装。 有问题可能是C编译环境没准备好（现在没发现，如有度量）

```
./make.sh install
```

可以看出，libfastcommon默认安装到 `/usr/lib64/` 下

（5）创建软连接。

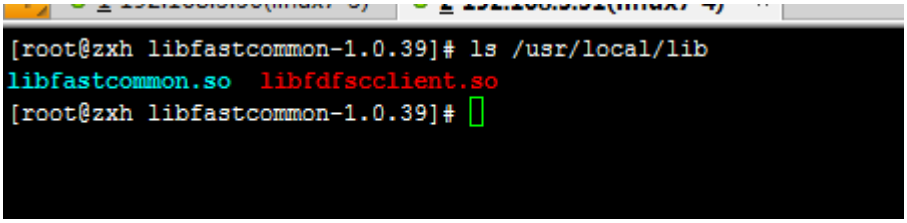
将FastDFS主程序设置的目录为 `/usr/local/lib/` ，所有需要创建 `/usr/lib64/` 下的一些核心执行程序的软连接文件。

如果没有则创建 `/usr/local/lib/` 目录

```
mkdir /usr/local/lib/
```

创建软连接

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



3、安装FastDFS（六台机器）

（1）解压fastdfs-5.11.tar.gz

```
cd /usr/local/fast
tar -zxvf /usr/local/fast/fastdfs-5.11.tar.gz
```

（2）进入解压目录

```
cd /usr/local/fast/fastdfs-5.11
```

（3）编译安装

编译：

```
./make.sh
```

安装

```
./make.sh install
```

```
[root@zxh fastdfs-5.11]# ./make.sh install
mkdir -p /usr/bin
mkdir -p /etc/fdfs
cp -f fdfs_trackerd /usr/bin
if [ ! -f /etc/fdfs/tracker.conf.sample ]; then cp -f ../conf/tracker.conf /etc/fdfs/tracker.conf.sample; fi
if [ ! -f /etc/fdfs/storage_ids.conf.sample ]; then cp -f ../conf/storage_ids.conf /etc/fdfs/storage_ids.conf.sample; fi
mkdir -p /usr/bin
mkdir -p /etc/fdfs
cp -f fdfs_storaged  /usr/bin
if [ ! -f /etc/fdfs/storage.conf.sample ]; then cp -f ../conf/storage.conf /etc/fdfs/storage.conf.sample; fi
mkdir -p /usr/bin
mkdir -p /etc/fdfs
mkdir -p /usr/lib64
mkdir -p /usr/lib
cp -f fdfs_monitor fdfs_test fdfs_test1 fdfs_crc32 fdfs_upload_file fdfs_download_file fdfs_delete_file fdfs_file_info fdfs_appender_test fdfs_appender
_test1 fdfs_append_file fdfs_upload_appender /usr/bin
if [ 0 -eq 1 ]; then cp -f libfdfsclient.a /usr/lib64; cp -f libfdfsclient.a /usr/lib/;fi
if [ 1 -eq 1 ]; then cp -f libfdfsclient.so /usr/lib64; cp -f libfdfsclient.so /usr/lib/;fi
mkdir -p /usr/include/fastdfs
cp -f ../common/fdfs_define.h ../common/fdfs_global.h ../common/mime_file_parser.h ../common/fdfs_http_shared.h ../tracker/tracker_types.h ../tracker/t
racker_proto.h ../tracker/fdfs_shared_func.h ../storage/trunk_mgr/trunk_shared.h tracker_client.h storage_client.h storage_client1.h client_func.h clie
nt_global.h fdfs_client.h /usr/include/fastdfs
if [ ! -f /etc/fdfs/client.conf.sample ]; then cp -f ../conf/client.conf /etc/fdfs/client.conf.sample; fi
[root@zxh fastdfs-5.11]#
```

安装完成后，查看配置文件和脚本。

1、服务脚本（在 `/etc/init.d` 下）：

```
[root@zxh fastdfs-5.11]# ls  /etc/init.d
fdfs_storaged  fdfs_trackerd  functions  netconsole  network  README
[root@zxh fastdfs-5.11]#
```

- fdfs_storaged：Storage server脚本
- fdfs_trackerd：Trackerd server脚本

2、配置文件（在 `/etc/fdfs/` 下）

```
[root@zxh fastdfs-5.11]# ls /etc/fdfs
client.conf.sample  storage.conf.sample  storage_ids.conf.sample  tracker.conf.sample
[root@zxh fastdfs-5.11]#
```

- client.conf.sample：客户端配置文件样板
- storage.conf.sample：storage服务配置样板
- tracker.conf.sample：tracker服务配置样板
- storage_ids.conf.sample：storage id配置

3、命令行执行脚本（在 `/usr/bin/` 下）

```
client.conf.sample  storage.conf.sample  storage_ids.conf.sample  tracker.conf.sample
[root@zxh fastdfs-5.11]# cd /usr/bin/&& ls | grep fdfs
fdfs_appender_test
fdfs_appender_test1
fdfs_append_file
fdfs_crc32
fdfs_delete_file
fdfs_download_file
fdfs_file_info
fdfs_monitor
fdfs_storaged
fdfs_test
fdfs_test1
fdfs_trackerd
fdfs_upload_appender
fdfs_upload_file
[root@zxh bin]#
```

（4）修改FastDFS服务脚本配置（旧版本需要，当前安装版本不需要）

如果正确则无需修改。

因为FastDFS服务脚本设置的bin目录为 `/usr/local/bin` 下，但是实际上我们按照在 `/usr/bin/` 下。所以我们需要修改FastDFS配置文件中的路径。

- 修改文件 `/etc/init.d/fdfs_storaged` :将 `/usr/local/bin` 替换 `/usr/bin/`
- 修改文件 `/etc/init.d/fdfs_trackerd` :将 `/usr/local/bin` 替换 `/usr/bin/`

4、配置跟踪器（两台机器）

192.168.3.5 tracker-group1

192.168.3.13 tracker-group2

（1）进入/etc/fdfs目录，并复制一份tracker.conf配置文件

```
cd /etc/fdfs
cp tracker.conf.sample tracker.conf
```

```
[root@zxh fastdfs-5.11]# cd /etc/fdfs
[root@zxh fdfs]# ls
client.conf.sample  storage.conf.sample  storage_ids.conf.sample  tracker.conf.sample
[root@zxh fdfs]# cp tracker.conf.sample tracker.conf
[root@zxh fdfs]#
```

（2）修改tracker.conf文件

修改base_path为自己定义的路径：

```
base_path=/fastdfs/tracker
store_lookup=0
```

```
# the base path to store data and log files
base_path=/fastdfs/tracker
```

存储路径

```
# max concurrent connections this server supported
max_connections=256

# accept thread count
# default value is 1
# since V4.07
accept_threads=1

# work thread count, should <= max_connections
# default value is 4
# since V2.00
work_threads=4

# min buff size
# default value 8KB
min_buff_size = 8KB

# max buff size
# default value 128KB
max_buff_size = 128KB

# the method of selecting group to upload files
# 0: round robin
# 1: specify group
# 2: load balance, select the max free space group to upload file
store_lookup=0
```

上传图片的方式：0是轮询 1是特定group上传 2是均衡

更多配置可查看：http://bbs.chinaunix.net/thread-1941456-1-1.html

（3）创建指定的tracker保存路径

上一步我们指定tracker保存的目录为/fastdfs/tracker。所有要创建该目录

```
mkdir -p /fastdfs/tracker
```

（4）防火墙开放端口

tracker.conf配置文件默认指定端口为22122，所有要开放此端口

开放端口：

```
firewall-cmd --zone=public --add-port=22122/tcp --permanent
```

重启防火墙：

```
systemctl restart firewalld.service
```

（5）启动跟踪器

启动命令：

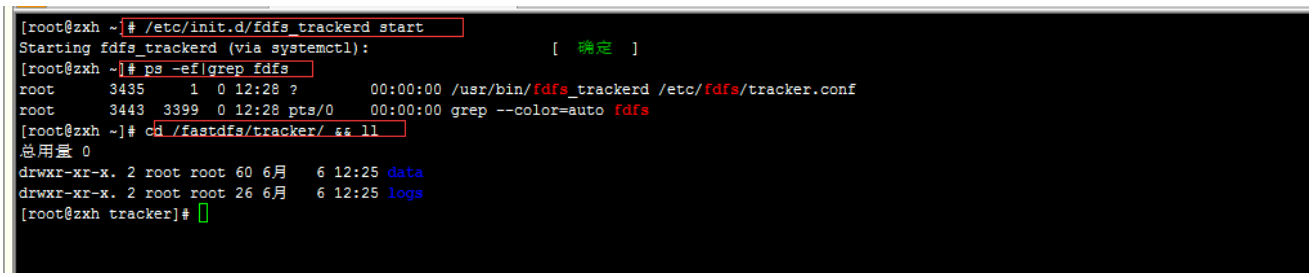
```
/etc/init.d/fdfs_trackerd start
```

查看进程：

```
ps -ef|grep fdfs
```

查看trackerd的数据目录

```
cd /fastdfs/tracker/ && ll
```



如果要关闭跟踪器,命令如下：

```
/etc/init.d/fdfs_trackerd stop
```

注意：千万不要用kill命令关闭，因为这样可能会丢失数据。

（6）设置开机启动

生成环境一般需求设置开机启动一些服务，如keepalived、tomcat等

修改 `/etc/rc.d/rc.local` ，加入配置： `/etc/init.d/fdfs_trackerd start`

```
#!/bin/bash
# THIS FILE IS ADDED FOR COMPATIBILITY PURPOSES
#
# It is highly advisable to create own systemd services or udev rules
# to run scripts during boot instead of using this file.
#
# In contrast to previous versions due to parallel execution during boot
# this script will NOT be run after all other services.
#
# Please note that you must run 'chmod +x /etc/rc.d/rc.local' to ensure
# that this script will be executed during boot.

touch /var/lock/subsys/local
/etc/init.d/fdfs_trackerd start
```

5、配置FastDFS服务（四台服务器）

192.168.3.47 storage-group1-1

192.168.3.49 storage-group1-2

192.168.3.50 storage-group2-1

192.168.3.51 storage-group2-2

（1）进入/etc/fdfs目录，并复制一份storage.conf

```
cd /etc/fdfs/
```

```
cp storage.conf.sample  storage.conf
```

（2）修改storage.conf文件

修改内容：

3.47和3.49一组（group1）、3.50和3.51一组（group2）

```
disabled=false
group_name=group1 # 组名
port=23000  #storage端口，同组的端口号必须相同
base_path=/fastdfs/storage # 数据存储路径
store_path_count=1  # 存储路径个数，需要和store_path个数匹配
store_path0=/fastdfs/storage  # 数据存储路径
tracker_server=192.168.3.5:22122 #tracker服务器ip和端口，多个时直接添加多行记录
tracker_server=192.168.3.13:22122
http.server_port=8888 #http端口号
```

（3）创建存储目录

```
mkdir -p /fastdfs/storage
```

（4）防火墙开放端口

Storage server配置文件/etc/fdfs/storage.conf默认端口是23000。所有防火墙需要开发端口。

开放端口：

```
firewall-cmd --zone=public --add-port=23000/tcp --permanent
```

重启防火墙：

```
systemctl restart firewalld.service
```

（5）启动storage

注意：启动Storage server之前必须先启动Traker server

启动命令：

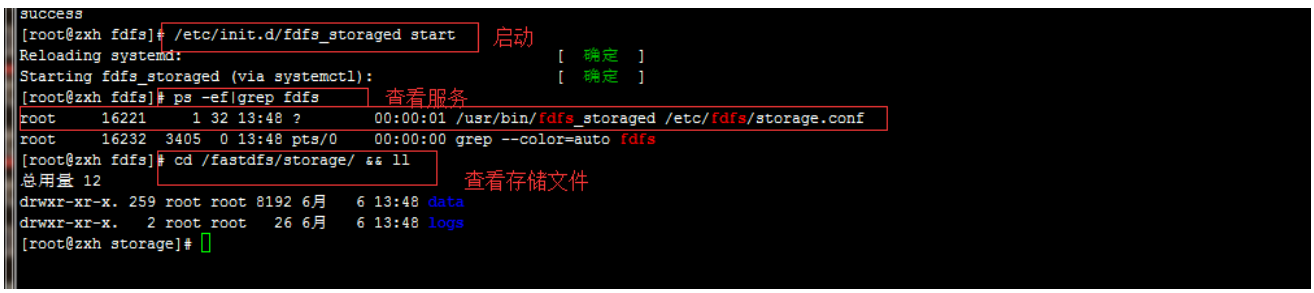
```
/etc/init.d/fdfs_storaged start
```

查看进程：

```
ps -ef|grep fdfs
```

查看trackerd的数据目录

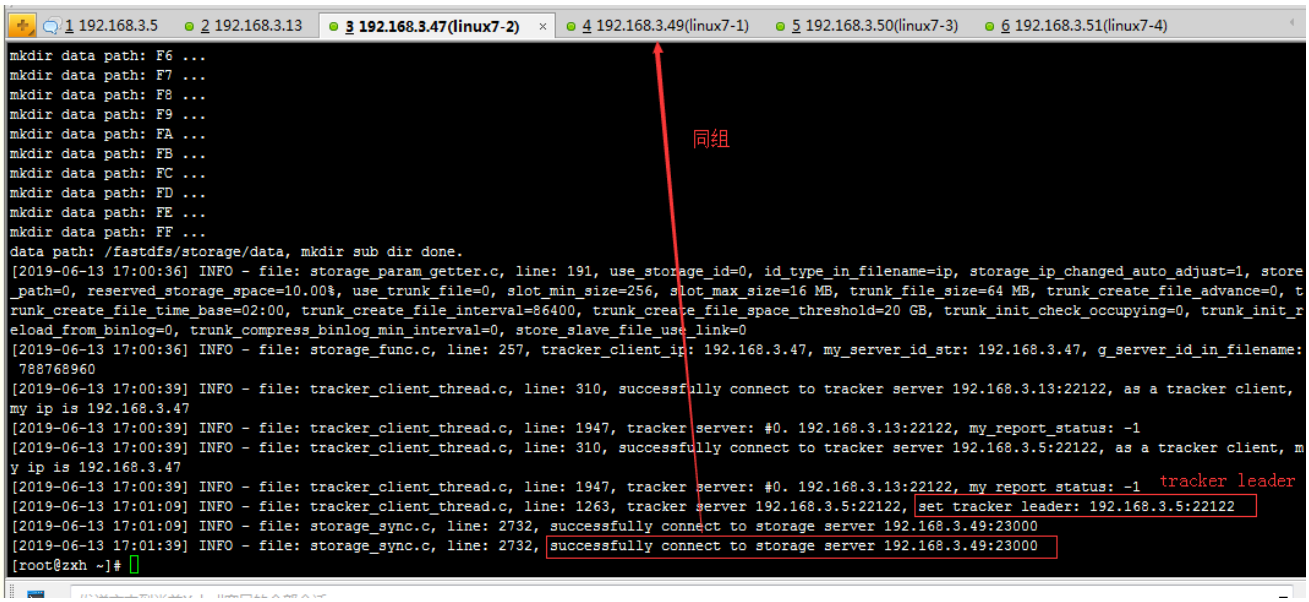
```
cd /fastdfs/storage/ && ll
```



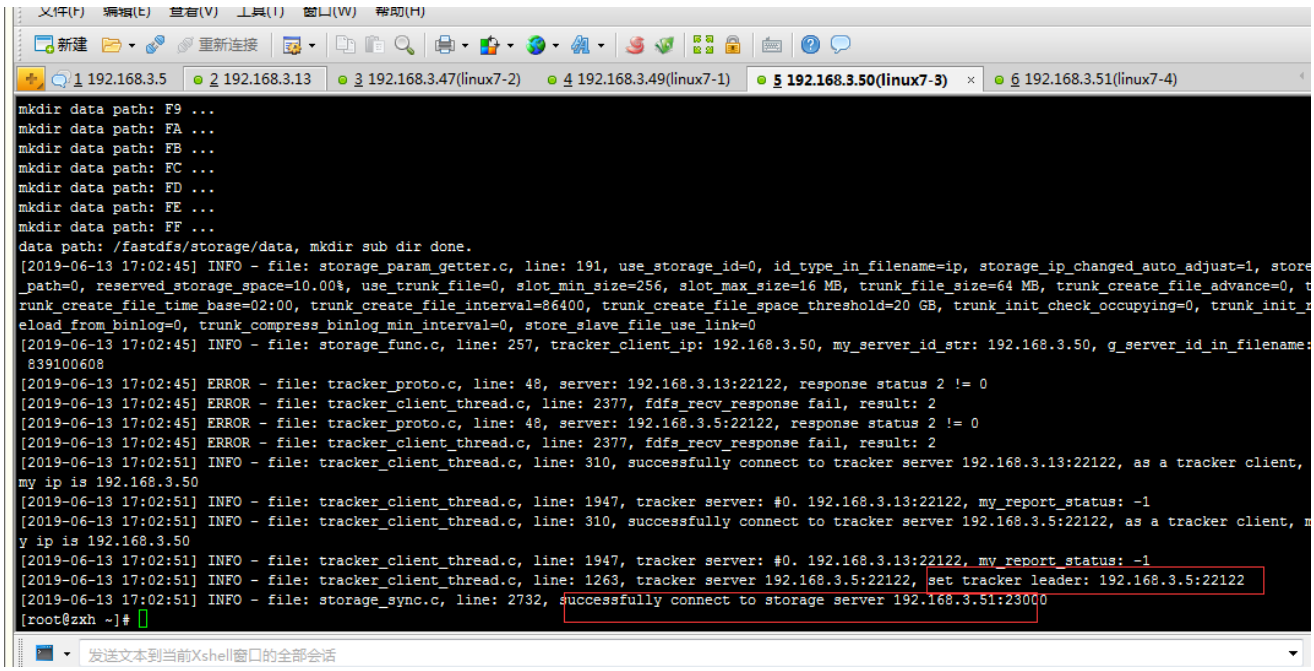
查看日志

```
tail -n 100 /fastdfs/storage/logs/storaged.log
```

查看 192.168.3.47的日志，如下：



查看 192.168.3.50的日志，如下：



当集群环境能互相知道对方存在的时候，启动成功。

如果要关闭跟踪器,命令如下：

```
/etc/init.d/fdfs_storaged stop
```

注意：千万不要用kill命令关闭，因为这样可能会丢失数据。

（6）设置开机启动

生成环境一般需求设置开机启动一些服务，如keepalived、tomcat等

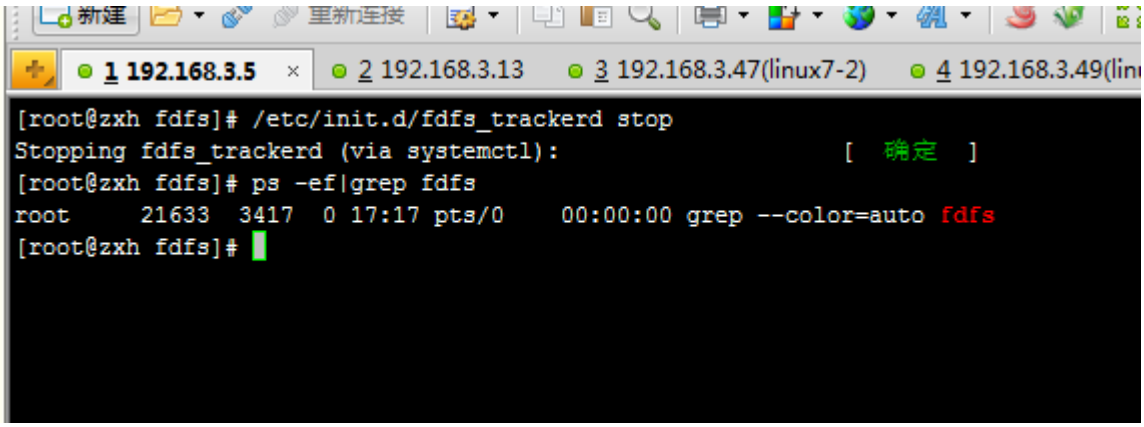
修改 `/etc/rc.d/rc.local` ，加入配置： `/etc/init.d/fdfs_storaged start`

（7）测试tracker leader

上图日志可以看到：tracker leader是192.168.3.5。现在关闭192.168.3.5服务，看看是否切换leader。

在192.168.3.5上执行停止服务：

```
/etc/init.d/fdfs_trackerd stop
```



继续查看storage节点的日志，可以看出tracker leader已经切换

```
mkdir data path: FC ...
mkdir data path: FD ...
mkdir data path: FE ...
mkdir data path: FF ...
data path: /fastdfs/storage/data, mkdir sub dir done.
[2019-06-13 17:00:36] INFO - file: storage_param_getter.c, line: 191, use_storage_id=0, id_type_in_filename=ip, storage_ip_changed_auto_adjust=1, store
_path=0, reserved_storage_space=10.00%, use_trunk_file=0, slot_min_size=256, slot_max_size=16 MB, trunk_file_size=64 MB, trunk_create_file_advance=0, t
runk_create_file_time_base=02:00, trunk_create_file_interval=86400, trunk_create_file_space_threshold=20 GB, trunk_init_check_occupying=0, trunk_init_r
eload_from_binlog=0, trunk_compress_binlog_min_interval=0, store_slave_file_use_link=0
[2019-06-13 17:00:36] INFO - file: storage_func.c, line: 257, tracker_client_ip: 192.168.3.47, my_server_id_str: 192.168.3.47, g_server_id_in_filename:
 788768960
[2019-06-13 17:00:39] INFO - file: tracker_client_thread.c, line: 310, successfully connect to tracker server 192.168.3.13:22122, as a tracker client,
my ip is 192.168.3.47
[2019-06-13 17:00:39] INFO - file: tracker_client_thread.c, line: 1947, tracker server: #0. 192.168.3.13:22122, my_report_status: -1
[2019-06-13 17:00:39] INFO - file: tracker_client_thread.c, line: 310, successfully connect to tracker server 192.168.3.5:22122, as a tracker client, m
y ip is 192.168.3.47
[2019-06-13 17:00:39] INFO - file: tracker_client_thread.c, line: 1947, tracker server: #0. 192.168.3.13:22122, my_report_status: -1
[2019-06-13 17:01:09] INFO - file: tracker_client_thread.c, line: 1263, tracker server 192.168.3.5:22122, set tracker leader: 192.168.3.5:22122
[2019-06-13 17:01:09] INFO - file: storage_sync.c, line: 2732, successfully connect to storage server 192.168.3.49:23000
[2019-06-13 17:01:39] INFO - file: storage_sync.c, line: 2732, successfully connect to storage server 192.168.3.49:23000   3.13成为leader
[2019-06-13 17:18:09] ERROR - file: tracker_client_thread.c, line: 1148, tracker server 192.168.3.5:22122, recv data fail, errno: 107, error info: Tran
sport endpoint is not connected.
[2019-06-13 17:18:09] INFO - file: tracker_client_thread.c, line: 1263, tracker server 192.168.3.13:22122, set tracker leader: 192.168.3.13:22122
[2019-06-13 17:18:09] ERROR - file: connection_pool.c, line: 130, connect to 192.168.3.5:22122 fail, errno: 111, error info: Connection refused
[2019-06-13 17:18:10] ERROR - file: tracker_client_thread.c, line: 277, connect to tracker server 192.168.3.5:22122 fail, errno: 111, error info: Conne
ction refused
[root@zxh ~]#
                                                                                                   3.5连接不上
```

当启动192.168.3.5服务后，很快stroage节点连上3.5，此时3.5作为tracker client。

```
mkdir data path: FE ...
mkdir data path: FF ...
data path: /fastdfs/storage/data, mkdir sub dir done.
[2019-06-13 17:00:36] INFO - file: storage_param_getter.c, line: 191, use_storage_id=0, id_type_in_filename=ip, storage_ip_changed_auto_adjust=1, store
_path=0, reserved_storage_space=10.00%, use_trunk_file=0, slot_min_size=256, slot_max_size=16 MB, trunk_file_size=64 MB, trunk_create_file_advance=0, t
runk_create_file_time_base=02:00, trunk_create_file_interval=86400, trunk_create_file_space_threshold=20 GB, trunk_init_check_occupying=0, trunk_init_r
eload_from_binlog=0, trunk_compress_binlog_min_interval=0, store_slave_file_use_link=0
[2019-06-13 17:00:36] INFO - file: storage_func.c, line: 257, tracker_client_ip: 192.168.3.47, my_server_id_str: 192.168.3.47, g_server_id_in_filename:
 788768960
[2019-06-13 17:00:39] INFO - file: tracker_client_thread.c, line: 310, successfully connect to tracker server 192.168.3.13:22122, as a tracker client,
my ip is 192.168.3.47
[2019-06-13 17:00:39] INFO - file: tracker_client_thread.c, line: 1947, tracker server: #0. 192.168.3.13:22122, my_report_status: -1
[2019-06-13 17:00:39] INFO - file: tracker_client_thread.c, line: 310, successfully connect to tracker server 192.168.3.5:22122, as a tracker client, m
y ip is 192.168.3.47
[2019-06-13 17:00:39] INFO - file: tracker_client_thread.c, line: 1947, tracker server: #0. 192.168.3.13:22122, my_report_status: -1
[2019-06-13 17:01:09] INFO - file: tracker_client_thread.c, line: 1263, tracker server 192.168.3.5:22122, set tracker leader: 192.168.3.5:22122
[2019-06-13 17:01:09] INFO - file: storage_sync.c, line: 2732, successfully connect to storage server 192.168.3.49:23000
[2019-06-13 17:01:39] INFO - file: storage_sync.c, line: 2732, successfully connect to storage server 192.168.3.49:23000
[2019-06-13 17:18:09] ERROR - file: tracker_client_thread.c, line: 1148, tracker server 192.168.3.5:22122, recv data fail, errno: 107, error info: Tran
sport endpoint is not connected.
[2019-06-13 17:18:09] INFO - file: tracker_client_thread.c, line: 1263, tracker server 192.168.3.13:22122, set tracker leader: 192.168.3.13:22122
[2019-06-13 17:18:09] ERROR - file: connection_pool.c, line: 130, connect to 192.168.3.5:22122 fail, errno: 111, error info: Connection refused
[2019-06-13 17:18:10] ERROR - file: tracker_client_thread.c, line: 277, connect to tracker server 192.168.3.5:22122 fail, errno: 111, error info: Conne
ction refused
[2019-06-13 17:21:10] INFO - file: tracker_client_thread.c, line: 310, successfully connect to tracker server 192.168.3.5:22122, continuous fail count:
 6, as a tracker client, my ip is 192.168.3.47
[root@zxh ~]#
```

（8）查看存储集群信息

当所有tracker和storage节点都启动成功后，可以在任意一个存储节点（storage）上查看存储集群信息。

查看命令：

```
/usr/bin/fdfs_monitor /etc/fdfs/storage.conf
```

由于信息太多，只能截部分图片，如下：

```
[root@zxh ~]# /usr/bin/fdfs_monitor /etc/fdfs/storage.conf
[2019-06-13 17:24:12] DEBUG - base_path=/fastdfs/storage, connect_timeout=30, network_timeout=60, tracker_server_count=2, anti_steal_token=0, anti_stea
l_secret_key length=0, use_connection_pool=0, g_connection_pool_max_idle_time=3600s, use_storage_id=0, storage server id count: 0

server_count=2, server_index=0

tracker server is 192.168.3.13:22122

group count: 2

Group 1:
group name = group1
disk total space = 51175 MB
disk free space = 49820 MB
trunk free space = 0 MB
storage server count = 2
active server count = 2
storage server port = 23000
storage HTTP port = 8888
store path count = 1
subdir count per path = 256
current write server index = 0
current trunk file id = 0

        Storage 1:
                id = 192.168.3.47
                ip_addr = 192.168.3.47 (zxh)  ACTIVE
                http domain =
                version = 5.11
                join time = 2019-06-13 17:00:34
                up time = 2019-06-13 17:00:34
                total storage = 51175 MB
                free storage = 49845 MB
                upload priority = 10
                store_path_count = 1
                subdir_count_per_path = 256
                storage_port = 23000
                storage_http_port = 8888
```

两个group

group1

存储空间

存储节点两个

group1的存储节点1

（9）上传文件测试

使用命令上传文件。在任意一台跟踪节点（3.5或3.13）操作。

1、进入/etc/fdfs目录

```
cd /etc/fdfs/
```

2、copy一份client.conf配置文件

```
cp client.conf.sample client.conf
```

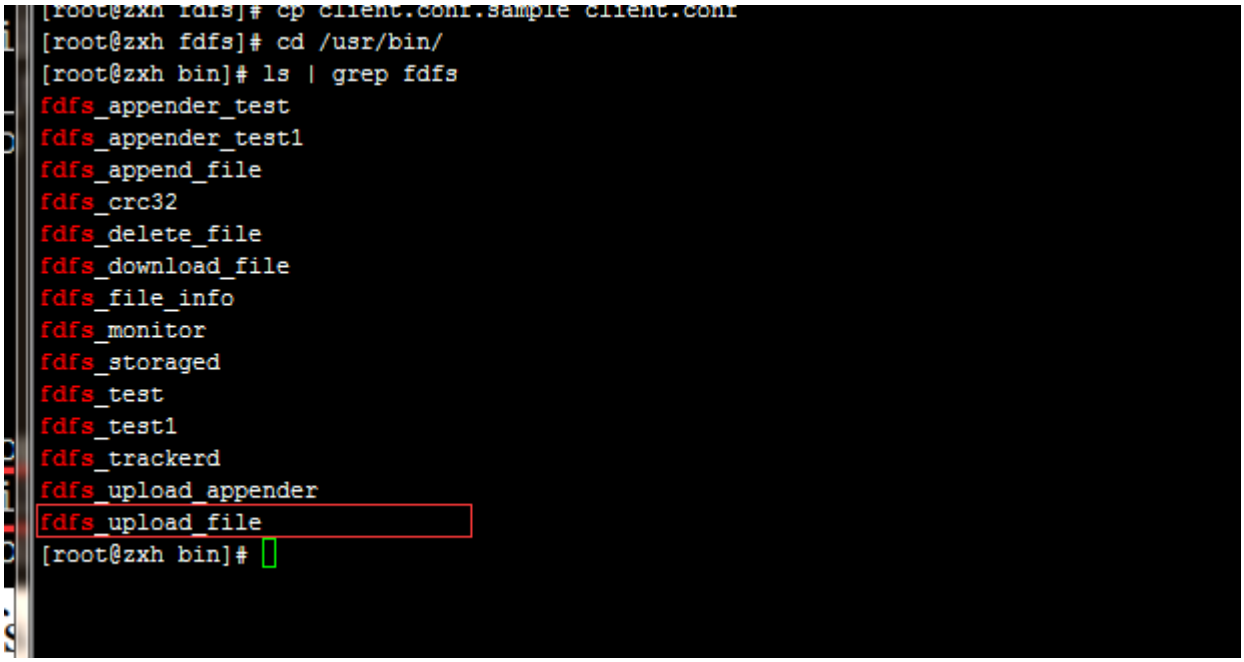
1.3、修改client.conf配置文件，修改内容为：

```
base_path=/fastdfs/tracker  #tracker存储路径
tracker_server=192.168.3.5:22122 #tracker 服务
tracker_server=192.168.3.13:22122 #多个tracker 服务，多行配置
```

1.4、找到命令脚本，并使用命令进行文件上传

进入脚本目录：

```
cd /usr/bin/
ls | grep fdfs #查看脚本
```

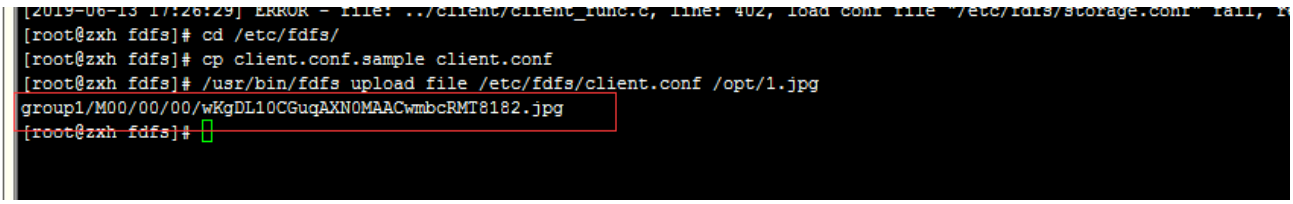


上传附件：

```
/usr/bin/fdfs_upload_file /etc/fdfs/client.conf /opt/1.jpg
```

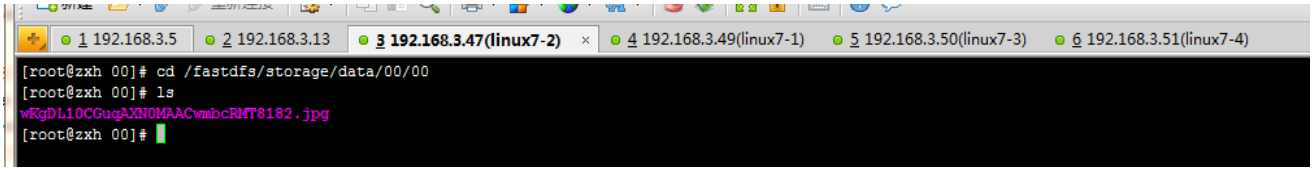
- /usr/bin/fdfs_upload_file：上传命令脚本
- /etc/fdfs/client.conf：客户端配置文件
- /opt/1.jpg：上传的文件

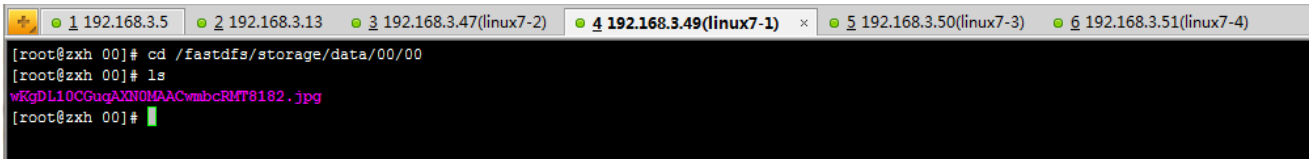
上传成功返回上传文件地址：

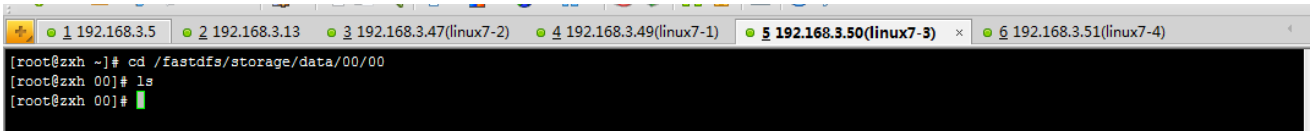


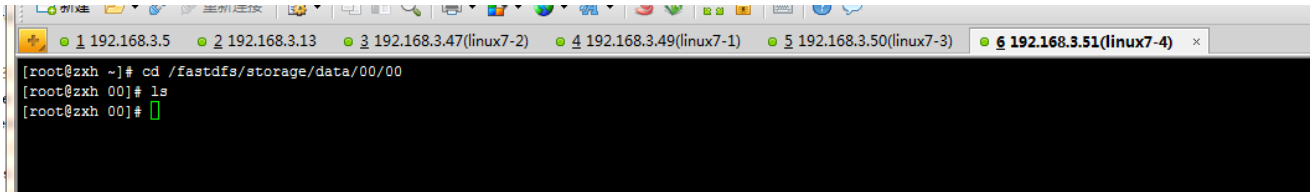
返回的图片地址是`group1/M00/00/00/wKgDL10CGuqAXN0MAACwmbcRMT8182.jpg`。可以看出图片在group1组的/M00/00/00路径下。 group1对应的是3.47和3.49节点。

看图看真相：









ps1：再执行一次上传，这次文件会上传到group2。因为我们在搭建环境的时候 修改tracker.conf文件的上传文件模式：`store_looup=0` 轮询

ps2: 3.74宕机后，如果还有文件上传，上传到3.49后。3.47再启动，3.49会同步数据到3.47。

6、配置nginx

4个存储节点配置nginx，然后2个跟踪器节点配置nginx

6.1、4个存储节点配置nginx

192.168.3.47 、192.168.3.49 、192.168.3.50 、192.168.3.51

（1）进入目录，解压fastdfs-nginx-module-1.20.tar.gz

进入安装包目录并解压，解压后进入fastdfs-nginx-module-1.20/src/

```
cd /usr/local/fast
tar -zxvf fastdfs-nginx-module-1.20.tar.gz

cd  fastdfs-nginx-module-1.20/src/
```

```
[root@zxh local]# cd fast/
[root@zxh fast]# ls
fastdfs-5.11  fastdfs-5.11.tar.gz  fastdfs-nginx-module-1.20.tar.gz  libfastcommon-1.0.39  libfastcommon-1.0.39.tar.gz  nginx-1.1
[root@zxh fast]# pwd
/usr/local/fast
[root@zxh fast]# tar -zxvf fastdfs-nginx-module-1.20.tar.gz
fastdfs-nginx-module-1.20/
fastdfs-nginx-module-1.20/HISTORY
fastdfs-nginx-module-1.20/INSTALL
fastdfs-nginx-module-1.20/src/
fastdfs-nginx-module-1.20/src/common.c
fastdfs-nginx-module-1.20/src/common.h
fastdfs-nginx-module-1.20/src/config
fastdfs-nginx-module-1.20/src/mod_fastdfs.conf
fastdfs-nginx-module-1.20/src/ngx_http_fastdfs_module.c
[root@zxh fast]# ls
fastdfs-5.11          fastdfs-nginx-module-1.20          libfastcommon-1.0.39          nginx-1.16.0.tar.gz
fastdfs-5.11.tar.gz   fastdfs-nginx-module-1.20.tar.gz   libfastcommon-1.0.39.tar.gz
[root@zxh fast]# cd  fastdfs-nginx-module-1.20/src/
[root@zxh src]# ls
common.c  common.h  config  mod_fastdfs.conf  ngx_http_fastdfs_module.c
[root@zxh src]#
```

（2）修改fastdfs-nginx-module-1.20/src/config文件

config文件修改：

`/usr/local/include` 修改为 `/usr/include/fastdfs /usr/include/fastcommon/`

```
ngx_addon_name=ngx_http_fastdfs_module

if test -n "${ngx_module_link}"; then
    ngx_module_type=HTTP
    ngx_module_name=$ngx_addon_name
    ngx_module_incs="/usr/include/fastdfs /usr/include/fastcommon/"
    ngx_module_libs="-lfastcommon -lfdfsclient"
    ngx_module_srcs="$ngx_addon_dir/ngx_http_fastdfs_module.c"
    ngx_module_deps=
    CFLAGS="$CFLAGS -D_FILE_OFFSET_BITS=64 -DFDFS_OUTPUT_CHUNK_SIZE='256*1024' -DFDFS_MOD_CONF_FILENAME='\"/etc/fdfs/mod_fastdfs.conf\"'"
    . auto/module
else
    HTTP_MODULES="$HTTP_MODULES ngx_http_fastdfs_module"
    NGX_ADDON_SRCS="$NGX_ADDON_SRCS $ngx_addon_dir/ngx_http_fastdfs_module.c"
    CORE_INCS="$CORE_INCS /usr/include/fastdfs /usr/include/fastcommon/"
    CORE_LIBS="$CORE_LIBS -lfastcommon -lfdfsclient"
    CFLAGS="$CFLAGS -D_FILE_OFFSET_BITS=64 -DFDFS_OUTPUT_CHUNK_SIZE='256*1024' -DFDFS_MOD_CONF_FILENAME='\"/etc/fdfs/mod_fastdfs.conf\"'"
fi
```

（3）FastDFS与ngingx进行集成

1、进入nginx的安装包存放目录并解压。

```
cd /usr/local/fast/
tar -zxvf nginx-1.16.0.tar.gz
```

2、进入解压目录，并配置加入模块

```
cd nginx-1.16.0
./configure --add-module=/usr/local/fast/fastdfs-nginx-module-1.20/src
```

注意：这里没有设置nginx的安装目录，默认会安装到/usr/local下

3、nginx编译和安装

```
make && make install
```

（4）复制mod_fastdfs.conf到/etc/fdfs/目录

```
cp /usr/local/fast/fastdfs-nginx-module-1.20/src/mod_fastdfs.conf /etc/fdfs/
```

（5）修改/etc/fdfs/mod_fastdfs.conf

修改内容：

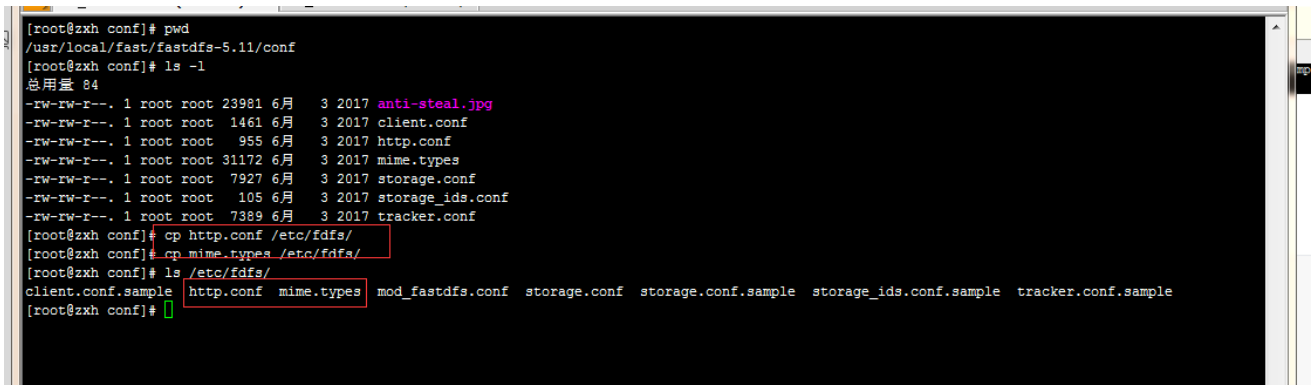
```
connect_timeout=10
tracker_server=192.168.3.5:22122
tracker_server=192.168.3.13:22122
storage_server_port=23000
url_have_group_name = true
store_path0=/fastdfs/storage
group_name=group1  #这里看节点修改：第一组为group1，第二组为group2
group_count=2

[group1]
group_name=group1
storage_server_port=23000
store_path_count=1
store_path0=/fastdfs/storage

[group2]
group_name=group2
storage_server_port=23000
store_path_count=1
store_path0=/fastdfs/storage
```

（6）复制文件http.conf、mime.types到/etc/fdfs/

```
cp /usr/local/fast/fastdfs-5.11/conf/http.conf /etc/fdfs/
cp /usr/local/fast/fastdfs-5.11/conf/mime.types /etc/fdfs/
```



（7）创建软连接

在/fastdfs/storage文件存储目录下创建软连接，将其连接到实际存放数据的目录

```
ln -s /fastdfs/storage/data/ /fastdfs/storage/data/M00
```

（8）修改ngin配置

nginx安装在/usr/local下载， 找到/usr/local/nginx/conf/nginx.conf文件修改

```
    #gzip  on;

    server {
        listen       8888;
        server_name  localhost;

        #charset koi8-r;

        #access_log  logs/host.access.log  main;

        location ~/group([0-9])/M00 {
            ngx_fastdfs_module;
        }

        #error_page  404              /404.html;

        # redirect server error pages to the static page /50x.html
        #
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }

        # proxy the PHP scripts to Apache listening on 127.0.0.1:80
```

修改端口

修改匹配规则

注意：nginx里的端口要和FastDFS存储中的storage.conf文件配置一致。http.server_port=8888

启动nginx

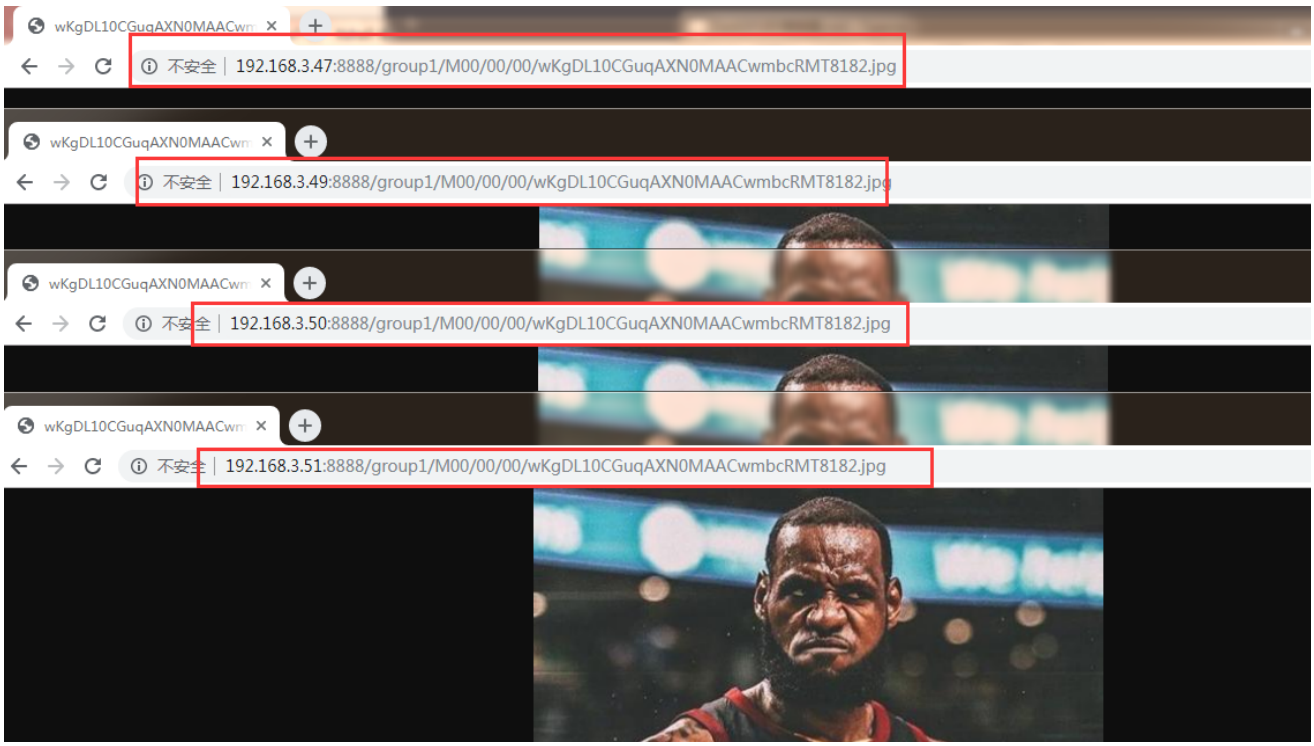
```
./nginx
```

如果http://192.168.3.51:8888 返回不到，则需要防火墙开放8888端口

```
firewall-cmd --zone=public --add-port=8888/tcp --permanent
systemctl restart firewalld.service
```

（9）测试

之前使用命令上传过图片，返回地址 `/group1/M00/00/00/wKgDL10CGuqAXN0MAACwmbcRMT8182.jpg` 。我们使用http访问文件服务器。可以访问即可



到此，集群FastDFS与Nginx整合完成。

6.2、2个跟踪器节点配置nginx

192.168.3.5 、192.168.3.13

需要在两个跟踪器上安装nginx，已提供方向代理服务，目的是使用统一的一个ip地址对外提供服务。

（1）进入目录，解压ngx_cache_purge-2.3.tar.gz

进入安装包目录并解压，解压后进入fastdfs-nginx-module-1.20/src/

```
cd /usr/local/fast
tar -zxvf ngx_cache_purge-2.3.tar.gz
```

（2）并添加ngx_cache_purge模块，安装nginx

1、进入nginx的安装包存放目录并解压。

```
cd /usr/local/fast/
tar -zxvf nginx-1.16.0.tar.gz
```

2、进入解压目录，并配置加入模块

```
cd nginx-1.16.0
./configure --add-module=/usr/local/fast/ngx_cache_purge-2.3
```

注意：这里没有设置nginx的安装目录，默认会安装到/usr/local下

3、nginx编译和安装

```
make && make install
```

（3）修改ngin配置，配置负载均衡和缓存（2个跟踪节点一致）

nginx安装在/usr/local下载， 找到/usr/local/nginx/conf/nginx.conf文件修改

nginx.conf

```
#user  nobody;
worker_processes  1;

#error_log  logs/error.log;
#error_log  logs/error.log  notice;
#error_log  logs/error.log  info;

#pid        logs/nginx.pid;


events {
    worker_connections  1024;
```

```
    use epoll;
}


http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    tcp_nopush     on;
    keepalive_timeout  65;

    #设置缓存
    server_names_hash_bucket_size 128;
    client_header_buffer_size 32k;
    large_client_header_buffers 4 32k;
    client_max_body_size 300m;

    proxy_redirect off;
    proxy_set_header Host $http_host;
    proxy_set_header Cookie $http_cookie;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;

    proxy_connect_timeout 90;
    proxy_read_timeout 90;
    proxy_send_timeout 90;
    proxy_buffer_size 16k;
    proxy_buffers 4 64k;
    proxy_busy_buffers_size 128k;
    proxy_temp_file_write_size 128k;

    #设置缓存缓存路径、存储方式、分配内存大小、磁盘最大空间、缓存期限
    proxy_cache_path /fastdfs/cache/nginx/proxy_cache levels=1:2
    keys_zone=http-cache:200m max_size=1g inactive=30d;
    proxy_temp_path /fastdfs/cache/nginx/proxy_cache/temp;

    #设置weigth权重 max_fails失败重试次数  fail_timeout连接失败超时时间
    #设置group1的服务器
    upstream fdfs_group1{
        server 192.168.3.47:8888 weight=1 max_fails=2 fail_timeout=30s;
        server 192.168.3.49:8888 weight=1 max_fails=2 fail_timeout=30s;
    }
    #设置group2的服务器
    upstream fdfs_group2{
        server 192.168.3.50:8888 weight=1 max_fails=2 fail_timeout=30s;
        server 192.168.3.51:8888 weight=1 max_fails=2 fail_timeout=30s;
    }


    server {
        listen       8000;
        server_name  localhost;
```

```
        #charset koi8-r;

        #access_log  logs/host.access.log  main;

        #设置group的负载均衡参数
        location /group1/M00 {
            proxy_next_upstream http_502 http_504 error timeout invalid_header;
            proxy_cache http-cache;
            proxy_cache_valid 200 304 12h;
            proxy_cache_key $uri$is_args$args;
            proxy_pass http://fdfs_group1;
            expires 30d;
        }

        location /group2/M00 {
            proxy_next_upstream http_502 http_504 error timeout invalid_header;
            proxy_cache http-cache;
            proxy_cache_valid 200 304 12h;
            proxy_cache_key $uri$is_args$args;
            proxy_pass http://fdfs_group2;
            expires 30d;
        }

        # 设置清除缓存的访问权限
        location ~/purge(/.*){
            allow 127.0.0.1;
            allow 192.168.1.0/24;
            deny all;
            proxy_cache_purge http-cache $1$is_args$args;
        }

        #error_page  404              /404.html;

        # redirect server error pages to the static page /50x.html
        #
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }


    }


}
```

（4）创建缓存目录

按以上nginx配置文件的要求，需要创建对应缓存目录

```
mkdir -p /fastdfs/cache/nginx/proxy_cache
mkdir -p /fastdfs/cache/nginx/proxy_cache/temp
```

（5）防火墙开放端口

```
firewall-cmd --zone=public --add-port=8000/tcp --permanent
systemctl restart firewalld.service
```

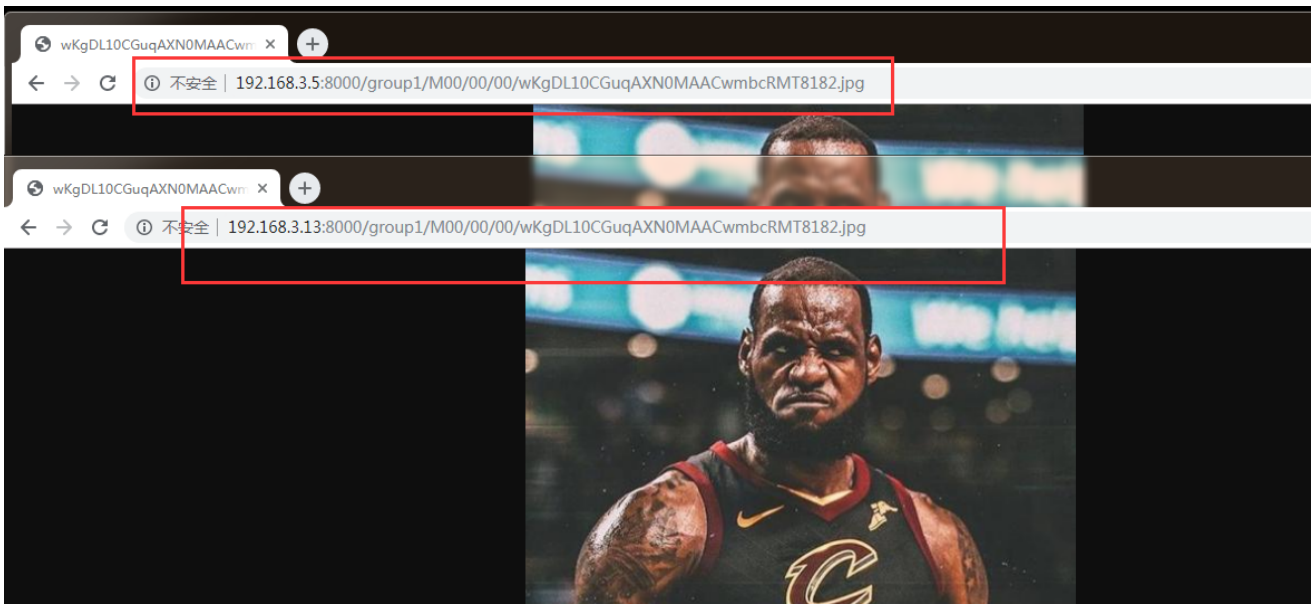
（6）启动nginx测试

```
cd /usr/local/nignx/sbin
./nginx
```

检查是否启动成功

```
ps -ef|grep nginx
```

启动成功后，可以http进行上传、访问、下载。



7、使用keepalived+nginx组成高可用负载均衡集群

上面完成了FastDFS集群环境的搭建，但是并没有做到nginx的高可用负载均衡集群。所以下面使用keepalived+nginx实现分布式高可用文件系统。