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

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之关键词介绍

本文由ilanniweb提供友情赞助，首发于烂泥行天下

上一篇文章我们简单讲解了有关haproxy的安装与搭建，在这篇文章我们把haproxy配置文件中使用到的关键词一一介绍下。

关注我微信ilanniweb

1、关键词balance

balance用于定义负载均衡的算法，可用于defaults、listen和backend中。

balance使用方法如下：

balance <algorithm> [<arguments>]

balance url_param <param> [check_post [<max_wait>]]

<algorithm>用于在负载均衡场景中挑选一个server，其仅应用于持久信息不可用的条件下或需要将一个连接重新派发至另一个服务器时。

balance支持的算法有：

roundrobin：基于权重进行轮询，在服务器的处理时间保持均匀分布时，这是最平衡、最公平的算法。此算法是动态的，这表示其权重可以在运行时进行调整。不过在设计上，每个后端服务器仅能最多接受4128个连接。

source：将请求的源地址进行hash运算，并由后端服务器的权重总数相除后派发至某匹配的服务器。这可以使得同一个客户端IP的请求始终被派发至某特定的服务器；不过，当服务器权重总数发生变化时，如某服务器宕机或添加了新的服务器，许多客户端的请求可能会被派发至与此前请求不同的服务器；常用于负载均衡无cookie功能的基于TCP的协议；其默认为静态，不过也可以使用hash-type修改此特性。

static-rr：基于权重进行轮询，与roundrobin类似，但是为静态方法，在运行时调整其服务器权重不会生效；不过，其在后端服务器连接数上没有限制。

leastconn：新的连接请求被派发至具有最少连接数目的后端服务器；在有着较长时间会话的场景中推荐使用此算法，如LDAP、SQL等，其并不太适用于较短会话的应用层协议，如HTTP；此算法是动态的，可以在运行时调整其权重。

uri：对URI的左半部分(“问题”标记之前的部分)或整个URI进行hash运算，并由服务器的总权重相除后派发至某匹配的服务器；这可以使得对同一个URI的请求总是被派发至某特定的服务器，除非服务器的权重总数发生了变化；此算法常用于代理缓存或反病毒代理以提高缓存的命中率；需要注意的是，此算法仅应用于HTTP后端服务器场景；其默认为静态算法，不过也可以使用hash-type修改此特性。

url_param：通过<argument>为URL指定的参数在每个HTTP GET请求中将会被检索；如果找到了指定的参数且其通过等于号“=”被赋予了一个值，那么此值将被执行hash运算并被服务器的总权重相除后派发至某匹配的服务器；此算法可以通过追踪请求中的用户标识进而确保同一个用户ID的请求将被送往同一个特定的服务器，除非服务器的总权重发生了变化；如果某请求中没有出现指定的参数或其没有有效值，则使用轮叫算法对相应请求进行调度；此算法默认为静态的，不过其也可以使用hash-type修改此特性。

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)

hdr(<name>)：对于每个HTTP请求，通过<name>指定的HTTP首部将会被检索；如果相应的首部没有出现或其没有有效值，则使用轮叫算法对相应请求进行调度；其有一个可选选项“use_domain_only”，可在指定检索类似Host类的首部时仅计算域名部分(比如通过www.ilanni.com来说，仅计算ilanni词符串的hash值)以降低hash算法的运算量；此算法默认为静态的，不过其也可以使用hash-type修改此特性；

2、关键词bind

bind仅能用于frontend和listen区段，用于定义一个或几个监听的套接词。

bind使用方法如下：

bind [<address>]:<port_range> [, ...]

bind [<address>]:<port_range> [, ...] interface <interface>

<address>：可选选项，其可以为主机名、IPv4地址、IPv6地址或*。省略此选项、将其指定为*或0.0.0.0时，将监听当前系统的所有IPv4地址。

<port_range>：可以是一个特定的TCP端口，也可是一个端口范围(如5005-5010)，代理服务器将通过指定的端口来接收客户端请求。

需要注意的是，每组监听的套接词<address:port>在同一个实例上只能使用一次，而且小于1024的端口需要有特定权限的用户才能使用，这可能需要通过uid参数来定义。

<interface>：指定物理接口的名称，仅能在Linux系统上使用。其不能使用接口别名，而仅能使用物理接口名称，而且只有管理有权限指定绑定的物理接口。

3、关键词mode

mode用于设定实例的运行模式或协议。当实现内容交换时，前端和后端必须工作于同一种模式(一般说来都是HTTP模式)，否则将无法启动实例。

mode可被用与listen、defaults、frontend、backend区段。

mode使用方法如下：

mode { tcp|http|health }

tcp：实例运行于纯TCP模式（即4层），在客户端和服务器端之间将建立一个全双工的连接，且不会对7层报文做任何类型的检查；此为默认模式，通常用于SSL、SSH、SMTP等应用。

http：实例运行于HTTP模式（即7层），客户端请求在转发至后端服务器之前将被深度分析，所有不与RFC格式兼容的请求都会被拒绝。

health：实例工作于health模式，其对入站请求仅响应“OK”信息并关闭连接，且不会记录任何日志信息；此模式将用于响应外部组件的健康状态检查请求；目前此模式已经废弃，因为tcp或http模式中的monitor关键词可完成类似功能；

4、关键词hash-type

hash-type定义用于将hash码映射至后端服务器的方法；其不能用于frontend区段；可用方法有map-based和consistent，在大多数场景下推荐使用默认的map-based方法。

hash-type使用方法如下：

hash-type <method>

map-based：hash表是一个包含了所有在线服务器的静态数组。其hash值将会非常平滑，会将权重考虑在列，但其为静态方法，对在线服务器的权重进行调整将不会生效，这意味着其不支持慢速启动。此外，挑选服务器是根据其在数组中的位置进行的，因此，当一台服务器宕机或添加了一台新的服务器时，大多数连接将会被重新派发至一个与此前不同的服务器上，对于缓存服务器的工作场景来说，此方法不甚适用。

consistent：hash表是一个由各服务器填充而成的树状结构；基于hash键在hash树中查找相应的服务器时，最近的服务器将被选中。此方法是动态的，支持在运行时修改服务器权重，因此兼容慢速启动的特性。添加一个新的服务器时，仅会对一小部分请求产生影响，因此，尤其适用于后端服务器为cache的场景。不过，此算法不甚平滑，派发至各服务器的请求未必能达到理想的均衡效果，因此，可能需要不时的调整服务器的权重以获得更好的均衡性。

5、关键词log

log为每个实例启用事件和流量日志，因此可用于所有区段。每个实例最多可以指定两个log参数，不过，如果使用了“log global”且"global"段已经定了两个log参数时，多余了log参数将被忽略。

log global

log <address> <facility> [<level> [<minlevel>]]

global：当前实例的日志系统参数同"global"段中的定义时，将使用此格式；每个实例仅能定义一次“log global”语句，且其没有任何额外参数。

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OpenVPN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？

--鲧

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(14602)

3. 烂泥：Postfix邮件服务器搭建之软件安装与配置(14302)

<address>：定义日志发往的位置，其格式之一可以为<IPv4_address:PORT>，其中的port为UDP协议端口，默认为514；格式之二为Unix套接词文件路径，但需要留心chroot应用及用户的读写权限。

<facility>：可以为syslog系统的标准facility之一。

<level>：定义日志级别，即输出信息过滤器，默认为所有信息；指定级别时，所有等于或高于此级别的日志信息将会被发送。

6、关键词maxconn

maxconn设定一个前端的最大并发连接数，因此，其不能用于backend区段。

maxconn使用方法如下：

maxconn <conns>

对于大型站点来说，可以尽可能提高此值以便让haproxy管理连接队列，从而避免无法应答用户请求。当然，此最大值不能超出“global”段中的定义。此外，需要留心的是，haproxy会为每个连接维持两个缓冲，每个缓冲的大小为8KB，再加上其它的数据，每个连接将大约占用17KB的RAM空间。这意味着经过适当优化后，有着1GB的可用RAM空间时将能维护40000-50000并发连接。

如果为<conns>指定了一个过大值，极端场景下，其最终占据的空间可能会超出当前主机的可用内存，这可能会带来意想不到的结果；因此，将其设定了一个可接受值方为明智决定。其默认为2000。

7、关键词default_backend

default_backend定义在没有匹配的use_backend规则时为实例指定使用的默认后端服务器，因此，其不可应用于backend区段。在frontend和backend之间进行内容交换时，通常使用use-backend定义其匹配规则；而没有被规则匹配到的请求将由此参数指定的后端服务器接收。

default_backend使用方法如下：

default_backend <backend>

<backend>：指定使用的后端的名称。

使用案例：

use_backend dynamic if url_dyn

use_backend static if url_css url_img

default_backend dynamic

8、关键词server

server为后端声明一个server，因此，不能用于defaults和frontend区段。

server使用方法如下：

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

<name>：为此服务器指定的内部名称，其将出现在日志及警告信息中；如果设定了http-send-server-name，它还将被添加至发往此服务器的请求首部中。

<address>：为此服务器的的IPv4地址，支持使用可解析的主机名，同时也支持域名，只不过在启动时需要解析主机名至相应的IPv4地址。

[:port]：指定将连接请求所发往的此服务器时的目标端口，其为可选项；未设定时，将使用客户端请求时的同一相端口。

[param*]：为此服务器设定的一系参数；其可用的参数非常多，具体请参考官方文档中的说明，下面仅说明几个常用的参数；

服务器或默认服务器参数：

backup：设定为备用服务器，仅在负载均衡场景中的其它server均不可用于启用此server。

check：启动对此server执行健康状态检查，其可以借助于额外的其它参数完成更精细的设定，如：

inter <delay>：设定健康状态检查的时间间隔，单位为毫秒，默认为2000；也可以使用fastinter和downinter来根据服务器端状态优化此时间延迟；

rise <count>：设定健康状态检查中，某离线的server从离线状态转换至正常状态需要成功检查的次数；

fall <count>：确认server从正常状态转换为不可用状态需要检查的次数；

cookie <value>：为指定server设定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server将在后续的请求中被选中，其目的在于实现持久连接的功能；

maxconn <maxconn>：指定此服务器接受的最大并发连接数；如果发往此服务器的连接数目高于此处指定的值，其将被放置于请求队列，以等待其它连接被释放；

maxqueue <maxqueue>：设定请求队列的最大长度；

4. 烂泥：高负载均衡学习haproxy之安装与配置(14121)

5. 烂泥：ubuntu 14.04搭建OpenVPN服务器(11167)

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：jira7.2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)

observe <mode>：通过观察服务器的通信状况来判定其健康状态，默认为禁用，其支持的类型有“layer4”和“layer7”，“layer7”仅能用于http代理场景；

一个标准的使用案例，如下：

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

以上就是[param*]中比较经常使用的参数。

redir <prefix>：启用重定向功能，将发往此服务器的GET和HEAD请求均以302状态码响应；需要注意的是，在prefix后面不能使用/，且不能使用相对地址，以免造成循环；例如：

server srv1 192.168.5.174:80 redir http://imags.ilanni.com check

weight <weight>：权重，默认为1，最大值为256，0表示不参与负载均衡。

检查方法：

option httpchk

option httpchk <uri>

option httpchk <method> <uri>

option httpchk <method> <uri> <version>：不能用于frontend段

例如：

backend https_relay

mode tcp

option httpchk OPTIONS * HTTP/1.1\r\nHost:\ www.ilanni.com

server apache1 192.168.1.1:443 check port 80

9、关键词stats enable

stats enable启用基于程序编译时默认设置的统计报告，stats enable不能用于frontend区段。只要没有另外的其它设定，它们就会使用如下的配置：

stats uri : /stats

stats realm : "HAProxy Statistics"

stats auth : no authentication

stats scope : no restriction

尽管stats enable一条就能够启用统计报告，但还是建议设定其它所有的参数，以免其依赖于默认设定而带来非期望的后果。下面是一个配置案例。

backend public_www

server websrv1 192.168.5.171:80

stats enable

stats hide-version

stats scope

stats uri /stats

stats realm Haproxy\ Statistics

stats auth admin:password

stats auth admin:password

10、关键词stats hide-version

stats hide-version隐藏统计报告中haproxy版本号，不能用于frontend区段。默认情况下，统计页面会显示一些有用信息，包括haproxy的版本号。

然而，向所有人公开haproxy的精确版本号是非常有风险的，因为它能帮助恶意用户快速定位版本的缺陷和漏洞。

11、关键词stats realm

stats realm启用统计报告并高精认证领域，不能用于“frontend”区段。

stats realm <realm>

haproxy在读取realm时会将其视作一个单词，因此，中间的任何空白词符都必须使用反斜线进行转义。此参数仅在与“stats auth”配置使用时有意义。

<realm>：实现HTTP基本认证时显示在浏览器中的领域名称，用于提示用户输入一个用户名和密码。

12、关键词stats scope

stats scope启用统计报告并限定报告的区段，不能用于frontend区段。

当指定此语句时，统计报告将仅显示其列举出区段的报告信息，所有其它区段的信息将被隐藏。如果需要显示多个区段的统计报告，此语句可以定义多次。需要注意的是，区段名称检测仅仅是以词符串比较的方式进行，它不会真检测指定的区段是否真正存在。

stats scope { <name> | "." }

<name>：可以是一个listen、frontend或backend区段的名称，而“.”则表示stats scope语句所定义的当前区段。

13、关键词stats auth

stats auth启用带认证的统计报告功能并授权一个用户帐号，其不能用于frontend区段。

stats auth <user>:<passwd>

<user>：授权进行访问的用户名；

<passwd>：此用户的访问密码，明文格式；

此语句将基于默认设定启用统计报告功能，并仅允许其定义的用户访问，其也可以定义多次以授权多个用户帐号。可以结合“stats realm”参数在提示用户认证时给出一个领域说明信息。在使用非法用户访问统计功能时，其将会响应一个“401 Forbidden”页面。其认证方式为HTTP Basic认证，密码传输会以明文方式进行，因此，配置文件中也使用明文方式存储以说明其非保密信息故此不能相同于其它关键性帐号的密码。

14、关键词stats admin

stats admin在指定的条件满足时启用统计报告页面的管理级别功能，它允许通过web接口启用或禁用服务器，不过，基于安全的角度考虑，统计报告页面应该尽可能为只读的。此外，如果启用了HAProxy的多进程模式，启用此管理级别将有可能导致异常行为。

stats admin { if | unless } <cond>

目前来说，POST请求方法被限制于仅能使用缓冲区减去保留部分之外的空间，因此，服务器列表不能过长，否则，此请求将无法正常工作。因此，建议一次仅调整少数几个服务器。下面是两个案例，第一个限制了仅能在本机打开报告页面时启用管理级别功能，第二个定义了仅允许通过认证的用户使用管理级别功能。

backend stats_localhost

stats enable

stats admin if LOCALHOST

backend stats_auth

stats enable

stats auth admin:password

stats admin if TRUE

15、关键词option httplog

option httplog启用记录HTTP请求、会话状态和计时器的功能。

option httplog [clf]

clf：使用CLF格式来代替HAProxy默认的HTTP格式，通常在使用仅支持CLF格式的特定日志分析器时才需要使用此格式。

默认情况下，日志输入格式非常简陋，因为其仅包括源地址、目标地址和实例名称，而“option httplog”参数将会使得日志格式变得丰富许多，其通常包括但不限于HTTP请求、连接计时器、会话状态、连接数、捕获的首部及cookie、“frontend”、“backend”及服务器名称，当然也包括源地址和端口号等。

16、关键词option logasap

option logasap

启用提前将HTTP请求记入日志，不能用于backend区段。

默认情况下，HTTP请求是在请求结束时进行记录以便能将其整体传输时长和词节数记入日志，由此，传较大的对象时，其记入日志的时长可能会略有延迟。option logasap参数能够在服务器发送complete首部时即时记录日志，只不过，此时将不记录整体传输时长和词节数。此情形下，捕获Content-Length响应首部来记录传输的词节数是一个较好选择。下面是一个例子。

listen http_proxy 0.0.0.0:80

mode http

option httplog

option logasap

log 172.16.100.9 local2

17、关键词option forwardfor

option forwardfor允许在发往服务器的请求首部中插入“X-Forwarded-For”首部。即启用获取客户端真实IP功能。

option forwardfor [except <network>] [header <name>] [if-none]

<network>：可选参数，当指定时，源地址为匹配至此网络中的请求都禁用此功能。

<name>：可选参数，可使用一个自定义的首部，如“X-Client”来替代“X-Forwarded-For”。有些独特的web服务器的确需要用于一个独特的首部。

if-none：仅在此首部不存在时才将其添加至请求报文问道中。

haproxy工作于反向代理模式，其发往服务器的请求中的客户端IP均为haproxy主机的地址而非真正客户端的地址，这会使得服务器端的日志信息记录不了真正的请求来源，“X-Forwarded-For”首部则可用于解决此问题。haproxy可以向每个发往服务器的请求上添加此首部，并以客户端IP为其value。

需要注意的是，haproxy工作于隧道模式，其仅检查每一个连接的第一个请求，因此，仅第一个请求报文被附加此首部。如果想为每一个请求都附加此首部，请确保同时使用了“option httpclose”、“option forceclose”和“option http-server-close”几个option。

下面是一个例子。

frontend www

mode http

option forwardfor except 127.0.0.1

18、关键词errorfile

errorfile在用户请求不存在的页面时，返回一个页面文件给客户端而非由haproxy生成的错误代码；可用于所有段中。

errorfile <code> <file>

<code>：指定对HTTP的哪些状态码返回指定的页面；这里可用的状态码有200、400、403、408、500、502、503和504。

<file>：指定用于响应的页面文件。

例如：

errorfile 400 /etc/haproxy/errorpages/400badreq.http

errorfile 403 /etc/haproxy/errorpages/403forbid.http

errorfile 503 /etc/haproxy/errorpages/503sorry.http

19、关键词errorloc和errorloc302

errorloc <code> <url>

errorloc302 <code> <url>

请求错误时，返回一个HTTP重定向至某URL的信息；可用于所有配置段中。

<code>：指定对HTTP的哪些状态码返回指定的页面；这里可用的状态码有200、400、403、408、500、502、503和504；

<url>：Location首部中指定的页面位置的具体路径，可以是在当前服务器上的页面的相对路径，也可以使用绝对路径；需要注意的是，如果URI自身错误时产生某特定状态码信息的话，有可能会导致循环定向；

需要留意的是，这两个关键词都会返回302状态吗，这将使得客户端使用同样的HTTP方法获取指定的URL，对于非GET法的场景(如POST)来说会产生问题，因为返回客户的URL是不允许使用GET以外的其它方法的。如果的确有这种问题，可以使用errorloc303来返回303状态码给客户端。

20、关键词errorloc303

errorloc303 <code> <url>

请求错误时，返回一个HTTP重定向至某URL的信息给客户端，可用于所有配置段中。

<code>：指定对HTTP的哪些状态码返回指定的页面；这里可用的状态码有400、403、408、500、502、503和504；

<url>：Location首部中指定的页面位置的具体路径，可以是在当前服务器上的页面的相对路径，也可以使用绝对路径；需要注意的是，如果URI自身错误时产生某特定状态码信息的话，有可能会导致循环定向；

例如：

backend webserver

server 172.16.100.6 172.16.100.6:80 check maxconn 3000 cookie srv01

server 172.16.100.7 172.16.100.7:80 check maxconn 3000 cookie srv02

errorloc 403 /etc/haproxy/errorpages/sorry.htm

errorloc 503 /etc/haproxy/errorpages/sorry.htm

分类: Linux

标签: 烂泥, 负载, 均衡, 学习, haproxy, 关键词, mode, balance, roundrobin, source

烂泥行天下
关注 - 7
粉丝 - 76
+加关注

0 0

« 上一篇：烂泥：高负载均衡学习haproxy之安装与配置
» 下一篇：烂泥：高负载均衡学习haproxy之TCP应用

posted @ 2015-08-22 17:47 烂泥行天下 阅读(506) 评论(0) 编辑 收藏

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