Documentation : HAProxy (High Availability Proxy)

Présentation

HAProxy (High Availability Proxy) est une solution open-source de répartition de charge et de proxy inverse pour les protocoles HTTP, HTTPS et TCP. Il est extrêmement stable, rapide, et utilisé dans les architectures haute disponibilité pour répartir le trafic entre plusieurs serveurs backend.

- **Type** : Load balancer / Reverse proxy
- **Protocoles supportés** : HTTP, HTTPS, TCP, WebSocket
- **Fonctionnalités** : Terminaison SSL, surveillance, session sticky, interface web de statistiques
- **Interface de gestion** : Fichier `haproxy.cfg`, Web UI, CLI

Prérequis système

- **CPU** : x86_64 / ARM supportés
- **RAM** : 512 Mo min (1 Go recommandé)
- **Disque** : < 100 Mo
- **Accès root ou** `sudo`
- **Réseau** : Interfaces configurées

Installation de HAProxy

Debian / Ubuntu

```
sudo apt update && sudo apt install haproxy -y
```

CentOS

```
sudo dnf install haproxy -y
```

Arch Linux

```
sudo pacman -Syu haproxy
```

Vérification de la version

```
haproxy -v
# Exemple : HAProxy version 2.6.9
```

Configuration de base

Fichier de configuration principal

> Emplacement : `/etc/haproxy/haproxy.cfg`

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

Exemple : Load Balancing HTTP (Round Robin)

```
global
    log /dev/log local0
    maxconn 2048
    daemon
    user haproxy
    group haproxy

defaults
    log global
    mode http
    option httplog
    option dontlognull
    timeout connect 5000
    timeout client 50000
    timeout server 50000

frontend http_front
    bind *:80
    default_backend web_servers

backend web_servers
    balance roundrobin
    server srv1 192.168.10.101:80 check
    server srv2 192.168.10.102:80 check
```

Interface Statistique Web (optionnel)

Ajouter à la configuration :

```
listen stats
    bind *:8080
    stats enable
    stats uri /haproxy?stats
    stats refresh 10s
    stats auth admin:motdepasse
```

Accès :

```
http://IP_SERVEUR:8080/haproxy?stats
```

SSL / HTTPS (Optionnel)

Exemple de frontend HTTPS

```
frontend https_front
    bind *:443 ssl crt /etc/ssl/certs/cert.pem
    default_backend web_servers
```

> Le certificat `.pem` doit contenir : certificat + clé privée

Sécurité & Optimisations

- Restreindre l'accès à l’interface stats (IP ou VPN)
- Intégrer un pare-feu (iptables/nftables, UFW, firewalld)
- Intégrer `fail2ban` ou WAF si frontal web exposé
- Superviser avec Prometheus, Grafana ou Zabbix via exporter

Service & Démarrage

Activer au boot et démarrer

Debian / Ubuntu / Arch

```
sudo systemctl enable haproxy
sudo systemctl start haproxy
```

CentOS / RHEL

```
sudo systemctl enable --now haproxy
```

Vérification du fonctionnement

Tester les ports ouverts

```
sudo ss -tuln | grep 80
```

Curl de test :

Depuis un autre appareil :

```
curl -I http://IP_SERVEUR
```

Vérification finale

- HAProxy démarre sans erreurs : `systemctl status haproxy`
- Logs actifs : `journalctl -u haproxy` ou `/var/log/haproxy.log`
- Web app accessible via IP publique
- Statistiques visibles si activées
- Certificats SSL valides si HTTPS activé