我们先从单机环境搭建，后续再搭建集群模式。

1 搭建环境

> 服务器：阿里云
> 操作系统： Ubuntu 16.04 64位
> IP： 114.215.169.66

本机器已经开启了root权限，所以在操作的时候不会使用sudo去获取一些执行权限。

2 gcc、g++编译器

ubuntu平台在线安装指令：

```
apt-get install gcc
apt-get install g++
apt-get install build-essential
apt-get install libtool
```

3 Nginx的安装和配置

如果已经安装过nginx可以不用重新安装，**但需要注意配置文件。**

3.1 使用Nginx的必备软件

PCRE库

PCRE库源码包下载地址: https://sourceforge.net/projects/pcre/files/pcre/8.44/pcre-8.44.tar.gz
编译和安装PCRE库相关命令：

```
wget https://sourceforge.net/projects/pcre/files/pcre/8.44/pcre-8.44.tar.gz
tar -zxvf pcre-8.44.tar.gz
cd pcre-8.44/
./configure
make
make install
```

zlib库

zlib 源码包下载地址：https://nchc.dl.sourceforge.net/project/libpng/zlib/1.2.11/zlib-1.2.11.tar.gz
编译和安装zlib库相关命令：

```
wget https://nchc.dl.sourceforge.net/project/libpng/zlib/1.2.11/zlib-
1.2.11.tar.gz
tar -zxvf zlib-1.2.11.tar.gz
cd zlib-1.2.11/
./configure
make
make install
```

OpenSSL开发库

OpenSSL源码包下载地址：https://www.openssl.org/source/openssl-1.1.1g.tar.gz
编译和安装OpenSSL开发库相关命令：

```
wget https://www.openssl.org/source/openssl-1.1.1g.tar.gz
tar -zxvf openssl-1.1.1g.tar.gz
cd openssl-1.1.1g/
./config
make
make install
```

3.2 Nginx的安装和启动

编译安装Nginx

Nginx源码包下载地址：http://nginx.org/download/nginx-1.16.1.tar.gz
编译和安装Nginx相关命令：

```
wget http://nginx.org/download/nginx-1.16.1.tar.gz
tar -zxvf nginx-1.16.1.tar.gz
nginx-1.16.1/
./configure --prefix=/usr/local/nginx --with-http_stub_status_module --with-
http_ssl_module --with-http_realip_module --with-http_v2_module --with-
openssl=../openssl-1.1.1g
make
make install
```

Nginx的启动和关闭

默认情况下，Nginx被安装在目录/usr/local/nginx中：

```
cd usr/local/nginx
ls
显示：conf  html  logs  sbin
```

其中，其中Nginx的配置文件存放于conf/nginx.conf，bin文件是位于sbin目录下的nginx文件。

1)默认方式启动Nginx服务器

/usr/local/nginx/sbin/nginx （需要sudo权限）

这时，会自动读取配置文件：/usr/local/nginx/conf/nginx.conf

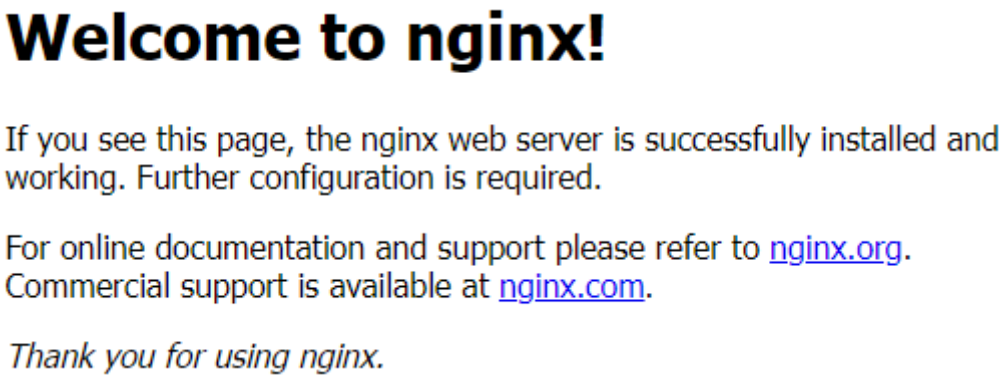
2)查看nginx进程

$ ps -ef|grep nginx

root 47583 1 0 20:15 ? 00:00:00 nginx: master process /usr/local/nginx/sbin/nginx

nobody 47584 47583 0 20:15 ? 00:00:00 nginx: worker process

打开浏览器访问此机器的IP，如果浏览器出现 Welcome to nginx! 则表示 Nginx 已经安装并运行成功：



3)指定配置文件启动服务器

```
# /usr/local/nginx/sbin/nginx -c /usr/local/nginx/conf/nginx.conf
```

4)测试配置信息

```
# /usr/local/nginx/sbin/nginx -t
```

提示：

nginx: the configuration file /usr/local/nginx/conf/nginx.conf syntax is ok

nginx: configuration file /usr/local/nginx/conf/nginx.conf test is successful

4 FastDFS 安装与配置

版本：

- libfastcommon 1.0.50
- fastdfs 6.0.7

4.1 安装 libfastcommon

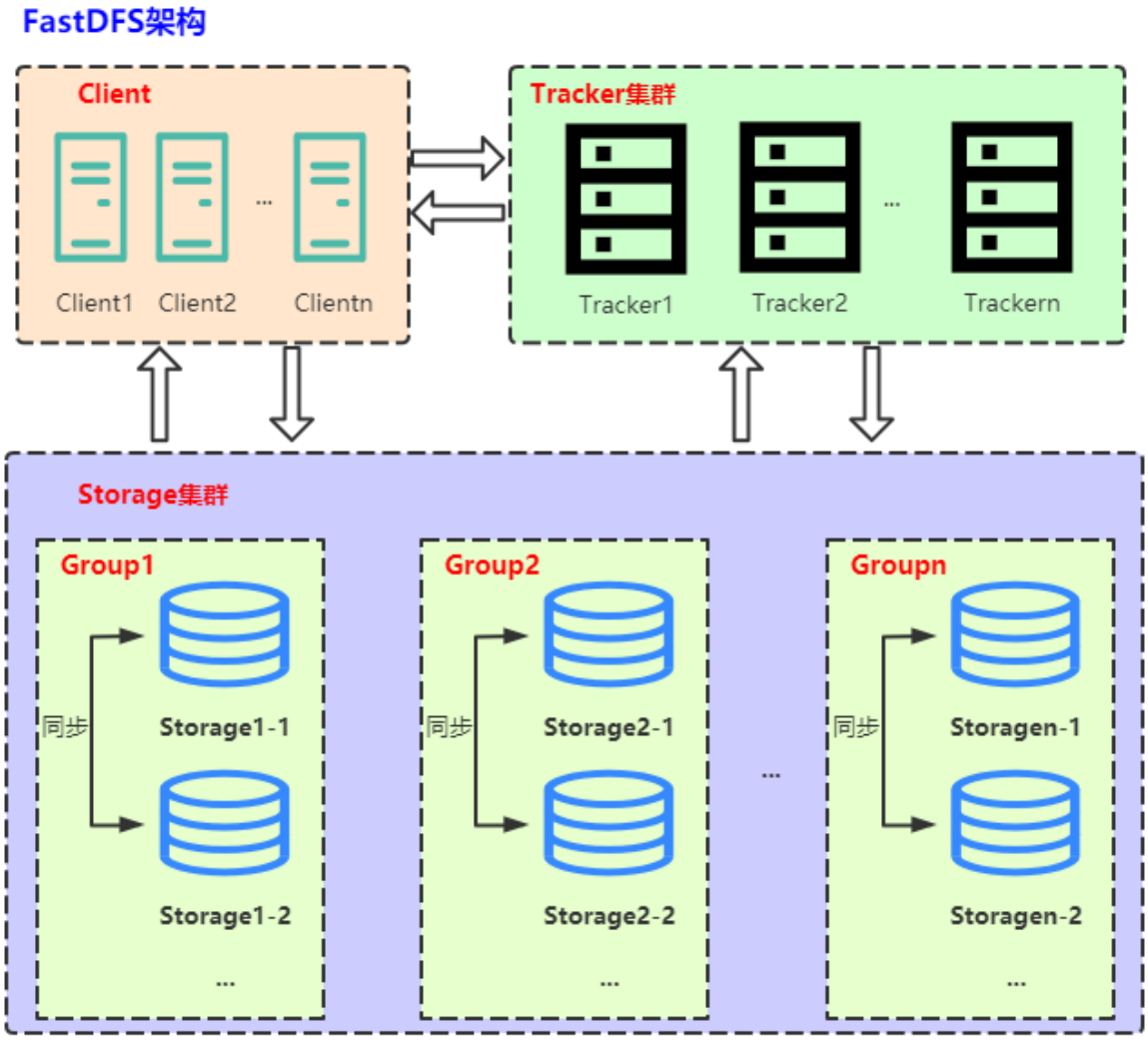
libfastcommon 的git下载地址：https://github.com/happyfish100/libfastcommon

安装FastDFS前，需要先安装libfastcommon

```
git clone https://gitee.com/fastdfs100/libfastcommon.git
cd libfastcommon
git checkout V1.0.50
./make.sh
./make.sh install
```

4.2 安装 FastDFS

```
git clone https://gitee.com/fastdfs100/fastdfs.git
cd fastdfs
git checkout V6.07
./make.sh
./make.sh install
```



FastDFS服务有三个角色:跟踪服务器(tracker server)、存储服务器(storage server)和客户端(client)

4.2.1 Tracker server

Tracker是FastDFS的协调者，负责管理所有的storage server和group，每个storage在启动后会连接Tracker，告知自己所属的group等信息，并保持周期性的心跳，tracker根据storage的心跳信息，建立group==>[storage server list]的映射表。

Tracker需要管理的元信息很少，会全部存储在内存中；另外tracker上的元信息都是由storage汇报的信息生成的，本身不需要持久化任何数据，这样使得tracker非常容易扩展，直接增加tracker机器即可扩展为tracker cluster来服务，cluster里每个tracker之间是完全对等的，所有的tracker都接受stroage的心跳信息，生成元数据信息来提供读写服务。

4.2.2 Storage server

Storage server（后简称storage）以组（卷，group或volume）为单位组织，一个group内包含多台storage机器，数据互为备份，存储空间以group内容量最小的storage为准，所以建议group内的多个storage尽量配置相同，以免造成存储空间的浪费。

以group为单位组织存储能方便的进行应用隔离、负载均衡、副本数定制（group内storage server数量即为该group的副本数），比如将不同应用数据存到不同的group就能隔离应用数据，同时还可根据应用的访问特性来将应用分配到不同的group来做负载均衡；缺点是group的容量受单机存储容量的限制，同时当group内有机器坏掉时，数据恢复只能依赖group内地其他机器，使得恢复时间会很长。

group内每个storage的存储依赖于本地文件系统，storage可配置多个数据存储目录，比如有10块磁盘，分别挂载在/data/disk1-/data/disk10，则可将这10个目录都配置为storage的数据存储目录。

storage接受到写文件请求时，会根据配置好的规则，选择其中一个存储目录来存储文件。为了避免单个目录下的文件数太多，在storage第一次启动时，会在每个数据存储目录里创建2级子目录，每级256个，总共65536个文件，新写的文件会以hash的方式被路由到其中某个子目录下，然后将文件数据直接作为一个本地文件存储到该目录中。

4.2.3 Client

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

4.3 配置 Tracker

```
# 创建 Tracker 的存储日志和数据的根目录
mkdir -p /home/fastdfs/tracker
cd /etc/fdfs
cp tracker.conf.sample tracker.conf
# 配置 tracker.conf
vim tracker.conf
```

在这里，tracker.conf 只是修改一下 Tracker 存储日志和数据的路径

```
# 启用配置文件（默认为 false，表示启用配置文件）
disabled=false
# Tracker 服务端口（默认为 22122）
port=22122
# 存储日志和数据的根目录
base_path=/home/fastdfs/tracker
```

主要修改**base_path**路径。

4.4 配置 Storage

```
# 创建 Storage 的存储日志和数据的根目录
mkdir -p /home/fastdfs/storage
cd /etc/fdfs
cp storage.conf.sample storage.conf
# 配置 storage.conf
vim storage.conf
```

在这里，storage.conf 只是修改一下 storage 存储日志和数据的路径

```
# 启用配置文件（默认为 false，表示启用配置文件）
disabled=false
# Storage 服务端口（默认为 23000）
port=23000
# 数据和日志文件存储根目录
base_path=/home/fastdfs/storage
# 存储路径，访问时路径为 M00
# store_path1 则为 M01，以此递增到 M99（如果配置了多个存储目录的话，这里只指定 1 个）
# store_path0 M00
store_path0=/home/fastdfs/storage
# Tracker 服务器 IP 地址和端口，单机搭建时也不要写 127.0.0.1
# tracker_server 可以多次出现，如果有多个，则配置多个
tracker_server=114.215.169.66:22122
# 设置 HTTP 访问文件的端口。这个配置已经不用配置了，配置了也没什么用
# 这也是为何 Storage 服务器需要 Nginx 来提供 HTTP 访问的原因
http.server_port=8888
```

主要修改：**base_path、store_path0、tracker_server**

Tracker 服务器 IP 地址和端口，单机搭建时也不要写 127.0.0.1

tracker_server 可以多次出现，如果有多个，则配置多个**（切记不要老师的IP地址，写自己的IP地址）**

tracker_server=114.215.169.66:22122

4.5 启动 Tracker 和 Storage 服务

4.5.1 启动服务

```
# 启动 Tracker 服务
# 其它操作则把 start 改为 stop、restart、reload、status 即可。Storage 服务相同
/etc/init.d/fdfs_trackerd start
# 启动 Storage 服务
/etc/init.d/fdfs_storaged start
```

可以通过对应服务的端口查看服务是否正常启动（**如果是云服务器，注意要开放对应端口**）

```
root@iZbp1h2l856zgoegc8rvnhZ:/home/fastdfs# lsof -i:22122
COMMAND     PID USER   FD   TYPE  DEVICE SIZE/OFF NODE NAME
fdfs_trac 26191 root    5u  IPv4 5570489      0t0  TCP *:22122 (LISTEN)


root@iZbp1h2l856zgoegc8rvnhZ:/etc/fdfs# lsof -i:23000
COMMAND     PID USER   FD   TYPE  DEVICE SIZE/OFF NODE NAME
fdfs_stor 26338 root    5u  IPv4 5574248      0t0  TCP *:23000 (LISTEN)
```

可以通过 fdfs_monitor 查看集群的情况

```
# 查看 Storage 是否已经注册到 Tracker 服务器中
# 当查看到 ip_addr = 114.215.169.66: (localhost.localdomain)  ACTIVE
# ACTIVE 表示成功
/usr/bin/fdfs_monitor /etc/fdfs/storage.conf
```

```
root@iZbp1h2l856zgoegc8rvnhZ:~# /usr/bin/fdfs_monitor /etc/fdfs/storage.conf
[2022-10-28 15:23:50] DEBUG - base_path=/home/fastdfs/storage, connect_timeout=5, net
use_storage_id=0, storage server id count: 0

server_count=1, server_index=0
[2022-10-28 15:23:50] DEBUG - file: connection_pool.c, line: 296, server 172.19.24.11

tracker server is 172.19.24.119:22122

group count: 1

Group 1:
group name = group1
disk total space = 40,187 MB
disk free space = 7,986 MB
trunk free space = 0 MB
storage server count = 1
active server count = 1
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 = 172.19.24.119
                ip_addr = 172.19.24.119  ACTIVE
                http domain =
```

PS:
可以去查看/etc/init.d/fdfs_trackerd文件， fdfs_trackerd的实际执行程序为：/usr/bin/fdfs_trackerd，配置文件为：/etc/fdfs/tracker.conf。
这样后续我们也可以单台机器通过修改端口的方式去启动多个tracker、storage。

4.5.2 查看日志的方式

tracker:

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

storage:

- tail -f /home/fastdfs/storage/logs/storaged.log

4.5.3 更多服务命令

tracker:

- 启动：/etc/init.d/fdfs_trackerd start
- 停止：/etc/init.d/fdfs_trackerd stop
- 重启： /etc/init.d/fdfs_trackerd restart

storage:

- 启动：/etc/init.d/fdfs_storagedstart
- 停止：/etc/init.d/fdfs_storagedstop
- 重启： /etc/init.d/fdfs_storagedrestart

4.6 测试上传文件

```
# 修改 Tracker 服务器客户端配置文件
mkdir -p /home/fastdfs/client
cp /etc/fdfs/client.conf.sample /etc/fdfs/client.conf
vim /etc/fdfs/client.conf
```

client.conf 中修改 base_path 和 Tracker 服务器的 IP 地址与端口号即可

```
# 存储日志文件的基本路径
base_path=/home/fastdfs/client
# Tracker 服务器 IP 地址与端口号
tracker_server=114.215.169.66:22122
```

主要修改base_path和tracker_server

格式： /usr/bin/fdfs_upload_file 配置文件 要上传的文件

```
# 存储到 FastDFS 服务器中
/etc/fdfs# /usr/bin/fdfs_upload_file /etc/fdfs/client.conf ./client.conf
```

当返回文件 ID 号，如 group1/M00/00/00/ctepQmIWJTCAcldiAAAHuj79dAY04.conf
则表示上传成功

如果报错：tracker_query_storage fail, error no: 2, error info: No such file or directory，一般都是因为路径没有设置对。可以

```
tail -f /home/fastdfs/storage/logs/storaged.log
```

查看日志。

4.7 下载文件测试

查看文件路径

```
root@iZbp1d83xkvoja33dm7ki2Z:/home/fastdfs/storage/data/00/00# pwd

/home/fastdfs/storage/data/00/00
```

在client所在的机器完成下载、删除测试
下载：

```
下载文件
root@iZbp1h2l856zgoegc8rvnhZ:~# fdfs_download_file /etc/fdfs/client.conf
group1/M00/00/00/ctepQmIWJTCAcldiAAAHuj79dAY04.conf

查看当前目录
root@iZbp1h2l856zgoegc8rvnhZ:~# l
ctepQmIWJTCAcldiAAAHuj79dAY04.conf
```

删除：

```
root@iZbp1h2l856zgoegc8rvnhZ:~# fdfs_delete_file /etc/fdfs/client.conf
group1/M00/00/00/ctepQmIWJTCAcldiAAAHuj79dAY04.txt
```

查看文件是否已经被删除
可以进去/home/fastdfs/storage/data/00/00路径查看刚才的文件是否被删除。

以上则完成了 FastDFS 的安装与配置，可以使用 api 来完成文件的上传、同步和下载。

当然，接下来我们还会安装 Nginx。目的如下：

- Storage 安装 Nginx，为了提供 http 的访问和下载服务，同时解决 group 中 Storage 服务器的同步延迟问题
- Tracker 安装 Nginx，主要是为了提供 http 访问的反向代理、负载均衡以及缓存服务

5 nginx-fastdfs、upload-module安装

一定要记得选择合适的版本(根据文档提示即可)，主要有git checkout相关的操作。

5.0 备份原来的nginx

为避免端口直接的冲突，建议备份原有的nginx服务。
比如进入到/usr/local/目录，

```
mv nginx bk-nginx-20221031
```

5.1 模块包的安装

注意：全部安装条件在确保之前的FastDFS 的tracker、storage和client可以正常使用。
模块包源码包本地下载路径：https://github.com/happyfish100/fastdfs-nginx-module

5.1.1 下载fastdfs-nginx-module

```
git clone https://github.com/happyfish100/fastdfs-nginx-module.git
cd fastdfs-nginx-module
git checkout V1.22
```

cd fastdfs-nginx-module/ 会发现里面有个INSTALL 和 src目录，这个不需要make而是需要重新编译一下storage的Nginx模块。

查看fastdfs-nginx-module模块src路径

```
root@iZbp1h2l856zgoegc8rvnhZ:~/tuchuang/fastdfs-nginx-module# pwd
/root/tuchuang/fastdfs-nginx-module
```

5.1.2 下载nginx-upload-module

```
下载原始的版本（原始版本有bug）：
wget http://www.grid.net.ru/nginx/download/nginx_upload_module-2.2.0.tar.gz

解压：
tar -zxvf nginx_upload_module-2.2.0.tar.gz

下载修改过的nginx upload模块，替换部分原始的文件
git clone https://github.com/winshining/nginx-upload-module.git

cd nginx-upload-module
拷贝当前该目录的内容到nginx_upload_module-2.2.0
~/tuchuang/nginx-upload-module# cp -arf * ../nginx_upload_module-2.2.0

cd ../nginx_upload_module-2.2.0
~/tuchuang/nginx_upload_module-2.2.0# pwd
/root/tuchuang/nginx_upload_module-2.2.0
获取到路径
```

5.1.3 安装和编译Nginx 并添加FastDFS模块

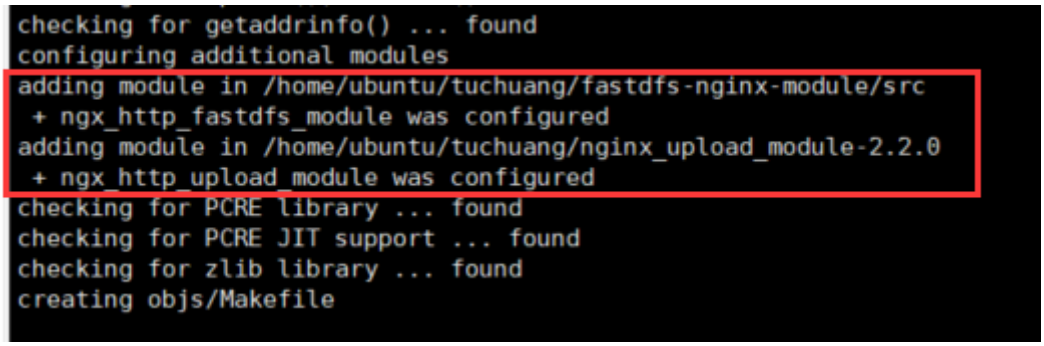
```
#进入到nginx源码目录
cd nginx-1.16.1/              (本人路径：/root/0voice/guanwang/nginx-1.16.1)

配置添加模块：
./configure --prefix=/usr/local/nginx --with-http_stub_status_module --with-
http_ssl_module --with-http_realip_module --with-http_v2_module --with-
openssl=../openssl-1.1.1g  --add-module=/root/tuchuang/fastdfs-nginx-module/src
--add-module=/root/tuchuang/nginx_upload_module-2.2.0
 --add-module=/root/tuchuang/fastdfs-nginx-module/src
```

/configure --add-module=/root/tuchuang/fastdfs-nginx-module/src (自己的路径)

其中/root/tuchuang/fastdfs-nginx-module/src 是刚才下载的的fastdfs_nginx_module模块的绝对路径，就是在编译Nginx时候，连同这个模块一起编译。

configure时, 出现以下信息表示添加成功。

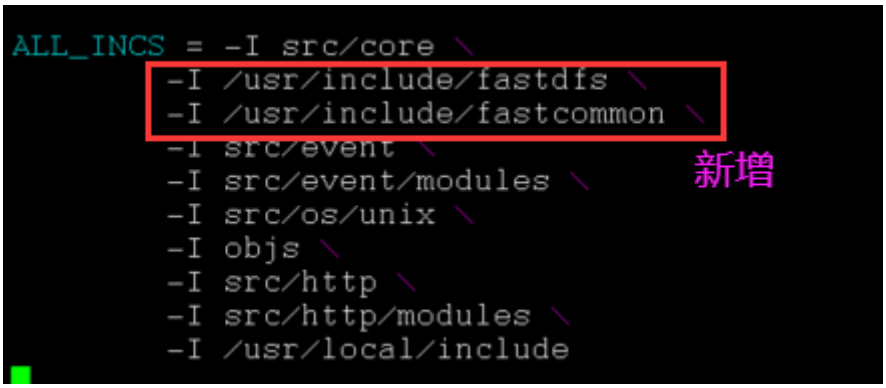


5.1.4 给 nginx 目录下的 objs/Makefile 文件中增加头文件目录

```
vim objs/Makefile
```

添加

```
ALL_INCS = -I src/core \
    -I /usr/include/fastdfs \
    -I /usr/include/fastcommon \
    -I src/event \
    -I src/event/modules \
```



特别需要注意加入两行后 -I 和 \ 的颜色要和原来一致，否则报错：Makefile:8: recipe for target 'build' failedmake: *** [build] Error 2

5.1.5 重新编译及安装nginx

```
make
make install
```

安装成功后：通过nginx -V 测试编译是否正常

$ sudo /usr/local/nginx/sbin/nginx -V

nginx version: nginx/1.16.1
built by gcc 7.5.0 (Ubuntu 7.5.0-3ubuntu1~18.04)
built with OpenSSL 1.1.1g 21 Apr 2020
TLS SNI support enabled
configure arguments: --prefix=/usr/local/nginx --with-http_stub_status_module --with-http_ssl_module --with-http_realip_module --with-http_v2_module --with-openssl=../openssl-1.1.1g --add-c --add-module=/home/ubuntu/tuchuang/nginx_upload_module-2.2.0

5.2 fastdfs-nginx-module 的配置

5.2.1 拷贝配置文件

1. 拷贝fastdfs-nginx-module配置文件
切换到fastdfs-nginx-module/src路径

```
root@iZbp1h2l856zgoegc8rvnhZ:~/tuchuang/fastdfs-nginx-module/src# pwd
/root/tuchuang/fastdfs-nginx-module/src

root@iZbp1h2l856zgoegc8rvnhZ:~/tuchuang/fastdfs-nginx-module/src# l
common.c  common.h  config  mod_fastdfs.conf  ngx_http_fastdfs_module.c
```

将fastdfs-nginx-module/src/mod_fastdfs.conf 拷贝到/etc/fdfs/下：

```
cp mod_fastdfs.conf /etc/fdfs/
```

2. **拷贝fastdfs/conf 配置文件**

fastdfs的部分配置文件到 /etc/fdfs

```
cd /root/tuchuang/fastdfs         # 为fastdfs源码路径
cp conf/http.conf /etc/fdfs/
cp conf/mime.types /etc/fdfs/
```

5.2.2 修改配置文件

创建目录（一定要记得）
mkdir -p /home/fastdfs/mod_fastdfs

修改vim /etc/fdfs/mod_fastdfs.conf

base_path =/home/fastdfs/mod_fastdfs #保存日志目录
tracker_server =114.215.169.66**:22122** #tracker服务器的IP地址以及端口号, 确保跟storage.conf一致即可

```
base_path =/home/fastdfs/mod_fastdfs
# Tracker 服务器IP和端口修改
tracker_server=114.215.169.66:22122
# url 中是否包含 group 名称，改为 true，包含 group
url_have_group_name = true
# store_path0的路径必须和storage.conf的配置一致
store_path0=/home/fastdfs/storage
# 其它的一般默认即可，例如
group_name=group1
storage_server_port=23000
store_path_count=1
```

主要修改base_path 、tracker_server、url_have_group_name、store_path0。

5.2.3 配置nginx

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

配置为支持 group0-group9，以及 M00-M99，以便于以后扩容

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

具体位置：

```
server {
    listen       80;
    server_name  localhost;

    #charset koi8-r;

    #access_log  logs/host.access.log  main;

    location / {
        root   html;
        index  index.html index.htm;
    }
    location ~/group([0-9])/M([0-9])([0-9]) {
            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
    #
    #location ~ \.php$ {
    #    proxy_pass   http://127.0.0.1;
    #}
```

5.2.4 重启nginx

配置完fastdfs-nginx-module后需要重启nginx

```
停止：/usr/local/nginx/sbin/nginx -s stop
启动：/usr/local/nginx/sbin/nginx
```

如果出现如下错误。需要先停止nginx再启动nginx

```
/usr/local/nginx/sbin/nginx -c /usr/local/nginx/conf/nginx.conf
```

nginx: [emerg] unknown directive "ngx_fastdfs_module" in /usr/local/nginx/conf/nginx.conf:92

查看nginx是否启动正常：
root@iZbp1h2l856zgoegc8rvnhZ:~# ps -ef | grep nginx
root   29763   1 0 21:06 ?   00:00:00 nginx: master process /usr/local/nginx/sbin/nginx

如果木有work进程说明启动是异常的，一般有以下原因：

1. base_path对应的路径木有创建
2. tracker_server 配置错误
3. store_path0 配置错误

5.2.5 测试环境

服务器中测试上传。

```
root@iZbp1d83xkvoja33dm7ki2Z:~# touch 0voice.txt
root@iZbp1d83xkvoja33dm7ki2Z:~# echo "You are not strong, no one brave for you"
>  0voice.txt
root@iZbp1d83xkvoja33dm7ki2Z:~# fdfs_upload_file /etc/fdfs/client.conf
0voice.txt

得到:
group1/M00/00/00/ctepQmIWLzWAHzHrAAAAKTIQHvk745.txt
```

拿到存储位置：group1/M00/00/00/ctepQmIWLzWAHzHrAAAAKTIQHvk745.txt
如果是group1开头，查看group1中storage文件是否存在
[

](http://120.27.131.197:80/group1/M00/00/00/eBuDxWCb2qmAQ89yAAAAKeR1ilo162.txt)
浏览器输入：

```
http://114.215.169.66:80/group1/M00/00/00/ctepQmIWLzWAHzHrAAAAKTIQHvk745.txt
```

浏览器能显示对应的文本信息。

参考](https://blog.csdn.net/hochenchong/article/details/81008229)
centos fastdfs 多服务器 多硬盘 多组 配置详解 [https://www.cnblogs.com/hujihon/p/5578851.html

](https://www.cnblogs.com/hujihon/p/5578851.html)