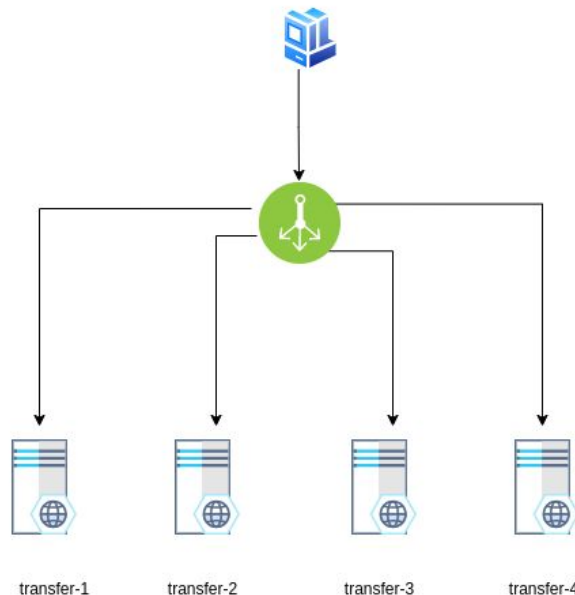
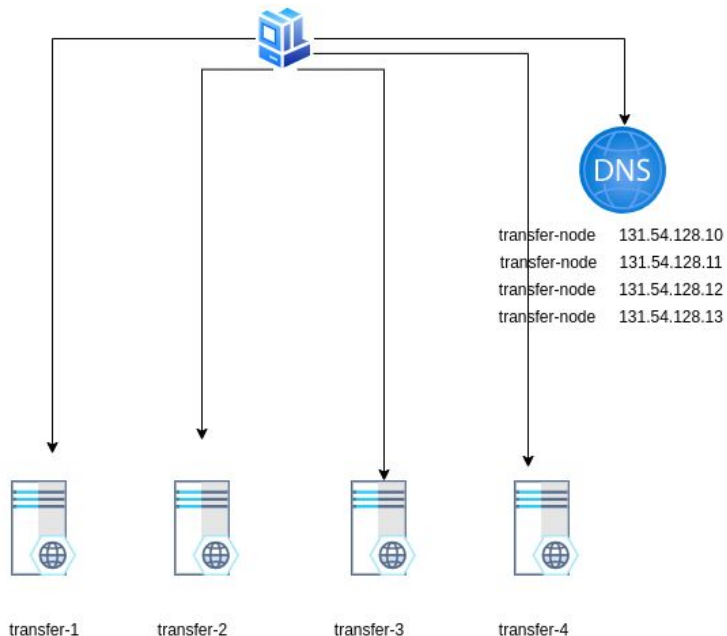


Load Balancing

A (not so) new deployment strategy

The problem



What we have (except xrootd)

- currently we rely on DNS Round Robin
- we record in DNS a hostname for several IP address

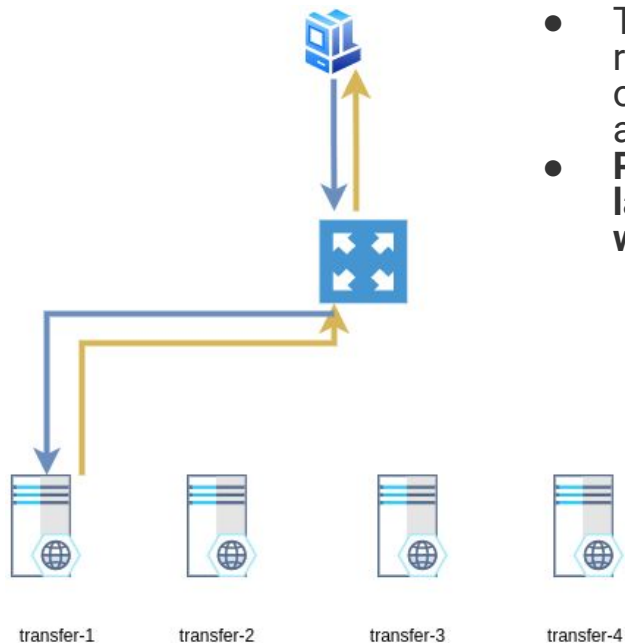
What we want

- Fine tune the distribution algorithm
- i.e. DNS doesn't account for server load or any other metric for service health

First Try - HAProxy

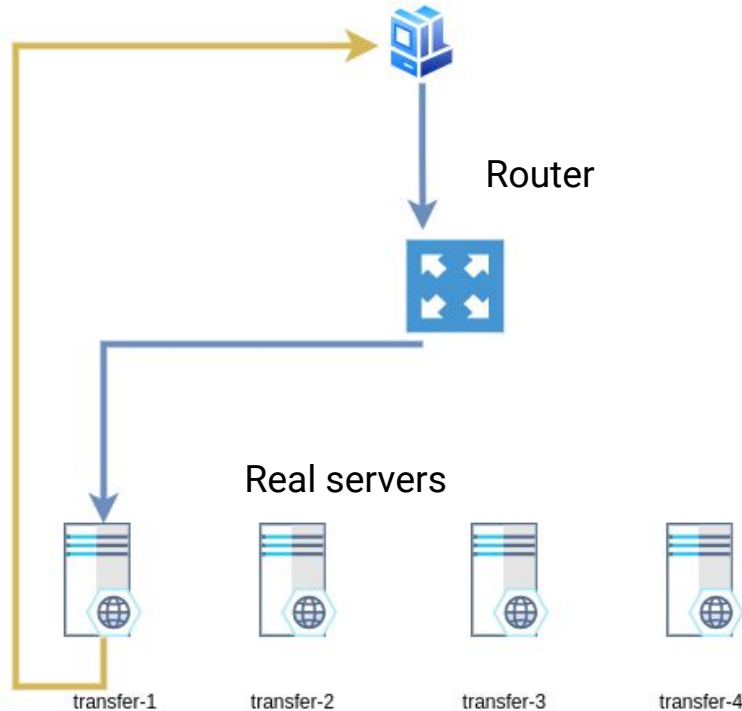
HAProxy has lot of features:

- <http://docs.haproxy.org/dev/intro.html>
- Layer 4 Proxy Mode (TCP)
- Layer 7 Proxy Mode (HTTP)



- The packets are routed IN and OUT of the load balancer application
- **Proxy mode at any layer is not what we want**

Solution - DR or Direct Routing



DR or direct routing:

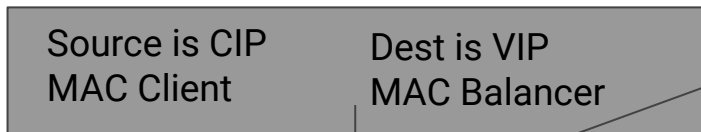
routing by MAC address at Layer 2

DR mode works by changing the MAC address of the inbound packets to match the Real Server selected by the load balancing algorithm.

DR - Packet Flow Analysis

CIP : Client IP
VIP : Virtual IP

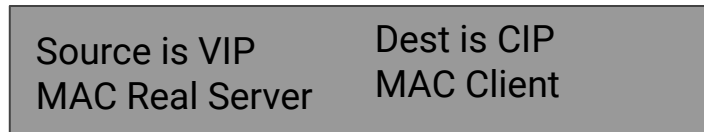
Packet to Balancer



Balancer

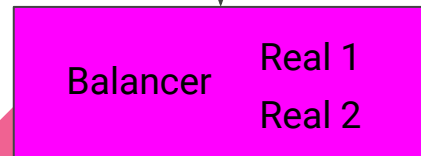
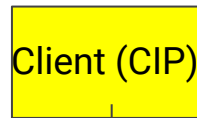


Real Server



IPVS(IP virtual server) compares whether the service requested by the packet is a cluster service. If it is a cluster service, the packet is re encapsulated

The data packet re encapsulates the message (the source IP address is VIP and the target IP is CIP), transmits the response message to the physical network card through the lo interface, and then sends it out.

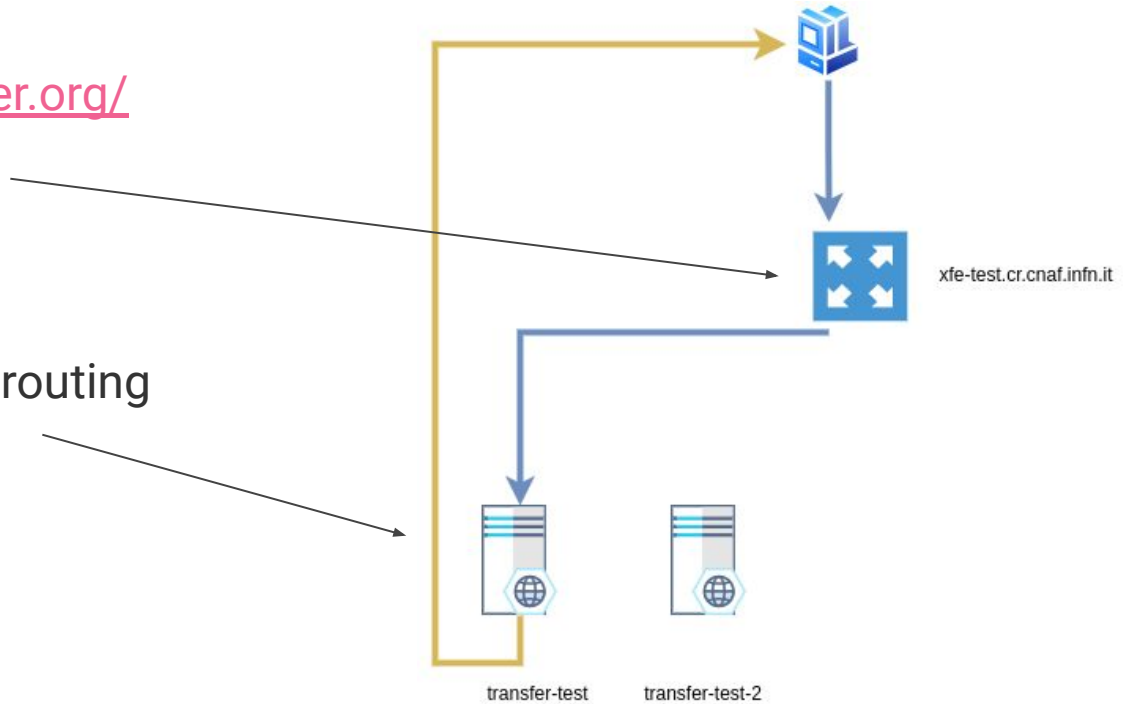


LVS

- <http://www.linuxvirtualserver.org/>

modprobe ip_vs

Add VIP on **lo** interface (no routing
outside the server)

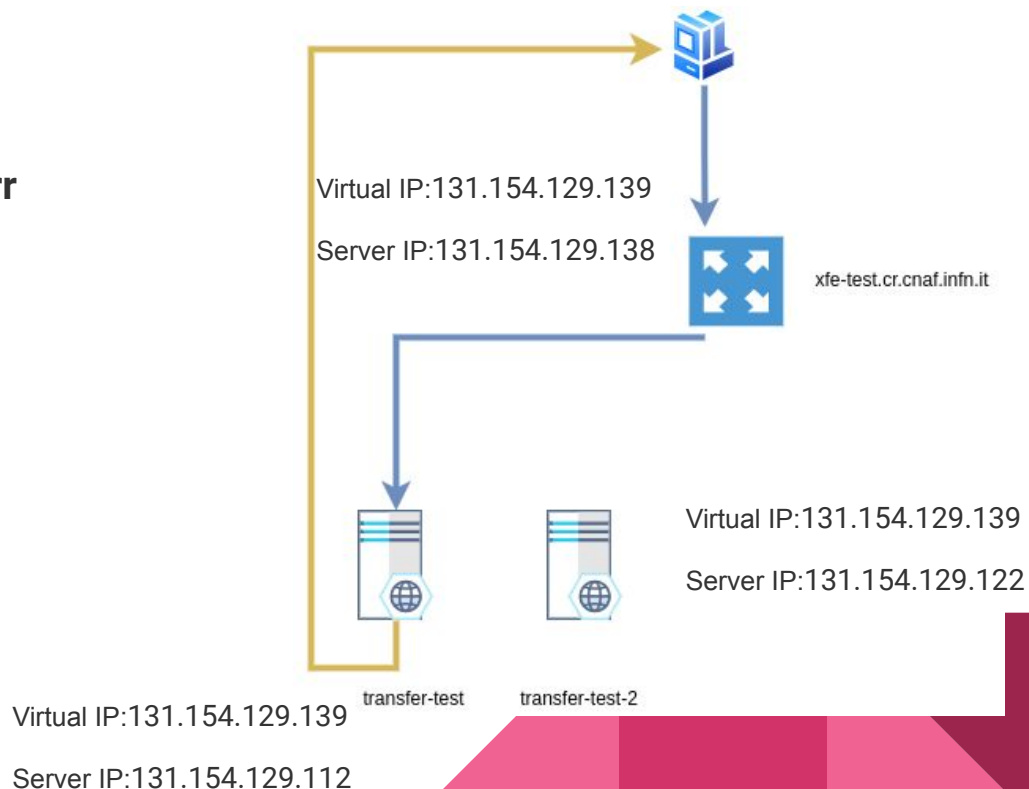


LVS - Define a service

```
ipvsadm -A -t 131.154.129.139:8443 -s rr
```

```
ipvsadm -a -t 131.154.129.139:8443 -r  
131.154.128.112:8443 -g
```

```
ipvsadm -a -t 131.154.129.139:8443 -r  
131.154.129.122:8443 -g
```



LVS - Arp problem

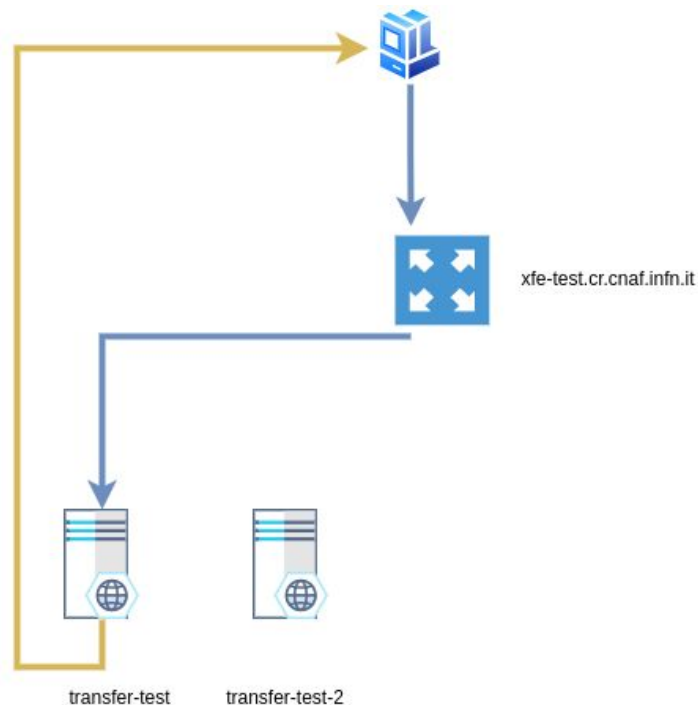
ARP replies for the interface with the VIP must be disabled

```
echo "1" > /proc/sys/net/ipv4/conf/lo/arp_ignore
```

```
echo "2" > /proc/sys/net/ipv4/conf/lo/arp_announce
```

```
echo "1" > /proc/sys/net/ipv4/conf/all/arp_ignore
```

```
echo "2" > /proc/sys/net/ipv4/conf/all/arp_announce
```

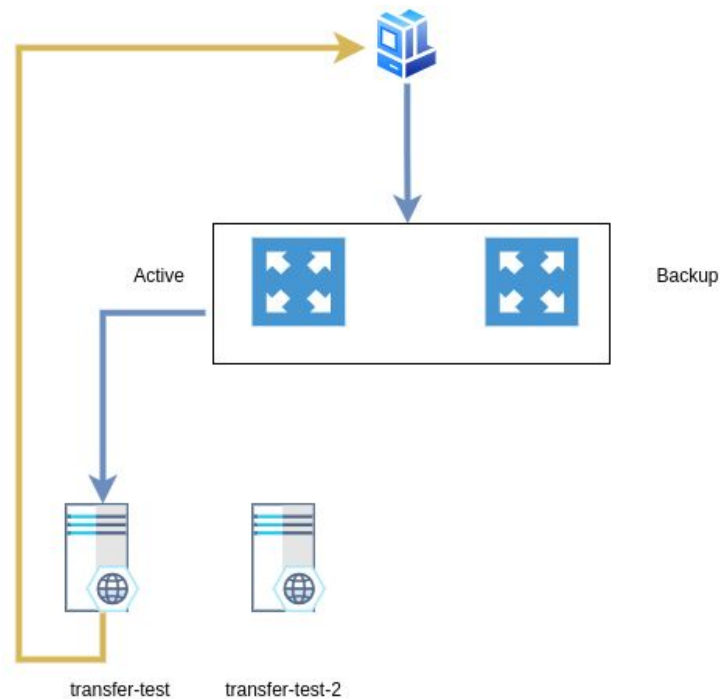


Keepalived (frontend for LVS)

- File conf of LVS
- ACTIVE / BACKUP (HA)

```
vrp_instance VI_1 {  
    state MASTER/BACKUP  
    interface eth1  
    virtual_router_id 51  
    priority 100  
    advert_int 10  
    virtual_ipaddress {  
        192.168.35.100/24  
    }  
}
```

```
virtual_server 192.168.35.100 8080 {  
    delay_loop 6  
    lb_algo rr  
    lb_kind DR  
    #persistence_timeout 5  
    protocol TCP  
  
    real_server 192.168.35.9 8080 {  
        TCP_CHECK {  
            #connect_timeout 5  
            #connect_port 8080  
        }  
    }  
    real_server 192.168.35.10 8080 {  
        TCP_CHECK {  
            #connect_timeout 5  
            #connect_port 8080  
        }  
    }  
}
```



And the Upload?

- Cannot work!
- Client is the initiator
- Packet always pass through the balancing servers and routed to the real server



A possible solution (by Carmelo Pellegrino)

- Return :
 - 301 (Tested successfully with PUT)
 - 302, 307, 308
- All these has slightly different semantics
- Easy implementation i.e. nginx or haproxy
- Still to be understood the flexibility of the method. The requirement is still to to implement a balancing algorithm different than round-robin

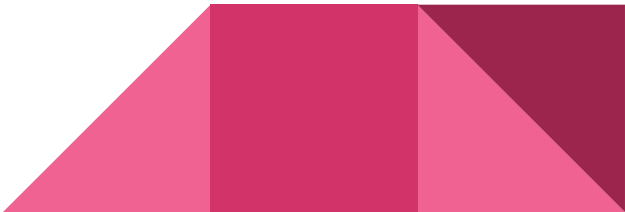


Thanks



Some docs and Reference

- <https://programmer.ink/think/building-lvs-load-balancing-cluster-direct-routing-mode-lvs-dr.html>
- <https://programs.wiki/wiki/web-cluster-load-balancing-cluster-lvs.html>
- <https://blog.katastros.com/a?ID=00700-1b168cb9-f419-4ccf-be32-8925edf4e5bf>
- <https://programming.vip/docs/deployment-of-lvs-dr-cluster-with-load-balancing.html>
- <https://debugged.it/blog/ipvs-the-linux-load-balancer/>
- https://keepalived.readthedocs.io/en/latest/configuration_synopsis.html
- <https://www.ibm.com/docs/en/i/7.1?topic=methods-routing-virtual-ip>



LVS Algos

`rr` Round robin scheduling(Round-Robin)

Note: assign requests to different in turn RS That is, in RS Request for equal sharing in.

`wrr` Weighted round robin scheduling(Weighted Round-Robin)


Note: according to different RS Weight assignment task. Higher weight RS Priority will be given to the task, and the number of connections assigned will be lower than the weight RS more.

`dh` Destination hash scheduling(Destination Hashing)

Description: use the destination address as the keyword to find a static address hash Table to get the required RS.

`sh` Source address hash scheduling(source hashing)

Description: use the source address as the keyword to find a static address hash Table to get the required RS.



LVS Algos

wlc Weighted minimum connection scheduling(weighted leastconnection)

Note: suppose each set RS The weights of are $w_i (i=1..n)$ Current TCP The number of connections is $T_i (i=1..n)$, Select in turn T_i/W_i Is the smallest RS As next assigned RS.

lc Minimum connection scheduling(Least-Connection)

Description: IPVS The table stores all active connections. Send a new connection request to the one with the smallest number of current connections RS.

lblc Address based minimum connection scheduling(locality-Based Least-Connection)

Note: assign requests from the same destination address to the same computer RS If this server is not fully loaded, it will be allocated to the server with the smallest number of connections RAnd take it as the first consideration for the next distribution.



LVS Algos

lb_lcr Scheduling based on the minimum number of repeated connections in address band(Locality-Based Least-Connection with Replication)

Note: for a destination address, there is one corresponding to it RS Subset. The minimum number of connections in the allocation subset for this address request RS;

If all subsets in the server are fully loaded, select a server with a small number of connections from the cluster, add it to this subset and allocate connections;

If no modification has been made within a certain period of time, the node with the largest load in the subset will be deleted from the subset.

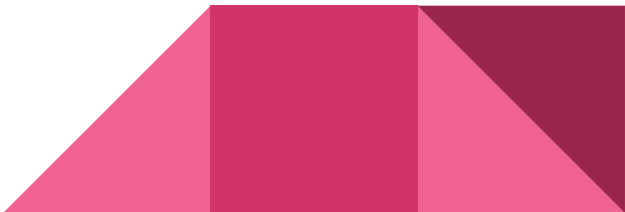
SED Minimum expected delay(shortest expected delay scheduling SED)

Description: Based on wlc Algorithm. for example ABC The weight of the three machines is 123 and the number of connections is 123 respectively. So if you use wlc Algorithm, when a new request enters, it may be distributed to ABC Any one of them.

use sed Such an operation will be carried out after the algorithm $A(1+1)/1$ $B(1+2)/2$ $C(1+3)/3$ According to the calculation result, give the connection to C.

NQ Least queue scheduling(Never Queue Scheduling NQ)

Description: no queue is required. If there is one realserver Number of connections=0 Just distribute it directly in the past, and it doesn't need to be carried out in the future sed operation



Load Balancing VS High Availability

Load Balancing: Load balancing is the process of spreading a system over multiple machines. This process typically has the goal of making a system more scalable by leveraging more than a single server.

High Availability: High availability is more of a feature than a process. Essentially high availability means that if one of a system's components goes down, it won't bring the entire system down with it

