## MogileFS

[分布式存储之MogileFS基于Nginx实现负载均衡（Nginx+MogileFS）](http://www.cnblogs.com/daixiang/p/5610012.html)

**MogileFS分布式文件系统特点：**

　　1.具有raid的性能

　　2.不存在单点故障

　　3.简单的命名空间： 每个文件对应一个key：用于domain定义名称空间

  4.不共享任何数据

  5.传输中立，无特殊协议：可以通过NFS或HTTP进行通信

  6.自动文件复制：复制的最小单位不是文件，而class

　 7.应用层： 用户空间文件系统，无须特殊的核心组件

**Nginx+MogileFS的好处：**

　 1、将请求代理至后端MogileFS服务器集群中，能实现负载均衡的效果。

　 2、能对后端的tracker节点进行健康检测。

3、将第三方模块“nginx\_mogilefs\_module”编译进Nginx中，能实现直接使用key访问对应的文件，如下：

使用nginx做代理之前：http://192.168.80.137:7500/dev2/0/000/000/0000000007.fid

使用nginx做代理之后：http://192.168.80.132/image/1.jpg

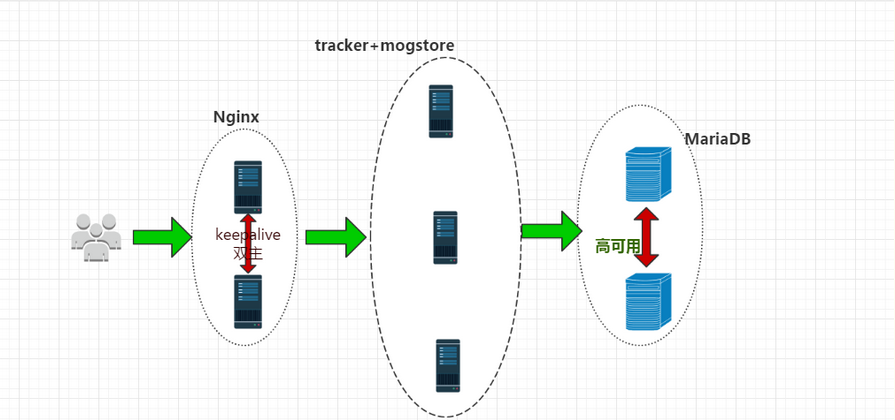
**MogileFS是由三个组件组成的：**

1、tracker：MogileFS的核心，是一个调度器，服务进程为mogilefsd，职责：删除数据、复制数据、监控、查询等。

　　 2、database：为tracker存储元素据

　　 3、数据存储的位置，通常是一个HTTP（webDAV）服务器，用来做数据的创建（put）、删除（delete）、获取（get），监听端口7500, storage节点使用http进行数据传输, 依赖于perbal, 进程为mogstored。

**理想中模型：**



Mysql 节点配置：

[root@t1 ~]# cat /etc/hosts

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.113.120 t1

192.168.113.121 t2

安装mysql：

[root@mariadb mysql]# useradd -r -s /sbin/nologin mysql

[root@mariadb ~]# tar xf mariadb-10.1.14-linux-x86\_64.tar.gz -C /usr/local/

[root@mariadb ~]# ln -sv /usr/local/mariadb-10.1.14-linux-x86\_64 /usr/local/mysql

[root@mariadb mysql]# chown -R mysql.mysql /usr/local/mysql/

[root@mariadb mysql]# ./scripts/mysql\_install\_db --user=mysql --datadir=/data

[root@mariadb mysql]# cp support-files/mysql.server /etc/rc.d/init.d/mysqld

[root@mariadb mysql]# chmod +x /etc/rc.d/init.d/mysqld

[root@mariadb mysql]# cp support-files/my-large.cnf /etc/my.cnf

[root@mariadb mysql]# vim /etc/my.cnf

         datadir = /data

[root@mariadb mysql]# ln -sv /usr/local/mysql/include/ /usr/include/mysql/

[root@mariadb mysql]# vim /etc/ld.so.conf.d/mysql.conf

         /usr/local/mysql/lib

[root@mariadb mysql]# vim /etc/profile.d/mysql.sh

         export PATH=/usr/local/mysql/bin:$PATH

[root@mariadb mysql]# ldconfig

[root@mariadb mysql]# ldconfig -p | grep mysql

    libmysqld.so.18 (libc6,x86-64) => /usr/local/mysql/lib/libmysqld.so.18

    libmysqld.so (libc6,x86-64) => /usr/local/mysql/lib/libmysqld.so

    libmysqlclient\_r.so.16 (libc6,x86-64) => /usr/lib64/mysql/libmysqlclient\_r.so.16

    libmysqlclient.so.18 (libc6,x86-64) => /usr/local/mysql/lib/libmysqlclient.so.18

    libmysqlclient.so.16 (libc6,x86-64) => /usr/lib64/mysql/libmysqlclient.so.16

    libmysqlclient.so (libc6,x86-64) => /usr/local/mysql/lib/libmysqlclient.so

    libgalera\_smm.so (libc6,x86-64) => /usr/local/mysql/lib/libgalera\_smm.so

**对用户进行授权：**

MariaDB [(none)]> grant all on \*.\* to 'root'@'192.168.113.%' identified by '123456';

Query OK, 0 rows affected (0.06 sec)

MariaDB [(none)]> grant all on mogilefs.\* to ' mogile'@'192.168.113.%' identified by '123456';

Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> flush privileges;

Query OK, 0 rows affected (0.00 sec)

**MogileFS 安装**

**上传包**

yum -y install \*.rpm

yum install perl-IO-AIO -y

**修改mogilefsd进程的配置文件：**

[root@mog1 ~]# vim /etc/mogilefs/mogilefsd.conf

# Enable daemon mode to work in background and use syslog

4 daemonize = 1 #以守护进程的形式运行

5 # Where to store the pid of the daemon (must be the same in the init script)

6 pidfile = /var/run/mogilefsd/mogilefsd.pid

7 # Database connection information

8 db\_dsn = DBI:mysql:mogilefs:host=192.168.80.135 #定义数据库名为mogilefs和数据库服务器地址

9 db\_user = moguser #定义管理此数据库的用户名

10 db\_pass = mogpass #定义密码

11 # IP:PORT to listen on for mogilefs client requests

12 listen = 0.0.0.0:7001 #定义监听的地址和端口

13 # Optional, if you don't define the port above.

14 conf\_port = 7001

15 # Number of query workers to start by default.

16 query\_jobs = 10 #定义启动查询线程个数

17 # Number of delete workers to start by default.

18 delete\_jobs = 1 #定义启动删除线程个数

19 # Number of replicate workers to start by default.

20 replicate\_jobs = 5 #定义启动复制线程个数

21 # Number of reaper workers to start by default.

22 # (you don't usually need to increase this)

23 reaper\_jobs = 1 #响应客户端请求，在磁盘失败后将请求重新放到队列中

24 # Number of fsck workers to start by default.

25 # (these can cause a lot of load when fsck'ing)

26 #fsck\_jobs = 1 #对磁盘进行检测，默认没有启动

27 # Minimum amount of space to reserve in megabytes

28 # default: 100 #默认保留100M空间

29 # Consider setting this to be larger than the largest file you

30 # would normally be uploading.

31 #min\_free\_space = 200 #最小空闲空间为200M

32 # Number of seconds to wait for a storage node to respond.

33 # default: 2 #等待存储节点相应的时间，默认为2s

34 # Keep this low, so busy storage nodes are quickly ignored.

35 #node\_timeout = 2 #等待节点的超时时长

36 # Number of seconds to wait to connect to a storage node.

37 # default: 2 #连接存储节点的超时时长

38 # Keep this low so overloaded nodes get skipped.

39 #conn\_timeout = 2

40 # Allow replication to use the secondary node get port,

41 # if you have apache or similar configured for GET's

42 #repl\_use\_get\_port = 1

**设置tracker对应的数据库，生成mogilefs库：**

[root@mog1 ~]# mogdbsetup --dbhost=192.168.80.135 --dbport=3306 --dbname=mogilefs --dbrootuser=root --dbrootpass=rootpass --dbuser=moguser --dbpass=mogpass --yes

#多个tracker共享一个库，所以其他tracker节点上就不需要在做此操作，直接修改配置文件/etc/mogilefs/mogilefs.conf就可以

对mogstored进程进行配置：

https://images.cnblogs.com/OutliningIndicators/ExpandedBlockStart.gif

[root@mog1 ~]# vim /etc/mogilefs/mogstored.conf

maxconns = 10000

httplisten = 0.0.0.0:7500

mgmtlisten = 0.0.0.0:7501

docroot = /var/mogdata

[root@t2 log]# mkdir /var/mogdata

[root@t2 log]# chown mogilefs.mogilefs /var/mogdata -R

启动 [root@t2 log]# service mogstored start

检查启动端口 [root@t2 log]# ss -tanlp|grep 7500

**对各节点进行管理：**

[root@mog1 ~]# mogadm --trackers=192.168.80.136:7001 host add mog1 --ip=192.168.80.136 --status=alive

[root@mog1 ~]# mogadm --trackers=192.168.80.136:7001 device add mog1 001 --status=alive

创建目录设备目录：

mkdir /var/mogdata/dev1

[root@mog1 ~]# mogadm --trackers=192.168.80.136:7001 host add mog2 --ip=192.168.80.137 --status=alive

[root@mog1 ~]# mogadm --trackers=192.168.80.136:7001 device add mog2 002 --status=alive

 创建目录设备目录：

mkdir /var/mogdata/dev2

[root@mog1 ~]# mogadm --trackers=192.168.80.136:7001 host add mog3 --ip=192.168.80.138 --status=alive

[root@mog1 ~]# mogadm --trackers=192.168.80.136:7001 device add mog3 003 --status=alive

创建目录设备目录：

mkdir /var/mogdata/dev3

添加域

[root@mog1 ~]# mogadm domain add tcjfweb

[root@mog1 ~]# mogadm domain add yundong

添加类：

[root@mog1 ~]# mogadm class add tcjfweb class1 --mindevcount=3

[root@mog1 ~]# mogadm class add tcjfweb class2 --mindevcount=2

[root@mog1 ~]# mogadm class add yundong dx1 --mindevcount=2

查看trackers状态：

[root@t1 ~]# mogadm --trackers=192.168.113.120:7001,192.168.113.121:7001 check

查看主机列表：

[root@t1 ~]# mogadm --trackers=192.168.113.120:7001,192.168.113.121:7001 host list

查看设备：

[root@t1 ~]# mogadm --trackers=192.168.113.120:7001,192.168.113.121:7001 device list

查看域列表：

[root@t1 ~]# mogadm --trackers=192.168.113.120:7001,192.168.113.121:7001 domain list

上传文件进行测试：

[root@mog1 ~]# mogupload --trackers=192.168.113.120:7001 --domain=linux1 --key='/1.jpg' --file='/root/centos.jpg'

查看文件

[root@mog1 ~]# mogfileinfo --tracker=192.168.113.120:7001 --domain=linux1 --key='/1.jpg'

[root@mog1 ~]# mogupload --trackers=192.168.113.120:7001 --domain=python1 --key='/fstab.html' --file='/etc/fstab'

查看文件

[root@mog1 ~]# mogfileinfo --tracker=192.168.113.120:7001 --domain=python1 --key='/fstab.html'

安装nginx：

上传nginx安装包和mogilefs模块包

[root@nginx ~]# # yum groupinstall "Development Tools" "Server Platform Deveopment" -y

[root@nginx ~]# # yum install openssl-devel pcre-devel -y

[root@nginx ~]# useradd -r nginx

[root@nginx ~]# tar xf nginx\_mogilefs\_module-1.0.4.tar.gz

[root@nginx ~]# tar xf nginx-1.10.0.tar.gz

[root@nginx ~]# cd nginx-1.10.0

[root@nginx nginx-1.10.0]#  ./configure \

  --prefix=/usr/local/nginx \

  --sbin-path=/usr/sbin/nginx \

  --conf-path=/etc/nginx/nginx.conf \

  --error-log-path=/var/log/nginx/error.log \

  --http-log-path=/var/log/nginx/access.log \

  --pid-path=/var/run/nginx/nginx.pid  \

  --lock-path=/var/lock/nginx.lock \

  --user=nginx \

  --group=nginx \

  --with-http\_ssl\_module \

  --with-http\_flv\_module \

  --with-http\_stub\_status\_module \

  --with-http\_gzip\_static\_module \

  --http-client-body-temp-path=/var/tmp/nginx/client/ \

  --http-proxy-temp-path=/var/tmp/nginx/proxy/ \

  --http-fastcgi-temp-path=/var/tmp/nginx/fcgi/ \

  --http-uwsgi-temp-path=/var/tmp/nginx/uwsgi \

  --http-scgi-temp-path=/var/tmp/nginx/scgi \

  --with-pcre \

  --with-debug \

  --add-module=/root/nginx\_mogilefs\_module-1.0.4

#添加的第三方模块，实现直接使用key作为uri进行访问

[root@nginx nginx-1.10.0]#  make && make install

**提供nginx启动脚本：**

[root@nginx ~]# 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: /etc/sysconfig/nginx

# pidfile: /var/run/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/sbin/nginx"

prog=$(basename $nginx)

NGINX\_CONF\_FILE="/etc/nginx/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' -`

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

[root@nginx ~]# chmod +x /etc/rc.d/init.d/nginx

修改nginx的配置如下：

user root;

worker\_processes 1;

error\_log /var/log/nginx/error.log;

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 images {

server 192.168.113.120:7001;

server 192.168.113.121:7001;

}

server {

listen 80;

server\_name localhost;

#charset koi8-r;

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

location /image {

mogilefs\_tracker images;

mogilefs\_domain tcjfweb;

mogilefs\_methods GET;

mogilefs\_noverify on;

mogilefs\_pass {

proxy\_pass $mogilefs\_path;

proxy\_hide\_header Content-Type;

proxy\_buffering off;

}

}

location /files {

mogilefs\_tracker images;

mogilefs\_domain yundong;

mogilefs\_methods GET;

mogilefs\_noverify on;

mogilefs\_pass {

proxy\_pass $mogilefs\_path;

proxy\_hide\_header Content-Type;

proxy\_buffering off;

}

}

}

}

**启动nginx：**

[root@nginx ~]# service nginx start

访问测试：

http://192.168.113.120/image/3.jpg

