Réseau et administration système

boré uk

13 décembre 2014

1 Administration réseau : iptables

Sous GNU/Linux, une seule commande ici nous interesse : *iptables*.
Cette commande permet d'effectuer **3 types de traitements** :

- **Filtrer**, c'est a dire déterminer quels paquets seront acceptés ou pas *(filter)*
- **Translater des adresses** : une seule adresse ip publique pour plein d'adresses ip privées *(nat)*
- **Modifier les entetes de paquets**, mais ici on s'en fout *(mangle)*

La partie la plus importante du côté réseau concerne le **filtre**. Ainsi, on va commencer par ça.

1.1 Filtrer les paquets

```
iptable -t filter [option iptable] [chaine] [option de filtrage] --jump [action]
```

Pour filtrer les paquets, on va agir sur la table *filter*. Ainsi, toutes les commandes *iptable* commenceront de la manière suivante :

```
iptable -t filter
```

Jusque là, rien de compliqué. De plus, dans la plupart des cas, on va chercher à **-A**jouter des règles par rapport à celles existantes. Du coup, on peut même dire que la plupart des commandes *iptable* commenceront de cette façon :

```
iptable -t filter -A
```

Pour votre gouverne, sachez qu'*iptables* accepte d'autres options : on peut **supprimer** une règle (*-D*), **tout supprimer** (*-F*), ...

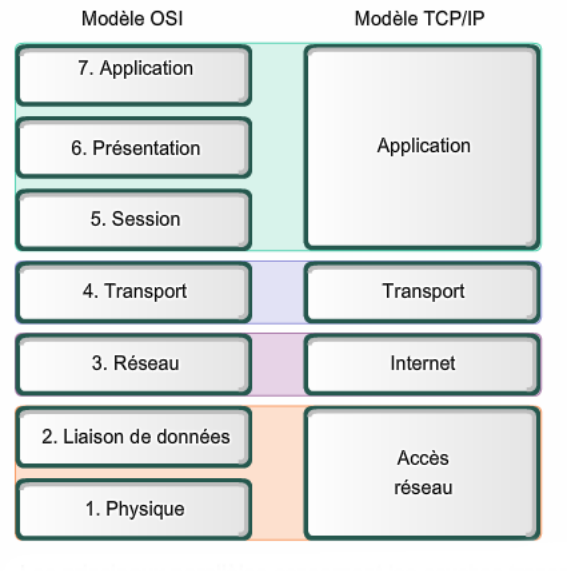
Maintenant, il faut voir qu'est-ce qu'une chaine et quels sont les chaines disponibles.
Une chaine définit en fait quels seront les paquets affectés par notre filtrage. Il existe 3 chaines :

- **INPUT**, c'est à dire les paquets **entrants** à **destination de l'hôte**.
- **OUTPUT**, c'est à dire les paquets **sortants** provenant **directement de l'hôte**.
- **FORWARD**, c'est à dire les paquets **transitants** par l'hôte.

On a notre chaine, à présent on aimerait bien agir dessus. On a 4 actions possibles sur ces chaines :

- **ACCEPT**, les paquets continuent leur petit bonhomme de chemin.
- **DROP**, les paquets sont ignorés
- **REJECT**, les paquets sont ignorés et l'expéditeur est prévenu
- **LOG**, on enregistre toutes les entrées/sorties de paquet dans un fichier log

On peut maintenant agir sur nos chaines. SUPER.
Vous comprendrez qu'agir sur *tous les paquets*, ce n'est pas vraiment précis, mais heureusement, il existe des options de filtrage.
Un petit rappel du fameux modele OSI :



Il existe 3 types d'options de filtrage, du plus généraliste au plus précis, qui peuvent s'utiliser ensemble :
- Les filtrages sur **Trames**, qui correspond au niveau 2 du modele OSI. On agit sur les interfaces réseaux ;
- Les filtrages sur les **paquets ip**, c'est à dire la couche 3 du modele OSI. On agit ici par rapport aux adresses ip sources et destinations mais aussi aux protocoles (tcp/udp/icmp) utilisés par les paquets ;
- Les filtrages sur les **paquets tcp/udp**, c'est à dire la couche 4 du modele OSI. On agit par rapport aux ports sources et destinations, et à l'état de la connexion.

Filtrage sur trames .
Il existe deux options : *–in-interface* [nom d'interface] et *–out-interface* [nom d'interface].
–in-interface désigne l'interface par laquelle les paquets **entrent** (l'hôte reçoit alors des paquets) et, de la même manière, *–out-interface* désigne l'interface par laquelle les paquets **sortent** (l'hôte envoie alors des paquets).

Filtrage sur paquets ip Ici, on sera un peu plus précis.
On peut préciser l'adresse ip de l'expediteur (*–source* [adresse ip]), ou du destinataire (*–destination* [adresse ip]) ainsi que le protocole affecté par le filtrage (*–protocol* [tcp/udp/tcmp/all]).

Filtrage sur paquets tcp/udp .
On précise ici les ports expediteur (*–source-port* [port]) ou port destinataire (*–destination-port* [port]) affectés au filtrage. On peut aussi préciser les types de connexion concernés par le filtrage avec *-m state –state* [NEW/ESTABLISHED/RELATED].

1.2 Exemples commentés

```
# supprime toutes les regles mais garde les existantes
iptables -t filter -F
# tous les paquets entrants par l'interface eth0 à destination de l'hote sont acceptés
iptables -t filter -A INPUT --in-interface eth0 --jump ACCEPT
```

```
# tous les paquets à destination de 192.168.30.45 de la part de l'hote sont jetés
iptables -t filter -A OUTPUT --destination 192.168.30.45 --jump DROP
# tous les paquets provenant de l'hote depuis le port 80 est accepté
iptables -t filter -A OUTPUT --protocol tcp --source-port 80 -jump ACCEPT
```

1.3 Translation d'adresses : NAT

La translation d'adresses permet en fait d'utiliser **1 adresse ip publique** pour représenter **plusieurs machines**. En fait, le serveur va "faire correspondre" un port sur son adresse ip à une adresse du réseau privé.
Sous GNU/Linux, on peut effectuer 3 actions différentes :

- Masquer l'adresse ip de l'emetteur par l'adresse ip d'une interface du serveur (*Masquerade*) - POSTROUTING
- Changer l'adresse ip de l'emetteur par une adresse fixe spécifiée (*SNAT*) - POSTROUTING
- Changer l'adresse ip du destinataire par une adresse fixe spécifiée (*DNAT*) - PREROUTING

Il faut savoir qu'enfin, il existe 2 étapes bien distinctes, *POSTROUTING* et *PREROUTING*. Le *POSTROUTING* concerne les paquets qui arrivent depuis l'exterieur vers le serveur, et qui sont destinés à une machine du réseau interne, alors que le *PREROUTING* concerne les paquets envoyés depuis le réseau interne vers l'exterieur.
C'est pas très clair ? Regardons l'exemple commenté :

```
# tous les paquets sortant par l'interface ppp0 pour le reseau interne
# auront l'adresse ip du serveur
iptables -t nat -A POSTROUTING --out-interface ppp0 --jump MASQUERADE
# les paquets concernés sont les mêmes qu'au dessus
# sauf que cette fois ils auront l'adresse ip xxx.xxx.xxx.xxx
iptables -t nat -A POSTROUTING --out-interface ppp0 --jump SNAT --to-source xxx.xxx.xxx.xxx
```

2 Administration système : gestion des utilisateurs

Il n'y a pas grand chose à ajouter par rapport au guide de Mr.Lacour.
Hmm, non franchement, je vois pas.
Rien à ajouter en fait.

je me suis fait PRESHOT