链滴

CentOS7 部署 fastDFS 集群

作者：k8s
原文链接：https://ld246.com/article/1589728307138
来源网站：链滴

1. 部署前准备

1.1 服务器信息

主机名	IP 地址	相关服务和角色
node01	10.40.96.4	keepalived ipvsadm
node02	10.40.96.5	keepalived ipvsadm
node03	10.40.96.6	Nginx cache + fastdfs Tracker
node04	10.40.96.7	Nginx cache + fastdfs Tracker
node05	10.40.96.8	Group1/fastdfs-nginx-module + fastdfs Storage
node06	10.40.96.9	Group1/fastdfs-nginx-module + fastdfs Storage
node07	10.40.96.10	Group2/fastdfs-nginx-module + fastdfs Storage
node08	10.40.96.11	Group2/fastdfs-nginx-module + fastdfs Storage
--	10.40.96.30	VIP

1.2 集群架构图

1.3 下载源文件到对应节点

源文件	存放节点
tengine-2.2.3.tar.gz	node01 - node08
ngx_cache_purge-2.3.tar.gz	node03、node04
FastDFS_v5.08.tar.gz	node03 - node08
libfastcommon-master.zip	node03 - node08
fastdfs-nginx-module_v1.16.tar.gz	node05 - node08

2. 编译安装服务

2.1 安装 FastDFS

在 node03 - node08 节点上编译安装 fastdfs，执行如下脚本

```
#!/bin/bash
# @Author zhanjie.me
# @email i@zhanjie.me
set -e

# 设置源文件存放目录
src_dir=/usr/local/src
# 安装make、cmake 和 gcc 编译器
yum install -y make cmake gcc gcc-c++

# 安装FastDFS核心库
cd $src_dir && \
unzip libfastcommon-master.zip && \
cd libfastcommon-master && \
./make.sh && \
./make.sh install

# 为fastdfs主程序引用库文件
ln -s /usr/lib64/libfastcommon.so /usr/local/lib/libfastcommon.so || true
ln -s /usr/lib64/libfastcommon.so /usr/lib/libfastcommon.so       || true
ln -s /usr/lib64/libfdfsclient.so /usr/local/lib/libfdfsclient.so || true
ln -s /usr/lib64/libfdfsclient.so /usr/lib/libfdfsclient.so       || true

# FastDFS主程序安装
cd $src_dir && \
tar zxf FastDFS_v5.08.tar.gz && \
cd FastDFS && \
./make.sh && \
./make.sh install


echo "=============Install Success!=============="
```

2.1.1 分发执行脚本

```
for ((i=3;i<9;i++));do rsync -avzP fastDFS_install.sh root@node0$i:~/ ;done

for ((i=3;i<9;i++));do ssh root@node0$i bash /root/fastDFS_install.sh ;done
```

2.2 安装 tengine

我们在此处使用 tengine 代替原版 nginx

在 node03、node04 上扩展 ngx_cache_purge 安装 tengine

在 node05 - node08 上扩展 fastdfs-nginx-module、ngx_cache_purge 安装 tengine

执行脚本如下

```
#!/bin/bash
# @Author zhanjie.me
```

```
# @email i@zhanjie.me
set -e
name=`basename $0`
# 设置源文件存放目录
src_dir=/usr/local/src

case $1 in
  0)
    echo 0
    cd $src_dir && \
    tar zxf tengine-2.2.3.tar.gz
    configure_command="./configure --prefix=/usr/local/tengine"
    ;;
  1)
    echo 1
    cd $src_dir && \
    tar zxf tengine-2.2.3.tar.gz && \
    tar zxf ngx_cache_purge-2.3.tar.gz
    configure_command="./configure --prefix=/usr/local/tengine --add-module=${src_dir}/ng
_cache_purge-2.3"
    ;;
  2)
    echo 2
    cd $src_dir && \
    tar zxf tengine-2.2.3.tar.gz && \
    tar zxf ngx_cache_purge-2.3.tar.gz && \
    tar zxf fastdfs-nginx-module_v1.16.tar.gz && \
    sed -i "s#/usr/local/include#/usr/include#g" fastdfs-nginx-module/src/config && \
    mkdir -p /etc/fdfs && \
    cp fastdfs-nginx-module/src/mod_fastdfs.conf /etc/fdfs
    configure_command="./configure --prefix=/usr/local/tengine --add-module=${src_dir}/ng
_cache_purge-2.3 --add-module=${src_dir}/fastdfs-nginx-module/src"
    ;;
  *)
    echo "Usage: bash $name [0|1|2]"
    exit 1
    ;;
esac

# 安装make、cmake 和 gcc 编译器
yum install -y make cmake gcc gcc-c++
# 安装依赖
yum install -y pcre-devel openssl-devel

# 编译安装tengine
cd ${src_dir}/tengine-2.2.3 && \
bash -c \
"$configure_command && make && make install"

# 注入启动文件
cat > /usr/lib/systemd/system/tengine.service << EOF
[Unit]
Description=The nginx HTTP and reverse proxy server
After=syslog.target network.target remote-fs.target nss-lookup.target
```

```
[Service]
Type=forking
PIDFile=/usr/local/tengine/logs/nginx.pid
ExecStartPre=/usr/local/tengine/sbin/nginx -t
ExecStart=/usr/local/tengine/sbin/nginx -c /usr/local/tengine/conf/nginx.conf
ExecReload=/bin/kill -s HUP $MAINPID
ExecStop=/bin/kill -s QUIT $MAINPID
PrivateTmp=true

[Install]
WantedBy=multi-user.target
EOF
systemctl daemon-reload

echo "=========Install Success============"
```

2.2.1 分发执行脚本

```
for ((i=3;i<9;i++)); do rsync -avzP tengine_install.sh root@node0$i:~/ ;done
```

针对 node03、node04 节点执行：

```
for ((i=3;i<5;i++)); do ssh root@node0$i bash /root/tengine_install.sh 1  ;done
```

针对 node05 - node08 节点执行：

```
for ((i=5;i<9;i++)); do ssh root@node0$i bash /root/tengine_install.sh 2  ;done
```

2.3 安装 keepalived

在 node01、node02 上安装 keepalived、ipvsadm

```
yum install -y keepalived ipvsadm
```

3. 配置服务

3.1 配置 fastdfs-tracker 服务

分别在 node03、node04 节点上执行如下操作

创建 base_path 目录

```
[root@node03 ~]# mkdir /var/data/fastdfs-tracker
```

生成配置文件

```
[root@node03 ~]# cd /etc/fdfs/
[root@node03 fdfs]# ls
client.conf.sample  storage.conf.sample  tracker.conf.sample
[root@node03 fdfs]# cp tracker.conf.sample tracker.conf
```

修改 base_url 路径

```
[root@node03 fdfs]# vim tracker.conf
...
# the base path to store data and log files
base_path=/var/data/fastdfs-tracker
...
```

设置服务开机启动

```
[root@node03 ~]# systemctl enable fdfs_trackerd.service
```

启动服务

```
[root@node03 ~]# systemctl start fdfs_trackerd
[root@node03 ~]# systemctl status fdfs_trackerd.service
```

3.2 配置 fastdfs-storage 服务

分别在 node05-node08 节点上执行如下操作

创建 base_path 目录

```
mkdir -p /var/data/fastdfs-storage/base
```

创建 store_path0 目录

```
mkdir -p /var/data/fastdfs-storage/store
```

生成配置文件

```
[root@node05 ~]# cd /etc/fdfs/
[root@node05 fdfs]# cp storage.conf.sample storage.conf
```

修改 group_name,base_path, store_path0 和 tracker_server

```
[root@node05 fdfs]# vim storage.conf
...
group_name=group1  #在node07、node08上为group2
base_path=/var/data/fastdfs-storage/base
store_path0=/var/data/fastdfs-storage/store
tracker_server=10.40.96.6:22122
tracker_server=10.40.96.7:22122
# the port of the web server on this storage server
http.server_port=80
...
```

为 store_path0 生成虚拟目录 M00

```
ln -s /var/data/fastdfs-storage/store/data  /var/data/fastdfs-storage/store/data/M00
```

设置服务开机启动

```
[root@node05 ~]# systemctl enable fdfs_storaged.service
```

启动服务

```
[root@node05 fdfs]# systemctl start fdfs_storaged
[root@node05 fdfs]# systemctl status fdfs_storaged
```

3.3 配置 fastdfs-storage 节点上的 nginx（tengine）服务

在 node05-node08 节点上执行如下操作

生成 http.conf 配置文件

```
cp /usr/local/src/FastDFS/conf/http.conf /etc/fdfs/
```

生成 mime.types 配置文件

```
cp /usr/local/src/FastDFS/conf/mime.types /etc/fdfs/
```

生成 fastdfs-nginx-module 的配置文件

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

修改配置文件

```
vim /etc/fdfs/mod_fastdfs.conf
...
connect_timeout=10
#node05、node06 对应 group 1
#node07、node08 对应 group 2
group_name=group1/2
group_count = 2
tracker_server=10.40.96.6:22122
tracker_server=10.40.96.7:22122
url_have_group_name = true
[group1]
group_name=group1
storage_server_port=23000
store_path_count=1
store_path0=/var/data/fastdfs-storage/store
[group2]
group_name=group2
storage_server_port=23000
store_path_count=1
store_path0=/var/data/fastdfs-storage/store
```

修改 nginx（tengine）配置文件

```
  listen       80;

  location ~ /group([0-9])/M00 {
    add_header Content-Disposition "attachment;filename=$arg_attname";
    ngx_fastdfs_module;
  }
```

配置开机启动并启动服务

```
systemctl enable tengine
```

```
systemctl start tengine
```

3.4 配置 fastdfs-tracker 节点上的 nginx（tengine）服务

创建缓存目录

```
mkdir -p /var/data/cache/nginx/proxy_cache
```

修改 nginx.conf 文件

```
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  /var/log/nginx/access.log  main;

    sendfile        on;

    keepalive_timeout  65;

    #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
    #levels=1:2 表示缓存文件有两级目录 1表示第一级目录名为1位数，2表示第二级目录名为2位数
    proxy_cache_path /var/data/cache/nginx/proxy_cache levels=1:2
    #keys_zone 缓存区域名字，分配200m空间，最大缓存1g,有效期30天
    keys_zone=http-cache:200m max_size=1g inactive=30d;

    proxy_temp_path /var/data/cache/nginx/proxy_cache/tmp;

    #设置 group1 的服务器
    upstream fdfs_group1 {
      server 10.40.96.8:80 weight=1 max_fails=2 fail_timeout=30s;
      server 10.40.96.9:80 weight=1 max_fails=2 fail_timeout=30s;
    }
```

```
    #设置 group2 的服务器
    upstream fdfs_group2 {
      server 10.40.96.10:80 weight=1 max_fails=2 fail_timeout=30s;
      server 10.40.96.11:80 weight=1 max_fails=2 fail_timeout=30s;
    }


    server {
        listen       80;
        #charset koi8-r;
        #access_log  logs/host.access.log  main;
        #access_log  "pipe:rollback logs/host.access_log interval=1d baknum=7 maxsize=2G"  m
in;
        server_name  localhost;
        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 10.40.96.0/20;
            deny all;
            proxy_cache_purge http-cache $1$is_args$args;
        }
    }
}
```

配置开机启动并启动服务

```
systemctl enable tengine
systemctl start tengine
```

3.5 配置 keepalived 服务

在 node01、node02 上执行如下操作

开启网络转发

```
# open ip_forward
```

```
echo "1" > /proc/sys/net/ipv4/ip_forward
# edit sysctl.conf
vi /etc/sysctl.conf
net.ipv4.ip_forward = 1
sysctl -p
```

修改 keepalived.conf

node01 节点:

```
! Configuration File for keepalived

global_defs {
  ## keepalived 自带的邮件提醒需要开启 sendmail 服务。建议用独立的监控或第三方 SMTP
  router_id node01  ## 标识本节点的字条串，通常为 hostname
}
## 定义虚拟路由，VI_1 为虚拟路由的标示符，自己定义名称
vrrp_instance VI_1 {
   state MASTER  ## 主节点为 MASTER，对应的备份节点为 BACKUP
   interface eth1  ## 绑定虚拟 IP 的网络接口，与本机 IP 地址所在的网络接口相同，我的是 eth1
   virtual_router_id 96  ## 虚拟路由的 ID 号，两个节点设置必须一样，可选 IP 最后一段使用，相
的 VRID 为一个组，他将决定多播的 MAC 地址
   mcast_src_ip 10.40.96.4  ## 本机 IP 地址
   priority 100  ## 节点优先级，值范围 0-254，MASTER 要比BACKUP 高
   nopreempt ## nopreempt 解决异常恢复后再次抢占的问题
   advert_int 1  ## 组播信息发送间隔，两个节点设置必须一样，默认 1s
   ## 设置验证信息，两个节点必须一致
   authentication {
      auth_type PASS
      auth_pass zhanjie.me
   }

   ## 虚拟 IP 池, 两个节点设置必须一样
   virtual_ipaddress {
      10.40.96.30/20   dev  eth1  label  eth1:1
   }
}
## 定义lvs
virtual_server 10.40.96.30 80 {
   delay_loop 6        #(每隔6秒查询realserver状态)
   lb_algo wlc            #(lvs 算法)
   lb_kind DR           #(Direct Route)
   persistence_timeout 60    #(同一IP的连接60秒内被分配到同一台realserver)
   protocol TCP        #(用TCP协议检查realserver状态)
   real_server 10.40.96.6 80 {
   weight 100        #(权重)
   TCP_CHECK {
     connect_timeout 10    #(10秒无响应超时)
     delay_before_retry 3
     connect_port 80
      }
   }
   real_server 10.40.96.7 80 {
   weight 100        #(权重)
```

```
    TCP_CHECK {
      connect_timeout 10    #(10秒无响应超时)
      delay_before_retry 3
      connect_port 80
       }
    }
}
```

node02 节点：

```
! Configuration File for keepalived

global_defs {
   ## keepalived 自带的邮件提醒需要开启 sendmail 服务。建议用独立的监控或第三方 SMTP
   router_id node02  ## 标识本节点的字条串，通常为 hostname
}
## 定义虚拟路由，VI_1 为虚拟路由的标示符，自己定义名称
vrrp_instance VI_1 {
    state BACKUP  ## 主节点为 MASTER，对应的备份节点为 BACKUP
    interface eth1  ## 绑定虚拟 IP 的网络接口，与本机 IP 地址所在的网络接口相同，我的是 eth1
    virtual_router_id 96  ## 虚拟路由的 ID 号，两个节点设置必须一样，可选 IP 最后一段使用，相
的 VRID 为一个组，他将决定多播的 MAC 地址
    mcast_src_ip 10.40.96.5  ## 本机 IP 地址
    priority 90  ## 节点优先级，值范围 0-254，MASTER 要比BACKUP 高
    nopreempt ## nopreempt 解决异常恢复后再次抢占的问题
    advert_int 1  ## 组播信息发送间隔，两个节点设置必须一样，默认 1s
    ## 设置验证信息，两个节点必须一致
    authentication {
        auth_type PASS
        auth_pass zhanjie.me
    }

    ## 虚拟 IP 池, 两个节点设置必须一样
    virtual_ipaddress {
        10.40.96.30/20   dev  eth1  label  eth1:1
    }
}
## 定义lvs
virtual_server 10.40.96.30 80 {
    delay_loop 6                #(每隔6秒查询realserver状态)
    lb_algo wlc                 #(lvs 算法)
    lb_kind DR                  #(Direct Route)
    persistence_timeout 60      #(同一IP的连接60秒内被分配到同一台realserver)
    protocol TCP                #(用TCP协议检查realserver状态)
    real_server 10.40.96.6 80 {
        weight 100              #(权重)
        TCP_CHECK {
          connect_timeout 10   #(10秒无响应超时)
          delay_before_retry 3
          connect_port 80
        }
    }
    real_server 10.40.96.7 80 {
        weight 100              #(权重)
```

```
        TCP_CHECK {
          connect_timeout 10   #(10秒无响应超时)
          delay_before_retry 3
          connect_port 80
        }
    }
}
```

配置 keepalived 开机启动，并启动服务

```
systemctl enable keepalived.service
systemctl start keepalived.service
```

3.6 配置 lvs_real-server

在 node03、node04 上执行如下操作

为 look 网卡加类型配置

```
echo "TYPE=Loopback" >> /etc/sysconfig/network-scripts/ifcfg-lo
```

新建 lvs_reald 服务

```
[root@node03 ~]# vim /etc/init.d/lvs_reald
#!/bin/bash
### BEGIN INIT INFO
# Provides:
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Start realserver
# Description:       Start realserver
### END INIT INFO

# change the VIP to proper value
VIP=10.40.96.30

case "$1" in
    start)

    echo "Start REAL Server"
    /sbin/ifconfig lo:0 $VIP broadcast $VIP netmask 255.255.255.255 up
    echo "1" >/proc/sys/net/ipv4/conf/lo/arp_ignore
    echo "2" >/proc/sys/net/ipv4/conf/lo/arp_announce
    echo "1" >/proc/sys/net/ipv4/conf/all/arp_ignore
    echo "2" >/proc/sys/net/ipv4/conf/all/arp_announce

    ;;

    stop)

    /sbin/ifconfig lo:0 down
    echo "Stop REAL Server"
    echo "0" >/proc/sys/net/ipv4/conf/lo/arp_ignore
    echo "0" >/proc/sys/net/ipv4/conf/lo/arp_announce
```

```
    echo "0" >/proc/sys/net/ipv4/conf/all/arp_ignore
    echo "0" >/proc/sys/net/ipv4/conf/all/arp_announce

    ;;

    restart)

    $0 stop
    $0 start

    ;;

    *)

    echo "Usage: $0 {start|stop}"
    exit 1

    ;;
esac

exit 0

[root@node03 ~]# chmod a+x /etc/init.d/lvs_reald
```

配置开机启动并启动服务

```
chkconfig lvs_reald on
service lvs_reald start
```