**Yêu cầu bài toàn LBaas**

1. **Cấu hình transparent mode cho haproxy:**

Tham khảo: <https://www.haproxy.com/blog/howto-transparent-proxying-and-binding-with-haproxy-and-aloha-load-balancer/>

*HAProxy and the Linux Kernel*

Unfortunately, HAProxy can’t do transparent binding or proxying alone. It must stand on a compiled and tuned Linux Kernel and operating system.  
Below, I’ll explain how to do this in a standard Linux distribution.  
Here is the check list to meet:  
  1. appropriate HAProxy compilation option  
  2. appropriate Linux Kernel compilation option  
  3. sysctl settings  
  4. iptables rules  
  5. ip route rules  
  6. HAProxy configuration

*HAProxy compilation requirements*

First of all, HAProxy must be compiled with the option TPROXY enabled.  
It is enabled by default when you use the target LINUX26 or LINUX2628.

*Linux Kernel requirements*

You have to ensure your kernel has been compiled with the following options:  
  – CONFIG\_NETFILTER\_TPROXY  
  – CONFIG\_NETFILTER\_XT\_TARGET\_TPROXY

Of course, iptables must be enabled as well in your kernel 🙂

*sysctl settings*

The following sysctls must be enabled:  
  – net.ipv4.ip\_forward  
  – net.ipv4.ip\_nonlocal\_bind

*iptables rules*

You must setup the following iptables rules:

iptables -t mangle -N DIVERT

iptables -t mangle -A PREROUTING -p tcp -m socket -j DIVERT

iptables -t mangle -A DIVERT -j MARK --set-mark 1

iptables -t mangle -A DIVERT -j ACCEPT

Purpose is to mark packets which matches a socket bound locally (by HAProxy).

*IP route rules*

Then, tell the Operating System to forward packets marked by iptables to the loopback where HAProxy can catch them:

ip rule add fwmark 1 lookup 100

ip route add local 0.0.0.0/0 dev lo table 100

*HAProxy configuration*

Finally, you can configure HAProxy.  
  \* Transparent binding can be configured like this:

[...]

frontend ft\_application

bind 1.1.1.1:80 transparent

mode tcp

[...]

  \* Transparent proxying can be configured like this:

[...]

backend bk\_application

source 0.0.0.0 usesrc clientip

mode tcp

[...]

Ví dụ cấu hình hiện tại của haproxy (bỏ qua mấy cái %s đi nhé):

frontend http\_in\_%s  
mode tcp  
bind 127.0.0.1:%s transparent  
default\_backend http\_out\_%s  
  
backend http\_out\_%s  
mode tcp  
source 0.0.0.0 usesrc clientip  
balance source  
hash-type consistent  
#server bk1 125.235.32.106:80 maxconn 512 check  
%s

1. **Tối ưu hiệu năng mong muốn**

Tối ưu hiệu năng mong muốn: 1.000.000 req/s/1VM

Gợi ý (Đã test được khoảng 500.000 req/s):

*Cấu hình tối ưu hiệu năng của HĐH:*

Sửa file /etc/sysctl.conf thêm các tham số:

net.ipv4.tcp\_max\_syn\_backlog = 8192

net.ipv4.tcp\_synack\_retries = 3

net.ipv4.tcp\_fin\_timeout = 3

net.core.somaxconn = 2000000

fs.file-max = 2000000

net.ipv4.tcp\_tw\_reuse = 1

net.ipv4.tcp\_max\_tw\_buckets = 200000

net.ipv4.ip\_local\_port\_range = 2048 65535

net.netfilter.nf\_conntrack\_max = 2000000

net.netfilter.nf\_conntrack\_tcp\_timeout\_syn\_recv = 3

net.netfilter.nf\_conntrack\_tcp\_timeout\_fin\_wait = 3

net.netfilter.nf\_conntrack\_tcp\_timeout\_close\_wait = 3

net.netfilter.nf\_conntrack\_tcp\_timeout\_time\_wait = 3

Sửa file /etc/rc.local thêm 1 dòng sau vào cuối file (trước exit 0)

echo 250000 > /sys/module/nf\_conntrack/parameters/hashsize

Sửa file /etc/security/limits.conf thêm 2 dòng sau:

\* soft nofile 500000

\* hard nofile 500000

*Cấu hình tối ưu hiệu năng của haproxy (Chạy nhiều tiến trình haproxy, mỗi tiến trình bind vào 1 CPU của VM).*

Thêm vào block global và defaults các tham số sau:

global  
 maxconn 2048  
 nbproc 4  
 cpu-map 1 0  
 cpu-map 2 1  
 cpu-map 3 2  
 cpu-map 4 3  
  
defaults  
 timeout connect 5000ms  
 timeout client 50000ms  
 timeout server 50000ms