La haute disponibilité

Sommaire

Définitions et principe

La **haute disponibilité** sont toutes les dispositifs visant à garantir la disponibilité d'un service et son bon fonctionnement **24H/24.**

Redondance de serveurs :

Round Robin DNS

- Plusieurs serveurs proposant le même service
- Pouvoir rediriger les requêtes des clients de manière équitable sur tout les serveurs

Load balancer ou répartiteur de charges

- Prendre en compte la puissance des machines , le nombre d'utilisateurs connectées etc.

Redondances de serveurs HeartBeat :

Principe et termes techniques :

- Basculement (actif/passif) ou Failover
- Un logiciel "battement de coeur" (heartbeat) est installer sur le serveur maître et sur chaque serveurs secondaire
- L'ensemble forme une grappe de serveurs ou cluster
- Chaque serveur du "cluster" est un "un noeud" (node en anglais)

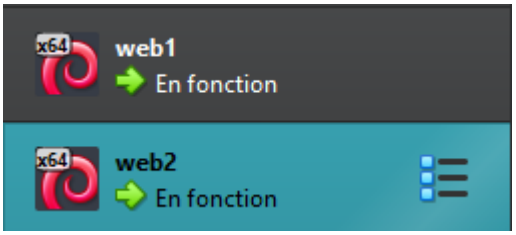
- Le serveur secondaire (passif) va surveiller en permanence les battements de cœur du serveur principal (actif)...
- Et donc prendre le relais automatiquement en cas d'arrêt des **battements**. Alors il devient **actif**

Principe et adresse flottante :

- Chaque serveur possède sa propre adresse IP mais seul "le nœud" actif possède, en plus, une adresse virtuelle : c'est par cette adresse flottante que les clients accèdent au service.
- Lorsque le serveur secondaire prend la place du maître, il s'attribue cette adresse flottante en devenant actif.
- Les clients continuent les accès sur cette adresse qui cette fois correspond donc au serveur secondaire

Haute disponibilité (Heartbeat)

Création et configurations des machines



Modifiez le nom d'hôte sur les deux machines Debian en web1 et web2 respectivement, en éditant le fichier /etc/hostname et nano /etc/hosts

```
web1
web2
```

Configuration de leurs adresses ip avec la commande nano /etc/network/interfaces :

192.168.56.11 pour **web1**

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
address 192.168.56.11
gateway 192.168.56.254
```

192.168.56.12 pour **web2**

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
address 192.168.56.12
gateway 192.168.56.254
```

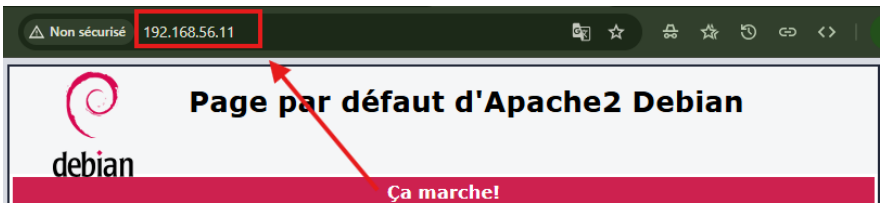
Avant toutes installation, il faut mettre à jour les paquets “apt update”

```
root@web1:~# apt update_
```

Ensuite, il faut installer apache2 avec la commande “apt install apache2”

```
root@bullseye:~# apt install apache2
```

Depuis la machine hôte “**WindowsR211**” on peut vérifier que le site par défaut d’Apache2 fonctionne bien en tapant l’adresse ip de **web1/2** dans le navigateur.



La configuration de heartbeat doit être identique sur tous les serveurs de la grappe
Elle repose sur 3 fichiers

- “Ha.cf”
- “Authkeys”
- “Haresources”

Installation de Heartbeat sur chacune des 2 machines **web 1 et 2**

```
apt install heartbeat_
```

Les 3 fichiers demandés doivent se trouver dans les répertoires suivants :

“nano /etc/ha.d/ha.cf”
“nano /etc/ha.d/authkeys”
“nano /etc/ha.d/haresources”

On observe que les fichiers ne sont pas présents sur la machine donc il va falloir les créer.

Configuration des 3 fichiers de configuration Hearbeat (sur les 2 machines **web**)

Configuration de “ha.cf” :

On se déplace dans le répertoire /etc/ha.d ou on veut y créer le fichier de conf ha.cf

```
root@web1:~# cd /etc/ha.d
```

Exécuter la commande nano /etc/ha.d/ha.cf pour y ajouter les infos.

“**bcast**” = Interface réseau

“**node**”= noeud **web1 web2**
Enregistrer le fichier et faire la manipulation pour la seconde machine

deadtime : Temps (en secondes) avant de considérer un nœud comme défaillant si aucun signal n'est reçu (ex. : 5 secondes).*

keepalive : Fréquence d'envoi des signaux de vie (ex. : 1 seconde).

Configuration de “authkeys” :

On se déplace dans le répertoire /etc/ha.d ou on veut y créer le fichier de conf **authkeys**.

Exécuter la commande nano /etc/ha.d/authkeys pour y ajouter les infos.

Il s’agit d’une clé partagée entre les serveurs de la grappe (Même chose sur les 2 serveurs donc…). Ce fichier détermine la clé et le protocole de protection utilisée

⚠️ Attention ! Le service **Heartbeat** exige une protection de ce fichier sinon il ne démarrera pas et sera visible par n’importe qui.

Donc exécuter la commande suivante :

```
chmod 600 /etc/ha.d/authkeys
```

Cela définit les permissions du fichier **authkeys** de manière à ce qu'il soit accessible en **lecture** et **écriture** uniquement par le **propriétaire** du fichier. Assurez-vous d'ajuster les permissions en fonction de vos besoins de **sécurité spécifiques**.

Vous pouvez vérifier les nouvelles permissions en utilisant la commande :

```
ls -l /etc/ha.d/authkeys
```

Enregistrer le fichier et faire la manipulation pour la seconde machine

Configuration de “haresources” :

Haresources correspond à liste des ressources (adresses virtuelles et services concernés) fournies par la grappe.

La configuration sur chacune des 2 machines doit être la même, donc on laisse **web1** car ca va être la machine de base

On se déplace dans le répertoire /etc/ha.d ou on veut y créer le fichier de conf **haresources**.

Exécuter la commande nano /etc/ha.d/haresources pour y ajouter les infos.

- **web1** : Le nom de la machine qui sera activée par défaut au démarrage de Heartbeat. Assurez-vous que ce nom est identique sur les deux machines.

- **IPaddr::192.168.56.10** : Une **adresse IP virtuelle flottante** que Heartbeat gérera. Assurez-vous que cette adresse est unique et accessible sur le réseau.

- **apache2** : Le service à surveiller. Vous pouvez ajouter d'autres services si nécessaire.

Enregistrer le fichier et faire la manipulation pour la seconde machine

Modification du fichier “hosts” (2 machines web) :

Le fichier **/etc/hosts** est utilisé pour associer des adresses IP à des noms d'hôtes sur un système. Si vous n'avez pas de **service DNS** installé et que vous souhaitez déclarer les hôtes **web1** et **web2** dans ce fichier, vous pouvez le faire de la manière suivante.

web1 et **web2** doivent être declaré dans /etc/hosts (excepté si un service DNS est installée)

Configuration de **web1** :

On met **web1** dans 127.0.0.1 (lui même) et on déclare **web2** comme machine sauveuse.

```
  GNU nano 5.4
127.0.0.1       localhost
127.0.1.1       web1
192.168.56.12   web2
```

Configuration de **web2** :

On met **web2** dans 127.0.0.1 (lui même) et on déclare **web1** comme machine sauveuse

```
  GNU nano 5.4
127.0.0.1       localhost
127.0.1.1       web2
192.168.56.11   web1
```

On ne modifie rien pour cette partie elle doit rester identique sur les 2 machines

```
# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Modification de la page par défaut d'Apache (2 machines web) :

Nous devons maintenant modifier le code html par défaut des pages :

On doit faire la commande nano /var/www/html/index.html

On se retrouve dans le code de la **page html par défaut d'apache** et on descend pour modifier le titre de la page

```
  </style>
</head>
<body>
  <div class="main_page">
    <div class="page_header floating_element">
      <img src="/icons/openlogo-75.png" alt="Debian Logo" class="floating_element"/>
      <span class="floating_element">
        Web1 Apache page
```

On peut modifier la partie du dessous

```
<div class="section_header section_header_red">
  <div id="about"></div>
  It works! ip address 192.168.56.11
```

On enregistre le tout

On observe que la page a été modifiée, le titre correspond bien au titre que nous avons changé et la partie du dessous également.

Le service heartbeat est-il bien actif ?

En premier lieu, **stopper** le service d'apache

```
root@web1:/etc/ha.d# systemctl stop apache2.service
```

Vérifier que le service est bien arrêté. Remplacer stop par "systemctl status apache2.service"

```
root@web1:~# systemctl status apache2.service
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; disabled; vendor preset: enabled)
     Active: inactive (dead)
       Docs: https://httpd.apache.org/docs/2.4/
```

Inactive, on observe que le service à bien été désactiver

Désactiver le service apache2 sur les 2 machines Heartbeat le démarrera lui-même

```
root@web1:/etc/ha.d# systemctl disable apache2.service
Synchronizing state of apache2.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable apache2
Removed /etc/systemd/system/multi-user.target.wants/apache2.service.
root@web1:/etc/ha.d#
```

Tester que le service Heartbeat est bien actif. Le **redémarrer** avec la commande "systemctl restart heartbeat.service"

```
root@web1:/etc/ha.d# systemctl restart heartbeat.service
```

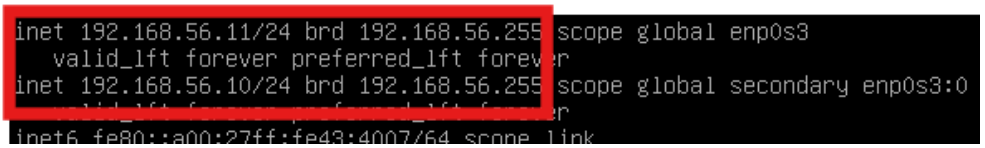
On peut ensuite faire un "systemctl status heartbeat" pour **vérifier** son état, on observe qu'il est **actif.**

```
root@web1:~# systemctl status heartbeat
● heartbeat.service - Heartbeat High Availability Cluster Communication and Membership
     Loaded: loaded (/lib/systemd/system/heartbeat.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2025-01-03 23:55:41 CET; 17s ago
```

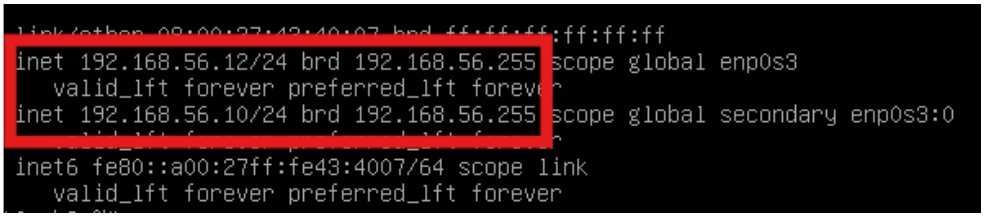
Si le service heartbeat ne redémarre pas correctement :

1. - Relire les fichiers de configurations
2. - Redémarrer les vm
3. - Stopper les services apache2
4. - Redémarrer Heartbeat

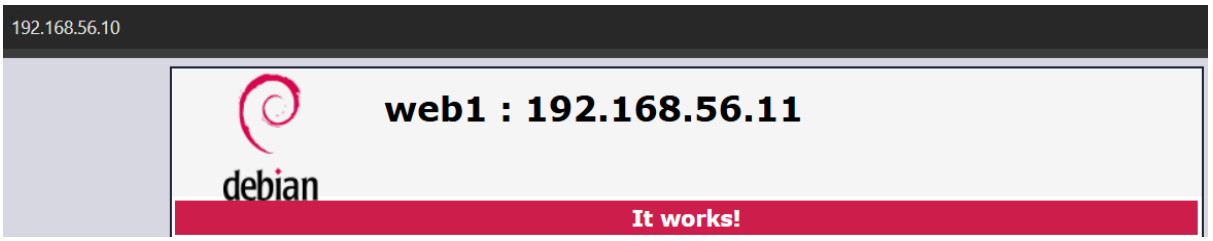
Utilisez la commande "ip a" sur la machine "**web1**" pour vérifier son adresse IP prévue, qui devrait être "**192.168.56.10**".



Vérifier également sur la machine “**web2**” pour voir si l’adresse "192.168.56.10" s’affiche bien.

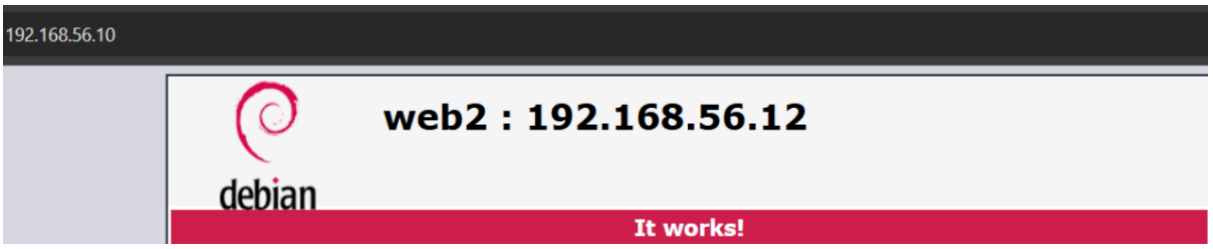


En vérifiant l'adresse IP de la grappe, connectez-vous depuis la machine hôte à l'adresse "**192.168.56.10**".
Vous devriez être redirigé vers la page par défaut d'Apache sur la machine "**web1**".



Afin de confirmer que "**web2**" prend effectivement l'IP "**192.168.56.10**" en cas d'indisponibilité de "**web1**", arrêtez le service Apache sur "**web1**" en utilisant la commande suivante :

- **“systemctl stop apache2.service”**



Nous constatons que '**Web2**' prend en charge l'adresse IP “**192.168.56.10”**, comme prévu, suite à la coupure de notre machine '**Web1**'.

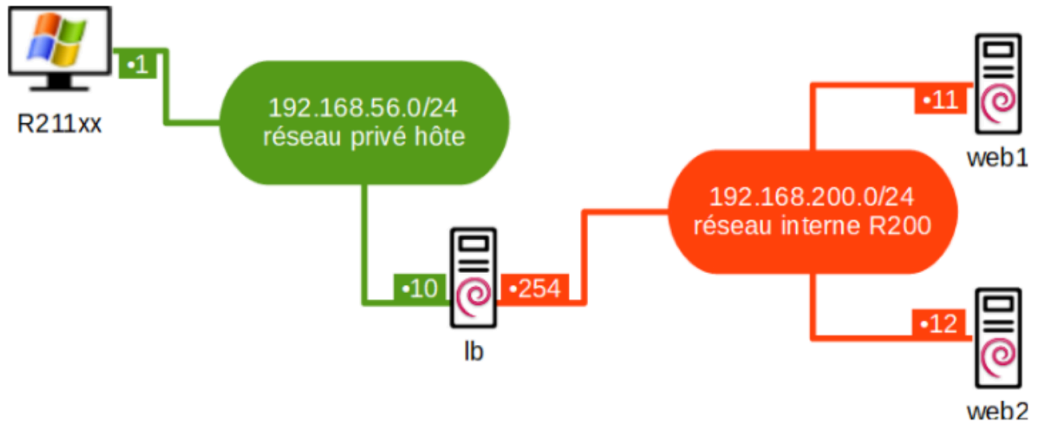
Le relais s'effectue donc en toute sécurité. Il suffit simplement d'attendre quelques secondes de chargement.

Load Balancing

Qu’est ce que le load balancing ?

Le **load balancing** répartit le trafic réseau ou les tâches entre plusieurs serveurs pour optimiser les performances, éviter les surcharges et garantir la disponibilité des services.

Objectifs :



Qu’est ce que le réseau interne ?

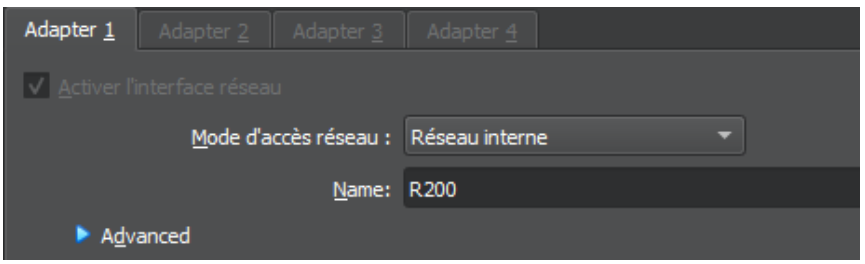
Le **réseau interne** dans VirtualBox permet de connecter plusieurs machines virtuelles entre elles sur un **réseau isolé**, sans accès à l'extérieur (ni Internet), idéal pour tester des configurations réseau en toute **sécurité**.

Load Balancing avec une grappe de 2 serveurs Web

Création et configurations des machines

Premièrement, nous créons 2 VM nommé “ **Web1**” et “**Web2**”

- Modification de l’**adaptateur réseau** en **réseau interne** pour **Web1** et **Web2**



Ensuite, nous modifions les fichiers **/etc/hosts** et **/etc/hostname** afin de changer le nom des machines :

```
 GNU nano 3.2                                    /etc/hosts

127.0.0.1       localhost
127.0.1.1       web1
```

```
 GNU nano 3.2                                  /etc/hostname

web1
```

Nous configurons les 2 VM avec des adresses IP en **192.168.200**.(**11** ou **12**) et avec la passerelle en **192.168.200.254**

Configuration ip de Web 1 :

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
        address 192.168.200.11
        gateway 192.168.200.254
```

Configuration ip de Web 2 :

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
address 192.168.200.12/24
gateway 192.168.200.254
```

Avant toutes installation, il faut mettre à jour les paquets “apt update”

```
root@web1:~# apt update_
```

Ensuite, il faut installer apache2 avec la commande “apt install apache2”

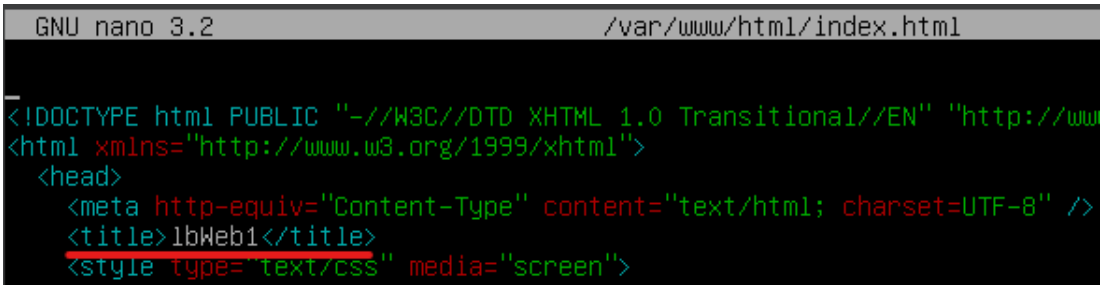
```
root@bullseye:~# apt install apache2
```

Modification du fichier de configuration Apache :

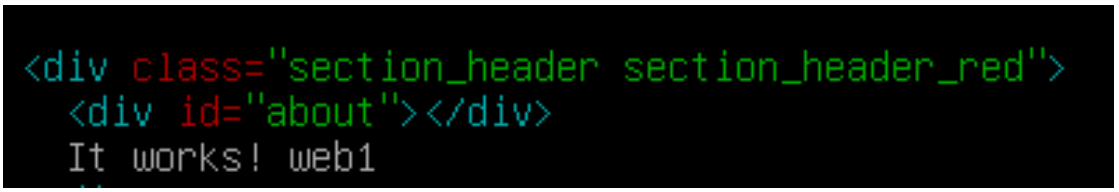
Nous devons maintenant modifier le code html par défaut des pages :

- Utilisation de la commande **nano /var/www/html/index.html**

- On se retrouve dans le code de la **page html par défaut d'apache** et on descend pour modifier le titre de la page



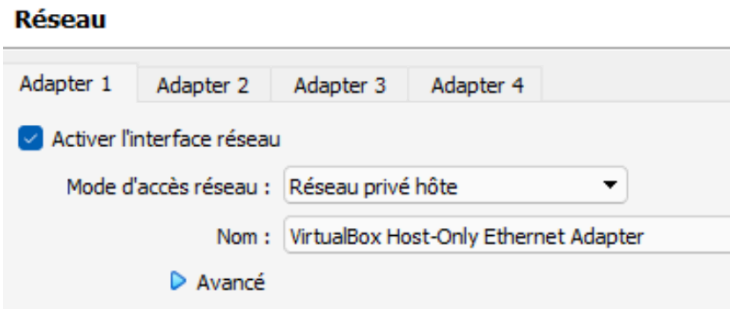
- Modification d'un morceau de code afin d'afficher " IT Works ! " suivit du nom de la page :



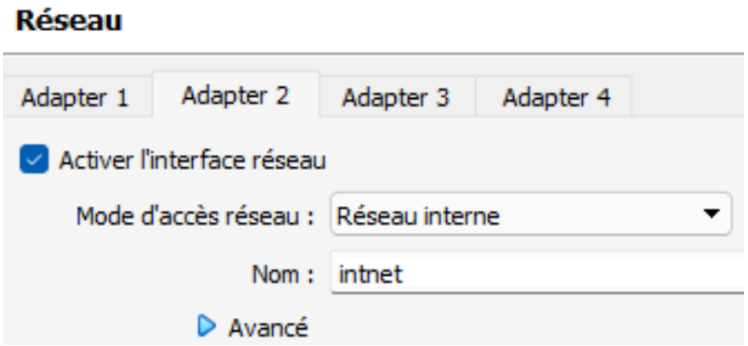
- Modification à effectuer sur **Web1** et **Web2**

Installation et configuration du serveur lb

Nous allons mettre en place 2 adaptateurs réseaux, le premier concerne le réseau **192.168.56.0**



Le deuxième adaptateur réseau concerne le réseau **192.168.200.0**



Configuration des interfaces :

Serveur "lb"

- Configuration des interfaces sur la machine Debian

- Utilisation de la commande **nano /etc/network/interfaces**

Configuration d'abord en Réseau privé hôte (.10)

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
        address 192.168.56.10
        gateway 192.168.56.254
```

Ensuite, configuration en Réseau interne (.254)

```
# The 2nd network interface enp0s8
allow-hotplug enp0s8
iface enp0s8 inet static
address 192.168.200.254/24
```

Activation du routage :

Vous devez activer le routage IP sur la machine. Pour cela, ouvrez le fichier de configuration **sysctl.conf** :

“nano /etc/sysctl.conf “

Recherchez la ligne qui contient “**net.ipv4.ip_forward**” et assurez-vous qu'elle est **décommentée**. Si la ligne n'existe pas, ajoutez-la à la fin du fichier :

Le “**1**” **active le routage :** “net.ipv4.ip_forward=1”

```
# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
```

Vérification de l’activation du routage :

Pour vérifier si le routage IP est **activé** sur votre système, vous pouvez examiner la valeur du paramètre **ip_forward** dans le répertoire “/proc/sys/net/ipv4/**”**. Vous pouvez le faire en utilisant la commande **cat**.

“**cat /proc/sys/net/ipv4/ip_forward** “

Pensez à redémarrer la machine pour que le fichier s’actualise, avec la commande “reboot”

```
root@lb:~# cat /proc/sys/net/ipv4/ip_forward
```

Le **1** signifie que le **routage** est bien **activé**.

Configuration des fichiers ipvsadm :

Nous allons maintenant configurer les 2 fichiers suivants :

/etc/default/ipvsadm

/etc/ipvsadm.rules

Fichier “ipvsadm” :

Voici l’ancienne configuration :

```
# ipvsadm

# if you want to start ipvsadm on boot set this to true
AUTO="false"

# daemon method (none|master|backup)
DAEMON="none"

# use interface (eth0,eth1...)
IFACE="eth0"

# syncid to use
# (0 means no filtering of syncids happen, that is the default)
# SYNCID="0"
```

Ici, la nouvelle configuration :

```
# ipvsadm

# if you want to start ipvsadm on boot set this to true
AUTO="true"

# daemon method (none|master|backup)
DAEMON="master"

# use interface (eth0,eth1...)
IFACE="enp0s3"

# syncid to use
# (0 means no filtering of syncids happen, that is the default)
# SYNCID="0"
```

- **AUTO = “true”** : Chargement de l’application et des règles au démarrage.
- “**Maître**” par défaut puisqu’il est le seul load balancer.
- **IFACE=**”**enp0s3**” : C’est par cette interface qu’arrivent les requêtes vers le cluster de serveurs Web.

Fichier “ipvsadm.rules” :

```
# emtpy rules file for ipvsadm

#Définition du service
ipvsadm -A -t 192.168.56.10:80 -s rr

#Membres du clusters
ipvsadm -a -t 192.168.56.10:80 -r 192.168.200.11:80 -m
ipvsadm -a -t 192.168.56.10:80 -r 192.168.200.12:80 -m
```

- **A:** Indique **l'ajout d'un service**.
- **t [VIP:Port]**: Spécifie l'adresse IP virtuelle (**VIP**) et le **port** du service.
- **s [Protocole]**: Spécifie le protocole du service.
- **rr** : Algorithme de **répartition Round Robin**

Vérification du paramétrage :

On vérifie le paramétrage avec la commande “**ipvsadm -ln”** :

```
root@lb:~# ipvsadm -ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.56.10:80 rr
  -> 192.168.200.11:80            Masq    1      0          0
  -> 192.168.200.12:80            Masq    1      0          0
```

Tests de Fonctionnement

Dans le navigateur, on recherche l'adresse IP **192.168.56.10**

On observe que la page est bien celle de Web1 :

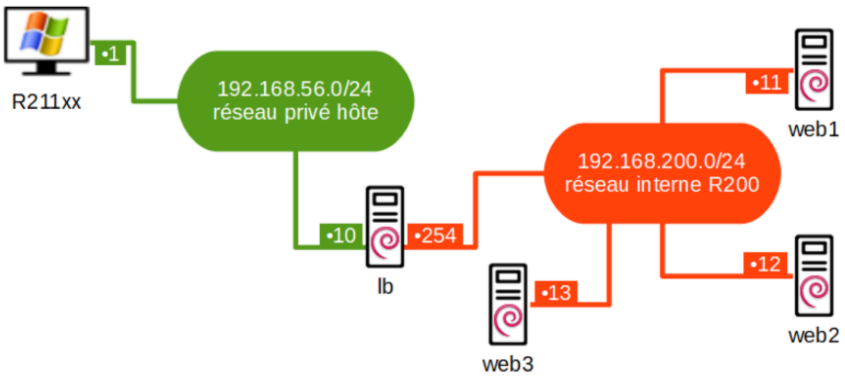
On observe que la page est bien celle de Web2 :

Load Balancing avec une grappe de 3 serveurs Web

Nous allons à présent mettre en place cette solution en ajoutant une nouvelle VM “**web3**” :

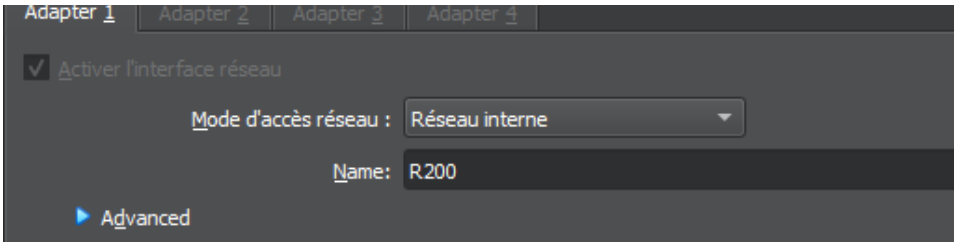
Objectif :

Mettre à présent en place la solution suivante.



Création et configuration de la VM web3 :

- **Modification** de l’**adaptateur réseau** pour **réseau interne** :



- **Modification** de la configuration **IP** en .13

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
        address 192.168.200.13/24
        gateway 192.168.200.254
```

- **Installation** de **Apache2** :

```
root@web3:~# apt install apache2
```

- **Modification** du **fichier de configuration Apache** :

```
<meta http-equiv="Content-Type" content
<title>lb Web3</title>
```

```
    <div id="about"></div>
    It works! web3
```

Modification du fichier “**ipvsadm.rules**”

- On vient ajouter la VM **Web3** : **192.168.200.13**

```
  GNU nano 3.2                                  /etc/ipvsadm.r

#Définition du service
ipvsadm -A -t 192.168.56.10:80 -s rr

#Les membres du clusters
ipvsadm -a -t 192.168.56.10:80 -r 192.168.200.11:80 -m
ipvsadm -a -t 192.168.56.10:80 -r 192.168.200.12:80 -m
ipvsadm -a -t 192.168.56.10:80 -r 192.168.200.13:80 -m
```

Test de Fonctionnement :

Nous pouvons alors faire un **ping** (test de connexion) du serveur lb vers les machines **web 1/2/3**

Web1 :

```
root@lb:~# ping 192.168.200.11
PING 192.168.200.11 (192.168.200.11) 56(84) bytes of data.
64 bytes from 192.168.200.11: icmp_seq=1 ttl=64 time=1.93 ms
64 bytes from 192.168.200.11: icmp_seq=2 ttl=64 time=1.25 ms
```

Web2 :

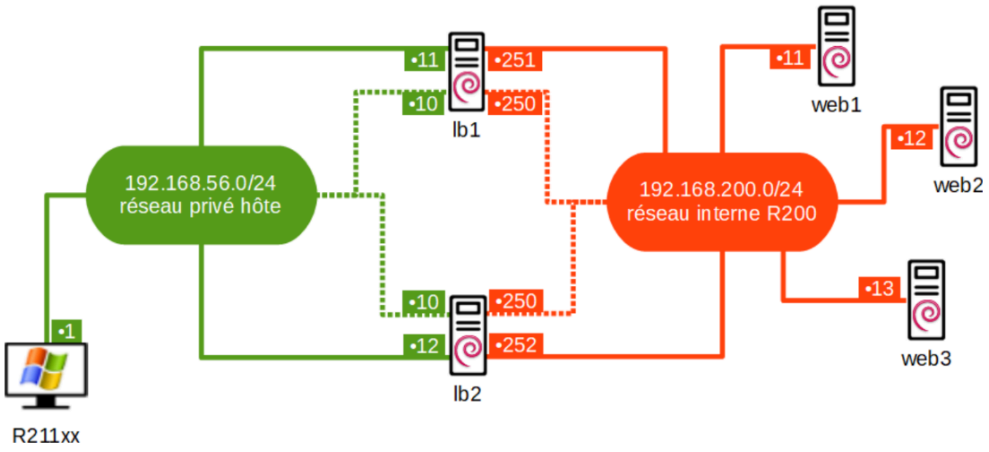
```
root@lb:~# ping 192.168.200.12
PING 192.168.200.12 (192.168.200.12) 56(84) bytes of data.
64 bytes from 192.168.200.12: icmp_seq=1 ttl=64 time=3.04 ms
64 bytes from 192.168.200.12: icmp_seq=2 ttl=64 time=1.08 ms
```

Web3 :

```
root@lb:~# ping 192.168.200.13
PING 192.168.200.13 (192.168.200.13) 56(84) bytes of data.
64 bytes from 192.168.200.13: icmp_seq=1 ttl=64 time=0.987 ms
64 bytes from 192.168.200.13: icmp_seq=2 ttl=64 time=1.08 ms
```

Load Balancing avec une grappe de 3 serveurs Web et 2 serveurs lb

Objectif :



Nous allons donc réaliser une configuration avec 2 serveurs lb et 3 serveurs Web

Création et configuration de lb2

- **Création** de la VM lb2
- **Modification** de l'**étiquette réseau** pour **réseau interne**
- **Modification** de la **configuration IP**

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
address 192.168.56.12/24
gateway 192.168.56.254

# The 2nd network interface enp0s8
allow-hotplug enp0s8
iface enp0s8 inet static
address 192.168.200.252/24
```

- **Changement** de la configuration IP sur le serveur **lb1** également

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
address 192.168.56.11/24
gateway 192.168.56.254

# The 2nd network interface enp0s8
allow-hotplug enp0s8
iface enp0s8 inet static
address 192.168.200.251/24
```

Modification des interfaces réseaux des serveurs web :

Web1 : 192.168.200.11 / gateway : *192.168.200.250*

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
        address 192.168.200.11/24
        gateway 192.168.200.250
```

Web2 : 192.168.200.12 / gateway : *192.168.200.250*

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
        address 192.168.200.12/24
        gateway 192.168.200.250
```

Web3 : 192.168.200.13 / gateway : *192.168.200.250*

```
# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
        address 192.168.200.13/24
        gateway 192.168.200.250
```

Installation de HeartBeat et load balancing sur lb2 :

Après avoir configuré notre machine lb2, nous pouvons installer **HeartBeat** et le **Load balancing :**

Installation de HeartBeat :

- **apt update**
- **apt install heartbeat**

Installation du load balancing :

- **apt update**
- **apt install ipvsadm**

Configuration de HeartBeat sur lb1 et lb2 :

- Création des fichiers “**ha.cf**”, “**authkeys**”, “**haresources**”

```
  GNU nano 3.2                                  /etc/ha.d/ha.cf

bcast enp0s3
deadtime 5
keepalive 1
node lb1 lb2
```

```
  GNU nano 3.2                                /etc/ha.d/authkeys

auth 1
1 sha1 MaClefSecrete
```

Le service heartbeat exige une protection de ce fichier sinon il ne démarrera pas et sera visible par n’importe qui. **chmod 600 /etc/ha.d/authkeys**

```
  GNU nano 3.2                              /etc/ha.d/haresources

lb1 IPaddr::192.168.56.10       ipvsadm
lb1 IPaddr::192.168.200.250     enp0s8
```

Activation du routing :

Les **2 machines Load balancer** doivent prendre en compte le routage, on peut donc les activer exactement de la même manière que précédemment c’est-à-dire dans le fichier :

nano /etc/sysctl.conf

Vérification avec : **cat /proc/sys/net/ipv4/ip_forward**

```
root@lb2:~# cat /proc/sys/net/ipv4/ip_forward
1
```

Configuration de heartbeat sur les serveurs lb1 et lb2 :

- Mettre en place une répartition des charges avec 3 charges pour web1 et une pour web2 et web3.

Sur les 2 serveurs lb1 et lb2 configurer le fichier **nano /etc/ipvsadm.rules**

```
  GNU nano 3.2                                  /etc/ipvsadm.rules

#Définition du service
ipvsadm -A -t 192.168.56.10:80 -s wrr

#Les membres du clusters
ipvsadm -a -t 192.168.56.10:80 -r 192.168.200.11:80 -m -w 3
ipvsadm -a -t 192.168.56.10:80 -r 192.168.200.12:80 -m -w 1
ipvsadm -a -t 192.168.56.10:80 -r 192.168.200.13:80 -m -w 1
```

Vérification avec la commande **ipvsadm -ln :**

```
root@lb1:~# ipvsadm -ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.56.10:80 wrr
  -> 192.168.200.11:80            Masq    3      0          0
  -> 192.168.200.12:80            Masq    1      0          0
  -> 192.168.200.13:80            Masq    1      0          0
root@lb1:~#
```

Tests de Fonctionnement

Vérification de l'interface réseau, afin de voir si l'IP flottante en 192.168.200.250 est présente.

```
root@lb1:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state
    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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
000
    link/ether 08:00:27:fe:f3:68 brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.10/24 brd 192.168.56.255 scope global en
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fefe:f368/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
000
    link/ether 08:00:27:dc:52:d5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.200.254/24 brd 192.168.200.255 scope globa
       valid_lft forever preferred_lft forever
    inet 192.168.200.250/24 brd 192.168.200.255 scope globa
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fedc:52d5/64 scope link
       valid_lft forever preferred_lft forever
```