Politique d'accès

Différents comportements du système s'ajustent en modifiant des paramètres pris en compte par le noyau. On classe ces comportements dans deux catégories : les ajustements relatifs à la sécurité et ceux relatifs à l'ajout de fonctionnalités.

IPv6, protocole Internet adopté comme standard par l'IETF (*Internet Engineering Task Force*), bien que lancé en 1990 n'est qu'imparfaitement intégré sur le réseau des réseaux entraînant parfois des problèmes d'implantation. Même si certains fournisseurs d'accès comme Free commencent à proposer l'activation d'IPv6, la gestion de ce protocole ne sera pris en compte que dans la prochaine version de ce manuel. Le pare-feu d'Ubuntu **`ufw`**, désactive d'ailleurs par défaut ce protocole.

1. Réglages essentiels

a. Désactivation/Activation d'IPv6

La méthode consistant à supprimer l'alias dans le fichier `/etc/modprobe.d/aliases` est à proscrire pour deux raisons : d'abord parce que le commentaire en tête du fichier l'indique (!) et ensuite parce qu'une modification induira des problèmes lors d'une mise à jour via le paquetage `module-init-tools`. Il faut lui préférer la méthode de la liste noire et, avec Ubuntu, la création de son propre fichier `blacklist` :

Fichier `/etc/modprobe.d/blacklist-admin` :

```
blacklist ipv6
```

Un dernier réglage porte sur le fichier `/etc/hosts` où il ne doit pas rester de lignes concernant Ipv6 : le mieux étant de les mettre en commentaire (on peut en avoir besoin lors d'un rétablissement de la configuration) :

```
# The following lines are desirable for Ipv6 capable hosts
#::1        ip6-localhost ip6-loopback
#fe00::0    ip6-localnet
#ff00::0    ip6-mcastprefix
#ff02::1    ip6-allnodes
#ff02::2    ip6-allrouters
#ff02::3    ip6-allhosts
```

b. Fichier sysctl.conf et routage

Le fichier `/etc/sysctl.conf` constitue le "pendant" de la commande `sysctl` et traite des ajustements systèmes pour modifier des paramètres du noyau. Concrètement pour modifier "à la volée", c'est-à-dire lorsque le système est en fonctionnement et jusqu'à un prochain redémarrage, on peut :

- soit utiliser la commande `sysctl` ;
- soit écrire directement dans les variables fichiers du répertoire `/proc`.

Le fichier `/etc/sysctl.conf` est utilisé pour prendre en compte ces changements au démarrage du système. Vous trouverez ci-dessous, une sélection de variables à gérer, généralement utiles pour l'administrateur d'un système.

Transformation du système en routeur

Sur une machine de type Ubuntu Linux, on parlera de routage logiciel pour la commutation de paquets entre réseaux différents, d'un port d'entrée vers un port de sortie (nombre d'interfaces réseau minimum : 2). Comme il est préférable de garder la main sur la fonctionnalité afin de pouvoir l'interrompre le cas échéant, on utilise l'écriture directe dans la variable, voire à l'intérieur d'un script gérant les règles d'un pare-feu.

```
sysctl net.ipv4.ip_forward
```

Ou (`ipv6` comme `ipv4`) :

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

La commande `sysctl -a` affiche l'ensemble des paramètres passés au noyau.

Ajustements liés à la sécurité réseau

Tous les réglages suivants s'inscrivent dans le fichier `/etc/sysctl.conf` afin de protéger le système au démarrage.

Protection contre les attaques de type `spoofing`

Ce réglage protège contre les arrivées de paquets provenant d'une source usurpée. Pour l'interface de `loopback`, le bouclage attendu interdit une source autre qu'elle-même. Sur un système à plusieurs interfaces cela revient à vérifier la cohérence entre les interfaces et la table de routage.

Sur Ubuntu, ce réglage est activé par défaut (mettez des espaces entre le signe égal pour avoir la coloration syntaxique de VIM).

```
net.ipv4.conf.default.rp_filter = 1
net.ipv4.conf.all.rp_filter = 1
```

Protection contre les attaques de type `syn flooding`

Ce réglage active un numéro de séquence destiné à rendre non nécessaire la sauvegarde du SYN initial dans la séquence `three-way handshake` pour une connexion TCP. Afin d'éviter trop de connexions ouvertes et donc un déni de service, Ubuntu gère une liste (*backlog*) d'une longueur de 1024 (voir le contenu pour cela de `/proc/sys/net/ipv4/tcp_max_syn_backlog`).

Activer les `syncookies` permet de protéger au mieux la surcharge d'un serveur :

```
net.ipv4.tcp_syncookies = 1
```

Redirection ICMP

L'utilisation de la redirection ICMP pour informer d'une route plus adaptée est inutile si votre réseau est bien configuré. Cela peut même s'avérer dangereux par le biais d'injection de routes dans la table de routage. Les trois variables (en commentaire) sur Ubuntu sont positionnées par défaut (faire un `sysctl -a | grep redirects`) à 1 pour l'acceptation, 1 pour l'émission et 1 pour la redirection via une passerelle par défaut reconnue. Pour un routeur, seule l'émission garde la position booléenne à "vrai".

```
net/ipv4/conf/all/accept_redirects = 0
net/ipv4/conf/all/secure_redirects = 0
net/ipv4/conf/all/send_redirects = 1
```

Vous ne devez pas interdire les PING : d'abord, parce que cela va à l'encontre de la RFC 792, ensuite parce que cela ne sert maintenant à rien, si ce n'est qu'à satisfaire une tendance paranoïaque...

Surveiller les "martiens"

Cette appellation fantaisiste dénomme les paquets sur le réseau ayant des adresses sources/destination invalides. Mettre la valeur permet d'inscrire dans les logs ces paquets :

```
net/ipv4/conf/all/log_martians = 1
```

Autres paramètres

Trois derniers paramètres en commentaire dans le fichier `sysctl.conf` restent en l'état par défaut (à ne pas changer donc) avec respectivement :

- ne pas autoriser la fonctionnalité du `ping` en `broadcast` soit sur l'ensemble des machines du réseau.
- ne pas autoriser l'enregistrement dans les logs des paquets ICMP mal formés.
- ne pas autoriser l'inscription de passerelles à l'intérieur d'un paquet IP.

```
net/ipv4/icmp_echo_ignore_broadcasts = 1
net/ipv4/icmp_ignore_bogus_error_responses  = 1
net/ipv4/conf/all/accept_source_route  = 0
```

2. Mise en place de règles de pare-feu

a. En ligne de commandes avec UFw

La gestion d'un pare-feu sous Ubuntu s'apparente à celle d'une distribution Linux traditionnelle avec l'utilisation de **Netfilter** (le module du noyau pour le filtrage des paquets IP) et **IPtables** (l'interface de configuration en ligne de commandes). Ayant toujours le souci de "démocratiser" leur distribution, ou tout au moins de la rendre plus accessible, les développeurs Ubuntu ont introduit une nouvelle interface plus agréable à utiliser : **UFw**. Dans l'esprit de l'ouvrage, vous n'aurez pas ici une thématique sur IPtables (vu et revu dans nombres de livres sur Linux) mais sur la spécificité Ubuntu avec l'utilisation de ce nouvel outil, UFw.

UFw ne veut pas dire Ubuntu Firewall mais *Uncomplicated Firewall* (le mot anglais n'est pas cette fois-ci un faux ami).

Le service UFw se met en place non pas par un démarrage par `/etc/init.d/ufw start` mais par la commande :

```
ufw enable
```

Cette commande positionne la valeur `ENABLED` à `yes` dans le fichier `/etc/ufw/ufw.conf` (il ne contient d'ailleurs que cette ligne). Le lancement manuel (ou au démarrage du système) facultatif sans cette valeur à `yes` indiquera un service non démarré (`skipped`).

On voit tout de suite son fonctionnement avec les règles mises en place à partir de l'ancienne interface IPtables (toujours opérationnelle) :

```
iptables -L
```

Ce qui donne en sortie (extrait) :

```
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere              anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination
ufw-before-forward  all  --  anywhere           anywhere
ufw-after-forward  all  --  anywhere            anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ufw-before-output  all  --  anywhere            anywhere
ufw-after-output  all  --  anywhere             anywhere

Chain ufw-after-forward (1 references)
target     prot opt source               destination
LOG        all  --  anywhere             anywhere            limit: avg 3/min burst 10 LOG level war
ning prefix `[UFW BLOCK FORWARD]: '
RETURN     all  --  anywhere             anywhere

Chain ufw-after-input (1 references)
target     prot opt source               destination
RETURN     udp  --  anywhere             anywhere            udp dpt:netbios-ns
RETURN     udp  --  anywhere             anywhere            udp dpt:netbios-dgm
RETURN     tcp  --  anywhere             anywhere            tcp dpt:netbios-ssn
RETURN     tcp  --  anywhere             anywhere            tcp dpt:microsoft-ds
RETURN     udp  --  anywhere             anywhere            udp dpt:bootps
RETURN     udp  --  anywhere             anywhere            udp dpt:bootpc
LOG        all  --  anywhere             anywhere            limit: avg 3/min burst 10 LOG level war
ning prefix `[UFW BLOCK INPUT]: '
RETURN     all  --  anywhere             anywhere

Chain ufw-after-output (1 references)
target     prot opt source               destination
RETURN     all  --  anywhere             anywhere
:_
```

Il en ressort qu'UFw introduit les chaînes `ufw-before-*` et `ufw-after-*` dans les trois chaînes principales INPUT, OUTPUT et FORWARD (table `filter` ou `table` contenant les règles de filtrage des paquets en entrée ou en sortie dans la machine, et ceux routés d'une interface à l'autre).

Une sous-chaîne est aussi présente dans les trois : `ufw-user-*` et dans les nouvelles versions d'UFw, les chaînes `ufw-user-limit` et `ufw-user-limit-accept`.

Le fichier `sysctl.conf` pris en compte après l'établissement d'UFw est celui qui se trouve dans le répertoire `/etc/ufw` et non le fichier se trouvant dans `/etc`.

Fonctionnement d'UFw

L'établissement des règles par défaut fait que le système accepte tout sur l'interface de la boucle locale (*loopback*) et en sortie. En entrée, toujours en ACCEPT par défaut, la règle est configurable et n'accepte que quelques services (ICMP, udp). Les paquets jetés sont enregistrés (LOG) avec une limite.

Syntaxe générale (et complète) :

```
ufw allow|deny [proto <protocole>]
 [from <addresse> [port <port>]] [to <addresse> [port <port>]]
```

Exemple :

Une fois UFw lancé avec les règles par défaut, un `ping` venant d'une autre machine fonctionne mais pas d'autres services. Autrement dit, si vous avez par exemple un service Apache (WEB), il n'est plus accessible à partir d'autres postes. Il faut le rétablir :

```
ufw allow proto tcp from 192.168.3.1 to 192.168.3.134 port 80
```

Ce qui se traduit par : accepter pour le protocole TCP les demandes faites par la machine d'IP 192.168.3.1 pour 192.168.3.134 (IP du serveur WEB) sur le port 80.

Un exemple plus simple :

```
ufw allow ssh
```

Dans ce cas, la permission est donnée pour une connexion SSH à partir de n'importe quel poste externe. Voir le manuel en ligne d'UFw pour plus d'exemples. On aurait pu remplacer la commande par :

```
ufw allow to any port 22 from any
```

Ou :

```
ufw allow proto tcp to any port 22 from any
```

```
ufw allow proto udp to any port 22 from any
```

La vérification de ces règles se fait par la commande :

```
ufw status
```

Ce qui donne après les deux règles précédentes :

```
root@server:/etc# ufw status
Firewall loaded

To                         Action  From
--                         ------  ----
192.168.3.134 80:tcp       ALLOW   192.168.3.1
22:tcp                     ALLOW   Anywhere
22:udp                     ALLOW   Anywhere

root@server:/etc#
```

Destruction d'une règle

La suppression d'une règle simple dans sa syntaxe de base, **nécessite de connaître la règle exacte précédemment utilisée** :

```
ufw delete allow|deny règle
```

L'outil graphique GUFw

Suite logique d'UFw, le développement d'un outil graphique sous GNOME permettant le paramétrage des règles autrement que par les lignes de commandes. Encore en développement, **GUFw** apparaît dans les dépôts de la version **8.10 Intrepid Ibex**. Pour l'instant, en environnement graphique, l'outil **FireStarter** (voir ci-dessous) remporte plus les suffrages de ceux qui désirent s'initier et établir un pare-feu sans trop de complications.

b. Par une interface graphique avec FireStarter

Dans le cas d'un poste de travail, le travail d'une équipe de développeurs (http://www.fs-security.com/) a abouti au projet **FireStarter** en OpenSource.

L'établissement de règles de pare-feu se fait par une interface graphique (Gnome sous GTK) et d'un assistant pour la configuration d'un pare-feu, ce qui en facilite considérablement la mise en place. Ses principales qualités :

- Utilisable pour les postes de travail ou une passerelle.
- Gestion en temps réel des évènements comme les tentatives d'intrusion.
- Offre le partage de connexion Internet (plus le service DHCP si besoin).
- Gestion des ports simple, en entrée comme en sortie.
- Gestion de listes noires et/ou blanches pour les flux.
- Protection contre les attaques de type `flooding`, `broadcasting`, `spoofing`, `DoS`...
- Possibilité d'écrire des scripts utilisateur (*rulesets*) avant ou après activation du pare-feu.

Disponible sous Ubuntu dans le dépôt `Universe`, l'application s'installe classiquement et se trouve, une fois chargée, dans le menu **Applications - Internet - FireStarter** (vous devez entrer le mot de passe d'administration bien sûr au lancement) :

```
aptitude install firestater
```

Le premier lancement de l'application fait intervenir l'assistant de configuration :

c. Configuration

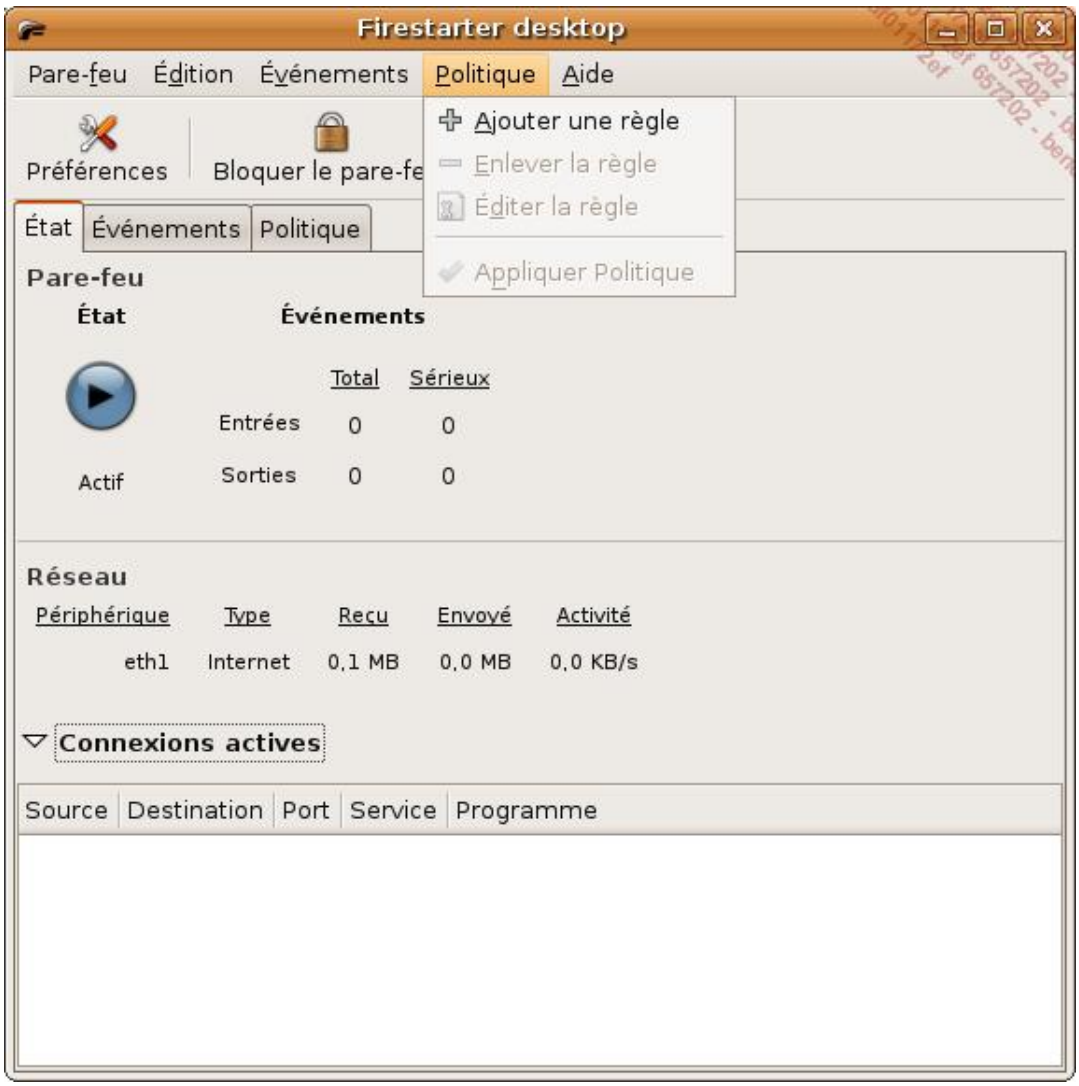
Les étapes de configuration concernent :

- **La configuration des périphériques réseaux**. Vous vérifiez la bonne détection de la carte et cochez le lancement du pare-feu à la connexion. Indiquez aussi si l'adresse IP est donnée par DHCP (acceptable dans le cas d'un poste de travail au lieu d'une IP fixe).

- **L'autorisation du partage de la connexion Internet**. Cela revient à définir la translation d'adresses (NAT) et à s'appuyer sur un serveur DHCP. Deux remarques importantes : ce dernier **n'est pas installé par défaut** avec FireStarter, vous devez le faire en utilisant le paquet logiciel `dhcp3-server` ; enfin la translation d'adresses est possible si vous avez deux interfaces réseau.

- **Le démarrage du pare-feu**. L'option est cochée par défaut, vous enregistrez les paramètres par l'appui du bouton de même nom, ce qui fait quitter l'assistant. Le menu **Pare-feu - Lancer l'assistant** relance l'assistant.

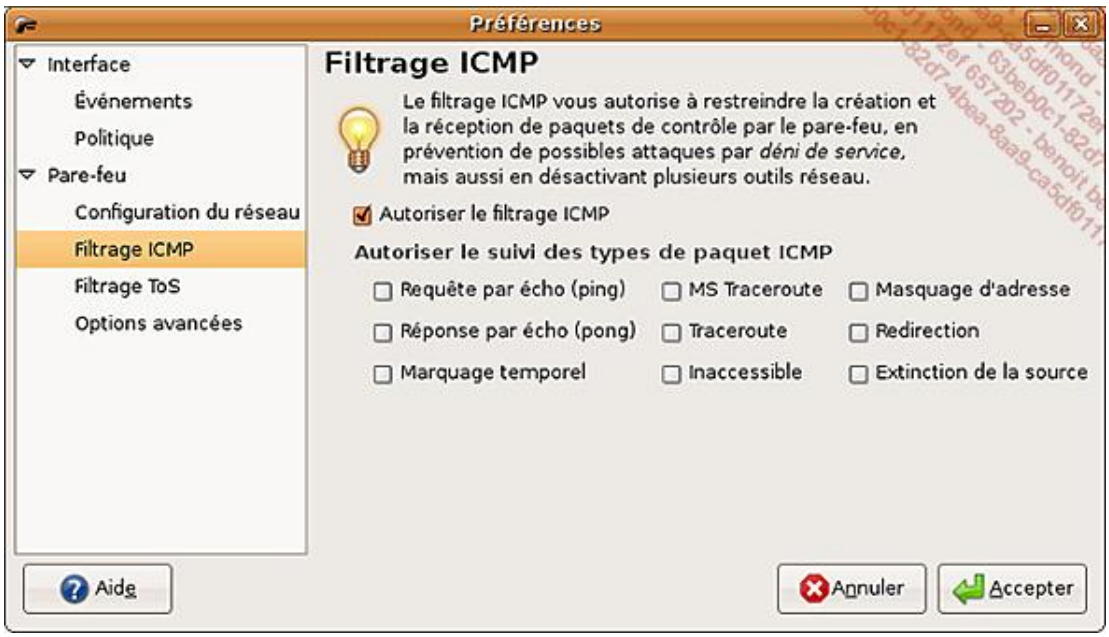
L'écran de **FireStarter** montre l'état du pare-feu et d'autres renseignements comme les évènements, les connexions actives, etc. L'un des menus les plus importants traite de la politique et des règles à ajouter :



Vous pouvez par ce menu, ajouter une politique sur le trafic entrant et/ou sortant, en fonction ou non d'un service (clic droit de la souris pour faire apparaître le menu conceptuel à partir de l'onglet **Politique** et dans la zone choisie), suivant des permissions par rapport à la liste noire et/ou blanche.

Attention : certaines options ne sont disponibles et visibles qu'en cas de partage de la connexion Internet.

Vous vous reporterez avec profit à la documentation de **FireStarter** et à son manuel en ligne. Concernant les blocages de base vus dans le premier paragraphe, leur configuration se situe au niveau du menu **Edition - Préférences**. Par exemple, vous trouverez sur la ligne Filtrage ICMP le suivi des types de paquets :



Dès lors, le filtrage mis en place provoque des évènements (réglez les colonnes visibles dans le menu **Événements - Montrer la colonne**).