**《FastDFS分布式文件服务器高可用集群部署文档》**

**大数据分析部 苏航**

**2019年6月25日**

目录

[分布式集群搭建结构 2](#_Toc14771778)

[集群规划清单 3](#_Toc14771779)

[安装清单 3](#_Toc14771780)

[集群规划 3](#_Toc14771781)

[云服务器配置 3](#_Toc14771782)

[服务器规划 3](#_Toc14771783)

[整体架构图 4](#_Toc14771784)

[集群安装 4](#_Toc14771785)

[安装LibFastCommon/FastDFS模块 5](#_Toc14771786)

[安装Tracker并实现节点信息配置 6](#_Toc14771787)

[安装Storage模块并实现配置 9](#_Toc14771788)

[建立存储目录 9](#_Toc14771789)

[安装fastdfs-nginx-module、Nginx模块 10](#_Toc14771790)

[文件上传测试 14](#_Toc14771791)

[配置反向代理：Tracker安装Nginx、 ngx\_cache\_purge 模块 15](#_Toc14771792)

[安装编译 Nginx 所需的依赖包 15](#_Toc14771793)

[构建Keepalived+Nginx 实现虚拟IP的代理 21](#_Toc14771794)

[安装Nginx 21](#_Toc14771795)

[Nginx快捷启动和开机启动配置 25](#_Toc14771796)

[KeepAlived安装和配置 31](#_Toc14771797)

# 分布式集群搭建结构

* 双Tracker
* 2组Group
* 轮询存储策略
* Keepalived+Nginx高可用
* Nginx缓存
* 4个存储节点

# 集群规划清单

## 安装清单

| **软件名称** | **版本** | **存放名称** |
| --- | --- | --- |
| FastDFS | 5.11 | fastdfs-5.11.zip |
| FastDFS-Nginx-module | 无 | fastdfs-nginx-module-master.zip |
| LibFastCommon | 1.0.36 | libfastcommon-1.0.36.zip |
| nginx | 1.10.3 | nginx-1.10.3.tar.gz |
| nginx-pure-cache | 2.3 | ngx\_cache\_purge-2.3.tar.gz |

## 集群规划

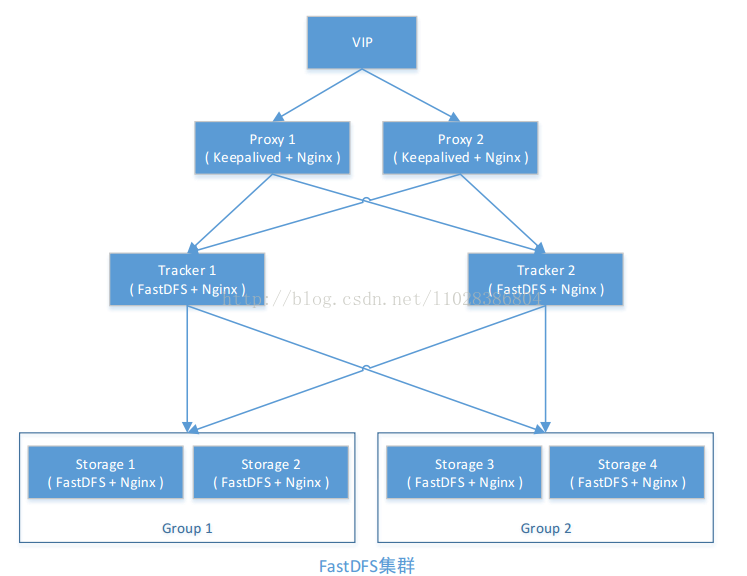
### 云服务器配置



### 服务器规划

|  |  |  |
| --- | --- | --- |
| **虚拟机** | **IP（cloud）** | **说明** |
| VIP（云桌面可测试，只分配IP，不分配资源） | 172.16.100.72 | 虚拟漂移IP |
| webapi接口服务器 | 172.16.100.71 | jdk+tomcat+mysql |
| Keepalived+Nginx1[Master] | 172.16.100.68 | Nginx Server 01 |
| Keeepalived+Nginx[Backup] | 172.16.100.69 | Nginx Server 02 |
|  |  |  |
| Tracker01 | 172.16.100.62 | Tracker01服务器 |
| Tracker02 | 172.16.100.63 | Tracker02服务器 |
| Storage01 | 172.16.100.64 | Storage01服务器【group1】 |
| Storage02 | 172.16.100.65 | Storage02服务器【group1】 |
| Storage03 | 172.16.100.66 | Storage03服务器【group2】 |
| Storage04 | 172.16.100.67 | Storage04服务器【group2】 |

### 整体架构图

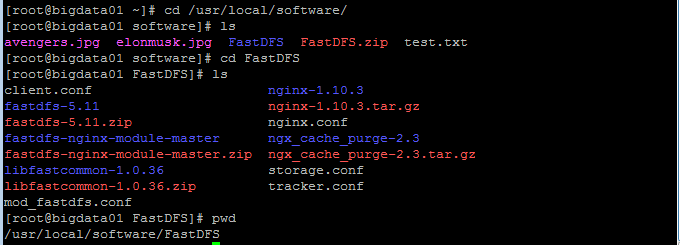


# 集群安装

**以下操作均在关闭所有节点防火墙进行的，请根据个人情况开启相关端口或关闭防火墙**

## 安装LibFastCommon/FastDFS模块

**执行节点 Tracker01、Tracker02、Storage01、Storage03、Storage04**



yum -y install unzip

unzip FastDFS.zip

cd /usr/local/software/FastDFS

unzip libfastcommon-1.0.36.zip

yum -y install gcc-c++

unzip fastdfs-5.11.zip

yum install perl

cd /usr/local/software/FastDFS/libfastcommon-1.0.36 && ./make.sh && ./make.sh install

cd /usr/local/software/FastDFS/fastdfs-5.11 && ./make.sh && ./make.sh install

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

cd /etc/fdfs

cp client.conf.sample client.conf

cp storage.conf.sample storage.conf

cp tracker.conf.sample tracker.conf

## 安装Tracker并实现节点信息配置

**执行节点 Tracker01、Tracker02**

mkdir /opt/fastdfs\_tracker

配置 /etc/fdfs目录下tracker.conf

1.disabled=false

2.port=22122 #默认端口号

3.base\_path=/opt/fastdfs\_tracker #我刚刚创建的目录

4.http.server\_port=8080 #默认端口是8080

5.store\_lookup=0 #采用轮询策略进行存储，0 轮询 1：始终定向到某个group 2：负载进行存储文件

完整tracker.conf 文件信息如下：

disabled=false

bind\_addr=

port=22122

connect\_timeout=30

network\_timeout=60

base\_path=/opt/fastdfs\_tracker

max\_connections=512

accept\_threads=1

work\_threads=4

min\_buff\_size = 8KB

max\_buff\_size = 128KB

store\_lookup=0

**store\_group=**

store\_server=0

store\_path=0

download\_server=0

reserved\_storage\_space = 10%

log\_level=info

run\_by\_group=

run\_by\_user=

allow\_hosts=\*

sync\_log\_buff\_interval = 10

check\_active\_interval = 120

thread\_stack\_size = 64KB

storage\_ip\_changed\_auto\_adjust = true

storage\_sync\_file\_max\_delay = 86400

storage\_sync\_file\_max\_time = 300

use\_trunk\_file = false

slot\_min\_size = 256

slot\_max\_size = 16MB

trunk\_file\_size = 64MB

trunk\_create\_file\_advance = false

trunk\_create\_file\_time\_base = 02:00

trunk\_create\_file\_interval = 86400

trunk\_create\_file\_space\_threshold = 20G

trunk\_init\_check\_occupying = false

trunk\_init\_reload\_from\_binlog = false

trunk\_compress\_binlog\_min\_interval = 0

use\_storage\_id = false

storage\_ids\_filename = storage\_ids.conf

id\_type\_in\_filename = ip

store\_slave\_file\_use\_link = false

rotate\_error\_log = false

error\_log\_rotate\_time=00:00

rotate\_error\_log\_size = 0

log\_file\_keep\_days = 0

use\_connection\_pool = false

connection\_pool\_max\_idle\_time = 3600

http.server\_port=8080

http.check\_alive\_interval=30

http.check\_alive\_type=tcp

http.check\_alive\_uri=/status.html

修改保存后创建软引用

ln -s /usr/bin/fdfs\_storaged /usr/local/bin

启动tracker，并加入开机启动项

service fdfs\_trackerd start

将tracker加入开机启动项

echo "service fdfs\_trackerd start" |tee -a /etc/rc.d/rc.local

## 安装Storage模块并实现配置

**执行节点 Storage01、Storage02、Storage03、Storage04**

### 建立存储目录

在/opt/存储各节点建了两个目录fastdfs\_storage\_data,fastdfs\_storage

mkdir /opt/fastdfs\_storage

mkdir /opt/fastdfs\_storage\_data

配置 /etc/fdfs/storage.conf，注意划分group1和group2

disabled=false

group\_name=**group1**

port=23000

base\_path=/opt/fastdfs\_storage

store\_path\_count=1

store\_path0=/opt/fastdfs\_storage\_data

tracker\_server=172.16.100.62:22122

tracker\_server=172.16.100.63:22122

http.server\_port=8888

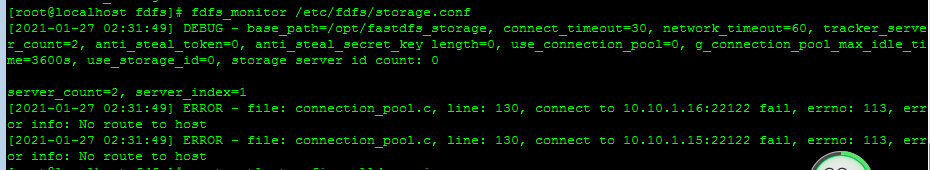
各节点配置好信息好之后，启动Storage

service fdfs\_storaged start

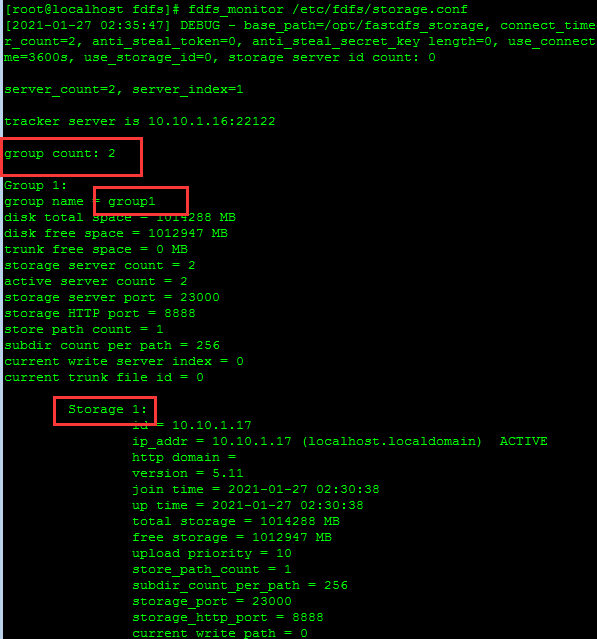
启动后查看日志情况。如果看到有2组Storage信息，则表示配置信息配置成功，并注册到Tracker中。

fdfs\_monitor /etc/fdfs/storage.conf

如果有如下Error，可能是防火墙的原因，可以开放需要的端口或者关闭防火墙。



成功下图。



查看日志启动情况

[2019-06-25 09:52:01] INFO - file: tracker\_client\_thread.c, line: 310, successfully connect to tracker server 172.16.100.62:22122, continuous fail count: 1, as a tracker client, my ip is 172.16.100.64

[2019-06-25 09:52:01] INFO - file: tracker\_client\_thread.c, line: 1263, tracker server 172.16.100.62:22122, set tracker leader: 172.16.100.62:22122

[2019-06-25 09:53:27] WARNING - file: storage\_service.c, line: 7285, client ip: 172.16.102.72, logic file: M00/00/00/rBBkQF0RdmiABqkCAACFhh9enww251.jpg not exist

[2019-06-25 09:54:56] WARNING - file: storage\_service.c, line: 7285, client ip: 172.16.102.72, logic file: M00/00/00/rBBkQF0RdmiABqkCAACFhh9enww251.jpg not exist

[2019-06-25 09:56:16] INFO - file: tracker\_client\_thread.c, line: 310, successfully connect to tracker server 172.16.100.63:22122, continuous fail count: 9, as a tracker client, my ip is 172.16.100.64

发现此时172.16.100.62作为Tracker的Leader。

设置Storage开机自启动

echo "service fdfs\_storaged start" |tee -a /etc/rc.d/rc.local

## 安装fastdfs-nginx-module、Nginx模块

**执行节点 Storage01、Storage02、Storage03、Storage04**

安装Nginx模块所需的依赖环境

yum -y install pcre pcre-devel

yum -y install zlib zlib-devel

yum -y install openssl openssl-devel

解压nginx和fastdfs-nginx-module

tar -zxvf nginx-1.10.3.tar.gz

unzip fastdfs-nginx-module-master.zip

进入Nginx解压目录进行

编译安装

./configure --prefix=/usr/local/nginx --add-module=/usr/local/software/FastDFS/fastdfs-nginx-module-master/src #解压后fastdfs-nginx-module所在的位置

安装：make

安装：make install

安装成功后，nginx会安装在/usr/local/nginx,安装后查看

ll /usr/local/nginx/

总用量 8

drwxr-xr-x. 2 root root 4096 1月 17 13:17 conf

drwxr-xr-x. 2 root root 40 1月 17 13:17 html

drwxr-xr-x. 2 root root 58 1月 17 13:49 logs

drwxr-xr-x. 2 root root 19 1月 17 13:17 sbin

安装成功后，nginx尚未运行时，nginx文件夹没有临时文件夹，例如fastcgi\_temp这些文件。

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

~~cp /usr/local/software/FastDFS/fastdfs-nginx-module-master/src/fastdfs-nginx-module/src/mod\_fastdfs.conf /etc/fdfs/~~

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

vi /etc/fdfs/mod\_fastdfs.conf

(1)第1组 Storage 的 mod\_fastdfs.conf 配置如下：

connect\_timeout=10

base\_path=/opt/fastdfs\_storage

tracker\_server=172.16.100.62:22122

tracker\_server=172.16.100.63:22122

storage\_server\_port=23000

group\_name=group1

url\_have\_group\_name = true

store\_path0=/opt/fastdfs\_storage\_data

group\_count = 2

[group1]

group\_name=group1

storage\_server\_port=23000

store\_path\_count=1

store\_path0=/opt/fastdfs\_storage\_data

[group2]

group\_name=group2

storage\_server\_port=23000

store\_path\_count=1

store\_path0=/opt/fastdfs\_storage\_data

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

group\_name=group2

复制 FastDFS 安装目录的部分配置文件到/etc/fdfs 目录

cd /usr/local/software/FastDFS/fastdfs-5.11/conf

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

创建M00至storage存储目录的符号连接

ln -s /opt/fastdfs\_storage\_data/data/ /opt/fastdfs\_storage\_data/data/M00

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

下面location部分，storage1和2写group1，storage3和4写group2，

location ~/group1/M00 {

#alias /fastdfs/storage/data;

ngx\_fastdfs\_module;

}

其他修改项如下：

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 ~/group1/M00 {

#alias /fastdfs/storage/data;

ngx\_fastdfs\_module;

}

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

location = /50x.html {

root html;

}

}

}

重新启动各节点的Nginx服务

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

设置nginx开机启动

vi /etc/rc.d/rc.local，添加如下语句：

/usr/local/nginx/sbin/nginx

## 文件上传测试

**执行节点Tracker01、Tracker02**

修改 Tracker 服务器中的客户端配置文件

# vi /etc/fdfs/client.conf

# the base path to store log files

base\_path=/opt/fastdfs\_tracker

# tracker\_server can ocur more than once, and tracker\_server format is

# "host:port", host can be hostname or ip address

tracker\_server=172.16.100.62:22122

tracker\_server=172.16.100.63:22122

执行如下文件上传命令

fdfs\_upload\_file /etc/fdfs/client.conf P71022-205803.jpg

group2/M00/00/00/wKgrSlpjC3aAARrXAAvUQrxXbkA048.jpg

进行2次上传后，会发现文件被均衡分到2个group。

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf taishan.jpg

group1/M00/00/00/CgoBEWARMh-Ad0I7AACD0Mwtxlc827.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf taishan2.jpg

group1/M00/00/00/CgoBEmARMxKAcVnKAApk9bcNBno970.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf taishan3.jpg

group2/M00/00/00/CgoBE2ARM2iAFdRCAAFHP9Q2SwI692.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf taishan4.jpg

group2/M00/00/00/CgoBFGARM6SAYKbJAA9iAB1UMnA682.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf taishan.jpg

group1/M00/00/00/CgoBEWASObmAbzQGAACD0Mwtxlc162.jpg

[root@localhost conf]# fdfs\_monitor /etc/fdfs/storage.conf

[2021-01-27 04:34:49] DEBUG - base\_path=/opt/fastdfs\_storage, connect\_timeout=30, network\_timeout=60, tracker\_server\_count=2, anti\_steal\_token=0, anti\_steal\_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 10.10.1.15:22122

group count: 2

Group 1:

group name = group1

disk total space = 1014288 MB

disk free space = 1012907 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 = 10.10.1.17

ip\_addr = 10.10.1.17 (localhost.localdomain) ACTIVE

http domain =

version = 5.11

join time = 2021-01-27 02:30:38

up time = 2021-01-27 02:30:38

total storage = 1014288 MB

free storage = 1012907 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 =

if\_trunk\_server = 0

connection.alloc\_count = 256

connection.current\_count = 1

connection.max\_count = 2

total\_upload\_count = 1

success\_upload\_count = 1

total\_append\_count = 0

success\_append\_count = 0

total\_modify\_count = 0

success\_modify\_count = 0

total\_truncate\_count = 0

success\_truncate\_count = 0

total\_set\_meta\_count = 0

success\_set\_meta\_count = 0

total\_delete\_count = 0

success\_delete\_count = 0

total\_download\_count = 0

success\_download\_count = 0

total\_get\_meta\_count = 0

success\_get\_meta\_count = 0

total\_create\_link\_count = 0

success\_create\_link\_count = 0

total\_delete\_link\_count = 0

success\_delete\_link\_count = 0

total\_upload\_bytes = 33744

success\_upload\_bytes = 33744

total\_append\_bytes = 0

success\_append\_bytes = 0

total\_modify\_bytes = 0

success\_modify\_bytes = 0

stotal\_download\_bytes = 0

success\_download\_bytes = 0

total\_sync\_in\_bytes = 681205

success\_sync\_in\_bytes = 681205

total\_sync\_out\_bytes = 0

success\_sync\_out\_bytes = 0

total\_file\_open\_count = 2

success\_file\_open\_count = 2

total\_file\_read\_count = 0

success\_file\_read\_count = 0

total\_file\_write\_count = 4

success\_file\_write\_count = 4

last\_heart\_beat\_time = 2021-01-27 04:34:37

last\_source\_update = 2021-01-27 04:27:58

last\_sync\_update = 2021-01-27 04:32:06

last\_synced\_timestamp = 2021-01-27 04:32:02 (-1s delay)

Storage 2:

id = 10.10.1.18

ip\_addr = 10.10.1.18 ACTIVE

http domain =

version = 5.11

join time = 2021-01-27 02:30:45

up time = 2021-01-27 02:30:45

total storage = 1014288 MB

free storage = 1012907 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 = 10.10.1.17

if\_trunk\_server = 0

connection.alloc\_count = 256

connection.current\_count = 1

connection.max\_count = 2

total\_upload\_count = 1

success\_upload\_count = 1

total\_append\_count = 0

success\_append\_count = 0

total\_modify\_count = 0

success\_modify\_count = 0

total\_truncate\_count = 0

success\_truncate\_count = 0

total\_set\_meta\_count = 0

success\_set\_meta\_count = 0

total\_delete\_count = 0

success\_delete\_count = 0

total\_download\_count = 0

success\_download\_count = 0

total\_get\_meta\_count = 0

success\_get\_meta\_count = 0

total\_create\_link\_count = 0

success\_create\_link\_count = 0

total\_delete\_link\_count = 0

success\_delete\_link\_count = 0

total\_upload\_bytes = 681205

success\_upload\_bytes = 681205

total\_append\_bytes = 0

success\_append\_bytes = 0

total\_modify\_bytes = 0

success\_modify\_bytes = 0

stotal\_download\_bytes = 0

success\_download\_bytes = 0

total\_sync\_in\_bytes = 33744

success\_sync\_in\_bytes = 33744

total\_sync\_out\_bytes = 0

success\_sync\_out\_bytes = 0

total\_file\_open\_count = 2

success\_file\_open\_count = 2

total\_file\_read\_count = 0

success\_file\_read\_count = 0

total\_file\_write\_count = 4

success\_file\_write\_count = 4

last\_heart\_beat\_time = 2021-01-27 04:34:39

last\_source\_update = 2021-01-27 04:32:01

last\_sync\_update = 2021-01-27 04:28:05

last\_synced\_timestamp = 2021-01-27 04:27:59 (-1s delay)

Group 2:

group name = group2

disk total space = 1014288 MB

disk free space = 1012907 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 = 10.10.1.19

ip\_addr = 10.10.1.19 ACTIVE

http domain =

version = 5.11

join time = 2021-01-27 02:30:49

up time = 2021-01-27 02:30:49

total storage = 1014288 MB

free storage = 1012907 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 = 10.10.1.20

if\_trunk\_server = 0

connection.alloc\_count = 256

connection.current\_count = 1

connection.max\_count = 2

total\_upload\_count = 1

success\_upload\_count = 1

total\_append\_count = 0

success\_append\_count = 0

total\_modify\_count = 0

success\_modify\_count = 0

total\_truncate\_count = 0

success\_truncate\_count = 0

total\_set\_meta\_count = 0

success\_set\_meta\_count = 0

total\_delete\_count = 0

success\_delete\_count = 0

total\_download\_count = 0

success\_download\_count = 0

total\_get\_meta\_count = 0

success\_get\_meta\_count = 0

total\_create\_link\_count = 0

success\_create\_link\_count = 0

total\_delete\_link\_count = 0

success\_delete\_link\_count = 0

total\_upload\_bytes = 83775

success\_upload\_bytes = 83775

total\_append\_bytes = 0

success\_append\_bytes = 0

total\_modify\_bytes = 0

success\_modify\_bytes = 0

stotal\_download\_bytes = 0

success\_download\_bytes = 0

total\_sync\_in\_bytes = 1008128

success\_sync\_in\_bytes = 1008128

total\_sync\_out\_bytes = 0

success\_sync\_out\_bytes = 0

total\_file\_open\_count = 2

success\_file\_open\_count = 2

total\_file\_read\_count = 0

success\_file\_read\_count = 0

total\_file\_write\_count = 5

success\_file\_write\_count = 5

last\_heart\_beat\_time = 2021-01-27 04:34:38

last\_source\_update = 2021-01-27 04:33:28

last\_sync\_update = 2021-01-27 04:34:34

last\_synced\_timestamp = 2021-01-27 04:34:28 (-1s delay)

Storage 2:

id = 10.10.1.20

ip\_addr = 10.10.1.20 ACTIVE

http domain =

version = 5.11

join time = 2021-01-27 02:30:54

up time = 2021-01-27 02:30:54

total storage = 1014288 MB

free storage = 1012907 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 =

if\_trunk\_server = 0

connection.alloc\_count = 256

connection.current\_count = 1

connection.max\_count = 2

total\_upload\_count = 1

success\_upload\_count = 1

total\_append\_count = 0

success\_append\_count = 0

total\_modify\_count = 0

success\_modify\_count = 0

total\_truncate\_count = 0

success\_truncate\_count = 0

total\_set\_meta\_count = 0

success\_set\_meta\_count = 0

total\_delete\_count = 0

success\_delete\_count = 0

total\_download\_count = 0

success\_download\_count = 0

total\_get\_meta\_count = 0

success\_get\_meta\_count = 0

total\_create\_link\_count = 0

success\_create\_link\_count = 0

total\_delete\_link\_count = 0

success\_delete\_link\_count = 0

total\_upload\_bytes = 1008128

success\_upload\_bytes = 1008128

total\_append\_bytes = 0

success\_append\_bytes = 0

total\_modify\_bytes = 0

success\_modify\_bytes = 0

stotal\_download\_bytes = 0

success\_download\_bytes = 0

total\_sync\_in\_bytes = 83775

success\_sync\_in\_bytes = 83775

total\_sync\_out\_bytes = 0

success\_sync\_out\_bytes = 0

total\_file\_open\_count = 2

success\_file\_open\_count = 2

total\_file\_read\_count = 0

success\_file\_read\_count = 0

total\_file\_write\_count = 5

success\_file\_write\_count = 5

last\_heart\_beat\_time = 2021-01-27 04:34:35

last\_source\_update = 2021-01-27 04:34:27

last\_sync\_update = 2021-01-27 04:33:36

last\_synced\_timestamp = 2021-01-27 04:33:28 (0s delay)

Tracker1

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf taishan1.jpg

group1/M00/00/00/CgoBEWASfiGAYPBNAACD0Mwtxlc078.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf taishan2.jpg

group1/M00/00/00/CgoBEmASftKACkh\_AApk9bcNBno776.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf taishan1.jpg

group1/M00/00/00/CgoBEWASgPeAC3sUAACD0Mwtxlc859.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf taishan2.jpg

group1/M00/00/00/CgoBEmASgW-AXyOXAApk9bcNBno911.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf taishan3.jpg

group1/M00/00/00/CgoBEWASgXGAI5CzAAFHP9Q2SwI943.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf taishan4.jpg

group1/M00/00/00/CgoBEmASgXOAYa3ZAA9iAB1UMnA664.jpg

Tracker2

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf im1.jpg

group2/M00/00/00/CgoBE2ASgyuAYU1HAADGOVLrvjw560.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf im2.jpg

group2/M00/00/00/CgoBFGASgy6AL9vXAABlq7KSGEs550.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf im3.jpg

group2/M00/00/00/CgoBE2ASgzCAEwO0AAJPZoAZsFs997.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf im4.jpg

group2/M00/00/00/CgoBFGASgzKAWikaAACYcB2jhkk994.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf im1.jpg

group2/M00/00/00/CgoBFGAU0DeANQmnAADGOVLrvjw710.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf im2.jpg

group2/M00/00/00/CgoBE2AU0DqAZP2RAABlq7KSGEs721.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf im3.jpg

group2/M00/00/00/CgoBFGAU0DyAI6qjAAJPZoAZsFs287.jpg

[root@localhost FastDFS]# fdfs\_upload\_file /etc/fdfs/client.conf im4.jpg

group2/M00/00/00/CgoBE2AU0D-AZsKgAACYcB2jhkk750.jpg

## 配置反向代理：Tracker安装Nginx、 ngx\_cache\_purge 模块

我们需要在两个跟踪器上安装nginx（也就是62和63）以提供反向代理服务，目的是使用统一的一个IP地址对外提供服务。

### 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

解压Nginx和ngx\_cache\_pure模块

cd /usr/local/software/FastDFS

tar -xvf ngx\_cache\_purge-2.3.tar.gz

tar -xvf nginx-1.10.3.tar.gz

编译安装 Nginx（添加 ngx\_cache\_purge 模块）

./configure --prefix=/usr/local/nginx

--add-module=/usr/local/software/FastDFS/ngx\_cache\_purge-2.3

&& make && make install

配置 Nginx， 设置负载均衡以及缓存

vi /usr/local/nginx/conf/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;

}

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;

####sh

#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，分别内存大小，磁盘最大空间，缓存期限

proxy\_cache\_path /opt/fastdfs\_tracker/proxy\_cache levels=1:2

keys\_zone=http-cache:200m max\_size=1g inactive=30d;

proxy\_temp\_path /opt/fastdfs\_tracker/tmp;

#group1的服务设置

upstream fdfs\_group1 {

server 172.16.100.64:8888 weight=1 max\_fails=2 fail\_timeout=30s;

server 172.16.100.65:8888 weight=1 max\_fails=2 fail\_timeout=30s;

}

#group2的服务设置

upstream fdfs\_group2 {

server 172.16.100.66:8888 weight=1 max\_fails=2 fail\_timeout=30s;

server 172.16.100.67:8888 weight=1 max\_fails=2 fail\_timeout=30s;

}

###sh

server {

listen 8000;

server\_name localhost;

#charset koi8-r;

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

#group1的负载均衡配置

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;

#对应group1的服务设置

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;

#对应group2的服务设置

proxy\_pass http://fdfs\_group2;

expires 30d;

}

location ~/purge(/.\*) {

allow 127.0.0.1;

allow 172.16.100.0/24;

deny all;

proxy\_cache\_purge http-cache $1$is\_args$args;

}

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;

}

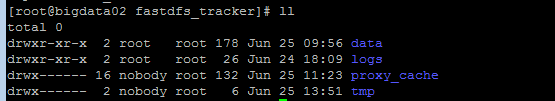
}

}

根据Nginx配置，创建对应目录下的文件夹

cd /opt/fastdfs\_tracker

mkdir proxy\_cache tmp



重启Nginx进行访问测试

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

加入开机启动 vi /etc/rc.d/rc.local

/usr/local/nginx/sbin/nginx

前面直接通过访问 Storage 节点中的 Nginx 的文件  
  
http://172.16.100.64:8888/group1/M00/00/00/wKgrSFpjC26AH1g2AAvUQrxXbkA557.jpg  
  
http:// 172.16.100.66:8888/group2/M00/00/00/wKgrSlpjC3aAARrXAAvUQrxXbkA048.jpg  
  
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:// 172.16.100.62:8000/group1/M00/00/00/wKgrSFpjC26AH1g2AAvUQrxXbkA557.jpg  
  
http:// 172.16.100.62:8000/group2/M00/00/00/wKgrSlpjC3aAARrXAAvUQrxXbkA048.jpg  
  
(2)通过 Tracker2 中的 Nginx 来访问  
  
http:// 172.16.100.63:8000/group1/M00/00/00/wKgrSFpjC26AH1g2AAvUQrxXbkA557.jpg  
  
http:// 172.16.100.63:8000/group2/M00/00/00/wKgrSlpjC3aAARrXAAvUQrxXbkA048.jpg

## 构建Keepalived+Nginx 实现虚拟IP的代理

**172.16.100.68 和 172.16.100.69**

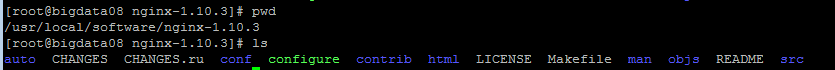
### 安装Nginx

yum -y install gcc

yum -y install pcre-devel

yum -y install zlib-devel

定位到nginx 解压文件位置，执行编译安装命令



./configure && make && make install

修改配置文件vi /usr/local/nginx/conf/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;

}

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;

upstream fastdfs\_tracker {

server 172.16.100.62:8000 weight=1 max\_fails=2 fail\_timeout=30s;

server 172.16.100.63:8000 weight=1 max\_fails=2 fail\_timeout=30s;

}

server {

listen 80;

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;

}

location /fastdfs {

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;

}

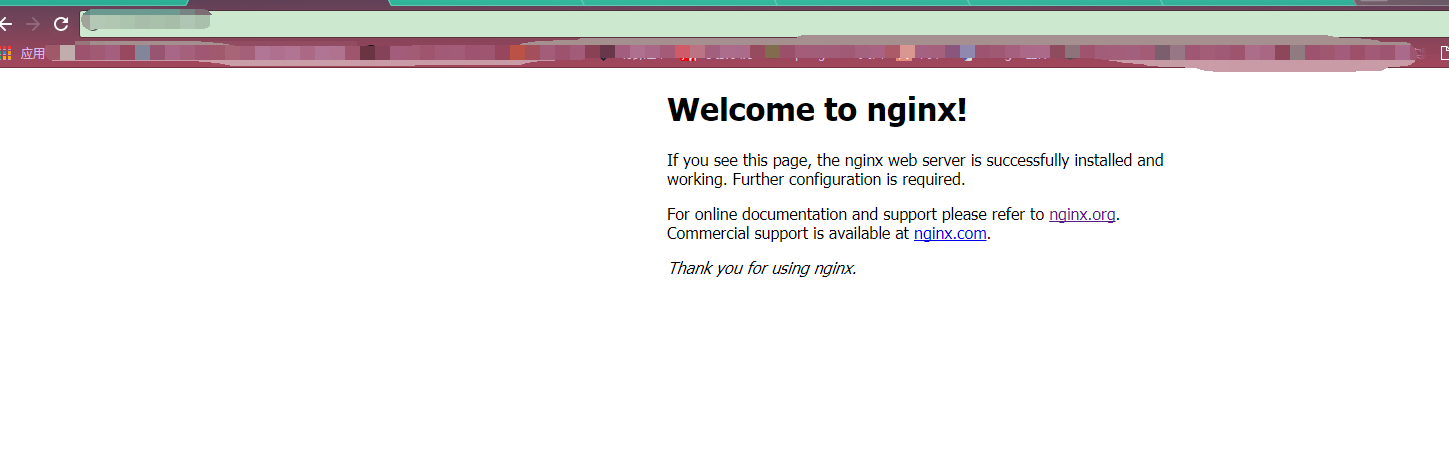
}

}

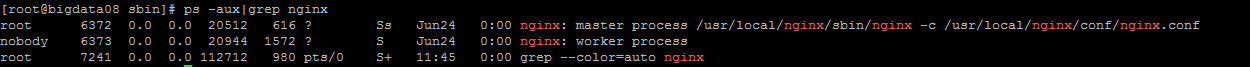
启动Nginx

/usr/local/nginx/sbin/nginx &

访问ip查看Nginx是否启动，如果不能访问，可以关闭防火墙或者开端口。



或通过进程查看Nginx启动情况



到此Nginx安装完成并启动成功。

### Nginx快捷启动和开机启动配置

编辑Nginx快捷启动脚本【**注意Nginx安装路径**，**需要根据自己的NGINX路径进行改动**】

vim /etc/rc.d/init.d/nginx

#!/bin/sh

#

# nginx - this script starts and stops the nginx daemon

#

# chkconfig: - 85 15

# description: Nginx is an HTTP(S) server, HTTP(S) reverse \

# proxy and IMAP/POP3 proxy server

# processname: nginx

# config: /etc/nginx/nginx.conf

# config: /usr/local/nginx/conf/nginx.conf

# pidfile: /usr/local/nginx/logs/nginx.pid

# Source function library.

. /etc/rc.d/init.d/functions

# Source networking configuration.

. /etc/sysconfig/network

# Check that networking is up.

[ "$NETWORKING" = "no" ] && exit 0

nginx="/usr/local/nginx/sbin/nginx"

prog=$(basename $nginx)

NGINX\_CONF\_FILE="/usr/local/nginx/conf/nginx.conf"

[ -f /etc/sysconfig/nginx ] && . /etc/sysconfig/nginx

lockfile=/var/lock/subsys/nginx

make\_dirs() {

# make required directories

user=`$nginx -V 2>&1 | grep "configure arguments:" | sed 's/[^\*]\*--user=\([^ ]\*\).\*/\1/g' -`

if [ -z "`grep $user /etc/passwd`" ]; then

useradd -M -s /bin/nologin $user

fi

options=`$nginx -V 2>&1 | grep 'configure arguments:'`

for opt in $options; do

if [ `echo $opt | grep '.\*-temp-path'` ]; then

value=`echo $opt | cut -d "=" -f 2`

if [ ! -d "$value" ]; then

# echo "creating" $value

mkdir -p $value && chown -R $user $value

fi

fi

done

}

start() {

[ -x $nginx ] || exit 5

[ -f $NGINX\_CONF\_FILE ] || exit 6

make\_dirs

echo -n $"Starting $prog: "

daemon $nginx -c $NGINX\_CONF\_FILE

retval=$?

echo

[ $retval -eq 0 ] && touch $lockfile

return $retval

}

stop() {

echo -n $"Stopping $prog: "

killproc $prog -QUIT

retval=$?

echo

[ $retval -eq 0 ] && rm -f $lockfile

return $retval

}

restart() {

#configtest || return $?

stop

sleep 1

start

}

reload() {

#configtest || return $?

echo -n $"Reloading $prog: "

killproc $nginx -HUP

RETVAL=$?

echo

}

force\_reload() {

restart

}

configtest() {

$nginx -t -c $NGINX\_CONF\_FILE

}

rh\_status() {

status $prog

}

rh\_status\_q() {

rh\_status >/dev/null 2>&1

}

case "$1" in

start)

rh\_status\_q && exit 0

$1

;;

stop)

rh\_status\_q || exit 0

$1

;;

restart|configtest)

$1

;;

reload)

rh\_status\_q || exit 7

$1

;;

force-reload)

force\_reload

;;

status)

rh\_status

;;

condrestart|try-restart)

rh\_status\_q || exit 0

;;

\*)

echo $"Usage: $0 {start|stop|status|restart|condrestart|try-restart|reload|force-reload|configtest}"

exit 2

esac

为启动脚本授权 并加入开机启动

cd /etc/rc.d/init.d/

chmod -R 777 /etc/rc.d/init.d/nginx

chkconfig nginx

./nginx start

echo 'export PATH=$PATH:/usr/local/nginx/sbin'>>/etc/profile && source /etc/profile

Nginx命令 [ service nginx (start|stop|restart) ]

service nginx start

Starting nginx (via systemctl): [ 确定 ]

### KeepAlived安装和配置

安装Keepalived依赖环境

yum install -y popt-devel

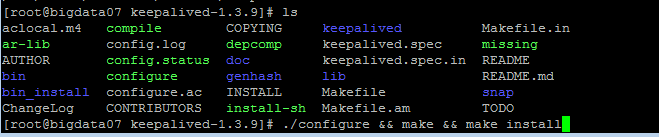
yum install -y ipvsadm

yum install -y libnl\*

yum install -y libnf\*

yum install -y openssl-devel

编译Keepalived并安装



将Keepalive 安装成系统服务

mkdir /etc/keepalived

cp /usr/local/software/FastDFS/keepalived-1.3.9/keepalived/etc/keepalived/keepalived.conf /etc/keepalived/

手动复制默认的配置文件到默认路径

mkdir /etc/sysconfig

cp /usr/local/software/FastDFS/keepalived-1.3.9/keepalived/etc/sysconfig/keepalived

/etc/sysconfig/

为keepalived 创建软链接

ln -s /usr/local/software/FastDFS/keepalived-1.3.9/bin/keepalived /usr/sbin/

设置Keepalived开机自启动

chkconfig keepalived on

注意：正在将请求转发到“systemctl enable keepalived.service”。

Created symlink from /etc/systemd/system/multi-user.target.wants/keepalived.service to /usr/lib/systemd/system/keepalived.service

启动Keepalived服务

keepalived -D -f /etc/keepalived/keepalived.conf

修改之后，重新启动Keepalive+Nginx2台主备节点。

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

在172.16.100.68 和 172.16.100.69添加如下

#### Vi /etc/keepalived/check\_nginx.sh

#!/bin/bash

#时间变量，用于记录日志

d=`date --date today +%Y%m%d\_%H:%M:%S`

#计算nginx进程数量

n=`ps -C nginx --no-heading|wc -l`

#如果进程为0，则启动nginx，并且再次检测nginx进程数量，

#如果还为0，说明nginx无法启动，此时需要关闭keepalived

if [ $n -eq "0" ]; then

/etc/rc.d/init.d/nginx start

n2=`ps -C nginx --no-heading|wc -l`

if [ $n2 -eq "0" ]; then

echo "$d nginx down,keepalived will stop" >> /var/log/check\_ng.log

systemctl stop keepalived

fi

Fi

添加完成后，为check\_nginx.sh 文件授权，便于脚本获得执行权限。

chmod -R 777 /etc/keepalived/check\_nginx.sh

在172.16.100.68

vi /etc/keepalived/keepalived.conf

! Configuration File for keepalived

vrrp\_script chk\_nginx {

script "/etc/keepalived/check\_nginx.sh" #检测nginx进程的脚本

interval 2

weight -20

}

global\_defs {

notification\_email {

#可以添加邮件提醒

}

}

vrrp\_instance VI\_1 {

state MASTER #标示状态为MASTER 备份机为BACKUP

interface eth0 #设置实例绑定的网卡(ip addr查看，需要根据个人网卡绑定)

virtual\_router\_id 51 #同一实例下virtual\_router\_id必须相同

mcast\_src\_ip 172.16.100.68

priority 250 #MASTER权重要高于BACKUP 比如BACKUP为240

advert\_int 1 #MASTER与BACKUP负载均衡器之间同步检查的时间间隔，单位是秒

nopreempt #非抢占模式

authentication { #设置认证

auth\_type PASS #主从服务器验证方式

auth\_pass 123456

}

track\_script {

check\_nginx

}

virtual\_ipaddress { #设置vip

# 172.16.100.71 #可以多个虚拟IP，换行即可,71 has a server

172.16.100.72 #可以多个虚拟IP，换行即可,72 just ocupy ip addr,has no server

}

}

在172.16.100.69

vi /etc/keepalived/keepalived.conf

! Configuration File for keepalived

vrrp\_script chk\_nginx {

script "/etc/keepalived/check\_nginx.sh" #检测nginx进程的脚本

interval 2

weight -20

}

global\_defs {

notification\_email {

#可以添加邮件提醒

}

}

vrrp\_instance VI\_1 {

state BACKUP #标示状态为MASTER 备份机为BACKUP

interface eth0 #设置实例绑定的网卡(ip addr查看)

virtual\_router\_id 51 #同一实例下virtual\_router\_id必须相同

mcast\_src\_ip 172.16.100.69

priority 240 #MASTER权重要高于BACKUP 比如BACKUP为240

advert\_int 1 #MASTER与BACKUP负载均衡器之间同步检查的时间间隔，单位是秒

nopreempt #非抢占模式

authentication { #设置认证

auth\_type PASS #主从服务器验证方式

auth\_pass 123456

}

track\_script {

check\_nginx

}

virtual\_ipaddress { #设置vip

#172.16.100.71 #可以多个虚拟IP，换行即可,71 has a server

172.16.100.72 #可以多个虚拟IP，换行即可,72 just ocupy ip addr,has no server

}

}

启动keepalived

systemctl start keepalived.service

查看状态

systemctl status keepalived.service

我们现在就用虚拟IP 172.16.100.72来访问我们刚才上传的图片，只是注意在地址栏中要记得输入fastdfs（这是我们nginx.conf文件中location /fastdfs{}规则规定的）。如下图所示，发现，我们通过虚拟IP便可以访问我们上传的图片了。这样的好处是，对用户来说，只需要访问这个虚拟IP就可以了，不用关心FastDFS集群内部的转发机制。