**Heartbeat**

**Objectif :** Installer et configurer le logiciel Heartbeat sur deux serveurs Debian de façon à assurer une haute disponibilité des serveurs

Sommaire

[Schéma réseaux 1](#_Toc124124876)

[Configuration de l’environnement 2](#_Toc124124877)

[Qu’est-ce que Heartbeat ? 2](#_Toc124124878)

[Installation et configuration de Heartbeat 2](#_Toc124124879)

[Configuration du fichier ha.cf sur les serveurs web1 et web2 2](#_Toc124124880)

[Configuration du fichier authkeys 3](#_Toc124124881)

[Configuration du fichier haresources 3](#_Toc124124882)

[Heartbeat et Load Balancing 7](#_Toc124124883)

[Schémas du réseau 7](#_Toc124124884)

[Création des serveurs Load Balancer lb1 et lb2 sur VirtualBox 7](#_Toc124124885)

[Installation de heartbeat sur les serveurs lb1 et lb2 7](#_Toc124124886)

[Création des serveurs Web1 Web2 et Web3 8](#_Toc124124887)

[Sur VirtualBox 8](#_Toc124124888)

[Sur lb1 et lb2 8](#_Toc124124889)

[Activer le routage sur les serveurs lb1 et lb2 9](#_Toc124124890)

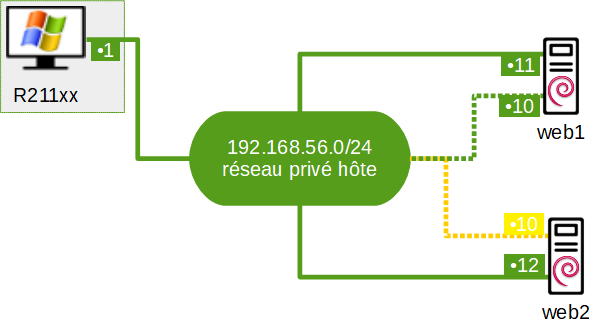
[Activer et configurer les Load Balancer 9](#_Toc124124891)

[Répartir la charge aux serveurs web 10](#_Toc124124892)

[Sur les Serveurs web1 web2 et web3 11](#_Toc124124893)

[Phase de tests 11](#_Toc124124894)

# Schéma réseaux



# Configuration de l’environnement

- Configuration réseaux des machines

- Suppression du proxy

- Changement de DNS

- Mettre à jour les paquets avec la commande apt update

# Qu’est-ce que Heartbeat ?

Heartbeat est un logiciel de surveillance de la disponibilité des programmes pour les OS Linux distribué sous licence GPL (General Public Licence). Il est installé sur le serveur maître et sur chaque permet de capter les « battements de cœurs », signal émis par les autres serveurs du cluster.

# Installation et configuration de Heartbeat

Effectuer les manipulations suivantes sur les deux serveurs web1 et web2 :

* Installer Heartbeat avec la commande **apt install heartbeat**
* Installer apache2 avec **apt install apache2**
* Modifier le code source de la page par défaut d’apache pour identifier les deux serveurs une fois qu’ils auront la même ip avec la commande **nano /var/www/html/index.html**

## Configuration du fichier ha.cf sur les serveurs web1 et web2

La config de heartbeat doit être unique sur tous

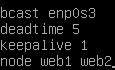
Dans le répertoire /etc/ha.d -> ha pour High Availability

Créer le fichier de config ha.cf dans lequel nous ajouterons ceci

1. Interface réseau
2. Temps nécessaire avant de considérer qu’un nœud set mort. Le temps est en secondes par défaut. Rajouter ms derrière pour l’avoir en millisecondes.

**Attention avec cette valeur** : si elle est trop courte, le système risque de s’ auto-déclarer mort. Si elle est trop grande, l’autre machine mettra un temps conséquent avant de s’en apercevoir et reprendre la main

1. Intervalle entre 2 battements de cœur. La valeur est en secondes par défaut. Pour la spécifier en millisecondes, on rajoutera ‘ms’ derrière. (Par exemple, 200ms).
2. « Nœud » web1 et web2, liste les machines utilisé pour la haute disponibilité séparée par des espaces



## Configuration du fichier authkeys

Clé partagée entre les serveurs de grappe (même chose sur les 2 serveurs donc…) Ce fichier détermine la clé et le protocole de protection utilisé



Attention ! le service heartbeat exige une protection de ce fichier sinon il de démarrera pas et sera visible par n’importe qui



## Configuration du fichier haresources

Liste des ressources (adresses virtuelles et services concernés) fournies par la grappe. La configuration sur chacune des machines est la même. Ce nom doit être le même pour les 2 machines. C’est le nom de la machine qui sera activée par défaut au démarrage de heartbeat



(Faire une flèche vers capture) web1 : le serveur web1 reste le serveur « maître »

Modifier le fichier « hosts »

Web1 et web2 doivent être déclarés dans /etc/hosts (excepté si un service DNS est installé)

Web1 web2



Le service est-il bien actif ?

Stopper apache 2 sur les 2 vm



Vérifier que service bien arrête remplacer **stop** par **status**





Désactiver le service apache2 sur les 2 vm  heartbeat le démarrera lui-même

Tester que le service heartbeat est bien actif. Le redémarrer avec la commande suivante

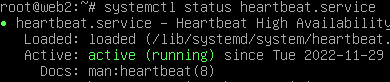


Vérifier son état avec **systemctl status heartbeat.service**

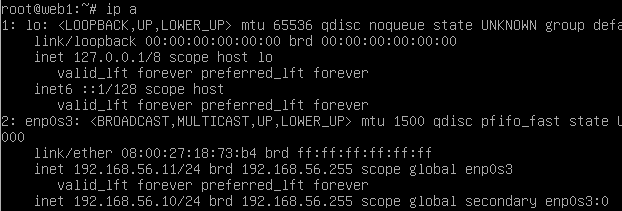
Si le service heartbeat ne redémarre pas correctement :

* Relire fichier de config
* Redémarrer les vm
* Stopper les services apache2
* Redémarrer heartbeat

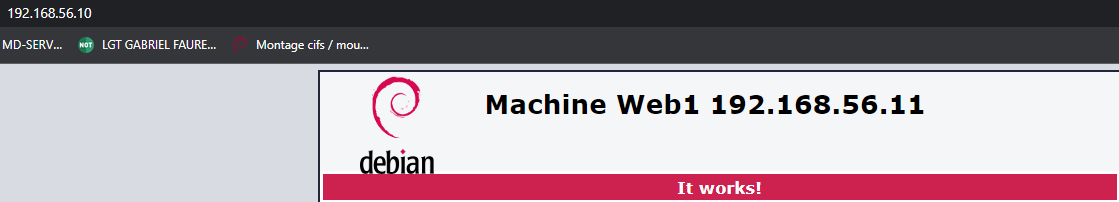
Le service heartbeat est bien actif



Vérifier l’ip de la machine web1 (censé récupérer l’ip 192.168.56.10) avec la commande **ip a**



On vérifie l’ip de la grappe en se connectant depuis la machine hôte à 192.168.56.10 : nous devrions tomber sur la page par défaut d’apache de web1.



Pour vérifier que web2 récupère bien l’IP 192.168.56.10 en cas de coupure de web1, on stop le service apache2 de web1 avec **systemctl stop apache2.service** sur le machine web1.

Load Balancing

* Les clients envoient leurs requêtes au « load balancer » qui se charge de les transmettre à la grappe de serveurs.
* La charge de travail est donc répartie entre les différents serveurs car ils sont tous actifs simultanément
* En cas de panne de l’un d’eux, le travail se portera sir le serveur restant
* Le « load balancer », en isolant les serveurs en les cachant à la vue des clients qui ne connaissent que l’adresse du « load balancer », comme c’est le cas dans les DMZ
* L’algorithme de répartition de charge utilisés dans l’exemple est le « round robin » (tourniquet) qui attribue chaque nouvelle requête au serveur suivant disponible de la grappe
* Avec 2 serveurs, les requêtes seront donc attribuées alternativement à l’un et à l’autre serveur.
* Dans ce mode, les serveurs ignorent qu’ils sont en grappe
* Aucune configuration particulière sur les serveurs, toute la configuration se fait sur le « load balancer »
* La seule contrainte est que la passerelle par défaut des serveurs du cluster doit être le « load balancer »

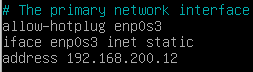
Serveur « lb »

**Configuration réseaux de lb**

Config web1



Config web2



Activer routage de lb avec la commande **nano /etc/sysctl.conf**

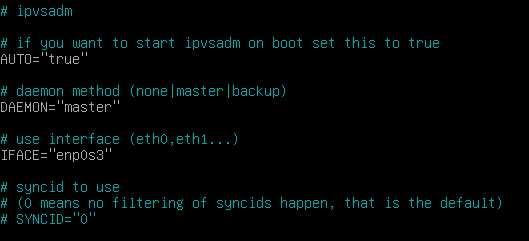
dans le fichier enlever le commentaire sur la ligne suivante



reboot la machine et faire la commande suivante un 1 doit s’afficher



Activer le Load Balancing avec la commande **nano /etc/default/ipvsadm**



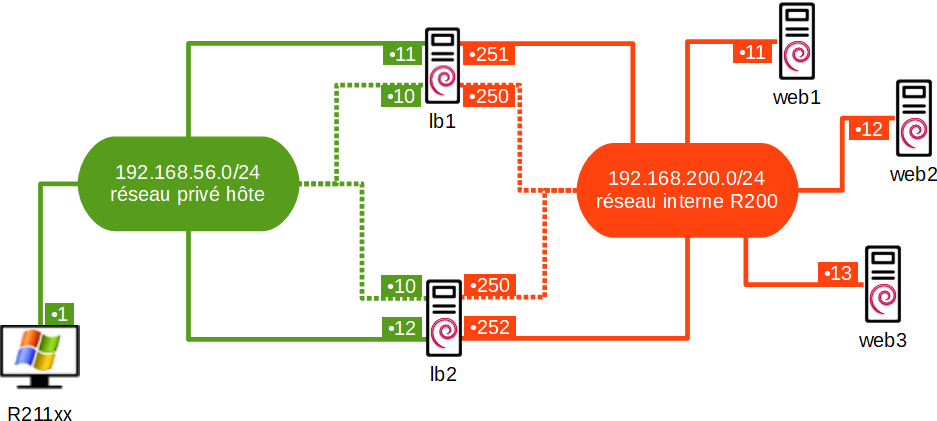
1. Chargement de l’application et des règles au démarrage
2. Maître par défaut puisqu’il est le load balancer
3. C’est par cette interface qu’arrivent les requêtes vers la grappe de serveurs web

# Heartbeat et Load Balancing

**Objectif :** Créer 2 serveurs load balancer « protégés » par Heartbeat répartissant la charge vers 3 serveurs web.

Toutes les machines sont créées à partir d’images .ova préalablement à jour (apt update inutile).

## Schémas du réseau



Création des serveurs Load Balancer lb1 et lb2 sur VirtualBox

Sur ces derniers, effectuer :

- la configuration réseau

- l’installation de apache2 (uniquement pour la phase de test de heartbeat ensuite le supprimer avec **apt remove apache2**)

## Installation de heartbeat sur les serveurs lb1 et lb2

- installer heartbeat avec la commande apt install heartbeat

- création des fichiers de configurations (étapes identiques aux précédentes manipulations en remplaçant le nom des machines lb1 et lb2)

- ha.cf

- authkeys

- haresources

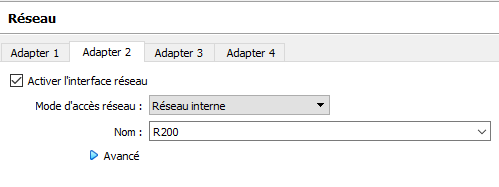
Eteindre le serveur lb1 et tenter une connexion sur l’IP virtuelle ci-dessus, si elle affiche la page de lb2, heartbeat fonctionne, il faut donc supprimer apache2 sur les deux serveurs lb et passer à l’étape suivante.

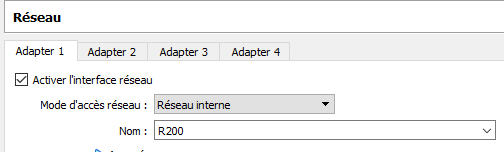
## Création des serveurs Web1 Web2 et Web3

**Objectif :** créer 3 serveurs web afin de répartir la charge entre eux à l’aide des load balancer lb1 et lb2

### Sur VirtualBox

* Créer les 3 serveurs web.
* Ajouter une interface réseaux interne dans Virtualbox aux 2 load balancer (sur laquelle seront rattachés les 3 serveurs web)



Configurer les 3 serveurs web en réseaux interne et les rattachés à la même que les load balancer

## Sur lb1 et lb2

Ajouter la nouvelle interface (enp0s8 correspondant au réseaux interne) dans le répertoire **/etc/network/interfaces**

****

**Remarque :** ne pas définir de passerelle car l’IP entrée sera la passerelle des serveurs web

Répéter l’opération pour lb2

****

### Activer le routage sur les serveurs lb1 et lb2

Dans le répertoire **/etc/sysctl.conf** enlever le commentaire sur la ligne suivante

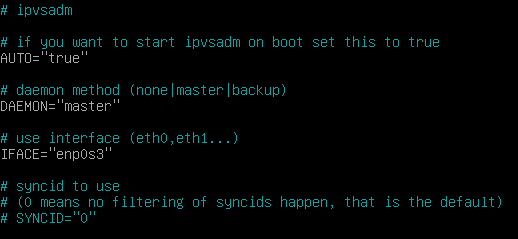
****

Reboot les machines et afficher le contenu du répertoire suivant si le résultat retourné est « 1 », le routage est effectif.



Activer et configurer les Load Balancer

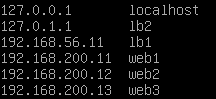
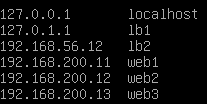
Dans le répertoire **/etc/default/ipvsadm** sur lb1 et lb2

****

Créer l’IP flottante (virtuelle) du réseaux interne (on l’affilie à l’interface enp0s8) dans le répertoire **/etc/ha.d/haresources** de lb1 et lb2

****

Ajouter les 3 serveurs web dans la liste de hôtes sur lb1 et lb2

****

## Répartir la charge aux serveurs web

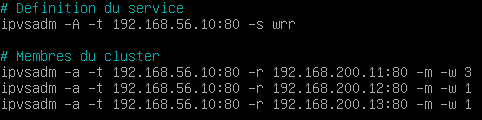
**Objectif :** Définir précisément la charge que l’on souhaite attribuer aux différents serveurs web dans notre cas :

- 3 requêtes au serveur web1

- 1 requêtes au serveur web2

- 1 requête au serveur web3

Pour se faire il faut éditer **/etc/ipvsadm.rules** (définir nos règles de répartition de charge)

****

**Explications :**

**- A →** ajoute un service, suivi du protocole, l’IP flottante:Port cible

**- t →** protocole TCP

**- S →** (save), sauvegarde les règles

**- wrr →** (weighted round robin), annonce que la répartition est définie

**-a →** affilie un serveur réel à une IP virtuelle

**-r → (**real serveur), suivi de l’IP du serveur « physique »

**- m →** (masquerade), pour que les réponses semblent revenir du répartiteur

### Sur les Serveurs web1 web2 et web3

- Installer apache2 avec la commande **apt install apache2**

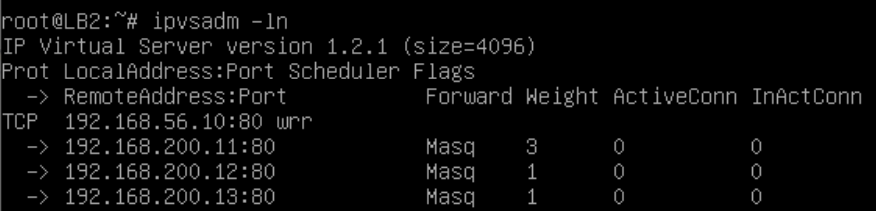
- effectuerla configuration ip des 3 serveurs en mettant l’IP flottante interne des serveurs lb1 et lb2 (ici, 192.168.200.250)

- modifier les fichiers index de la page par défaut d’apache sur les 3 serveurs web pour les identifier pendant la phase de test.

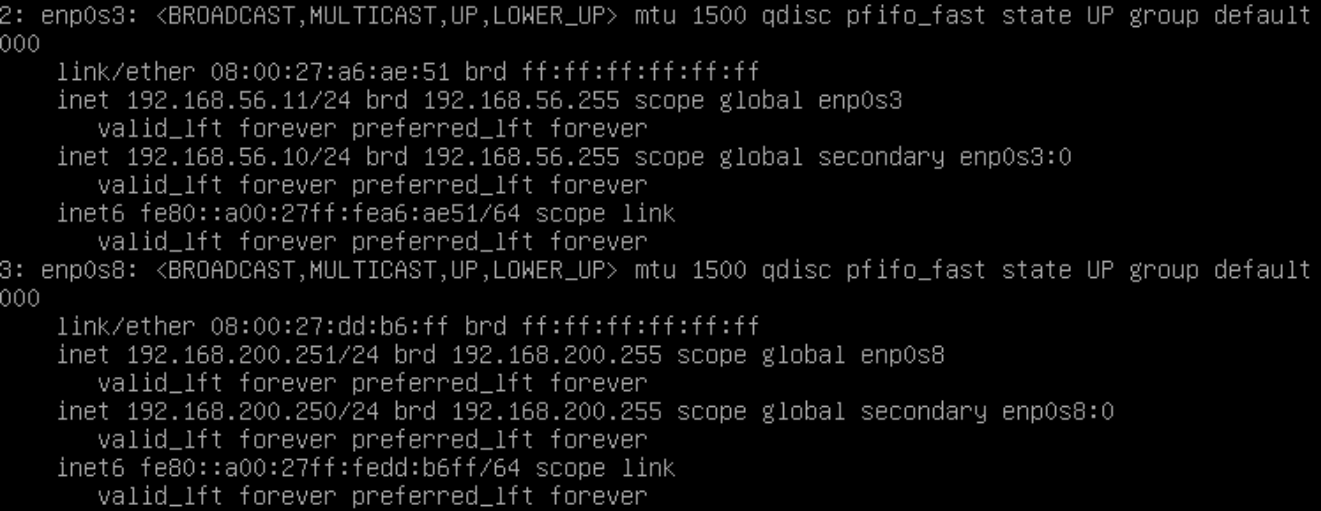
## Phase de tests

**Objectif :** tester la répartition de charge en se connectant à l’IP flottante externe des serveurs lb1 et lb2 c’est à dire 192.168.56.10 depuis la machine hôte.

Vérifier la charge affectée aux serveurs lb1 et lb2 avec la commande **ipvsadm -ln**



On éteint le serveur lb1 et on affiche la configuration réseau du serveur lb2 avec la commande **ip a**



On observe que l’IP virtuelle externe et interne ont bien été récupérées par le serveur lb2.