[**Mycat之——Mycat集群部署（基于HAProxy + Mycat）**](http://blog.csdn.net/l1028386804/article/details/75331934)

参考：http://blog.csdn.net/l1028386804/article/details/76397064

### 一、软件版本

操作系统：CentOS-6.5-x86\_64  
JDK版本：jdk1.7.0\_80  
HAProxy版本：haproxy-1.5.19.tar.gz  
Mycat版本：5.6.29-mycat-1.6-RELEASE-20161028204710 MyCat Server

MySQL版本：mysql-5.7

### 二、部署规划



实际 HAProxy 192.168.0.71

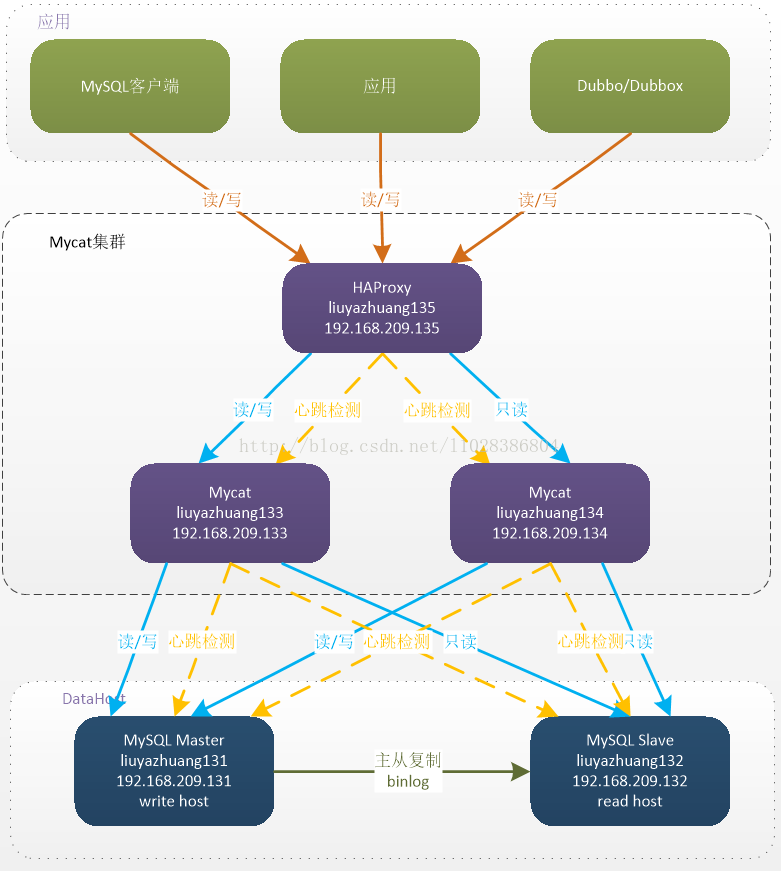
Mycat 1 mysql 192.168.0.70

Mycat 2 mysql 192.168.0.69

Mycat root 123456

Mysql root root AAaa\_123456

### 三、Mycat集群部署架构图



图解说明：  
HAProxy负责将请求分发到Mycat上，起到负载均衡的作用，同时HAProxy也能检测到Mycat是否存活，HAProxy只会将请求转发到存活的Mycat上。如果一台Mycat服务器宕机，HAPorxy转发请求时不会转发到宕机的Mycat上，所以Mycat依然可用。

### 四、Mycat节点2的部署

Mycat主机2（liuyazhuang134，192.168.209.134）请参考《[Mycat之——Mycat在MySQL主从复制基础上实现读写分离](http://blog.csdn.net/l1028386804/article/details/75269196" \t "_blank)》  
注意：liuyazhuang133（192.168.209.133）和 liuyazhuang134（192.168.209.134） 中都要加上（或更新）主机名映射配置。

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # vi /etc/hosts
2. 192.168.209.131 liuyazhuang131
3. 192.168.209.132 liuyazhuang132
4. 192.168.209.133 liuyazhuang133
5. 192.168.209.134 liuyazhuang134
6. 192.168.209.135 liuyazhuang135

### 五、配置Mycat状态检查服务（在Mycat节点主机上配置）

Mycat服务主机（liuyazhuang133、liuyazhuang134）上需要增加Mycat服务的状态检测脚本，并开放相应的检测端口，以提供给HAProxy对Mycat的服务状态进行检测判断。可以使用xinetd来实现，通过xinetd，HAProxy可以用httpchk来检测Mycat的存活状态。（xinetd即extended internet daemon，xinetd是新一代的网络守护进程服务程序，又叫超级Internet服务器。经常用来管理多种轻量级Internet服务。xinetd提供类似于inetd+tcp\_wrapper的功能，但是更加强大和安全。xinetd为linux系统的基础服务）

#### 1、安装xinetd

如果xinetd还没有安装，可使用如下命令安装：

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

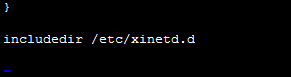
1. # yum install xinetd

#### 2、添加 includedir /etc/xinetd.d

检查/etc/xinetd.conf的末尾是否有 includedir /etc/xinetd.d ，没有就加上

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # vi /etc/xinetd.conf



#### 3、创建/etc/xinetd.d 目录

检查 /etc/xinetd.d 目录是否存在，不存在则创建

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # mkdir /etc/xinetd.d/

#### 4、增加Mycat存活状态检测服务配置

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # touch /etc/xinetd.d/mycat\_status
2. # vi /etc/xinetd.d/mycat\_status

增加以下内容：

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. service mycat\_status
2. {
3. flags = REUSE
4. ## 使用该标记的socket\_type为stream，需要设置wait为no
5. socket\_type = stream ## 封包处理方式，Stream为TCP数据包
6. port = 48700 ## 服务监听端口
7. wait = no ## 表示不需等待，即服务将以多线程的方式运行
8. user = root ## 执行此服务进程的用户
9. server =/usr/local/bin/Mycat\_status ## 需要启动的服务脚本
10. log\_on\_failure += USERID ## 登录失败记录的内容
11. disable = no ## 要启动服务，将此参数设置为no
12. }

#### 5、添加 /usr/local/bin/Mycat\_status 服务脚本

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # touch /usr/local/bin/mycat\_status
2. # vi /usr/local/bin/mycat\_status

增加以下内容：

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. #!/bin/bash
2. #/usr/local/bin/mycat\_status.sh
3. # This script checks if a Mycat server is healthy running on localhost.
4. # It will return:
5. # "HTTP/1.x 200 OK\r" (if Mycat is running smoothly)
6. # "HTTP/1.x 503 Internal Server Error\r" (else)
7. Mycat=`/usr/local/Mycat/bin/Mycat status | grep 'not running' | wc -l`
8. if [ "$Mycat" = "0" ]; then
9. /bin/echo -e "HTTP/1.1 200 OK\r\n"
10. else
11. /bin/echo -e "HTTP/1.1 503 Service Unavailable\r\n"
12. fi

#### 6、给新增脚本赋予可执行权限

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # chmod a+x /usr/local/bin/mycat\_status

#### 7、在 /etc/services 中加入 mycat\_status 服务

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # vi /etc/services

在末尾加入：

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. mycat\_status 48700/tcp # mycat\_status

  
保存后，重启xinetd服务

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

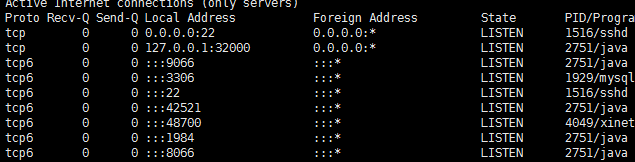
1. # service xinetd restart

#### 8、验证mycat\_status服务是否成功启动

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # netstat -antup|grep 48700
2. netstat –lntp

http://img.blog.csdn.net/20170719000203368?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvbDEwMjgzODY4MDQ=/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center

  
能看到上图这样的信息，说明服务配置成功。

#### 9、Mycat服务主机的防火墙上打开 48700端口

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # vi /etc/sysconfig/iptables

增加：

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. -A INPUT -m state --state NEW -m tcp -p tcp --dport 48700 -j ACCEPT

保存后重启防火墙

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # service iptables restart

脚本测试：

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # /usr/local/bin/mycat\_status

### 六、HAProxy介绍

HAProxy官网：[http://www.haproxy.org/](http://www.haproxy.org/" \t "_blank)  
HAProxy各版本的官方文档：[http://cbonte.github.io/haproxy-dconv/index.html](http://cbonte.github.io/haproxy-dconv/index.html" \t "_blank)  
HAProxy 是一款提供高可用性、负载均衡以及基于TCP（第四层）和HTTP（第七层）应用的代理软件，支持虚拟主机，它是免费、快速并且可靠的一种解决方案。  
HAProxy目前主要有三个版本：1.4、1.5、1.6、1.7，CentOS6.6自带的RPM包为1.5的。  
HAProxy1.5版开始，支持SSL、DDoS防护等功能，可看官网说明：  
version 1.5 : the most featureful version, supports SSL, IPv6, keep-alive, DDoS protection, etc...  
Mycat官方推荐使用HAProxy做MyCat的高可用负载均衡代理。

### 七、HAProxy的安装（192.168.209.135）

#### 1、下载（或上传） haproxy-1.5.19.tar.gz 到 /usr/local/src，解压安装

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. [root@liuyazhuang135 src]# cd /usr/local/src/
2. #wget http://www.haproxy.org/download/1.5/src/haproxy-1.5.19.tar.gz
3. [root@liuyazhuang135 src]# tar -zxvf haproxy-1.5.19.tar.gz
4. [root@liuyazhuang135 src]# cd haproxy-1.5.19

#### 2、如需了解安装注意点，可查看HAProxy的软件说明

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. [root@liuyazhuang135 haproxy-1.5.19]# less README

#### 3、安装编译所需的依赖包

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # yum install gcc gcc-c++ pcre pcre-devel zlib zlib-devel openssl openssl-devel

#### 4、编译

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # make TARGET=linux2628 ARCH=x86\_64 USE\_PCRE=1 USE\_OPENSSL=1 USE\_ZLIB=1 PREFIX=/usr/local/haproxy
2. ## TARGET是指定内核版本，高于2.6.28的建议设置为linux2628，Linux操作系统内核版本查看命令# uname -r， ARCH指定系统架构，openssl pcre zlib 这三个包需要安装不然不支持

#### 5、创建安装目录 /usr/local/haproxy

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # mkdir /usr/local/haproxy

#### 6、执行安装

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. [root@liuyazhuang135 haproxy-1.5.19]# make install PREFIX=/usr/local/haproxy
2. install -d "/usr/local/haproxy/sbin"
3. install haproxy "/usr/local/haproxy/sbin"
4. install -d "/usr/local/haproxy/share/man"/man1
5. install -m 644 doc/haproxy.1 "/usr/local/haproxy/share/man"/man1
6. install -d "/usr/local/haproxy/doc/haproxy"
7. for x in configuration architecture haproxy-en haproxy-fr; do \
8. install -m 644 doc/$x.txt "/usr/local/haproxy/doc/haproxy" ; \
9. done

#### 7、创建配置文件目录

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # mkdir -p /usr/local/haproxy/conf
2. # mkdir -p /etc/haproxy/

#### 8、从配置文件模版复制配置文件，并添加配置文件软连接

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # cp /usr/local/src/haproxy-1.5.19/examples/haproxy.cfg /usr/local/haproxy/conf/
2. # ln -s /usr/local/haproxy/conf/haproxy.cfg /etc/haproxy/haproxy.cfg

#### 9、拷贝错误页面，并添加目录软连接（HTTP模式选配）

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # cp -r /usr/local/src/haproxy-1.5.19/examples/errorfiles /usr/local/haproxy/
2. # ln -s /usr/local/haproxy/errorfiles /etc/haproxy/errorfiles

#### 10、拷贝开机启动文件，并赋予可执行权限

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # cp /usr/local/src/haproxy-1.5.19/examples/haproxy.init /etc/rc.d/init.d/haproxy
2. # chmod +x /etc/rc.d/init.d/haproxy

#### 11、添加haproxy命令脚本软连接

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # ln -s /usr/local/haproxy/sbin/haproxy /usr/sbin

#### 12、设置HAProxy开机启动

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # chkconfig --add haproxy
2. # chkconfig haproxy on

### 八、HAProxy配置Mycat负载均衡集群

HAProxy支持TCP（第四层）和HTTP（第七层）应用的代理，本节课程我们使用HAProxy来做MyCat的负载均衡代理使用的是TCP模式。在4层模式下HAProxy仅在客户端和服务器之间转发双向流量。HAProxy配置简单，拥有非常不错的服务器健康检查功能，当其代理的后端服务器出现故障，HAProxy会自动将该服务器摘除，故障恢复后会自动将该服务器加入进来｡

#### 1、修改haproxy.cfg 配置文件

具体参数说明可参考官方配置文档 /usr/local/haproxy/doc/haproxy/configuration.txt  
或GitHub连接：[http://cbonte.github.io/haproxy-dconv/configuration-1.5.html](http://cbonte.github.io/haproxy-dconv/configuration-1.5.html" \t "_blank)

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # vi /usr/local/haproxy/conf/haproxy.cfg

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. ## global配置中的参数为进程级别的参数，通常与其运行的操作系统有关
3. global
5. log 127.0.0.1 local0 info ## 定义全局的syslog服务器，最多可以定义2个
7. ### local0是日志设备，对应于/etc/rsyslog.conf中的配置，默认回收info的日志级别
9. #log 127.0.0.1 local1 info
11. chroot /usr/share/haproxy ## 修改HAProxy的工作目录至指定的目录并在放弃权限之前执行
13. ### chroot() 操作，可以提升 haproxy 的安全级别
15. group haproxy ## 同gid，不过这里为指定的用户组名
17. user haproxy ## 同uid，但这里使用的为用户名
19. daemon ## 设置haproxy后台守护进程形式运行
21. nbproc 1 ## 指定启动的haproxy进程个数，
23. ### 只能用于守护进程模式的haproxy；默认为止启动1个进程，
25. ### 一般只在单进程仅能打开少数文件描述符的场中中才使用多进程模式
27. maxconn 4096 ## 设定每个haproxy进程所接受的最大并发连接数，
29. ### 其等同于命令行选项"-n"，"ulimit-n"自动计算的结果正式参照从参数设定的
31. # pidfile /var/run/haproxy.pid ## 进程文件（默认路径 /var/run/haproxy.pid）
33. node liuyazhuang135 ## 定义当前节点的名称，用于HA场景中多haproxy进程共享同一个IP地址时
35. description liuyazhuang135 ## 当前实例的描述信息
37. ## defaults：用于为所有其他配置段提供默认参数，这默认配置参数可由下一个"defaults"所重新设定
39. defaults
41. log global ## 继承global中log的定义
43. mode http ## mode:所处理的模式 (tcp:四层 , http:七层 , health:状态检查,只会返回OK)
45. ### tcp: 实例运行于纯tcp模式，在客户端和服务器端之间将建立一个全双工的连接，
47. #### 且不会对7层报文做任何类型的检查，此为默认模式
49. ### http:实例运行于http模式，客户端请求在转发至后端服务器之前将被深度分析，
51. #### 所有不与RFC模式兼容的请求都会被拒绝
53. ### health：实例运行于health模式，其对入站请求仅响应“OK”信息并关闭连接，
55. #### 且不会记录任何日志信息 ，此模式将用于相应外部组件的监控状态检测请求
57. option httplog
59. retries 3
61. option redispatch ## serverId对应的服务器挂掉后,强制定向到其他健康的服务器
63. maxconn 2000 ## 前端的最大并发连接数（默认为2000）
65. ### 其不能用于backend区段，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，
67. ### 从而避免无法应答用户请求。当然，此最大值不能超过“global”段中的定义。
69. ### 此外，需要留心的是，haproxy会为每个连接维持两个缓冲，每个缓存的大小为8KB，
71. ### 再加上其他的数据，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 ，
73. ### 有着1GB的可用RAM空间时将维护40000-50000并发连接。
75. ### 如果指定了一个过大值，极端场景中，其最终所占据的空间可能会超过当前主机的可用内存，
77. ### 这可能会带来意想不到的结果，因此，将其设定一个可接受值放为明智绝对，其默认为2000
79. timeout connect 5000ms ## 连接超时(默认是毫秒,单位可以设置us,ms,s,m,h,d)
81. timeout client 50000ms ## 客户端超时
83. timeout server 50000ms ## 服务器超时
85. ## HAProxy的状态信息统计页面
87. listen admin\_stats
89. bind :48800 ## 绑定端口
91. stats uri /admin-status ##统计页面
93. stats auth admin:admin ## 设置统计页面认证的用户和密码，如果要设置多个，另起一行写入即可
95. mode http
97. option httplog ## 启用日志记录HTTP请求
99. ## listen: 用于定义通过关联“前端”和“后端”一个完整的代理，通常只对TCP流量有用
101. listen mycat\_servers
103. bind :3307 ## 绑定端口
105. mode tcp
107. option tcplog ## 记录TCP请求日志
109. option tcpka ## 是否允许向server和client发送keepalive
111. option httpchk OPTIONS \* HTTP/1.1\r\nHost:\ www ## 后端服务状态检测
113. ### 向后端服务器的48700端口（端口值在后端服务器上通过xinetd配置）发送 OPTIONS 请求
115. ### (原理请参考HTTP协议) ，HAProxy会根据返回内容来判断后端服务是否可用.
117. ### 2xx 和 3xx 的响应码表示健康状态，其他响应码或无响应表示服务器故障。
119. balance roundrobin ## 定义负载均衡算法，可用于"defaults"、"listen"和"backend"中,默认为轮询方式
121. server mycat\_01 192.168.209.133:8066 check port 48700 inter 2000ms rise 2 fall 3 weight 10
123. server mycat\_02 192.168.209.134:8066 check port 48700 inter 2000ms rise 2 fall 3 weight 10
125. ## 格式：server <name> <address>[:[port]] [param\*]
127. ### serser 在后端声明一个server，只能用于listen和backend区段。
129. ### <name>为此服务器指定的内部名称，其将会出现在日志及警告信息中
131. ### <address>此服务器的IPv4地址，也支持使用可解析的主机名，但要在启动时需要解析主机名至响应的IPV4地址
133. ### [:[port]]指定将客户端连接请求发往此服务器时的目标端口，此为可选项
135. ### [param\*]为此server设定的一系列参数，均为可选项，参数比较多，下面仅说明几个常用的参数：
137. #### weight:权重，默认为1，最大值为256，0表示不参与负载均衡
139. #### backup:设定为备用服务器，仅在负载均衡场景中的其他server均不可以启用此server
141. #### check:启动对此server执行监控状态检查，其可以借助于额外的其他参数完成更精细的设定
143. #### inter:设定监控状态检查的时间间隔，单位为毫秒，默认为2000，
145. ##### 也可以使用fastinter和downinter来根据服务器端专题优化此事件延迟
147. #### rise:设置server从离线状态转换至正常状态需要检查的次数（不设置的情况下，默认值为2）
149. #### fall:设置server从正常状态转换至离线状态需要检查的次数（不设置的情况下，默认值为3）
151. #### cookie:为指定server设定cookie值，此处指定的值将会在请求入站时被检查，
153. ##### 第一次为此值挑选的server将会被后续的请求所选中，其目的在于实现持久连接的功能
155. #### 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，
157. #####其将被放置于请求队列，以等待其他连接被释放

注意：多节点部署时node 、 description的值要做相应调整。

#### 2、根据以上HAProxy配置文件要求做以下配置

(1)添加haproxy用户组和用户

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # groupadd haproxy
2. # useradd -g haproxy haproxy

(2)创建chroot运行的路径

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # mkdir /usr/share/haproxy

(3)防火墙中打开3306端口和48800端口

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # vi /etc/sysconfig/iptables
2. -A INPUT -m state --state NEW -m tcp -p tcp --dport 3306 -j ACCEPT
3. -A INPUT -m state --state NEW -m tcp -p tcp --dport 48800 -j ACCEPT

重启防火墙

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

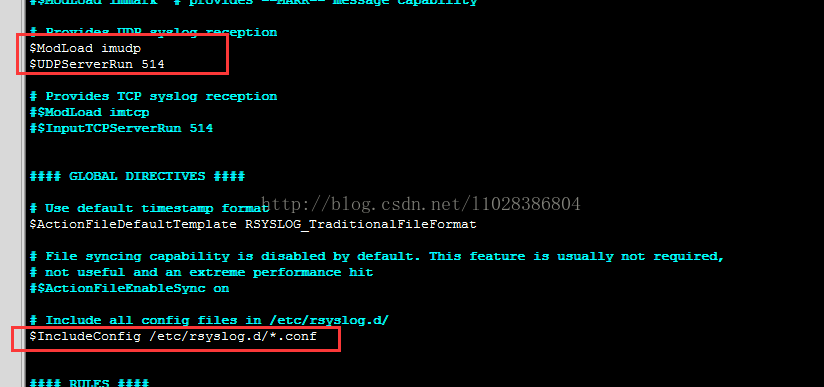
1. # service iptables restart

#### 3、开启rsyslog的haproxy日志记录功能

默认情况下 haproxy是不记录日志的，如果需要记录日志，还需要配置系统的syslog，在linux系统中是rsyslog服务。syslog服务器可以用作一个网络中的日志监控中心，rsyslog是一个开源工具，被广泛用于Linux系统以通过TCP/UDP协议转发或接收日志消息。安装配置rsyslog服务：

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # yum install rsyslog ## 没安装的情况下执行安装
2. # vi /etc/rsyslog.conf

把 $ModLoad imudp 和 $UDPServerRun 514前面的 # 去掉  
$ModLoad imudp ## 是模块名，支持UDP协议  
$UDPServerRun 514  
##允许514端口接收使用UDP和TCP协议转发过来的日志，  
##而rsyslog在默认情况下，正是在514端口监听UDP  
确认 #### GLOBAL DIRECTIVES #### 段中是否有 $IncludeConfig /etc/rsyslog.d/\*.conf没有则增加上此配置，增加后的效果：  


**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # cd /etc/rsyslog.d/ ## rsyslog服务会来此目录加载配置
2. # touch haproxy.conf ## 创建haproxy的日志配置文件
3. # vi /etc/rsyslog.d/haproxy.conf

增加以下内容：

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. local0.\* /var/log/haproxy.log
3. &~

##如果不加上面的的"&~"配置则除了在/var/log/haproxy.log中写入日志外，也会写入/var/log/message文件中  
配置保存后重启rsyslog服务

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # service rsyslog restart
2. Shutting down system logger: [ OK ]
3. Starting system logger: [ OK ]

（等到HAProxy服务启动后，就能在/var/log/haproxy.log中看到日志了）

#### 4、配置系统内核的IP包转发功能

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # vi /etc/sysctl.conf
2. net.ipv4.ip\_forward = 1

使配置生效

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # sysctl -p

#### 5、启动HAProxy

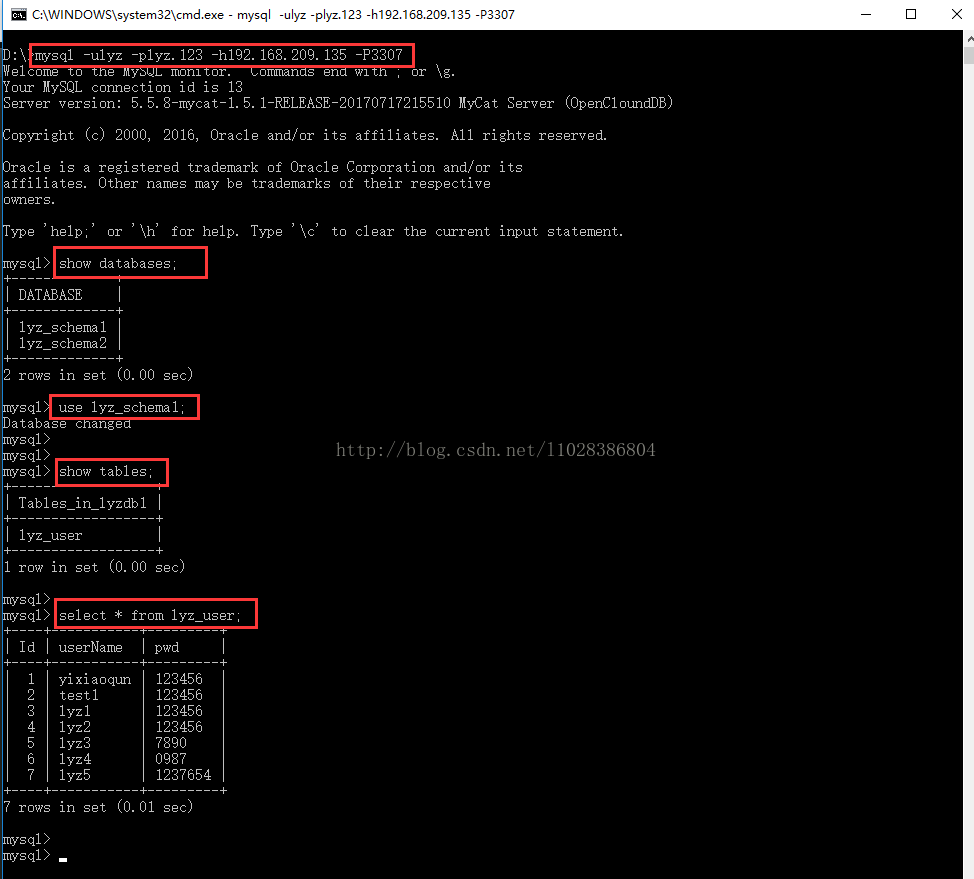
**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. # service haproxy start
2. # ps -ef | grep haproxy
3. haproxy 23921 1 0 23:27 ? 00:00:00 /usr/sbin/haproxy -D -f /etc/haproxy/haproxy.cfg -p /var/run/haproxy.pid
4. root 23924 23179 0 23:27 pts/1 00:00:00 grep haproxy
5. 如果出现报错：/haproxy.main()] Cannot chroot(/usr/share/haproxy)
6. 手动新建该目录
7. #mkdir /usr/share/haproxy

#### 6、使用MySQL客户端通过HAProxy连接Mycat

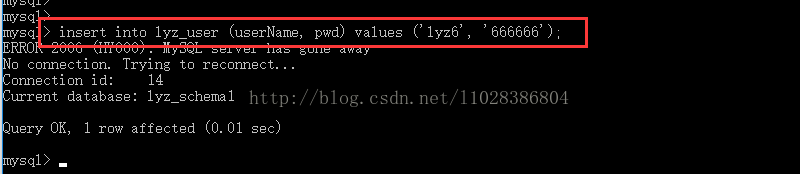
**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

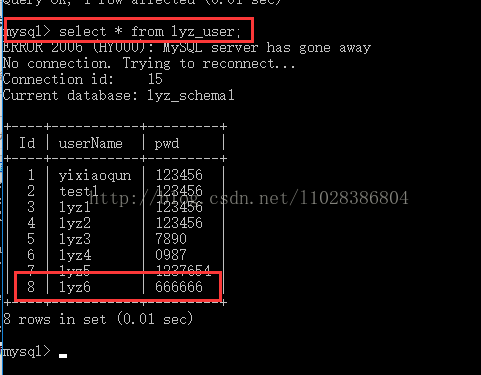
1. mysql -ulyz -plyz.123 -h192.168.209.135 -P3307
2. mysql> show databases;
3. mysql> lyz\_schema1;
4. mysql> show tables;
5. mysql> select \* from lyz\_user;

  
写数据测试

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. insert into lyz\_user (userName, pwd) values ('lyz6'666666');

  
查询后的结果如下：



#### 7、登录HAProxy的状态信息统计页面

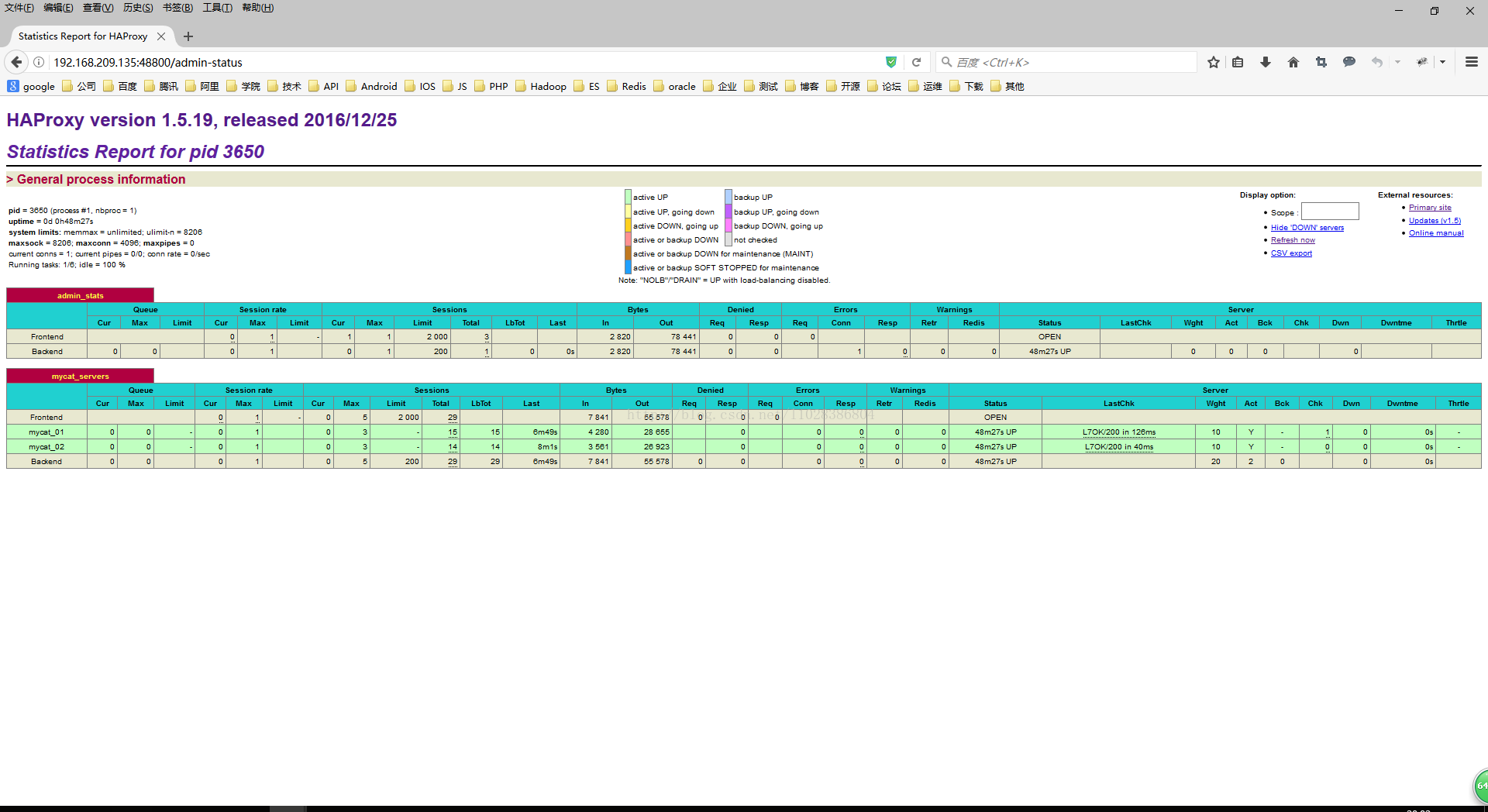
<http://192.168.209.135:48800/admin-status>

用户名和密码都是admin，对应的haproxy.cfg配置片段

**[plain]** [view plain](http://blog.csdn.net/l1028386804/article/details/75331934) [copy](http://blog.csdn.net/l1028386804/article/details/75331934)

1. ## HAProxy的状态信息统计页面
2. listen admin\_stats
3. bind :48800 ## 绑定端口
4. stats uri /admin-status ##统计页面
5. stats auth admin:admin ## 设置统计页面认证的用户和密码，如果要设置多个，另起一行写入即可
6. mode http
7. option httplog ## 启用日志记录HTTP请求

统计页面如下图所示：



至此，整个配置过程结束