# Un peu de sécurité ne fait pas de mal

**Table of Contents**



Règles de base pour sécuriser : mise en place d'un firewall avec **IPtables** et de règles de filtrage. ajout de **fail2ban** et **rkhunter**.

Ces étapes sont à étudier avec soin. Elles ne constituent pas de parade ultime, mais sont un premier pas nécessaire. Vérifier bien les choix avant toute mise en place pour éviter de s'exclure soit-même par une règle trop restrictive (il est recommandé de le faire en phase de tests lorsqu'on peut encore réinitialiser totalement le serveur).

La première précaution consiste avant tout à se tenir informé : il existe des mailing-lists spécialisées dans la sécurité, telles que Debian Security Announce et à procéder à des mises à jour régulières (via apt-get upgrade par exemple).

### Modifier le mot de passe root

Il est conseillé de modifier le mot de passe surtout si celui-ci a été attribué par défaut. S'identifier d'abord en root (voir ci-dessus) puis entrer la commande :

```
passwd root
```

### Configuration SSH

Afin de sécuriser l'accès SSH au serveur, éditer le fichier `/etc/ssh/sshd_config`. Changer le port de connexion par défaut pour éviter quelques attaques par bruteforce sur le port 22, qui est bien connu pour héberger ce service. Il faudra préciser ce nouveau port (dans Putty ou en ligne de commande ssh sous Linux) à la prochaine connexion.

```
vi /etc/ssh/sshd_config
```

```
Port 2222                  # Changer le port par défaut (à retenir!)
PermitRootLogin no         # Ne pas permettre de login en root
AllowUsers dew             # N'autoriser qu'un utilisateur précis
```

Redémarrer le service SSH après ces modifications :

```
/etc/init.d/ssh restart
```

### Alerte login Root

On peut éditer le fichier `/root/.bashrc` qui est exécuté au démarrage d'une sesion root pour envoyer un e-mail de notification. De cette façon, on sera prévenu lorsqu'un login est effectué.

```
vi /root/.bashrc
```

Ajouter la ligne (en modifiant l'adresse e-mail de destination) :

```
echo 'Accès Shell Root le ' `date` `who` | mail -s `hostname` Shell Root de
`who | cut -d"(" -f2 | cut -d")" -f1` monitoring@test.com
```

On peut faire un peu de personnalisation esthétique avec ces lignes :

```
alias ls='ls $LS_OPTIONS --color=auto'
alias ll='ls $LS_OPTIONS -al --color=auto'
alias vi='vim'
```

### Services inutiles

Si on n'utilise pas les services **portmap**, **nfs** et **inetd** (dans le cas d'un serveur web on n'en a pas besoin). Il en existe d'autres, selon la distribution et les choix d'installation originaux. Cela économisera aussi de la mémoire vive.

```
/etc/init.d/portmap stop

/etc/init.d/nfs-common stop

update-rc.d -f portmap remove

update-rc.d -f nfs-common remove

update-rc.d -f inetd remove

apt-get remove portmap

apt-get remove ppp
```

### Autorisations diverses

N'autoriser les compilateurs et installeurs que pour root (le numéro de version est à adapter selon la fraîcheur de l'installation) :

```
chmod o-x /usr/bin/gcc-4.1

chmod o-x /usr/bin/make

chmod o-x /usr/bin/apt-get

chmod o-x /usr/bin/aptitude

chmod o-x /usr/bin/dpkg
```

### IPtables / Netfilter

**IPtables** (associé à **Netfilter**) est un des meilleurs firewalls pour Linux, et certainement le plus répandu. On peut trouver de nombreux scripts de configuration à son sujet. En voici un, à adapter à votre configuration. A tout instant, utiliser la commande `iptables -L -v` pour lister les règles en place.

Celles-ci portent sur 3 chaînes : **INPUT** (en entrée), **FORWARD** (dans le cas d'un routage réseau) et **OUPUT** (en sortie). Les actions à entreprendre sont **ACCEPT** (accepter le paquet), **DROP** (le jeter), **QUEUE** et **RETURN**.

Arguments utilisés :

- **i** : interface d'entrée (input)
- **i** : interface de sortie (output)
- **t** : table (par défaut filter contenant les chaînes INPUT, FORWARD, OUTPUT)
- **j** : règle à appliquer (Jump)
- **A** : ajoute la règle à la fin de la chaîne (Append)
- **I** : insère la règle au début de la chaîne (Insert)
- **R** : remplace une règle dans la chaîne (Replace)
- **D** : efface une règle (Delete)
- **F** : efface toutes les règles (Flush)
- **X** : efface la chaîne
- **P** : règle par défaut (Policy)
- **lo** : localhost (ou 127.0.0.1, machine locale)

Créer un script qui sera lancé à chaque démarrage pour mettre en place des règles de base.

```
vi /etc/init.d/firewall
```

```
#!/bin/sh
```

```
# Vider les tables actuelles
iptables -t filter -F

# Vider les règles personnelles
iptables -t filter -X

# Interdire toute connexion entrante et sortante
iptables -t filter -P INPUT DROP
iptables -t filter -P FORWARD DROP
iptables -t filter -P OUTPUT DROP

# ---

# Ne pas casser les connexions etablies
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

# Autoriser loopback
iptables -t filter -A INPUT -i lo -j ACCEPT
iptables -t filter -A OUTPUT -o lo -j ACCEPT

# ICMP (Ping)
iptables -t filter -A INPUT -p icmp -j ACCEPT
iptables -t filter -A OUTPUT -p icmp -j ACCEPT

# ---

# SSH In
iptables -t filter -A INPUT -p tcp --dport 2222 -j ACCEPT

# SSH Out
iptables -t filter -A OUTPUT -p tcp --dport 2222 -j ACCEPT

# DNS In/Out
iptables -t filter -A OUTPUT -p tcp --dport 53 -j ACCEPT
iptables -t filter -A OUTPUT -p udp --dport 53 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 53 -j ACCEPT
iptables -t filter -A INPUT -p udp --dport 53 -j ACCEPT

# NTP Out
iptables -t filter -A OUTPUT -p udp --dport 123 -j ACCEPT
```

Si on héberge un sevreur web (Apache) :

```
# HTTP + HTTPS Out
iptables -t filter -A OUTPUT -p tcp --dport 80 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 443 -j ACCEPT

# HTTP + HTTPS In
iptables -t filter -A INPUT -p tcp --dport 80 -j ACCEPT
```

```
iptables -t filter -A INPUT -p tcp --dport 443 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 8443 -j ACCEPT
```

Si on héberge un serveur FTP :

```
# FTP Out
iptables -t filter -A OUTPUT -p tcp --dport 20:21 -j ACCEPT

# FTP In
modprobe ip_conntrack_ftp # ligne facultative avec les serveurs OVH
iptables -t filter -A INPUT -p tcp --dport 20:21 -j ACCEPT
iptables -t filter -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Si on héberge un serveur de mail avec SMTP, POP3 et IMAP :

```
# Mail SMTP:25
iptables -t filter -A INPUT -p tcp --dport 25 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 25 -j ACCEPT

# Mail POP3:110
iptables -t filter -A INPUT -p tcp --dport 110 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 110 -j ACCEPT

# Mail IMAP:143
iptables -t filter -A INPUT -p tcp --dport 143 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 143 -j ACCEPT

# Mail POP3S:995
iptables -t filter -A INPUT -p tcp --dport 995 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 995 -j ACCEPT
```

Si on utilise l'outil de monitoring Monit sur le port 1337 (à modifier selon votre configuration) autoriser cette connexion :

```
# Monit
iptables -t filter -A INPUT -p tcp --dport 1337 -j ACCEPT
```

Si on utilise un serveur RPS d'OVH, le disque iSCSI nécessite un accès réseau qui rend obligatoire une règle supplémentaire au début des filtres. Sans cela, votre serveur deviendra inutilisable :

```
iptables -A OUTPUT -p tcp --dport 3260 -m state --state NEW,ESTABLISHED -j
ACCEPT
```

Lorsqu'on a défini toutes les règles, render ce fichier exécutable :

```
chmod +x /etc/init.d/firewall
```

On peut le tester en l'exécutant directement en ligne de commande. Il faut s'assurer d'avoir toujours le contrôle de votre machine (se reconnecter en SSH, vérifier la disponibilité des services web, ftp, mail...). En cas d'erreur, redémarrer le serveur, les règles seront oubliées et cela permettra de reprendre la main. En revanche, si les tests s'avèrent concluants, ajouter le script au démarrage pour que celui-ci protège le serveur dès le boot.

Afin de l'ajouter aux scripts appelés au démarrage :

```
update-rc.d firewall defaults
```

Pour le retirer, on peut utiliser la commande suivante :

```
update-rc.d -f firewall remove
```

Redémarrer, ou exécuter `/etc/init.d/firewall` pour activer le filtrage.

Avant d'activer la configuration au boot, il faut absolument tester les règles qui doivent être adaptées à la configuration réseau. Un mauvais choix peut entraîner une indisponibilité du serveur ou une perte de contrôle sur celui-ci avec le blocage de la connexion SSH.

On peut utiliser **IPtables** sans passer par un script de démarrage et entrer directement les instructions en mode console. Pour bannir temporairement une adresse IP en cas de nécessité, utiliser la commande `iptables -A INPUT -s adresse_ip -j DROP`

### Fail2ban

Fail2ban est un script surveillant les accès réseau grâce aux logs des serveurs. Lorsqu'il détecte des erreurs d'authentification répétées, il prend des contre-mesures en bannissant l'adresse IP grâce à iptables. Cela permet d'éviter nombre d'attaques bruteforce et/ou par dictionnaire.

#### Installation

```
apt-get install fail2ban
```

#### Configuration

```
vi /etc/fail2ban/fail2ban.conf
```

```
loglevel                        # Niveau de détail des logs (défaut 3)
logtarget = /var/log/fail2ban.log # Chemin vers le fichier de log
```

```
(description des actions entreprises par fail2ban)
```

Les services à monitorer sont stockés dans jail.conf. Il est recommandé d'en effectuer une copie nommée jail.local qui sera automatiquement utilisée à la place du fichier exemple.

```
cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
```

```
vi /etc/fail2ban/jail.local
``

Quelques paramètres globaux :
```

ignoreip = 127.0.0.1 # Liste des adresses IP de confiance à ignorer par fail2ban bantime = 600 # Temps de ban en secondes maxretry = 3 # Nombre d'essais autorisés pour une connexion avant d'être banni destmail monitoring@test.com # Adresse e-mail destinataire des notifications action # Action à entreprendre en cas de détection positive (voir dans /etc/fail2ban/action.d/)

```
Chaque section possède ses propres paramètres qui prennent le pas sur les globaux s'ils sont mentionnés :
```

enabled # Monitoring activé (true) ou non (false) maxretry, bantime, ignoreip, destmail # Voir ci-dessus port # Port IP concerné logpath # Fichier de log à analyser pour détecter des anomalies filter # Filtre utilisé pour l'analyser du log

```
Les filtres par défaut sont stockés dans `/etc/fail2ban/filter.d`. Ils contiennent en général une instruction failregex suivie d'une expression régulière matchant la détection d'une authentification erronée. Par exemple pour le service Courier :
```

failregex = LOGIN FAILED, ip=[<HOST>]$

```
<WRAP info>Celle-ci peut être précisée directement dans jail.local à la section appropriée pour prendre le pas sur la directive filter.</WRAP>

Modifier les ports le cas échéant dans la section ssh si on a suivi la recommandation ci-dessus...
```

enabled = true port = 2222

```
Après modification de la configuration, redémarrer fail2ban : `/etc/init.d/fail2ban restart`
```

```
####Rootkit Hunter

**Rootkit Hunter** est un programme de détection de rootkits. On peut
l'installer grâce à :
```

apt-get install rkhunter

```


Il procédera à des détections journalières anti-rootkits et enverra des
notifications par e-mail si nécessaire. Il est conseillé de l'installer très
tôt car il calcule l'empreinte MD5 des programmes installés afin de détecter
d'éventuels changements. Editer `/etc/default/rkhunter` pour indiquer
l'adresse de notification et l'exécution journalière :
```

vi /etc/default/rkhunter

```



REPORT_EMAIL="monitoring@test.com"
CRON_DAILY_RUN="yes"
```

En cas de fausses détections positives sur des répertoires ou fichiers existants et sains, éditer `/etc/rkhunter.conf` pour les ajouter à la liste des éléments autorisés.

```
vi /etc/rkhunter.conf
```

```
ALLOWHIDDENDIR=/dev/.udev
ALLOWHIDDENDIR=/dev/.static
```

On peut également utiliser **chkrootkit** qui est un équivalent.