

Continuité de service HAProxy

Introduction

1. Mise en place d'un HAProxy
2. Modification sur les deux serveurs GLPI
3. Test du HAProxy et des serveurs
4. Mise en place des statistiques
5. Test de continuité de service

Introduction

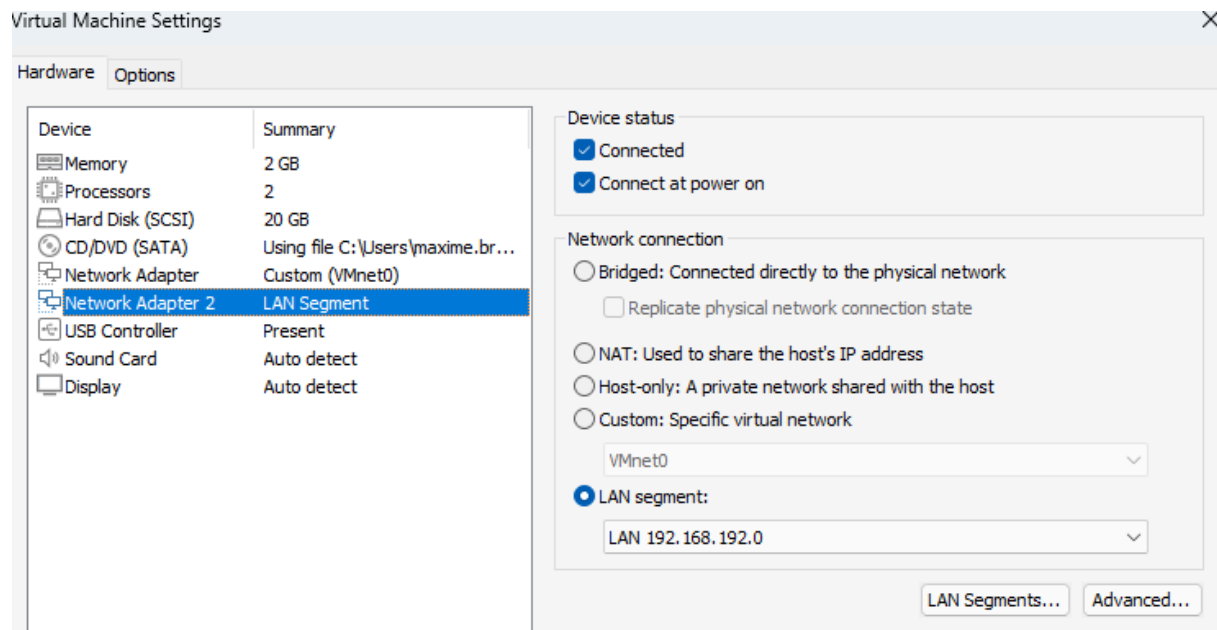
HAProxy (High Availability Proxy) est un répartiteur de charge (load balancer) open-source reconnu pour sa performance, sa fiabilité et sa flexibilité. Il est couramment utilisé pour distribuer le trafic réseau entre plusieurs serveurs, assurant ainsi une meilleure disponibilité, une montée en charge efficace et une tolérance aux pannes.

Grâce à ses nombreuses fonctionnalités, HAProxy est un composant clé dans les architectures orientées haute disponibilité et dans la gestion du trafic HTTP(s) et TCP à grande échelle.

Dans ce guide, nous verrons comment installer et configurer HAProxy sur un serveur LAMP (Linux, Apache, MySQL, PHP) fonctionnant sous Debian 12. L'objectif sera de mettre en place une solution simple et fonctionnelle pour répartir le trafic vers plusieurs backends tout en assurant stabilité et performance.

1. Mise en place d'un HAProxy

Voici les paramètres à appliquer à la machine HAProxy:



On installe apache et haproxy:

```
apt -y install apache2
apt -y install haproxy
```

On va ajouter les deux adresses IP des serveurs:

```
nano /etc/hosts
```

```
GNU nano 7.2
127.0.0.1      localhost
127.0.1.1      haproxy
192.168.192.130 srv1
192.168.192.131 srv2_
```

On va paramétrer l'IP de la machine HAProxy pour communiquer avec les serveurs

```
nano /etc/network/interfaces
```

```
# This file describes the network interfaces
# and how to activate them. For more infor

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug ens33
iface ens33 inet dhcp

allow-hotplug ens37
iface ens37 inet static
address 192.168.192.132/24
```

On ajoute le frontend et le backend au HAProxy:

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

```

GNU nano 7.2 /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=intermediate
    ssl-default-bind-ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-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
    option        httplog
    option        dontlognull
    timeout      connect 5000
    timeout      client  50000
    timeout      server  50000
    errorfile    400 /etc/haproxy/errors/400.http
    errorfile    403 /etc/haproxy/errors/403.http
    errorfile    408 /etc/haproxy/errors/408.http
    errorfile    500 /etc/haproxy/errors/500.http
    errorfile    502 /etc/haproxy/errors/502.http
    errorfile    503 /etc/haproxy/errors/503.http
    errorfile    504 /etc/haproxy/errors/504.http

frontend frontend-base
    bind *:80
    default_backend backend-base
    option forwardfor

backend backend-base
    balance roundrobin
    server web1 192.168.192.130:80 check
    server web2 192.168.192.131:80 check

```

2. Modification sur les deux serveurs GLPI

On va s'assurer que les hostname et les IP des machines correspondent à celles du HAProxy:

```
hostnamectl set-hostname nomDuServeur
```

```
nano /etc/network/interfaces
```

```
# This file describes the network in
# and how to activate them. For more

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug ens33
iface ens33 inet static
address 192.168.192.130/24
```

```
GNU nano 7.2
# This file describes the network inte

source /etc/network/interfaces.d/*

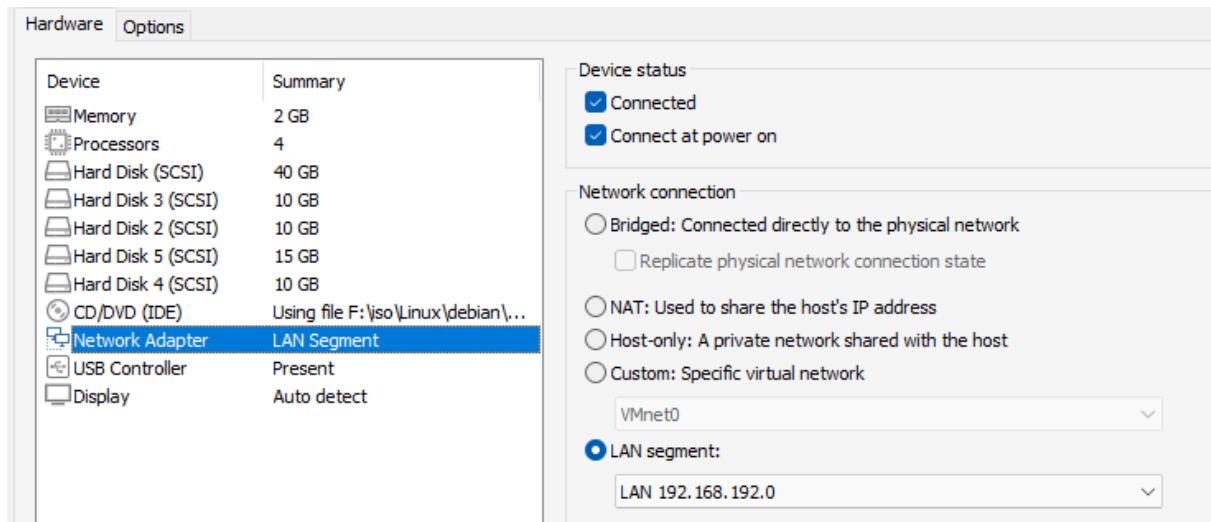
# The loopback network interface
auto lo
iface lo inet loopback
# The primary network interface
allow-hotplug ens33
iface ens33 inet static
address 192.168.192.131
netmask 255.255.255.0
```

On va modifier la page sur laquelle on sera redirigé pour voir quel serveur est utilisé

```
nano /var/www/html/index.html
```

```
div.validator {
}
</style>
</head>
<body>
  <div class="main_page">
    <div class="page_header floating_element">
      
      <span class="floating_element">
        SRV2
      </span>
    </div>
    <!-- <div class="table_of_contents floating_element">
      <div class="section header section header grey">
```

On s'assure que les serveurs GLPI soient bien sur le même LAN Segment que le HAProxy



Redémarrer les deux machines

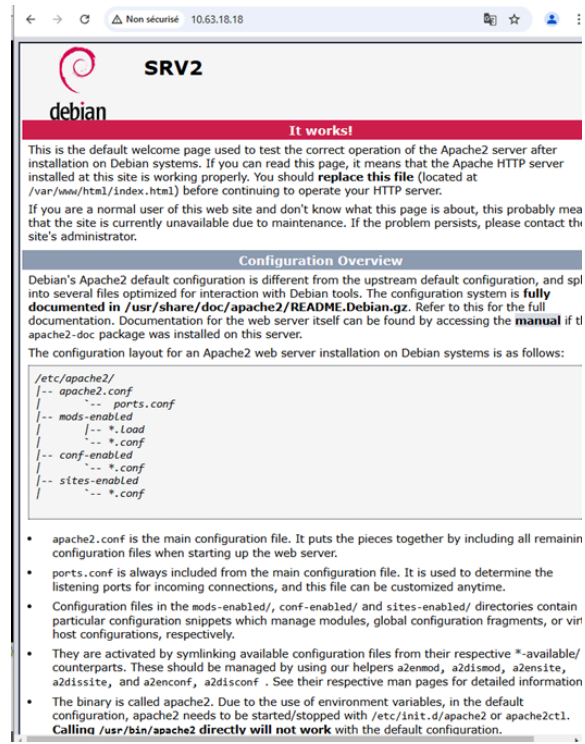
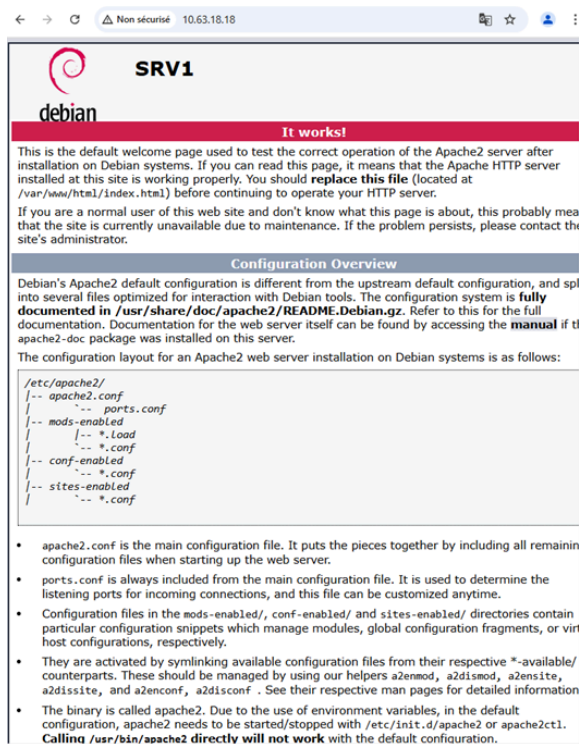
```
reboot
```

Redémarrer le HAProxy

```
systemctl restart haproxy
```

3. Test du HAProxy et des serveurs

On navigue vers l'ip associé au serveur HAProxy, ici 10.63.18.18. On obtient donc en affichage une page avec le nom du serveur utilisé. En appuyant sur F5 on voit le balancement entre les serveurs:



4. Mise en place des statistiques

On va retourner dans le fichier de config du HAProxy pour ajouter la page de statistiques et mieux visualiser:

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

```
listen httpProxy
    bind 10.63.18.18:80
    balance roundrobin
    server srv1 192.168.192.130:80 check
    server srv2 192.168.192.131:80 check

listen stats
    bind *:8080
    stats enable
    stats uri /statsHaproxy
    stats auth admin:password
    stats refresh 30s
```

On redémarre le HAProxy pour les modifications:

systemctl restart haproxy

On navigue vers l'adresse 10.63.18.18:8080/statsHaproxy et on obtient l'affichage suivant:

← → ↻ Non sécurisé 10.63.18.18:8080/statsHaproxy

HAProxy version 2.6.12-1+deb12u1, released 2023/12/16

Statistics Report for pid 1052

> General process information

pid = 1052 (process #1, nbproc = 1, nbthread = 2)
uptime = 0d 0h02m08s
system limits: memmax = unlimited; ulimit-n = 624288
maxsock = 524288; maxconn = 262121; maxpipes = 0
current conns = 2; current pipes = 0/0; conn rate = 0/sec; bit rate = 0.000 kbps
Running tasks: 0/22; idle = 100 %

active UP backup UP
active UP, going down backup UP, going down
active DOWN, going up backup DOWN, going up
active or backup DOWN not checked
active or backup DOWN for maintenance (MAINT)
active or backup SOFT STOPPED for maintenance

Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

Display option:
• Scope:
• Hide 'DOWN' servers
• Disable refresh
• Refresh now
• CSV export
• JSON export (schema)

External resources:
• Primary site
• Updates (v2.6)
• Online manual

frontend-base																															
	Queue			Session rate			Sessions				Bytes		Denied	Errors		Warnings		Server													
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle	
Frontend	0	0	-	0	0	0	0	0	262	121	0	?	0	0	0	0	0	0	0	0	0	0	OPEN								

backend-base																														
	Queue			Session rate			Sessions				Bytes		Denied	Errors		Warnings		Server												
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
srv1	0	0	-	0	0	0	0	0	-	0	0	?	0	0	0	0	0	0	0	0	0	2m8s UP	L4OK in 1ms	1/1	Y	-	0	0	0s	-
srv2	0	0	-	0	0	0	0	0	-	0	0	?	0	0	0	0	0	0	0	0	0	2m8s UP	L4OK in 1ms	1/1	Y	-	0	0	0s	-
Backend	0	0	-	0	0	0	0	0	26	213	0	?	0	0	0	0	0	0	0	0	0	2m8s UP		2/2	2	0		0	0s	

httpProxy																														
	Queue			Session rate			Sessions				Bytes		Denied	Errors		Warnings		Server												
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend	0	2	-	0	2	0	2	262	121	2			631	3 293	0	0	1					OPEN								
srv1	0	0	-	0	1	0	1	-	1	1	2m1s	631	3 293	0	0	0	0	0	0	0	0	2m8s UP	L4OK in 0ms	1/1	Y	-	0	0	0s	-
srv2	0	0	-	0	0	0	0	-	0	0	?	0	0	0	0	0	0	0	0	0	0	2m8s UP	L4OK in 1ms	1/1	Y	-	0	0	0s	-
Backend	0	0	-	0	1	0	1	26	213	1	1	2m1s	631	3 293	0	0	0	0	0	0	0	2m8s UP		2/2	2	0		0	0s	

stats																														
	Queue			Session rate			Sessions				Bytes		Denied	Errors		Warnings		Server												
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend	0	5	-	2	2	262	121	10					5 496	100 597	0	0	1					OPEN								
Backend	0	0	-	0	4	0	1	26	213	6	0	0s	5 496	100 597	0	0	6	0	0	0	0	2m8s UP		0/0	0	0		0	0s	

5. Test de continuité de service

Pour tester si la continuité de service est assurée, on va effectuer un test. On va shutdown l'un des deux serveurs, vérifier avec les statistiques si le serveur est bien down et ensuite on devrait voir uniquement le serveur 1 utilisé avec l'aide de la page.

shutdown now


Ici les statistiques montrent bien que le serveur 2 est down et que seul le 1 est up:

backend-base																														
	Queue			Session rate			Sessions				Bytes		Denied		Errors			Warnings			Server									
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
svr1	0	0	-	0	0		0	0	-	0	0	?	0	0	0	0	0	0	0	0	0	4m UP	L4OK in 0ms	1/1	Y	-	0	0	0s	-
svr2	0	0	-	0	0		0	0	-	0	0	?	0	0	0	0	0	0	0	0	0	3s DOWN	L4TOUT in 2000ms	1/1	Y	-	3	1	33s	-
Backend	0	0		0	0		0	0		26 213	0	0	?	0	0	0	0	0	0	0	0	4m UP		1/1	1	0	0	0	0s	

httpProxy																																
Queue				Session rate				Sessions				Bytes		Denied		Errors		Warnings		Server												
	Cur	Max	Limit	Cur	Max	Limit		Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrle	
Frontend				0	2		-	0	2	262	121	2				631	3 293	0	0	1			OPEN									
svr1	0	0	-	0	1			0	1	-	1	1	3m53s	631	3 293	0	0	0	0	0	0	4m UP	L4OK in 0ms	1/1	Y	-	0	0	0	0s	-	
svr2	0	0	-	0	0			0	0	-	0	0	?	0	0	0	0	0	0	0	0	30s DOWN	* L4TOUT in 2001ms	1/1	Y	-	3	1	30s	-		
Backend	0	0		0	1			0	1	26	213	1	1	3m53s	631	3 293	0	0	0	0	0	4m UP		1/1	1	0	0	1	0	0s		

On navigue vers la page du serveur HAProxy et on refresh en boucle:

Non sécurisé
10.63.18.18


SRV1
debian

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/
|-- apache2.conf
```

Donc seul la page du serveur 1 s'affiche, la continuité du service est donc assurée.