

HAPROXY

Compte rendu

Année Scolaire :
2022 / 2023

COLBAUT Mathis
BTS SIO – Option SLAM

SOMMAIRE :



QU'EST-CE QUE HAPROXY.....2



PROCEDURE DE MISE EN PLACE.....3



TEST BALANCING.....11



CONCLUSION.....13



ANNEXE / PLAN RESEAU.....14

Qu'est ce que HaProxy



HAProxy (High Availability Proxy) est un logiciel open-source qui agit comme un proxy et un équilibreur de charge (load balancer) pour les applications web. Il est conçu pour améliorer la disponibilité, les performances et la résilience des sites web, en distribuant le trafic entre plusieurs serveurs de manière efficace et en redirigeant automatiquement le trafic en cas de défaillance d'un serveur.

HAProxy peut être utilisé pour gérer le trafic web pour une grande variété d'applications, notamment les sites de commerce électronique, les applications de messagerie instantanée, les sites de médias sociaux et les applications en temps réel. Il prend en charge de nombreux protocoles, notamment HTTP, HTTPS, TCP et UDP, et peut être configuré pour offrir une haute disponibilité et une scalabilité horizontale pour les applications web.

HAProxy est largement utilisé dans les environnements de production pour améliorer la disponibilité et les performances des applications web à haute charge. Il est considéré comme l'un des meilleurs équilibreurs de charge open-source disponibles aujourd'hui, offrant des fonctionnalités avancées telles que la mise en cache de contenu, la compression de données, la limitation de taux et la gestion de la session.

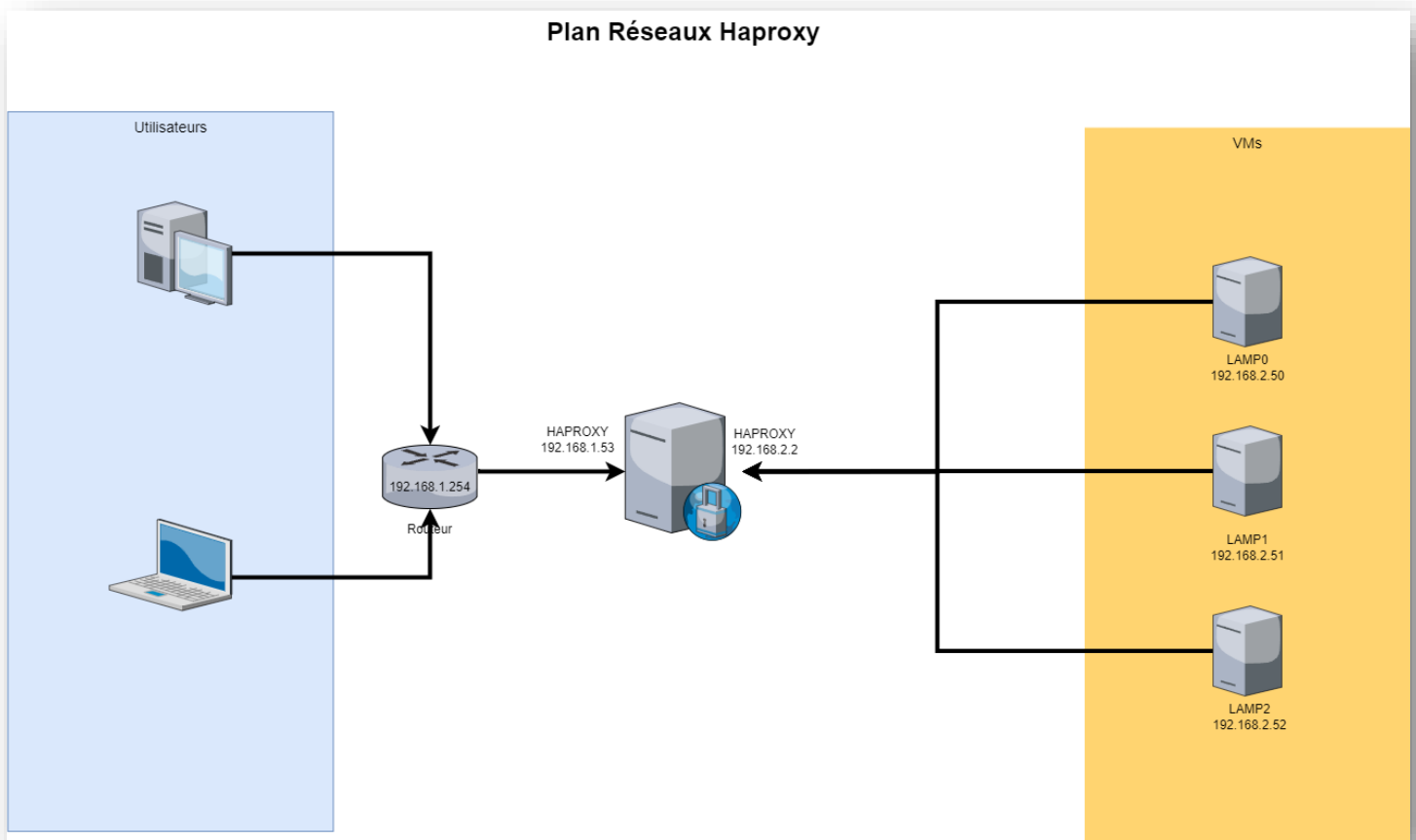
Procédure de mise en place



1.1 Prérequis.

Pour installer un serveur HaProxy il vous faudra au minimum 3 VMs (1 haproxy, 2 serveurs web apache).

Je vais reprendre la même configuration que ma documentation Galera Cluster donc 3 vm qui seront 3 serveurs web. Et j'ajoute une nouvelle VM ou nous aurons l'installation de HAPROXY.

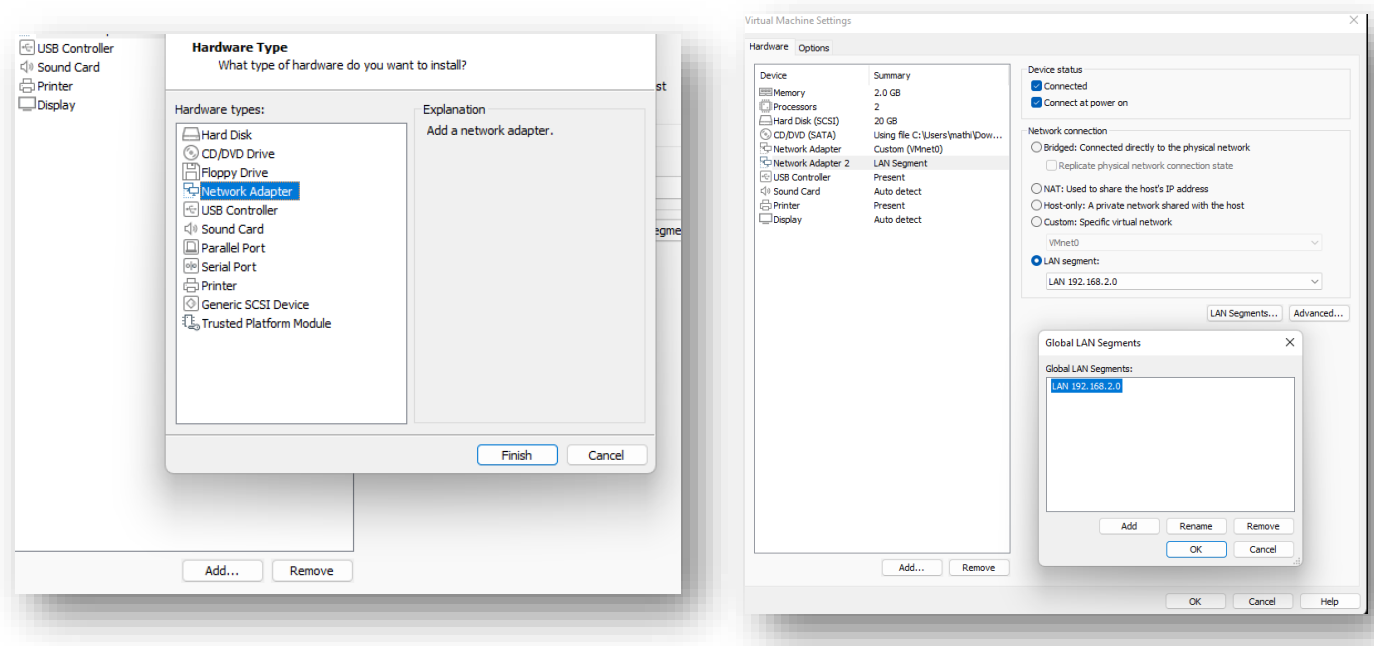


Attention :

La VM haproxy a besoin de deux interfaces réseaux une en nat et une en LAN segment.

Car haproxy aura une patte rattachée aux serveurs web et une patte rattachée à internet là où l'utilisateur aura accès.

Pour cela sur VmWare ajouter un adaptateur réseau en lan segment



Maintenant que nous avons deux interfaces sur la VM HAPROXY.

Nous allons attribuer à la VM Haproxy les IP.

Faites un :

```
netplan/
root@HAPROXY:~# sudo nano /etc/netplan/00-installer-config.yaml
```

Et mettre cette configuration réseaux (à adapter) :

```
GNU nano 6.2 /etc/netplan/00-installer-config.yaml
# This is the network config written by 'subiquity'
network:
  ethernets:
    ens33:
      dhcp4: false
      addresses: [192.168.1.53/24]
      gateway4: 192.168.1.254
      nameservers:
        addresses: [1.1.1.1,1.0.0.1]
    ens37:
      dhcp4: false
      addresses: [192.168.2.2/24]
```

Ici nous venons dire que l'interface réseaux ens33 (la nat) aura l'ip 192.168.1.53 et sa gateway 192.168.1.254 soit pour ma part la passerelle de ma box Free.

Nous avons également configurer notre deuxième interface la ens37 (le lan segment) et nous lui attribuons l'adresse 192.168.2.2 qui sera pour nos serveur web leur passerelle.

Pour connaître les noms des interfaces car ils peuvent différer faites un :

Ip addr show

```
root@HAPROXY:~# ip show interface
Object "show" is unknown, try "ip help".
root@HAPROXY:~# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:4a:2d:a1 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.1.53/24 brd 192.168.1.255 scope global ens33
        valid_lft forever preferred_lft forever
    inet6 2a01:e0a:3b1:db50:20c:29ff:fe4a:2da1/64 scope global dynamic mngtmpaddr noprefixroute
        valid_lft 86120sec preferred_lft 86120sec
    inet6 fe80::20c:29ff:fe4a:2da1/64 scope link
        valid_lft forever preferred_lft forever
3: ens37: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:4a:2d:ab brd ff:ff:ff:ff:ff:ff
    altname enp2s5
    inet 192.168.2.2/24 brd 192.168.2.255 scope global ens37
        valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe4a:2dab/64 scope link
        valid_lft forever preferred_lft forever
root@HAPROXY:~# _
```

Maintenant que notre Haproxy est accessible sur mon réseaux depuis l'ip 192.168.1.53

Nous allons configurer les serveurs web pour qu'ils puissent communiquer avec le serveur haproxy.

Sur les trois serveur web faite un :

```
netconfig netplan/ networkd=dispatcher/ networks
root@HAPROXY:~# sudo nano /etc/netplan/00-installer-config.yaml
```

Et entrer cette configuration ici le serveur LAMP2 aura l'adresse ip 192.168.2.52

Et comme vous le remarquer la gateway(passerelle) à l'adresse 192.168.2.2

```
GNU nano 6.2 /etc/netplan/00-installer-config.yaml
# This is the network config written by 'subiquity'
network:
  ethernets:
    ens33:
      dhcp4: false
      addresses: [192.168.2.52/24]
      gateway4: 192.168.2.2
      nameservers:
        addresses: [1.1.1.1,1.0.0.1]
  version: 2
```

Refaites l'opération autant de fois que vous avez de serveur web.

Et n'oubliez pas le

netplan apply

Pour actualiser l'adresse IP (sur chaque serveur).

A partir de maintenant les serveurs lamp ont accès au haproxy et le haproxy à accès à internet et aux serveurs Lamp.

1.2 Installation de HAPROXY.

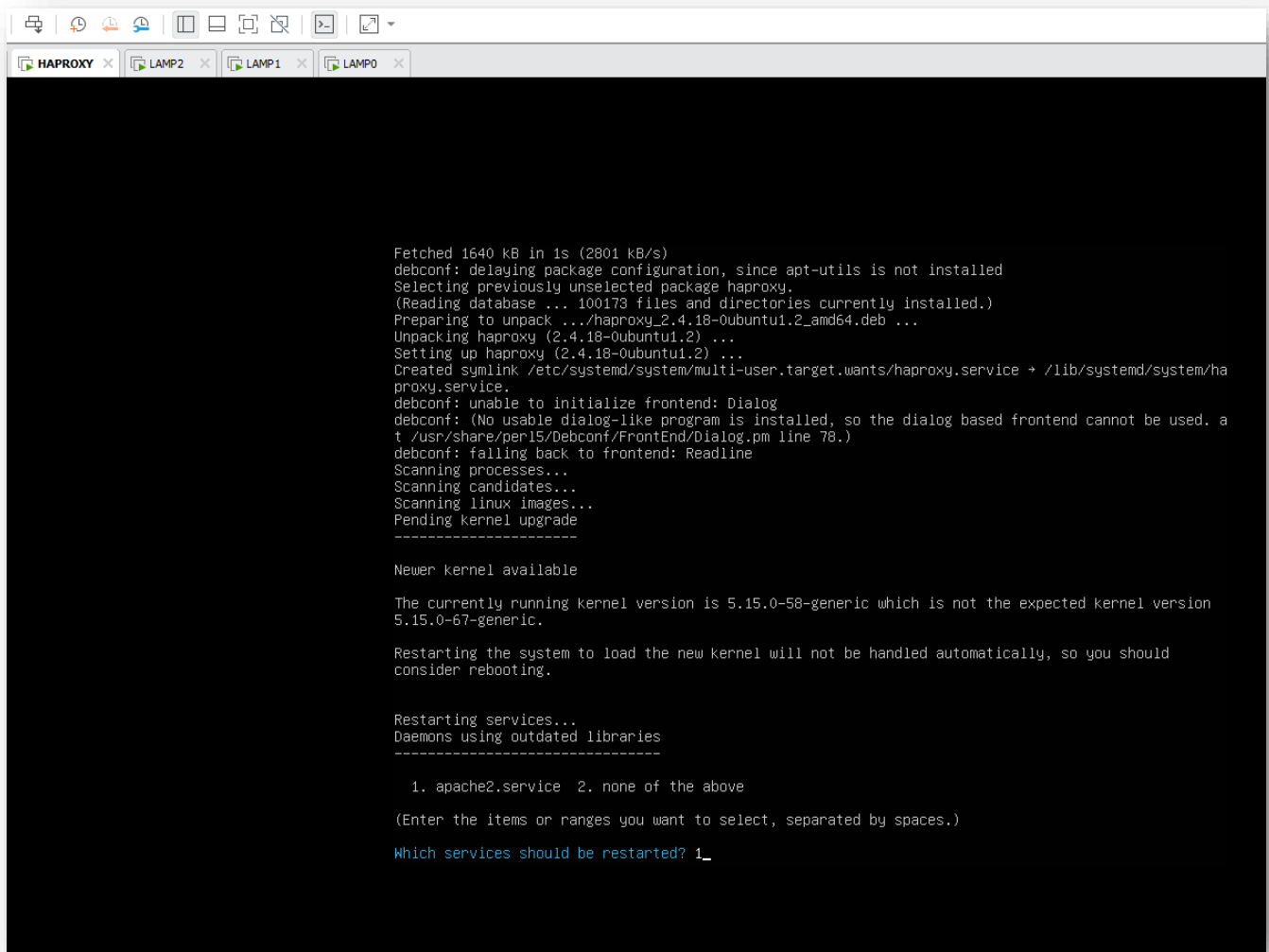
1.2.1 Si vous n'avez pas encore installer vos serveurs apache2 faite la commande suivante sur vos 3 serveurs / vm LAMP.

```
apt-get install apache2 php5
```

1.2.2 Maintenant que vos serveur on un service web installer nous allons nous occupez du serveur Haproxy ou nous allons installer le service haproxy. Pour cela exécuter cette commande :

```
apt-get install haproxy
```

A la fin il vous demandera de choisir votre serveur web sélectionner bien apache2.



```
Fetches 1640 kB in 1s (2801 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package haproxy.
(Reading database ... 100173 files and directories currently installed.)
Preparing to unpack .../haproxy_2.4.18-0ubuntu1.2_amd64.deb ...
Unpacking haproxy (2.4.18-0ubuntu1.2) ...
Setting up haproxy (2.4.18-0ubuntu1.2) ...
Created symlink /etc/systemd/system/multi-user.target.wants/haproxy.service → /lib/systemd/system/haproxy.service.
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based frontend cannot be used. a
t /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 78.)
debconf: falling back to frontend: Readline
Scanning processes...
Scanning candidates...
Scanning linux images...
Pending kernel upgrade
-----

Newer kernel available

The currently running kernel version is 5.15.0-58-generic which is not the expected kernel version
5.15.0-67-generic.

Restarting the system to load the new kernel will not be handled automatically, so you should
consider rebooting.

Restarting services...
Daemons using outdated libraries
-----

1. apache2.service 2. none of the above

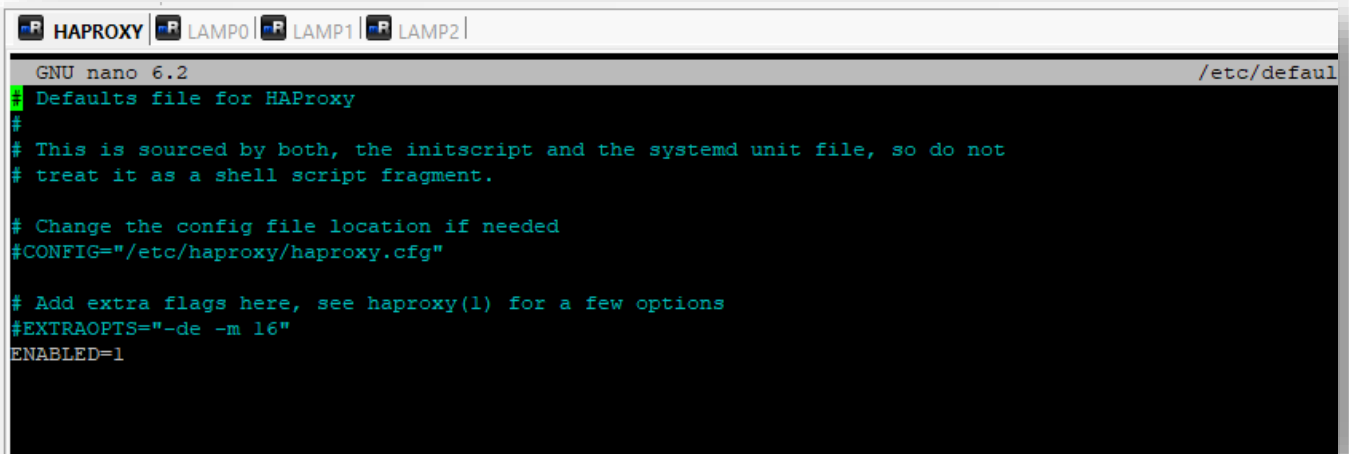
(Enter the items or ranges you want to select, separated by spaces.)

Which services should be restarted? 1_
```


1.2.3 Il faut que Haproxy soit considéré par le serveur comme un service pour cela :

```
nano /etc/default/haproxy
```

Ajouter à la fin du fichier ENABLED = 1 puis enregistrer.

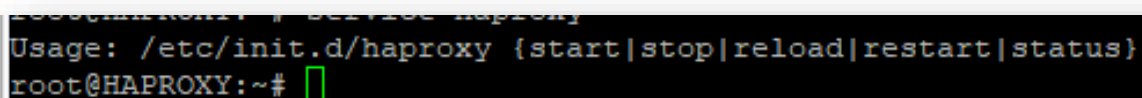


```
GNU nano 6.2 /etc/default/haproxy
# Defaults file for HAProxy
#
# This is sourced by both, the initscript and the systemd unit file, so do not
# treat it as a shell script fragment.
#
# Change the config file location if needed
#CONFIG="/etc/haproxy/haproxy.cfg"
#
# Add extra flags here, see haproxy(1) for a few options
#EXTRA_OPTS="--de -m 16"
ENABLED=1
```

Vous pouvez vérifier que Haproxy est bien détecter comme service en exécutant :

```
Service haproxy
```

Si le retour suivant vous est renvoyer alors haproxy est reconnu comme service.



```
Usage: /etc/init.d/haproxy {start|stop|reload|restart|status}
root@HAPROXY:~#
```

1.3 Configuration de HAPROXY.

1.3.1 Création du fichier de configuration :

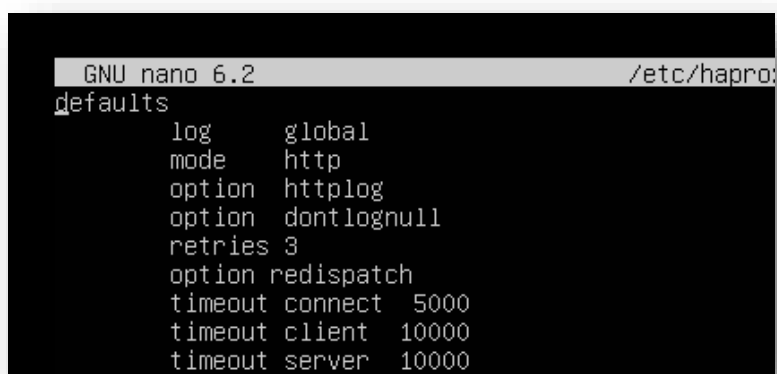
Créez et modifiez un nouveau fichier de configuration :

```
nano /etc/haproxy/haproxy.cfg
```

Commençons par ajouter la configuration bloc par bloc à ce fichier :

Nous mettons la configuration defaults qui permet de dire au bout de combien de temps de chargement le haproxy va rediriger sa requête sur un autre serveur ext...

```
defaults
    log global
    mode http
    option httplog
    option dontlognull
    retries 3
    option redispatch
    timeout connect 5000
    timeout client 10000
    timeout server 10000
```



```
GNU nano 6.2 /etc/haproxy/haproxy.cfg
defaults
    log global
    mode http
    option httplog
    option dontlognull
    retries 3
    option redispatch
    timeout connect 5000
    timeout client 10000
    timeout server 10000
```

La dernière partie du fichier de configuration qui est la plus important va nous permettre de configurer les serveurs web qui seront rattacher à notre Haproxy.

```
Frontend frontend-basee
    Bind *:80
    Default_backend backend-base
    Option forwardfor
Backend backend-base
    Balance roundrobin
    server lamp0 192.168.2.50:80 check
    server lamp1 192.168.2.51:80 check
    server lamp2 192.168.2.52:80 check
```

```
GNU nano 6.2 /etc/haproxy
defaults
    log      global
    mode     http
    option    httplog
    option    dontlognull
    retries  3
    option    redispatch
    timeout  connect 5000
    timeout  client  10000
    timeout  server  10000
frontend frontend-basee
    bind *:80
    default_backend backend-base
    option forwardfor
backend backend-base
    balance roundrobin
    server lamp0 192.168.2.50:80 check
    server lamp1 192.168.2.51:80 check
    server lamp2 192.168.2.52:80 check
```

Après cela faites CTRL+X et yes pour enregistrer et lancer votre serveur HAPRORY.

```
service haproxy start
```

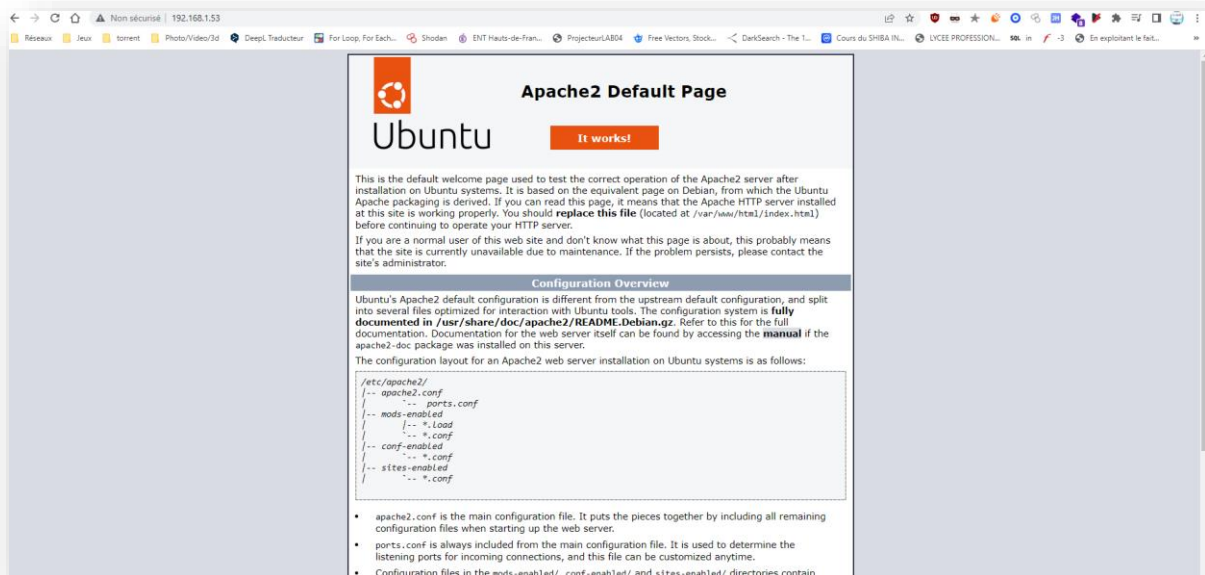
2 Test de HaProxy.

Maintenant que notre HaProxy est installer, configurer, et en production.
Nous allons faire des tests pour s'assurer que le balancing fonctionne.

2.1 Se rendre sur notre site

En premier lieu vous allez rentrer l'adresse IP de votre haproxy sur votre navigateur.

Pour nous 192.168.1.53

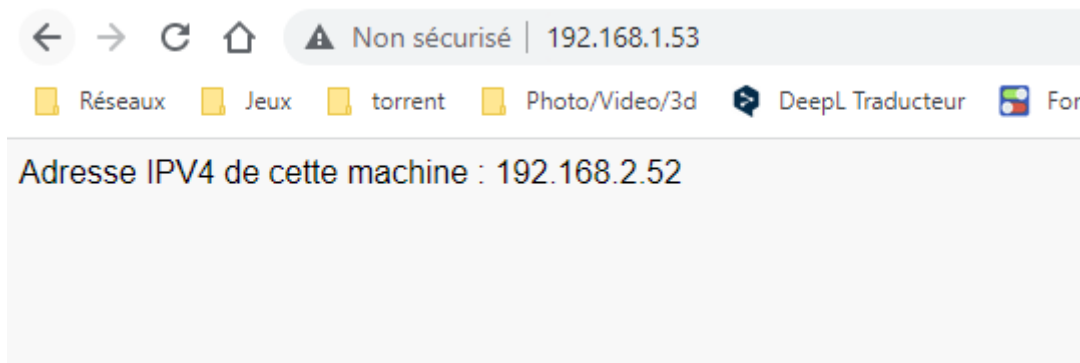


Nous avons bien accès à notre serveur web mais je ne sais pas sur quel serveur web mon HaProxy ma rediriger.

Pour contrôler sur quel serveur nous arrivons vous allez entre la commande suivante dans votre terminal sur chacun des serveur web.

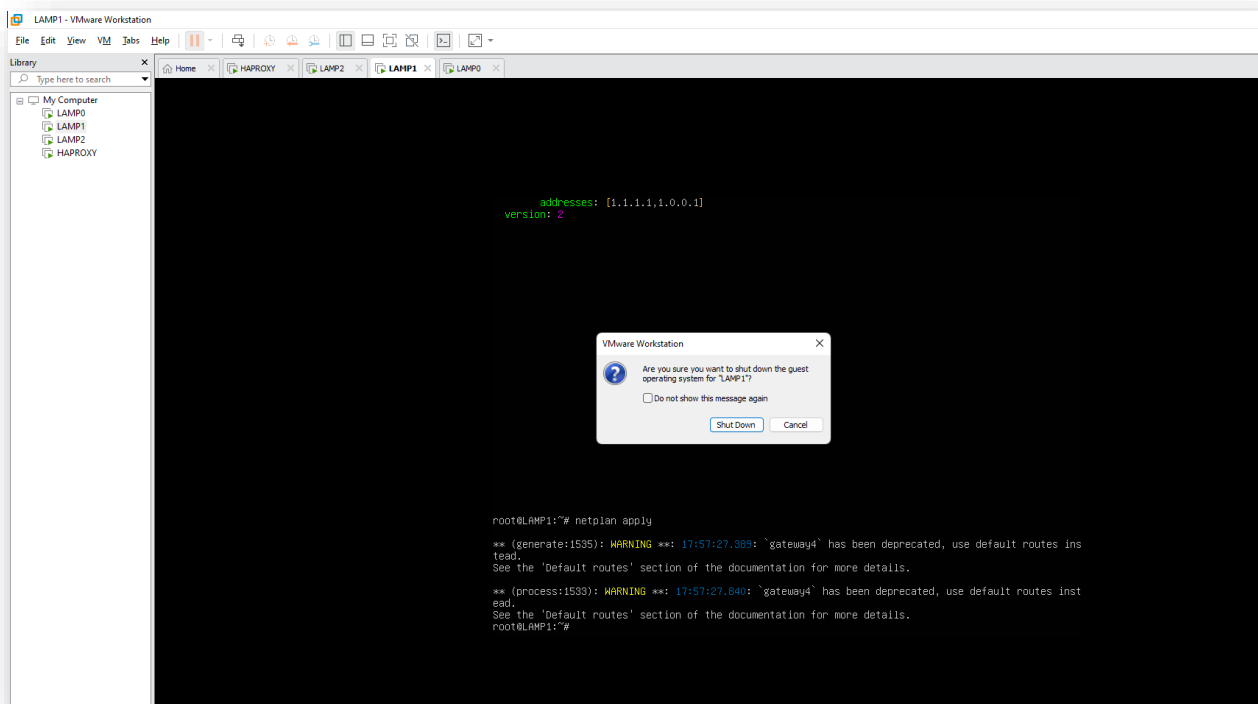
```
echo "Adresse IPv4 de cette machine : $(hostname -I | awk '{print $1}')" > /var/www/html/index.html
```

Dès à présent retourner une nouvelle fois sur votre haproxy à l'adresse 192.168.1.53
Et nous voyons maintenant sur qu'elle serveur nous sommes.



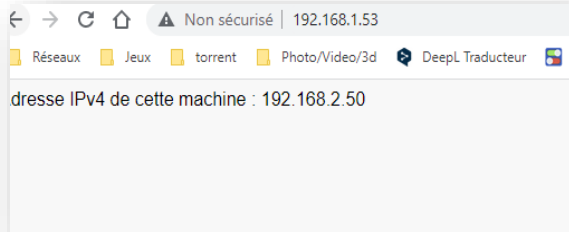
Maintenant que notre haproxy est fonctionnelle et que nous venons de tester que la charge est bien répartie entre les 3 serveurs web il faut maintenant vérifier le facteur disponibilité c'est-à-dire que si j'éteins un serveur le service web continue à fonctionner.

J'éteins dans mon cas le LAMP1 avec l'ip 192.168.2.51

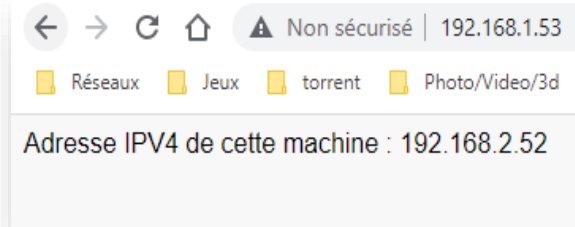


Nous retournons sur notre navigateur à l'adresse du haproxy 192.168.1.53

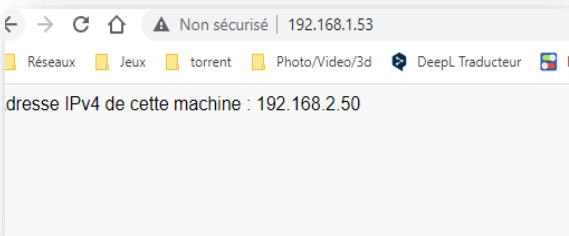
Cas 1 :



Cas 2 :



Cas 3 :



Résultat :

Comme vous venez de le remarquer normalement le cas 3 aurez du nous amenez sur le serveur 192.168.2.51 mais notre haproxy à remarquer qu'il était down est donc nous à rediriger sur un autre serveur web.

3 Conclusion

Nous avons donc réussis l'installation de haproxy pour le balancing qui permet à notre futur site web de ne « jamais » tomber en panne cela permettra une haute disponibilité car si un serveur tombe haproxy le remarque et redirige le trafic ailleurs et également de réduire le taux de charge en divisant la charge à part égal pas rapport au nombre de serveur.

ANNEXES

Plan Réseaux Haproxy

