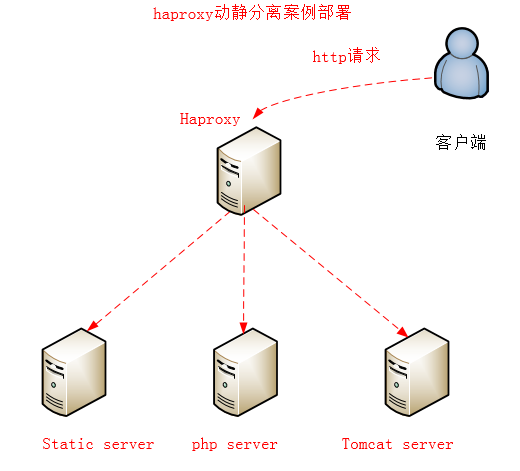
# HAproxy指南之haproxy实现动静分离（案例篇）

 实际应用环境中，往往需要根据业务请求将相关不同请求跳转到指定的后端server，比如客户静态资源请求交给静态资源server处理，php请求交给php server处理，jsp请求交给tomcat处理，即业务上的应用请求分离，而haproxy完全可以利用acl匹配规则实现这一目的 。

一. haproxy实现应用动静分离

如图所示为整体的拓扑图：

[](http://s2.51cto.com/wyfs02/M02/7D/77/wKioL1bpHNWglhhsAACZOKiqvAw447.png)

**需求说明:**

当客户端访问haproxy时，请求的是静态文件内容时，请求转交给static server，请求的是php内容时，请求转交给php server，请求的是jsp内容时，请求转交给tomcat server，以实现动静分离

**一.部署前说明：**

(1)系统版本: centos 6.6（64位）

(2)角色及ip相关信息:

|  |  |
| --- | --- |
| 角色名称 | ip信息 |
| haproxy server | eth0:172.51.96.233/24  &&  eth1:192.168.0.233/24 |
| static server | eth1:192.168.0.247/24 |
| php server | eth1:192.168.0.235/24 |
| tomcat server | eth1:192.168.0.238/24 |

**二. 部署操作**

haproxy server上操作

编译安装haproxy

1.1 到haproxy官网下载haproxy源码包如下

**cd ~  
wget http://www.haproxy.org/download/1.5/src/haproxy-1.5.15.tar.gz**

1.2 创建haproxy运行用户

**groupadd -r haproxy   
useradd -g haproxy -M -s /sbin/nologin haproxy**

1.3 编译安装haproxy：

**cd ~  
tar zxvf haproxy-1.5.15.tar.gz -C /usr/local/src  
cd /usr/local/src/haproxy-1.5.15  
make TARGET=linux26  PREFIX=/usr/local/haproxy  
make install PREFIX=/usr/local/haproxy**

注意：TARGET=Linux26 是通过uname -a 来查看Linux内核版本的

1.4 创建haproxy主配置文件：

**mkdir /etc/haproxy/  
touch /etc/haproxy/haproxy.cfg**

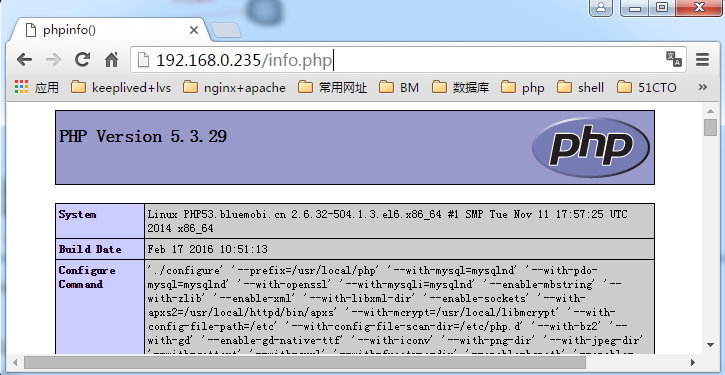
后端web server上操作

1.5 分别在img server，php server，tomcat server安装相应的web环境并创建测试页，其中:

(1)static server的访问url为:<http://192.168.0.247/img/haproxy.PNG>  ，页面内容如下:

[](http://s4.51cto.com/wyfs02/M00/7D/7B/wKiom1bpJwzgq1eBAABzj4f3jhk518.png)

(2)php server的访问url为:<http://192.168.0.235/info.php>  ，页面内容如下:

[](http://s2.51cto.com/wyfs02/M00/7D/7C/wKiom1bpKDPAer7fAADfw1EOkFI770.png)

(3)tomcat server的访问url为:[http://192.168.238:8086/test.jsp](http://blief.blog.51cto.com/test.jsp)  ，页面内容如下:

[](http://s1.51cto.com/wyfs02/M02/7D/7C/wKiom1bpKUnB8SndAAF-aSVn-pk548.png)

1.6 编辑haproxy server的haproxy主配置文件：

代码内容如下

#---------------------------------------------------------------------

# Global settings

#---------------------------------------------------------------------

global

        log 127.0.0.1   local3

        maxconn 204800

        chroot /usr/local/haproxy

        user  haproxy

        group haproxy

        daemon

        nbproc 1

        pidfile /var/run/haproxy.pid

        stats socket /usr/local/haproxy/stats

        description haproxy server

#---------------------------------------------------------------------

# common defaults that all the 'listen' and 'backend' sections will

# use if not designated in their block

#---------------------------------------------------------------------

defaults

        log     global

        mode    http

        maxconn 10000

        option  httplog

        option  httpclose

        option  dontlognull

        option  forwardfor      except 127.0.0.0/8

        retries 3

        option redispatch

        option abortonclose

        balance roundrobin

        timeout http-request    10s

        timeout queue           1m

        timeout connect         10s

        timeout client          1m

        timeout server          1m

        timeout http-keep-alive 10s

        timeout check           10s

#---------------------------------------------------------------------

# use listen setting the haproxy status for site

#---------------------------------------------------------------------

listen admin\_status     #设置haproxy监控状态

        bind \*:3030

        mode http

        log 127.0.0.1 local3 err

        stats refresh 5s

        stats uri /status     #监控状态页面访问url

        stats realm www.skeryp.com

        stats auth admin:admin

        stats hide-version

        stats admin if TRUE

#---------------------------------------------------------------------

# main listen which proxys to the backends

#---------------------------------------------------------------------

listen  www

        bind \*:80

        maxconn 5000

        mode http

        log global

        option httplog

        option httpclose

        option forwardfor

        log         global

        default\_backend default   #设置默认访问页面

        #定义当请求的内容是静态内容时，将请求转交给static server的acl规则

        acl url\_static path\_beg  -i /static /images /img /javascript /stylesheets

        acl url\_static path\_end  -i .jpg .gif .png .css .js .html

        acl host\_static hdr\_beg(host)  -i img. video. download. ftp. imags. videos.

        #定义当请求的内容是php内容时，将请求转交给php server的acl规则

        acl url\_php path\_end     -i .php

        #定义当请求的内容是.jsp或.do内容时，将请求转交给tomcat server的acl规则

        acl url\_jsp path\_end     -i .jsp .do

        #引用acl匹配规则

        use\_backend static\_pool if  url\_static or host\_static

        use\_backend php\_pool    if  url\_php

        use\_backend tomcat\_pool if  url\_jsp

        #定义后端backend server

 backend static\_pool

        option  httpchk GET /index.html

        server static1 192.168.0.247:80 cookie id1  check inter 2000 rise 2 fall 3

 backend php\_pool

        option  httpchk GET /info.php

        server php1 192.168.0.235:80 cookie id1 check inter 2000 rise 2 fall 3

 backend tomcat\_pool

        option  httpchk GET /index.jsp

        server tomcat1 192.168.0.238:8086 cookie id2 check inter 2000 rise 2 fall 3

#<----------------------default site for listen and frontend------------------------------------>

backend default

        mode http

        option  httpchk GET /index.html

        server default 192.168.0.127:80 cookie id1 check inter 2000 rise 2 fall 3 maxconn 5000

注意:

上面的www的配置部分也可以用frontend配置块来替换，如下所示:

frontend www

        bind \*:80

        maxconn 5000

        mode http

        log global

        option httplog

        option httpclose

        option forwardfor

        log         global

        default\_backend default

        acl url\_static path\_beg  -i /static /images /img /javascript /stylesheets

        acl url\_static path\_end  -i .jpg .gif .png .css .js .html

        acl host\_static hdr\_beg(host)  -i img. video. download. ftp. imags. videos.

        acl url\_php path\_end     -i .php

        acl url\_jsp path\_end     -i .jsp .do

        use\_backend static\_pool if  url\_static or host\_static

        use\_backend php\_pool    if  url\_php

        use\_backend tomcat\_pool if  url\_jsp

backend static\_pool

        option  httpchk GET /index.html

        server static1 192.168.0.247:80 cookie id1  check inter 2000 rise 2 fall 3

 backend php\_pool

        option  httpchk GET /info.php

        server php1 192.168.0.235:80 cookie id1 check inter 2000 rise 2 fall 3

 backend tomcat\_pool

        option  httpchk GET /index.jsp

        server tomcat1 192.168.0.238:8086 cookie id2 check inter 2000 rise 2 fall 3

备注:  listen配置块是frontend和backend的组合体，listen里面可以单独配置backend不配置frontend，也可以组合使用，即listen配置区域可以交差使用frontend和backend的配置,如acl可以配置到frontend块， 也可以直接配置到listen块，但是不能配置到backend块。如backend的中server可以直接配置到listen配置区域，但不能直接配置到frontend配置区域

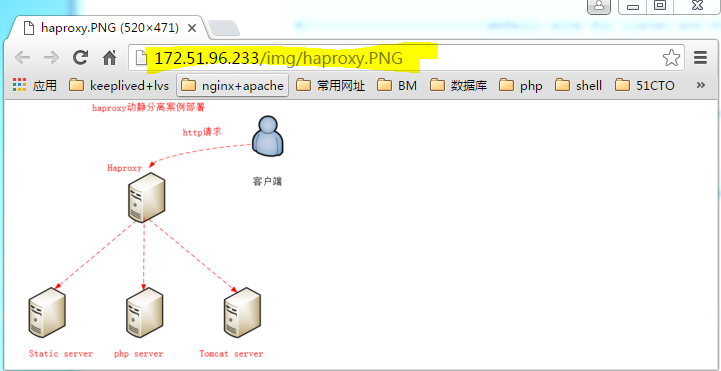
1.7 重启haproxy服务：

server haproxy restart

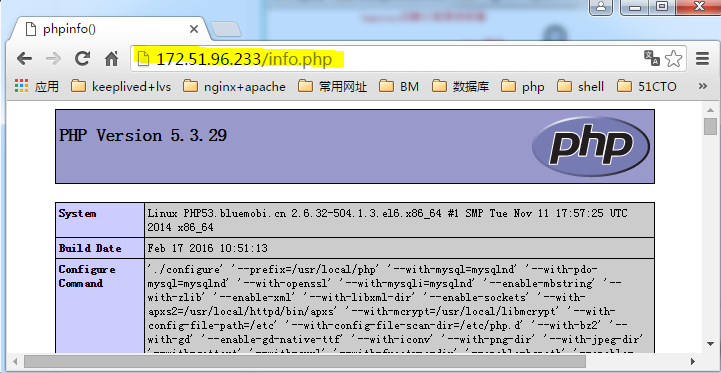
关于haproxy服务脚本代码请访问:<http://blief.blog.51cto.com/6170059/1750573>

**三. 结论测试**

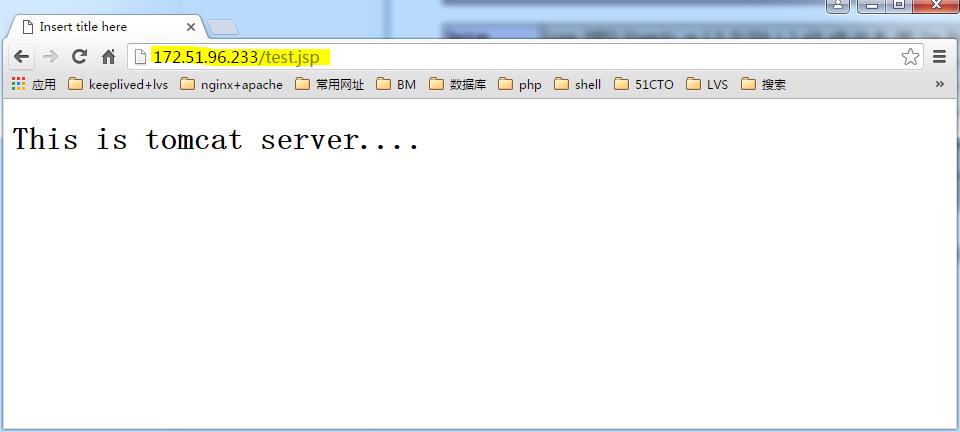
在客户端访问:<http://172.51.96.233/img/haproxy.PNG>，可以发现haproxy将请求转交给后端static server了。如下:

[](http://s4.51cto.com/wyfs02/M02/7D/87/wKioL1bqVhuBmE_tAAEAqibq8ks375.png)

访问:<http://172.51.96.233/info.php>，可以发现haproxy将请求转交给后端php server了。如下:

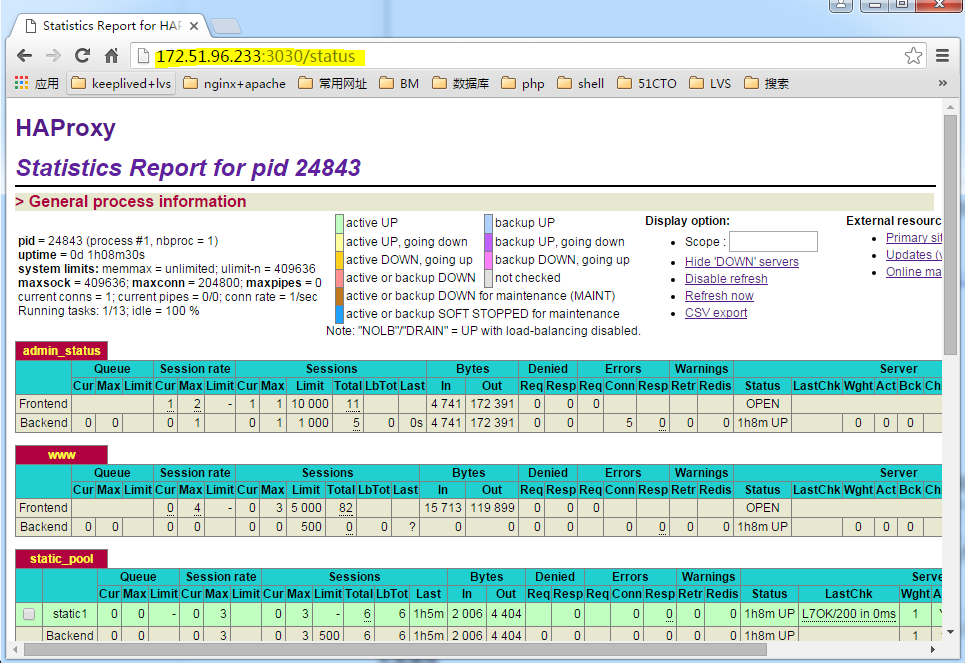
[](http://s2.51cto.com/wyfs02/M01/7D/8A/wKiom1bqVlDD5PigAADp0MfZP7Y109.png)

访问:<http://172.51.96.233/test.jsp>，可以发现haproxy将请求转交给后端tomcat server了。如下所示:

[](http://s2.51cto.com/wyfs02/M01/7D/8A/wKiom1bqVlDD5PigAADp0MfZP7Y109.png)

从上可知：haproxy已经成功实现了动静分离，即静态内容交由静态server处理，动态内容交由动态处理的server处理

如果我们要查看各个server的健康状态，可以登录:<http://172.51.96.233:3030/admin>,如下

[](http://s3.51cto.com/wyfs02/M01/7D/8A/wKioL1bqZueyyZFVAAHz2SWvE_s838.png)

总结:haproxy可以利用acl规则匹配url做相应的请求跳转，比如动静分离，域名跳转等等应用需求，haproxy是一款性能很强大的四层以及七层代理server。