Forteresse

Vous avez installé Debian et déployé plein de logiciels, c'est très bien, mais avez-vous pensé à la sécurité de votre poste de travail ? Pas la peine, Linux est inviolable et insensible au virus… Enfin, c'est ce que vous pensez, et vous n'avez pas tout à fait tort. Mais il vaut mieux se prémunir des attaques avant d'avoir une désagréable surprise.

Voici donc de quoi armer votre poste de travail et le transformer en une véritable forteresse, un espace inviolable où personne ne viendra fouiller, dans lequel personne ne pourra pénétrer.

Un système à jour

Le secret pour éviter tout problème est tout simplement d'avoir un système à jour. En effet, Debian, publie chaque jour des mises à jour de sécurité de ses paquets et vous assure donc une plus grande sécurité.

Comme nous l'avons vu dans un article précédent, il est possible d'utiliser cron-apt pour automatiser le téléchargement de mise à jour. L'installation est simple :

```
apt-get install cron-ap
```

Mais maintenant que vous êtes un utilisateur aguerri, nous pouvons aller un peu plus loin. Avec cron-apt, vous automatisez l'installation des paquets téléchargés. L'avantage est que vous n'avez plus besoin de penser aux mises à jour, le PC fait tout à votre place.

Pour éviter tout problème, nous allons configurer seulement les mises à jour de sécurité. À cette fin, créez un fichier */etc/apt/sources.list.cron-apt* que vous complétez ainsi :

```
deb  http://security.debian.org/  ↵
   squeeze/updates  main  contrib  ↵
   non-free
```

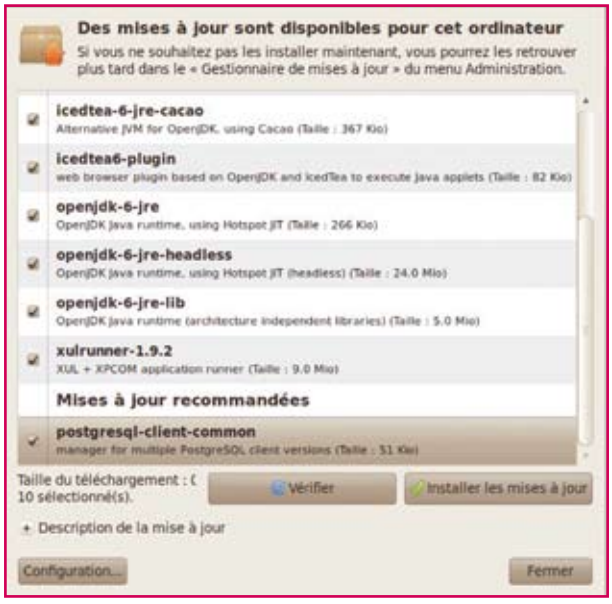

Figure 1. *L'update manager vous aidera, de manière graphique, à maintenir votre système à jour*

Figure 2. *ClamAV est un antivirus performant*

Figure 3. *ClamTK permet de contrôler ClamAV graphiquement*

Maintenant, pour configurer l'installation automatique, créez un fichier */etc/cron-apt/action.d/5-install* et ajoutez ces lignes :

```
autoclean -q -y
dist-upgrade -q -y -o APT: ↵
   :Get::Show-Upgraded=true ↵
   -o Dir::Etc::sourcelist=/ ↵
   etc/apt/sources.list.    ↵
   cron-apt -o Dir::Etc::so ↵
   urceparts=nonexistent -o ↵
   DPkg::Options::=--force- ↵
   confdef -o DPkg::Options: ↵
   :=--force-confold
```

Par défaut, toutes les mises à jour se font à 4h00 du matin (`cat /etc/cron.d/cron-apt` pour vérifier), mais tout le monde ne passe pas sa vie devant son écran. Donc, nous allons juste ajouter une règle qui permettra de faire la mise à jour dans la journée. Pour cela, la solution de facilité consiste à faire un lien symbolique :

```
ln -s /usr/sbin/cron-apt /etc/ ↵
   cron.daily/cron-apt
```

Désormais, votre système s'actualisera sans que vous vous en rendiez compte. Un système à jour est la clé de la sécurité car la plupart des intrusions se font par le biais de failles de sécurité présentes dans les applications installées.

Figure 4. *Pas besoin d'aller si loin pour créer un firewall, il y a iptables !*

Killer de virus

Il n'y a pas de virus sous Linux donc nous allons installer... un antivirus. Raisonnement étrange, n'est-ce pas ? Mais peut-être que, tout simplement, l'absence de virus sous Linux est un mythe. Supercherie ? Non, pas vraiment, le nombre de virus identifiés sous Linux est très faible, mais le risque existe cependant. Rassurez-vous, les antivirus pour Linux existent et sont même rudement efficaces.

Notre choix s'est porté ici sur ClamAV, un antivirus GPL, très léger et très performant. Bien qu'invisible puisqu'il fonctionne en tâche de fond, cet outil se révèle efficace. ClamAV est présent dans les dépôts et il s'installe par cette commande :

```
apt-get install clamav clamav- ↵
   freshclam clamav-daemon
```

Et voilà, votre antivirus est installé. Avoir un antivirus, c'est bien, mais savoir s'en servir, c'est mieux. Voici quelques commandes de base à connaître, sachant qu'elles doivent s'exécuter sous le profil de l'utilisateur root ou en utilisant sudo.

- Mettre à jour sa base de définition des virus : `freshclam`
- Scanner les fichiers d'un répertoire : `clamscan -r -i /home/toto`
- Scanner les fichiers d'un répertoire avec alerte sonore et écriture dans un fichier de log : `clamscan --bell -r -i --log=/tmp/mesvirus.log /home/toto`

Nous allons également automatiser la mise à jour de la base antivirus, grâce à la commande `crontab`. Sous le profil de l'utilisateur root, lancez la commande `crontab -e` et ajoutez la ligne suivante :

```
33  *  *  *  *  /usr/bin/ ↵
   freshclam --quiet
```

Ainsi, la base sera actualisée chaque heure et de manière transparente. Cette configuration est parfaite pour un serveur par exemple, mais elle est également adaptée à un poste de travail et elle ne ralentit pas votre machine.

Pour tester votre antivirus ClamAV, il suffit juste de mettre un virus sur votre PC... C'est un peu radical comme méthode, nous allons nous contenter d'un « faux virus », il s'agit du test EICAR (du nom de European Institute for Computer Antivirus Research). De plus, vous allez créer ce virus vous-même, vous allez devenir un vrai méchant. Créez un fichier */tmp/eicar* que vous complétez comme ceci :

```
X5O!P%@AP[4\PZX54(P^)7CC)7} ↵
   $EICAR-STANDARD-ANTIVIRUS- ↵
   TEST-FILE!$H+H*
```

Le test virus est maintenant réalisé, c'est l'instant de vérité, nous allons savoir si votre ClamAV est totalement actif, sinon, pas d'inquiétude, ce faux virus n'effacera que la totalité de vos données personnelles... Non, nous plaisantons, voici la description de EICAR selon Wikipedia : « Ce fichier ne contient pas de virus mais une signature qui doit être détectée par le logiciel antivirus si celui-ci est basé sur une méthode de recherche par signature ». Bien, vous voilà rassuré, passons au test :

```
clamscan --bell -r -i --log=/ ↵
   tmp/mesvirus.log /tmp/eicar
```

Si vous avez entendu un joli bruit, c'est que le fichier a été détecté, d'ailleurs, ClamAV a dû vous l'afficher à l'écran :

```
/tmp/eicar:   Eicar-Test-   ↵
   Signature FOUND
```

Vous pourrez toujours retrouver ce message dans le fichier de log qui a été créé pendant ce scan (*/tmp/mesvirus.log*).

Puisque la ligne de commande n'est pas votre fort, nous allons vous mettre à l'aise en installant une interface graphique pour ClamAV. Elle aussi se trouve dans les dépôts Debian :

```
apt-get install clamtk
```

Grâce à cette interface graphique, vous scannerez des fichiers directement d'un clic droit dans Nautilus (l'explorateur de fichier). Pour cela, installez une extension supplémentaire :

```
apt-get  install  nautilus- ↵
   clamscan
```

N'oubliez pas redémarrer Nautilus pour que l'extension soit bien prise en compte. Désormais, votre poste de travail est à l'abri des virus.

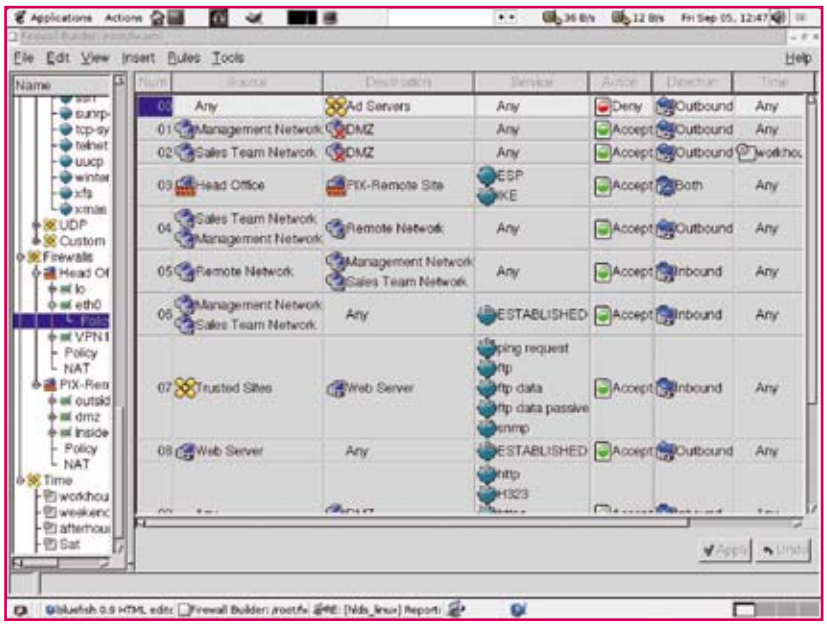

Figure 5. *Fwbuilder peut vous aider à utiliser iptables de manière graphique*

Un vigile à l'entrée

Pourquoi n'y a-t-il plus de bagarre le samedi soir dans les boîtes de nuit ? Tout simplement parce qu'un vigile filtre l'entrée. Nous allons utiliser le même principe pour votre serveur mais en faisant plus discret.

D'abord, le moyen le plus simple consiste à fermer les portes qu'il n'est pas nécessaire de garder ouvertes. Moins il y aura d'accès, plus votre PC sera sécurisé. Nous allons donc utiliser iptables, et là pas d'installation car il s'agit d'une application intégrée au noyau Linux. Ce logiciel vous permettra de créer un véritable pare-feu.

Toutefois, cette application fonctionne en ligne de commande, et nous ne vous conseillons pas forcément d'utiliser une interface graphique. Pour ce logiciel, il vaut mieux comprendre ce que vous faites.

Le principe que nous allons mettre en œuvre est très simple : en premier lieu, nous bloquons tous les accès vers la machine et ensuite, nous ouvrons seulement les portes devant l'être.

Créez le fichier */etc/init.d/monfirewall*, nous le compléterons progressivement, en expliquant chaque section rajoutée. D'abord, nous créons l'en-tête du fichier et nous nous assurons que d'autres règles ne viennent pas perturber ce que nous voulons réaliser. Recopiez ces lignes en début de fichier :

```
#!/bin/sh
iptables -F
iptables -X
```

Comme indiqué précédemment, nous bloquons toutes les connexions entrantes et sortantes, en ajoutant ces lignes :

```
iptables -t filter -P INPUT DROP
iptables -t filter -P FORWARD DROP
iptables -t filter -P OUTPUT DROP
```

Maintenant que nous avons tout bloquer, ajoutons une règle pour ne pas filtrer les connexions déjà établies :

```
iptables -A INPUT -m state ↵
   --state RELATED,ESTABLISHED ↵
   -j ACCEPT
```

```
#!/bin/sh
iptables -F
iptables -X
iptables -t filter -P INPUT DROP
iptables -t filter -P FORWARD DROP
iptables -t filter -P OUTPUT DROP
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -t filter -A INPUT -i lo -j ACCEPT
iptables -t filter -A OUTPUT -o lo -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 53 -j ACCEPT
iptables -t filter -A OUTPUT -p udp --dport 53 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 53 -j ACCEPT
iptables -t filter -A INPUT -p udp --dport 53 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 80 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 443 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 443 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 22 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 22 -j ACCEPT
```

Figure 6. *Contenu du fichier « monfirewall »*

```
-----BEGIN DSA PRIVATE KEY-----
MIIBuwIBAAKBgQDaZkzi8o5Nl2lxWPb0DqezwNNWFnwIQERp/EQmNVZHdKxv7YRh
oxdL/slqmcXQJBhjfBjb8W/unHWSuxm5C4j5B0BkqWSt8AuULvs8sOoyN7M+bhnX
f9YLfXu3dVfO5ujv8u/TmmD0YnAdfGlpkjBsoulNI7xA8cj+NexwkYRU4QIVADee
oB7lovLElwb1c4F+iuTnrLXpAoGAZxFJgZpM2QC15q8njm1cxnil4IdEOO+zN6YF
xUSowTOPUydcQsQoAvAoyWI53vgc4TCOpe8teabbnVpv4H06Xo/aQHPGbxkVHP82
05xnz4YUkDjS0TAkniCspnZfMPPUI1AN6r9tn6L6Od13X6q9zu9ZL35a+1d1q97/
27nLMtoCgYBi1OV8LBVkRfOIlxiCwy+DsjyYpLcM2QsR1T5bpsaZjN335qrn1Mxr
v0pgu4nZPEzuOHOvUMvfQWhGFAbIMNdLOU6youlFjLnKGoGKhvMVr+23I2c3nCvB
vqfMAAkgOqVnK34Cfzi9//NtDYUrXQTZn3kbOEI90zWz3fxEsvw0ZAIVAOSubDFM
tt06jnjTOepx1laJhBDO
-----END DSA PRIVATE KEY-----
```

Figure 7. *Exemple d'une clé privée*

```
iptables -A OUTPUT -m state ↵
    --state RELATED,ESTABLISHED ↵
    -j ACCEPT
```

Dans ce qui suit, nous demanderons à iptables de ne pas bloquer tout ce qui vient de l'interface loopback. Cette interface permet au serveur de se contacter lui-même quand il est appelé par l'adresse 127.0.0.1 (localhost) par exemple.

```
iptables -t filter -A INPUT -i ↵
    lo -j ACCEPT
iptables -t filter -A OUTPUT ↵
    -o lo -j ACCEPT
```

Il faut également qu'iptables accepte les requêtes DNS (protocole qui permet d'associer un nom de domaine à une IP) :

```
iptables -t filter -A OUTPUT ↵
    -p tcp --dport 53 -j ACCEPT
iptables -t filter -A OUTPUT ↵
    -p udp --dport 53 -j ACCEPT
iptables -t filter -A INPUT -p ↵
    tcp --dport 53 -j ACCEPT
iptables -t filter -A INPUT -p ↵
    udp --dport 53 -j ACCEPT
```

Ensuite, c'est un peu du cas par cas, tout dépend de ce que vous voulez faire avec votre serveur. Pour chaque utilisation, il faudra ajouter une règle. Prenons un exemple, vous voulez aller sur internet, ce qui signifie utiliser les ports 80 (http) et 443 (https), il faudra donc débloquer ces ports en sortie (de votre PC vers internet). Voici les règles que vous pourriez ajouter :

```
iptables -t filter -A OUTPUT ↵
    -p tcp --dport 80 -j ACCEPT
iptables -t filter -A OUTPUT ↵
    -p tcp --dport 443 -j ACCEPT
```

À contrario, si votre PC héberge un site web, il faudra autoriser les gens à s'y connecter, donc créer une règle pour ne pas filtrer les ports 80 et 443 (d'internet vers votre PC) :

```
iptables -t filter -A INPUT -p ↵
    tcp --dport 80 -j ACCEPT
iptables -t filter -A INPUT -p ↵
    tcp --dport 443 -j ACCEPT
```

Pour chaque protocole que vous utilisez, il vous faudra définir une règle en entrée et une règle en sortie. Si vous ne connaissez pas les ports utilisés par vos applications, vous pouvez toujours regarder dans le fichier */etc/services*, vous trouverez le numéro de port ainsi que le protocole utilisé (icmp, tcp ou udp). Voici un dernier exemple pour accepter les connexions SSH mais aussi pour utiliser SSH en tant que client :

```
iptables -t filter -A INPUT -p ↵
    tcp --dport 22 -j ACCEPT
iptables -t filter -A OUTPUT ↵
    -p tcp --dport 22 -j ACCEPT
```

Dès que votre fichier est prêt, il faut le rendre exécutable et ensuite ordonner au système qu'il le prenne en compte à chaque démarrage :

```
chmod +x /etc/init.d/ ↵
    monfirewall
update-rc.d monfirewall ↵
    defaults
```

Ne vous inquiétez pas pour le warning, c'est simplement que l'entête du fichier ne respecte pas les standards. Mais il fonctionnera, redémarrez votre machine et ensuite, tapez `iptables -L`, si tout s'est bien passé, vous devriez voir la liste de vos règles sur l'écran : vous venez de créer un firewall !

Accès Autorisé

Vous avez déjà vu dans les films d'espionnage, trois vaines tentatives de connexion à un système hyper sensible de la CIA place toute l'armée américaine en branle-bas de combat. Ici encore, nous nous appuierons sur le même principe : en effet, autoriser l'accès à certains ports est bien, encore faut-il gérer la sécurité après.

Donc, après trois vaines tentatives de connexion à notre serveur, nous bannirons temporairement l'IP qui tente ces connexions. C'est simple et efficace. Pour cela nous allons utiliser fail2ban :

```
apt-get install fail2ban
```

Vous modifierez la configuration de cette application et ajouterez éventuellement d'autres services à surveiller en éditant le fichier */etc/fail2ban/fail2ban.conf*.

La clé du SSH

SSH est une application indispensable pour l'administration à distance, le problème est qu'elle ouvre une porte sensible vers votre machine. Nous allons donc essayer de mettre quelques verrous supplémentaires sur cette porte.

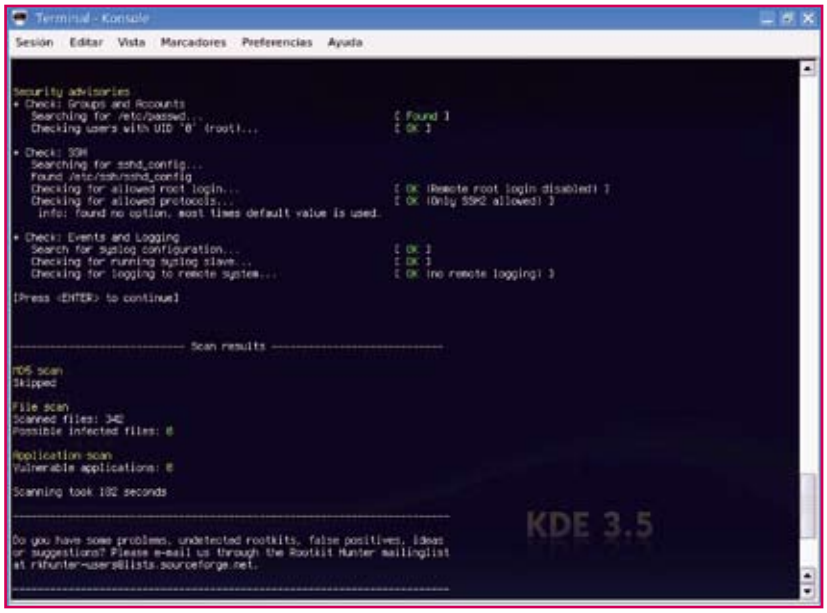

Figure 8. *rkhunter en action*

Habituellement, nous nous connectons à SSH via un couple login / password, mais si quelqu'un dérobe notre mot de passe, il a accès à notre serveur. Nous allons, donc, travailler avec une authentification par paire de clés (clé publique et clé privée). Dans un premier temps, nous générons ces deux clés :

```
ssh-keygen -t dsa -b 1024 -f / ↵
    tmp/macle_dsa
```

Normalement, le système doit vous demander une passphrase, il s'agit d'un mot de passe qui sera associé à votre clé. Il peut être différent du mot de passe de votre utilisateur Linux. Un conseil certainement très utile : souvenez-vous en ! En effet, de nombreuses personnes doivent être dépannées pour avoir simplement oublié leur mot de passe.

Figure 9. *Votre système est entre de bonnes mains avec ce genre de garde du corps ;)*

Autre conseil, n'écrivez pas votre mot de passe sur un papier caché sous votre clavier : cette cachette n'est pas du tout originale ;)

Il est possible de ne pas mettre de mot de passe avec votre clé, l'intérêt étant de se connecter directement à votre machine ou d'exécuter des scripts à distance. Vous seul connaissez le niveau de sécurité dont vous avez besoin.

Maintenant, copiez la clé au bon endroit ; si vous l'avez générée sur votre machine, l'IP sera simplement 127.0.0.1 (localhost) sinon, il vous faudra renseigner la bonne IP.

```
ssh-copy-id -i /tmp/macle_dsa. ↵
    pub monutilisateur@127.0.0.1
```

Sauvegardez impérativement votre paire de clés (*macle_dsa* et *macle_dsa.pub*) générée sinon, toutes ces manipulations ne serviraient à rien.

Dernière étape, modifiez la configuration de SSH pour que l'authentification par mot de passe soit désactivée. Éditez le fichier */etc/ssh/sshd_config* et remplacez la ligne suivante :

```
#PasswordAuthentication yes
```

par :

```
PasswordAuthentication no
```

Redémarrez SSH pour prendre en compte les modifications (*/etc/init.d/ssh restart*). Désormais, pour vous connecter à distance à votre serveur, il vous faudra utiliser cette ligne de commande :

```
ssh -i macle_dsa.pub ↵
    monutilisateur@monserveur.com
```

Bien entendu, il y a moyen d'automatiser cela avec des outils comme putty, par exemple, qui sont multi-plate-forme. La seule obligation est de sauvegarder votre pair de clés !

Trop tard

Vous êtes intervenu trop tard, dommage, quelqu'un a réussi à entrer dans votre système ! Ce n'est pas grave, vous l'avez détecté grâce à rkhunter (à ne pas confondre avec *Rick Hunter*). Il s'agit d'un détecteur de rootkit qui vérifie chaque jour si l'empreinte md5 de vos applications a été modifiée.

```
apt-get install rkhunter
```

Le paramétrage par défaut est parfait, nous allons juste exclure quelques éléments pour éviter d'avoir des messages d'alerte qui n'en sont pas. Éditez le fichier */etc/rkhunter.conf* et ajoutez ces lignes :

```
ALLOWHIDDENDIR=/dev/.udev
ALLOWHIDDENDIR=/dev/.static
```

C'en est terminé pour rkhunter, normalement plus aucun rootkit ne devrait vous surprendre. Le secret est tout simplement de mettre cette application le plus tôt possible sur votre système.

Seul face au monde

À vous de jouer maintenant, votre système est paré pour bon nombre d'attaques, mais souvenez-vous que l'élément le plus fragile de la chaîne de sécurité et bien c'est... VOUS ! Donc, pour continuer à avoir un système sûr, un seul conseil : du bon sens ! Bref, changez immédiatement le mot de passe de root car, sérieusement, « toto » n'est pas un mot de passe sécurisé ;)