

Iptables : clients et serveurs

Sommaire

I. Rappels.....	1
a) Les différents types de filtrages : les tables.....	1
b) Fonctionnement de base : les chaînes et les règles.....	2
II. La table filter : utilisation en client ou serveur.....	2
III. Action communes aux clients et serveurs.....	3
a) Vidage des chaînes utilisées.....	3
b) Mise à zéro des compteurs de paquets.....	3
c) Positionnement de la politique par défaut.....	3
d) Autorisation de la boucle locale.....	4
IV. Cas des connexions clients.....	4
a) Autoriser les programmes à sortir vers l'extérieur.....	4
b) Mais aussi autoriser les réponses (plus rapide, moins sûr).....	4
c) Mais aussi autoriser les réponses (plus sûr, moins rapide).....	5
d) Autorisation de l'ICMP et des pings.....	5
V. Cas du serveur.....	5
a) Autoriser les nouvelles connexions.....	5
b) Mais aussi laisser sortir les réponses aux connexions clients.....	5
VI. Sauvegarde des règles du pare-feu.....	6
a) Première méthode (pour le backup).....	6
b) Seconde méthode (la meilleure).....	6
VII. Exemple de configuration d'un client.....	6
a) Minimum vital pour l'utilisation en local et pour débiter une configuration de pare-feu.....	6
b) Une protection de base.....	7
c) Un pare-feu un peu plus sûr.....	7
VIII. Exemple de configuration d'un serveur.....	7
a) Minimum vital pour l'utilisation en local et pour débiter une configuration de pare-feu.....	8
b) Une configuration de base.....	8
IX. Cas particulier du FTP.....	9
a) Côté client.....	10
b) Côté serveur.....	10
X. Log des paquets rejetés.....	11
a) Première méthode (un peu encombrante).....	11
b) Seconde méthode (de loin la meilleure).....	12
I. Bibliographie.....	12

I. Rappels

a) Les différents types de filtrages : les tables

Une table contient des chaînes relatives au filtrage qu'elle réalise.

Il existe 3 tables distinctes :

- `filter` (Tables de filtrage) : c'est la table par défaut. Elle sert pour les entrées/sorties/traversées sur la machine.
- `nat` (translation d'adresse) : elle sert pour le gérer le changement d'adresses IPs et de ports
- `mangle` (table spécifique) : elle sert pour gérer

Dans la table de filtrage (FILTER) on peut filtrer les paquets avec trois chaînes :

- Paquets entrants (INPUT) (vers des applications) (client ou server)
- Paquets sortants (OUTPUT) (émis par des applications) (client ou server)
- Paquet passant par le firewall (FORWARD) (passerelle)

Dans la table de translation (NAT) on peut gérer les paquets (dans le cas d'une passerelle) :

- Entrant dans le firewall (PREROUTING) : DNAT
- Sortant du firewall (POSTROUTING) : SNAT

b) Fonctionnement de base : les chaînes et les règles

Iptables est basé sur des chaînes de pare-feu (ou simplement chaîne). Une chaîne est un ensemble ordonné (une liste) de règles. Une règle indique quoi faire d'un paquet quand il a certaines caractéristiques.

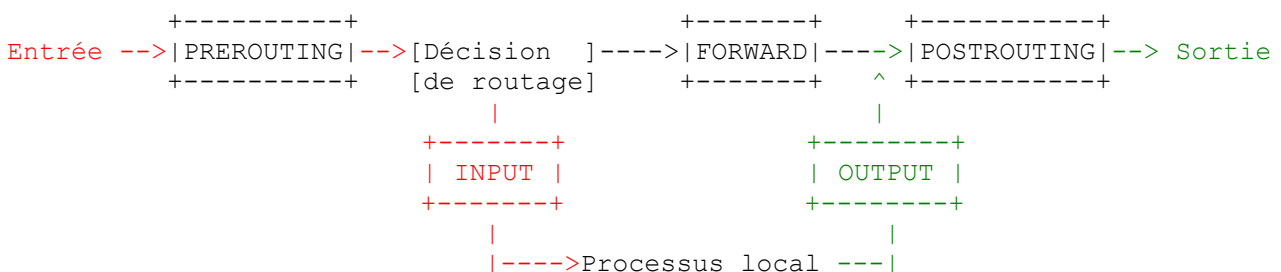
Ainsi, un paquet traverse les règles d'une chaîne jusqu'à ce qu'il corresponde à une règle. Dans ce cas, la règle indique si le paquet est transmis à sa destination (ACCEPT) ou supprimer (DROP).

Dès qu'une règle (autre que LOG) capte un paquet, elle prend une action sur le paquet et la parcourt de la chaîne s'arrête.

Si le paquet ne correspond à aucune règle alors on applique la police par défaut de la chaîne. Si elle est à ACCEPT, tout paquet non interdit sera délivré à sa destination. Si elle est à DROP, tout paquet non autorisé sera supprimé (Cette solution est la plus sûr).

II. La table filter : utilisation en client ou serveur

Le fonctionnement de la table FILTER (et NAT) dans le cas d'un serveur/client est le suivant :



Le parcourt d'un paquet est indiqué en rouge pour les paquets entrants et en vert pour les paquets sortant.

La décision de routage se fait en regardant si l'adresse IP de destination est celle de l'interface par laquelle le paquet est entré :

- si l'adresse de destination est celle d'une entrée de la machine, le paquet est pour la machine (serveur ou client) : le paquet passe dans la chaîne INPUT
- si le paquet vient d'une application : le paquet créé passe dans la chaîne OUTPUT
- sinon le paquet ne fait que passer par la machine (passerelle) (adresse destination différente de l'interface entrante d'où vient le paquet): chaîne FORWARD. Si le forwarding n'est pas autorisé alors le paquet est détruit.

Dans le cas d'un simple serveur (ou client), on n'utilisera que les chaînes INPUT et OUTPUT. Les chaînes PREROUTING et POSTROUTING ne devraient pas être utilisées par un serveur.

III. Action communes aux clients et serveurs

a) Vidage des chaînes utilisées

On commence par vider les chaînes pour être sûr de partir de zéro :

```
[root@server ~]# iptables -F INPUT
[root@server ~]# iptables -F OUTPUT
[root@server ~]# iptables -F FORWARD
[root@server ~]#
```

b) Mise à zéro des compteurs de paquets

On remet à zéro les compteurs (nombre de paquets et nombre d'octet traités) :

```
[root@server ~]# iptables -Z INPUT
[root@server ~]# iptables -Z OUTPUT
[root@server ~]# iptables -Z FORWARD
[root@server ~]#
```

On vérifie qu'il n'y a rien comme filtrage :

```
[root@server ~]# iptables -L
```

```
Chain INPUT (policy ACCEPT)
target      prot opt source                destination
```

```
Chain FORWARD (policy ACCEPT)
target      prot opt source                destination
```

```
Chain OUTPUT (policy ACCEPT)
target      prot opt source                destination
```

```
[root@server ~]#
```

c) Positionnement de la politique par défaut

Il est toujours préférable de tout interdire sauf ce que l'on autorise :

```
[root@server ~]# iptables -P INPUT DROP
[root@server ~]# iptables -P OUTPUT DROP
[root@server ~]# iptables -P FORWARD DROP
```

Attention : vous ne pouvez plus sortir de la machine, donc pas d'Internet et risque de perte de connexion SSH.

d) Autorisation de la boucle locale

Certaines applications ont besoin de la boucle locale pour fonctionner correctement. Aussi est-il nécessaire de tout autoriser sur cet interface (lo) et pour l'instant uniquement sur cet interface :

```
[root@server ~]# iptables -A INPUT -i lo -j ACCEPT
[root@server ~]# iptables -A OUTPUT -o lo -j ACCEPT
```

Remarque :

-i : interface d'entrée (valide uniquement pour INPUT, FORWARD, PREROUTING)

-o : interface de sortie (valide uniquement pour FORWARD, OUTPUT, POSTROUTING)

IV. Cas des connexions clients

a) Autoriser les programmes à sortir vers l'extérieur

Pour autoriser les programmes à sortir sur Internet, par exemple, on doit autoriser les paquets en sortie à destination d'un port connu. Par exemple, pour autoriser l'affichage de pages web, on fera :

```
[root@server ~]# iptables -A OUTPUT --protocol tcp --destination-
port 80 -j ACCEPT
```

En effet le web fonctionne bien en tcp sur le port 80. Si on passe par un proxy il faudra ajouter la même règle avec le port du proxy.

b) Mais aussi autoriser les réponses (plus rapide, moins sûr)

Le problème est qu'il faut aussi accepter les réponses !! Pour cela nous avons besoin d'une extension d'iptables : **state**, qui permet de préciser s'il s'agit d'une nouvelle connexion : NEW, d'une réponse : ESTABLISHED, ou d'une connexion en rapport avec une autre déjà établie : RELATED. Pour cela on spécifie `-m state` en option pour être sur que le module sera chargé. Ce qui donne :

```
[root@server ~]# iptables -A INPUT -m state --state
ESTABLISHED,RELATED -j ACCEPT
```

Ceci acceptera les réponses à TOUTES les requêtes et pas seulement celles de serveurs web, ce n'est pas optimal mais bon c'est déjà bien.

c) Mais aussi autoriser les réponses (plus sûr, moins rapide)

Si l'on veut autoriser uniquement les réponses des serveurs web, on fera :

```
[root@server ~]# iptables -A INPUT -p tcp -sport 80 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

ATTENTION : dans ce cas, il faut taper cette commande pour chaque service distant.

d) Autorisation de l'ICMP et des pings

Si l'on veut avoir écho des erreurs de transmission de paquets, il peut être utile d'autoriser l'ICMP

```
[root@server ~]# iptables -A INPUT -p icmp --icmp-type any -j ACCEPT
```

Remarque :

-p : raccourci vers --protocol

--icmp-type : type de paquets icmp, plus d'info avec iptables -p icmp -h et premier tuto

V. Cas du serveur

a) Autoriser les nouvelles connexions

Pour autoriser les clients à se connecter à un service web local, il faut autoriser les nouvelles connexions sur le port du service en question. Par exemple, pour autoriser les clients à se connecter sur notre serveur ssh :

```
[root@server ~]# iptables -A INPUT -m state --state NEW -p tcp --dport 22 -j ACCEPT
```

b) Mais aussi laisser sortir les réponses aux connexions clients

Il faut aussi laisser sortir les paquets de réponses :

```
[root@server ~]# iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
```

Ceci semble être assez sûr car si une application serveur veut sortir, c'est qu'elle a une demande client.

On pourrait bien sûr filtrer plus précisément les connexions sortantes :

```
[root@server ~]# iptables -A OUTPUT -p tcp -sport 22 -m state --state RELATED,ESTABLISHED -j ACCEPT
```

VI. Sauvegarde des règles du pare-feu

Maintenant il faudrait pouvoir sauvegarder tout ça !!!

a) Première méthode (pour le backup)

L'enregistrement des règles se fait avec la commande `iptables-save` et restaurer les règles enregistrées avec `iptables-restore`. Ces deux programmes affichent les règles sur `stdout`. Il faut donc rediriger le tout vers un fichier. Ensuite il faut se faire un script de démarrage qui charge les règles au boot de la machine.

b) Seconde méthode (la meilleure)

Toutefois il y a plus simple. Il existe un fichier de configuration `/etc/sysconfig/iptables` dans lequel on peut sauvegarder la configuration du firewall. Le format est le même que celui utilisé par `iptables-save`.

Pour faire cela on utilise la commande :

```
[root@server ~]# service iptables save
```

ou encore

```
[root@server ~]# /etc/init.d/iptables save
```

Maintenant nos règles seront actives au démarrage, il existe également des options pour le service `iptables` qui se trouvent dans le fichier `/etc/sysconfig/iptables-config`.

VII. Exemple de configuration d'un client

Pour un client, le principal est de ne pas laisser entrer des attaquants sur des services qui tournent en local, comme un `ftp`, un `samba`, un `smtp/pop` ou un `ssh`. En principe, un client peut sortir sur tous les ports sans restriction et n'accepte que les paquets entrant qui appartiennent à une connexion existante.

a) Minimum vital pour l'utilisation en local et pour débiter une configuration de pare-feu

```
#on vide les chaînes de la table filter (par défaut)
iptables -F INPUT
iptables -F OUTPUT
iptables -F FORWARD
```

```
#on remet à zéro les compteurs de paquets
```

```
iptables -Z FORWARD
iptables -Z OUTPUT
iptables -Z INPUT
```

#on interdit tous les paquets qui ne correspondent pas à une règle
#ceci est la police la plus sûre.

```
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
```

#on accepte tout le trafic en boucle locale (loopback)
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

b) Une protection de base

#on accepte le ICMP entrant et sortant (peut être plus restrictif)
iptables -A INPUT -p icmp --icmp-type any -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type any -j ACCEPT

on accepte le trafic entrant des connexions déjà établies (les
réponses des serveurs)
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

#on autorise toutes les sorties
iptables -A OUTPUT -j ACCEPT

c) Un pare-feu un peu plus sûr

autorise la connexion à des serveurs distants sur des ports
connus (note : cela n'autorise pas IRC par exemple)
iptables -A OUTPUT -p tcp --dport 1:1023 -j ACCEPT

#on pourrait aussi restreindre à une liste de ports connus sur le
serveur distant

#autorise la résolution DNS (requête et réponse)
iptables -A OUTPUT -p udp --dport 53 -j ACCEPT
iptables -A INPUT -p udp --sport 53 -j ACCEPT

#autorise le mode PORT
iptables -A INPUT -m state --state NEW,RELATED,ESTABLISHED -p tcp
--sport 20 -j ACCEPT

VIII. Exemple de configuration d'un serveur

Pour un serveur, le principal est de ne laisser entrer les utilisateurs que pour les services que l'on veut.

En principe, un serveur ne peut sortir que sur les ports de ses services (à l'exception du passive ftp) et n'accepte que les paquets sortants qu'en rapport avec une connexion entrante acceptée.

a) Minimum vital pour l'utilisation en local et pour débiter une configuration de pare-feu

```
#on vide les chaînes de la table filter (par défaut)
iptables -F INPUT
iptables -F OUTPUT
iptables -F FORWARD

#on remet à zéro les compteurs de paquets
iptables -Z FORWARD
iptables -Z OUTPUT
iptables -Z INPUT

#on interdit tous les paquets qui ne correspondent pas à une règle
#ceci est la police la plus sûre.
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

#on accepte tout le trafic en boucle locale (loopback)
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
```

b) Un configuration de base

```
# autorise les connexions FTP entrantes
iptables -A INPUT -p tcp --dport 21 -j ACCEPT
# mode PORT de ftp
iptables -A OUTPUT -m state --state NEW -p tcp --sport 20 -j
ACCEPT
# mode PASSIF (entre le port 60000 et65000) de ftp
iptables -A INPUT -p tcp --dport 60000:65000 -j ACCEPT

#autorise les connexions SSH
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
#autorise les connexions SMTP
iptables -A INPUT -p tcp --dport 25 -j ACCEPT
#autorise les connexions HTTP
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
#autorise les connexions POP3
iptables -A INPUT -p tcp --dport 110 -j ACCEPT
#autorise les connexions NTP
iptables -A INPUT -p tcp --dport 123 -j ACCEPT
#autorise les connexions IMAP
iptables -A INPUT -p tcp --dport 143 -j ACCEPT
```

```

#autorise les connexions HTTPS
iptables -A INPUT -p tcp --dport 443 -j ACCEPT
#autorise les connexions IMAPS
iptables -A INPUT -p tcp --dport 993 -j ACCEPT
#autorise les connexions POP3S
iptables -A INPUT -p tcp --dport 995 -j ACCEPT

#autorise les connexions SAMBA
#(depuis le réseau local @reseau_local)
#en effet, ce n'est pas prudent de l'exposé à l'extérieur
iptables -A INPUT -s @reseau_local -p tcp --dport 445 -j ACCEPT
iptables -A INPUT -s @reseau_local -p tcp --dport 137 -j ACCEPT
iptables -A INPUT -s @reseau_local -p tcp --dport 138 -j ACCEPT
iptables -A INPUT -s @reseau_local -p tcp --dport 139 -j ACCEPT

#autorise les connexions DNS (requête et recopie primaire)
iptables -A INPUT -p udp --dport 53 -j ACCEPT
iptables -A INPUT -p tcp --dport 53 -j ACCEPT

#log le reste : les paquets non valides ou non autorisés
iptables -A INPUT -j LOG

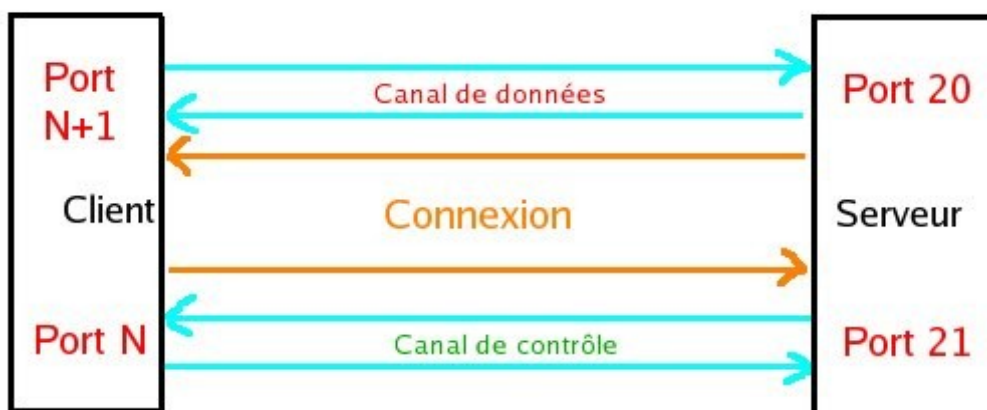
#on enregistre notre firewall
service iptables save

```

IX. Cas particulier du FTP

Le protocole FTP fonctionne dans l'un des deux modes suivants :

- le mode actif (port) : le client se connecte depuis un port N sur le port 21 du serveur pour le canal de contrôle (envoi des commandes) et le serveur se connecte depuis son port 20 vers le port N+1 du client.



- le mode passif (passive) : le client se connecte sur le port 21 du serveur pour le canal de contrôle (envoi des commandes) et sur un autre port (que le serveur choisit) du serveur pour le transfert des données.



a) Côté client

Il faut autoriser la connexion sur le port 21 distant :

```
iptables -A OUTPUT -m state --state NEW -p tcp --dport 21 -j
ACCEPT
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -p tcp --
sport 21 -j ACCEPT
```

Pour autoriser le mode actif sur le serveur :

```
iptables -A OUTPUT -m state --state NEW -p tcp --sport 20 -j
ACCEPT
iptables -A INPUT -m state --state RELATED,ESTABLISHED -p tcp --
dport 20 -j ACCEPT
```

Pour autoriser le mode passif sur n'importe quel port du serveur :

```
iptables -A INPUT -m state --state RELATED -p tcp -j ACCEPT
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -p tcp -j
ACCEPT
```

Les serveurs FTPs proposent, en général, un moyen de limiter la plage des ports utilisés pour le mode passif. Dans ce cas, le pare-feu est plus sécurisé (par exemple, entre 60000 et 65000) :

```
iptables -A INPUT -m state --state NEW -p tcp --dport 60000:65000
-j ACCEPT
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -p tcp -j
ACCEPT
```

b) Côté serveur

Il faut autoriser la connexion sur le port 21 :

```
iptables -A INPUT -m state --state NEW -p tcp --dport 21 -j ACCEPT
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -p tcp --
sport 21 -j ACCEPT
```

Pour autoriser le mode actif sur le serveur :

```
iptables -A OUTPUT -m state --state NEW -p tcp --sport 20 -j
ACCEPT
iptables -A INPUT -m state --state RELATED,ESTABLISHED -p tcp --
dport 20 -j ACCEPT
```

Pour autoriser le mode passif sur n'importe quel port du serveur :

```
iptables -A INPUT -m state --state RELATED -p tcp -j ACCEPT
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -p tcp -j
ACCEPT
```

Les serveurs FTPs proposent, en général, un moyen de limiter la plage des ports utilisés pour le mode passif. Dans ce cas, le pare-feu est plus sécurisé (par exemple, entre 60000 et 65000) :

```
iptables -A INPUT -m state --state NEW -p tcp --dport 60000:65000
-j ACCEPT
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -p tcp -j
ACCEPT
```

X. Log des paquets rejetés

Il peut être intéressant aussi de loguer les tentatives qui sont refusées.

Pour cela, il y a deux solutions.

a) Première méthode (un peu encombrante)

On peut utiliser la cible LOG

```
iptables -A chaine -j LOG --log-level info
```

Dans /etc/syslog.conf :

```
kern.=info -/var/log/iptables
```

Les options que l'on peut ajouter après le -j LOG, sont les suivantes :

<i>Options</i>	<i>Descriptions</i>
--log-level <i>niveau</i>	Indique le niveau auquel les logs sont faits dans syslog dans la catégorie noyau. Par défaut, il s'agit du niveau warn.
--log-prefix <i>préfixe</i>	Indique un préfixe à ajouter à chaque ligne de log pour faciliter le parsing du fichier, par exemple, avec grep. Maximum 29 caractères

ATTENTION : la cible LOG met toujours ses logs dans syslog dans la catégorie KERN (noyau). En outre, par défaut, elle log au niveau warn. Ceci a pour effet de polluer très massivement les logs et les consoles.

ATTENTION : même avec la configuration précédente de syslog, il se peut que le noyau émette des logs au niveau info et donc dans les logs de syslog. Cette méthode est donc à utiliser

UNIQUEMENT POUR DES TESTS.

b) Seconde méthode (de loin la meilleure)

On peut aussi utiliser ULOG :

- Installer ULOG (yum install ulogd)
- démarrer ulogd (service ulogd start et chkconfig ulogd on)
- configurer ulogd :

```
[LOGEMU]
```

```
#indique le fichier dans lequel on met les logs d'iptables
```

```
file="/var/log/ulogd/ulogd.syslogemu"
```

```
sync=1
```

- remplacer `iptables -A chaine -j LOG -log-level info` par `iptables -A chaine -j ULOG`

Note : ULOG est à réserver pour une journalisation intensive et continue et pas pour des tests. Il permet par exemple de stocker les logs dans une base MySQL.

I. Bibliographie

[netfilter/iptables project homepage - The netfilter.org project](#)

[freshmeat.net: Project details for ulogd](#)