

烂泥行天下

烂泥：起于尘土，翱翔于九天！！！ 烂泥行天下<http://www.ilanni.com>@秀依  
林枫<http://xiuylf.taobao.com>

博客园

首页

新随笔

联系

订阅

管理

随笔 - 135 文章 - 4 评论 - 32

烂泥：起于尘土，翱翔于九天！  
昵称：烂泥行天下  
园龄：9年8个月  
粉丝：76  
关注：7  
+加关注

< 2019年1月 >						
日	一	二	三	四	五	六
30	31	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31	1	2
3	4	5	6	7	8	9

搜索

找找看

谷歌搜索

常用链接

- 我的随笔
- 我的评论
- 我的参与
- 最新评论
- 我的标签

我的标签

- 烂泥(117)
- 安装(23)
- KVM(19)
- 学习(18)
- 配置(17)
- ubuntu(13)
- centos(13)
- 服务器(12)
- mysql(11)
- 使用(10)

烂泥：高负载均衡学习haproxy之安装与配置

本文由秀依林枫提供友情赞助，首发于烂泥行天下

有关高负载均衡的软件，目前使用比较多的是haproxy、nginx和lvs。下面我们就开始学习haproxoy这款软件。

一、haproxy介绍

以下开始介绍有关haproxy的原理及其优点。

1.1、haproxy原理

haproxy提供高可用性、负载均衡以及基于TCP(第四层)和HTTP（第七层）应用的代理，支持虚拟主机，它是免费、快速并且可靠的一种解决方案。

haproxy特别适用于那些负载特别大的web站点，这些站点通常又需要会话保持或七层处理。haproxy运行在时下的硬件上，完全可以支持数以万计的并发连接，并且它的运行模式使得它可以很简单安全的整合进您当前的架构中，同时可以保护你的web服务器不被暴露到网络上。

haproxy实现了一种事件驱动、单一进程模型，此模型支持非常大的并发连接数。多进程或多线程模型受内存限制、系统调度器限制以及无处不在的锁限制，很少能处理数千并发连接。

事件驱动模型因为在有更好的资源和时间管理的用户端(User-Space)实现所有这些任务，所以没有这些问题。此模型的弊端是，在多核系统上，这些程序通常扩展性较差。这就是为什么他们必须进行优化以使每个CPU时间片(Cycle)做更多的工作。

1.2、haproxy的优点

- (1) 免费开源，稳定性也是非常好。单haproxy也跑得不错，稳定性可以与硬件级的F5相媲美。
- (2) 根据官方文档，haproxy可以跑满10Gbps，这个数值作为软件级负载均衡器是相当惊人的。
- (3) haproxy支持连接拒绝:因为维护一个连接的打开的开销是很低的，有时我们很需要限制攻击蠕虫（attack bots），也就是说限制它们的连接打开从而限制它们的危害。这个已经为一个陷于小型DDoS攻击的网站开发了而且已经拯救了很多站点，这个优点也是其它负载均衡器没有的。
- (4) haproxy支持全透明代理（已具备硬件防火墙的典型特点）:可以用客户端IP地址或者任何其他地址来连接后端服务器。这个特性仅在Linux 2.4/2.6内核打了tcp proxy补丁后才可以使使用。这个特性也使得为某特殊服务器处理部分流量同时又不修改服务器的地址成为可能。
- (5) haproxy现多于线上的Mysql集群环境，我们常用于它作为MySQL（读）负载均衡。
- (6) 自带强大的监控服务器状态的页面，实际环境中我们结合Nagios进行邮件或短信报警。
- (7) HAProxy支持虚拟主机，许多朋友说它不支持虚拟主机是错误的，通过测试我们知道，HAProxy是支持虚拟主机的。

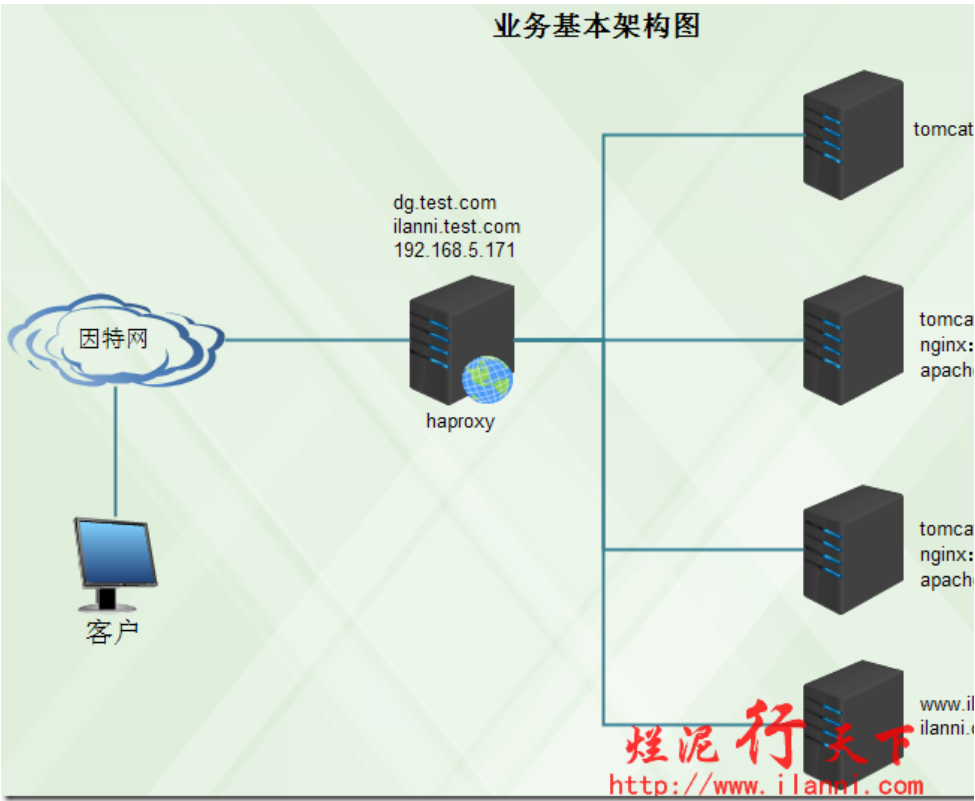
PS：本次实验的OS为ubuntu server 14.04。

二、业务架构图

现在我们以实际的业务架构图，来使用haproxy。业务架构图如下：

更多
随笔分类(133)
Linux(82)
实战(17)
数据库(11)
虚拟化(23)
随笔档案(135)
2016年12月 (7)
2016年5月 (1)
2016年4月 (2)
2016年3月 (3)
2016年2月 (1)
2016年1月 (2)
2015年11月 (4)
2015年10月 (2)
2015年9月 (6)
2015年8月 (6)
2015年7月 (3)
2015年6月 (1)
2015年5月 (3)
2015年3月 (4)
2015年2月 (3)
2015年1月 (8)
2014年12月 (5)
2014年11月 (10)
2014年10月 (7)
2014年9月 (16)
2014年8月 (25)
2014年7月 (14)
2012年9月 (2)

文章档案(4)
---------



这个是基本的业务架构图，对外是haproxy这台服务器。目前暂时没有考虑haproxy的单点故障问题，这个问题我们会在后续的keepalived文章中会进行介绍。

现在要求如下：

2.1、域名跳转

客户端访问http://dg.test.com时，要把请求分发到192.168.5.171:8080、192.168.5.174:8080、192.168.5.178:8080，这三台服务器上。

客户端访问http://ilanni.test.com时，要把请求分发到ilanni.com，这台服务器上。

2.2、IP地址跳转

客户端访问http://192.168.5.171时，要把请求分发到192.168.5.174:80、192.168.5.178:80这两台服务器上。同时还要求客户端每一次访问，都跳转到不同的服务器上。

2.3、端口跳转

客户端访问http://dg.test.com:8090和http://ilanni.test.com:8090这两个地址时，要把请求分发到192.168.5.174:8090、192.168.5.178:8090，这两台服务器上。

2.4、默认跳转

如果客户端访问的不是dg.test.com与192.168.5.171，这两个地址的话，要把请求全部分发到192.168.5.178:8080上。

2.5、多ACL匹配

如果客户端的IP是192.168.5.140，同时访问的是http://192.168.5.171时，要把请求分发到www.ilanni.com上。

三、安装haproxy

haproxy的官网是<http://www.haproxy.org/>，如果打不开此站点，请FQ（[如何FQ](#)，[联系我哦](#)）。haproxy的安装我们可以分为源码方式和apt-get方式。下面对其安装方式进行一一讲解。

3.1 源码方式安装haproxy

haproxy目前最新的版本为1.6，为了业务的稳定，在此我们选择的是1.3.15.27这个版本。在进行源码安装之前，首先要安装相关的软件库。如下：

```
sudo apt-get -y install make gcc
```

2014年10月 (1)
2014年7月 (1)
2014年5月 (1)
2012年9月 (1)

最新评论

1. Re:烂泥：ubuntu 14.04搭建OpenV PN服务器

你是精神病么，图上面整的乱七八糟的，谁还能盗你的图？

--鯨

2. Re:烂泥：ubuntu下vsftpd虚拟用户配置

博主这篇博客写的非常完美。但是有一个问题我不太明白，我采用centos6.x配置vsftpd虚拟账户的操作，在ubuntu上完全可行。我采用的是深度操作系统，部署的，应该没啥区别。博主有空可以再试试看.....

--运维笔记

3. Re:烂泥：U盘安装Centos6.5

66666

--规格严格-功夫到家

4. Re:烂泥：redis3.2.3安装与配置

资深老玩家，666

--从未太晚

5. Re:烂泥：jira7.2安装、中文及破解

运行一段时间后重启，会出现An error occurred when trying to parse the version information JSON object版本信息出错。无法启动，怎.....

--羽之

阅读排行榜

- 1. 烂泥：jira7.2安装、中文及破解(14987)
- 2. 烂泥：学习ubuntu远程桌面（一）：配置远程桌面(14598)
- 3. 烂泥：Postfix邮件服务器搭建之软件安装与配置(14300)

```
ilanni@ubuntu-8:~$ sudo apt-get -y install make gcc
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  binutils cpp cpp-4.9 gcc-4.9 gcc-4.9-base libasan1 libatomic1 libc-dev-bin
  libc6 libc6-dev libcilkrts5 libcloog-is14 libgcc-4.9-dev libgcc1 libgmp10
  libgomp1 libisl10 libitm1 liblsan0 libmpc3 libmpfr4 libquadmath0 libtsan0
  libubsan0 linux-libc-dev manpages-dev
Suggested packages:
  binutils-doc cpp-doc gcc-4.9-locales gcc-multilib autoconf automake libtool
  flex bison gdb gcc-doc gcc-4.9-multilib gcc-4.9-doc libgcc1-dbg libgomp1-dbg
  libitm1-dbg libatomic1-dbg libasan1-dbg liblsan0-dbg libtsan0-dbg
  libubsan0-dbg libcilkrts5-dbg libquadmath0-dbg glibc-doc make-doc
The following NEW packages will be installed:
```

创建运行haproxy时，使用的用户。在此我们使用haproxy这个用户，而且此用户不能登录到系统。如下：

sudo useradd -m haproxy

cat /etc/passwd |grep haproxy

```
ilanni@ubuntu-8:~$ sudo useradd -m haproxy
ilanni@ubuntu-8:~$ cat /etc/passwd |grep haproxy
haproxy:x:1005:1005::/home/haproxy:/bin/bash
ilanni@ubuntu-8:~$
```

通过上图，我们可以看到haproxy用户及用户组的ID均为1005。

现在开始下载源码包如下：

wget <http://www.haproxy.org/download/1.3/src/haproxy-1.3.15.27.tar.gz>

解压源码包，如下：

tar -xf haproxy-1.3.15.27.tar.gz

cd haproxy-1.3.15.27/

```
ilanni@ubuntu-8:~$ tar -xf haproxy-1.3.27.tar.gz
ilanni@ubuntu-8:~$ cd haproxy-1.3.27/
ilanni@ubuntu-8:~/haproxy-1.3.27$ ll
total 200
drwxr-xr-x 8 ilanni ilanni 4096 Feb 1 2015 ./
drwxr-xr-x 4 ilanni ilanni 4096 Aug 13 15:39 ../
-rw-r--r-- 1 ilanni ilanni 82339 Feb 1 2015 CHANGELOG
drwxr-xr-x 5 ilanni ilanni 4096 Feb 1 2015 contrib/
-rw-r--r-- 1 ilanni ilanni 1880 Feb 1 2015 CONTRIB
drwxr-xr-x 4 ilanni ilanni 4096 Feb 1 2015 doc/
drwxr-xr-x 3 ilanni ilanni 4096 Feb 1 2015 examples/
-rw-r--r-- 1 ilanni ilanni 132 Feb 1 2015 .gitignore
drwxr-xr-x 6 ilanni ilanni 4096 Feb 1 2015 include/
-rw-r--r-- 1 ilanni ilanni 1767 Feb 1 2015 LICENSE
-rw-r--r-- 1 ilanni ilanni 19558 Feb 1 2015 Makefile
-rw-r--r-- 1 ilanni ilanni 4409 Feb 1 2015 Makefile.bsd
-rw-r--r-- 1 ilanni ilanni 4460 Feb 1 2015 Makefile.osx
-rw-r--r-- 1 ilanni ilanni 4304 Feb 1 2015 README
-rw-r--r-- 1 ilanni ilanni 4139 Feb 1 2015 ROADMAP
drwxr-xr-x 2 ilanni ilanni 4096 Feb 1 2015 src/
```

查看haproxy的安装文件，如下：

more README

4. 烂泥：高负载均衡学习haproxy之安装与配置(14111)
5. 烂泥：ubuntu 14.04搭建OpenVPN服务器(11165)

- 评论排行榜
1. 烂泥：KVM虚拟机Linux系统增加硬盘(5)
2. 烂泥：学习ubuntu远程桌面（一）：配置远程桌面(4)
3. 烂泥：Linux源码包制作RPM包之Apache(3)
4. 烂泥：nginx、php-fpm、mysql用户权限解析(3)
5. 烂泥：jira.7.2安装、中文及破解(2)

- 推荐排行榜
1. 烂泥：高负载均衡学习haproxy之安装与配置(4)
2. 烂泥：学习ubuntu远程桌面（一）：配置远程桌面(3)
3. 烂泥：zabbix3.0安装与配置(2)
4. 烂泥：Postfix邮件服务器搭建之软件安装与配置(2)
5. 烂泥：Postfix邮件服务器搭建之虚拟用户配置(2)

```
ilanni@ubuntu-8:~/haproxy-1.3.27$ more README
-----
H A - P r o x y
How to build it
-----
version 1.3.27
willy tarreau
2015/02/01

To build haproxy, you will need :
- GNU make. Neither Solaris nor OpenBSD's make work with this makefile.
  However, specific Makefiles for BSD and OSX are provided.
- GCC between 2.91 and 4.3. Others may work, but not tested.
- GNU ld

Also, you might want to build with libpcre support, which will provide a very
efficient regex implementation and will also fix some badness on Solaris's one.

To build haproxy, you have to choose your target OS amongst the following ones
and assign it to the TARGET variable :

- linux22      for Linux 2.2
- linux24      for Linux 2.4 and above (default)
- linux24e     for Linux 2.4 with support for a working epoll (> 0.21)
- linux26      for Linux 2.6 and above
- solaris      for Solaris 8 or 10 (others untested)
- freebsd     for FreeBSD 5 to 6.2 (others untested)
- openbsd     for OpenBSD 3.1 to 3.7 (others untested)
- cygwin      for Cygwin
```

这张图很明显的告诉我们，编译安装haproxy需要的软件库。除此之外还需要选择OS的内核版本，linux2.6以上的版本，我们都选择linux26。

```
In order to build a 32-bit binary on an x86_64 Linux system :

$ make TARGET=linux26 ARCH=i386

If you need to pass other defines, includes, libraries, etc., then please
check the Makefile to see which ones will be available in your case, and
use the USE_* variables in the GNU Makefile, or ADDING_DEFINES, or DEFINES.
```

通过查看安装文档，我们可以很清楚的看出要编译haproxy，我们首先要知道OS内核版本以及OS的位数。如下：

```
uname -a
ilanni@ubuntu-8:~/haproxy-1.3.27$ uname -a
Linux ubuntu-8 3.16.0-30-generic #40~14.04.1-Ubuntu SMP Thu Jan 15 17:43:14 UTC 2015 x86_64 x86_64 GNU/Linux
```

通过上图，我们可以很容易的看出linux内核的版本与OS的位数。

现在开始编译haproxy，如下：

```
make TARGET=linux26 ARCH=x86_64 PREFIX=/usr/local/haproxy
ilanni@ubuntu-8:~/haproxy-1.3.27$ make TARGET=linux26 ARCH=x86_64 PREFIX=/usr/local/haproxy
gcc -Iinclude -Wall -m64 -march=x86-64 -O2 -g -DPROXY -DENABLE_POLL -DENABLE_EPOLL -DDEFINITE
TE="2015/02/01" \
  -DBUILD_TARGET="linux26" \
  -DBUILD_ARCH="x86_64" \
  -DBUILD_CPU="generic" \
  -DBUILD_CC="gcc" \
  -DBUILD_CFLAGS="-m64 -march=x86-64 -O2 -g" \
  -DBUILD_OPTIONS="" \
  -c -o src/haproxy.o src/haproxy.c
src/haproxy.c: In function 'main':
```

其中TARGET表示OS的内核版本，ARCH表示OS的位数，PREFIX表示haproxy的安装路径。

现在开始安装haproxy，如下：

```
sudo make install PREFIX=/usr/local/haproxy
```

```
ilanni@ubuntu-8:~/haproxy-1.3.27$ sudo make install PREFIX=/usr/local/haproxy
install -d /usr/local/haproxy/sbin
install haproxy /usr/local/haproxy/sbin
install -d /usr/local/haproxy/share/man/man1
install -m 644 doc/haproxy.1 /usr/local/haproxy/share/man/man1
install -d /usr/local/haproxy/doc/haproxy
for x in configuration architecture haproxy-en haproxy-fr; do
    install -m 644 doc/$x.txt /usr/local/haproxy/doc/haproxy/$x
done
ilanni@ubuntu-8:~/haproxy-1.3.27$
```

烂泥行天下  
http://www.ilanni.com

查看安装后的文件，如下：

`ll /usr/local/haproxy/`

```
ilanni@ubuntu-8:~/haproxy-1.3.27$ ll /usr/local/haproxy/
total 20
drwxr-xr-x  5 root root 4096 Aug 13 16:20 ./
drwxr-xr-x 11 root root 4096 Aug 13 16:20 ../
drwxr-xr-x  3 root root 4096 Aug 13 16:20 doc/
drwxr-xr-x  2 root root 4096 Aug 13 16:20 sbin/
drwxr-xr-x  3 root root 4096 Aug 13 16:20 share/
ilanni@ubuntu-8:~/haproxy-1.3.27$
```

烂泥行天下  
http://www.ilanni.com

编辑haproxy的配置文件，haproxy默认给我们提供一个配置文件模版。如下：

`sudo cp examples/haproxy.cfg /usr/local/haproxy/`

```
ilanni@ubuntu-8:~/haproxy-1.3.27$ sudo cp examples/haproxy.cfg /usr/local/haproxy/
ilanni@ubuntu-8:~/haproxy-1.3.27$ ll /usr/local/haproxy/
total 24
drwxr-xr-x  5 root root 4096 Aug 13 16:27 ./
drwxr-xr-x 11 root root 4096 Aug 13 16:20 ../
drwxr-xr-x  3 root root 4096 Aug 13 16:20 doc/
-rw-r--r--  1 root root 2366 Aug 13 16:27 haproxy.cfg
drwxr-xr-x  2 root root 4096 Aug 13 16:20 sbin/
drwxr-xr-x  3 root root 4096 Aug 13 16:20 share/
ilanni@ubuntu-8:~/haproxy-1.3.27$
```

烂泥行天下  
http://www.ilanni.com

查看haproxy的版本，如下：

`/usr/local/haproxy/sbin/haproxy -v`

```
ilanni@ubuntu-8:~/haproxy-1.3.27$ /usr/local/haproxy/sbin/haproxy -v
HA-Proxy version 1.3.27 2015/02/01
Copyright 2000-2015 Willy Tarreau <w@1wt.eu>
ilanni@ubuntu-8:~/haproxy-1.3.27$
```

烂泥行天下  
http://www.ilanni.com

通过上图，可以看到haproxy的版本确实是1.3.27。有关haproxy的配置实例见第三章。

### 3.2 apt-get方式安装haproxy

apt-get方式安装haproxy，如下：

`sudo apt-get -y install haproxy`

```
ilanni@ubuntu-8:~$ sudo apt-get -y install haproxy
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  init-system-helpers
Suggested packages:
  vim-haproxy haproxy-doc
The following NEW packages will be installed:
  haproxy
The following packages will be upgraded:
  init-system-helpers
```

烂泥行天下  
http://www.ilanni.com

查看haproxy安装的文件

`dpkg -L haproxy`

```
ilanni@ubuntu-8:~$ dpkg -L haproxy
./
./lib
./lib/systemd
./lib/systemd/system
./lib/systemd/system/haproxy.service
./usr
./usr/lib
./usr/lib/tmpfiles.d
./usr/lib/tmpfiles.d/haproxy.conf
./usr/share
./usr/share/doc
./usr/share/doc/haproxy
./usr/share/man/man1
./usr/share/man/man1/halog.1.gz
./usr/share/man/man1/haproxy.1.gz
./usr/share/lintian
./usr/share/lintian/overrides
./usr/share/lintian/overrides/haproxy
```

安装完毕后, haproxy默认已经启动。如下:

ps -ef |grep haproxy

```
ilanni@ubuntu-8:~$ ps -ef |grep haproxy
haproxy 1600 1 0 17:20 ? 00:00:00 /usr/sbin/haproxy -f /etc/haproxy/haproxy.cfg -D
ilanni 1616 1237 0 17:29 pts/0 00:00:00 grep --color=auto haproxy
ilanni@ubuntu-8:~$
```

烂  
http

查看haproxy版本,如下:

haproxy -v

```
ilanni@ubuntu-8:~$ haproxy -v
HA-Proxy version 1.5.4 2014/09/02
Copyright 2000-2014 Willy Tarreau <w@1t.eu>
ilanni@ubuntu-8:~$
```

通过上图, 我们知道现在haproxy的版本为1.5.4。

注意: apt-get方式安装haproxy, 如果版本为1.4.24的话。我们一定要修改/etc/default/haproxy, 如下:

sudo vi /etc/default/haproxy

ENABLED=1

```
wangxy@VM-12-163-ubuntu:~$ cat /etc/default/haproxy
# Set ENABLED to 1 if you want the init script to start haproxy.
ENABLED=1
# Add extra flags here.
#EXTRA_OPTS="-de -m 16"
wangxy@VM-12-163-ubuntu:~$
```

如果不修改的话, 使用haproxy启动脚本的话, 是没有用处的, 也就是说脚本不会重新加载haproxy的配置。

查看haproxy的配置文件,如下:

cat /etc/haproxy/haproxy.cfg

```
ilanni@ubuntu-8:~$ cat /etc/haproxy/haproxy.cfg
global
    log /dev/log      local0
    log /dev/log      local1 notice
    chroot /var/lib/haproxy
    stats socket /run/haproxy/admin.sock mode 660 level admin
    stats timeout 30s
    user haproxy
    group haproxy
    daemon

    # Default SSL material locations
    ca-base /etc/ssl/certs
    crt-base /etc/ssl/private

    # Default ciphers to use on SSL-enabled listening sockets.
    # For more information, see ciphers(1SSL).
    ssl-default-bind-ciphers kEECDH+aRSA+AES:kRSA+AES+AESGCM:RC4+SHA+!SSLv2
```

通过上图, 我们可以很明显的看出默认的配置文件中是没有内容的。



#### 四、配置haproxoy

haproxy安装完毕后，我们来配置haproxy。源码安装的haproxoy在前面我们已经讲解了，haproxy提供的配置模版haproxy.cfg。

##### 4.1、haproxy配置实例

我们现在就以这个模版为例，配置haproxy。haproxy配置文件内容，如下：

```
grep -vE "^#|^$" haproxy.cfg

global

log 127.0.0.1 local0

log 127.0.0.1 local1 notice

maxconn 4096

uid 1005

gid 1005

daemon

defaults

log global

mode http

option httplog

option dontlognull

retries 3

option redispatch

maxconn 2000

timeout 5000

clitimeout 50000

srvtimeout 50000

listen admin_stats

bind 192.168.5.171:1080

mode http

option httplog

maxconn 10

stats refresh 30s

stats uri /stats

stats auth admin:admin

stats hide-version

frontend weblb

bind *:80

acl is_dg hdr_beg(host) dg.test.com

acl is_ilanni hdr_beg(host) ilanni.test.com

acl is_171 hdr_beg(host) 192.168.5.171

acl is_ip src 192.168.5.140

use_backend acl if is_171 is_ip

use_backend dgserver if is_dg

use_backend ilanni if is_ilanni

use_backend 171server if is_171

default_backend backend_default

backend dgserver
```

balance source

server web1 192.168.5.171:8080 maxconn 1024 weight 3 check inter 2000 rise 2 fall 3

server web2 192.168.5.174:8080 maxconn 1024 weight 3 check inter 2000 rise 2 fall 3

server web3 192.168.5.178:8080 maxconn 1024 weight 3 check inter 2000 rise 2 fall 3

backend 171server

balance roundrobin

server dg1 192.168.5.174:80 check

server dg2 192.168.5.178:80 check

backend ilanni

server web1 www.yuanbaopu.com:80 weight 3 check inter 2000 rise 2 fall 3

backend acl

balance source

server web1 www.ilanni.com:80 maxconn 1024 weight 3 check inter 2000 rise 2 fall 3

backend backend\_default

server web1 192.168.5.178:8080 weight 3 check inter 2000 rise 2 fall 3

listen 8090

bind 0.0.0.0:8090

mode http

balance roundrobin

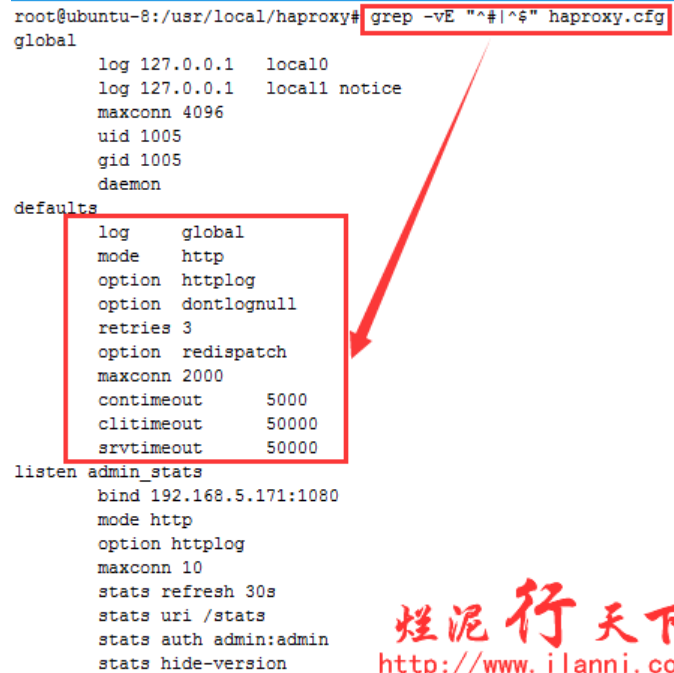
server web1 192.168.5.174:8090 maxconn 1024 weight 5 check inter 2000 rise 2 fall 3

server web2 192.168.5.178:8090 maxconn 1024 weight 3 check inter 2000 rise 2 fall 3

```
root@ubuntu-8:/usr/local/haproxy# grep -vE "^#|^$" haproxy.cfg
global
    log 127.0.0.1    local0
    log 127.0.0.1    local1 notice
    maxconn 4096
    uid 1005
    gid 1005
    daemon

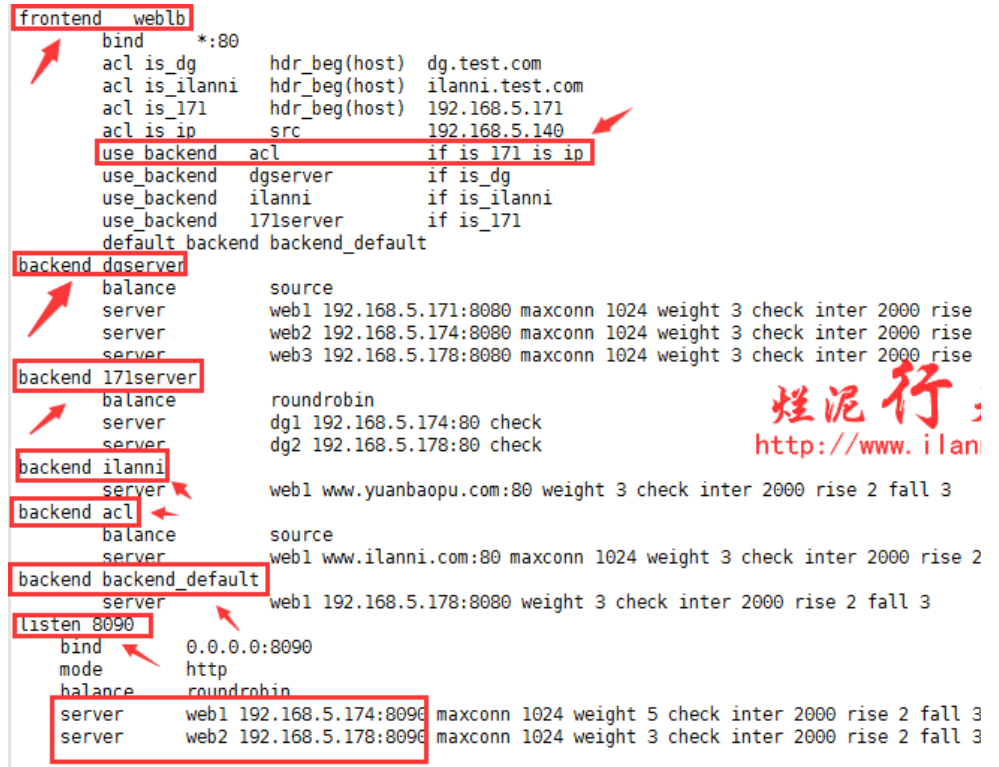
defaults
    log          global
    mode         http
    option        httplog
    option        dontlognull
    retries      3
    option        redispatch
    maxconn      2000
    timeout      5000
    clitimeout   50000
    srvtimeout   50000

listen admin_stats
    bind 192.168.5.171:1080
    mode http
    option httplog
    maxconn 10
    stats refresh 30s
    stats uri /stats
    stats auth admin:admin
    stats hide-version
```



烂泥行天下  
<http://www.ilanni.com>





```

frontend web1b
    bind *:80
    acl is_dg hdr_beg(host) dg.test.com
    acl is_ilanni hdr_beg(host) ilanni.test.com
    acl is_171 hdr_beg(host) 192.168.5.171
    acl is_ip src 192.168.5.140
    use_backend acl if is_171 is_ip
    use_backend dgserver if is_dg
    use_backend ilanni if is_ilanni
    use_backend 171server if is_171
    default_backend backend_default

backend dgserver
    balance source
    server web1 192.168.5.171:8080 maxconn 1024 weight 3 check inter 2000 rise
    server web2 192.168.5.174:8080 maxconn 1024 weight 3 check inter 2000 rise
    server web3 192.168.5.178:8080 maxconn 1024 weight 3 check inter 2000 rise

backend 171server
    balance roundrobin
    server dg1 192.168.5.174:80 check
    server dg2 192.168.5.178:80 check

backend ilanni
    server web1 www.yuanbaopu.com:80 weight 3 check inter 2000 rise 2 fall 3

backend acl
    balance source
    server web1 www.ilanni.com:80 maxconn 1024 weight 3 check inter 2000 rise 2

backend backend_default
    server web1 192.168.5.178:8080 weight 3 check inter 2000 rise 2 fall 3

listen 8090
    bind 0.0.0.0:8090
    mode http
    balance roundrobin
    server web1 192.168.5.174:8090 maxconn 1024 weight 5 check inter 2000 rise 2 fall 3
    server web2 192.168.5.178:8090 maxconn 1024 weight 3 check inter 2000 rise 2 fall 3

```

烂泥行  
http://www.ilanni.com

#### 4.2、haproxy配置实例讲解

有关haproxy配置文件我们先简单介绍，如下：

global配置段，用于设定全局配置参数。

代理配置段中，主要是使用defaults、frontend、backend、listen关键词。

defaults配置段用于为所有其它配置段提供默认参数，这配置默认配置参数可由下一个“defaults”所重新设定。

frontend配置段用于定义一系列监听的套接字，这些套接字可接受客户端请求并与之建立连接。

backend配置段用于定义一系列“后端”服务器，代理将会将对应客户端的请求转发至这些服务器。

listen配置段通过关联“前端”和“后端”定义了一个完整的代理，通常只对TCP流量有用。

在上述haproxy配置文件中，我们主要讲解ACL的匹配规则。

acl is\_dg hdr\_beg(host) dg.test.com

该行定义一个is\_dg规则，如果客户端请求的是dg.test.com这个域名，则定义该规则为is\_dg。

use\_backend dgserver if is\_dg

该定义一个dgserver服务器组，如果客户端请求符合is\_dg定义的规则，则把该客户端的请求分发到dgserver服务器组。

acl is\_ip src 192.168.5.140

该行定义一个is\_ip规则，如果客户端的IP地址是192.168.5.140，则定义该规则为is\_ip。

use\_backend acl if is\_171 is\_ip

该定义一个acl服务器组，如果客户端请求同时符合is\_ip和is\_171定义的规则，则把该客户端的请求分发到acl服务器组。

default\_backend backend\_default

该行定义一个默认服务器组，如果客户端请求不符合上述定义的任何一个规则，则把该客户端的请求分发到backend\_default服务器组。

其他的规则就不一一进行讲解了。

以上就是按照目前的业务要求，配置的haproxy。有关这些配置项参数，我们会在下一篇文章中进行详细介绍。

#### 五、查看haproxy监控页面

haproxy配置完毕后，现在来启动haproxy，使用如下命令：

/usr/local/haproxy/sbin/haproxy -f /usr/local/haproxy/haproxy.cfg

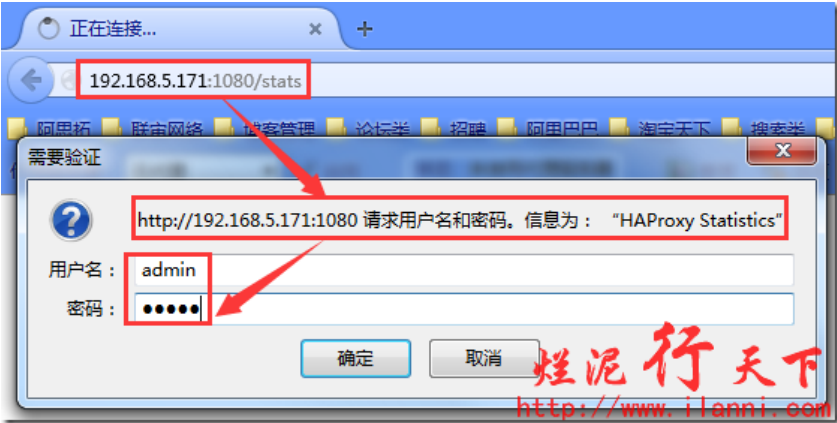
```
ps -ef |grep haproxy

netstat -tunlp|grep 3173

root@ubuntu-8:/usr/local/haproxy# /usr/local/haproxy/sbin/haproxy -f /usr/local/haproxy/hap
root@ubuntu-8:/usr/local/haproxy# ps -ef |grep haproxy
haproxy 3173 1 0 17:37 ? 00:00:00 /usr/local/haproxy/sbin/haproxy -f /usr/lo
root 3175 1541 0 17:37 pts/0 00:00:00 grep --color=auto haproxy
root@ubuntu-8:/usr/local/haproxy# netstat -tunlp|grep 3173
tcp 0 0 0.0.0.0:80 0.0.0.0:* LISTEN 3173/haproxy
tcp 0 0 192.168.5.171:1080 0.0.0.0:* LISTEN 3173/haproxy
tcp 0 0 0.0.0.0:8090 0.0.0.0:* LISTEN 3173/haproxy
udp 0 0 0.0.0.0:58353 0.0.0.0:* 3173/haproxy
root@ubuntu-8:/usr/local/haproxy#
```

现在我们来打开haproxy的监控页面，如下：

http://192.168.5.171:1080/stats



输入用户名和密码后，haproxy监控后台服务器的情况，如下：

Statistics Report for HAProxy

192.168.5.171:1080/stats

阿思拓 联合网络 博客管理 论坛类 招聘 阿里巴巴 淘宝天下 搜索类 网盘类 Linux

代理服务器：无代理 启用 状态 未使用代理服务器 管理 选项

## HAProxy

Statistics Report for pid 3173

> General process information

pid = 3173 (process #1, nbproc = 1)  
uptime = 0d 0h05m52s  
system limits: memmax = unlimited; ulimit-n = 8214  
maxsock = 8214; maxconn = 4096; maxpipes = 0  
current conns = 1; current pipes = 0/0  
Running tasks: 1/10

	Queue			Session rate			Sessions			
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total
Frontend				1	3	-	1			1
Backend	0	0		0	2		0			0

现在来看看haproxy所有后台服务器的情况，先来看看dgserver组，如下：

dgserver																		
	Queue			Session rate			Sessions				Bytes		Denied		Errors			
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp
web1	0	0	-	0	0	0	0	0	1024	0	0	0	0	0	0	0	0	0
web2	0	0	-	0	0	0	0	0	1024	0	0	0	0	0	0	0	0	0
web3	0	0	-	0	0	0	0	0	1024	0	0	0	0	0	0	0	0	0
Backend	0	0		0	0		0	0	0	0	0	0	0	0	0	0	0	0

通过上图，我们可以看到dgserver组，目前web1这台服务器处于宕机状态。也就是说haproxy不会把客户端的请求分发到web1服务器上了。

现在我们再来看看171server组，如下：

171server										
	Queue			Session rate			Sessions			
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	To
dg1	0	0	-	0	1	-	0	1	-	-
dg2	0	0	-	0	0	-	0	0	-	-
Backend	0	0	-	0	1	-	0	1	-	-

通过上图，我们可以很明显的看出171server组的服务器目前都是正常运行的。

8090组服务器的情况，如下：

8090										
	Queue			Session rate			Sessions			
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	To
Front-end	0	0	-	0	1	-	0	1	2 000	-
web1	0	0	-	0	0	-	0	0	1024	-
web2	0	0	-	0	1	-	0	1	1024	-
Backend	0	0	-	0	1	-	0	1	2 000	-

通过上图，我们可以看到8090组的服务器也是有一台是宕机的。

六、测试haproxy负载均衡

haproxy配置完毕并正常启动后，我们现在来根据业务的要求进行测试。

6.1 测试域名跳转

使用如下命令测试域名跳转：

```
curl http://dg.test.com

ilanni@asto-server-developer:~$ curl http://dg.test.com
<?xml version="1.0" encoding="ISO-8859-1">
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>Apache Tomcat</title>
</head>
<body>
<h1>192.168.5.174</h1>
```



通过上图，我们可以看到当我们使用dg.test.com这个域名访问时，haproxy确实为我们进行跳转了，而且跳转到了192.168.5.174这台服务器上。

因为考虑到客户端session会话的问题，所以我们在配置haproxy负载均衡没有使用roundrobin轮询的方法，而是使用source方法。所以haproxy目前没有把请求分发到192.168.5.171:8080、192.168.5.178:8080这两台服务器上。

现在我们访问ilanni.test.com，如下：



通过上图，我们可以看到访问不同的域名，haproxy把请求分发到不同的服务器上。

## 6.2 测试IP地址跳转

使用如下命令测试IP地址跳转：

```
curl http://192.168.5.171
```



通过上图，我们可以很明显的看到，第一次使用curl http://192.168.5.171访问时，haproxy是把请求分发171server组的192.168.5.174这台机器上。而第二次访问时，把请求分发171server组的192.168.5.178这台机器上。

因为在haproxy配置中，我们使用的是roundrobin轮询方法，所以客户端的每一次请求，haproxy会把请求分发到不同的服务器上。

## 6.3 测试端口跳转

使用如下命令测试IP地址跳转：

```
curl http://dg.test.com:8090
```

```
curl http://ilanni.test.com:8090
```



通过上图，我们可以很明显的看到客户端在通过dg.test.com:8090和ilanni.test.com:8090进行访问时，haproxy确实把请求分发到了8090组服务器上的192.168.5.178:8090上。

## 6.4 测试默认跳转

要测试默认跳转，我们可以随便使用一个域名进行测试。如下：

```
curl http://test.test.com
```

```
ilanni@gasto-server-developer:~$ curl http://test.test.com
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
  <title>Apache Tomcat</title>
</head>
<body>
<h1>192.168.5.178-directory</h1>
<p>If you're seeing this page via a web browser, it means you've se
<p>This is the default Tomcat home page. It can be found on the loc
<p>Tomcat6 veterans might be pleased to learn that this system inst
e>CATALINA_BASE</code> in <code>/var/lib/tomcat6/code> to follow
```

通过上图，我们可以很明显的看出haproxy把test.test.com的请求转发到了默认服务器组的192.168.5.178:8080上。

## 6.5 测试多ACL匹配

要测试默认跳转，我们先切换到192.168.5.140这台机器上，然后访问http://192.168.5.171。如下：

```
[C:\Users\Administrator] ipconfig

Windows IP 配置

以太网适配器 本地连接 2:

    媒体状态 . . . . . : 媒体已断开
    连接特定的 DNS 后缀 . . . . . :

以太网适配器 本地连接:

    连接? 否? DNS 后缀 . . . . . :
    本地链接 IPv6 地址 . . . . . : fe80::296c:93fc:214b:6119%12
    IPv4 地址 . . . . . : 192.168.5.140
    子网掩码 . . . . . : 255.255.255.0
    默认网关 . . . . . : 192.168.5.1
```

通过上图，我们可以很明显的看到在192.168.5.171这台机器上访问http://192.168.5.171时，haproxy确实把请求分发到www.ilanni.com这台机器上。

## 七、centos安装haproxy

有关在centos上安装haproxy，我们在此就不多做介绍了。只把相关的操作命令贴出来。

yum方式安装：yum -y install haproxy

源码方式安装：

useradd haproxy

cat /etc/passwd |grep haproxy

uname -a

yum -y install gcc make

wget <http://www.haproxy.org/download/1.3/src/haproxy-1.3.27.tar.gz>

tar -xf haproxy-1.3.27.tar.gz

cd haproxy-1.3.27

make TARGET=linux26 ARCH=x86\_64 PREFIX=/usr/local/haproxy

make install PREFIX=/usr/local/haproxy

分类：Linux

好文要顶

关注我

收藏该文



烂泥行天下

关注 - 7

粉丝 - 76

+加关注

« 上一篇: [烂泥：aws搭建shadowsocks服务器](#)

» 下一篇: [烂泥：高负载均衡学习haproxy之关键词介绍](#)

4

0

posted @ 2015-08-22 13:18 烂泥行天下 阅读(14113) 评论(0) 编辑 收藏

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

【推荐】超50万VC++源码: 大型组态工控、电力仿真CAD与GIS源码库！

相关博文：

- [烂泥：高负载均衡学习haproxy之关键词介绍](#)
- [负载均衡-haproxy安装配置](#)
- [烂泥：高负载均衡学习haproxy之TCP应用](#)
- [高负载均衡学习haproxy之安装与配置](#)
- [centos之Haproxy 负载均衡学习笔记](#)

最新新闻：

- [世道变坏 从雷军生气开始](#)
  - [哈啰出行执行总裁：目前估值50亿美元 短期无融资计划](#)
  - [这个地方可能比火星和月球，更适合太空殖民](#)
  - [外媒：滴滴在压力中重组 运营重点转向安全和用户体验](#)
  - [苹果向老iPhone用户疯狂发邮件：549美元XR拿回家](#)
- » [更多新闻...](#)

Copyright ©2019 烂泥行天下

秀依林枫<http://xiuylf.taobao.com>®烂泥行天下<http://www.ilanni.com>