GUIDE D'UTILISATION DE NETFILTER_CFG

Olivier ALLARD-JACQUIN olivieraj@free.fr

Version 0.5.8alpha3 - 11 mars 2004

PLAN :

Le but de ce programme est de configurer le firewall intégré aux kernels Linux 2.4 et supérieurs appelé " netfilter".
Le principal outil pour configurer ce firewall est la commande "iptables" qui doit être lancée en temps que root.

Vous trouverez la dernière version de ce document :

- sur son site principal : http://olivieraj.free.fr/fr/linux/programme/netfilter_cfg/.
- en version PDF. Pour le lire, utilisez le lecteur Xpdf ou Acrobat Reader.
- sous forme d'archive complète, au format .tgz (.tar.gz), afin de lire en hors ligne.

I Téléchargement

La licence de ce programme et de sa documentation est évidement la GPL

La dernière version de "Netfilter_cfg" est la 0.5.8

Vous pouvez télécharger d'autres versions plus anciennes ici :

- netfilter_cfg 0.5.7
- netfilter_cfg 0.5.6
- netfilter_cfg 0.5.5
- netfilter_cfg 0.5.4

II Historique

Après de nombreuses années à entendre parler et à survoler de très haut le problème du firewall sous Linux, je me suis finalement décidé à y regarder de plus près, et à me plonger dans la documentation.

On appelle "firewall" (littéralement "mur de flammes" en français) tout ce qui touche au filtrage des connexions sur un ordinateur ou un réseau. Plutôt que de laisser des transmissions réseaux se faire dans tout les sens, et donc qu'une personne extérieur au réseau interne puisse récupérer des informations dont il n'est pas supposé pouvoir accéder, le firewall est là pour contrôler les flux de données, interdire et éventuellement enregistrer (on utilise le terme de "log" en anglais) les flux non autorisés ou "bizarres".

L'approche des techniques de firewall de Linux est très différente de ce que l'on peut trouver dans un OS largement connu comme Windows®. Contrairement à cet OS, et notamment à sa dernière monture (en 2003) appelée "Windows XP®", le firewall Linux est intégré directement dans le coeur de l'OS, au plus près des fonctions réseau du système.

Il en résulte que :

- Il n'existe qu'un seul moyen de configurer le firewall Linux, à savoir via la commande système "iptables". La création d'un firewall se résume alors à l'écriture d'un script, plus ou moins complexe, adapté à la configuration de l'ordinateur ou du réseau.
- Par défaut, il n'existe pas dans Linux d'interface graphique, de "drag'n drop", ou autre GUI pour configurer le firewall. Les utilisateurs Windows® en seront sans doute chagrinés, mais ils peuvent cependant trouver sur Internet (par exemple sur freshmeat.net) de nombreuses interfaces graphiques de ce type.

- Le lecteur pourra comme moi trouver sur Internet de très nombreux scripts d'exemples permettant de configurer son firewall. Cependant, j'ai trouvé que :
  - Aucun de ces scripts n'était vraiment adapté à ma configuration.
  - Comme ma machine passe souvent d'un réseau à un autre, et d'un moyen de connexion à Internet à un autre, il me fallait un script plus complexe, qui puisse s'adapter facilement, et le plus automatiquement possible, au réseau où il se trouvait.
  - Qu'il était finalement beaucoup plus intéressant de construire soit même son propre script, à la fois en lisant de la documentation, mais aussi en analysant des scripts d'exemples afin d'y trouver de l'inspiration.

Ainsi est donc né "netfilter_cfg". Ce script / programme ne se veut pas révolutionnaire, ou exceptionnel, mais uniquement pratique et le plus facilement modifiable, adaptable, et utilisable possible.

Enfin, ce script a principalement été pensé dans le cadre d'une utilisation non permanente à Internet, via un modem par exemple ou par une connexion intermittente à l'ADSL. De plus, possédant plusieurs cartes réseaux et adresses IP locales, j'ai voulu écrire un script qui soit le plus adaptable à des configurations "bizarres".

Cela explique donc certains choix techniques, et notamment que ce script doit être exécuté à chaque fois que la connexion Internet est activée . Un des gros intérêt de ce script est qu'il détecte automatiquement l'état de la connexion à Internet, et qu'il se configure automatiquement en conséquence, sans attendre de la part de l'utilisateur un paramétrage quelconque ...

A noter que dans ce document je parle principalement du firewall, mais que le rôle de netfilter ne se résume pas qu'à ceci. Cette couche du kernel Linux sert aussi au NAT (Network Adress Translation), au marquage des paquet (la table "Mangue"), et à plein d'autres choses ...

Enfin, je suis l'auteur d'un autre document ayant pour sujet Netfilter et les mesures de protections à prendre afin de protéger sa machine ou son réseau local. Il s'agit d'un tutoriel que j'ai écris en parallèle à une conférence que j'ai présenté dans le cadre de l'association Guilde, le 2 juillet 2003. Vous y trouverez un grand nombre d'informations, tant destinées aux néophytes qu'aux utilisations plus complexes, qui vous aiderons à mieux comprendre l'utilisation et le fonctionnement de "netfilter_cfg".

III Philosophie générale de filtrage

Les capacités de filtrage de ce programme se veulent les plus simples possibles (sans pour autant transformer votre machine en gruyère en terme de sécurité), aussi :

- Toutes connexions entrantes par l'interface loopback ou sortantes sur l'interface loopback sont autorisées (voir la fonction "LoopbackRules").
- Toute connexions entrant par les interfaces réseaux ET venant des réseaux locaux sont autorisées. De même que toutes les connexions sortant par les interfaces réseaux ET à destination des réseaux locaux. Voir la fonction "LanRules".
- Vis à vis d'Internet (le "WAN" ou "Wide Area Network" ou "réseau large" en Français) seule les connexions initialisées par la machine (ou le réseau local dans le cas d'utilisation du NAT) sont autorisées. Les connexions qui n'ont pas été demandées par la machine, et qui sont généralement des "port scanning", sont donc refusées. Exception fait évidement des connexions de données FTP en mode actif. Tout ceci se passe dans les fonctions "ModemRules et GatewayRules".
- Ce script accorde une totale confiances aux programmes exécutés sur la machine, ou dans le cas de l'utilisation du NAT, aux machines du réseaux qui lancent des connexions sortantes. Donc tout les programmes de la machine peuvent demander des connexions sur Internet, le firewall les laissera passer.

Certains utilisateurs Windows® vont sans doute trouver ce dernier point aberrant, mais je justifie ce choix par 2 choses :

- J'ai confiance dans les logiciels Linux que j'emploie, car ils sont GPL. Je n'estime pas qu'ils sont bourrés de spywares ou autre cochonneries de ce genre, et je dors tranquille lorsque ma machine reste connectée de longues heures durant la nuit (les téléchargements via un modem RTC, c'est long ...).
- Je ne passe pas mon temps à lutter contre mon OS ou ses applications natives (Internet Explorer®, Windows Media Player®, Microsoft Office®, etc ...) , afin que celles-ci ne se connectent pas sur Internet, et ne diffuses mes données personnelles , mes numéros de série de logiciels , ou la liste des applications installées sur ma machine.

Quoi qu'il en soit, je conçois que l'utilisateur venant du monde Windows® éprouve certaines craintes vis à vis du comportement de ses logiciels. Une prochaine version de ce script rajoutera donc un niveau de protections supplémentaires, et permettra de limiter les connexions à destination d'Internet.

Les conséquences de ces règles de filtrages sont :

- La présence du firewall sera complètement transparente pour les applications lancées sur la machine, ou sur les connexions initialisés par le réseaux local et passant par le NAT.
- Vu de l'extérieur, la machine sera vu comme un "trou noir", qui ne répondra à aucune des tentatives de connexions initialisés par l'extérieur. Le port scanning sera donc inopérant. Sous Windows®, les vendeurs de logiciels de firewall appelle cela une technologie "stealth" (furtive).

IV Configuration du programme

Avant d'utiliser ce programme, il va falloir tout d'abord que vous le configuriez. Et pour cela, il va falloir que vous modifiez son code source... Ce n'est pas en soit très très difficile, il suffit d'avoir quelques notions de base du langage utilisé : le "sh" / "bash".

Les modifications du code source doivent se limiter à la première partie du script, où sont déclarées les variables globales de celui-ci. Si vous modifiez la suite du programme, vous le faite à vos risques et périls...

Par convention :

- Les termes entre "**<**" et "**>**" désignent des paramètres. Exemple "<Interface>". C'est comme la variable "x" en mathématique ...
- Le caractère "**|**" parmi les "<" et ">" signifie "ou", c'est à dire que l'on peut choisir l'une ou l'autre des options indiquées. Exemple : "<routable|non_routable>" signifie que le paramètre peut prendre la valeur "routable" ou la valeur "non_routable".

IV-1 Configuration du réseau local

Le script gère autant de réseaux locaux que vous en avez. Vous pouvez avoir plusieurs cartes réseaux sur votre machine, et / ou plusieurs adresses IP par carte réseau. Pour chaque réseau que vous avez déclarés dans votre configuration de Linux, vous devez rajouter une ligne de type :

LAN[<un nombre>]="<Interface>|<Adresse IP>|<Masque de sous réseau> |<Adresse de broadcast>|<Adresse de la passerelle>|<Options>"

- <un nombre> est le numéro du réseau. Cela doit être un nombre croissant supérieur ou égal à 0.

  Exemple :

  LAN[0]=...
  LAN[1]=...
  LAN[2]=...

- <Interface> est le nom de l'interface réseau .

  En général, c'est eth0, eth1, etc...

  Si vous avez plusieurs adresses IP pour une carte réseau, comme par exemple "eth0:0", "eth1:0", "eth1:1", etc... cela sera toujours le nom de l'interface réseau physique auquel sont accrochés les interfaces virtuelles :

  - eth0:0 => eth0
  - eth1:0 => eth1
  - eth1:1 => eth1

- <Adresse IP> : C'est l'adresse IP de la machine.

  Pour un réseau local, il est préférable d'avoir une adresse IP de l'un ou l'autre de ces types :

  - classe A privé : 10.0.0.0 / 255.0.0.0 (Exemple : 10.0.0.1).
  - classe B privé : 172.16.0.0 / 255.255.0.0 (Exemple : 172.16.0.1).
  - classe C privé : 192.168.0.0 / 255.255.255.0 (Exemple : 192.168.0.1

- <Masque de sous réseau> : Le masque de sous-réseau . Pour un réseau local, c'est en général 255.255.255.0 (Réseau de classe C).

- <Adresse de broadcast> : C'est une adresse permettant d'envoyer un message à toutes les machines du réseau en même temps. Par exemple, pour une adresse classe C, il faut remplacer le dernier chiffre de l'adresse IP de la machine par "255".

  Exemple :

  - Adresse IP : 192.168.0.1 .
  - Masque de sous-réseau : 255.255.255.0 .
  - Adresse de broadcast : 192.168.0.255 .

- <Adresse de la passerelle> ("gateway" en anglais) : Cette option n'est là que pour une raison de compatibilité ascendante du programme, mais elle n'est plus d'actualité. **Ne mettez donc aucune valeur**, même si vous êtes dans un réseau local et que vous utiliser une passerelle pour en sortir. Netfilter_cfg a été écrit afin de retrouver par lui-même l'adresse IP de votre passerelle.

- <Options> : Il s'agit d'une série d'options sur la configuration de l'interface réseau. Si vous utilisez plusieurs options vous devez séparer chaque options par un ":" . Les différentes options possibles sont :

  - routable : Utilisez cette option si vous voulez que les autres machines du réseau (décrit par cette ligne LAN[..]) puisse utiliser votre machine pour se connecter à

internet. C'est ce que l'on appelle le NAT. Votre machine sera alors transformé en passerelle pour le réseau indiqué.
- non_routable : Cette option est la valeur par défaut de la technique de NAT. Elle indique que le réseau local n'est pas autorisé à utiliser votre machine pour se connecter à Internet.
- déroupé : Si vous utilisez une connexion ADSL dégroupée, vous devez utiliser cette option. Elle permet de renforcer la sécurité de votre machine vis à vis des autres clients de votre FAI.
- dhcp_servers : Cette option permet à votre machine de recevoir les requêtes DHCP lancées sur votre réseau local par des machines demandant une adresse IP. Ceci n'est intéressant que si votre machine fait tourner un serveur DHCP, ce qui est assez rare sur une machine personnelle.

Remarques :

- Pour pouvoir utiliser l'option "dhcp_servers", vous devez lancer "netfilter_cfg" avec l'option "--lan-dhcp-server on". Ou alors, il faut activer cette option par défaut, en changeant 'WAITING_PARAMETERS[15]="--lan-dhcp-server|on|off|**off**"' par 'WAITING_PARAMETERS[15]="--lan-dhcp-server|on|off|**on**"'

- Pour que le NAT fonctionne, il faut passer le paramètre "--nat on" au programme, ou alors activer par défaut le NAT dans le programme, en changeant 'WAITING_PARAMETERS[3]="--nat|on|off|**off**"' par 'WAITING_PARAMETERS[3]="--nat|on|off|**on**"'.

- Le NAT permet aux autres machines du réseau de se connecter à Internet, en utilisant l'adresse IP de votre propre machine. Sous Windows®, cela s'appelle du "partage de connexion". Tout comme votre machine, les machines du réseau deviendront elles aussi "invisibles" aux "attaquants" d'Internet. Par contre, le firewall de votre machine ne peut pas filtrer les connexions faites par les machines du réseau. En bref, cela veut dire que :

  - Si il y a un spyware, un virus, ou une autre cochonnerie de ce type sur une machine du réseau, celui-ci pourra "sortir" sur Internet, en se faisant passer pour votre machine.
  - Vous ne pourrez pas empêcher Windows® ou les logiciels associés qui tournent sur les machines du réseau de se connecter sur Internet.

  Si vous voulez museler les machines de votre réseau, il faudra soit :

  - Que vous installiez des "firewalls personnels" sur chacune des machines de votre réseau.
  - Acheter des licences pour chacun des logiciels propriétaires de ces machines afin d'avoir la conscience tranquille... 😀
  - Passer toutes les machines du réseau local sous Linux...

Exemple :

```
LAN[0]="eth0|192.168.0.1|255.255.255.0|192.168.0.255||routable:dhcp_server"
LAN[1]="eth0|192.168.1.1|255.255.255.0|192.168.1.255||routable"
LAN[2]="eth0|192.168.2.1|255.255.255.0|192.168.2.255||non_routable"
LAN[3]="eth1|62.147.73.23|255.255.255.0|62.147.73..255||non_routable:degroupe"
```

Dans cette exemple :

- La machine possède physiquement 2 cartes réseaux ("eth0" et "eth1").
- Elle a 3 adresses IP sur la première carte réseau ("192.168.0.1", "192.168.1.1" et "192.168.2.1").
- Les "interfaces" "eth0:0" et "eth0:1" sont des interfaces réseaux virtuelles pour les 2 adresses IP "192.168.1.1" et "192.168.2.1". Mais notez bien que l'on écrit dans netfilter_cfg le nom de l'interface réseau physique "eth0" et non les noms des interfaces réseaux virtuelles ("eth0:0" et "eth0:1")
- Les 2 premiers réseaux connectés sur la carte réseaux " eth0" sont routables (ils peuvent utiliser le NAT), par contre le 2nd réseau virtuel (eth0:1/192.168.2.1/255.255.255.0) ne peut pas se connecter sur Internet.

- Les machines des réseaux locaux connectés sur la carte réseau eth0 pourront recevoir automatiquement une adresse IP, grâce au serveur DHCP se trouvant sur la machine.
- La 2nd carte réseau physique (eth1) est connectée à un modem ADSL sur une ligne dégroupée.

En général, vous n'avez qu'une seule carte réseau local, donc cela devrait aller assez vite à configurer. Utilisez les commandes "ifconfig" en temps que root ou "/sbin/ifconfig" en temps qu'utilisateur simple afin de remplir à ces informations.

Dans une prochaine version de ce programme, je rajouterai un module de détection automatique de la configuration réseau ce qui évitera de configurer cette partie là du script...

IV-2 Debug

Ce programme peut afficher ou sauver dans un fichier des messages d'informations, permettant de trouver plus facilement des erreurs de paramétrage. C'est ce que l'on appellera par la suite le debug.

Si vous ne voulez pas vous ennuyer, laissez les valeurs par défaut, et passez à la suite...

- DEBUG_DISPLAY=<y|n>
  - "y" : Tous les messages de debug sont affichés à l'écran. C'est très pratique, mais le programme est très verbeux, donc il y a beaucoup à lire...
  - "n" : Le programme n'affichera que le strict nécessaire à l'écran. Les messages d'erreurs notamment...
- IPTABLES_DEBUG_DISPLAY=<y|n>
  - "y" : Les commandes "iptables" utilisées sont affichées, ce qui permet de comprendre comment fonctionne le firewall sous Linux... Là encore, c'est très verbeux. Et puis dans tout les cas, ces commandes sont stockés dans le fichier de debug...
  - "n" : N'affiche pas les commandes "iptables".

  Astuce :

  - Si vous utiliser toutes les options de debug ("DEBUG_DISPLAY=y" et "IPTABLES_DEBUG_DISPLAY=y"), vous pouvez créer un script ("script.sh" par exemple) contenant les commandes "iptables" utilisées.
    Pour cela, tapez :
    /usr/local/sbin/netfilter_cfg | grep "#[D]*I#" | sed "s/#I# *//g;s/#DI#/#/g" > script.sh
  - Si vous stockez les messages de debug dans un fichier (voir paragraphe suivant), vous pouvez aussi extraire ces informations de ce fichier :
    grep "#[D]*I#" /var/log/netfilter_cfg | sed "s/#I# *//g;s/#DI#/#/g" > script.sh
    Mais cela extraira toutes les commandes iptables lancées depuis le début de l'utilisation de netfilter_cfg...
- DEBUG_FILE=...

  Tous les messages de debug et d'erreurs peuvent êtres stockés dans un fichier, afin d'être analysés plus tard. 3 moyens sont proposés, mais vous pouvez faire autrement si vous le voulez :

  - DEBUG_FILE=/dev/null" : Tous les messages de debug sont détruits automatiquement, rien n'est sauvé ... C'est simple, mais pas très instructif ...
  - DEBUG_FILE=/var/log/$PROGRAM_NAME" : Tous les messages de debug sont sauvés dans un fichier unique, " /var/log/netfilter_cfg" par exemple.
  - DEBUG_FILE=/tmp/$PROGRAM_NAME.$$" : A chaque lancement de "netfilter_cfg", un fichier de debug /tmp/netfilter_cfg.xxxxx est créé. Le "xxxx" est le PID du processus, c'est à dire un numéro compris entre 1 et 65536. Cela permet de comparer les messages de debug entre plusieurs utilisation du programme.

IV-3 Log

Lorsque "Netfilter" n'arrive pas à trouver une règles l'autorisant à laisser passer un paquet de données, et parce qu'il est configuré comme ceci par ce programme, Netfilter les détruit impitoyablement. Mais auparavant, il peut stocker cette information via un mécanisme de "log" (on peut traduire cette action par l'anglicisme barbare "logger").

La technique de log utilisé est défini par la variable " LOG_TYPE=<LOG|ULOG|NO_LOG>"

Si vous ne voulez pas vous ennuyer, laisser les valeurs par défaut, et passez à la suite...

- "LOG_TYPE=NO_LOG" : Aucun log n'est sauvé. Cela à le mérite d'être simple, mais ce n'est pas particulièrement intelligent, car vous ne savez pas ce qui se passe. Mais enfin, vous êtes grand, alors vous faites comme vous voulez...

- "LOG_TYPE=LOG" : Les log sont sauvés par le mécanisme de log standard de Linux, à savoir le démon " klog". Il en résulte que les informations seront sauvées dans le fichier /var/log/messages. C'est la technique que je vous conseille et qui est part défaut.

- "LOG_TYPE=ULOG" : On utilise ici un autre mécanisme de log appelé " ulogd". C'est très pratique, car ainsi les log du firewall sont séparés des log du kernel.

  Cependant, "ulogd" n'est pas fournit en standard dans les distributions Linux, et il faut l'installer à la main. Ce n'est pas très compliqué, et je vous laisse lire les explications à ce sujet dans le code source de "netfilter_cfg" ... Pour plus d'information, visitez le site de ulogd.

  Personnellement, j'utilise "ulogd" et c'est vraiment le super pied modèle 0xFFFFFFFF...

- "LOG_TABLES=" : Indique quelles sont les tables et les chaînes qu'il faut logger. La syntaxe est une suite de noms de tables suivit du nom de la chaînes à logger, séparés par des ":" et des "|" : [nom de table]**:**[nom de chaîne]**|**[nom de table]**:**[nom de chaîne]**|**...

  Exemple :

  ```
  LOG_TABLES="filter:INPUT|filter:FORWARD|filter:OUTPUT|\
        nat:PREROUTING|nat:OUTPUT|nat:POSTROUTING|\
        mangle:PREROUTING|mangle:INPUT|mangle:FORWARD|mangle:OUTPUT|mangle:P
  "
  ```

  Remarque : Vous pouvez rajouter des espaces dans cette syntaxe, afin d'éclaircir la syntaxe.

Astuce : Un moyen très pratique de visualiser un fichier de log en temps réel est la commande : tail -f <le nom du fichier>

Par exemple :

```
[root@phoenix /]# tail -f /var/log/messages
```

Laissez cette commande s'exécuter dans un terminal, elle affichera en temps réel les modifications du fichiers "/var/log/messages". Ainsi, vous verrez immédiatement les connexions qui sont refusées par le firewall.

Les autres variables pour le log sont "LOG_PREFIX=" et "LOG_LOG_LEVEL=". Ce n'est pas très important, regardez dans le code source pour avoir plus d'informations...

IV-4 Paramètres du programme

A priori, vous n'êtes pas supposé changer cette partie là, sauf si vous voulez modifier le comportement par défaut du programme. Comme vous le verrez plus tard, le programme peut se lancer en lui fournissant des paramètres, afin de le configurer suivant vos besoins.

Les différents paramètres du programme se trouvent dans la section "WAITING_PARAMETERS". Chaque ligne correspond à un paramètre que sait gérer

"netfilter_cfg". Par exemple, on trouve le paramètre WAITING_PARAMETERS[3]="--nat|on|off|off", ou --nat est le paramètre.

C'est la dernière valeur de chaque ligne (après le dernier "|") qui définie la valeur par défaut. Il faut que ce soit une des précédentes valeurs (sauf la première, évidement).

Par exemple, pour 'WAITING_PARAMETERS[1]="--drop-rules|on|off|on"', le paramètre "--drop-rules" attend 2 valeurs possibles : "on" et "off". Et par défaut, c'est à dire si ce paramètre n'est pas passé au programme, ou si il est passé sans "on" ou "off", c'est la valeur " on" qui sera utilisé.

Un conseil tout simple : N'y touchez pas, les options par défaut vont satisfaire tout le monde dans la majorité des cas... 

IV-5 Règles de rejet systématique

Lorsque vous êtes connectés sur Internet, une grand quantité de trafic va faire réagir votre firewall, et lui faire remplir le fichier de log. C'est notamment le cas de l'activité "peer-to-peer", comme "kazaa", "edonkey", "emule", et autres logiciels de ce genre.

Il peut être donc intéressant de ne pas logger ce type de connexions entrantes, en même temps que de les ignorer. C'est le but du paramétrage "DROP" ("rejeter" en Français) :

DROP[<un nombre>]="<Chaîne>|<Interface>|<Source>|<Cible>|<Type de connexion>|<Port source>|<Port cible>"

- <Un nombre> est le numéro de la règle de rejet. Cela doit être un nombre croissant supérieur ou égal à 0.
- <Chaîne> : C'est la chaîne "filter" qu'il faut renseigner. La valeur doit être " INPUT" ou "OUTPUT". Mettez toujours "INPUT", à moins que vous ne sachiez ce que vous faites.
- <Interface> : C'est le nom de l'interface réseau que l'on va contrôler. Mettre "ppp0" pour une connexion Internet de type ADSL ou modem.
  Ce paramètre peut être vide.
- <Source> : C'est la source du paquet de données. Il peut y avoir plusieurs syntaxes :
  - xx.xx.xx.xx : Une adresse IP. Exemple : 25.68.48.157 .
  - xx.xx.xx.xx/yy : Un réseau avec son masque. Exemple : 25.68.48.0/24 .
  - xx.xx.xx.xx/zz.zz.zz.zz : Un réseau avec son masque. Exemple : 25.68.48.0/255.255.255.0 .

  Ce paramètre peut être vide.
- <Cible> : C'est la cible du paquet de données. La syntaxe est identique à celle de "<Source>".
  Ce paramètre peut être vide.
- <Type de connexion> : C'est le type de paquet. Il y a 3 options : "tcp", "udp" ou "icmp". Ou sinon, utiliser une valeur de protocole définie dans le fichier /etc/protocols.
  Ce paramètre peut être vide.
- <Port source> : Port source de la connexion. Il peut y avoir plusieurs syntaxes :
  - xxxx : Un port bien précis. Exemple : 10738 .
  - xxxx:yyyy : Une plage de ports. Exemple : 4660:4700 .
- <Port cible> : Port cible de la connexion. La syntaxe est identique à celle de "<Port source>".

Exemples d'utilisation :

DROP[0]="INPUT|ppp0|||tcp||4660:4700"
DROP[1]="INPUT|ppp0|||udp||4660:4700"

Ces 2 règles permettent de supprimer tout les paquets de type P2P venant ("INPUT") d'Internet ("ppp0"), quelque soit leur adresses IP source ou destination ("|||"), de type "tcp" ou "udp" (et donc pas du type "icmp"), quelque soit leur port source ("||"), et à destination des ports compris entre "4660" et "4700" ("4660:4700").

Comment rajouter de nouvelles règles de rejet ? C'est très simple : Vous lancez votre firewall, et vous regardez de temps en temps votre fichier de log. Si vous voyer qu'il se remplit de messages de connexion d'un type particulier (faites plus précisément attention au port cible), vous pouvez rajouter une règle de rejet.

Par exemple :

DROP[0]="INPUT|ppp0|||**tcp**||**139**"
DROP[1]="INPUT|ppp0|||**udp**||**137**"

permet d'éliminer les requêtes NETBIOS faites par des "vilains pirates" qui veulent accéder à vos répertoires partagés Windows® ou Samba.

IV-6 Paramétrages particuliers pour les serveurs

A priori, vous n'êtes pas intéressé par ce chapitre. Il s'agit des options permettant de laisser passer des paquets de données à destination de divers serveurs que vous avez pu installer sur votre ordinateur, et dont vous voulez laisser l'accès à Internet .

Hoouu la la, vous êtes sûr que vous avez fait cela ??? Z'êtes pas fou, non ? . Bon, commençons par voir comment paramétrer l'accès à ces serveurs, puis nous verrons un peu plus loin comment indiquer à "netfilter_cfg" de laisser ces serveurs accessibles depuis Internet.

IV-6-1 Tactical Ops

C'est un jeu se jouant en réseau local ou sur Internet. Ces options permettent de configurer les fonctions de serveur sur Internet.

- TO_PORT=<numéro du port> : Définit sur quel port écoute le serveur Tactical Ops. C'est 7777 par défaut.
- TO_WAN_BROADCAST=<y|n> : Autorise le serveur Tactical Ops à émettre des messages destinés à tout le monde sur Internet.
  C'est destiné à informez les "masters servers" que vous proposez un serveur TO. Si vous voulez que votre partie reste discrète, il est préférable de le mettre à " n".
- TO_LAN_BROADCAST=<y|n> : Autorise le serveur Tactical Ops à émettre des messages destinés à tout le monde sur les réseaux locaux. Bon, ce n'est pas très grave, c'est un réseau local après tout. Vous pouvez le mettre à " y".

IV-6-2 Serveur FTP

Si vous avez un serveur FTP, que vous voulez rendre accessible depuis Internet.

- FTP_CMD_PORT=<numéro> : C'est le numéro de port pour les commandes lors de connexions sur un serveur FTP tournant en local. En général, c'est " 21". On peut aussi mettre la valeur "ftp" qui est indiqué dans votre fichier /etc/services.
- FTP_DATA_PORT=<numéro> : C'est le numéro de port pour les données lors de connexions sur un serveur FTP tournant en local. En général, c'est " 20". On peut aussi mettre la valeur "ftp-data" qui est indiqué dans votre fichier /etc/services.

IV-6-3 Serveur HTTP/HTTPS

Si vous faite tourner un serveur web avec des pages encodées (HTTPS <=> port 443) ou non (HTTP <=> port 80).

- HTTP_PORT=<numéro> : C'est le numéro de port lorsque l'on publie des pages HTML non encodées. En général, c'est " 80". On peut aussi mettre la valeur "http" qui est indiqué dans votre fichier /etc/services.

- HTTPS_PORT=<numéro> : Ce numéro de port est utilisé lorsque l'on publie des pages HTML encodées. En général, c'est " 443". On peut aussi mettre la valeur "https" qui est indiqué dans votre fichier /etc/services.

IV-6-4 Serveur SSH (Secure SHell)

Cette option configure le firewall pour rendre un serveur SSH (un shell utilisateur) accessible depuis Internet.

- SSH_PORT=<numéro> : Numéro de port du serveur SSH. En général, c'est " 22". On peut aussi mettre la valeur "ssh" qui est indiqué dans votre fichier /etc/services.

IV-6-5 Serveur Jabber

Jabber est un système de client/serveur de discussion (chat) sur Internet. Le modèle est quelque peu particulier, car les serveurs Jabber sont distribués sur Internet, et n'importe quel machine connecté à Internet peut serveur Jabber. Si vous hébergez un serveur Jabber sur votre machine, il vous faudra ouvrir 2 ports.

- JABBER_C2S_PORT=<numéro> : Ce port permet aux clients Jabber de se connecter à votre serveur. C'est généralement le port " 5222" qui est utilisé.

- JABBER_S2S_PORT=<numéro> : Ce port permet aux serveurs de dialoguer entre eux. C'est généralement le port " 5269" qui est utilisé.

IV-6-6 Autres serveurs

Pour l'instant, je n'ai pas prévu de nouvelles extensions pour serveurs. Mais c'est tout à fait faisable d'en rajouter de nouvelles. Je vous laisse donc le soin de les rajouter vous-même, à moins que vous ne m'en fassiez la suggestion ... Après tout, la licence GPL est faite pour cela ! 

IV-7 Paramétrages particuliers pour les clients

Tout comme les logiciels de type serveurs, certains logiciels de type clients doivent pouvoir être accessible depuis Internet. Il convient dans ce cas de leur ouvrir certains ports, afin qu'ils puissent communiquer et remplir leur office.

IV-7-1 Client GnomeMeeting

GnomeMeeting est un fabuleux logiciel de conférence audio, vidéo et texte. Il permet de faire communiquer plusieurs utilisateurs à travers Internet. Comme il doit pourvoir être contacté par vos correspondants, il a besoin d'un certains nombre de ports ouverts. Consultez la documentation de GnomeMeeting pour plus d'informations sur les besoins d'ouvertures de ports

- GM_TCP_LISTENING_PORT=<nombre> : C'est le port par lequel GnomeMeeting est informé d'un appel entrant utilisant le protocole H.323. La configuration par défaut est "1720". On peut aussi mettre la valeur "h323hostcall" qui est indiqué dans votre fichier /etc/services.

- GM_RTP_PORT_RANGE=<intervalle de ports> : Si vous utilisez un tunnel H.245, c'est la configuration par défaut, GnomeMeeting va utiliser cet intervalle de ports pour communiquer avec vos correspondants. La configuration par défaut est " 5000:5007", mais vous pouvez utiliser un autre intervalle si en même temps vous reconfigurez ce paramètre là dans GnomeMeeting.

- GM_TCP_PORT_RANGE=<intervalle de ports> : Si votre correspondant utilise un logiciel comme "Netmeeting" sous Windows, vous devrez ouvrir ces ports ci pour communiquer avec lui. La configuration par défaut est " 30000:30010", mais vous pouvez utiliser un autre

intervalle si en même temps vous reconfigurez ce paramètre là dans GnomeMeeting.

- GM_GK_PORT_RANGE=<intervalle de ports> : Dans le cas où vous ne communiquez pas directement avec votre correspondant, mais que vous utilisez une "passerelle GateKeeper", vous devez ouvrir ces ports ci. La configuration par défaut est " 30000:30010", mais vous pouvez utiliser un autre intervalle si en même temps vous reconfigurez ce paramètre là dans GnomeMeeting.

IV-7-2 Client xMule

xMule est un logiciel de Peer-To-Peer, c'est à dire d'échange de fichiers. Il utilise le réseau "eMule".

- XMULE_TCP_PORT=<nombre> : Ce port TCP doit être ouvert afin de régler le problèmes des "lowID". La valeur par défaut est " 4662".
- XMULE_UDP_PORT=<nombre> : Ce port UDP doit aussi être ouvert. La valeur par défaut est "4672".

IV-7-3 Client BitTorrent

BitTorrent (BT) est un logiciel de peer-to-peer, à la manière de "xMule". Contrairement à xMule, il utilise un plage de ports TCP "flottants" de 6881 à 6999. On peut change la plage de port sur lesquels BT écoute en lançant le programme BT avec les options "--minport" et "--maxport".

- BT_PORT_RANGE=<intervalle de ports> : Ce sont les ports standards sur lequel le client BitTorrent peut écouter. Si vous utilisez plusieurs clients BT en parallèle, chaque client utilisera un port différents des autres, afin de ne pas interférer avec eux. La valeur par défaut est " 6881:6999" (oui, cela fait beaucoup...).
- BT_TRACKER_PORT=<nombre> : Ce port n'est à utiliser que si vous faites tourner un "tracker" BitTorrent sur votre machine. Ce n'est en général pas nécessaire, donc vous pouvez laisser commenté cette valeur. La valeur par défaut est " 6969".

IV-8 Plus de sécurité avec le kernel

Les couches réseaux du kernel Linux peuvent apporter un lot d'options intéressantes. Netfilter_cfg propose d'utiliser certaines d'entre elles, afin de renforcer un peu plus la sécurité de la machine.

Ces options sont :

- KERNEL_SPOOFING_PROTECTION=<y|n>
  - "y" : C'est l'option par défaut. Les paquets entrants dans une interface mais à destination d'une adresse IP d'une autre interface sont bloqués. Cette technique d'envoi de paquets a pour nom le "spoofing", et est clairement destinée à pénétrer vos réseaux internes. Cette option devrait toujours être à " y". Remarquez que même ainsi, cela n'empêche pas le NAT.
  - "n" : Les paquets de "spoofing" ne sont pas filtrés par les couches réseaux du kernel. Mais ils devraient l'être par Netfilter.
- KERNEL_PING_PROTECTION=<y|n>
  - "y" : Ainsi configuré, toute commande de "ping" est refusé sur toutes les interfaces réseaux. C'est très efficace mais cela peut être un peu trop limitatif pour les réseaux internes, car ils ne peuvent plus utiliser le "ping" à destination de cette machine.
  - "n" : C'est l'option par défaut. Les "ping" ne sont pas filtrés, mais "Netfilter" se chargera de filtrer au moins ceux venant d'Internet. A moins que vous n'utilisiez le paramètre "--wan-ping on".

V Utilisation

V-1 Mise en garde

Bien, maintenant que vous avez modifié le script, il ne vous reste plus qu'à l'exécuter. Comme il est écrit au tout début de ce document , la principale commande qui va être utilisé dans ce script est "iptables", qui doit être lancé en temps que root. Donc si vous n'êtes pas l'administrateur de votre machine, jetez ce programme et allez jouer à Frozzen Bubble. Ou retournez sous Windows® vous faire cracker, espionner et piéger... 😀

Bon, tout cela pour dire que, afin que ce programme ("netfilter_cfg") marche, vous devez le lancer avec les droits root, les droits absolus sur votre machine quoi ... Vous devez donc pour cela avoir confiance dans mon programme .... Glups, pas évident, hein ? Vous savez, vous avez encore le temps d'aller jouer à Frozzen Bubble... 😀

Vous êtes toujours là ? Biennnnnnn !!!! Bon, merci de votre confiance. A priori je n'ai pas fait de bêtises en écrivant ce logiciel, et je l'utilise tous les jours depuis déjà quelques temps ... A défaut de vous rassurer, passons tout de suite à ce qui est important, à savoir : L'installer et le lancer ...

V-2 Installation

Commencez par vous logger en temps que root. Puis sauvez le fichier "netfilter_cfg" quelque part sur votre disque dur. Personnellement, le répertoire "/usr/local/sbin/" me semble le meilleur endroit. Ne vous soucier pas pour l'instant du fichier "netfilter_cfgd", nous aurons l'occasion d'en reparler un peu plus loin. Maintenant, donnez les droits d'exécution au fichier "netfilter_cfg" avec la commande "chmod".

Par exemple :

```
[root@phoenix /]# chmod +x /usr/local/sbin/netfilter_cfg
```

Il est de bon goût de donner la propriété et l'usage exclusif au root de ce fichier. Tapez donc les commandes suivantes :

```
[root@phoenix /]# chown root:root /usr/local/sbin/netfilter_cfg
[root@phoenix /]# chmod 750 /usr/local/sbin/netfilter_cfg
```

La première commande indique que ce fichier appartient au root. La seconde autorise uniquement le root à l'exécuter, et les autres utilisateurs ne peuvent même pas le lire ...

V-3 Droits d'utilisation

Comment lancer le programme ? En fait, il y a 4 méthodes différentes :

- Le plus logique est de lancer un terminal, dans lequel vous vous logger en root ("su -"), puis vous lancez le programme. Cela marche bien, mais c'est un tantinet lourd... Il y a plus simple quand même... Nous réserverons cette technique uniquement pour nos tests, puis nous utiliserons une technique plus évoluée.

  ```
  [olivier@phoenix /]$ su -
  [root@phoenix /]# /usr/local/sbin/netfilter_cfg
  ```

- Autre manière de faire : Vous donnez au programme les droits SUID. C'est une très très mauvaise idée , car c'est une technique non sécurisée , qui peut permettre à un petit malin de prendre les droits root sur votre machine. Mais c'est aussi une technique très simple ...

  Bon, si vous voulez réellement l'utiliser (ne venez pas vous plaindre après ...), faites ceci :

```
[root@phoenix /]# chown root:root /usr/local/sbin/netfilter_cfg
[root@phoenix /]# chmod 750 /usr/local/sbin/netfilter_cfg
[root@phoenix /]# chmod +S /usr/local/sbin/netfilter_cfg
```

- La technique du "sudo" est par contre beaucoup plus "propre". Grosso modo, vous lancez le programme via un autre programme, qui a lui-même hérité des droits root. Cela permet donc à n'importe quel utilisateur de lancer ce programme, et ce, de manière transparente

  Commencez par configurer "sudo" en lançant la commande "visudo" (cela édite et contrôle le fichier /etc/sudoers). Puis modifiiez la configuration comme ceci :

  <mon utilisateur> <ma machine>= NOPASSWD:/usr/local/sbin/netfilter_cfg
  où "<mon utilisateur>" est votre login
  et "<ma machine>" est le nom de votre machine

  Exemple :

  ```
  [root@phoenix /]# visudo
  olivier phoenix=NOPASSWD:/usr/local/sbin/netfilter_cfg
  ```

  Utiliser "man sudoers" et "man sudo" pour avoir plus d'informations sur la commande "sudo".

  Puis pour lancez le programme, au lieu de lancer "/usr/local/sbin/netfilter_cfg", vous devez lancer (ici en temps que simple utilisateur "olivier") :

  ```
  [olivier@phoenix /]$ /usr/bin/sudo /usr/local/sbin/netfilter_cfg"
  ```

  Pas trop compliqué, hein ?

- La dernière technique est celle que je recommande. Elle consiste à automatiser l'exécution de "netfilter_cfg", en le faisant lancé automatiquement lorsque c'est nécessaire. Nous verrons plus bas comment faire. Cependant avant d'automatiser cette opération, il convient de tester la configuration de "netfilter_cfg". Et pour cela, la 1ère méthode est de loin la plus efficace.

Bien, passons maintenant aux paramètres

V-4 Paramètres

Comme nous l'avons vu au-dessus, ce programme attend certains paramètres pour s'exécuter. La liste complète des paramètres s'obtient avec la commande : netfilter_cfg --help

Si il s'avère que vous lancez systématiquement "netfilter_cfg" avec les même paramètres, vous pouvez modifier le comportement par défaut du programme, en forçant la valeur de ces paramètres dans le script. Pour cela, modifiez la valeur par défaut des paramètres en modifiant le tableau WAITING_PARAMETERS[..], comme indiqué plus haut.

Explications des paramètres :

- --drop-rules <on|off> : Valeur par défaut : "on".
  Active les règles de rejet systématique. Les paquets répondant aux règles du tableau "DROP" (voir IV-5) vont donc êtres détruits systématiquement, sans même être loggé. Pour les premières utilisations de ce script, je vous conseille de de désactiver cette option ("off"), afin que vous puissiez vous rendre compte du trafic généré par des activités comme le P2P. Puis après, activez cette option (c'est ce qui est fait par défaut), afin de ne pas trop polluer vos logs de ce type de connexions.
- --spoofing-filter <on|off> : Valeur par défaut : "on".
  Cette option devrait être toujours activée. Elle interdit certains types de connexions qui sont aberrantes, et qui ont clairement le but de se faire passer pour autre chose qu'elles ne sont. En bref, ce sont des techniques de piratage très très classiques. A laisser sur "on", à

moins que vous ne cherchiez les coups...

- --nat <on|off> : Valeur par défaut : "off".
  Cette option active le NAT, et permet donc aux réseaux routables (ceux qui ont le mot "routable" dans le tableau "LAN[...]") de se connecter à Internet. Si vous n'avez pas de réseaux locaux, ou que vous ne voulez pas que ces réseaux accèdent à Internet via votre machine, laissez l'option par défaut ("off").

- --wait-connexion <on|off> : Valeur par défaut : "off".
  Ce paramètre est tout spécialement destiné aux utilisateurs de l'ADSL. Sous Linux, quand vous voulez démarrer votre connexion ADSL, vous lancez "adsl-start" (utilisation de "rp-pppoe") ou "startadsl" (utilisation "pppoa"). Or, afin d'assurer la re-configuration de netfilter_cfg, il convient de le lancer tout de suite après la connexion ADSL, par exemple en utilisant un mécanisme automatisé (voir plus loin). Le problème, c'est que certains modem mettent un certain temps à se connecter à Internet. Le mécanisme automatique s'exécute alors trop tôt, rendant inefficace "netfilter_cfg".

  La solution de ce problème est tout simplement de lancer "netfilter_cfg" avec le paramètre "--wait-connexion on". Ainsi, "netfilter_cfg" va attendre que la connexion ADSL ait terminé de s'établir, et il configurera proprement la machine.

  Si vous ne voulez pas utiliser le mécanisme automatique, utilisez alors les commandes suivantes afin de lancer dans la foulée votre connexion ADSL et "netfilter_cfg" :

  ```
  [root@phoenix /]# adsl-start && netfilter_cfg --wait-connexion on
  ```

  ou :

  ```
  [root@phoenix /]# startadsl && netfilter_cfg --wait-connexion on
  ```

- --silence <on|off> : Valeur par défaut : "off".
  Cette commande permet de supprimer tout les messages que netfilter_cfg affiche par défaut. Cependant, vous pouvez toujours les retrouver dans le fichier de log (par défaut : "/var/log/netfilter_cfg").

- --wan-ping <on|off> : Valeur par défaut : "off".
  A priori, personne sur Internet n'est supposé avoir besoin de tester votre présence grâce à la commande "ping". Cependant, il peut-être pratique d'autoriser votre machine à répondre aux "ping" lancés par des machines d'Internet, notamment dans le cas où vous avez un serveur (FTP, TO, etc...). Si ce n'est pas le cas, laissez l'option par défaut ("off").

- --wan-to-server <on|off> : Valeur par défaut : "off".
  Rend votre serveur Tactical Ops accessible depuis Internet. C'est sympa de votre part, mais vous avez intérêt à savoir ce que vous faites.
  Afin de paramétrer les options de ce service, se référer aux options vues précédemment .
  Très important : Si vous utilisez cette option, lisez la remarque ci-dessous.

- --wan-ftp-server <on|off> : Valeur par défaut : "off".
  Permet aux Internautes de se connecter sur votre serveur FTP. A moins que vous n'ayez de bonnes raison, vous n'êtes pas supposé avoir un serveur FTP.
  Afin de paramétrer les options de ce service, se référer aux options vues précédemment .
  Très important : Si vous utilisez cette option, lisez la remarque ci-dessous.

- --wan-http-server <on|off> : Valeur par défaut : "off".
  Autorise les internautes à accéder au serveur web (HTTP/HTTPS) tournant sur votre machine.
  Afin de paramétrer les options de ce service, se référer aux options vues précédemment .
  Très important : Si vous utilisez cette option, lisez la remarque ci-dessous.

- --wan-ssh-server <on|off> : Valeur par défaut : "off".
  Permet à un utilisateur se trouvant en dehors de votre réseau de travailler à distance sur votre machine, en utilisant le service SSH (Secure SHell). Il n'est pas très habituel de faire tourner un tel serveur sur sa machine, pensez donc vraiment à restreindre à un nombre

limité de comptes l'accès à ce service.
Afin de paramétrer les options de ce service, se référer aux options vues précédemment .
Très important : Si vous utilisez cette option, lisez la remarque ci-dessous.

- --wan-jabber-server <on|off> : Valeur par défaut : "off".
  Si vous faites tourner votre propre serveur Jabber (serveur de discussion, concurrent de ICQ, MSN Messenger, etc...) sur votre machine, vous devrez activer cette option afin de le rendre accessible à vos contacts externes.
  Afin de paramétrer les options de ce service, se référer aux options vues précédemment .
  Très important : Si vous utilisez cette option, lisez la remarque ci-dessous.

- --wan-gnomemeeting <on|off> : Valeur par défaut : "off".
  Cette option vous permettra de discuter avec vos correspondants en utilisant le logiciel GnomeMeeting.
  Afin de paramétrer les options de ce service, se référer aux options vues précédemment .
  Très important : Si vous utilisez cette option, lisez la remarque ci-dessous.

- --wan-xmule <on|off> : Valeur par défaut : "off".
  Si vous utiliser des logiciels de Peer-to-Peer ("P2P" : partage de fichiers) comme xMule, et que vous ne voulez pas avoir de problème de "LowID", vous devrez utiliser cette option. Cette option est aussi utilisable avec les autres logiciels de P2P du réseau "eMule", comme "amule". Vous pouvez l'adapter très facilement à d'autres réseaux P2P utilisant un principe similaire, en changeant les paramètres XMULE_*_PORT.
  Afin de paramétrer les options de ce service, se référer aux options vues précédemment .
  Très important : Si vous utilisez cette option, lisez la remarque ci-dessous

- --wan-bittorrent <on|off> : Valeur par défaut : "off".
  BitTorrent est un autre logiciel de P2P, utilisant par contre une technique très différente de celle du réseau eMule et de ses concurrents. Il est bien connu sous Linux, et d'ailleurs la distribution Linux Mandrake l'utilise pour distribuer ses nouvelles versions.
  Afin de paramétrer les options de ce service, se référer aux options vues précédemment .
  Très important : Si vous utilisez cette option, lisez la remarque ci-dessous.

- --lan-dhcp-server <on|off> : Valeur par défaut : "off".
  Si vous faites tourner sur votre machine un serveur DHCP (un serveur de distribution d'adresses IP pour les machines de votre votre réseau local), vous devrez utiliser cette option. Sans quoi, les machines de votre réseau ne pourront pas interroger votre serveur DHCP, et elles ne pourront pas recevoir d'adresses IP. Pour utiliser cette option, il vous faudra aussi indiquer quelle(s) interface(s) réseau(x) est(sont) autorisé(s) à repondre aux requêtes DHCP. Pour cela, rajoutez l'option "dhcp_servers" dans le tableau LAN[..] tel que vu précédement .

Remarque importante. Ce n'est pas parce que vous avez un firewall que vous êtes protégés contre toutes tentatives d'intrusion. Et notamment si vous exécutez sur votre machine un "service" (serveur ou un client "actif"), le firewall ne pourra pas sécuriser le dit service ! En effet en utilisant les options "--wan-*" vues ci-dessus, vous explicitez très clairement au firewall de laisser passer le flux de données vers ces services tournant sur votre machine. Donc c'est au service lui-même d'assurer sa propre protection... A vous donc de lire à fond la documentation qui est associé à ce service !

Comme vous pouvez le voir, le nombre d'options est assez limité (j'ai voulu faire un programme simple à utiliser ...), et pour la plupart des utilisations, il suffit de lancer "/usr/local/sbin/netfilter_cfg" sans paramètre pour avoir une protection optimum...

Les grincheux pourraient dire qu'activer le NAT par défaut aurait été une bonne chose. Sans doute chez eux, personnellement je préfère décider de quand j'autorise les machines de mon réseau local à sortir sur Internet... Mais vous pouvez vous-même mettre par défaut cette option à "on" si vous voulez  .

Maintenant que vous savez quelles options sont à utiliser, vous pouvez executer votre firewall, et (enfin) sécuriser votre machine...

V-5 Exécution

Sur beaucoup de sites webs, vous trouverez des scripts semblables à "netfilter_cfg" (mais moins complexe, et sans paramètres). Et notamment, on vous parlera de la commande "/etc/init.d/iptables" et des ses options "start", "stop", "save", etc...

Les développeurs de ces scripts utilisent une technique particulière qui consiste à définir une fois pour toute la configuration de Netfilter, et à sauver les différentes commandes "iptables" utilisées grâce à la commande " /etc/init.d/iptables save". L'avantage de cette technique est qu'au prochain démarrage de la machine, le firewall sera automatiquement configuré, et qu'il n'y aura plus rien à faire. Cette technique est très séduisante, mais elle a un problème. En effet, à moins de posséder une adresse Internet statique, les règles iptables de ces scripts ne peuvent pas utiliser l'adresse IP Internet de la machine comme filtre, pour la simple raison que la connexion Internet n'est pas encore lancé, où que celle-ci à changé suite à la dernière déconnexion du fournisseur d'accès.

Par exemple, les règles de filtrage des connexions sortantes Internet que l'on pourrait avoir serait :

iptables -A OUTPUT -o ppp0 -p all -m state --state ! INVALID -j ACCEPT
iptables -A INPUT -i ppp0 -p all -m state --state RELATED,ESTABLISHED -j ACCEPT

Personnellement, je pense être un peu trop paranoïaque pour laisser des règles un peu trop souple de ce type. C'est pourquoi lorsque "netfilter_cfg" s'exécute, il détecte automatiquement la présence ou non de la connexion à Internet, et récupère l'adresse IP qui a été fournit par le fournisseur d'accès. Ainsi, si mon adresse IP est 26.87.145.23, mes règles seront automatiquement définies par :

iptables -A OUTPUT -o ppp0 **-s 26.87.145.23** -p all -m state --state ! INVALID -j ACCEPT
iptables -A INPUT -i ppp0 **-d 26.87.145.23** -p all -m state --state RELATED,ESTABLISHED -j ACCEPT

Contrairement aux 2 précédentes règles :

- Ma première règle interdit à un programme tournant sur ma machine de se faire passer pour une autre machine d'Internet. Je ne suis pas un "vilain pirate", donc ma machine n'a pas à avoir un tel comportement.
- Ma seconde règle interdit à une machine externe de se faire passer pour quelqu'un d'autre que ce qu'elle prêtent être. C'est un moyen supplémentaire d'éviter d'être piraté.

Bon, il commence à être tard, et vous vous demandez où je veux en venir, non ??? 

Le but de cette explication n'était pas de montrer que mon script est peut-être meilleur (ou pas ...) que les autres, mais uniquement de vous faire comprendre à quel moment il faut le lancer ... Car, à n'en point douter, vous avez compris que pour ce que ce programme marche, il doit être lancé une fois que la connexion Internet est activée. Car ce n'est qu'à ce moment là que votre adresse IP Internet est connue...

4 cas sont donc possibles :

- La méthode que je recommande est celle que nous avons déjà évoqué ici et là. Il s'agit tout simplement de laisser votre Linux lancer "netfilter_cfg" au démarrage et à l'arrêt de votre connexion Internet. Pour cela, nous allons utiliser une fonctionnalité de "PPP", le logiciel responsable de votre connexion à Internet.

  En effet, au démarrage (respectivement à l'arrêt) de ce programme, Linux lance automatiquement le script "/etc/ppp/ip-up.local" (respectivement "/etc/ppp/ip-down.local"). Il nous suffit donc de créer ces 2 petits scripts qui lanceront systématiquement "netfilter_cfg", afin qu'il prenne en compte l'activation et l'arrêt de la connexion Internet.

  Par exemple, ou pourra écrire les scripts ci-dessous :

  - Pour l'établissement de la connexion Internet (fichier "/etc/ppp/ip-up.local") :

```
[root@phoenix /]# cat /etc/ppp/ip-up.local
#!/bin/sh -norc
```

```
#
/usr/local/sbin/netfilter_cfg --wait-connexion on --silence on
```

- A l'arrêt de la connexion Internet (fichier "/etc/ppp/ip-down.local") :

```
[root@phoenix /]# cat /etc/ppp/ip-down.local
#!/bin/sh -norc
#
/usr/local/sbin/netfilter_cfg --silence on
```

Et il ne faudra pas oublier de les rendre exécutables :

```
[root@phoenix /]# chmod 744 /etc/ppp/ip-up.local
[root@phoenix /]# chmod 744 /etc/ppp/ip-down.local
```

Ces deux scripts seront automatiquement exécutés lors de l'utilisation d'une connexion :

- utilisant un modem téléphonique classique (connexion RTC).
- via un modem ADSL (dégroupé ou non), connecté à la machine par un câble USB (utilisation du pilote Eagle-USB)
- via un modem ADSL non dégroupée , connecté à la machine par une connexion Ethernet (utilisation du programme rp-ppoe).

Si vous avez la chance d'avoir à la fois un modem Ethernet et une connexion ADSL dégroupée (à l'heure où j'écris ces lignes, en France seul le fournisseur d'accès à Internet Free le propose), vous n'avez pas besoin d'utiliser "PPP" pour vous connecter à Internet. La connexion se fait en effet automatiquement en utilisant une requête DHCP. Il faut faudra alors utiliser une autre méthode pour lancer automatiquement "netfilter_cfg", et c'est ce que nous allons voir tout de suite.

- La 2nd méthode que je propose , est de démarrer "netfilter_cfg" lors de la phase de boot, c'est à dire lors de l'exécution des scripts d'inits. Ces scripts sont ceux qui s'affichent probablement au démarrage de votre machine, tout en écrivant à l'écran une série de lignes "OK" ou "ERROR".

  Pour cela, j'ai écris un autre (petit) script, " netfilter_cfgd" (notez bien le "d" à la fin du nom) que vous allez installer sur votre machine, qui se lancera automatiquement, lors de la phase de boot de la machine :

  - Copiez le fichier "netfilter_cfgd" dans votre répertoire "/etc/init.d/"
  - Donne au root la propriété de ce fichier, ainsi que les droits d'exécution :

```
[root@phoenix /]# chown root:root /etc/init.d/netfilter_cfgd
[root@phoenix /]# chmod 744 /etc/init.d/netfilter_cfgd
```

  - Puis configurer votre Linux pour que "netfilter_cfgd" soit lancé au démarrage :

```
[root@phoenix /]# chkconfig --add netfilter_cfgd
```

  - Et vérifiez que le script sera bien lancé au prochain démarrage de la machine :

```
[root@phoenix /]# ls -la /etc/rc*.d/ | grep net
lrwxrwxrwx    1 root     root           24 mar  9 22:28 S03net
lrwxrwxrwx    1 root     root           17 oct 18 16:06 S10net
lrwxrwxrwx    1 root     root           15 oct 18 16:06 S25net
lrwxrwxrwx    1 root     root           16 jan 10 00:36 S56xin
```

Que vous utilisiez une connexion RTC ou ADSL, dégroupé ou non, il est intéressant de lancer quand même "netfilter_cfgd" comme ceci au démarrage de la machine. Ainsi, même si votre connexion Internet n'est pas activée tout de suite, votre firewall sera quand

même opérationnel. Et si "netfilter_cfg" n'est pas démarré après votre connexion ADSL, comme vu avec la technique ci-dessus, au moins votre machine sera protégée. Vous n'aurez certes pas accès à Internet (Netfilter bloquera toutes vos connexions), mais au moins personne de l'extérieur ne pourra non plus accéder à votre machine.

Il reste deux cas que nous n'avons pas encore traité, il s'agit des cas :

- d'une connexion ADSL dégroupé , utilisant une connexion ethernet entre le PC et le modem.
- d'une connexion internet, utilisant un routeur.

Dans ces deux cas, on ne peut pas lancer "netfilter_cfg" :

- avant l'activation de la connexion réseau/Internet, car le programme ne pourra pas trouver la passerelle, et il se croira dans un réseau local, isolé de connexions externes.
- automatiquement après la connexion à Internet, car le mécanisme automatique fournit par PPP vu ci-dessus n'est pas utilisé.

Alors, mission impossible ? Non la solution existe, et elle est simple. Il suffit de lancer "netfilter_cfgd" plus tard, après que la configuration réseau soit complètement activée.

Pour cela, il faut éditer "/etc/init.d/netfilter_cfgd", et modifier la ligne configurant "chkconfig" :

```
[root@phoenix /]# less /etc/init.d/netfilter_cfgd
#!/bin/sh
#
########################################################################
# This is file /etc/rc.d/init.d/netfilter_cfgd
#
# chkconfig: 2345 03 97
#
....
```

Chkconfig est le mécanisme qui permet de gérer l'ordre de démarrage des scripts d'inits (lancez "man chkconfig" pour des informations plus détaillées sur ce mécanisme) :

- "2345" signifie que le script "netfilter_cfgd" sera lancé pour les runlevel 2, 3, 4, et 5 (c'est standard comme configuration).
- "03" signifie que pour les runlevel de démarrage de la machine (mode graphique ou texte), le script sera un des premiers à être lancé.
- "97" signifie que pour les runlevel d'arrêt de la machine (extinction ou reboot), le script sera un des derniers à être arrêté.

Bref, pour retarder le démarrage de "netfilter_cfgd", il faudra augmenter le chiffres "03" et réduire le "97" , en respectant une règle simple : la somme des deux chiffres doit être égale à 100 .

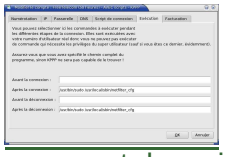
Concrètement, pour une distribution Linux Mandrake le script de démarrage du réseau ("/etc/init.d/network") démarre en position "11". Donc il faudra démarrer "netfilter_cfgd" un peu plus tard, en position "12" par exemple. Dans ce cas, on écrira :

```
[root@phoenix /]# vi /etc/init.d/netfilter_cfgd
#!/bin/sh
#
########################################################################
# This is file /etc/rc.d/init.d/netfilter_cfgd
#
# chkconfig: 2345 12 88
#
....
```

Et on n'oubliera pas de lancer "chkconfig", afin que la modification soit prise en compte :

```
[root@phoenix /]# chkconfig --add netfilter_cfgd
```

- 3ème technique, vous vous connectez par modem. Supposons que vous utilisiez un logiciel graphique pour vous connecter, comme "KPPP" par exemple, vous pouvez demander à ce logiciel de lancer automatiquement "/usr/bin/sudo /usr/local/sbin/netfilter_cfg" après la connexion à Internet. De même que vous pouvez aussi le lancer après la déconnexion , afin que votre firewall ne soit pas ennuyé par une interface "ppp0" n'existant plus.


KPPP avec lancement du script de firewall

  Utilisez pour cela dans "KPPP", l'onglet "Exécution" de chacun de votre compte Internet.

  Bien évidement, pour que cela marche, il faut que vous ayez configurer le mécanisme de "sudo" pour que vous puissiez lancer "netfilter_cfg" avec les droits root. Tout ceci étant déjà explique un peu plus haut.

- Dernière solution : Vous voulez vraiment vous faire suer, et lancer "netfilter_cfg" manuellement. C'est tout particulièrement pratique pour les phases de tests et de configuration de "netfilter_cfg", mais à terme c'est assez lourd à gérer. Mais si vous êtes masochiste à ce point, l'explication a déjà été donné ici, ici et là.

Bon, récapitulons un peu ce que nous venons de voir ci-dessus :

Type de connexion	Programme de connexion Internet	Technique à utiliser
Modem RTC	PPP/KPPP (ou autres)	Méthode 1 Netfilter_cfgd et ip-up.local && ip-down.local
		Méthode 2 Netfilter_cfgd et Lancement de scripts via KPPP et sudo
Modem ADSL non dégroupé & connexion USB	startadsl	Netfilter_cfgd et ip-up.local && ip-down.local
Modem ADSL non dégroupé & connexion ethernet	adsl-start	Netfilter_cfgd et ip-up.local && ip-down.local
Modem ADSL dégroupé & connexion USB	startadsl	Netfilter_cfgd et ip-up.local && ip-down.local
Modem ADSL dégroupé & connexion ethernet	Aucun	Netfilter_cfgd en modifiant le niveau d'init
Modem-routeur ADSL ou routeur simple	Aucun	Netfilter_cfgd en modifiant le niveau d'init

V-6 Test

Pour tout ce qui est des tests d'intrusion, afin de vérifier la validité de votre firewall, le programme le plus intéressant qu'il existe est "nmap", qui est un scannair de ports. Ce programme peut être utilisé pour lancer des connexions sur une machine d'un réseau, afin de voir quels sont les ports TCP/IP qui sont ouverts.

Vous pourriez l'utiliser pour tester vous même la qualité de votre firewall sur votre adresse IP Internet par exemple, mais malheureusement cela ne peut pas marcher. En effet, dans ce cas précis le kernel Linux comprend que l'expéditeur et le récepteur de la communication sont une seule et même machine (lui-même), et donc il fait passer la communication via l'interface loopback (même si "nmap" affichera votre adresse IP Internet). Or, les filtrages de cet interfaces lui sont spécifiques, et sans rapport avec ceux de votre connexion Internet...

Vous êtes déçu ? Nannn, il ne faut pas. Il existe 2 méthodes très efficaces pour tester malgré

tout votre configuration de netfilter :

- Demandez à un ami qui utilise Linux ou Windows® (nmap fonctionne dans les 2 environnements), de lancer la commande "nmap" depuis sa machine sur votre adresse IP. Tachez de bien lui donner votre adresse IP, et pas celle d'un site web militaire ou gouvernemental, car le cas échéant, la personne visé peut ne pas du tout apprécier. L'utilisation de "nmap" est considéré comme étant plutôt agressive par les administrateurs, voir à la limite de la tentative d'intrusion dans leurs systèmes.

- Si vous n'avez pas d'amis, vous pouvez toujours essayer les tests d'intrusion de PcFlank (Menu "Quick test", "Stealth test", etc ...), ou d'un site équivalent (voir à ce sujet cette autre documentation que j'ai écrit). Cela donne des résultats intéressants, mais ce type de site ne testent pas tout les ports IP, ni toutes les astuces de "port scanning". Donc ce n'est pas un test très complet...

Conclusion : Faites vous un ami !!!!

VI Conclusion

Voila, je pense avoir fait le tour de la question sur la philosophie, le paramétrage et l'utilisation de ce programme. Ce programme est tout particulièrement adapté dans le cas d'une connexion via modem RTC ou ADSL non permanent, en se lançant après le démarrage du démon ppp ou des programmes de connexion ADSL. Il peut aussi s'utiliser dans un réseau local, possédant une passerelle (ou "gateway" en anglais) pour la connexion à Internet.

Hormis la phase de configuration, qui peut sembler un brin ardus, l'utilisation de ce programme est très aisé, et le lancement sans aucun paramètre suffit à protéger votre machine dans la majeure partie des cas.

Bien évidement, je suis ouvert à toutes suggestions et toutes questions quand à son utilisation, son développement, ou à sa documentation.

VII Remerciements

Merci à Jean-Roch, Fouad, François, Keyvan et Serge pour avoir participé aux tests et à la mise en production.

Merci aussi à Antoine BUSCH, Christian CHANUEL, Jak HIGHLANDER, Francois-Xavier 'FiX' KOWALSKI, JP SYLVANIE, pour leurs idées, patchs et ajouts de fonctionnalités

Merci enfin aux futurs utilisateurs qui vont me renvoyer plein de retours d'expérience...

VIII Historique du document

Version de Netfilter_cfg	Date	Remarque
Version 0.5.8	2004/03/11	Ajout du support du client BitTorrent. Ajout du paramètre "--lan-dhcp-server" afin autoriser les broadcasts DHCP du LAN. L'option "--wan-xmule-server" est renommée en "--wan-xmule". Correction du "wan-gateway", et d'un possible port scanning de la part de la passerelle.
Version 0.5.7	2004/03/03	Ajout du support du client GnomeMeeting.
Version 0.5.6	2004/02/29	Ajout de filtrages plus restrictifs pour l'ADSL dégroupé, et corrections pour le "wan-gateway". Ajout du support du client xMule. Corrections sur le chargement des modules. Ajout du paramètre "--silence". Correction du problèmes des adresses IP multiples sur une

		interface réseau.
Version 0.5.5	2003/11/02	Correction des règles anti-spoofing. Ajout du "--wait-connexion". Amélioration du "DebugWrite". Ajout du support du support des serveurs HTTP, Jabber, SSH. Correction dans les règles ULOG.
Version 0.5.4	2003/06/17	Première version publique, documentation au format XHTML.
Version 0.5.2	2003/06/11	Première version semi-publique, documentation en format texte.

Remarques :

- La version 0.5.8 sera sans doute la dernière version écrite en langage "sh". La prochaine version prévue (numérotation > 0.7.0) sera très probablement écrite en Perl, afin de régler les problèmes de performance du script actuel. Elle sera surtout l'occasion de changer complètement la technique de filtrage (utilisation de tables utilisateurs, filtrage des connexions sortantes, séparation entre les paramètres et le code, etc...), afin de rendre Netfilter lui-même plus rapide et efficace.
- Les versions prévues entre la 1.0.0 et la 2.0.0 seront sans doute écrites en C++. Les ambitions sont très nombreuses, trop sans doute pour être déjà évoquées...

Nombre de visiteurs: 

Olivier Allard-Jacquin

Site de référence : http://olivieraj.free.fr/

Last modified: Thu Mar 11 22:21:27 CET 2004