iNFO — IUT GRAND OUEST NORMANDIE

R 3.06

2023 – 2024

Architecture des réseaux

TP n° 2
Pare-Feu

ANNE Jean-François
D'après le cours de M. JEANPIERRE

Pare-Feu

A. *Présentation :*

1°) *Introduction*

iptables est un outil de gestion de pare-feu, intégré au noyau Linux (depuis la version 2.4). Le pare-feu en lui-même s'appelle Netfilter. Le principe de fonctionnement est simple, lorsqu'un paquet est reçu, il est transmis à la partie Netfilter du noyau. Netfilter va ensuite étudier ce paquet en se basant sur des règles définies par l'administrateur. Il faut bien évidemment être root sur les machines pour manipuler le pare-feu.

Son fonctionnement repose sur des tables de « chaînes », c'est à dire une suite nommée et ordonnée de règles. La table principale est la table ***filter***. Elle contient 3 chaînes par défaut :

- **INPUT**, qui correspond à ce que reçoivent les programmes de la machine,
- **OUTPUT**, qui correspond à ce qu'envoient les programmes de la machine,
- **FORWARD**, qui est utilisée lorsque les paquets sont routés, c'est à dire ne concernent pas les programme de la machine.

Ces trois chaînes ont aussi une action par défaut (policy) qui exprime ce qui doit être fait si aucune règle ne convient.

Important : Lorsque le paquet entre dans une table, il est comparé aux règles de cette table, **dans l'ordre selon lequel les règles se présentent**. C'est-à-dire que si un paquet est autorisé par une règle mais que ses critères correspondent à une règle située avant, alors le paquet suivra ce qui est indiqué par la première règle. Cela peut avoir de l'importance par exemple dans le cas d'un protocole que l'on souhaite utiliser seulement sur un réseau local tout en étant connecté à l'internet.

Attention, le fonctionnement de routage est désactivé par défaut sous Linux :

```
cat /proc/sys/net/ipv4/ip_forward
```

Pour l'activer de manière permanente, il convient d'ajouter la directive « net.ipv4.ip_forward = 1 » au fichier */etc/sysctl.conf*. Vous pouvez aussi le faire de manière temporaire avec la commande « sysctl –w »

IPtables étant extensible, il a fait l'objet de nombreux ajouts. En particulier, de nouvelles tables sont parfois disponibles (en fonction des options du noyau ou des modules chargés). Par exemple :

- raw : sert à marquer les paquets avant le suivi de connexion (voir plus bas)
- nat : sert à modifier les adresses du paquet
- mangle : sert à modifier le contenu du paquet : TOS, TTL, …

IPtables peut fonctionner en mode *statefull*. Dans ce cas (module *nf_conntrack*) il marque les paquets IPv4 (pas encore IPv6 !) comme :

- NEW : Premier paquet d'une conversation
- ESTABLISHED : Paquet d'une conversation en cours
- RELATED : Paquet associé à une conversation en cours
- INVALID : Paquet non classable

Ce fonctionnement peut être observé dans /proc/net/ip_conntrack.

Les différentes tables/chaînes standard sont donc les suivantes, dont voici un petit récapitulatif des tables et de leurs fonctions :

- La table ***filter*** : permet le filtrage des paquets et des connexions. Elle contient la chaîne INPUT qui gère les paquets entrants (destiné aux sockets local), la chaîne OUTPUT qui gère les paquets sortants (générés localement) ; et la chaîne FORWARD qui gère les paquets qui vont passer par la machine mais sans lui être destinés. La table ***filter*** est la table par défaut, celle qui est utilisé si aucune autre n'est spécifié dans les options de la commande. C'est aussi celle que nous allons utiliser pour notre pare-feu.
- La table ***nat*** : contient les chaînes PREROUTING, POSTROUTING et OUTPUT. Elle permet de modifier la route que le paquet est censé prendre. C'est celle qui est utilisé par exemple pour faire du NAT et ses dérivés SNAT/DNAT/PAT….
- La table ***mangle*** : contient les chaînes PREROUTING, INPUT, FORWARD, OUTPUT et POSTROUTING. Elle permet de modifier un paquet traversant la machine, mais passant par un processus local.
- La table ***raw*** : pour configurer des dérogations à conntrack, cette table est prioritaire, elle travaille sur les paquets bruts. Elle contient les chaînes PREROUTING, INPUT, FORWARD, OUTPUT et POSTROUTING. Elle permet de modifier un paquet traversant la machine, mais passant par un processus local.
- Enfin, la table ***security*** : contient les chaînes INPUT, FORWARD et OUTPUT. Elle est utilisée avec les cibles SECMARK et CONNSECMARK, en combinaison avec un module comme SELinux, afin d'effectuer un contrôle d'accès mandataire.

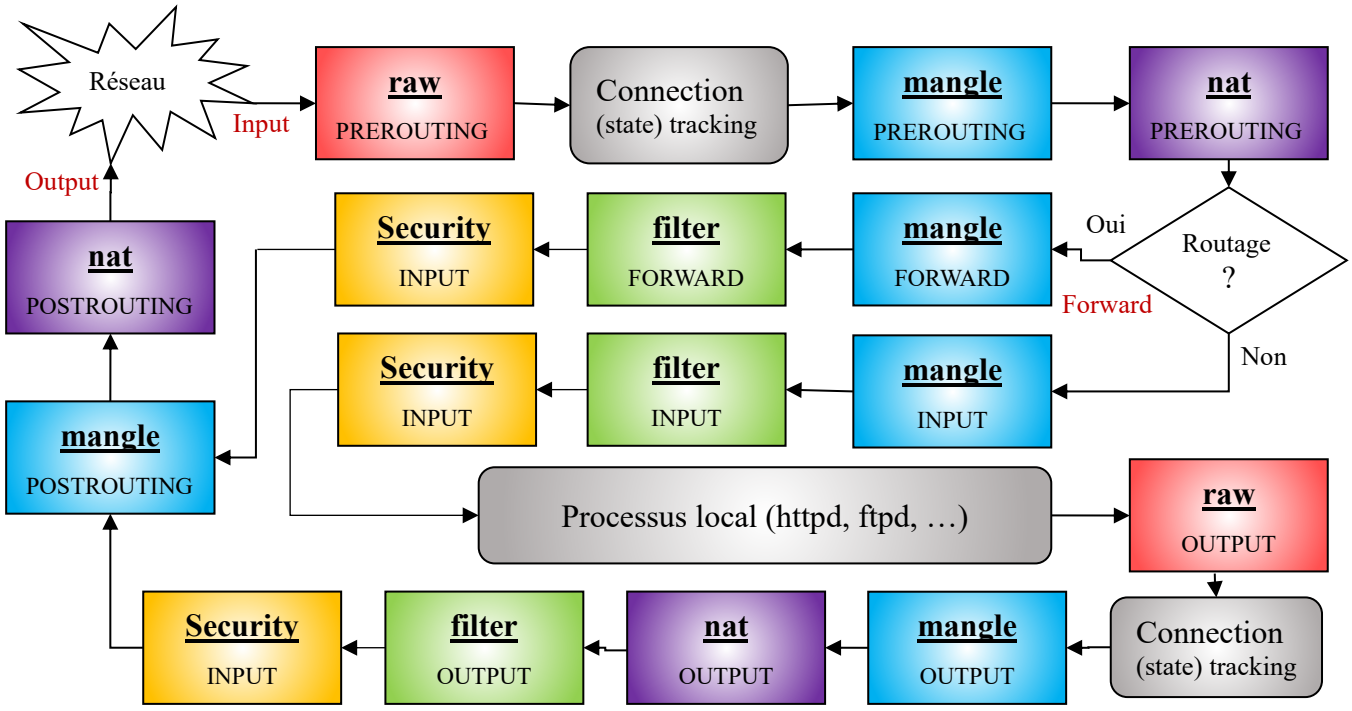
Intéressons-nous à la table ***filter*** qui s'appuie sur trois chaînes : INPUT, OUTPUT et FORWARD. Quand un paquet arrive de la carte réseau, le noyau regarde la destination de ce paquet. Deux possibilités :

1. Ce paquet est destiné à cette machine : il passe dans la chaîne INPUT, en direction des processus qui l'attendent
2. Ce paquet est destiné à une autre machine : là encore deux possibilités :
    - Le forwarding est autorisé : le paquet passe dans la chaîne FORWARD
    - Le forwarding n'est pas autorisé : le paquet est effacé

La chaîne OUTPUT concerne les paquets créés par la machine locale. Ces paquets passent par la chaîne OUTPUT immédiatement. Chacune de ces chaînes peut ensuite donner plusieurs réponses pour chaque paquet :

- ACCEPT : le paquet est accepté
- DROP : le paquet est refusé, on l'efface et on ne répond rien
- REJECT : le paquet est refusé et on signale le rejet à l'expéditeur

Chaque chaîne possède une politique (policy) de filtrage. Nous verrons par la suite que l'administrateur peut créer des chaînes personnalisées supplémentaires.



Vous pouvez ajouter une règle à une table avec la commande :

```
iptables -t table commande chaîne correspondance cible
```

Remplacez *commande* par :

- -A (--append)
- -D (--delete),
- -R (--replace),
- -I (--insert) suivies du numéro de ligne à supprimer/remplacer/insérer
- -L (--list),
- -F (--flush - vide les entrées)
- -Z (--zero) remet les compteurs à 0
- -N (--new-chain),
- -X (--delete-chain) suivies du nom de la chaîne à créer/supprimer
- -P (--policy) *ACCEPT / DROP* règle la politique par défaut de la chaîne sur *accept* ou *drop*)
- -E (--rename-chain) suivie de l'ancien nom et du nouveau nom de la chaîne à renommer

Remplacez *cible* par :

- -j *nom* saute à la chaîne spécifiée
- -j ACCEPT / DROP / REJECT termine la chaîne en acceptant/supprimant/rejetant le paquet
- -j REJECT --reject-with *msg* : rejette le paquet avec le message spécifié : *icmp-net-unreachable, icmp-host_unreachable, icmp-port-unreachable, icmp-proto-unreachable, icmp-net-prohibited, icmp-host-prohibited,* ou *tcp-reset* (TCP seulement)
- -j SNAT, -j DNAT : NAT sur l'adresse source / destination ex : iptables -t nat -A PREROUTING -p tcp -d 15.45.23.67 --dport 80 -j DNAT --to-destination 192.168.1.1-192.168.1.10 => redirige les trames qui seraient pour 15.45.23.67:80 vers l'une des adresses 192.168.1.1-10
- -j LOG inscrit le paquet dans le journal système avec les options suivantes :
  - --log-level *debug* : log le paquet avec la priorité *debug*
  - --log-prefix "AAA" : préfixe le log avec la chaîne *AAA*
  - --log-tcp-sequence : log les n° de séquence TCP
  - --log-tcp-options, --log-ip-options : log les options TCP / IP
- -j NETMAP --to NET_IP : modifie l'adresse de réseau exemple : iptables -t mangle -A PREROUTING -s 192.168.1.0/24 -j NETMAP --to 10.5.6.0/24 mappe 10.5.6.1 sur 192.168.1.1 le .2 sur le .2, ...
- -j REDIRECT --to-ports *x* : redirige le paquet vers le port *x* de la machine elle-même (proxy transparent)

- -j RETURN : arrête la chaîne active
- … et bien d'autres encore…

Remplacez enfin *correspondance* par :

- ! *correspondance* : tout sauf la correspondance spécifiée
- -p suivie d'un protocole : uniquement ce protocole
- --sport, --source-port, --dport, --destination-port : uniquement le(s) port(s) spécifié(s) exemple : -p tcp --sport 80 --dport 1000:2000 => uniquement les ports tcp de 80 vers 1000 à 2000
- --tcp-flags : seulement si les drapeaux (*SYN*, *RST*, *ACK*, *FIN*, *ACK SYN*) sont mis.
- --icmp-type *type* : seulement le type icmp spécifié. (voir *iptables --protocol icmp --help*)
- -s, --src, -d, --dst : uniquement l'adresse IP source/destination spécifiée. ex : -s 192.168.1.1 -d 192.168.1.0/24
- -i, -o, --in-interface, --out-interface suivies d'une interface réseau : seulement les paquets depuis/vers elle
- -m owner : correspondance sur le processus émetteur de la trame (OUTPUT)
  - --cmd-owner suivie d'une commande : uniquement les paquets émis par celle-ci
  - --uid-owner --gid-owner suivies d'un UID/GID : seulement les paquets émis par cet utilisateur ou ce groupe
  - --sid-owner suivie d'un N° de processus : seulement les paquets envoyés par lui ou ses fils
- -m state --state RELATED/INVALID/ESTABLISHED/NEW : seulement les paquets dans l'état considéré (les 2 options sont nécessaires ici)
- -m limit --limit-burst : limite le nombre d'activations de la règle à la valeur spécifiée
- -m limit --limit *x/minute* : définit la vitesse de restauration de la règle précédente (limit-burst) ex : -p icmp -m limit --limit-burst 5 --limit 5/second –j ACCEPT : Si 10 pings arrivent toutes les 0,1 seconde, alors : Oui, O, O, O, O, N, O, N, O, N car il faut 0,2 secondes pour régénérer une autorisation
- … et encore plein d'autres …

Exemple : iptables –A INPUT –p icmp –J DROP rejette tous les paquets ICMP reçus.

Allez, il est temps de commencer à bosser un peu…

2°) *Sécurité minimale (déconseillé)*

On désire autoriser tous les paquets entrant vers ou sortant de votre machine :

```
service firewalld stop  Eteint le pare-feu « pour les nuls »
service ufw stop  Eteint le pare-feu « pour les nuls »
iptables -F Vide les règles de la table
iptables -L --line-numbers   Affiche le contenu des tables
```

Corrigez éventuellement les politiques des chaînes INPUT, OUTPUT et FORWARD.

```
ping eduifs
```

Attention, les distributions Linux incluent souvent un pare-feu graphique, qui utilise IPtables, mais de manière relativement compliquée. On va donc commencer par le désactiver si nécessaire. Les plus courants sont *firewalld* et *ufw*, mais vous pouvez aussi rencontrer *firestarter* et bien d'autres encore…

3°) *Sécurité moyenne (standard)*

Le ping vers la machine eduIfs doit fonctionner…

- **Supprimez** tous les paquets entrant vers votre machine.
- **Testez** un ping vers votre machine.
- **Testez** un ping vers 192.168.0.2.
- **Corrigez** la chaîne INPUT pour que vous puissiez pinger 192.168.0.2.
- **Tester** un ping vers votre machine.
- **Corrigez** éventuellement la chaîne INPUT pour que cela ne fonctionne plus, c'est-à-dire n'autoriser en entrant QUE les réponses à vos requêtes.

- **Vérifiez** que le **ping** vers 192.168.0.2 fonctionne encore.
- **Vérifiez** que la connexion **sftp** vers ruche fonctionne : sftp login@ruche

Ceci permet de bien valider le fait que tout le trafic initié par votre machine est bien autorisé.

Vous avez ici le fonctionnement standard d'un pare-feu : tout le trafic sortant est autorisé, alors que tout le trafic entrant non sollicité est refusé. L'utilisateur lambda ne devrait donc pas avoir conscience de limitations.

4°) *Filtrage sortant*

- **Bloquez** tout le trafic sortant de votre machine.
- **Modifiez** la chaîne OUTPUT pour que le **ping** vers 192.168.0.2 fonctionne.
- **Testez** un ping vers la machine eduIfs.
- **Ajoutez** une règle à la chaîne OUTPUT de votre machine pour autoriser les paquets UDP dont le port destination est 53.
- **Vérifiez** qu'un ping vers la machine eduIfs fonctionne à nouveau.

5°) *Autorisez le SSH sortant*

- **Ajoutez** maintenant une règle autorisant le trafic **TCP** vers le port **22** de la machine PC-prof (RX-30) (Salle LP : 192.168.1.160) ou (Salle Réseau :192.168.4.30).
- **Vérifiez** par un `ssh adminetu@192.168.1.160` (ou `adminetu@192.168.4.30`) que votre règle est correcte.
- **Vérifiez** que le `sftp login@ruche` ne fonctionne plus.

6°) *Créons un petit serveur local…*

- **Créez** un serveur TCP sur votre machine à l'aide de l'outil *netcat* :
  `nc -l <port>` *Attention, il s'agit d'un L minuscule (Listen)*
- **Autorisez** ce port en entrée.
- **Vérifiez** à l'aide de PC-Prof que vous pouvez vous y connecter avec un telnet ou un netcat.
- **Testez** la connexion en local.

7°) *Un petit tunnel pour la route…*

- **Testez** les commandes suivantes :

```
ssh -f etudiant@192.168.1.160 -L 2222:<votre IP>:<votre port> -N
telnet 127.0.0.1 2222
```

- **Analysez** le comportement de la connexion à l'aide de Wireshark.

8°) *Deux autres pour les courageux :*

- **Ouvrez** maintenant un tunnel permettant au **sftp vers ruche** de fonctionner…
- **Ouvrez** un tunnel vers le port **3128** de la machine **192.168.1.1**.
- **Configurez** Firefox pour l'utiliser.

Vérifier que vous avez accès à Internet.

9°) Nettoyez votre machine !

N'oubliez pas le NETTOYAGE
en fin de séance

B. ***Webographie***

- *https://karchnu.fr/pages/securite-systemes-et-reseaux/iptables.html*
- *https://wiki.archlinux.fr/Iptables*
- *https://stuffphilwrites.com/2014/09/iptables-processing-flowchart/*