Nginx-反向代理实现

Nginx 反向代理操作案例

Nginx反向代理的组件模块

Nginx http 功能模块	模块说明
ngx_http_proxy_module	proxy 代理模块，用于把请求后抛给服务器节点或upstream 服务器迟
ngx_http_upstream_module	负载均衡模块，可以实现网站的负载均衡功能及节点的健康检查。创建一个池子，web服务器（服务员）

upstream模块介绍->点我<

http_proxy_module模块介绍->点我<

环境准备

IP	主机名	角色	
192.168.3.101	centos7-1	LB1	源码安装nginx
192.168.3.102	centos7-2	LB2	源码安装nginx
192.168.3.103	centos7-3	web1	yum安装nginx
192.168.3.104	centos7-3	web2	yum安装nginx

1）四台服务器都需操作如下步骤：

```
# systemctl stop firewalld        //关闭防火墙
# sed -i 's/^SELINUX=.*/SELINUX=disabled/' /etc/sysconfig/selinux        //关闭selinux，重启生效
# setenforce 0        //关闭selinux，临时生效
# ntpdate 0.centos.pool.ntp.org        //时间同步
```

2）LB节点源码安装nginx：

```
# yum install openssl openssl-devel pcre pcre-devel gcc gcc-c++ make autoconf automake -y        //安装依赖工具包

//编写安装脚本
# cat >installNginx.sh<<EOF
mkdir /home/tools
cd /home/tools
wget -q http://nginx.org/download/nginx-1.12.2.tar.gz
ls -l nginx-1.12.2.tar.gz
useradd nginx -s /sbin/nologin -M
tar xf nginx-1.12.2.tar.gz
cd nginx-1.12.2
./configure --user=nginx --group=nginx --prefix=/application/nginx-1.12.2 \
```

```
 --with-http_stub_status_module --with-http_ssl_module
make
make install
ln -s /application/nginx-1.12.2/ /application/nginx
EOF

# bash installNginx.sh        //执行安装脚本
```

说明：上面源码安装的nginx，配置文件路径：/application/nginx/conf/nginx.conf　二进制启动命令路径：/application/nginx/sbin/nginx

3）web节点yum安装nginx及准备测试文件：

```
# yum install nginx -y        //安装nginx
# mkdir /application/nginx/html/{www,bbs,blog} -p        //创建web站点目录
# for dir in www bbs blog; do echo "`hostname` $dir" >/application/nginx/html/$dir/index.html;done //创建站点目录测试文件

# vim /etc/nginx/nginx.conf        //编辑配置文件
user nginx;
worker_processes auto;
error_log /var/log/nginx/error.log;
pid /run/nginx.pid;

include /usr/share/nginx/modules/*.conf;

events {
    worker_connections 1024;
}

http {
    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;
tcp_nopush          on;
tcp_nodelay         on;
keepalive_timeout   65;
types_hash_max_size 2048;

include             /etc/nginx/mime.types;
default_type        application/octet-stream;
include /etc/nginx/conf.d/*.conf;

server {
    listen       80;
    server_name  www.etiantian.org;

    location / {
    root /application/nginx/html/www;
    }
    access_log      /var/log/nginx/access_www.log main;
}

server {
    listen       80;
    server_name  bbs.etiantian.org;

    location / {
    root /application/nginx/html/bbs;
    }
    access_log      /var/log/nginx/access_bbs.log main;
}
server {
    listen       80;
    server_name  blog.etiantian.org;
```

```
        location / {
        root /application/nginx/html/blog;
        }
        access_log      /var/log/nginx/access_blog.log main;
    }
}

# systemctl start nginx         //启动nginx
# systemctl enable nginx         //加入开机自启动
```

4）web站点配置hosts解析及测试nginx是否能够正常访问

```
//web1站点编辑后的/etc/hosts文件
[root@centos7-3 ~]# tail -3 /etc/hosts
192.168.3.103   www.etiantian.org
192.168.3.103   bbs.etiantian.org
192.168.3.103   blog.etiantian.org

//web2站点编辑后的/etc/hosts文件
[root@centos7-4 ~]# tail -3 /etc/hosts
192.168.3.104   www.etiantian.org
192.168.3.104   bbs.etiantian.org
192.168.3.104   blog.etiantian.org

//web1站点测试
[root@centos7-3 ~]# curl www.etiantian.org
centos7-3 www
[root@centos7-3 ~]# curl bbs.etiantian.org
centos7-3 bbs
[root@centos7-3 ~]# curl blog.etiantian.org
centos7-3 blog

//web2站点测试
```

```
[root@centos7-4 ~]# curl www.etiantian.org
centos7-4 www
[root@centos7-4 ~]# curl bbs.etiantian.org
centos7-4 bbs
[root@centos7-4 ~]# curl blog.etiantian.org
centos7-4 blog
```

案例

完成上面的lb节点的软件安装及web节点的测试文件准备后，下面开始配置案例，说明，先配置单节点的lb，也就是先只在（centos7-1）lb1 上面进行配置。

案例一：最基本的负载均衡

编辑lb1（192.168.3.101）配置文件，编辑之前记得将默认配置文件进行备份

```
# cp /application/nginx/conf/nginx.conf /application/nginx/conf/nginx.conf.default        //备份配置文件
# sed -i '/^[ ]*$/d' /application/nginx/conf/nginx.conf        //去掉配置文件中的注释及空行

# vim /application/nginx/conf/nginx.conf        //编辑配置文件
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;

    upstream myapp1 {
        #<== upstream 是关键字必须要有，后面的myapp1为一个Upstream集群组的名字，可以自定义，调用时就用这个名字。
        server 192.168.3.103 weight=1;
        #<==server 关键字是固定的，后面可以接域名或IP，如果不指定端口，
```

```
        #默认是80端口。weight 代表权重，数值越大被分配到请求越多，默认值为1，所以此处可写可不写。结尾有分号，别忘了；
        server 192.168.3.104 weight=1;
     }
    server {
        listen       80;
        server_name  localhost;
    location / {
            proxy_pass http://myapp1;
        }
    }
}

# /application/nginx/sbin/nginx -t        //检查语法是否错误
# /application/nginx/sbin/nginx -s reload        //重新加载配置文件
```

我们用lb2机器来测试（用任何一台都可以），测试结果可以看出，会轮循调度到后端web节点上

```
[root@centos7-2 ~]# curl 192.168.3.101
centos7-3 www
[root@centos7-2 ~]# curl 192.168.3.101
centos7-4 www
[root@centos7-2 ~]# curl 192.168.3.101
centos7-3 www
[root@centos7-2 ~]# curl 192.168.3.101
centos7-4 www
```

案例二：基于权重（wrr）

修改配置文件 upstream 段为以下内容

```
upstream myapp1 {
    server 192.168.3.103 weight=1;
    server 192.168.3.104 weight=2;
}
```

同样使用lb2机器来进行测试，可以发现调度后端节点编程了1:2，调度到web2节点上面总是会多一次。

```
[root@centos7-2 ~]# curl 192.168.3.101
centos7-3 www
[root@centos7-2 ~]# curl 192.168.3.101
centos7-4 www
[root@centos7-2 ~]# curl 192.168.3.101
centos7-4 www
[root@centos7-2 ~]# curl 192.168.3.101
centos7-3 www
[root@centos7-2 ~]# curl 192.168.3.101
centos7-4 www
[root@centos7-2 ~]# curl 192.168.3.101
centos7-4 www
```

案例三：较完整的 upstream 配置案例

修改配置文件 upstream 段为以下内容

```
upstream myapp1 {
    server 192.168.3.103 weight=1 max_fails=3 fail_timeout=20s;
    server 192.168.3.104 weight=1 max_fails=3 fail_timeout=20s;
}
//max_fails 尝试连接后端主机失败的次数; fail_timeout 在max_fails定义的失败次数后，距离下次检查的间隔时间。
```

同样使用lb2机器来进行测试，在测试过程中，关闭其中一个web节点，会发现只是调度到另外一个节点上面，然后再重启关闭的节点，观察测试输出内容，会发现尝试的时间。

```
[root@centos7-2 ~]# for n in {1..100}; do curl 192.168.3.101 ; date +%T; sleep 1; done
```

案例四：基于域名的负载

修改配置文件为以下内容

```
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;

    upstream myapp1 {
        server 192.168.3.103 weight=1 max_fails=3 fail_timeout=20s;
        server 192.168.3.104 weight=1 max_fails=3 fail_timeout=20s;
     }

    server {
        listen       80;
        server_name  www.etiantian.org;
    location / {
            proxy_pass http://myapp1;
            proxy_set_header Host $host;
        }
    }

    server {
        listen       80;
        server_name  bbs.etiantian.org;
    location / {
            proxy_pass http://myapp1;
            proxy_set_header Host $host;
        }
    }
```

```
    server {
        listen       80;
        server_name  blog.etiantian.org;
    location / {
            proxy_pass http://myapp1;
            proxy_set_header Host $host;
        }
    }
}
```

编辑/etc/hosts文件，进行域名解析，此处为了方便，直接在lb1节点上面编辑并测试（如果需要在别的节点进行测试，那么进行域名解析即可）

```
# vim /etc/hosts
192.168.3.101   www.etiantian.org bbs.etiantian.org blog.etiantian.org
```

测试发现基于域名ok，因为上面配置的权重都为1，所以不论我们访问哪一个域名，都会轮循去调度后端web节点。

```
[root@centos7-1 ~]# curl www.etiantian.org
centos7-4 www
[root@centos7-1 ~]# curl www.etiantian.org
centos7-3 www
[root@centos7-1 ~]# curl bbs.etiantian.org
centos7-4 bbs
[root@centos7-1 ~]# curl bbs.etiantian.org
centos7-3 bbs
[root@centos7-1 ~]# curl blog.etiantian.org
centos7-4 blog
[root@centos7-1 ~]# curl blog.etiantian.org
centos7-3 blog
```

案例五：记录客户端真实IP

先到web节点上面查看访问日志

```
[root@centos7-3 ~]# tailf /var/log/nginx/access_www.log
192.168.3.101 - - [08/Apr/2019:00:18:14 +0800] "GET / HTTP/1.0" 200 14 "-" "curl/7.29.0" "-"
192.168.3.101 - - [08/Apr/2019:00:18:16 +0800] "GET / HTTP/1.0" 200 14 "-" "curl/7.29.0" “-"
```

通过观察日志发现，记录的都负载均衡器节点的IP，实际生产环境中都是记录真实客户端IP。

进行修改配置文件（lb节点）将location 段加上 proxy_set_header X-Forwarded-For $remote_addr;

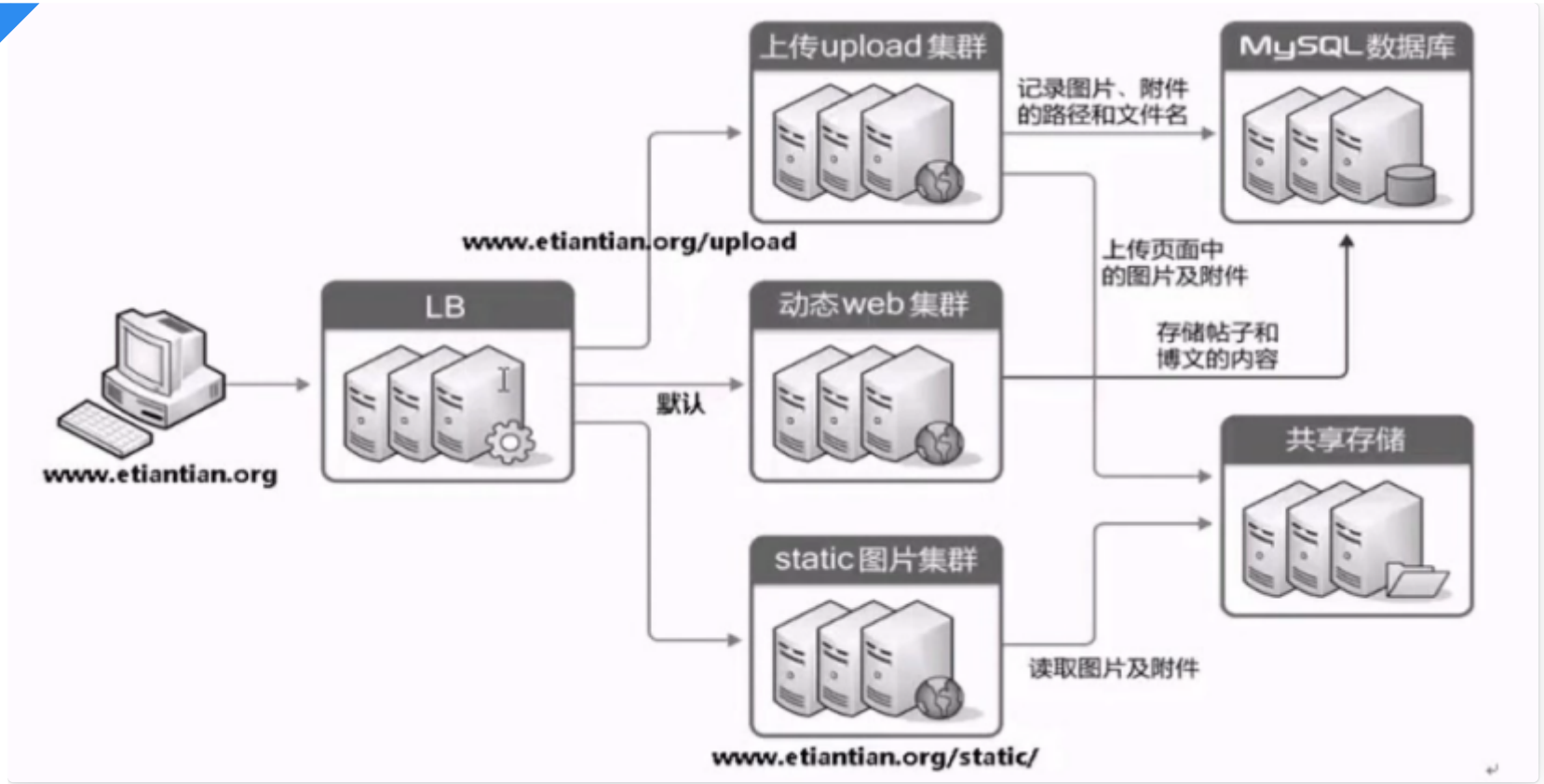
```
location / {
        proxy_pass http://myapp1;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $remote_addr;
    }
```

使用lb2节点进行测试（我们将lb2当做一个客户端访问），然后我们再去查看日志就可以发现，记录了客户端的真实IP地址。

```
[root@centos7-3 ~]# tailf /var/log/nginx/access_www.log
192.168.3.101 - - [08/Apr/2019:00:36:06 +0800] "GET / HTTP/1.0" 200 14 "-" "curl/7.29.0" "192.168.3.102"
192.168.3.101 - - [08/Apr/2019:00:36:07 +0800] "GET / HTTP/1.0" 200 14 "-" "curl/7.29.0" "192.168.3.102"
```

案例六：根据不同的URI 调度到不同的服务器

常见架构图



梳理：1、当用户请求 www.etiantian.org/upload/xx 地址时，实现由upload上传服务器池处理请求；2、当用户请求 www.etiantian.org/static/xx 地址时，实现由静态服务器池处理请求；3、除此之外，对于其它访问请求，全部由默认的动态服务器池处理请求。如下图：

用户请求的（URI）	负责处理的服务器	功能
/upload	192.168.3.103:80 （web1）	upload 服务器
/static	192.168.3.104:80 （web2）	static 静态服务器
/	192.168.3.104:8080 （web2）	默认

在wab服务器上面准备测试文件

```
//准备测试文件web1的80 upload
[root@centos7-3 ~]# mkdir /application/nginx/html/www/upload
[root@centos7-3 ~]# echo "upload web01 192.168.3.103 " > /application/nginx/html/www/upload/index.html
//准备测试文件web2的80 static
[root@centos7-4 ~]# mkdir /application/nginx/html/www/static
[root@centos7-4 ~]# echo "static web02 192.168.3.104 " > /application/nginx/html/www/static/index.html
//准备测试文件web2的8080 default
[root@centos7-4 ~]# mkdir /application/nginx/www_8080
[root@centos7-4 ~]# vim /etc/nginx/conf.d/www_8080.conf
server {
    listen          80;
    server_name     localhost;
    access_log      /var/log/nginx/access.log main;
    location / {
        root /application/nginx/www_8080;
        index index.html index.htm;
    }
}
[root@centos7-4 ~]# echo "default web02 192.168.3.104 " > /application/nginx/www_8080/index.html

//在lb1服务器测试后端web服务器是否能够正常访问
```

```
[root@centos7-1 ~]#curl 192.168.3.103/upload/index.html
upload web01 192.168.3.103
[root@centos7-1 ~]# curl 192.168.3.104/static/index.html
static web02 192.168.3.104
[root@centos7-1 ~]# curl 192.168.3.104:8080/index.html
default web02 192.168.3.104
```

配置LB，修改lb1配置文件进行配置

```
sendfile        on;
keepalive_timeout  65;

upstream upload_pools {
    server 192.168.3.103:80;
 }
upstream static_pools {
    server 192.168.3.104:80;
 }
upstream default_pools {
    server 192.168.3.104:8080;
 }

server {
    listen       80;
    server_name  www.etiantian.org;
location / {
        proxy_pass http://default_pools;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $remote_addr;
    }
location /upload {
        proxy_pass http://upload_pools;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $remote_addr;
```

```
    }
location /static {
        proxy_pass http://static_pools;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $remote_addr;
    }
}
```

在lb2(模拟客户端)节点上面进行测试，结果可以看到根据不同的url调度到不同的服务器上面了。

```
[root@centos7-2 ~]# curl 192.168.3.101
default web02 192.168.3.104
[root@centos7-2 ~]# curl 192.168.3.101/upload/index.html
upload web01 192.168.3.103
[root@centos7-2 ~]# curl 192.168.3.101/static/index.html
static web02 192.168.3.104
```