## HAProxy

#### 1、HAProxy简介

HAProxy是一个开源的、高性能的、基于TCP和HTTP应用的负载均衡软件，借助HAProxy可快速、可靠地提供基于TCP和HTTP应 用的 负载均衡解决方案

四层调度器

仅能根据IP地址及端口号实现调度

lvs

七层调度器

根据应用层数据实现不同的调度

nginx

#### 2、HAProxy的优点

1) 可靠性和稳定性非常好，可以与硬件的F5相媲美

2) 最高可以同时维护40000--50000个并发连接，单位时间内处理的最大请求数为20000个，最大数据处理能力可达10Gbps

3) 支持多于8种负载均衡算法 ，同时也支持session保持

4) 支持虚拟主机功能

5) 从HAProxy 1.3版本后开始支持连接拒绝、全透明代理等功能

6) HAProxy拥有一个功能强大的服务器状态监控页面

7) HAProxy拥有功能强大的ACL支持

HAProxy是借助于操作系统的技术特征来实现性能最大化的，因此，在使用HAProxy时，对操作系统进行性能调优是非常重要的。在业务系统方面，HAProxy非常适用于那些并发量特别大而且需要持久连接或四层和七层处理机制的web系统。另外，HAProxy也可以用于MySQL读操作的负载均衡

#### 3、HAProxy配置文件详解

根据功能、用途不同，其配置文件主要由5部分组成，分别为global部分，defaults部分，frontend部分、backend部分、listen部分

1) global部分

用于设置全局配置参数，属于进程级的配置，通常用操作系统配置相关

2) defaults部分

默认参数的配置部分。在些部分设置的参数，默认会自动引用到下面的frontend, backend和listen部分

3) frontend部分

用于设置接收用户请求的前端虚拟节点。frontend可以根据ACL规则直接指定要使用的后端backend

4) backend部分

用于设置集群后端服务集群的配置，也就是用来添加一组真实服务器，以处理前端用户的请求

5) listen部分

此部分是frontend和backend部分的结合体

#### 4、配置项说明：

1) global部分

global

log 127.0.0.1 local1 info

nbproc 1

maxconn 4096

user nobody

group nobody

daemon

pidfile /usr/local/haproxy/logs/haproxy.pid

log

全局的日志配置，local0是日志设备，info表示日志级别。其中日志级别有err, warning, info, debug 4种。这个配置表示使用127.0.0.1上的rsyslog服务中的local0日志设备，记录日志等级为info

maxconn

设置每个HAProxy进程可接受的最大并发连接数

nbproc

设置HAProxy启动时可创建的进程数，此参数要求将HAProxy运行模式设置为daemon，默认只启动一个进程；建议该值设置时小于CPU核数

daemon

设置HAProxy进程进入后台运行，这是推荐的运行模式

user/group

设置启动HAProxy进程的用户和组

pidfile

指定HAProxy进程ID的存放位置

2) defaults部分

defaults

mode http

retires 3

timeout check 5s

timeout connect 10

timeout client 20s

timeout server 30s

mode

设置HAProxy实例默认的运行模式，有tcp, http, health三个可选值

tcp模式：

在此模式下，客户端和服务器端间将建立一个全双工的连接，不会对七层报文做任何检查，为默认的模式；经常用于SSL, SSH, SMTP等应用

http模式

在此模式下，客户端请求在转发至后端服务器前将会被深度分析，所有不与RFC格式兼容的请求都会被拒绝

retires

设置连接后端服务器的失败重试次数，如果连接失败的次数超过该数值，HAProxy会将对应的后端服务器标记为不可用

timeout connect

设置成功连接到一台服务器的最长等待时间，默认单位是毫秒，但也可以使用其他时间单位作后缀

timeout client

设置连接客户端发送数据时最长等待时间，默认单位是毫秒，但也可以使用其他时间单位作后缀

timeout server

设置服务器端回应客户端数据发送的最长等待时间，默认单位是毫秒，但也可以使用其他时间单位作后缀

timeout check

设置对后端服务器的检测超时时间，默认单位是毫秒，但也可以使用其他时间单位作后缀

3) frontend部分

frontend www

bind 192.168.1.1:80

mode http

option httplog

option forwardfor

option httpclose

log global

default\_backend XXX

use\_backend XXX

通过frontend关键字定义了一个名为"www"的前端虚拟节点

bind

此选项用于定义一个或者几个监听的套接字，只能在frontend和listen中定义

格式如下：

bind [<address>:[port\_range]] [interface]

option httplog

默认情况下，HAProxy日志是不记录HTTP请求的，此选项的作用是启用日志记录HTTP请求

option forwardfor

此选项的作用是保证后端服务器可记录客户端真实的IP

option httpclose

此选项表示客户端和服务端完成一次连接请求后，HAProxy将主动关闭此TCP连接。这是对性能非常有帮助的一个参数

log global

表示使用global段中定义的日志格式

default\_backend htmpool

此选项用于指定后端默认的服务器池

4) backend部分

backend htmpool

mode http

option redispatch

cookie SERVERID

option abortonclose

balance roundrobin

server web1 10.1.1.1:80 cookie server1 weight 6 check inter 2000 rise 2 fall 3

server web2 10.1.1.2:80 cookie server2 weight 6 check inter 2000 rise 2 fall 3

backend用于定义一个名称为htmpool的后端服务器组，根据需要可以定义多个

option redispatch

此参数用于cookie保持的环境中。在默认情况下，HAProxy会将其请求的后端服务器的serverID插入cookie中，以保证会话的session持久性。而如果后端服务器出现故障，客户端的cookie是不会刷新的，这就会造成无法访问。此时，如果设置了此参数，就会将客户的请求强制定向到另外一台健康的后端服务器上，以保证服务正常

option abortonclose

此参数可以在服务器负载很高的情况下，自动结束当前队列中处理时间比较长的连接

balance roundrobin

指定负载均衡算法

HAProxy支持的负载均衡算法：

roundrobin

基于权重进行轮叫调度的算法

static-rr

基于权重进行轮叫调度的算法，不过此算法为静态算法，在运行时调整其服务器权重不会生效

source

基于请求源IP的算法。此算法先对请求的源IP进行HASH运算，然后将结果与后端服务器的权重总数相除后转发至某台匹配的后端服务器。这种方式可以使同一个客户端IP的请求始终转发到某特定的后端服务器

leastconn

此算法会将新的连接请求转发到具有最少连接数目的后端服务器。在会话时间较长的场景中推荐使用此算法 ，例如数据库负载均衡

uri

此算法会对部分或整个URI进行HASH运算，再经过与服务器的总权重相除，最后转发到某台匹配的后端服务器上

uri\_param

此算法会根据URL路径中的参数进行转发，这样可保证在后端真实服务器数据不变时，同一个用户的请求始终分发到同一台机器上

cookie SERVERID

表示允许向cookie插入SERVERID，每台服务器的SERVERID可在下面的server关键字中使用cookie关键字定义

server web1 10.1.1.1:80 cookie server1 weight 6 check inter 2000 rise 2 fall 3

server用于定义多台后端真实服务器，不能用于frontend和listen段

格式如下：

server <name> <address>:[port] [param\*]

name

为后端真实服务器指定一个内部名称，随便定义一个即可

address:port

指定后端服务器的IP地址及端口

param\*参数

常用的参数：

check 表示启用对此后端服务器执行健康状态检查

inter 设置健康状态检查的时间间隔，单位是毫秒

rise 检查多少次认为服务器可用

fall 检查多少次认为服务器不可用

weight 设置服务器的权重，默认为1， 最大为256。 设置为0表示不参与负载均衡

backup 设置备份服务器，用于所有后端服务器全部不可用时

cookie 为指定的后端服务器设置cookie值，此处指定的值将在请求入站时被检查，第一次为此值挑选的后端服务器将在后续的请求中一直被选中，其目的在于实现持久连接的功能

5) listen部分

listen admin\_status

bind 0.0.0.0:9188

log 127.0.0.1 local0 err

stats refresh 30s

stats uri /haproxy-status 1.1.1.1 http://1.1.1.1:9188/haproxy-status

stats realm Welcome login

stats auth admin:admin

stats hide-version

stats admin if TRUE

listen部分用于配置HAProxy监控页面相关的参数

stats refresh 30s

设置HAProxy监控统计页面自动刷新的时间

stats uri /haproxy-status

设置HAProxy监控页面访问的URI路径

stats realm Welcome login

设置登录监控页面时，密码框上的提示信息

stats auth admin:admin

设置登录监控页面的用户名，密码。用户密码用冒号隔开，可以设置多个，每行一个

stats hide-version

设置在监控页面上隐藏HAProxy的版本号

status admin if TRUE

设置此选项，可在监控页面上启用、禁用后端服务器，仅在1.4.9版本以后生效

#### 5、通过HAProxy的ACL规则实现七层负载均衡

ACL规则主要完成如下两种功能：

1、访问控制

2、七层调度

HAProxy的ACL规则常定义在frontend段中，格式如下：

acl <acl名称> <acl方法> -i [匹配的路径或文件]

常用的acl方法：

hdr\_reg(host) 以正则表达式的方式匹配主机名

hdr\_dom(host) 匹配主机名

hdr\_beg(host) 匹配主机名以xxx开头

url\_sub url地址

url\_dir uri地址

path\_end url以xxxx结尾

path\_beg url以xxxx开头

url\_ip 匹配数据目的IP

src 匹配数据源IP

method 匹配HTTP请求的方法；GET, POST

-i 表示不区分大小写

示例1： 实现七层调度

acl www\_policy hdr\_reg(host) -i ^(www.z.cn|z.cn)

acl bbs\_policy hdr\_dom(host) -i bbs.z.cn

acl url\_policy url\_sub -i buy\_sid=

use\_backend server\_www if www\_policy

use\_backend server\_app if bbs\_policy

use\_backend server\_bbs if url\_policy

default\_backend server\_cache

示例2： 实现七层调度

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

acl host\_www hdr\_beg(host) -i www

acl host\_static hdr\_beg(host) -i img. vedio. download. ftp.

use\_backend static if host\_static || host\_www

use\_backend www if host\_www

default\_backend server\_cache

示例3： 实现数据过滤

acl forbidden\_dst url\_ip 192.168.0.0/16

acl forbidden\_dst url\_ip 172.16.0.0/12

acl forbidden\_dst url\_ip 10.0.0.0/8

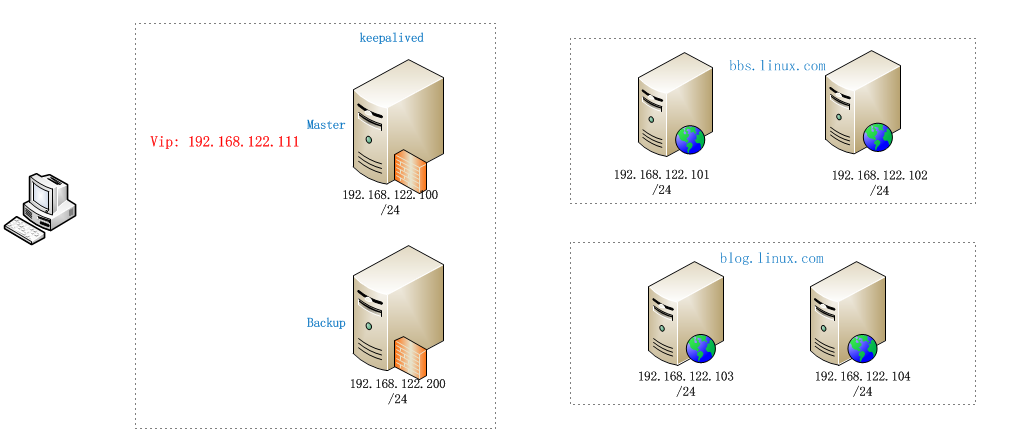
http-request deny if forbidden\_dst

示例4： 实现数据过滤

acl allow\_host src 192.168.200.150/32

http-request deny if !allow\_host

## 案例一： 七层调度 web服务



1、准备工作（集群中所有主机）

2、在所有web服务器上安装httpd，并建立测试页面

3、在haproxy01安装haproxy，并配置

[root@haproxy01 ~]# yum install -y haproxy

[root@haproxy01 ~]# cat /etc/haproxy/haproxy.cfg

global

maxconn 2000

nbproc 1

user nobody

group nobody

log 127.0.0.1 local0 info

daemon

pidfile /var/run/haproxy.pid

defaults

mode http

retries 3

timeout connect 5s

timeout client 10s

timeout server 30s

timeout check 2s

listen admin\_status

bind 192.168.122.100:8888

log 127.0.0.1 local0 err

stats refresh 30s

stats uri /haproxy-status

stats realm welcome login

stats auth admin:redhat

stats hide-version

stats admin if TRUE

frontend web\_service

bind 192.168.122.100:80

mode http

log global

option forwardfor

acl host\_bbs hdr\_dom(host) -i bbs.linux.com

acl host\_blog hdr\_beg(host) -i blog.

use\_backend server\_bbs if host\_bbs

use\_backend server\_blog if host\_blog

backend server\_bbs

mode http

option redispatch

cookie SERVERID

option abortonclose

balance roundrobin

server bbs01 192.168.122.101:80 cookie bbs01 weight 3 check inter 2000 rise 1 fall 2

server bbs02 192.168.122.102:80 cookie bbs02 weight 3 check inter 2000 rise 1 fall 2

backend server\_blog

mode http

option redispatch

cookie SERVERID

option abortonclose

balance roundrobin

server blog01 192.168.122.103:80 cookie blog01 weight 3 check inter 2000 rise 1 fall 2

server blog02 192.168.122.104:80 cookie blog02 weight 3 check inter 2000 rise 1 fall 2

4、启动haproxy服务

[root@haproxy\_01 ~]# systemctl start haproxy

[root@haproxy\_01 ~]# systemctl enable haproxy

[root@haproxy\_01 ~]# netstat -antp | grep haproxy

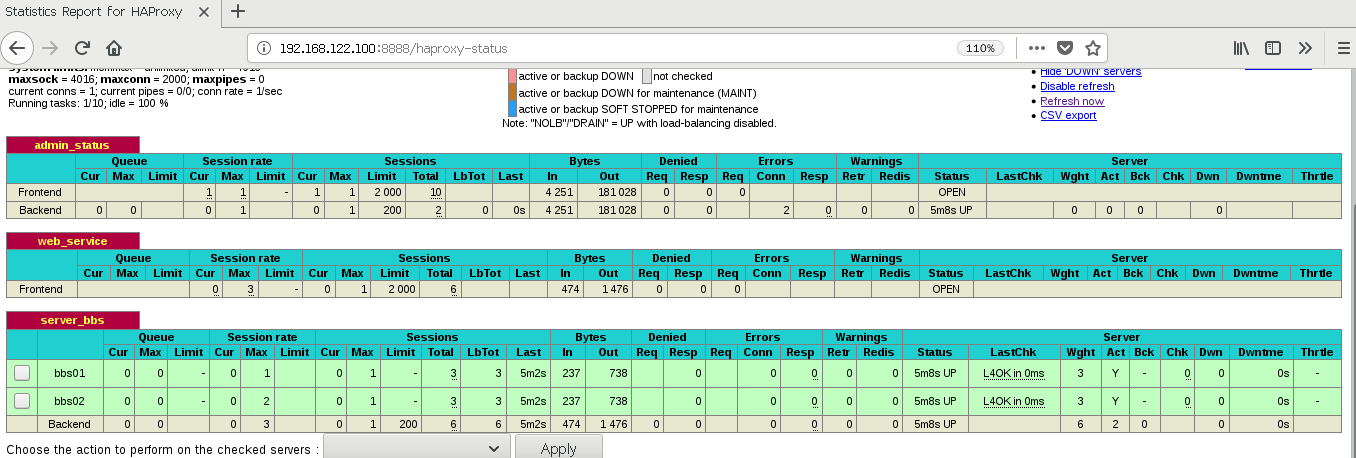
tcp 0 0 192.168.122.100:80 0.0.0.0:\* LISTEN 1320/haproxy

tcp 0 0 192.168.122.100:8888 0.0.0.0:\* LISTEN 1320/haproxy

5、客户端测试访问

通过IP访问时，默认返回503错误，可定义default\_backend解决

6、测试访问haproxy监控页面



7、后端web服务器记录日志客户端真实IP

LogFormat "%{X-Forwarded-For}i %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined

配置HAProxy\_02

[root@haproxy\_01 ~]# rsync -av /etc/haproxy/haproxy.cfg 192.168.122.200:/etc/haproxy/haproxy.cfg

[root@haproxy\_02 ~]# systemctl start haproxy

[root@haproxy\_02 ~]# systemctl enable haproxy

[root@haproxy\_02 ~]# netstat -antp | grep haproxy

tcp 0 0 192.168.122.200:80 0.0.0.0:\* LISTEN 10663/haproxy

tcp 0 0 192.168.122.200:8888 0.0.0.0:\* LISTEN 10663/haproxy

安装keepalived实现haproxy的HA

[root@haproxy\_01 ~]# cat /etc/keepalived/keepalived.conf

! Configuration File for keepalived

global\_defs {

router\_id haproxy01

}

vrrp\_instance haproxy\_server {

state MASTER

interface eth0

virtual\_router\_id 66

priority 100

advert\_int 1

authentication {

auth\_type PASS

auth\_pass redhat

}

virtual\_ipaddress {

192.168.122.111

}

}

[root@haproxy\_02 ~]# cat /etc/keepalived/keepalived.conf

! Configuration File for keepalived

global\_defs {

router\_id haproxy02

}

vrrp\_instance haproxy\_server {

state BACKUP

interface eth0

virtual\_router\_id 66

priority 80

advert\_int 1

authentication {

auth\_type PASS

auth\_pass redhat

}

virtual\_ipaddress {

192.168.122.111

}

}

## 案例二：MySQL 四层调度

1、配置MySQL双主复制(略)， 测试双主复制成功

2、haproxy配置

listen mysql\_service

bind \*:3306

mode tcp

balance roundrobin

server mysql01 192.168.122.101:3306 weight 1 check inter 2000 rise 1 fall 2

server mysql02 192.168.122.102:3306 weight 1 check inter 2000 rise 1 fall 2

[root@haproxy\_01 ~]# netstat -antp | grep haproxy

tcp 0 0 0.0.0.0:80 0.0.0.0:\* LISTEN 11241/haproxy

tcp 0 0 192.168.122.100:8888 0.0.0.0:\* LISTEN 11241/haproxy

tcp 0 0 0.0.0.0:3306 0.0.0.0:\* LISTEN 11241/haproxy