L’objectif de ce document est de présenter les actions réalisées et les vérifications effectuées afin de renforcer la sécurité d’un serveur web Nginx hébergeant plusieurs sites d’une entreprise, tout en garantissant la stabilité de l’environnement de production.

La démarche adoptée repose sur une approche progressive : analyse de l’existant, suppression des services inutiles, vérification des mécanismes de protection déjà en place, puis mise en œuvre de mesures de sécurité validées.

Analyse des services installés

La première étape consiste à identifier les services présents sur le serveur afin de réduire la surface d’attaque.

Une liste des services installés et actifs a été réalisée afin de vérifier leur légitimité par rapport à l'usage du serveur.

Analyse des ports ouverts

Une vérification des ports ouverts a été effectuée afin d'identifier les services exposés sur le réseau.

Cette étape permet de s'assurer que seuls les ports strictement nécessaires au fonctionnement des sites web sont accessibles depuis l'extérieur.

```
rami@vps-01b39619:/$ ss -tulnp
Netid               State                Recv-Q               Send-Q                              Local Address:Port                              Peer Address:Port              Process
udp                 UNCONN               0                    0                                      127.0.0.54:53                                   0.0.0.0:*
udp                 UNCONN               0                    0                                   127.0.0.53%lo:53                                   0.0.0.0:*
udp                 UNCONN               0                    0                                51.178.54.154%ens3:68                                 0.0.0.0:*
tcp                 LISTEN               0                    80                                   127.0.0.1:3306                                 0.0.0.0:*
tcp                 LISTEN               0                    4096                                127.0.0.53%lo:53                                   0.0.0.0:*
tcp                 LISTEN               0                    4096                                   127.0.0.54:53                                   0.0.0.0:*
tcp                 LISTEN               0                    4096                                     0.0.0.0:22                                    0.0.0.0:*
tcp                 LISTEN               0                    511                                      0.0.0.0:80                                    0.0.0.0:*
tcp                 LISTEN               0                    511                                      0.0.0.0:443                                   0.0.0.0:*
tcp                 LISTEN               0                    4096                                        [::]:22                                       [::]:*
tcp                 LISTEN               0                    511                                         [::]:443                                      [::]:*
```

Gestion des services web

Lors de l'analyse, il a été constaté que deux services web étaient présents sur le serveur :

- Nginx (utilisé en production)
- Apache2 (non utilisé)

Le service Apache2 n'étant pas fonctionnel et n'ayant aucune utilité dans l'architecture actuelle, il a été décidé de le supprimer afin d'éviter toute exposition inutile.

```
rami@vps-01b39619:/$ sudo systemctl status apache2
× apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: failed (Result: exit-code) since Mon 2026-01-05 14:12:13 UTC; 55min ago
 Invocation: acc0961f50804ba185a3f31df552ffe1
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 1035 ExecStart=/usr/sbin/apachectl start (code=exited, status=1/FAILURE)
   Mem peak: 7.8M
        CPU: 73ms

Jan 05 14:12:13 vps-01b39619 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Jan 05 14:12:13 vps-01b39619 apachectl[1084]: (98)Address already in use: AH00072: make_sock: could not bind to address [::]:80
Jan 05 14:12:13 vps-01b39619 apachectl[1084]: (98)Address already in use: AH00072: make_sock: could not bind to address 0.0.0.0:8
Jan 05 14:12:13 vps-01b39619 apachectl[1084]: no listening sockets available, shutting down
Jan 05 14:12:13 vps-01b39619 apachectl[1084]: AH00015: Unable to open logs
Jan 05 14:12:13 vps-01b39619 systemd[1]: apache2.service: Control process exited, code=exited, status=1/FAILURE
Jan 05 14:12:13 vps-01b39619 systemd[1]: apache2.service: Failed with result 'exit-code'.
Jan 05 14:12:13 vps-01b39619 systemd[1]: Failed to start apache2.service - The Apache HTTP Server.
rami@vps-01b39619:/$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
     Active: active (running) since Mon 2026-01-05 14:12:13 UTC; 55min ago
 Invocation: e80730723cef454b8e1890e8a0250b50
       Docs: man:nginx(8)
    Process: 1046 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
    Process: 1085 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Main PID: 1090 (nginx)
      Tasks: 5 (limit: 9250)
     Memory: 14.2M (peak: 15.8M)
        CPU: 954ms
     CGroup: /system.slice/nginx.service
             ├─1090 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             ├─1091 "nginx: worker process"
             ├─1092 "nginx: worker process"
             ├─1093 "nginx: worker process"
             └─1094 "nginx: worker process"

Jan 05 14:12:13 vps-01b39619 systemd[1]: Starting nginx.service - A high performance web server and a reverse proxy server...
Jan 05 14:12:13 vps-01b39619 systemd[1]: Started nginx.service - A high performance web server and a reverse proxy server.
```

Action réalisée

Suppression du service Apache2.

```
rami@vps-01b39619:/$ sudo systemctl stop apache2
rami@vps-01b39619:/$ sudo apt remove apache2
The following packages were automatically installed and are no longer required
  apache2-data  apache2-utils
Use 'sudo apt autoremove' to remove them.

REMOVING:
  apache2

Summary:
  Upgrading: 0, Installing: 0, Removing: 1, Not Upgrading: 16
  Freed space: 467 kB

Continue? [Y/n] Y
(Reading database ... 131636 files and directories currently installed.)
Removing apache2 (2.4.63-1ubuntu1.1) ...
Processing triggers for man-db (2.13.0-1) ...
```

Sécurisation des accès SSH

Le serveur étant hébergé sur un VPS OVH, certaines mesures de sécurité sont déjà mises en place par le fournisseur, notamment concernant l'accès root.

L'accès SSH root direct n'est pas utilisé pour l'administration quotidienne. L'administration se fait via un compte utilisateur disposant des privilèges nécessaires.

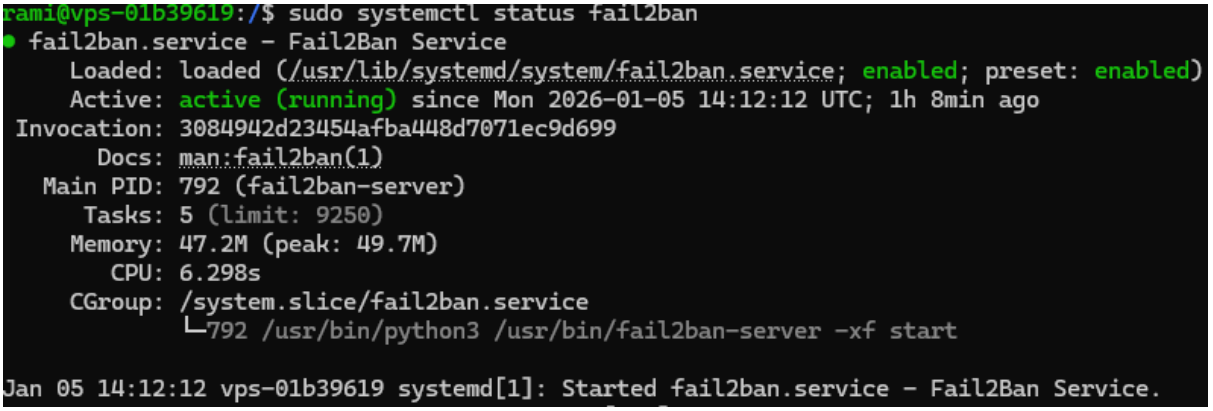
Afin de ne pas impacter la stabilité de l'environnement de production, aucune modification n'a été apportée à la configuration SSH existante.

Vérification du service Fail2ban

Le service Fail2ban a été vérifié afin de s'assurer qu'un mécanisme de protection contre les attaques par force brute est actif.

État du service

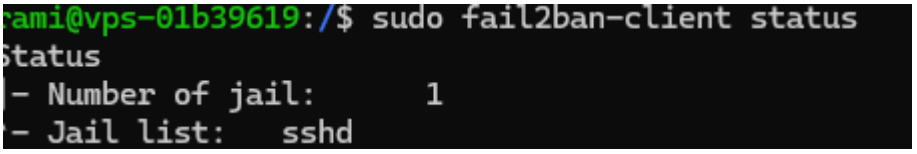
Fail2ban est actif et fonctionnel.

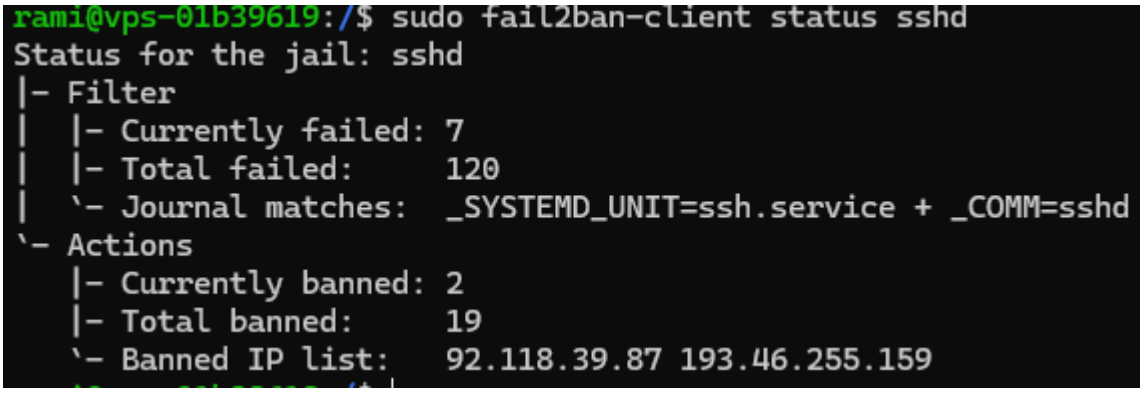


Analyse des jails Fail2ban

Les jails configurés ont été examinés afin de vérifier la protection des services critiques, notamment SSH.

Il a été constaté que plusieurs jails sont déjà configurés et actifs, assurant une protection contre les tentatives de connexion abusives.



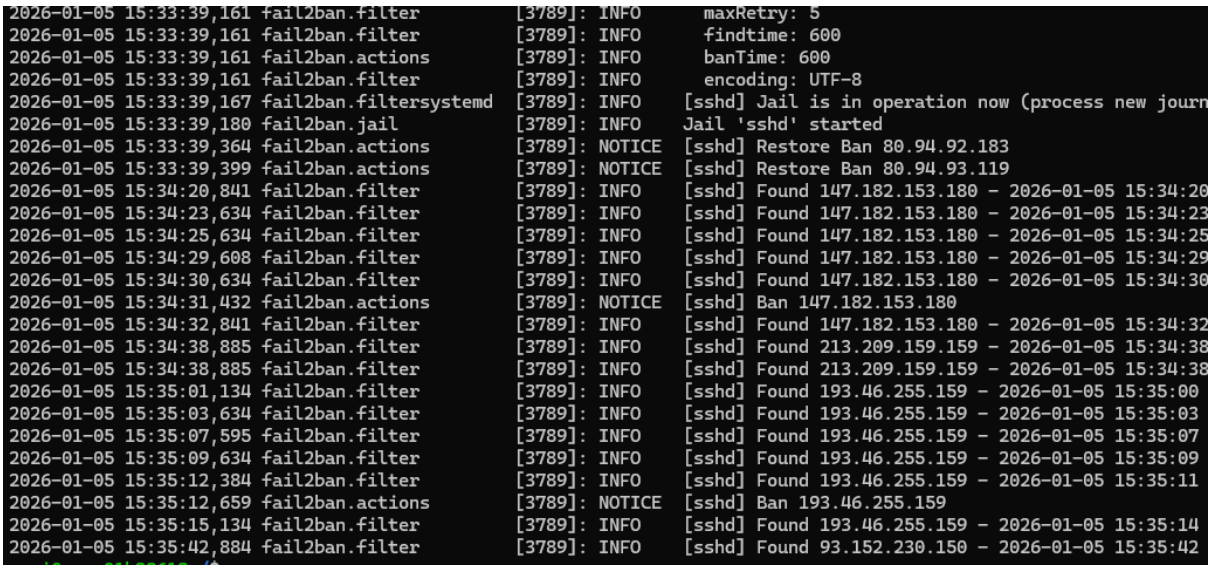


Vérification des bannissements récents

Une vérification des derniers bannissements effectués par Fail2ban a été réalisée afin de confirmer son bon fonctionnement.

Commande utilisée

sudo tail -n 50 /var/log/fail2ban.log



Mise en place du pare-feu système (UFW)

Afin de renforcer la sécurité réseau du serveur, il a été décidé de mettre en place un pare-feu système basé sur **UFW (Uncomplicated Firewall)**.

L'objectif est de **restreindre strictement les accès réseau** en n'autorisant que les services nécessaires au fonctionnement du serveur web.

Vérification de la présence d'UFW

Avant toute configuration, une vérification a été effectuée afin de confirmer la présence du pare-feu UFW sur le système.

Cette étape permet de s'assurer que l'outil de filtrage réseau est bien disponible avant son activation.

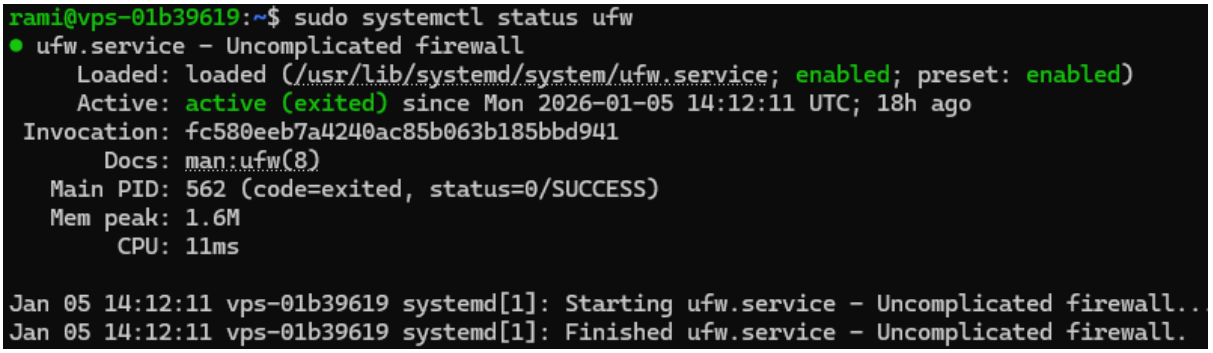
Vérification de la présence du pare-feu UFW

Avant toute configuration, une vérification a été effectuée afin de confirmer que le pare-feu UFW est bien installé et disponible sur le serveur.

Commande utilisée

sudo systemctl status ufw

Le service UFW est présent sur le système et activable.



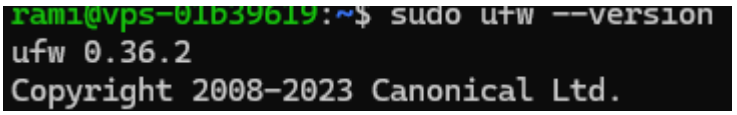
Vérification de la version d'UFW

La version du pare-feu a ensuite été vérifiée afin de confirmer l'outil utilisé.

Commande utilisée

sudo ufw --version

Le serveur utilise la version **0.36.2 (Canonical LTD)**.



Vérification de l'état actuel du pare-feu

Avant toute modification, l'état du pare-feu a été contrôlé afin de s'assurer qu'aucune règle n'était encore appliquée.

Commande utilisée

sudo ufw status verbose

Le pare-feu est **inactif** à ce stade, ce qui permet de préparer les règles sans impact immédiat sur les connexions en cours.

```
rami@vps-01b39619:~$ sudo ufw status verbose
Status: inactive
```

Autorisation des services nécessaires

Avant l'activation du pare-feu, les règles autorisant les services indispensables ont été ajoutées.

Autorisation de l'accès SSH

L'accès SSH est autorisé en priorité afin d'éviter toute coupure d'administration distante lors de l'activation du pare-feu.

sudo ufw allow ssh

```
rami@vps-01b39619:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
```

Autorisation des services web HTTP et HTTPS

Les ports nécessaires au fonctionnement des sites web ont ensuite été autorisés.

sudo ufw allow 80/tcp

```
rami@vps-01b39619:~$ sudo ufw allow 80/tcp
Rules updated
Rules updated (v6)
```

sudo ufw allow 443/tcp

```
rami@vps-01b39619:~$ sudo ufw allow 443/tcp
Rules updated
Rules updated (v6)
```

Vérification des règles configurées

Une vérification des règles ajoutées a été réalisée afin de s'assurer que seuls les ports nécessaires sont autorisés.

Les règles visibles concernent uniquement :

- le port **22 (SSH)**,
- le port **80 (HTTP)**,
- le port **443 (HTTPS)**.

sudo ufw show added

```
rami@vps-01b39619:~$ sudo ufw show added
Added user rules (see 'ufw status' for running firewall):
ufw allow 22/tcp
ufw allow 80/tcp
ufw allow 443/tcp
```

Activation du pare-feu UFW

Une fois l'ensemble des règles vérifiées, le pare-feu UFW a été activé.

Lors de l'activation, un message d'avertissement indique que l'opération peut perturber les connexions SSH existantes.
Cet avertissement est standard et a été validé après confirmation que l'accès SSH était bien autorisé.

sudo ufw enable

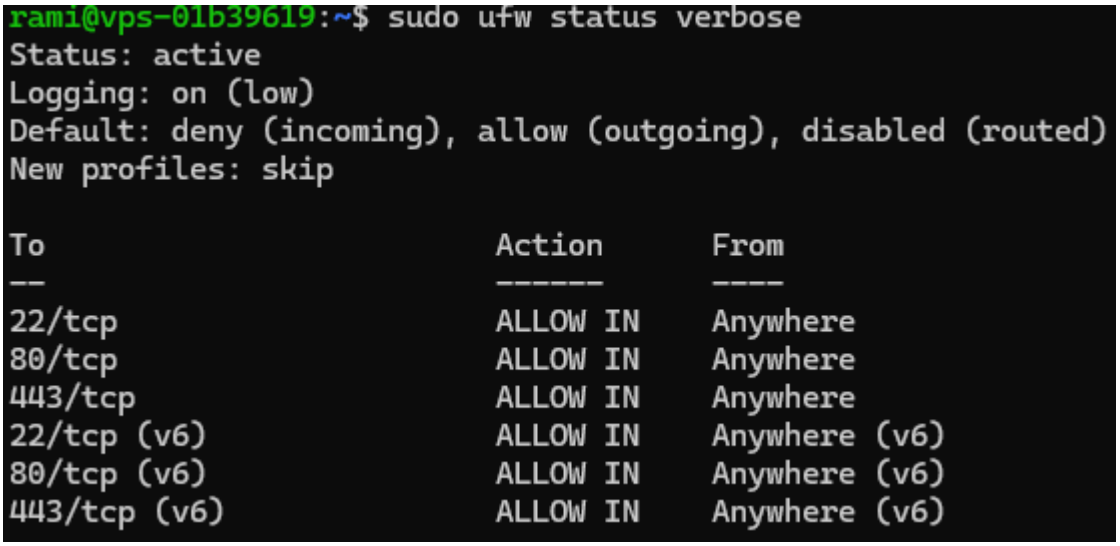
```
rami@vps-01b39619:~$ sudo ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
```

Vérification finale de l’état du pare-feu

Après activation, une dernière vérification a été effectuée afin de confirmer :

- que le pare-feu est bien actif,
- que la politique par défaut est restrictive,
- que seuls les services nécessaires sont accessibles.

sudo ufw status verbose
Le pare-feu est actif et les ports **22, 80 et 443** sont autorisés en entrée.



Conclusion – Pare-feu UFW

La mise en place du pare-feu UFW permet de renforcer efficacement la sécurité réseau du serveur en limitant les connexions entrantes aux seuls services nécessaires.

Cette configuration respecte les contraintes de l’environnement de production, sans interruption de service, tout en réduisant significativement la surface d’attaque du serveur.

Sécurisation du kernel Linux

L'objectif de cette étape est de renforcer la sécurité du noyau Linux du serveur afin de limiter certaines attaques réseau courantes (SYN flood, spoofing, redirections ICMP), tout en garantissant la stabilité de l'environnement de production.

Les actions réalisées reposent sur l'utilisation de paramètres **sysctl**, permettant d'ajuster le comportement du kernel **sans redémarrage du serveur**.

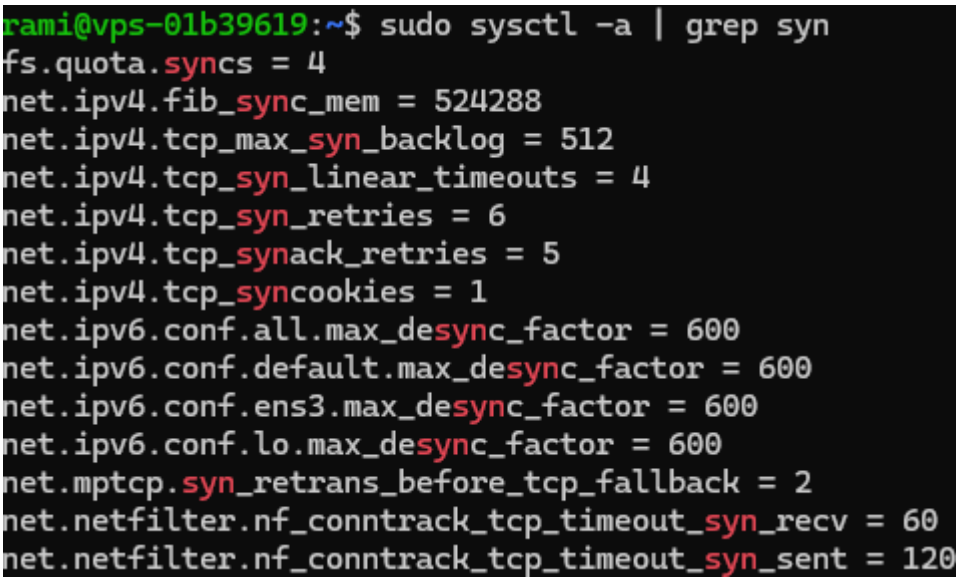
Vérification des paramètres sysctl existants

Avant toute modification, une vérification des paramètres de sécurité déjà actifs au niveau du kernel a été effectuée.

Commande utilisée :

```
sysctl -a | grep syn
sysctl -a | grep rp
```

Cette vérification permet d'identifier les protections déjà en place.
Il a été constaté que seules les protections basiques par défaut étaient actives.



```
rami@vps-01b39619:~$ sudo sysctl -a | grep rp
fs.binfmt_misc.python3/13 = interpreter /usr/bin/python3.13
net.core.rps_default_mask = 0
net.core.rps_sock_flow_entries = 0
net.ipv4.conf.all.arp_accept = 0
net.ipv4.conf.all.arp_announce = 0
net.ipv4.conf.all.arp_evict_nocarrier = 1
net.ipv4.conf.all.arp_filter = 0
net.ipv4.conf.all.arp_ignore = 0
net.ipv4.conf.all.arp_notify = 0
net.ipv4.conf.all.drop_gratuitous_arp = 0
net.ipv4.conf.all.proxy_arp = 0
net.ipv4.conf.all.proxy_arp_pvlan = 0
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.default.arp_accept = 0
net.ipv4.conf.default.arp_announce = 0
net.ipv4.conf.default.arp_evict_nocarrier = 1
net.ipv4.conf.default.arp_filter = 0
net.ipv4.conf.default.arp_ignore = 0
```

Protection contre les attaques de type SYN Flood

Afin de renforcer la protection contre les attaques par saturation de connexions TCP (SYN flood), l'activation des SYN cookies a été mise en place.

Commande utilisée :

```
sudo sysctl -w net.ipv4.tcp_syncookies=1
```

Cette protection permet au serveur de gérer plus efficacement les tentatives de connexions abusives sans impacter les connexions légitimes.

```
rami@vps-01b39619:~$ sudo sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1
```

Désactivation du source routing

Le source routing, fonctionnalité obsolète et potentiellement dangereuse, a été désactivé afin de réduire les risques d'usurpation de routage.

Commandes utilisées :

```
sudo sysctl -w net.ipv4.conf.all.accept_source_route=0
sudo sysctl -w net.ipv4.conf.default.accept_source_route=0
```

```
rami@vps-01b39619:~$ sudo sysctl -w net.ipv4.conf.all.accept_source_route=0
net.ipv4.conf.all.accept_source_route = 0
rami@vps-01b39619:~$ sudo sysctl -w net.ipv4.conf.default.accept_source_route=0
net.ipv4.conf.default.accept_source_route = 0
```

Désactivation des redirections ICMP

Les redirections ICMP ont été désactivées afin d'éviter toute modification non désirée de la table de routage du serveur.

Commandes utilisées :

sudo sysctl -w net.ipv4.conf.all.accept_redirects=0
sudo sysctl -w net.ipv4.conf.default.accept_redirects=0
sudo sysctl -w net.ipv4.conf.all.send_redirects=0

```
rami@vps-01b39619:~$ sudo sysctl -w net.ipv4.conf.all.accept_redirects=0
net.ipv4.conf.all.accept_redirects = 0
rami@vps-01b39619:~$ sudo sysctl -w net.ipv4.conf.default.accept_redirects=0
net.ipv4.conf.default.accept_redirects = 0
rami@vps-01b39619:~$ sudo sysctl -w net.ipv4.conf.all.send_redirects=0
net.ipv4.conf.all.send_redirects = 0
```

Activation du Reverse Path Filtering (mode sécurisé)

Le filtrage du chemin inverse (Reverse Path Filtering) a été activé en mode sécurisé afin de limiter les attaques par IP spoofing.

Commandes utilisées :

sudo sysctl -w net.ipv4.conf.all.rp_filter=1
sudo sysctl -w net.ipv4.conf.default.rp_filter=1

Le mode utilisé est compatible avec un environnement VPS standard et n'impacte pas le fonctionnement des services en production.

```
rami@vps-01b39619:~$ sudo sysctl -w net.ipv4.conf.all.rp_filter=1
net.ipv4.conf.all.rp_filter = 1
rami@vps-01b39619:~$ sudo sysctl -w net.ipv4.conf.default.rp_filter=1
net.ipv4.conf.default.rp_filter = 1
```

Réduction du bruit ICMP inutile

Afin de limiter le traitement de paquets ICMP inutiles, certaines réponses ICMP ont été désactivées.

Commandes utilisées :

sudo sysctl -w net.ipv4.icmp_echo_ignore_broadcasts=1
sudo sysctl -w net.ipv4.icmp_ignore_bogus_error_responses=1

```
rami@vps-01b39619:~$ sudo sysctl -w net.ipv4.icmp_echo_ignore_broadcasts=1
net.ipv4.icmp_echo_ignore_broadcasts = 1
rami@vps-01b39619:~$ sudo sysctl -w net.ipv4.icmp_ignore_bogus_error_responses=1
net.ipv4.icmp_ignore_bogus_error_responses = 1
```

Vérifications des changements :
sysctl net.ipv4.tcp_syncookies
sysctl net.ipv4.conf.all.accept_source_route
sysctl net.ipv4.conf.default.accept_redirects
sysctl net.ipv4.conf.all.rp_filter
sysctl net.ipv4.icmp_echo_ignore_broadcasts
sysctl net.ipv4.conf.all.log_martians

```
rami@vps-01b39619:~$ sysctl net.ipv4.tcp_syncookies
sysctl net.ipv4.conf.all.accept_source_route
sysctl net.ipv4.conf.default.accept_redirects
sysctl net.ipv4.conf.all.rp_filter
sysctl net.ipv4.icmp_echo_ignore_broadcasts
sysctl net.ipv4.conf.all.log_martians
net.ipv4.tcp_syncookies = 1
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv4.conf.all.rp_filter = 1
net.ipv4.icmp_echo_ignore_broadcasts = 1
net.ipv4.conf.all.log_martians = 1
```

Validation des paramètres à chaud

L'ensemble des paramètres a d'abord été **testé à chaud**, sans redémarrage du serveur, afin de vérifier leur impact sur l'environnement de production.

Aucun dysfonctionnement n'a été constaté :

- les sites web sont restés accessibles
- les services Nginx et SSH ont continué de fonctionner normalement

Mise en place d’une configuration sysctl permanente

Nous allons rendre ces paramètres **permanents**, afin qu’ils soient automatiquement appliqués au démarrage du serveur.

Création du fichier de configuration sysctl dédié :

sudo nano /etc/sysctl.d/99-minimal-hardening.conf

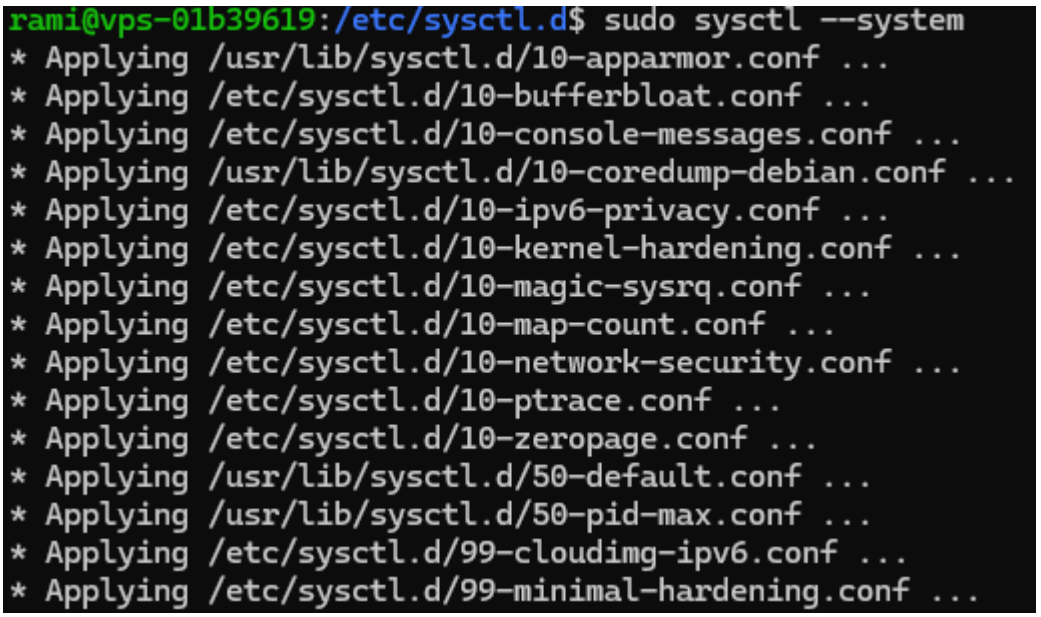
```
rami@vps-01b39619:/etc/sysctl.d$ sudo nano /etc/sysctl.d/99-minimal-hardening.conf
```

Valeurs ajoutées dans le fichier :

```
  GNU nano 8.3
net.ipv4.tcp_syncookies=1
net.ipv4.conf.all.accept_source_route=0
net.ipv4.conf.default.accept_source_route=0
net.ipv4.conf.all.accept_redirects=0
net.ipv4.conf.default.accept_redirects=0
net.ipv4.conf.all.send_redirects=0
net.ipv4.conf.all.rp_filter=1
net.ipv4.conf.default.rp_filter=1
net.ipv4.icmp_echo_ignore_broadcasts=1
net.ipv4.icmp_ignore_bogus_error_responses=1
```

Application des paramètres sans redémarrage :

sudo sysctl --system



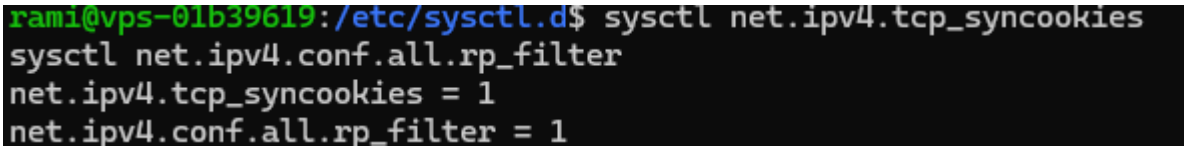
Vérification finale des paramètres

Une vérification finale a été effectuée afin de confirmer que les paramètres sont bien actifs et persistants.

Commandes utilisées :

sysctl net.ipv4.tcp_syncookies
sysctl net.ipv4.conf.all.rp_filter

La valeur = 1 confirme que les protections sont correctement activées.



Conclusion – Sécurisation du kernel

La sécurisation du kernel a permis de renforcer la protection réseau du serveur sans interruption de service.
Les paramètres appliqués sont standards, éprouvés et compatibles avec un environnement de production hébergeant plusieurs sites web.

Cette configuration améliore la résistance du serveur face aux attaques réseau tout en respectant les contraintes de stabilité et de disponibilité de l'infrastructure.