Ramier Jonahel

SIO2A

TP N°6 : Haute disponibilité et répartition des charges d'un serveur web avec HAProxy

Table des matières

Etape 1 : Clonage et configuration IP du cluster de serveurs web..... 3

Etape 2 : Mise à jour des fichiers de configuration d'Apache 2.0 des deux serveurs web.....6

Etape 3 : Installation et configuration de HAproxy sur un serveur basé sous Linux Debian nommé « SRVHAPROXY »..... 10

Etape 4 : Testez la répartition des charges du site web de la ville des Abymes.....13

Etape 1 : Clonage et configuration IP du cluster de serveurs web

Modification du plan d'adressage IP :

SRVWEB1 :

Nous accédons a cette via la commande nano /etc/network/interfaces.d et on modifie et ajoutes les interfaces

```
  GNU nano 7.2                                                          /e
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static

address 10.0.0.2
netmask 255.255.255.0
```

SRVWEB2 :

```
source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static

address 10.0.0.3
netmask 255.255.255.0
```

Vérification du plan d'adressage :

```
root@srvweb:/home/srvweb# ping 10.0.0.3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=0.767 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.501 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.527 ms
^C
--- 10.0.0.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2034ms
rtt min/avg/max/mdev = 0.501/0.598/0.767/0.119 ms
root@srvweb:/home/srvweb#
```

```
root@srvweb:/home/srvweb# ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.365 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.453 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=1.06 ms
^C
--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2461ms
rtt min/avg/max/mdev = 0.365/0.625/1.057/0.307 ms
root@srvweb:/home/srvweb#
```

Les serveurs se ping entre eux donc le plan d'adressage IP est fonctionnel.

Modification de /etc/hosts :

SRVWEB1 :

```
127.0.0.1       localhost
127.0.1.1       srvweb1
10.0.0.3        srvweb2

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

SRVWEB2 :

```
127.0.0.1       localhost
127.0.1.1       srvweb2
10.0.0.2        srvweb1

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

Le fichier /etc/hosts permet la résolution des noms de domaine, en permettant de traduire un nom d'hôte en adresse IP sans passer par un serveur DNS.

Explications des lignes :

127.0.0.1 localhost : Associe localhost à l'adresse de loopback, permettant aux applications locales de communiquer entre elles via localhost.

127.0.1.1 srvweb1 : Adresse utilisée par Debian pour identifier le nom d'hôte du système (si aucune IP statique n'est configurée).

10.0.0.2 srvweb2 : Définit un hôte interne nommé serveur-local avec une IP privée.

Test de ping via les noms de domaines :

SRVWEB1 :

```
srvweb@srvweb:~$ ping srvweb1
PING srvweb1 (127.0.1.1) 56(84) bytes of data.
64 bytes from srvweb1 (127.0.1.1): icmp_seq=1 ttl=64 time=0.054 ms
64 bytes from srvweb1 (127.0.1.1): icmp_seq=2 ttl=64 time=0.055 ms
64 bytes from srvweb1 (127.0.1.1): icmp_seq=3 ttl=64 time=0.051 ms
^C
--- srvweb1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2054ms
rtt min/avg/max/mdev = 0.051/0.053/0.055/0.001 ms
srvweb@srvweb:~$
```

SRVWEB2 :

```
srvweb@srvweb:~$ ping srvweb2
PING srvweb2 (127.0.1.1) 56(84) bytes of data.
64 bytes from srvweb2 (127.0.1.1): icmp_seq=1 ttl=64 time=0.050 ms
64 bytes from srvweb2 (127.0.1.1): icmp_seq=2 ttl=64 time=0.042 ms
64 bytes from srvweb2 (127.0.1.1): icmp_seq=3 ttl=64 time=0.040 ms
^C
--- srvweb2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2031ms
rtt min/avg/max/mdev = 0.040/0.044/0.050/0.004 ms
srvweb@srvweb:~$
```

Le ping est fonctionnel.

Etape 2 : Mise à jour des fichiers de configuration d'Apache 2.0 des deux serveurs web

Modification du fichier /villeabymes.conf :

La partie encadrer en rouge a été retirer pour faciliter la connexion aux sites et partie jaune car l'IP des SRVWEB ont changes.

Avant :

```
<VirtualHost 192.168.10.2:8080>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further virtual host explicitly.
        #ServerName www.example.com

        ServerName ville-abymes.fr
        ServerAlias www.ville-abymes.fr
        DocumentRoot /var/www/html/villeabymes

        <Directory /var/www/html/villeabymes>
                Options Indexes FollowSymLinks
                AllowOverride None
                Require all granted
        </Directory>

        <Directory /var/www/html/villeabymes>
                AuthType Basic
                AuthName "Accès privé - site ville des abymes"
                AuthBasicProvider file
                AuthUserFile /etc/apache2/.htusers
                Require valid-user
        </Directory>

        # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
        # error, crit, alert, emerg.
        # It is also possible to configure the loglevel for particular
        # modules, e.g.
        #LogLevel info ssl:warn

        ErrorLog ${APACHE_LOG_DIR}/villeabymes/error.log
        CustomLog ${APACHE_LOG_DIR}/villeabymes/access.log combined

        # For most configuration files from conf-available/, which are
        # enabled or disabled at a global level, it is possible to
        # include a line for only one particular virtual host. For example the
        # following line enables the CGI configuration for this host only
        # after it has been globally disabled with "a2disconf".
        #Include conf-available/serve-cgi-bin.conf
</VirtualHost>
```

Après :

```
                                                        /etc/apache2/sites-available
<VirtualHost 10.0.0.2:8080>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further virtual host explicitly.
        #ServerName www.example.com

        ServerName ville-abymes.fr
        ServerAlias www.ville-abymes.fr
        DocumentRoot /var/www/html/villeabymes

        <Directory /var/www/html/villeabymes>
                Options Indexes FollowSymLinks
                AllowOverride None
                Require all granted
        </Directory>


        # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
        # error, crit, alert, emerg.
        # It is also possible to configure the loglevel for particular
        # modules, e.g.
        #LogLevel info ssl:warn

        ErrorLog ${APACHE_LOG_DIR}/villeabymes/error.log
        CustomLog ${APACHE_LOG_DIR}/villeabymes/access.log combined

        # For most configuration files from conf-available/, which are
        # enabled or disabled at a global level, it is possible to
        # include a line for only one particular virtual host. For example the
        # following line enables the CGI configuration for this host only
        # after it has been globally disabled with "a2disconf".
        #Include conf-available/serve-cgi-bin.conf
</VirtualHost>
```

Modification du fichier index.html :

Avant :

```
<!DOCTYPE html>
<html lang="fr">
<head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>

<body>
        <header>
                <h1>Bienvenue sur le site ville des abymes</h1>
        </header>
</body>

</html>
```

SRVWEB1 apres :

```
  GNU nano 7.2                                                    /var/www/html/vill
<!DOCTYPE html>
<html lang="fr">
<head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>

<body>
        <header>
                <h1>Bienvenue sur site n°1 de la  ville des abymes</h1>
        </header>
</body>

</html>
```

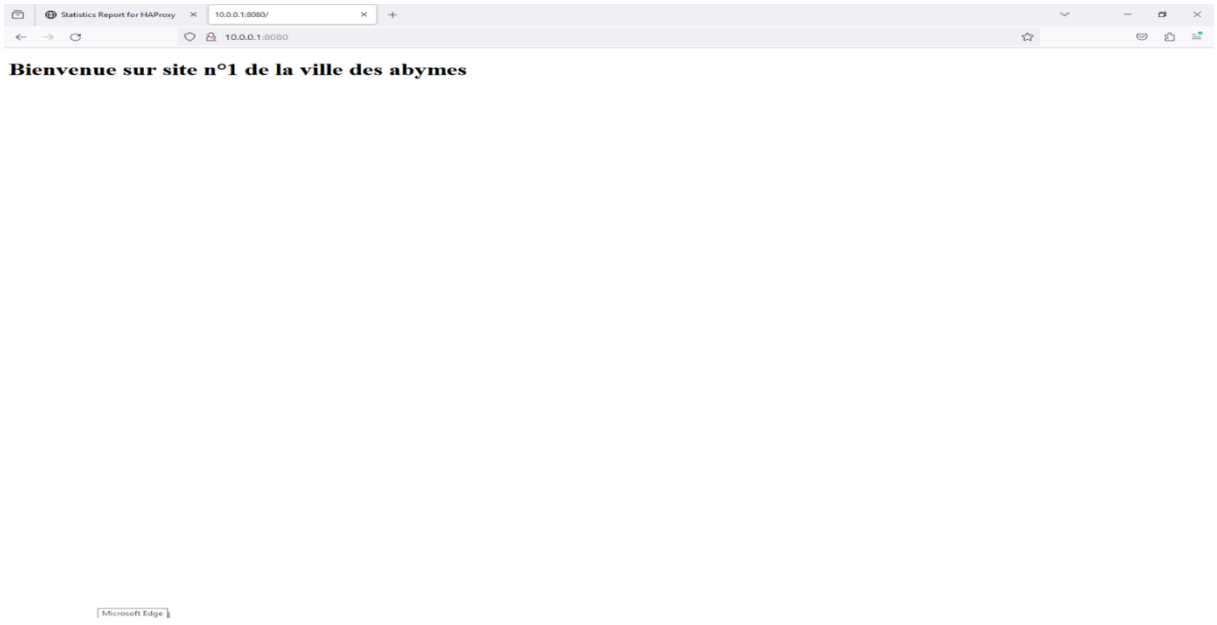
SRVWEB2 après :

```html
<!DOCTYPE html>
<html lang="fr">
<head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>

<body>
        <header>
                <h1>Bienvenue sur le site n°2 de la ville des abymes</h1>
        </header>
</body>

</html>
```

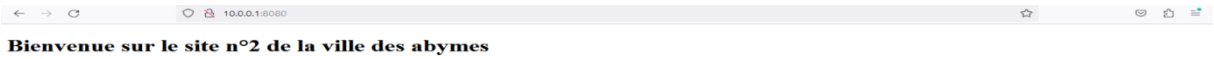
Test d’accessibilité aux sites :

Pour tester la connexion aux sites il faut s’y connecter via un client avec un navigateur internet en y tapant l’IP.

SRVWEB1 :



SRVWEB2 :



Les sites sont accessibles.

Etape 3 : Installation et configuration de HAproxy sur un serveur basé sous Linux Debian nommé « SRVHAPROXY »

Modification du plan d'adressage IP :

SRVHAPROXY :

Nous accédons a cette via la commande nano /etc/network/interfaces.d et on modifie et ajoutes les interfaces

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
address 10.0.0.1
netmask 255.255.255.0

allow-hotplug enp0s8
iface enp0s8 inet static
address 192.168.10.4
netmask 255.255.255.0
gateway 192.168.10.1
```

Modification de /etc/hosts :

SRVHAPROXY :

```
  GNU nano 7.2
127.0.0.1       localhost
127.0.1.1       srvhaproxy
192.168.10.4    srvhaproxy
10.0.0.2        srvweb1
10.0.0.3        srvweb2

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

Test de ping :

```
srvweb@srvweb:~$ ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=0.525 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.552 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.609 ms
```

Ping fonctionnel.

#systemctl enable haproxy :

```
root@srvnagios:/home/srvnagios# systemctl status haproxy
● haproxy.service - HAProxy Load Balancer
     Loaded: loaded (/lib/systemd/system/haproxy.service; enabled; preset: enabled)
     Active: active (running) since Mon 2025-03-17 15:22:18 CET; 10min ago
       Docs: man:haproxy(1)
             file:/usr/share/doc/haproxy/configuration.txt.gz
   Main PID: 24599 (haproxy)
      Tasks: 3 (limit: 2315)
     Memory: 47.1M
        CPU: 281ms
     CGroup: /system.slice/haproxy.service
             ├─24599 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid -S /run/haproxy-master.sock
             └─24601 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid -S /run/haproxy-master.sock

mars 17 15:22:18 srvnagios systemd[1]: Starting haproxy.service - HAProxy Load Balancer...
mars 17 15:22:18 srvnagios haproxy[24599]: [NOTICE]   (24599) : New worker (24601) forked
mars 17 15:22:18 srvnagios haproxy[24599]: [NOTICE]   (24599) : Loading success.
mars 17 15:22:18 srvnagios systemd[1]: Started haproxy.service - HAProxy Load Balancer.
```

Le service est lancer.

Rôle des lignes du fichier haproxy.cfg :

frontend site_web : Définit un frontend nommé site_web pour écouter les requêtes entrantes.

bind *:80 : Fait écouter HAProxy sur le port **80** (HTTP) sur toutes les interfaces réseau (*).

option httpclose : Ferme la connexion HTTP après chaque requête pour éviter les connexions persistantes et optimiser les ressources.

default_backend serveurs : Spécifie que toutes les requêtes reçues seront envoyées au backend nommé **serveurs**.

backend serveurs : Définit un backend nommé serveurs, qui recevra les requêtes du frontend.

balance roundrobin : Utilise l'algorithme **Round Robin**, distribuant les requêtes de manière équilibrée entre les serveurs.

server srvweb1 10.0.0.2:80 weight 1 check :

- Définit un serveur backend nommé srvweb1 avec l'adresse **10.0.0.2** et le port **80**.
- weight 1 : Indique un poids de **1** (tous les serveurs ont un poids égal, donc répartition égale du trafic).
- check : Active une vérification de l'état du serveur (monitoring de disponibilité).

server srvweb2 10.0.0.3:80 weight 1 check : Même principe que pour srvweb1, mais pour un second serveur.

listen stats : Déclare une nouvelle section permettant de superviser HAProxy via une interface web.

bind *:8404 : L'interface de statistiques est accessible via le port **8404**.

mode http : Définit le mode **HTTP** pour cette interface.

stats enable : Active les statistiques.

stats uri /statshaproxy : Définit l'URL d'accès à la page des statistiques (http://serveur:8404/statshaproxy).

stats auth admin:@admin2425@ : Ajoute une authentification avec l'utilisateur **admin** et le mot de passe **@admin2425@** pour accéder aux statistiques.

stats refresh 20s : Rafraîchit automatiquement la page des statistiques toutes les **20 secondes**.

#sudo haproxy –c –f haproxy.cfg :

Après cette commande taper il y a plusieurs erreurs détecter.

-c : Permet de vérifier la syntaxe du fichier de configuration sans démarrer HAProxy.

-f : Spécifie le **fichier de configuration** à utiliser (haproxy.cfg dans cet exemple).

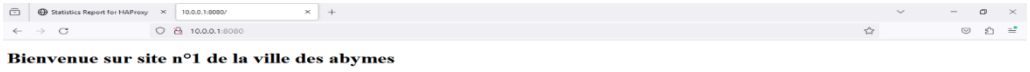
Commande pour relancer haproxy : systemctl restart haproxy

Etape 4 : Testez la répartition des charges du site web de la ville des Abymes

Scénario pour tester le bon fonctionnement de haproxy :

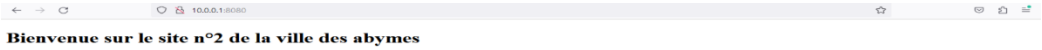
Le premier scenario, nous allons simplement rafraichir la page et nous constatons que les deux sites alterne via l'affichage

Avant rafraichissement :

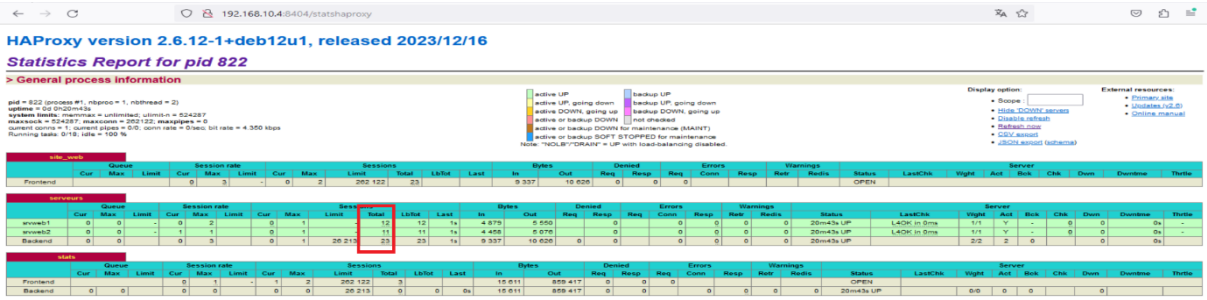


Bienvenue sur site n°1 de la ville des abymes

Après rafraichissement :



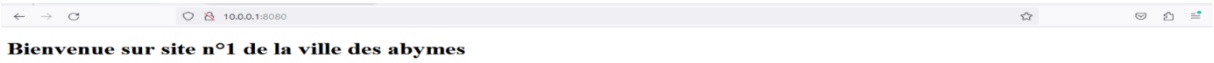
Vérification de la repartions des charges :



Nous constatons que le nbr de sessions est quasiment égal ce qui valide la répartition des charges.

Pour le deuxième scenario, nous allons fermer un serveur pour simuler une panne et accéder au site via l'IP de haproxy.

Après la fermeture du SRVWEB2 :



Bienvenue sur site n°1 de la ville des abymes

Ce scenario nous prouve encore la bon fonctionnement de la repartions des charger.