

Compte Rendu de la formation GNU/LINUX : SERVICES ET SECURITE RESEAUX.

Auteur Laurent RAYSSIGUIER

Informations pratiques diverses sur les versions 7.x de REDHAT :

- Depuis la version 7.0, REDHAT crée un alias pour la commande cp.
Si on utilise cp, on exécute en fait son alias "cp -i" qui demande confirmation en cas d'écrasement d'un fichier existant (mode interactif).
Il en est de même avec la commande mv.
Pour visualiser cela, taper "cp" puis la combinaison de touches "**ctrl + alt + e**", ce qui complètera la commande.
Cette combinaison peut aussi être utile pour un script en ligne de commande.
Par exemple, si on tape "for test in `ls /root` do" puis "ctrl + alt + e", cela va afficher l'intégralité du résultat de ls /root .
Cela permet de visualiser à l'avance ce que l'on va obtenir.
Pour solutionner le problème de l'alias, il faut soit le supprimer (et faire très attention par la suite) soit utiliser le nom complet de la commande cp ou mv (conseillé dans un script).

- La commande "**tune2fs**" permet de modifier le délai définissant le nombre de montage maximum d'une partition entre deux e2fsck.
De même, cette commande permet de passer du ext2fs en ext3fs.
Pour cela, compiler le noyau avec le ext3fs activé et mettre à jour tune2fs et fdisk.
Créer une partition ext2fs (par exemple sur /dev/hdb1) et lancer la commande :
"tune2fs -j /dev/hdb1"

- Par commodité, activer le **pavé numérique** au démarrage du PC.
Pour se faire, éditer le fichier **/etc/rc.d/rc.local** et ajouter les lignes suivantes en fin de script.

```
# Début de script pour l'activation du pavé numérique
INITTY=/dev/tty[1-8]
    for tty in $INITTY; do
        settleds -D +num < $tty
    done
# Fin de script pour l'activation du pavé numérique
```

BASES à connaître sur la configuration réseau de GNU/LINUX :

On peut dire que la configuration d'une machine linux est "propre" si les commandes affichent les informations prévues.

En effet, Linux est fait pour donner ses informations réseaux d'une certaine manière avec les commandes détaillées ci-dessous.

Prenons le cas d'une machine dont l'adresse IP est 192.168.1.1 et le nom complet station.linux-tarn.org

La commande "**hostname**" doit répondre le nom court de la machine, soit ici "station".
La commande "**hostname -f**" doit répondre le nom long de la machine, soit ici "station.linux-tarn.org".

Pour se faire, certains fichiers doivent respecter certaines contraintes.

La variable "HOSTNAME" est fixée au BOOT du PC par le script `/etc/rc.d/rc.sysinit`, qui lit sa configuration dans le fichier `/etc/sysconfig/network` dans la variable `"HOSTNAME=nom_de_la_machine"`.

Le fichier `/etc/hosts` doit avoir la forme suivante :

```
127.0.0.1    localhost.localdomain    localhost
192.168.1.1  station.linux-tarn.org    station
```

Toutes les interfaces réseau de la machine doivent y être référencées.

Le fichier `/etc/resolv.conf` qui donne les informations traitant des serveurs de noms a la forme suivante :

```
search      linux-tarn.org           / Nom de domaine de recherche par défaut
nameserver  213.190.70.1           / adresse ip du serveur Dns primaire
nameserver  X.X.X.X                / adresse ip du serveur Dns secondaire, etc...
```

Une fois que la configuration réseau est correcte, il faut penser à **tenir sa distribution à jour**. L'idéal est de **s'inscrire à la liste de distribution de REDHAT** sur leur site web.

De même, il faut se tenir informé des mises à jour de la distribution qu'on utilise et télécharger les mises à jour nécessaires.

Pour cela, aller sur <http://www.redhat.com> , puis choisir "Download" et enfin cliquer sur "Errata and Bug Fixes".

Les mises à jour sont classées par le numéro de version des distributions REDHAT.

Les principaux groupes de paquets à veiller à mettre à jour sont :

kernel, glibc, netfilter, modutils, + les paquetages des services réellement utilisés sur la machine.

Pour ce qui est du kernel, **il n'est pas nécessaire de rechercher le tout dernier noyau du site officiel www.kernel.org , mais il faut télécharger celui de REDHAT.**

La version du noyau sur le site REDHAT peut être d'un numéro de version inférieure, mais qui est gardé à jour, patché et testé pour la version REDHAT de Linux.

Ce noyau est généralement bien mieux adapté que la version officielle.

Le noyau LINUX est prévu pour se défendre contre certaines attaques dès sa couche IP.

Protection contre les DOS (deny of service) :

Mettre à 1 la valeur de `/pros/sys/net/ipv4/tcp_syncookies` par la commande :

```
echo 1 > /pros/sys/net/ipv4/tcp_syncookies
```

Ceci permet que le serveur ne soit pas "planté" en cas de multiples requêtes SYN par un assaillant qui veut saturer la machine.

Protection contre l'IP SPOOFING :

Cette protection est déconseillée sur l'interface locale de la passerelle, mais fortement conseillée sur l'interface accessible de l'internet.

```
echo 1 > /proc/sys/net/ipv4/conf/interface_public/rp_filter
```

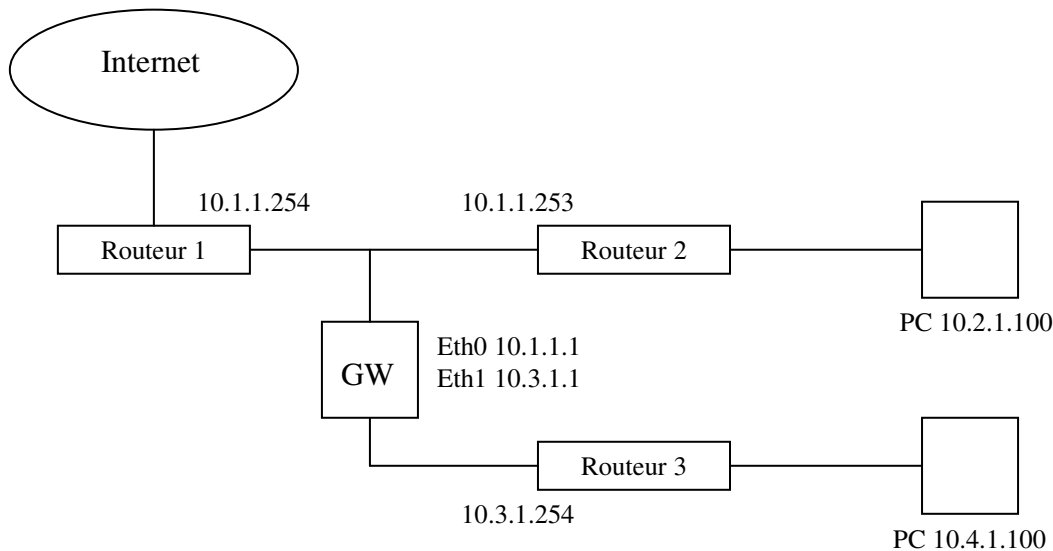
interface_public est ici le nom de la carte réseau connectée au réseau public.
Cette valeur peut être **all** (pour mettre la règle sur toutes les interfaces), **eth0** (pour la première), **eth1** (pour la deuxième), **eth....** (etc...).

Ce genre d'informations sur les possibilités du noyau peuvent se trouver dans la documentation fournie avec les sources dans **/usr/src/linux/Documentation**.
Les protections citées ci-dessus doivent se lancer dans un script de démarrage au choix de l'utilisateur.

Cela peut être **/etc/rc.d/rc.sysinit**, **/etc/rc.d/init.d/network** (modifié pour cela), **/etc/rc.d/init.d/iptables** (qui sera modifié pour cela dans le cas d'un Firewall).

Routing IP et modifications de celui-ci :

Prenons le cas suivant :



Pour accéder à partir de la machine GW à tous les réseaux locaux et à internet, on est obligé d'utiliser une table de routage. La définition de la passerelle par défaut ne suffit pas.

Définition d'une route :

route add , ajoute la route
destination , donne l'adresse du réseau (-net) ou **default** pour la route par défaut
gw adresse , donne l'adresse ip de la passerelle
ethx , donne le nom de l'interface réseau utilisée pour accéder à cette passerelle.

Soit ici :

1- Accès à internet.

Définir la route par défaut vers la passerelle internet :

```
route add default gw 10.1.1.254 eth0
```

2- Accès au réseau 2.

```
route add -net 10.2.1.0/8 gw 10.1.1.253 eth0
```

3- Accès au réseau 4.

```
route add -net 10.4.1.0/8 gw 10.3.1.254 eth1
```

Pour supprimer une route :

```
route del default
```

```
route del -net 10.4.1.0/8
```

Afficher les routes :

```
route -n
```

/ Evite la résolution de noms qui sinon bloque l'affichage, le temps d'avoir une réponse DNS.

Modification dynamique d'une adresse IP:

```
ifconfig eth0 192.168.1.1
```

Visualisation des adresse IP :

```
ifconfig
```

/Affiche les adresses IP des interfaces chargées

```
ifconfig -a
```

/Affiche les adresses IP de toutes les interfaces configurées (même non chargées)

```
ifconfig eth0
```

/Affiche l'adresse ip d'eth0 uniquement

DIVERS :

La commande hping -1 192.168.1.1 va faire un ping sur 192.168.1.1 et afficher des infos intéressantes pour déterminer le type de machine qui répond.

En effet, il apparaît l'information "id=xxxx" dans la réponse.

Suivant le comportement de cet id, on peut déterminer quelle couche IP répond et donc avoir une idée du système rencontré.

Par exemple, un linux kernel 2.4 incrémente de 1 cet id à chaque réponse, alors qu'un kernel 2.2 l'incrémente différemment.

Cependant, comme cette valeur "id" est prévisible quel que soit le système d'exploitation, un hacker peut scruter les ports ouverts d'une machine en se masquant derrière l'adresse IP d'une autre.

En effet, imaginons :

Un pirate cherche à savoir quels ports sont ouverts sur la machine A.

Il fait un hping sur une machine B qui lui répond avec un id s'incrémentant régulièrement de 1.

Puis il fait un scan de port sur la machine A en lançant des requêtes SYN de demande de connexion et masque son adresse par celle de B.

Quand le port de A est ouvert, la machine A répond à B par un ACK.

La machine B ne comprend pas pourquoi un ACK lui est envoyé et rejette le paquet.

Le fait de recevoir le ACK sur son interface réseau suffit à B pour retarder sa réponse à la machine du hacker, ce qui donne par exemple :

```
id=1201
```

```
id=1202
```

id=1203

id=1206

id=1207 etc...

Le hacker n'a donc plus qu'à vérifier quel port il interrogeait au moment où B a cessé de répondre, pour savoir que le port en question est ouvert sur A.

Sur A, si l'administrateur regarde ses logs, il va voir une activité suspecte venant de B et le hacker est tranquille 😊 puisque ce n'est pas sa véritable adresse.

Information Utile :

La société de formation utilise les antivirus TREND MICRO chez ses clients.

TREND MICRO est un éditeur très actif qui propose les produits suivants :

- OFFICE SCAN pour les stations clientes Windows
- INTERSCAN VIRUSWALL pour la passerelle GNU/LINUX (permet de scruter le trafic FTP, HTTP et SMTP)

FILTRE FIREWALL NETFILTER

Netfilter est le Firewall des systèmes GNU/LINUX à noyau 2.4.

C'est une très nette amélioration par rapport à ipchains (le Firewall des noyaux 2.2).

Netfilter est capable, contrairement à ipchains, de connaître l'état d'une connexion réseau. Il sait donc si le trafic qu'il reçoit est une demande de connexion ou une réponse liée à une connexion établie précédemment.

Cela lui permet de détecter des connexions invalides qui n'ont pas été établies de manière "normale".

Cela simplifie énormément les règles à donner au filtre, alors qu'ipchains imposait de penser à tout trafic entrant ou sortant, et compliquait donc l'établissement des règles.

De plus, ipchains imposait un mode passif pour le FTP, compliquant la configuration des clients.

Cette limitation n'a plus cours avec Netfilter qui accepte le FTP actif (ces principes FTP seront expliqués plus loin).

Pour utiliser le filtre Netfilter, il faut compiler le noyau sur une machine possédant souvent deux cartes réseau, car on l'utilisera en tant que passerelle internet.

Le fait de compiler le noyau permet d'activer ce filtre, mais aussi d'optimiser le noyau pour sa machine, et ainsi gagner en performance du système.

- S'assurer que l'on possède bien la dernière version du noyau proposée par REDHAT et que la commande iptables n'a pas de version plus récente sur leur site web.

- Installer les **kernel-headers** et **kernel-sources** en paquetages RPM.

- Aller dans le dossier **/usr/src** et créer un lien symbolique appelé linux sur le dossier du numéro de version du noyau installé par les RPMS.

ln -s /usr/src/linux_version /usr/src/linux

Remarque: Si on compile un noyau déjà installé sur la machine, il faut éditer le fichier Makefile du dossier et modifier le nom de la version. Le paquetage REDHAT met **numero_de_version_custom** par défaut. Cela évite que linux se perde dans les dossiers de modules.

A savoir : la commande '**make distclean**' supprime les fichiers objets ou temporaires issus d'une précédente compilation ou configuration du noyau (valable pour toute autre compilation de programme).

- Configurer le noyau avec '**make xconfig**' sous X ou '**make menuconfig**' en mode console.

- Passer les menus les uns après les autres pour choisir les options souhaitées, en pensant à bien mettre en dur dans le noyau le 'kernel module automounter' et le pilote pour les disques durs du système (IDE ou Contrôleur SCSI).

Mettre bien sûr le support de Netfilter...

Il faudra bien vérifier que l'on active les modules NAT et FTP CONNTRACK qui seront nécessaires pour le FTP et la translation d'adresse.

De plus il y a une option TCPMSS à sélectionner.

Cette fonctionnalité permet au noyau de connaître la taille maxi des trames qu'il peut envoyer (MTU). Si cette fonctionnalité n'est pas en place cela peut créer des problèmes.

Il faudra par la suite accepter ce trafic TCPMSS dans Netfilter dans le cas d'une connexion ADSL par la commande :

iptables -A FORWARD -p tcp --tcp-flags SYN,RST SYN -j TCPMSS --clamp-mss-to-pmtu

Si on ne l'active pas, on risque d'avoir une connexion ADSL qui résout bien les noms, mais qui est incapable de télécharger des fichiers ou des fichiers sur le Web.

- Une fois le noyau configuré, on passe à la compilation par les séquences suivantes :

make dep && make clean && make bzImage

L'option && exécute la commande qui la suit si celle qui la précède se passe correctement, sinon elle interrompt la séquence.

- Une fois que le noyau compilé correctement, il faut le placer sur la partition /boot :

cp /usr/src/linux/arc/i386/boot/bzImage /boot/vmlinuz-version

- Exécuter la compilation des modules et les mettre en place par la séquence suivante :

make modules && make modules_install

- Remplacer le fichier /boot/System.map installé par le noyau précédent par celui du nouveau noyau.

cp -f /usr/src/linux/System.map /boot

- Editer **/etc/lilo.conf**, copier le paragraphe démarrant le noyau précédent, supprimer la ligne définissant le fichier initrd car on ne l'utilise pas, remplacer le nom de l'ancien noyau par **/boot/vmlinuz-version** et enfin, modifiez l'alias de l'ancien noyau en ajoutant par exemple **-old** à la fin.

- Enfin exécuter la commande **lilo** qui, si tout va bien, va vous afficher la liste des options de démarrage avec un symbole * devant l'option par défaut.

- Rebooter le PC et prier pour que tout se passe bien ;-))

Remarque : Quand on compile le noyau sur un serveur, on peut choisir de ne pas installer le support pour l'USB dans la mesure où cela n'est pas nécessaire sur ce genre de machine. Cependant, lors du boot, l'USB étant tout de même détecté, l'ordinateur affiche un message d'erreur (non bloquant) au démarrage.

Pour corriger cela, deux solutions :

1- Au prompt de LILO, taper '**linux nousb**'.

2- Dans le fichier `/etc/lilo.conf`, se placer dans la rubrique du nouveau noyau et ajouter la ligne suivante :
append="nousb"

Netfilter est maintenant activé.

Netfilter est composé de 3 chaînes par défaut, auxquelles on pourra, si nécessaire, ajouter des chaînes supplémentaires dites "utilisateur" qui peuvent être utilisées pour regrouper des tests répétitifs.

Ces chaînes sont :

- filter, qui applique les règles de filtrage que nous définiront.
- nat, qui applique les règles de translation d'adresse éventuelles.
- mangle, qui permet de travailler les trames traitées.

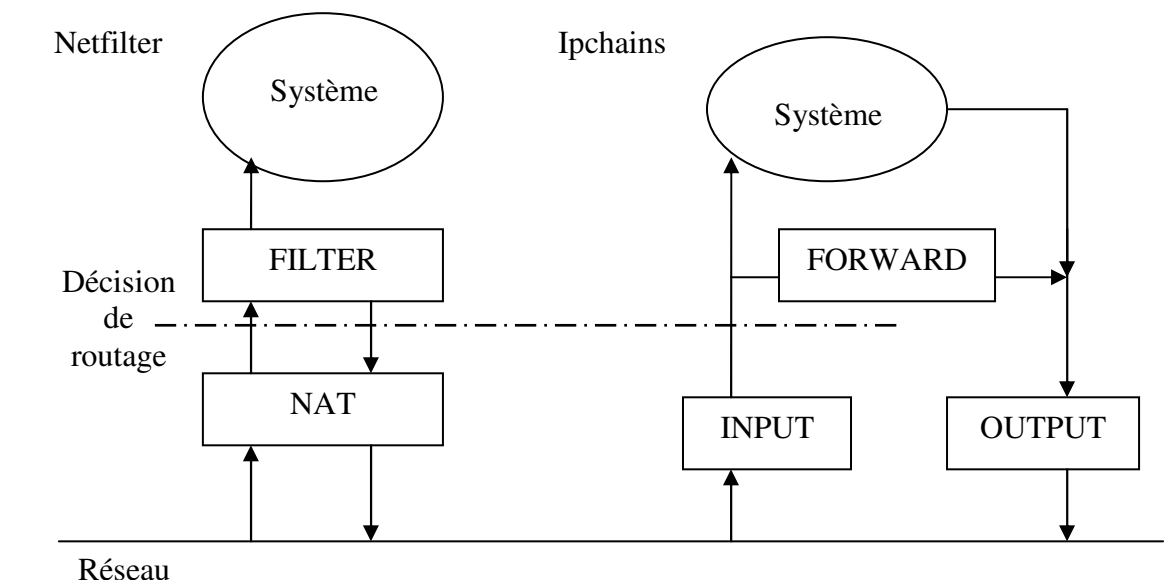
Nous n'utiliserons pas cette dernière, car elle est utilisée dans des cas peu fréquents.

Les actions à faire sur les trames traitées par Netfilter peuvent être appelées cibles. Les cibles **DROP** ou **REJECT** sont les deux actions possibles pour rejeter un paquet. Cela peut a priori sembler redondant sur le principe, mais il existe en fait une différence majeure!

REJECT rejette les paquets reçus, mais répond à la machine qui les a émis que l'ordinateur recherché n'est pas joignable. Cela implique que la couche IP de LINUX va traiter ce paquet et utiliser des ressources systèmes, mêmes minimes, pour cela.

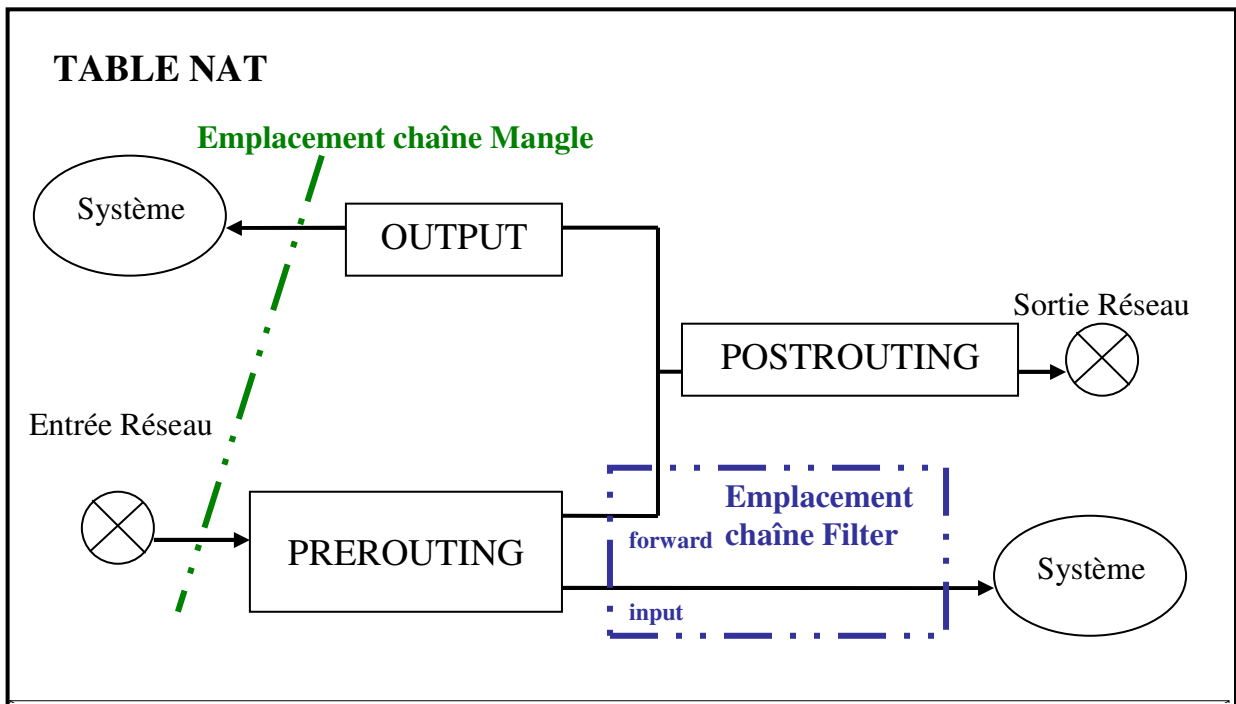
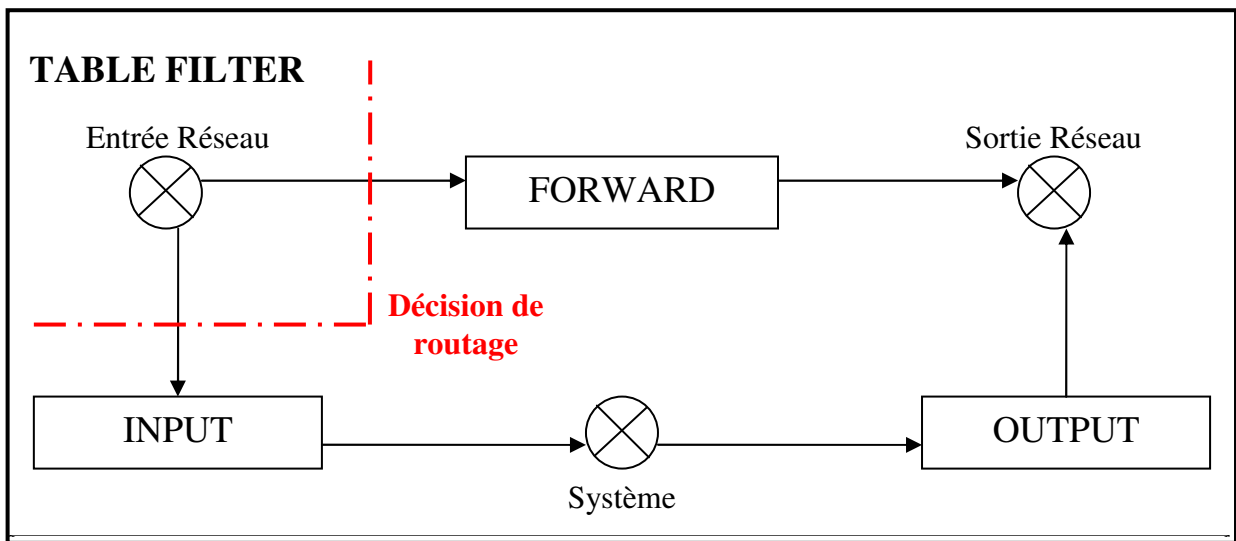
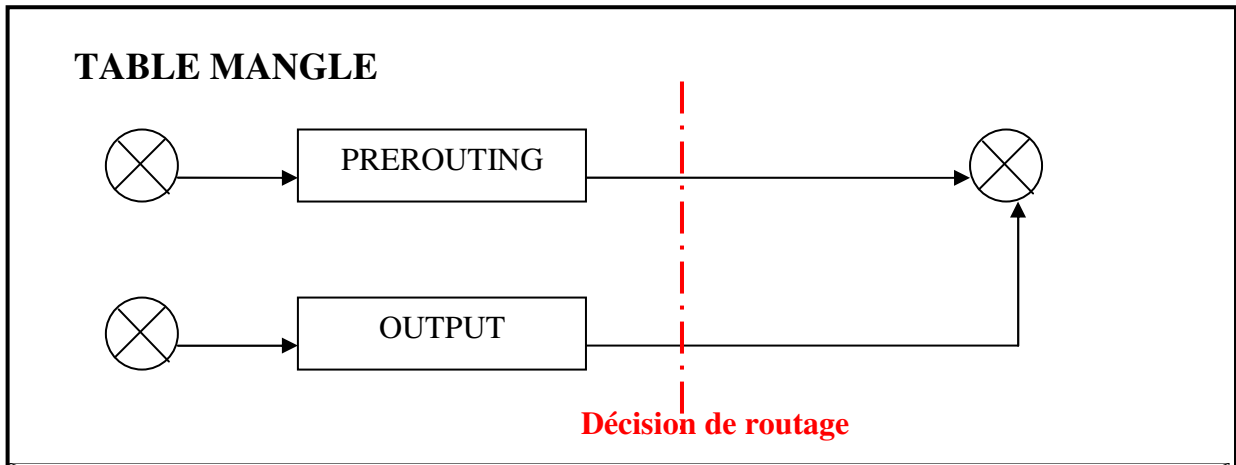
DROP rejette purement et simplement les paquets, cette méthode ne sollicite donc pas de ressources système mais elle laisse en attente le client qui a émis le paquet car celui-ci ne reçoit aucune information en retour. Cette inactivité durera donc jusqu'à ce qu'enfin le timeout de la commande qu'il a exécuté coupe la connexion.

On peut schématiser les Firewall GNU/LINUX comme suit :



Comme vous le constatez, le principe a été très simplifié sur Netfilter, ce qui permet de traiter la translation d'adresse d'un côté et les règles de filtrage avec des adresses IP réelles de l'autre.

On peut représenter théoriquement les couches NETFILTER de la manière suivante :



Conseils pratiques de mise en place de Netfilter :

1- Il est impératif de vérifier que les services réseau que l'on souhaite utiliser, soient testés et **fonctionnent correctement AVANT** la mise en place du Firewall.

2- **Ne pas laisser fonctionner des services inutiles**, même si leur accès est bloqué par le filtre.

En effet, un service ouvert par nécessité, peut être un jour "craqué" et le pirate se trouve alors derrière le filtre.

Dans ce cas, les autres services ouverts, même s'ils sont cachés de l'extérieur par le Firewall, peuvent devenir des brèches exploitables sur votre machine.

3- Faire les règles de filtrage sans tenir compte des éventuelles translations d'adresses (table filter).

4- Mettre en place par la suite les translations d'adresses nécessaires au fonctionnement (table nat).

5- **Tout interdire par défaut**, puis n'autoriser que le trafic "normal".

Attention, Unix étant un système très dépendant du service Dns, **il faut toujours penser à laisser passer les résolutions de noms** au travers du Firewall.

Commandes utiles :

Voir les règles de filtrage sur la table FILTER (par défaut si rien précisé), avec les numéros de ligne, ce qui permet de gérer plus facilement les modifications sur les règles :

iptables -L -v -n --line-numbers

Ou

iptables -t filter -L -v -n --line-numbers

Voir les règles sur la table NAT :

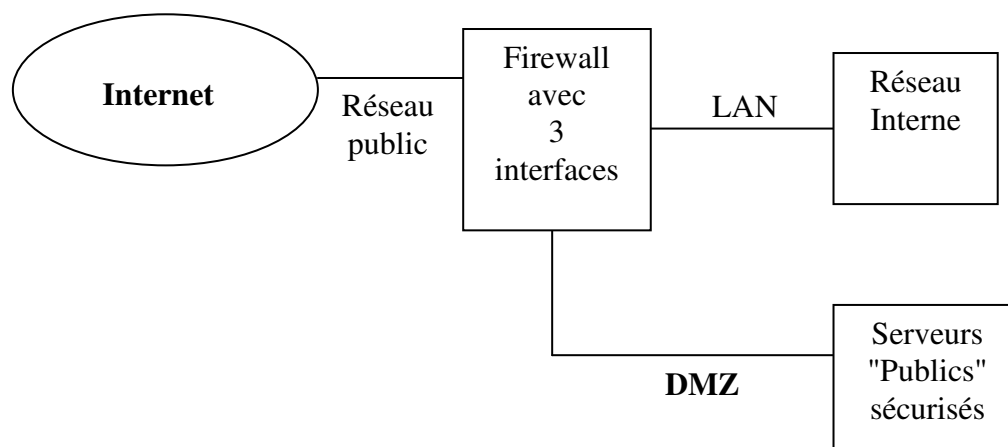
iptables -t nat -L -v -n --line-numbers

Les règles par défaut sur les chaînes du Firewall sont DROP ou ACCEPT.

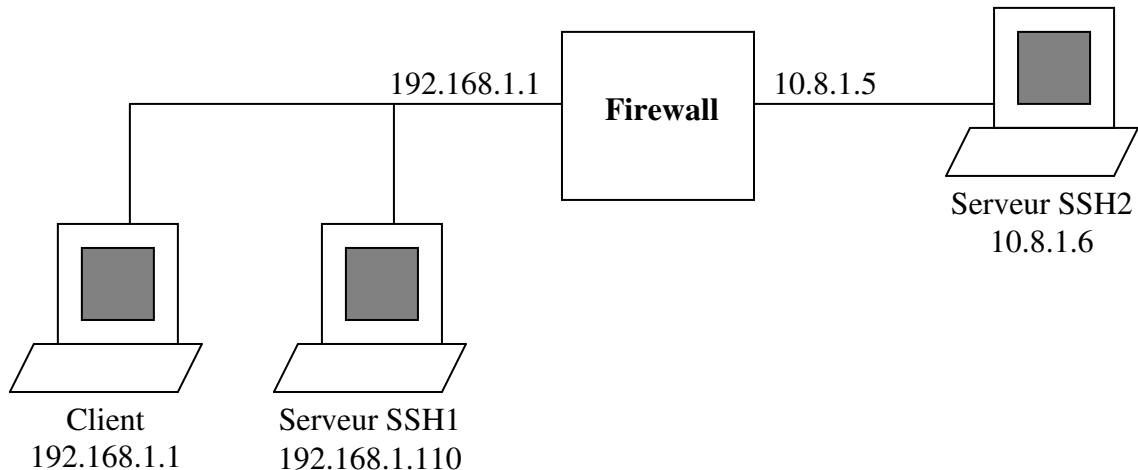
Modifier la règle par défaut de la chaîne FORWARD :

iptables -P FORWARD DROP

Exemple de sécurisation réseau courante :



Réseau de test installé :



1^{er} TEST : Rediriger toutes les connexions réseau sortantes vers une machine locale.

iptables -t nat -A PREROUTING -j DNAT --to-destination 192.168.1.110

Le Client fait une requête SSH vers Serveur SSH2 (de source Client vers adresse de destination FW).

Le Firewall intercepte la requête et la redirige vers Serveur SSH1 (masquage, soit : de source Client vers adresse de destination SSH1).

SSH1 répond à Client (source SSH1 vers destination Client) mais la connexion ne s'établit pas, car Client n'a pas demandé de connexion à SSH1 (source Client vers FW ≠ source Client vers SSH1).

La connexion pourrait fonctionner si le passage à travers la passerelle était obligatoire (les adresses sources et destination seraient bien masquées par le Firewall en entrée et sortie). Ici ce n'est pas le cas, car on renvoie sur le réseau local.

Pour que cela fonctionne, il faut changer l'adresse source de la requête, par la commande :

iptables -t nat -A POSTROUTING -j SNAT --to-destination 192.168.1.1

Remarque :

le masquage de l'adresse source (SNAT) se fait dans la table POSTROUTING.

le masquage de l'adresse destination (DNAT) se fait dans la table PREROUTING.

2^{ème} TEST : Rediriger une connexion à destination de la passerelle locale vers une machine externe.

On supprime les 2 règles établies précédemment.

iptables -t nat -D POSTROUTING 1 (où 1 est le numéro de ligne)

iptables -t nat -D PREROUTING 1

On crée la règle de natage.

iptables -t nat -A PREROUTING -j DNAT --to-destination 10.8.1.6

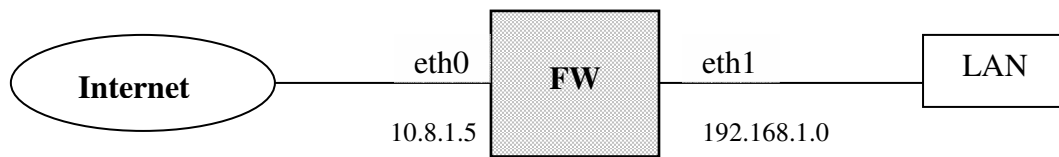
Le Client émet une requête ssh vers FW.
Celui-ci renvoie la requête à SSH2 en masquant l'adresse de Client par la sienne.
SSH2 réponds à FW, qui re-émet la réponse à l'adresse de Client.
La réponse de la requête de Client arrive bien de l'adresse de FW, la connexion s'établit.

Règle pour bloquer le ping (echo request) venant du Client vers le réseau externe :

```
iptables -t filter -A FORWARD -p icmp --icmp-type echo-request -s 192.168.1.10 -d  
10.8.1.0/24 -j DROP
```

Principales règles à appliquer sur un Firewall

Schéma du réseau décrit ci-dessous:



1- Refuser les paquets avec tous les flags à 1 ou 0 (scan nmap)

```
iptables -A INPUT -p tcp --tcp-flags ALL ALL -j DROP (tous les flags activés)
iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP (tous les flags désactivés)
iptables -A FORWARD -p tcp --tcp-flags ALL ALL -j DROP
iptables -A FORWARD -p tcp --tcp-flags ALL NONE -j DROP
```

2- Refuser une adresse privée (les 3 classes réservées) venant de l'interface publique Internet :

```
iptables -A INPUT -i eth0 -s 192.168.0.0/16 -j DROP
iptables -A INPUT -i eth0 -s 172.16.0.0/8 -j DROP
iptables -A INPUT -i eth0 -s 10.0.0.0/4 -j DROP
iptables -A FORWARD -i eth0 -s 192.168.0.0/16 -j DROP
iptables -A FORWARD -i eth0 -s 172.16.0.0/8 -j DROP
iptables -A FORWARD -i eth0 -s 10.0.0.0/4 -j DROP
```

3- Mettre la police par défaut DROP sur INPUT et FORWARD

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

4- Bloquer les paquets qui entrent par une interface et qui repartent par la même interface (IPSPHOOFING) :

```
iptables -A FORWARD -i eth0 -o eth0 -j DROP
```

✓ On peut la placer dans une chaîne utilisateur pour pouvoir l'affiner au cas où on fasse un forward un jour vers Internet (trafic suspect normalement, mais peut être utilisé lors d'une redirection vers une autre machine sur Internet).

```
iptables -N traffic-suspect
iptables -A traffic-suspect -i eth0 -o eth0 -j DROP
iptables -A traffic-suspect -i eth1 -o eth1 -j DROP
```

```
iptables -A FORWARD -j traffic-suspect
```

Remarque : A chaque fois qu'une règle est concernée par le trafic est vérifiée, un compteur l'incrémente.

Pour le remettre à 0 taper 'iptables -Z FORWARD' pour réinitialiser la chaîne FORWARD.

5- Droper tous les paquets invalides (connexions non établies, ni nouveaux, soit les paquets incohérents):

Ce test nécessite le module ipcontrack dans le noyau.

```
iptables -A INPUT -m state --state INVALID -j DROP
iptables -A FORWARD -m state --state INVALID -j DROP
```

6- Accepter le ping de l'intérieur vers l'extérieur :

✓ accepter les machines internes à pinguer la passerelle

```
iptables -A INPUT -p icmp --icmp-type echo-request -i eth1 -s 192.168.1.0/24 -j ACCEPT
```

✓ accepter les machines internes à pinguer l'extérieur

```
iptables -A FORWARD -p icmp --icmp-type echo-request -i eth1 -s 192.168.1.0/24 -o eth0 -j ACCEPT
```

7- Autoriser la résolution des noms Internet pour les stations :

```
iptables -A FORWARD -p udp -i eth1 -s 192.168.1.0/24 --sport 1024: -d 213.190.70.5 --dport 53 -j ACCEPT
```

✓ idem pour tcp

```
iptables -A FORWARD -p tcp -i eth1 -s 192.168.1.0/24 --sport 1024: -d 213.190.70.5 --dport 53 -j ACCEPT
```

✓ pour autoriser la réponse du serveur DNS (ou autre connexion établie)

```
iptables -I FORWARD 7 -m state --state RELATED,ESTABLISHED -j ACCEPT
```

✓ ici la valeur 7 va placer la règle juste avant celle acceptant le DNS qui était à 7 et passera 8

8- Finition :

✓ Penser à mettre le NAT pour le réseau interne

```
iptables -t nat -A POSTROUTING -o eth0 -s 192.168.1.0/24 -j SNAT --to-source 10.8.1.5
ou si l'adresse externe peut changer ( utile pour du pppoe ),
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

✓ Autoriser la résolution DNS pour le Firewall.

```
iptables -A INPUT -p udp -i eth0 -s 213.190.70.5 --sport 53 -d 10.8.1.5 --dport 1024: -j ACCEPT
```

9- Autoriser le HTTP pour les stations du réseau

```
iptables -A FORWARD -p tcp -i eth1 -s 192.168.1.0/24 --sport 1024: -o eth0 --dport 80 -j ACCEPT
```

✓ idem pour le GW

```
iptables -A INPUT -p tcp -i eth0 --sport 80 -j ACCEPT
```

✓ pour ne pas avoir à accepter les réponses de chaque services et autoriser icmp on utilise plutôt

```
iptables -I INPUT 6 -m state --state RELATED,ESTABLISHED -j ACCEPT
```

10- Autoriser le HTTPS pour les stations du réseau

```
iptables -A FORWARD -p tcp -i eth1 -s 192.168.1.0/24 --sport 1024: -o eth0 --dport 443 -j ACCEPT
```

Service FTP :

Description des modes FTP :

- Mode actif

1- Etablissement connexion : Le Client fait la requête à partir d'un port 1024 à 65535 sur le port 21 du serveur FTP.

2- Envoie d'un fichier ou du ls : Le serveur d'un port supérieur à 1024 vers un port supérieur 1024 du client.

- Mode passif

1- Etablissement connexion : Le Client fait la requête à partir d'un port 1024 à 65535 sur le port 21 du serveur FTP.

2- Le client ouvre un port supérieur à 1024 vers le port 20 du serveur FTP

Toutes les requêtes viennent du client pour passer les Firewalls passifs.

Attention : Pour le service ftp il faut charger les modules **ip_conntrack_ftp** et **ip_nat_ftp**.

Pour ce faire, utiliser les commandes :

```
/sbin/modprobe ip_conntrack_ftp
```

```
/sbin/modprobe ip_nat_ftp
```

L'idéal est de les placer dans les scripts de démarrage de linux ou d'iptables.

11- Accepter le service FTP

```
iptables -A FORWARD -p tcp -i eth1 -s 192.168.1.0/24 --sport 1024: -o eth0 --dport 21 -j ACCEPT
```

12- Accepter le service pop vers tous les serveur pop (à partir du lan)

```
iptables -A FORWARD -p tcp -i eth1 -s 192.168.1.0/24 --sport 1024: -o eth0 --dport 110 -j ACCEPT
```

13- Accepter le service smtp vers mail.caplaser.com pour les stations du réseau

```
iptables -A FORWARD -p tcp -i eth1 -s 192.168.1.0/24 --sport 1024: -o eth0 -d 213.190.12 --dport 25 -j ACCEPT
```

✓ Pour les services, on peut mettre le nom du service (voir dans /etc/services).

✓ Au lieu de mettre l'adresse ip d'une machine, on peut mettre son nom complet, mais cela nécessite que la passerelle puisse résoudre les noms. (cela est utile pour des sites webs en load-balancing par exemple, plusieurs machines répondent pour un même site)

Pour logger le trafic suspect détecté par le Firewall, on peut créer un chaîne particulière :

```
iptables -N traffic_suspect
iptables -A traffic_suspect -i eth0 -o eth0 -j LOG --log-prefix "\!\! Alerte Firewall : "
iptables -A traffic_suspect -i eth0 -o eth0 -j DROP
iptables -A traffic_suspect -i eth1 -o eth1 -j DROP
iptables -A FORWARD -j traffic_suspect
```

Remarque : Pour limiter le log de Netfilter au fichier /var/log/messages et ne pas avoir ces messages à l'écran, il faut spécifier le --log-level info pour que les logs aillent dans syslog et pas à l'écran.

```
iptables -A traffic_suspect -i eth0 -o eth0 -j LOG --log-prefix "\!\! Alerte Firewall : " --log-level 6
```

Attention : Pour le pb de demande d'écrasement de fichier avec la commande cp, il faut voir que cp est en fait un alias pour root de cp -i ,donc il demande toujours confirmation

Solution 1 : supprimer l'alias

Solution 2 : utiliser le nom complet de cp, avec son chemin

Quand des commandes sont des alias, on peut afficher la commande complète avec la combinaison de touche « ctrl + alt + e »

PROXY SQUID

Configuration matérielle recommandée pour un réseau d'une vingtaine de postes :

- Processeur Pentium 233 minimum.
- 128 Mo de RAM minimum, 256 Mo conseillé.
- Préférer des disques SCSI aux IDE car plus rapides.
- Créer 2 caches au minimum de 1Go chacun plutôt qu'un seul de 2 Go (mieux géré par SQUID).
- Pour optimiser encore le fonctionnement du proxy, on peut mettre les caches sur deux disques séparés et utilisés uniquement pour cette fonction.
- Réseau 100Mb/s switché.
- Désactiver le cache du navigateur du client afin qu'il n'utilise que le proxy.

Installation du logiciel :

On peut utiliser les RPMS fournis sur la distribution Redhat ou télécharger les fichiers sources sur le site de SQUID (www.squid-cache.org).

Avec le fichier tar.gz des sources, suivre cette procédure.

- Décompacter le fichier avec la commande :
tar xvfz squid-2.4.STABLE3-src.tar.gz

- Aller dans le dossier créé :
cd squid-2.4.STABLE3

- Configurer le paquetage pour fonctionner conjointement avec Netfilter et utiliser le module PAM pour authentifier les utilisateurs.

./configure --prefix=/opt/squid --sysconfdir=/etc/squid --enable-gnuregexp --enable-err-language=French --enable-linux-netfilter --disable-ident-lookups --enable-auth-modules=PAM

- Compiler le programme par la séquence :
make && make install

On a choisi d'authentifier les utilisateurs avec PAM, c'est à dire que SQUID crée un binaire **/opt/squid/libexec/squid/pam_auth** qui utilisera PAM pour valider le mot de passe d'un compte donné.

Pour que cela fonctionne, il faut créer un fichier **squid** dans le dossier **/etc/pam.d**.

Si il n'est pas créé par l'install, copier celui de **login** et supprimer les lignes :

**session optional /lib/security/pam_console.so et
auth required /lib/security/pam_securetty.so**

Le fichier **/etc/pam.d/squid** ressemblera donc à :

##PAM-1.0

**auth required /lib/security/pam_stack.so service=system-auth
auth required /lib/security/pam_nologin.so**

```
account required /lib/security/pam_stack.so service=system-auth
password required /lib/security/pam_stack.so service=system-auth
session required /lib/security/pam_stack.so service=system-auth
```

Le binaire **pam_auth** devant lire les fichiers **/etc/passwd** et **/etc/shadow**, il faut activer le sticky bit sur cet exécutable (**le fichier s'exécutera en tant que root**) :

```
chmod u+s /opt/squid/libexec/squid/pam_auth
chmod g+s /opt/squid/libexec/squid/pam_auth
chmod o+s /opt/squid/libexec/squid/pam_auth
```

-Initialiser le cache avec la commande :
squid -z

Pour prendre en compte des modifications du fichier de configuration de SQUID sans arrêter le service, utiliser la commande :
squid -k reconfigure

Description des valeurs importantes du fichier /etc/squid/squid.conf :

http_port 192.168.1.1:8080

Port d'écoute de SQUID, avec restriction d'écoute sur l'adresse 192.168.1.1.
On peut spécifier plusieurs directives http_port

cache_mem 8 MB

Variable indiquant la taille maximale des objets que SQUID va garder en cache mémoire, la valeur par défaut est correcte.

cache_swap_low 90

cache_swap_high 95

Pourcentage de la taille mémoire maximale utilisée (y compris le swap), on préférera 80 à 85 % aux valeurs par défaut.

maximum_object_size 16 MB

Taille maximale des objets que SQUID va conserver dans le cache disque (choisir 15 à 20 Mo).

minimum_object_size 0 KB

Taille minimale des objets que SQUID va conserver dans le cache disque, la valeur par défaut 0 est correcte.

cache_replacement_policy lru

A conserver tel quel, cette variable ne met pas en cache les pages web dynamiques.

cache_dir ufs /cache1 1000 16 256

cache_dir ufs /cache2 1000 16 256

Dossier de cache de SQUID, on peut en spécifier plusieurs (max. 1 Go) comme ci-dessus. Cette variable est une des plus importantes, on peut optimiser le cache en suivant les indications suivantes :

Une partition par cache afin d'éviter la saturation des partitions systèmes. L'idéal serait un

disque dur par cache pour optimiser la vitesse du cache.

On peut monter les partitions de cache sans droit d'exécution pour sécuriser encore la machine.

Si on ajoute un cache ultérieurement, il faut arrêter squid et lancer la commande **squid -z** pour initialiser le cache.

Si on supprime le fichier swap.state dans le dossier du cache, la commande **squid -z** réinitialisera TOUS les caches.

cache_access_log /var/log/squid/access.log

Emplacement du fichier enregistrant les requêtes des clients au serveur PROXY.

cache_log /var/log/squid/cache.log

Emplacement du fichier enregistrant logs de fonctionnement du service PROXY.

cache_swap_log

Fonction désactivée par défaut, ne pas l'activer.

emulate_httpd_log off

Possibilité de demander à SQUID d'enregistrer ses logs à la manière d'APACHE afin d'utiliser un logiciel d'analyse non prévu pour SQUID.

pid_filename /var/log/squid/squid.pid

Fichier d'id du service SQUID.

log_fqdn off

Possibilité de logger les noms complets après résolution DNS.

Il vaut mieux le désactiver, car cela perturbe le fonctionnement.

ftp_user Squid@

Identifiant envoyé aux serveurs FTP comme login de connexion.

ftp_passive off

Possibilité de passer SQUID en mode passif pour s'adapter à ipchains qui ne gère pas le mode actif.

Laisser à off pour Netfilter.

dns_nameservers

SQUID lit par défaut la configuration du fichier /etc/resolv.conf. On peut donc avec la variable dns_nameservers lui imposer des serveurs DNS.

authenticate_program /opt/squid/libexec/squid/pam_auth

Par défaut, SQUID ne demande pas d'authentification aux clients.

Si on souhaite gérer des comptes, on modifie la variable authenticate_program.

L'exemple ci-dessus indique que l'on utilise le module PAM dont on a parlé plus haut.

Cette authentification permet de générer des statistiques plus poussées, client par client.

authenticate_ttl 1 hour

Durée d'expiration du mot de passe.

SQUID demandera chaque heure le mot de passe de l'utilisateur, même si le navigateur reste ouvert.

cache_effective_user nobody
cache_effective_group nobody

Compte utilisateur et groupe utilisé pour lancer SQUID afin de ne pas démarrer le service en root.

cache_mgr webmaster

Boite locale recevant les messages de SQUID. Il s'agit d'une boîte locale, que l'on peut rediriger sur une autre.

logfile_rotate 10

Nombre de logs maxi que conserve SQUID.

Pour faire tourner les logs, utiliser la commande '**squid -k rotate**'.

append_domain .yourdomain.com

Variable indiquant l'extension de domaine à utiliser si un client ne donne pas de nom complet de machine.

Quand on utilise la fonction PROXY transparent en liaison avec le Firewall, il faut activer les variables ci-dessous, sinon cela ne fonctionne pas.

httpd_accel_host virtual

httpd_accel_port 80

httpd_accel_with_proxy on

httpd_accel_uses_host_header on

ACCESS LIST :

On peut utiliser des restrictions pour limiter l'accès au PROXY pour certaines personnes. Par défaut, personne n'est autorisé à utiliser SQUID, il faut activer au moins l'accès au réseau local.

Exemple d'acl utilisé :

acl all src 0.0.0.0/0.0.0.0

acl manager proto cache_object

acl localhost src 127.0.0.1/255.255.255.255

acl SSL_ports port 443 563

acl CONNECT method CONNECT

acl localNet src 192.168.1.0/255.255.255.0

acl pass proxy_auth REQUIRED

http_access allow manager localhost

http_access deny manager

http_access deny !Safe_ports

http_access deny CONNECT !SSL_ports

http_access allow localNet

http_access deny all

icp_access allow all