FastDFS 分布式文件系统集群安装与配置

目录

一、 FastDFS 集群规划

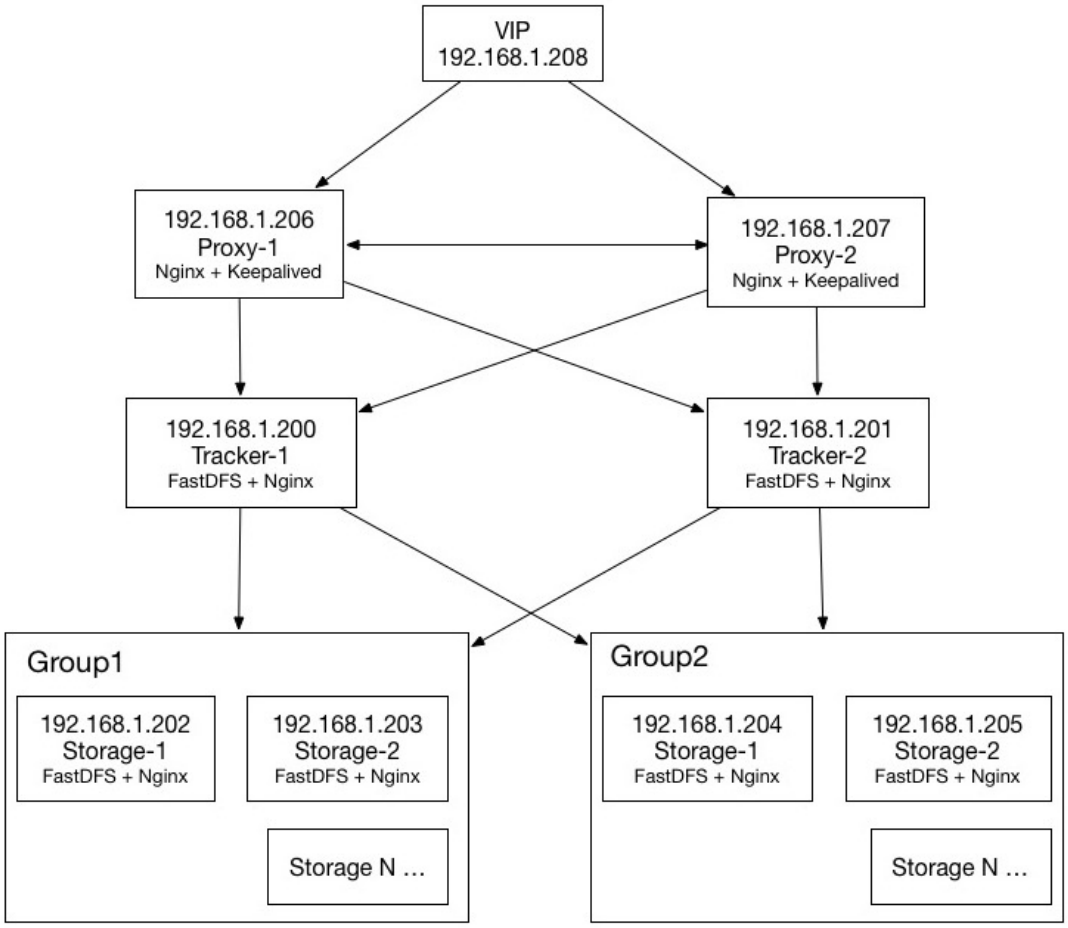
跟踪服务器负载均衡节点 1：192.168.1.206 dfs-nginx-proxy-1
跟踪服务器负载均衡节点 2：192.168.1.207 dfs-nginx-proxy-2
跟踪服务器 1：192.168.1.200 dfs-tracker-1
跟踪服务器 2：192.168.1.201 dfs-tracker-2
存储服务器 1：192.168.1.202 dfs-storage-group1-1
存储服务器 2：192.168.1.203 dfs-storage-group1-2
存储服务器 3：192.168.1.204 dfs-storage-group2-1
存储服务器 3：192.168.1.205 dfs-storage-group2-2
HA 虚拟 IP：192.168.1.208
HA 软件：Keepalived
操作系统：CentOS 7
用户：root
数据目录：/fastdfs

安装包：
fastdfs-master-V5.05.zip：FastDFS 源码
libfastcommon-master.zip：（从 FastDFS 和 FastDHT 中提取出来的公共 C 函数库）
fastdfs-nginx-module-master.zip：storage 节点 http 服务 nginx 模块
nginx-1.10.0.tar.gz：Nginx 安装包
ngx_cache_purge-2.3.tar.gz：Nginx 图片缓存清除模块
获取安装包的方式：
1> 从这里下载打包好的所有安装包：http://download.csdn.net/detail/xyang81/9667493
2> 从作者 github 官网挨个下载 fastdfs 源码及其依赖库：https://github.com/happyfish100 和 Nginx 缓存清除模块：https://github.com/FRiCKLE/ngx_cache_purge

开始前，先将所有安装包下载到各个节点的/usr/local/src 目录中。

1> 本文称节点 IP 最后一段就代表某个节点，如：192.168.1.206，文中提到 206 节点，就代表 192.168.1.206。
2> 本文称 tracker 或跟踪服务器是同一个意思
3> 本文称 storage 或存储服务器是同一个意思

二、 FastDFS 集群架构图



FastDFS集群

三、 安装集群节点

A. 安装所需的依赖包

```
shell> yum install make cmake gcc gcc-c++
```

B. 安装 libfatscommon

```
shell> cd /usr/local/src
shell> unzip libfastcommon-master.zip
shell> cd libfastcommon-master
## 编译、安装
shell> ./make.sh
shell> ./make.sh install
```

C. 安装 FastDFS

```
shell> cd /usr/local/src
shell> unzip fastdfs-master-V5.05.zip
shell> cd fastdfs-master
## 编译、安装
```

```
shell> ./make.sh
shell> ./make.sh install
```

四、 配置跟踪节点(192.168.1.200　192.168.1.201)

说明：每个节点执行相同的操作

A. 复制 tracker 样例配置文件并重命名

```
shell> cp /etc/fdfs/tracker.conf.sample
/etc/fdfs/tracker.conf
```

B. 修改 trackerconf 配置文件

```
shell> vim /etc/fdfs/tracker.conf
# 修改的内容如下：
disabled=false              # 启用配置文件
port=22122                  # tracker 服务器端口（默认 22122）
base_path=/fastdfs/tracker  # 存储日志和数据的根目录
```

C. 创建 base_path 指定的目录

```
shell> mkdir -p /fastdfs/tracker
```

D. 防火墙中打开 tracker 服务器端口 默认为 22122

```
shell> vi /etc/sysconfig/iptables
```

添加如下端口行：

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

重启防火墙：

```
shell> service iptables restart
```

E. 启动 tracker 服务器

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

初次启动，会在/fastdfs/tracker 目录下生成 logs、data 两个目录：

```
[root@localhost tracker]# pwd
/fastdfs/tracker
[root@localhost tracker]# ll
总用量 4
drwxr-xr-x. 2 root root 4096 10月 17 15:33 data
drwxr-xr-x. 2 root root   25 10月 17 15:31 logs
```

检查 FastDFS Tracker Server 是否启动成功：`ps -ef | grep fdfs_trackerd`

```
[root@localhost tracker]# ps -ef | grep fdfs_tracker
root     18143     1  0 15:53 ?        00:00:00 /usr/bin/fdfs_trackerd /etc/fdfs/tracker.conf
root     18156  7378  0 15:53 pts/0    00:00:00 grep --color=auto fdfs_tracker
[root@localhost tracker]#
```

F. 停止 tracker 服务器

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

G. 设置 tracker 服务开机启动

```
shell> chkconfig fdfs_trakcerd on
```

五、 配置存储节点

group1： 192.168.1.202，192.168.1.203

group2： 192.168.1.204，192.168.1.205

说明：每个节点执行相同的操作

A. 复制 storage 样例配置文件并重命名

```
shell> cp /etc/fdfs/storage.conf.sample
/etc/fdfs/storage.conf
```

B. 编辑配置文件

```
shell> vi /etc/fdfs/storage.conf
# 修改的内容如下:
disabled=false   # 启用配置文件
port=23000   # storage 服务端口
group_name=group1  # 组名（第一组为 group1，第二组为 group2，依次类推...）
base_path=/fastdfs/storage   # 数据和日志文件存储根目录
store_path0=/fastdfs/storage   # 第一个存储目录，第二个存储目录起名为：store_path1=xxx，其它存储目录名依次类推...
store_path_count=1   # 存储路径个数，需要和 store_path 个数匹配
tracker_server=192.168.0.200:22122  # tracker 服务器 IP 和端口
tracker_server=192.168.0.201:22122  # tracker 服务器 IP 和端口
http.server_port=8888   # http 访问文件的端口
```

其它参数保留默认配置

C. 创建基础数据目录

```
shell> mkdir -p /fastdfs/storage
```

D. 防火墙中打开 storage 服务器端口 默认为 23000

```
shell> vi /etc/sysconfig/iptables
```

添加如下端口行：

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

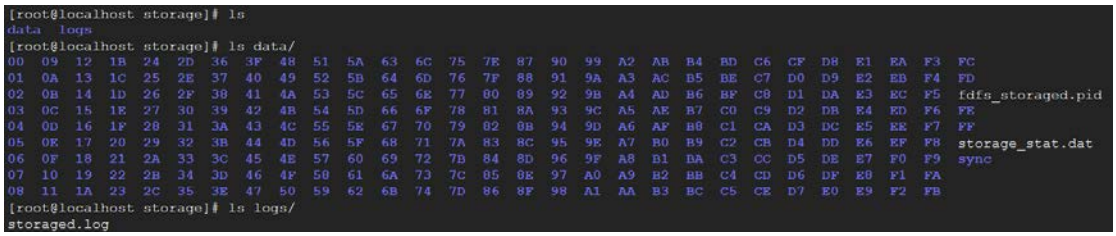
重启防火墙：

```
shell> service iptables restart
```

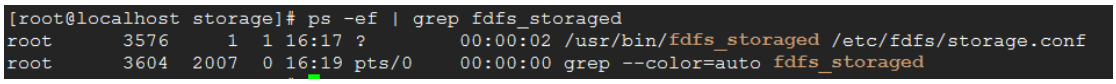
E. 启动 storage 服务器

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

初次启动，会在/fastdfs/storage 目录下生成 logs、data 两个目录



检查 FastDFS Tracker Server 是否启动成功：**ps -ef | grep** fdfs_storaged



各节点启动后，使用 **tail** -f /fastdfs/storage/logs/storaged.log 命令监听存储节点的日志，可以看到存储节点链接到跟踪服务器，并提示哪一个为 leader 跟踪服务器，同时也能看到同一组中其它节点加入进来的日志信息。

所有存储节点都启动之后，可以在任一存储节点上使用如下命令查看集群的状态信息：

```
1   shell> /usr/bin/fdfs_monitor /etc/fdfs/storage.conf
```

F. 停止 storage 服务器

```
1   shell> /etc/init.d/fdfs_storaged stop
```

G. 设置 storage 服务开机启动

```
1   shell> chkconfig fdfs_storaged on
```

六、 文件上传测试

A. 修改 tracker 服务器 clientconf 配置文件

```
1   shell> cp /etc/fdfs/client.conf.sample
    /etc/fdfs/client.conf
2   shell> vi /etc/fdfs/client.conf
3   base_path=/fastdfs/tracker
4   tracker_server=192.168.1.200:22122
5   tracker_server=192.168.1.201:22122
```

B. 执行文件上传命令

```
1   shell> /usr/bin/fdfs_upload_file /etc/fdfs/client.conf
    /usr/local/src/FastDFS_v5.05.tar.gz
```

返回以下 ID 号，说明文件上传成功：

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

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

（从返回的 ID 号中也可以看出，同一个文件分别存储在两个组内 group1 和 group2，但也有可能在同一组中，具体策略是由 FastDFS 根据服务器的存储情况来分配的）

七、 储节点安装 Nginx 和 fastdfs-nginx-module 模块

说明：每个节点执行相同的操作

A. 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 192.168.1.202，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 192.168.1.203，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 192.168.1.203 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（192.168.1.202）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。

B. 安装 nginx 和 fastdfs-nginx-module 模块

```
## 安装 nginx 所需的依赖包
shell> yum install gcc gcc-c++ make automake autoconf libtool pcre pcre-devel zlib zlib-devel openssl openssl-devel
## 编译安装 nginx（添加 fastdfs-nginx-module 模块）
shell> cd /usr/local/src
shell> tar -zxvf nginx-1.10.0.tar.gz
shell> unzip fastdfs-nginx-module-master.zip
shell> cd nginx-1.10.0
shell> ./configure --prefix=/opt/nginx --sbin-path=/usr/bin/nginx --add-module=/usr/local/src/fastdfs-nginx-module/src
shell> make && make install
```

C. 复制 fastdfs-nginx-module 源码中的配置文件到 etcfdfs 目录并修改

```
shell> cp /usr/local/src/fastdfs-nginx-module/src/mod_fastdfs.conf /etc/fdfs/
shell> vi /etc/fdfs/mod_fastdfs.conf
```

1. 第一组存储服务器的 mod_fastdfsconf 配置

```
connect_timeout=10
base_path=/tmp
tracker_server=192.168.1.200:22122
tracker_server=192.168.1.201:22122
storage_server_port=23000
group_name=group1      # 第一组 storage 的组名
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
```

```
17 storage_server_port=23000
18 store_path_count=1
19 store_path0=/fastdfs/storage
```

2. 第二组存储服务器的 mod_fastdfsconf 配置

```
1  connect_timeout=10
2  base_path=/tmp
3  tracker_server=192.168.1.200:22122
4  tracker_server=192.168.1.201:22122
5  storage_server_port=23000
6  group_name=group2      # 第二组 storage 的组名
7  url_have_group_name=true
8  store_path0=/fastdfs/storage
9  group_count=2
10 [group1]
11 group_name=group1
12 storage_server_port=23000
13 store_path_count=1
14 store_path0=/fastdfs/storage
15 [group2]
16 group_name=group2
17 storage_server_port=23000
18 store_path_count=1
19 store_path0=/fastdfs/storage
```

D. 复制 FastDFS 源文件目录中 HTTP 相关的配置文件到 etcfdfs 目录

```
1  shell> cd /usr/local/src/FastDFS/conf
2  shell> cp http.conf mime.types /etc/fdfs/
```

E. 创建数据存放目录的软链接

```
1  shell> ln -s /fastdfs/storage/data/
   /fastdfs/storage/data/M00
```

F. 配置 fastdfs-nginx-module（inx 简洁版样例）

```
1  shell> vi /opt/nginx/conf/nginx.conf
2  user nobody;
3  worker_processes 1;
4  events {
5     worker_cnnections  1024;
6  }
7  http {
8     include     mime.types;
9     default_type  application/octet-stream;
10    sendfile       on;
```

```
    keepalive_timeout  65;
    server {
        listen       8888;
        server_name  localhost;
        # FastDFS 文件访问配置(fastdfs-nginx-module 模块)
        location ~/group([0-9])/M00 {
            ngx_fastdfs_module;
        }
        error_page 500 502 503 504 /50x.html;
        location = /50x.html {
            root   html;
        }
    }
}
```

注意：

a、8888 端口值要与/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：http://xxxx/group1/M00/00/00/xxx，对应的 Nginx 配置为：

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

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

G. 防火墙中打开 Nginx 的 8888 端口

```
shell> vi /etc/sysconfig/iptables
## 添加
-A INPUT -m state --state NEW -m tcp -p tcp --dport 8888 -j ACCEPT
## 重启防火墙
shell> service iptables restart
```

H. 启动 Nginx

```
shell> /opt/nginx/sbin/nginx
ngx_http_fastdfs_set pid=xxx   astdfs-nginx-module 进程 ID
```

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

设置 Nginx 开机启动：

```
shell> vi /etc/rc.local
# 加入
/opt/nginx/sbin/nginx
shell> chmod +x /etc/rc.local  # centos7
```

I. 通过浏览器访问测试时上传的文件

http://192.168.1.202:8888/group1/M00/00/00/wKgBh1Xtr9-AeTfWAAVFOL7FJU4.tar.gz
http://192.168.1.204:8888/group2/M00/00/00/wKgBiVXtsDmAe3kjAAVFOL7FJU4.tar.gz

八、 跟踪节点安装 Nginx 和 ngx_cache_purge 模块

说明：每个节点执行相同的操作。

tracker 节点：192.168.1.200，192.168.1.201

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。

A. 安装 Nginx 所需的依赖包

```
shell> yum install gcc gcc-c++ make automake autoconf libtool
pcre pcre-devel zlib zlib-devel openssl openssl-devel
```

B. 安装 nginx 和 ngx_cache_purge 模块

```
shell> cd /usr/local/src
shell> tar -zxvf nginx-1.10.0.tar.gz
shell> tar -zxvf ngx_cache_purge-2.3.tar.gz
shell> cd nginx-1.10.0
shell> ./configure --prefix=/opt/nginx
--sbin-path=/usr/bin/nginx
--add-module=/usr/local/src/ngx_cache_purge-2.3
shell> make && make install
```

C. 配置 Nginx 设置 tracker 负载均衡以及缓存

```
shell> vi /opt/nginx/conf/nginx.conf
user  nobody;
worker_processes  1;
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  65;
```

```
18      #gzip on;
19
20      #设置缓存
21      server_names_hash_bucket_size 128;
22      client_header_buffer_size 32k;
23      large_client_header_buffers 4 32k;
24      client_max_body_size 300m;
25      proxy_redirect off;
26      proxy_set_header Host $http_host;
27      proxy_set_header X-Real-IP $remote_addr;
28      proxy_set_header X-Forwarded-For
    $proxy_add_x_forwarded_for;    proxy_connect_timeout 90;
29      proxy_send_timeout 90;
30      proxy_read_timeout 90;
31      proxy_buffer_size 16k;
32      proxy_buffers 4 64k;
33      proxy_busy_buffers_size 128k;
34      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
35      proxy_temp_path /fastdfs/cache/nginx/proxy_cache/tmp;
36
37      #设置 group1 的服务器
38      upstream fdfs_group1 {
39      server 192.168.1.202:8888 weight=1 max_fails=2
    fail_timeout=30s;
40      server 192.168.1.203:8888 weight=1 max_fails=2
    fail_timeout=30s;
41      }
42
43      #设置 group2 的服务器
44      upstream fdfs_group2 {
45      server 192.168.1.204:8888 weight=1 max_fails=2
    fail_timeout=30s;
46      server 192.168.1.205:8888 weight=1 max_fails=2
    fail_timeout=30s;
47      }
48
49      server {
50              listen      8000;
51              server_name  localhost;
52              #charset koi8-r;
53              #access_log  logs/host.access.log  main;
54
55              #设置 group 的负载均衡参数
```

```
56              location /group1/M00 {
57                              proxy_next_upstream http_502
    http_504 error timeout invalid_header;
58                              proxy_cache http-cache;
59                              proxy_cache_valid  200 304 12h;
60                              proxy_cache_key
    $uri$is_args$args;
61                              proxy_pass http://fdfs_group1;
62                              expires 30d;
63              }
64
65              location /group2/M00 {
66                              proxy_next_upstream http_502
    http_504 error timeout invalid_header; proxy_cache
    http-cache;
67                              proxy_cache_valid 200 304 12h;
68                              proxy_cache_key
    $uri$is_args$args;
69                              proxy_pass http://fdfs_group2;
70                              expires 30d;
71              }
72
73              #设置清除缓存的访问权限
74              location ~/purge(/.*) {
75                              allow 127.0.0.1;
76                              allow 192.168.1.0/24;
77                              deny all;
78                              proxy_cache_purge http-cache
    $uri$is_args$args;
79              }
80              #error_page  404              /404.html;
81              # redirect server error pages to the static page
    /50x.html
82              error_page   500 502 503 504  /50x.html;
83              location = /50x.html {
84                              root html;
85              }
86  }
```

按以上 nginx 配置文件的要求，创建对应的缓存目录：

```
1   shell> mkdir -p /fastdfs/cache/nginx/proxy_cache
2   shell> mkdir -p /fastdfs/cache/nginx/proxy_cache/tmp
```

D. 防火墙打开 Nginx 8000 端口

```
shell> vi /etc/sysconfig/iptables
## 添加如下配置
-A INPUT -m state --state NEW -m tcp -p tcp --dport 8000 -j ACCEPT
shell> service iptables restart # 重新启动防火墙
```

E. 启动 Nginx

```
shell> /opt/nginx/sbin/nginx
```

设置开机启动：

```
shell> vi /etc/rc.local
## 加入以下配置
/opt/nginx/sbin/nginx
shell> chmod +x /etc/rc.local  #centos7
```

F. 文件访问测试

前面直接通过访问 Storage 节点中的 Nginx 访问文件：

http://192.168.1.202:8888/group1/M00/00/00/wKgBh1Xtr9-AeTfWAAVFOL7FJU4.tar.gz

http://192.168.1.204:8888/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://192.168.1.200:8000/group1/M00/00/00/wKgBh1Xtr9-AeTfWAAVFOL7FJU4.tar.gz

http://192.168.1.200:8000/group2/M00/00/00/wKgBiVXtsDmAe3kjAAVFOL7FJU4.tar.gz

(2)、通过 Tracker2 中的 Nginx 来访问

http://192.168.1.201:8000/group1/M00/00/00/wKgBh1Xtr9-AeTfWAAVFOL7FJU4.tar.gz

http://192.168.1.201:8000/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

九、 配置 Tracker 服务器高可用反向代理与负载均衡

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。

A. 安装 keepalived 与 Nginx

分别在 192.168.1.206 和 192.168.1.207 两个节点安装 Keepalived 与 Nginx。

keepalived 安装与配置：http://blog.csdn.net/xyang81/article/details/52554398

Nginx 的安装与配置：http://blog.csdn.net/xyang81/article/details/51476293

B. 配置 Keeyalived Nginx 高可用

请参考《Keepalived+Nginx 实现高可用（HA）》

注意：将 VIP 的 IP 地址修改为 192.168.1.208

C. 配置 nginx 对 tracker 节点的负载均衡

2 个节点的 Nginx 配置相同，如下所示：

```
shell> vi /opt/nginx/conf/nginx.conf
user  root;
worker_processes  1;
events {
worker_connections  1024;
use epool;
}
http {
include      mime.types;
default_type  application/octet-stream;
sendfile       on;
keepalive_timeout  65;
## FastDFS Tracker Proxy
upstream fastdfs_tracker {
server 192.168.1.200:8000 weight=1 max_fails=2 fail_timeout=30s;
server 192.168.1.201:8000 weight=1 max_fails=2 fail_timeout=30s;
}
server {
listen      80;
server_name  localhost;
location / {
root html;
index  index.html index.htm;
}
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;
```

```
38  proxy_set_header X-Forwarded-Proto $scheme;
39  client_max_body_size  300m;
40  }
41  }
42  }
```

D. 重启 1921681206 和 1921681207 中的 Nginx

```
1   shell> /opt/nginx/sbin/nginx -s reload
```

E. 通过虚拟 IP 访问文件测试

现在可以通过 Keepalived+Nginx 组成的高可用负载集群的 VIP(192.168.1.208)来访问 FastDFS 集群中的文件了：

http://192.168.1.208/dfs/group1/M00/00/00/wKgBh1Xtr9-AeTfWAAVFOL7FJU4.tar.gz

http://192.168.1.208/dfs/group2/M00/00/00/wKgBiVXtsDmAe3kjAAVFOL7FJU4.tar.gz

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。