Table of Contents

HAProxy Debian 3
Optional step 3
www servers 3
haproxy servers 3
Types of load balancers 4
Types of load balancer algorithms 4
Layer 4 Load Balancer 4
Play with ACLs 7
Layer 7 Load Balancer 9
Make HAProxy Fault Tolerant 13
Database configuration 15
References 17

HAProxy Debian

- Install Debian 12 with SSH and no more
- Create two templates, haproxy-machote and www-machote

Optional step

If you are using VM this step will save aprox 200MB of space on every VM.

- Determine the name of the old kernel:

```
dpkg -l |grep linux-im
```

- Do the step as follow, replace linux-image-6.1.0-15-amd64 with the correct old image (take care to avoid to remove the actual working image)

```
apt-get remove --purge discover discover-data anacron apparmor emacsen-
common iw libdiscover2 ienglish-common ispell wamerican wireless-regdb
dictionaries-common libasound2-data alsa-topology-conf libdw1 libglib2.0-0
libglib2.0-data libicu72 libxml2 shared-mime-info xdg-user-dirs
installation-report alsa-ucm-conf wpasupplicant libnl-route-3-200
libpcsclite1 dmidecode eject intel-microcode iucode-tool laptop-detect
linux-image-6.1.0-15-amd64 firmware-linux-free nano busybox sudo powertop
libnl-3-200 libnl-genl-3-200 nftables locales task-english pciutils libpci3
pci.ids tasksel tasksel-data wireless-tools libiw30 libx11-data libx11-data
libx11-6 libxmuu1 xauth libxau6 libxcb1 libxdmcp6 whiptail zstd  xkb-data
console-setup console-setup-linux keyboard-configuration usbutils
libusb-1.0-0 libnuma1 liburing2
```

www servers

Follow the steps:

```
apt-get -y install sudo vim rsyslog
sudo apt-get -y install curl gnupg2 ca-certificates lsb-release debian-
archive-keyring
curl https://nginx.org/keys/nginx_signing.key | gpg --dearmor \
    | sudo tee /usr/share/keyrings/nginx-archive-keyring.gpg >/dev/null
echo "deb [signed-by=/usr/share/keyrings/nginx-archive-keyring.gpg] \
http://nginx.org/packages/debian `lsb_release -cs` nginx" \
    | sudo tee /etc/apt/sources.list.d/nginx.list
sudo apt-get update
sudo apt-get -y install nginx
```

haproxy servers

Follow the steps:

```
apt-get install sudo vim  rsyslog
sudo apt-get install curl gpg
curl https://haproxy.debian.net/bernat.debian.org.gpg \
      | gpg --dearmor > /usr/share/keyrings/haproxy.debian.net.gpg
echo deb "[signed-by=/usr/share/keyrings/haproxy.debian.net.gpg]" \
      http://haproxy.debian.net bookworm-backports-2.9 main \
      > /etc/apt/sources.list.d/haproxy.list
apt-get update
apt-get install haproxy=2.9.\*
```

Types of load balancers

- Layer 4: forward HTTP traffic to other servers
- Access Control List: allows forwarding traffic based on a certain pattern in content of user request
- Layer 7: known as HTTP, evalues the headers and fordward the traffic to other servers based on its content

This lab will play with them.

Types of load balancer algorithms

- Round-Robin – default algorithm which selects servers in a rotating basis.
- Least Connect – selects servers based on the least number of connections
- Source – selects servers based on a hash of the source IP such as the user IP address to ensure request goes to the same server until something changes in the hash (i.e. one backend server goes down).
- Sticky Sessions – enables persistence in order for applications to connect to same backend server to process requests.

Layer 4 Load Balancer

- This type of load balancer is better for other services than HTTP servers, but I am lazy, I don´t want to install mysql or something like that.
- Clone www-machote to www1 and www2
- Clone haproxy-machote to haproxy1
- Follow steps on haproxy1:

```
hostnamectl set-hostname haproxy1
mv /etc/haproxy/haproxy.cfg /etc/haproxy/haproxy.cfg.BAK
cat << EOF > /etc/haproxy/haproxy.cfg
global
    log /dev/log    local0
    log /dev/log    local1 notice
    chroot /var/lib/haproxy
    stats socket /run/haproxy/admin.sock mode 660 level admin
    stats timeout 30s
    user haproxy
```

```
    group haproxy
    daemon

    # Default SSL material locations
    ca-base /etc/ssl/certs
    crt-base /etc/ssl/private

    # See:
https://ssl-config.mozilla.org/#server=haproxy&server-version=2.0.3&config=i
ntermediate
        ssl-default-bind-ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-
AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-
SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-
AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384
        ssl-default-bind-ciphersuites
TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256
        ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tickets

defaults
    log    global
    mode    tcp

frontend www
    bind    :80
    default_backend www_servers

backend www_servers
    server www1 192.168.122.135:80
    server www2 192.168.122.235:80
EOF
haproxy -c -f /etc/haproxy/haproxy.cfg
systemctl restart haproxy
```

- On server www1:

```
hostnamectl set-hostname www1
```

- On server www2:

```
hostnamectl set-hostname www2
```

- Follow steps on www1 and www2:

```
mv /usr/share/nginx/html/index.html /usr/share/nginx/html/index.html.BAK
echo $(hostname) > /usr/share/nginx/html/index.html
```

- Open a console and make the following command (you can use a web browser but for some reason stick to one server sometimes), replace haproxy1 with the ip of your vm:

```
em1069@angellodebiansofia:~$ curl http://haproxy1
www1
em1069@angellodebiansofia:~$ curl http://haproxy1
```

```
www2
em1069@angellodebiansofia:~$ curl http://haproxy1
www1
em1069@angellodebiansofia:~$
```

- Now on server www1

```
systemctl stop nginx
```

- Open a console and make the following command (you can use a web browser but for some reason stick to one server sometimes), replace haproxy1 with the ip of your vm:

```
em1069@angellodebiansofia:~$ curl http://haproxy1
www2
em1069@angellodebiansofia:~$ curl http://haproxy1
curl: (52) Empty reply from server
em1069@angellodebiansofia:~$ curl http://haproxy1
www2
em1069@angellodebiansofia:~$ curl http://haproxy1
curl: (52) Empty reply from server
em1069@angellodebiansofia:~$
```

- Is not fault tolerant, let fix that, on haproxy1 follow the steps:

```
cat << EOF > /etc/haproxy/haproxy.cfg
global
    log /dev/log    local0
    log /dev/log    local1 notice
    chroot /var/lib/haproxy
    stats socket /run/haproxy/admin.sock mode 660 level admin
    stats timeout 30s
    user haproxy
    group haproxy
    daemon

    # Default SSL material locations
    ca-base /etc/ssl/certs
    crt-base /etc/ssl/private

    # See:
https://ssl-config.mozilla.org/#server=haproxy&server-version=2.0.3&config=i
ntermediate
        ssl-default-bind-ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-
AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-
SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-
AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384
        ssl-default-bind-ciphersuites
TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256
        ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tickets

defaults
    log    global
```

```
    mode    tcp

frontend www
    bind    :80
    default_backend www_servers

backend www_servers
    server www1 192.168.122.135:80 check
    server www2 192.168.122.235:80 check
EOF
haproxy -c -f /etc/haproxy/haproxy.cfg
systemctl restart haproxy
```

- Open a console and make the following command (you can use a web browser but for some reason stick to one server sometimes), replace haproxy1 with the ip of your vm:

```
em1069@angellodebiansofia:~$ curl http://haproxy1
www2
em1069@angellodebiansofia:~$ curl http://haproxy1
www2
em1069@angellodebiansofia:~$ curl http://haproxy1
www2
em1069@angellodebiansofia:~$ curl http://haproxy1
www2
em1069@angellodebiansofia:~$
```

- Now on server www1

```
systemctl start nginx
```

- Open a console and make the following command (you can use a web browser but for some reason stick to one server sometimes), replace haproxy1 with the ip of your vm:

```
em1069@angellodebiansofia:~$ curl http://haproxy1
www1
em1069@angellodebiansofia:~$ curl http://haproxy1
www2
em1069@angellodebiansofia:~$ curl http://haproxy1
www1
em1069@angellodebiansofia:~$
```

Play with ACLs

- On server haproxy1:

```
cat << EOF > /etc/haproxy/haproxy.cfg
global
    log /dev/log    local0
    log /dev/log    local1 notice
    chroot /var/lib/haproxy
    stats socket /run/haproxy/admin.sock mode 660 level admin
```

```
    stats timeout 30s
    user haproxy
    group haproxy
    daemon

    # Default SSL material locations
    ca-base /etc/ssl/certs
    crt-base /etc/ssl/private

    # See:
https://ssl-config.mozilla.org/#server=haproxy&server-version=2.0.3&config=i
ntermediate
        ssl-default-bind-ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-
AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-
SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-
AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384
        ssl-default-bind-ciphersuites
TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256
        ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tickets

defaults
    log    global
    mode    tcp

frontend www
    bind    :80
    acl server_url_www1 path_beg -i /www1
    use_backend www1_backend if server_url_www1
    acl server_url_www2 path_beg -i /www2
    use_backend www2_backend if server_url_www2

backend www1_backend
    server www1 192.168.122.135

backend www2_backend
    server www2 192.168.122.235
EOF
haproxy -c -f /etc/haproxy/haproxy.cfg
systemctl restart haproxy
```

- On server www1 and www2 do:

```
mkdir /usr/share/nginx/html/$(hostname)
cp /usr/share/nginx/html/index.html /usr/share/nginx/html/$(hostname)/
```

- Open a console and make the following command (you can use a web browser but for some reason stick to one server sometimes), replace haproxy1 with the ip of your vm:

```
em1069@angellodebiansofia:~$ curl -L http://haproxy1/www1
www1
em1069@angellodebiansofia:~$ curl -L http://haproxy1/www2
```

```
www2
```

It could fail several times you need to give time to HAProxy to get its stuff ready.

Layer 7 Load Balancer

- On server haproxy1

```
cat << EOF > /etc/haproxy/haproxy.cfg
global
    log /dev/log    local0
    log /dev/log    local1 notice
    chroot /var/lib/haproxy
    stats socket /run/haproxy/admin.sock mode 660 level admin
    stats timeout 30s
    user haproxy
    group haproxy
    daemon

    # Default SSL material locations
    ca-base /etc/ssl/certs
    crt-base /etc/ssl/private

    # See:
https://ssl-config.mozilla.org/#server=haproxy&server-version=2.0.3&config=i
ntermediate
        ssl-default-bind-ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-
AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-
SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-
AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384
        ssl-default-bind-ciphersuites
TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256
        ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tickets

defaults
    log    global
    mode    http

frontend www
    bind    :80
        default_backend www_servers

backend www_servers
    option httpchk
    server www1 192.168.122.135:80 check
    server www2 192.168.122.235:80 check
EOF
haproxy -c -f /etc/haproxy/haproxy.cfg
systemctl restart haproxy
```

- Open a console and make the following command (you can use a web browser but for some reason stick to one server sometimes), replace haproxy1 with the ip of your vm:

```
em1069@angellodebiansofia:~$ curl http://haproxy1
<html><body><h1>503 Service Unavailable</h1>
No server is available to handle this request.
</body></html>
```

- Is not working =(. Check why on HAProxy log on server haproxy1:

```
root@debian:/etc/haproxy# tail -f /var/log/haproxy.log
2024-03-10T22:05:16.677484-06:00 debian haproxy[1710]: [WARNING]  (1710) :
Server www_servers/www1 is DOWN, reason: Layer7 wrong status, code: 405,
info: "Not Allowed", check duration: 3ms. 1 active and 0 backup servers
left. 0 sessions active, 0 requeued, 0 remaining in queue.
2024-03-10T22:05:16.677877-06:00 debian haproxy[1710]: Server
www_servers/www1 is DOWN, reason: Layer7 wrong status, code: 405, info: "Not
Allowed", check duration: 3ms. 1 active and 0 backup servers left. 0
sessions active, 0 requeued, 0 remaining in queue.
2024-03-10T22:05:16.678090-06:00 debian haproxy[1710]: Server
www_servers/www1 is DOWN, reason: Layer7 wrong status, code: 405, info: "Not
Allowed", check duration: 3ms. 1 active and 0 backup servers left. 0
sessions active, 0 requeued, 0 remaining in queue.
2024-03-10T22:05:17.653011-06:00 debian haproxy[1710]: [WARNING]  (1710) :
Server www_servers/www2 is DOWN, reason: Layer7 wrong status, code: 405,
info: "Not Allowed", check duration: 0ms. 0 active and 0 backup servers
left. 0 sessions active, 0 requeued, 0 remaining in queue.
```

- Check nginx log on www1 or www2 server:

```
root@www1:~# tail -f /var/log/nginx/access.log
192.168.122.16 - - [10/Mar/2024:22:12:03 -0600] "OPTIONS / HTTP/1.0" 405 157
"-" "-" "-"
192.168.122.16 - - [10/Mar/2024:22:12:05 -0600] "OPTIONS / HTTP/1.0" 405 157
"-" "-" "-"
192.168.122.16 - - [10/Mar/2024:22:12:07 -0600] "OPTIONS / HTTP/1.0" 405 157
"-" "-" "-"
192.168.122.16 - - [10/Mar/2024:22:12:09 -0600] "OPTIONS / HTTP/1.0" 405 157
"-" "-" "-"
```

- HAProxy is requesting a call not allowed by the OPTIONS method.
- Reconfigure HAProxy as follows on server haproxy1:

```
cat <<EOF> /etc/haproxy/haproxy.cfg
global
    log /dev/log    local0
    log /dev/log    local1 notice
    chroot /var/lib/haproxy
    stats socket /run/haproxy/admin.sock mode 660 level admin
    stats timeout 30s
    user haproxy
    group haproxy
```

```
    daemon

    # Default SSL material locations
    ca-base /etc/ssl/certs
    crt-base /etc/ssl/private

    # See:
https://ssl-config.mozilla.org/#server=haproxy&server-version=2.0.3&config=i
ntermediate
        ssl-default-bind-ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-
AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-
SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-
AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384
        ssl-default-bind-ciphersuites
TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256
        ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tickets

defaults
    log    global
    mode    http

frontend www
    bind    :80
        default_backend www_servers

backend www_servers
    option httpchk GET /
    server www1 192.168.122.135:80 check
    server www2 192.168.122.235:80 check
EOF
haproxy -c -f /etc/haproxy/haproxy.cfg
systemctl restart haproxy
```

- Open a console and make the following command (you can use a web browser but for some reason stick to one server sometimes), replace haproxy1 with the ip of your vm:

```
em1069@angellodebiansofia:~$ curl http://haproxy1
www1
em1069@angellodebiansofia:~$ curl http://haproxy1
www2
```

* Port 80 can be open and web server could server a web page, but it is that sufficient to determine everything is correct?... in our case we expect that the server show the hostname, either www1 or www2, so let make sure that HAProxy check that to determine if NGINX is serving the correct webpage. On server haproxy1 do as follows:

```
cat <<EOF> /etc/haproxy/haproxy.cfg
global
    log /dev/log    local0
    log /dev/log    local1 notice
    chroot /var/lib/haproxy
```

```
    stats socket /run/haproxy/admin.sock mode 660 level admin
    stats timeout 30s
    user haproxy
    group haproxy
    daemon

    # Default SSL material locations
    ca-base /etc/ssl/certs
    crt-base /etc/ssl/private

    # See:
https://ssl-config.mozilla.org/#server=haproxy&server-version=2.0.3&config=i
ntermediate
        ssl-default-bind-ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-
AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-
SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-
AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384
        ssl-default-bind-ciphersuites
TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256
        ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tickets

defaults
    log    global
    mode    http

frontend www
    bind    :80
        default_backend www_servers

backend www_servers
    option httpchk GET /
    http-check expect string www
    server www1 192.168.122.135:80 check
    server www2 192.168.122.235:80 check
EOF
haproxy -c -f /etc/haproxy/haproxy.cfg
systemctl restart haproxy
```

- On server www1 do as follows:

```
echo "I love you Gohan. Att: $(hostname)" > /usr/share/nginx/html/index.html
```

- On server www2 do as follows:

```
echo 'I love you Gohan. Att: $(hostname)' > /usr/share/nginx/html/index.html
```

- Open a console and make the following command (you can use a web browser but for some reason stick to one server sometimes), replace haproxy1 with the ip of your vm:

```
em1069@angellodebiansofia:~$ curl http://haproxy1
I love you Gohan. Att: www1
em1069@angellodebiansofia:~$ curl http://haproxy1
```

```
I love you Gohan. Att: www1
em1069@angellodebiansofia:~$ curl http://haproxy1
I love you Gohan. Att: www1
```

- It's not working. Check the web page on server www2. Do as follows:

```
root@www2:~# cat /usr/share/nginx/html/index.html
I love you Gohan. Att: $(hostname)
root@www2:~#
```

- We put single quotes instead double quotes. Now repeat the step with the correct command on server www2 as follows:

```
echo "I love you Gohan. Att: $(hostname)" > /usr/share/nginx/html/index.html
```

- Open a console and make the following command (you can use a web browser but for some reason stick to one server sometimes), replace haproxy1 with the ip of your vm:

```
em1069@angellodebiansofia:~$ curl http://haproxy1
I love you Gohan. Att: www1
em1069@angellodebiansofia:~$ curl http://haproxy1
I love you Gohan. Att: www2
em1069@angellodebiansofia:~$ curl http://haproxy1
I love you Gohan. Att: www1
em1069@angellodebiansofia:~$ curl http://haproxy1
I love you Gohan. Att: www2
```

Make HAProxy Fault Tolerant

- Clone server haproxy1 to haproxy2
- Do as follows on both:

```
apt-get install keepalived psmisc
```

- haproxy1 will be master node and obviously haproxy2 the slave
- On server haproxy1 (make sure to change the network interface as needed):

```
cat <<EOF> /etc/keepalived/keepalived.conf
Master Node
global_defs {
  router_id haproxy1 # The hostname of this host.
}
vrrp_script haproxy {
  script "killall -0 haproxy"
  interval 2
  weight 2
}
vrrp_instance 50 {
  virtual_router_id 50
  advert_int 1
  priority 50
```

```
  state MASTER
  interface enp1s0
  virtual_ipaddress {
     192.168.122.50 dev enp1s0
  }
  track_script {
      haproxy
  }
}
EOF
systemctl restart keepalived
```

- Check on haproxy1 the IP address as follows:

```
root@haproxy1# ip a | grep enp1s0
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state
UP group default qlen 1000
    inet 192.168.122.16/24 brd 192.168.122.255 scope global dynamic enp1s0
    inet 192.168.122.50/32 scope global enp1s0
```

- On server haproxy2:

```
cat <<EOF> /etc/keepalived/keepalived.conf
global_defs {
  router_id haproxy1 # The hostname of this host!
}
vrrp_script haproxy {
  script "killall -0 haproxy"
  interval 2
  weight 2
}
vrrp_instance 50 {
  virtual_router_id 50
  advert_int 1
  priority 50
  state BACKUP
  interface enp1s0
  virtual_ipaddress {
     192.168.122.50 dev enp1s0
  }
  track_script {
    haproxy
  }
}
EOF
systemctl restart keepalived
```

- Test if everything is working as expected, on server haproxy1 shutdown HAProxy:

```
systemctl stop haproxy
```

- Check on haproxy1 the IP addresses as follows:

```
root@haproxy1# ip a | grep enp1s0
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state
UP group default qlen 1000
    inet 192.168.122.16/24 brd 192.168.122.255 scope global dynamic enp1s0
```

- Floating IP is gone. Check on haproxy2 the IP addresses as follows:

```
root@haproxy2:/etc/keepalived# ip a |grep enp1s0
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state
UP group default qlen 1000
    inet 192.168.122.62/24 brd 192.168.122.255 scope global dynamic enp1s0
    inet 192.168.122.50/32 scope global enp1s0
root@haproxy2:/etc/keepalived#
```

- IP address was moved.

Database configuration

- By critical acclaim... my lazyness was rejected... public insisted on a DB escenario, MariaDB, clone the clean template to db-machote. Do as follows

```
apt-get -y install mariadb-server sudo vim rsyslog
```

- Clone db-machote to db1 and db2
- On server db1:

```
hostnamectl set-hostname db1
echo -e "\n\n\n\manager\nmanager\n\n\n\n\n" | mysql_secure_installation
cat <<EOF> /etc/mysql/mariadb.conf.d/replication.cnf
[mariadb]
log-bin
server_id=1
log-basename=master1
binlog-format=mixed
EOF
cat <<EOF> /root/.my.cnf
[mysql]
user=root
password=manager
EOF
echo "CREATE USER 'replication_user'@'%' IDENTIFIED BY 'manager';" | mysql
echo "GRANT REPLICATION SLAVE ON *.* TO 'replication_user'@'%';" | mysql
```

- Replace bind-address on /etc/mysql/mariadb.conf.d/50-server.cnf
- Restart MariaDB

```
systemctl restart mariadb
```

- Run the following commands on db1, take note of File and Position rows:

```
echo -e "FLUSH TABLES WITH READ LOCK;" | mysql
```

```
echo -e "show master status;" |mysql
```

- On server db2:

```
hostnamectl set-hostname db2
echo -e "\n\n\n\manager\nmanager\n\n\n\n\n" | mysql_secure_installation
cat <<EOF> /etc/mysql/mariadb.conf.d/replication.cnf
[mariadb]
log-bin
server_id=2
log-basename=slave1
binlog-format=mixed
EOF
cat <<EOF> /root/.my.cnf
[mysql]
user=root
password=manager
EOF
```

- Restart MariaDB

```
systemctl restart mariadb
```

- Run the following commands on db2, take note of replace File and Position data from previous commands:

```
echo "CHANGE MASTER TO MASTER_HOST='192.168.122.135',
MASTER_USER='replication_user', MASTER_PASSWORD='manager',
MASTER_PORT=3306, MASTER_LOG_FILE='master1-bin.000003',
MASTER_LOG_POS=874,   MASTER_CONNECT_RETRY=10;"| mysql
echo -e "START SLAVE;"| mysql
echo -e "SHOW SLAVE STATUS \G" | mysql
```

- On server db1 do as follows:

```
UNLOCK TABLES;
```

- Tadaaaa
- Check to replicate the creation of one database. On server db1:

```
echo -e "SHOW DATABASES" | mysql
```

- On server db2:

```
echo -e "SHOW DATABASES" | mysql
```

- On server db1:

```
echo -e "CREATE DATABASE helloworld" | mysql
echo -e "SHOW DATABASES" | mysql
```

- On server db2:

```
echo -e "SHOW DATABASES" | mysql
```

- Great you can see the database replicated on db2

References

- https://docs.nginx.com/nginx/admin-guide/installing-nginx/installing-nginx-open-source/
- https://haproxy.debian.net/#distribution=Debian&release=bookworm&version=2.9
- https://infohub.delltechnologies.com/en-US/l/ecs-with-haproxy-load-balancer-2/haproxy-overview/
- https://www.haproxy.com/blog/layer-4-and-layer-7-proxy-mode
- https://www.digitalocean.com/community/tutorials/an-introduction-to-haproxy-and-load-balancing-concepts
- https://www.haproxy.com/blog/testing-your-haproxy-configuration
- https://www.warp.dev/terminus/curl-follow-redirect
- https://www.haproxy.com/documentation/haproxy-configuration-tutorials/core-concepts/acls/
- https://www.haproxy.com/blog/haproxy-configuration-basics-load-balance-your-servers
- https://www.haproxy.com/blog/how-to-enable-health-checks-in-haproxy
- https://www.haproxy.com/documentation/haproxy-configuration-tutorials/service-reliability/health-checks/
- https://www.haproxy.com/blog/introduction-to-haproxy-logging
- https://forum.netgate.com/topic/94060/solved-basic-usage-of-haproxy
- https://blog.taylorbuiltsolutions.com/haproxy-nginx-health-check-method/
- https://docs.vmware.com/en/vRealize-Operations/8.10/vrops-manager-load-balancing/GUID-EC001888-776B-42D5-9843-719EF08AB940.html
- https://sysadmins.co.za/achieving-high-availability-with-haproxy-and-keepalived-building-a-redundant-load-balancer/
- https://mariadb.com/kb/en/setting-up-replication/
- https://gist.github.com/vdvm/24754bf1aee6fd85e1aa
- https://stackoverflow.com/questions/17827340/protect-the-password-when-using-mysql-in-an-unattended-bash-script