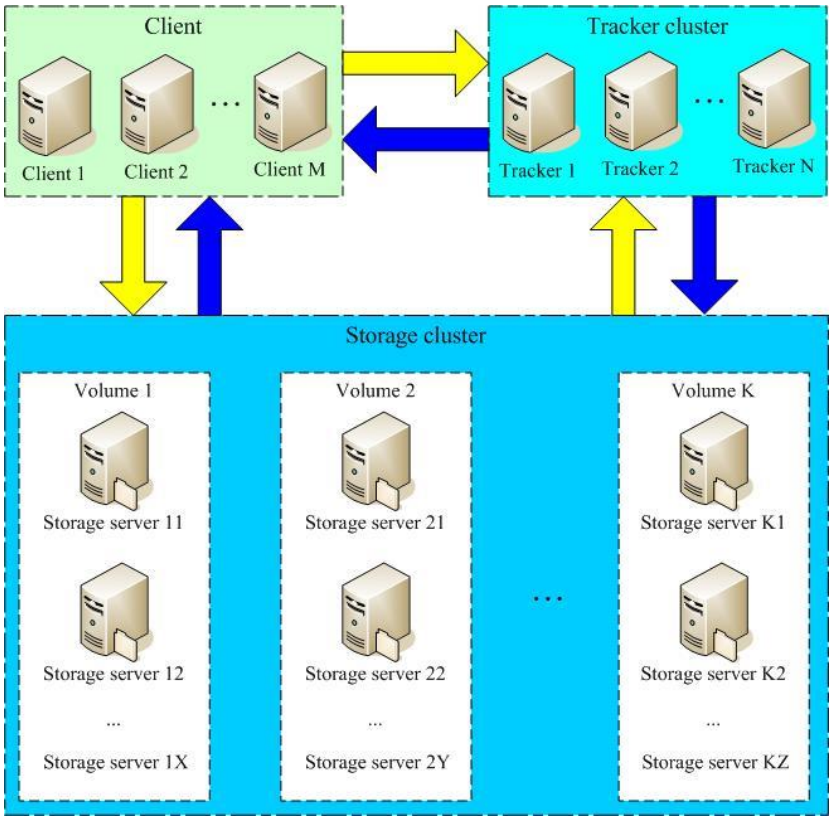
高可用架构篇

FastDFS 集群的安装、配置、使用

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。

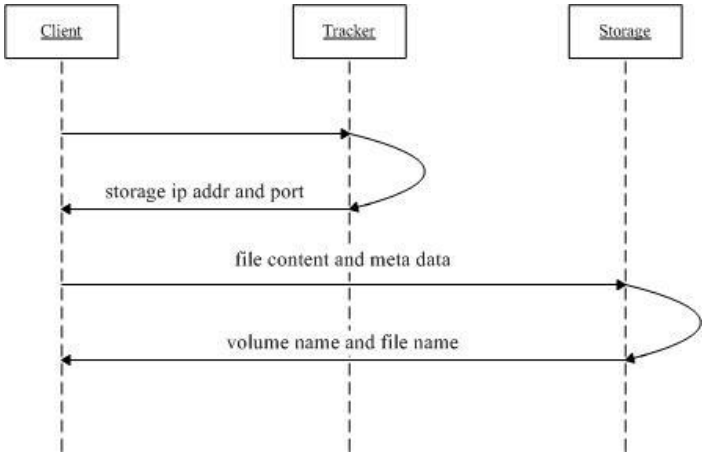
FastDFS 服务端有两个角色：跟踪器（tracker）和存储节点（storage）。跟踪器主要做调度工作，在访问上起负载均衡的作用。存储节点存储文件，完成文件管理的所有功能：存储、同步和提供存取接口，FastDFS 同时对文件的 meta data 进行管理。所谓文件的 meta data 就是文件的相关属性，以键值对（key value pair）方式表示，如：width=1024，其中的 key 为 width，value 为 1024。文件 meta data 是文件属性列表，可以包含多个键值对。FastDFS 系统结构如下图所示：



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 中的文件标识分为两个部分：卷名和文件名，二者缺一不可。

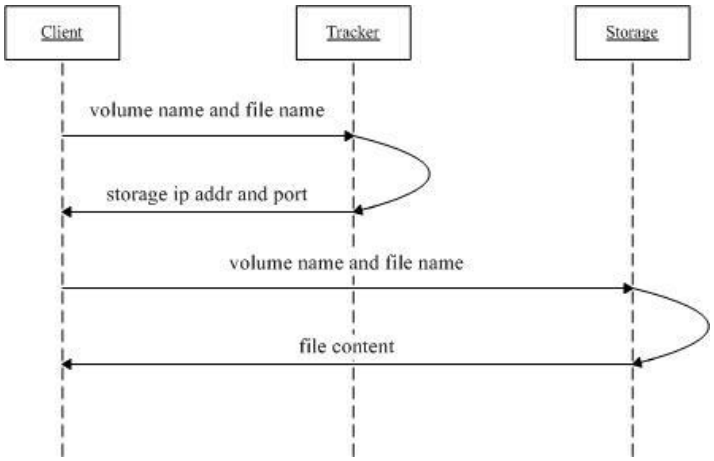
FastDFS 上传文件交互过程：



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。

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　　edu-dfs-tracker-1

跟踪服务器 2：192.168.1.132　　edu-dfs-tracker-2

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

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

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

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

环境：CentOS 6.6

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（随视频压缩包提供）：

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

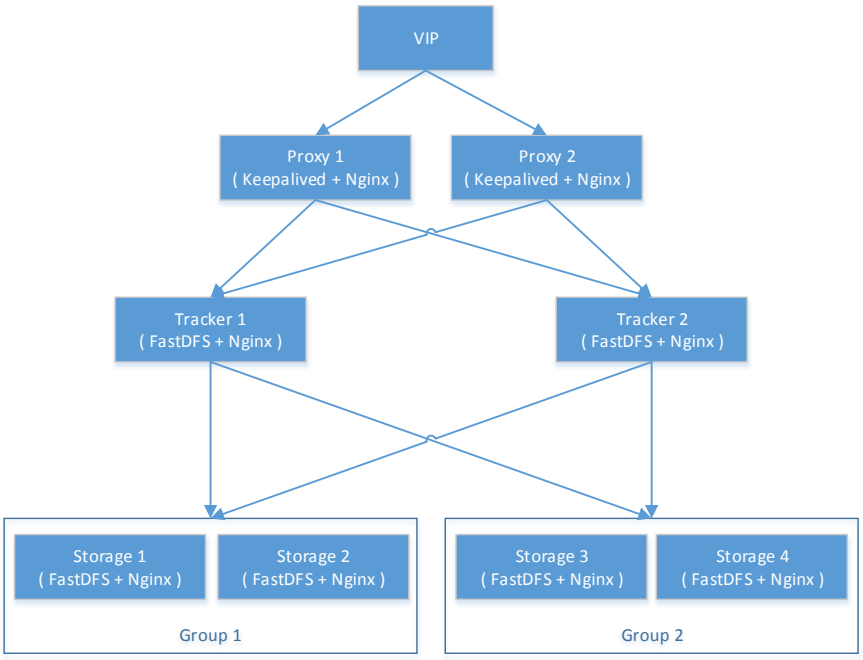
nginx-1.6.2.tar.gz

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

本教程的 FastDFS 文件系统集群的最终结构图

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
```

```
[root@edu-dfs-tracker-01 libfastcommon-master]# ll
total 28
-rw-r--r--. 1 root root 2913 Feb 27 17:27 HISTORY
-rw-r--r--. 1 root root  582 Feb 27 17:27 INSTALL
-rw-r--r--. 1 root root 1342 Feb 27 17:27 libfastcommon.spec
-rwxr-xr-x. 1 root root 2151 Feb 27 17:27 make.sh
drwxr-xr-x. 2 root root 4096 Feb 27 17:27 php-fastcommon
-rw-r--r--. 1 root root  617 Feb 27 17:27 README
drwxr-xr-x. 2 root root 4096 Feb 27 17:27 src
```

(3) 编译、安装

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

libfastcommon 默认安装到了

/usr/lib64/libfastcommon.so

/usr/lib64/libfdfsclient.so

(4)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
```

```
[root@edu-dfs-tracker-01 FastDFS]# ll
total 132
drwxr-xr-x. 3 8980 users  4096 Dec  2 11:26 client
drwxr-xr-x. 2 8980 users  4096 Dec  2 11:27 common
drwxr-xr-x. 2 8980 users  4096 Dec  2 11:26 conf
-rw-r--r--. 1 8980 users 35067 Dec  2 11:26 COPYING-3_0.txt
-rw-r--r--. 1 8980 users  2802 Dec  2 11:26 fastdfs.spec
-rw-r--r--. 1 8980 users 31386 Dec  2 11:27 HISTORY
drwxr-xr-x. 2 8980 users  4096 Dec  2 11:26 init.d
-rw-r--r--. 1 8980 users  7755 Dec  2 11:26 INSTALL
-rwxr-xr-x. 1 8980 users  5813 Dec  2 11:27 make.sh
drwxr-xr-x. 2 8980 users  4096 Dec  2 11:26 php_client
-rw-r--r--. 1 8980 users  2380 Dec  2 11:26 README.md
-rwxr-xr-x. 1 8980 users  1768 Dec  2 11:26 restart.sh
-rwxr-xr-x. 1 8980 users  1680 Dec  2 11:26 stop.sh
drwxr-xr-x. 4 8980 users  4096 Dec  2 11:27 storage
drwxr-xr-x. 2 8980 users  4096 Dec  2 11:26 test
drwxr-xr-x. 2 8980 users  4096 Dec  2 11:27 tracker
```

(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）：

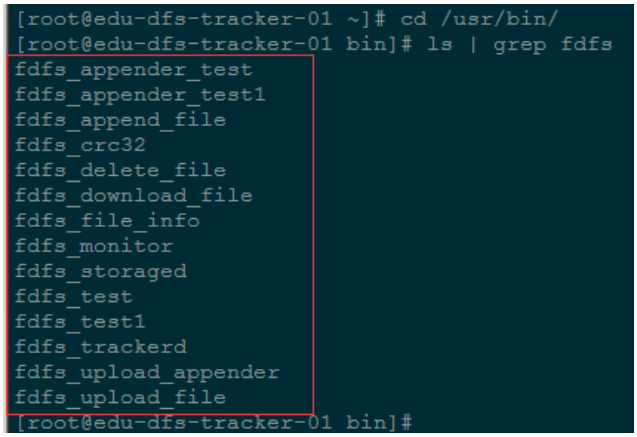
```
/etc/fdfs/client.conf.sample
/etc/fdfs/storage.conf.sample
/etc/fdfs/tracker.conf.sample
```

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)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/

```
[root@edu-dfs-tracker-01 fdfs]# ll
total 20
-rw-r--r--. 1 root root 1461 Mar 25 23:15 client.conf.sample
-rw-r--r--. 1 root root 7829 Mar 25 23:15 storage.conf.sample
-rw-r--r--. 1 root root 7102 Mar 25 23:15 tracker.conf.sample
```

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：

```
disabled=false                  #启用配置文件
port=22122                      #tracker 的端口号，一般采用 22122 这个默认端口
base_path=/fastdfs/tracker      #tracker 的数据文件和日志目录
```

（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/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

（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

```
[root@edu-dfs-tracker-1 data]# netstat -unltp|grep fdfs
tcp        0      0 0.0.0.0:22122               0.0.0.0:*                   LISTEN      4277/fdfs_trackerd
[root@edu-dfs-tracker-1 data]#
```

(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：

/etc/init.d/fdfs_trackerd stop

7、 设置 FastDFS 跟踪器开机启动：

vi /etc/rc.d/rc.local

添加以下内容：

FastDFS Tracker

/etc/init.d/fdfs_trackerd start

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/

```
[root@edu-dfs-storage-group1-1 FastDFS]# cd /etc/fdfs/
[root@edu-dfs-storage-group1-1 fdfs]# ll
total 20
-rw-r--r--. 1 root root 1461 Aug 31 05:13 client.conf.sample
-rw-r--r--. 1 root root 7829 Aug 31 05:13 storage.conf.sample
-rw-r--r--. 1 root root 7102 Aug 31 05:13 tracker.conf.sample
[root@edu-dfs-storage-group1-1 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 --state NEW -m tcp -p tcp --dport 23000 -j ACCEPT
```

重启防火墙：

```
# service iptables restart
```

5、 启动 Storage：

```
# /etc/init.d/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。

```
[2015-08-31 06:29:24] INFO - file: storage_func.c, line: 254, tracker_client_ip: 192.168.1.135, my_server_id_str: 192.168.1.135, g_server_id_in_filename: -2029934400
[2015-08-31 06:29:24] INFO - local_host_ip_count: 2, 127.0.0.1 192.168.1.135
[2015-08-31 06:29:24] INFO - file: tracker_client_thread.c, line: 310, successfully connect to tracker server 192.168.1.131:22122, as a tracker client, my ip is 192.168.1.135
[2015-08-31 06:29:24] INFO - file: tracker_client_thread.c, line: 310, successfully connect to tracker server 192.168.1.132:22122, as a tracker client, my ip is 192.168.1.135
[2015-08-31 06:29:54] INFO - file: tracker_client_thread.c, line: 1235, tracker server 192.168.1.132:22122, set tracker leader: 192.168.1.132:22122
[2015-08-31 06:30:54] INFO - file: storage_sync.c, line: 2698, successfully connect to storage server 192.168.1.136:23000
[2015-08-31 06:31:24] INFO - file: storage_sync.c, line: 2698, successfully connect to storage server 192.168.1.136:23000
```

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

```
[root@edu-dfs-storage-group1-1 logs]# netstat -unltp|grep fdfs
tcp        0      0 0.0.0.0:23000               0.0.0.0:*                   LISTEN      2668/fdfs_storaged
[root@edu-dfs-storage-group1-1 logs]#
```

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
```

```
[root@edu-dfs-storage-group1-1 logs]# /usr/bin/fdfs_monitor /etc/fdfs/storage.conf
[2015-08-31 06:48:12] DEBUG - base_path=/fastdfs/storage, connect_timeout=30, networ
pool_max_idle_time=3600s, use_storage_id=0, storage server id count: 0

server_count=2, server_index=0

tracker server is 192.168.1.131:22122

group count: 2

Group 1:
group name = group1
disk total space = 7934 MB
disk free space = 4829 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 = 192.168.1.135
                ip_addr = 192.168.1.135  ACTIVE
                http domain =
                version = 5.05
                join time = 2015-08-31 06:29:18
                up time = 2015-08-31 06:29:18
                total storage = 7934 MB
                free storage = 4831 MB
                upload priority = 10
                store_path_count = 1
                subdir_count_per_path = 256
                storage_port = 23000
                storage_http_port = 8888
                current_write_path = 0
                source storage id =
```

可以看到存储节点状态为 ACTIVE 则可

6、 关闭 Storage：

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

7、 设置 FastDFS 存储器开机启动：

```
# vi /etc/rc.d/rc.local
```

添加：

```
## FastDFS Storage
/etc/init.d/fdfs_storaged start
```

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
```

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 automake autoconf libtool pcre pcre-devel zlib zlib-devel openssl 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.log  info;
```

```
#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

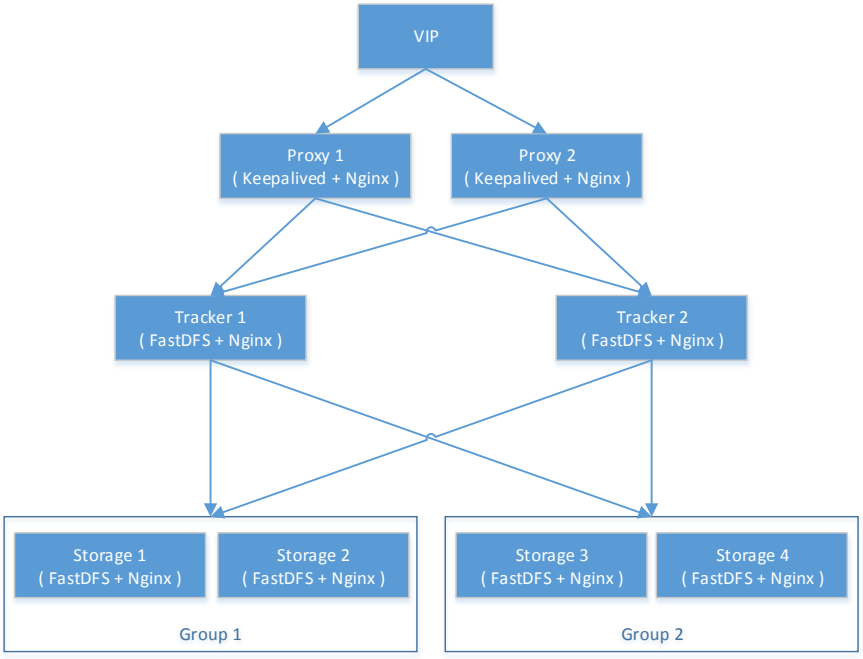
(2)通过 Tracker2 中的 Nginx 来访问

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



FastDFS集群

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。

最新课程列表、课程下载链接、课程知识互动，请关注“龙果”微信公众号！

内容持续完善中！