高可用架构篇

FastDFS集群的安装、配置、使用

## **1.FastDFS介绍**

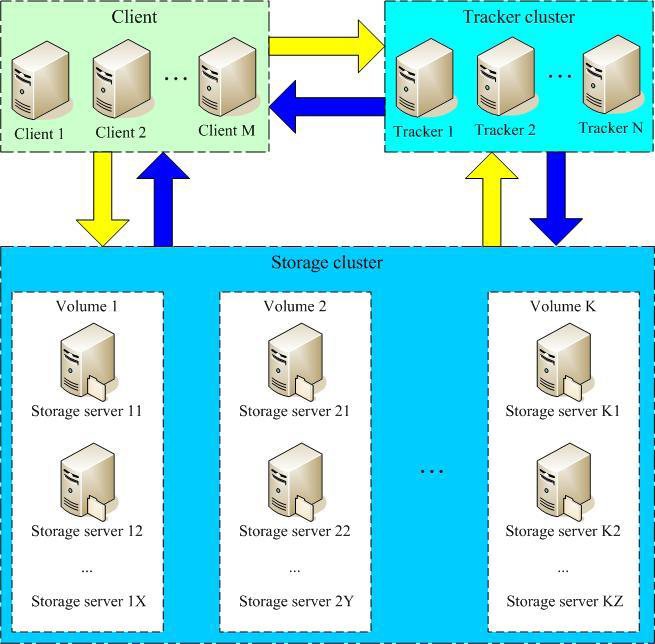
（参考：http://www.oschina.net/p/fastdfs）

FastDFS是一个开源的分布式文件系统，它对文件进行管理，功能包括：文件存储、文件同步、文件访问（文件上传、文件下载）等，解决了大容量存储和负载均衡的问题。特别适合以文件为载体的在线服务，如相册网站、视频网站等等。

FastDFS服务端有两个角色：跟踪器（tracker）和存储节点（storage）。

跟踪器主要做调度工作，在访问上起负载均衡的作用。存储节点存储文件，完成文件管理的所有功能：存储、同步和提供存取接口，FastDFS同时对文件的meta data进行管理。所谓文件的meta data就是文件的相关属性，以键值对（key value pair）方式表示，如：width=1024，其中的key为width，value为1024。文件meta data是文件属性列表，可以包含多个键值对。

FastDFS系统结构如下图所示：



跟踪器和存储节点都可以由一台多台服务器构成。跟踪器和存储节点中的服务器均可以随时增加或下线而不会影响线上服务。其中跟踪器中的所有服务器都是对等的，可以根据服务器的压力情况随时增加或减少。

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。FastDFS中的文件标识

### 1.FastDFS上传文件交互过程：



1. client询问tracker上传到的storage，不需要附加参数；

2. tracker返回一台可用的storage；

3. client直接和storage通讯完成文件上传。

客户端client 发起对FastDFS 的文件传输动作，是通过连接到某一台Tracker Server 的指定端口来实现的，Tracker Server 根据目前已掌握的信息，来决定选择哪一台Storage Server ，然后将这个Storage Server 的地址等信息返回给client，然后client 再通过这些信息连接到这台Storage Server，将要上传的文件传送到给Storage Server上。

### 2.FastDFS下载文件交互过程：



1. client询问tracker下载文件的storage，参数为文件标识（卷名和文件名）；

2. tracker返回一台可用的storage；

3. client直接和storage通讯完成文件下载。

### FastDFS 集群规划：

跟踪服务器1：192.168.1.131 du-dfs-tracker-1

跟踪服务器2：192.168.1.132 du-dfs-tracker-2

存储服务器1：192.168.1.135 du-dfs-storage-group1-1

存储服务器2：192.168.1.136 du-dfs-storage-group1-2

存储服务器3：192.168.1.137 du-dfs-storage-group2-1

存储服务器4：192.168.1.138 du-dfs-storage-group2-2

环境：CentOS6.6

用户：root

数据目录：/fastdfs（注：数据目录按你的数据盘挂载路径而定）

**安装包：**

FastDFS v5.05

libfastcommon-master.zip（是从FastDFS和FastDHT中提取出来的公共C函数库）

fastdfs-nginx-module\_v1.16.tar.gz

nginx-1.6.2.tar.gz

fastdfs\_client\_java.\_v1.25.tar.gz

源码地址：https://github.com/happyfish100/

下载地址：http://sourceforge.net/projects/fastdfs/files/

官方论坛：<http://bbs.chinaunix.net/forum-240-1.html>

FastDFS集群

VIP

**一级：**

Proxy 1 ( Keepalived + Nginx )

Proxy 2 ( Keepalived + Nginx )

**二级：**

Tracker 1 ( FastDFS + Nginx )

Tracker 2 ( FastDFS + Nginx )

**三级：**

Group 1

Storage 1 ( FastDFS + Nginx )

Storage 2 ( FastDFS + Nginx )

Group 2

Storage 3 ( FastDFS + Nginx )

Storage 4 ( FastDFS + Nginx )

## 一、FastDFS的安装

（所有跟踪服务器和存储服务器均执行如下操作）

### 1、编译和安装所需的依赖包:

yum install make cmake gcc gcc-c++

### 2、安装libfastcommon

**（https://github.com/happyfish100/libfastcommon）**

#### (1)上传或下载libfastcommon-master.zip到/usr/local/src目录，解压

cd /usr/local/src/

unzip libfastcommon-master.zip

cd libfastcommon-master



#### (2)编译、安装

./make.sh

./make.sh install

**libfastcommon默认安装到了**

/usr/lib64/libfastcommon.so

/usr/lib64/libfdfsclient.so

#### (3)因为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

### 3、安装FastDFS

**（https://github.com/happyfish100/fastdfs/releases）**

#### (1)上传或下载FastDFS源码包（FastDFS\_v5.05.tar.gz）到/usr/local/src目录，解压

cd /usr/local/src/

tar -zxvf FastDFS\_v5.05.tar.gz

cd FastDFS



#### (3)编译、安装（编译前要确保已经成功安装了libfastcommon）

./make.sh

./make.sh install

**采用默认安装的方式安装,安装后的相应文件与目录：**

**A、服务脚本在：**

/etc/init.d/fdfs\_storaged

/etc/init.d/fdfs\_tracker

**B、配置文件在（样例配置文件）:**

/etc/fdfs/client.conf.sample

/etc/fdfs/storage.conf.sample

/etc/fdfs/tracker.conf.sample

**C、命令工具在/usr/bin/目录下的：**

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

stop.sh

restart.sh

#### (4) 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

**因为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，**

**可以进入/user/bin目录使用以下命令查看fdfs的相关命令：**

# cd /usr/bin/

# ls | grep fdfs



**因此需要修改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：**

**vi /etc/init.d/fdfs\_trackerd**

使用查找替换命令进统一修改: %s+/usr/local/bin+/usr/bin

**vi /etc/init.d/fdfs\_storaged**

使用查找替换命令进统一修改: %s+/usr/local/bin+/usr/bin

**注意：**

**以上操作无论是配置tracker还是配置storage都是必须的，而tracker和storage的区别主要是在安装完fastdfs之后的配置过程中。**

## 二、配置FastDFS跟踪器Tracker(192.168.1.131、192.168.1.132)

### 1、 复制FastDFS跟踪器样例配置文件,并重命名:

# cd /etc/fdfs/



# cp tracker.conf.sample tracker.conf

### 2、 编辑跟踪器配置文件：

# vi /etc/fdfs/tracker.conf

**修改的内容如下：**

#启用配置文件

disabled=false

#tracker的端口号，一般采用22122这个默认端口

port=22122

#tracker的数据文件和日志目录

base\_path=/home/fastdfs/tracker

（其它参数保留默认配置，具体配置解释请参考官方文档说明：http://bbs.chinaunix.net/thread-1941456-1-1.html）

### 3、 创建基础数据目录（参考基础目录base\_path配置）:

# mkdir -p /home/fastdfs/tracker

### 4、 防火墙中打开跟踪器端口（默认为22122）:

# vi /etc/sysconfig/iptables

添加如下端口行：

## FastDFS Tracker Port

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

重启防火墙：

# service iptables restart

### 5、 启动Tracker：

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

或者service fdfs\_trackerd start

（初次成功启动，会在/fastdfs/tracker目录下创建data、logs两个目录）可以通过以下两个方法查看tracker是否启动成功:

(1)查看22122端口监听情况:netstat -unltp|grep fdfs

(2)通过以下命令查看tracker的启动日志，看是否有错误

tail -100f /fastdfs/tracker/logs/trackerd.log



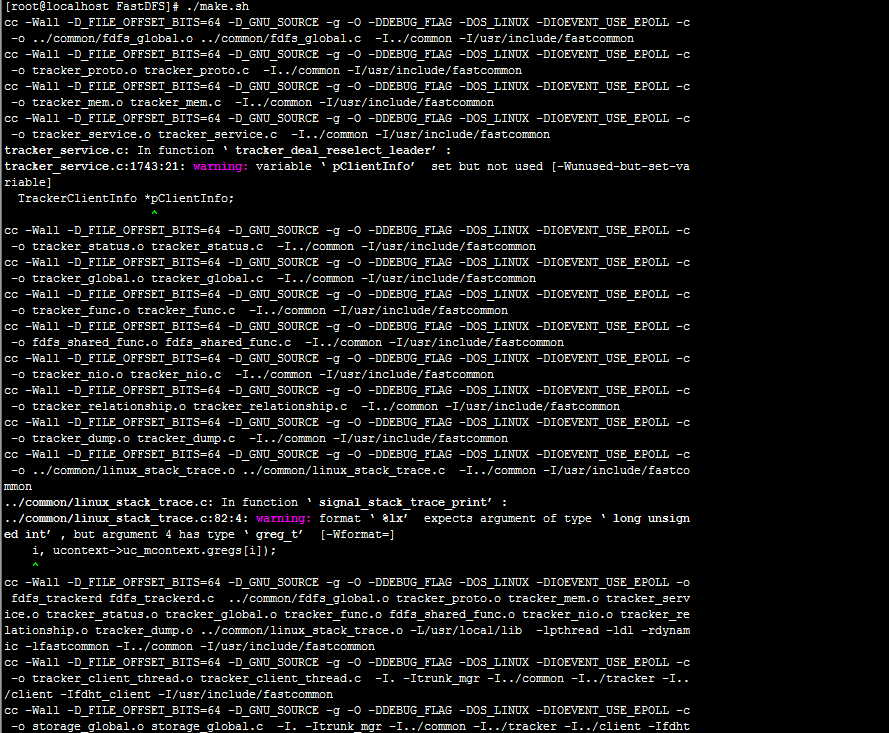
### 6、 关闭Tracker：

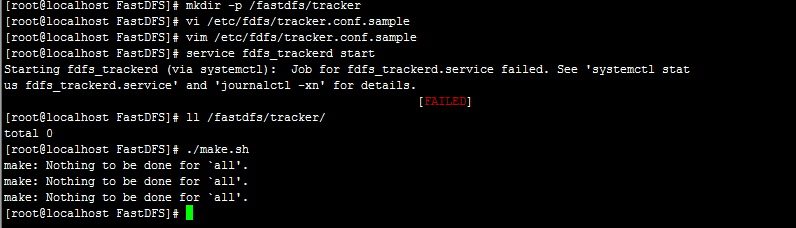
# service fdfs\_trackerd stop

### 7、 设置FastDFS跟踪器开机启动：

# chkconfig fdfs\_trackerd on

### 问题：





## 三、配置FastDFS存储(192.168.1.135、192.168.1.136 、192.168.1.137 、192.168.1.138 )

### 1、 复制FastDFS存储器样例配置文件,并重命名:

# cd /etc/fdfs/



# cp storage.conf.sample storage.conf

### 2、 编辑存储器样例配置文件（以group1中的storage节点的storage.conf为例）：

# vi /etc/fdfs/storage.conf

修改的内容如下:

disabled=false#启用配置文件

group\_name=group1#组名（第一组为group1，第二组为group2）

port=23000#storage的端口号,同一个组的storage端口号必须相同

base\_path=/fastdfs/storage#设置storage的日志目录

store\_path0=/fastdfs/storage#存储路径

store\_path\_count=1#存储路径个数，需要和store\_path个数匹配

tracker\_server=192.168.1.131:22122#tracker服务器的IP地址和端口

tracker\_server=192.168.1.132:22122#多个tracker直接添加多条配置

http.server\_port=8888#设置http端口号

（其它参数保留默认配置，具体配置解释请参考官方文档说明：

http://bbs.chinaunix.net/thread-1941456-1-1.html）

### 3、 创建基础数据目录（参考基础目录base\_path配置）:

# mkdir -p /fastdfs/storage

### 4、 防火墙中打开存储器端口（默认为23000）:

# vi /etc/sysconfig/iptables

添加如下端口行：

## FastDFS Storage Port

-A INPUT -m state --stateNEW -m tcp -p tcp --dport 23000-j ACCEPT

重启防火墙：

# service iptables restart

### 5、 启动Storage：

# service fdfs\_storaged start

（初次成功启动，会在/fastdfs/storage目录下创建数据目录data和日志目录logs）

各节点启动动，使用tail -f /fastdfs/storage/logs/storaged.log命令监听存储节点日志，可以看到存储节点链接到跟踪器，并提示哪一个为leader跟踪器。同时也会看到同一组中的其他节点加入进来的日志信息。



查看23000端口监听情况：netstat -unltp|grep fdfs



所有Storage节点都启动之后，可以在任一Storage节点上使用如下命令查看集群信息：

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



可以看到存储节点状态为ACTIVE则可

### 6、 关闭Storage：

# service fdfs\_storaged stop

### 7、 设置FastDFS存储器开机启动：

## FastDFS Storage

chkconfig fdfs\_storaged on

## 四、文件上传测试(192.168.1.131)

### 1、修改Tracker服务器中的客户端配置文件：

# cp /etc/fdfs/client.conf.sample /etc/fdfs/client.conf

#vi /etc/fdfs/client.conf

base\_path=/fastdfs/tracker

tracker\_server=192.168.1.131:22122

tracker\_server=192.168.1.132:22122

### 2、执行如下文件上传命令：

# /usr/bin/fdfs\_upload\_file /etc/fdfs/client.conf /usr/local/src/FastDFS\_v5.05.tar.gz

返回ID号：

group1/M00/00/00/wKgBh1Xtr9-AeTfWAAVFOL7FJU4.tar.gz

group2/M00/00/00/wKgBiVXtsDmAe3kjAAVFOL7FJU4.tar.gz

（能返回以上文件ID，说明文件上传成功）

## 六、在各存储节点（192.168.1.135、192.168.1.136、192.168.1.137、192.168.1.138）上安装Nginx

### 1、fastdfs-nginx-module作用说明

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入文件复制，有同步延迟的问题。假设Tracker服务器将文件上传到了192.168.1.135，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192.168.1.136，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192.168.1.136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。（解压后的fastdfs-nginx-module在nginx安装时使用）

### 2、上传fastdfs-nginx-module\_v1.16.tar.gz到/usr/local/src，解压

# cd /usr/local/src/

# tar -zxvf fastdfs-nginx-module\_v1.16.tar.gz

### 3、修改fastdfs-nginx-module的config配置文件

# vi /usr/local/src/fastdfs-nginx-module/src/config

CORE\_INCS="$CORE\_INCS /usr/local/include/fastdfs /usr/local/include/fastcommon/"

修改为：CORE\_INCS="$CORE\_INCS /usr/include/fastdfs /usr/include/fastcommon/"

（注意：这个路径修改是很重要的，不然在nginx编译的时候会报错的）

### 4、上传当前的稳定版本Nginx(nginx-1.6.2.tar.gz)到/usr/local/src目录

### 5、安装编译Nginx所需的依赖包

# yum install gcc gcc-c++ make automake autoconf libtool pcre\* pcre-devel zlib zlib-devel openssl openssl-devel

### 6、编译安装Nginx（添加fastdfs-nginx-module模块）

# cd /usr/local/src/

# tar -zxvf nginx-1.6.2.tar.gz

#cd nginx-1.6.2

./configure --prefix=/usr/local/nginx --add-module=/usr/local/src/fastdfs-nginx-module/src

# make && make install

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

# cp /usr/local/src/fastdfs-nginx-module/src/mod\_fastdfs.conf /etc/fdfs/

# vi /etc/fdfs/mod\_fastdfs.conf

#### (1)第一组Storage的mod\_fastdfs.conf配置如下：

connect\_timeout=10

base\_path=/tmp

tracker\_server=192.168.1.131:22122

tracker\_server=192.168.1.132:22122

storage\_server\_port=23000

group\_name=group1

url\_have\_group\_name = true

store\_path0=/fastdfs/storage

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

#### (2)第一组Storage的mod\_fastdfs.conf配置与第一组配置只有group\_name不同：

group\_name=group2

### 8、复制FastDFS的部分配置文件到/etc/fdfs目录

# cd /usr/local/src/FastDFS/conf

# cp http.conf mime.types /etc/fdfs/

### 9、在/fastdfs/storage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

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

ln -s /usr/local/src/FastDFS/data/storage/data/

/usr/local/src/FastDFS/data/storage/data/M00

### 10、配置Nginx，简洁版nginx配置样例：

# vi /usr/local/nginx/conf/nginx.conf

user root;

worker\_processes 1;

events {

worker\_connections 1024;

}

http {

include mime.types;

default\_type application/octet-stream;

sendfile on;

keepalive\_timeout 65;

server {

listen 8888;

server\_name localhost;

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

#alias /fastdfs/storage/data;

ngx\_fastdfs\_module;

}

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

}

}

注意、说明：

**A、8888端口值是要与/etc/fdfs/storage.conf中的http.server\_port=8888相对应，**

**因为http.server\_port默认为8888,如果想改成80，则要对应修改过来。**

**B、Storage对应有多个group的情况下，访问路径带group名，如/group1/M00/00/00/xxx，**

对应的Nginx配置为：

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

ngx\_fastdfs\_module;

}

**C、如查下载时如发现老报404，将nginx.conf第一行user nobody修改为user root后重新启动。**

11、防火墙中打开Nginx的8888端口

# vi /etc/sysconfig/iptables

添加：

## Nginx Port

-A INPUT -m state --state NEW -m tcp -p tcp --dport 8888-j ACCEPT

重启防火墙: # service iptables restart

### 12、启动Nginx

# /usr/local/nginx/sbin/nginx

ngx\_http\_fastdfs\_set pid=xxx



（重启Nginx的命令为：/usr/local/nginx/sbin/nginx-s reload）

设置Nginx开机启动

# vi /etc/rc.local

加入：

/usr/local/nginx/sbin/nginx

### 13、通过浏览器访问测试时上传的文件

http://192.168.1.135:8888/group1/M00/00/00/wKgBh1Xtr9-AeTfWAAVFOL7FJU4.tar.gz

http://192.168.1.137:8888/group2/M00/00/00/wKgBiVXtsDmAe3kjAAVFOL7FJU4.tar.gz

## 七、在跟踪器节点（192.168.1.131、192.168.1.132）上安装Nginx

### 1、在tracker上安装的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。

### 2、安装编译Nginx所需的依赖包

# yum install gcc gcc-c++ make automakeautoconf libtool pcre pcre-develzlibzlib-developenssl openssl-devel

### 3、上传ngx\_cache\_purge-2.3.tar.gz到/usr/local/src，解压

# cd /usr/local/src/

# tar -zxvf ngx\_cache\_purge-2.3.tar.gz

### 4、上传当前的稳定版本Nginx(nginx-1.6.2.tar.gz)到/usr/local/src目录

### 5、编译安装Nginx（添加ngx\_cache\_purge模块）

# cd /usr/local/src/

# tar -zxvf nginx-1.6.2.tar.gz

#cd nginx-1.6.2

# ./configure--prefix=/usr/local/nginx--add-module=/usr/local/src/ngx\_cache\_purge-2.3

# make && make install

### 6、配置Nginx，设置负载均衡以及缓存

# vi /usr/local/nginx/conf/nginx.conf

user root;

worker\_processes 1;

#error\_log logs/error.log;

#error\_log logs/error.log notice;

#error\_log logs/error.loginfo;

#pid logs/nginx.pid;

events {

worker\_connections 1024;

use epoll;

}

http {

include mime.types;

default\_type application/octet-stream;

#log\_format main '$remote\_addr -$remote\_user [$time\_local] "$request" '

# '$status $body\_bytes\_sent "$http\_referer" '

# '"$http\_user\_agent" "$http\_x\_forwarded\_for"';

#access\_log logs/access.log main;

sendfile on;

tcp\_nopush on;

#keepalive\_timeout 0;

keepalive\_timeout 65;

#gzip on;

#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 X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_connect\_timeout 90;

proxy\_send\_timeout 90;

proxy\_read\_timeout 90;

proxy\_buffer\_size 16k;

proxy\_buffers 4 64k;

proxy\_busy\_buffers\_size 128k;

proxy\_temp\_file\_write\_size 128k;

#设置缓存存储路径、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/tmp;

#设置group1的服务器

upstream fdfs\_group1 {

server 192.168.1.135:8888 weight=1 max\_fails=2 fail\_timeout=30s;

server 192.168.1.136:8888 weight=1 max\_fails=2 fail\_timeout=30s;

}

#设置group2的服务器

upstream fdfs\_group2 {

server 192.168.1.137:8888 weight=1 max\_fails=2 fail\_timeout=30s;

server 192.168.1.138:8888 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;

}

}

}

按以上nginx配置文件的要求，创建对应的缓存目录：

# mkdir -p /fastdfs/cache/nginx/proxy\_cache

# mkdir -p /fastdfs/cache/nginx/proxy\_cache/tmp

7、系统防火墙打开对应的端口

# vi /etc/sysconfig/iptables

## Nginx

-A INPUT -m state --state NEW -m tcp -p tcp --dport 8000 -j ACCEPT

# service iptables restart

8、启动Nginx

# /usr/local/nginx/sbin/nginx

重启Nginx

# /usr/local/nginx/sbin/nginx-s reload

设置Nginx开机启动

# vi /etc/rc.local

加入：/usr/local/nginx/sbin/nginx

9、文件访问测试

前面直接通过访问Storage节点中的Nginx的文件

http://192.168.1.135:8888/group1/M00/00/00/wKgBh1Xtr9-AeTfWAAVFOL7FJU4.tar.gz

http://192.168.1.137:8888/group2/M00/00/00/wKgBiVXtsDmAe3kjAAVFOL7FJU4.tar.gz

现在可以通过Tracker中的Nginx来进行访问

(1)通过Tracker1中的Nginx来访问

http://192.168.1.131:8000/group1/M00/00/00/wKgBh1Xtr9-AeTfWAAVFOL7FJU4.tar.gz

http://192.168.1.131:8000/group2/M00/00/00/wKgBiVXtsDmAe3kjAAVFOL7FJU4.tar.gz

(2)通过Tracker2中的Nginx来访问

http://192.168.1.132:8000/group1/M00/00/00/wKgBh1Xtr9-AeTfWAAVFOL7FJU4.tar.gz

http://192.168.1.132:8000/group2/M00/00/00/wKgBiVXtsDmAe3kjAAVFOL7FJU4.tar.gz

由上面的文件访问效果可以看到，每一个Tracker中的Nginx都单独对后端的Storage组做了负载均衡，但整套FastDFS集群如果想对外提供统一的文件访问地址，还需要对两个Tracker中的Nginx进行HA集群。

八、使用Keepalived + Nginx 组成的高可用负载均衡集群做两个Tracker 节点中Nginx 的负载均衡

1、《Dubbo 视频教程--高可用架构篇--第08 节--Keepalived+Nginx 实现高可用负载均衡》

2、在Keepalived+Nginx 实现高可用负载均衡集群中配置Tracker 节点中Nginx 的负载均衡反向代理

(192.168.1.51 和192.168.1.52 中的Nginx 执行相同的配置)

# vi /usr/local/nginx/conf/nginx.conf

user root;

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;

}

http {

include mime.types;

default\_type application/octet-stream;

#log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

# '$status $body\_bytes\_sent "$http\_referer" '

# '"$http\_user\_agent" "$http\_x\_forwarded\_for"';

#access\_log logs/access.log main;

sendfile on;

#tcp\_nopush on;

#keepalive\_timeout 0;

keepalive\_timeout 65;

#gzip on;

## FastDFS Tracker Proxy

upstream fastdfs\_tracker {

server 192.168.1.131:8000 weight=1 max\_fails=2 fail\_timeout=30s;

server 192.168.1.132:8000 weight=1 max\_fails=2 fail\_timeout=30s;

}

server {

listen 88;

server\_name localhost;

#charset koi8-r;

#access\_log logs/host.access.log main;

location / {

root html;

index index.html index.htm;

}

#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;

}

## FastDFS Proxy

location /dfs {

root html;

index index.html index.htm;

proxy\_pass http://fastdfs\_tracker/;

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\_set\_header X-Forwarded-Proto $scheme;

client\_max\_body\_size 300m;

}

}

}

3、重启192.168.1.51和192.168.1.52中的Nginx

# /usr/local/nginx/sbin/nginx -s reload

4、通过Keepalived+Nginx组成的高可用负载集群的VIP(192.168.1.50)来访问FastDFS集群中的文件

http://192.168.1.50:88/dfs/group1/M00/00/00/wKgBh1Xtr9-AeTfWAAVFOL7FJU4.tar.gz

<http://192.168.1.50:88/dfs/group2/M00/00/00/wKgBiVXtsDmAe3kjAAVFOL7FJU4.tar.gz>

九、FastDFS集群在简易版支付系统中的使用

具体内容请看视频教程

注意：千万不要使用kill -9命令强杀FastDFS进程，否则可能会导致binlog数据丢失。