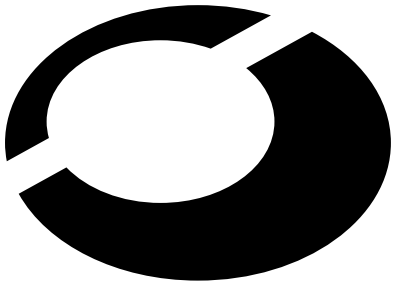
Filtrage de paquets sous GNU/Linux

Filtrage et politique de sécurité



Alcôve

http ://www.alcove.com/

Benjamin Drieu, Alcôve France <Benjamin.Drieu@fr.alcove.com>

06 octobre 2003

15, avenue de l'Agent Sarre
92700 Colombes
France
Tel : +33 1 46 49 25 00
Fax : +33 1 46 49 25 01

Résumé

Ce livre blanc décrit une solution informatique de sécurité basée sur le filtrage de paquets. Les technologies mises en oeuvre permettent de garantir la sécurité d'un réseau informatique par l'analyse des flux de données transitant.

Après avoir présenté succinctement les enjeux de la sécurité informatique, ce livre blanc décrit le rôle de la pièce maîtresse d'un système de sécurité réseau, le filtre de paquets. Puis, il détaille les solutions de filtrage pour GNU/Linux et situe le rôle du filtre de paquets au coeur d'un dispositif de sécurité global. Enfin, une liste de logiciels libres afférents à la sécurité est décrite en annexe de ce document.

Version 1.3
Réference : Filtrage de paquets sous GNU/Linux

Table des matières

Les enjeux de la sécurité

La mise en réseau et la circulation des informations sont aujourd'hui essentielles pour une entreprise. Pour autant, le déploiement de solutions réseau ne saurait se faire sans évaluer les risques inhérents et sans prendre les mesures adéquates. En effet, la compromission de la sécurité d'un réseau d'entreprise peut avoir des conséquences dramatiques sur le court comme sur le long terme :

- fuite de données confidentielles ;
- compromission ou perte de données ;
- utilisation des ressources de la société à des fins illégales ;
- plus généralement, perte de ressources et de productivité.

L'intégrité des données de l'entreprise passe donc par la définition d'une politique de sécurité stricte, tenant compte des réalités de production et de circulation d'informations tout comme des risques extérieurs ou intérieurs. La première étape est de se connaître et de connaître l'extérieur.

1 Se connaître

Se connaître est la première étape de la définition d'une politique de sécurité. Elle implique d'identifier :

- les utilisateurs de son réseau, leurs habitudes, leurs besoins et les vulnérabilités qu'ils pourraient introduire dans la sécurité du réseau (par exemple : ont-ils accès à certains mots de passe ? Peuvent-ils mettre en place un tunnel SSH vers l'extérieur ?). Les utilisateurs n'ont pas besoin d'être malicieux pour adopter un comportement dangereux pour l'ensemble de la sécurité du réseau ;
- les logiciels déployés, en particulier leurs versions. L'administrateur peut mettre à profit cette connaissance pour savoir si les failles de sécurité connues s'appliquent à son réseau, pour connaître les défaillances logicielles, pour savoir quels postes sont vulnérables à un virus, etc. ;
- la topologie de son réseau, afin de déterminer quelles sont les zones exposées et les ériger en bastions, tout en gardant à l'esprit qu'une zone non protégée, même peu exposée, peut compromettre la sécurité globale ;
- les flux de données transitant par le réseau, afin d'en déterminer la criticité et les éventuels points d'entrée pour les attaquants.

2 Connaître l'extérieur

Comme tout projet informatique, la sécurité informatique d'un réseau se dimensionne. S'il est possible d'estimer facilement la charge d'un projet d'infrastructure, un projet de sécurité demande de connaître l'environnement extérieur pour estimer les risques de sécurité et ainsi l'investissement nécessaire à l'établissement de la politique de sécurité.

Notamment, il est nécessaire d'estimer la motivation des pirates informatiques à pénétrer le réseau. Cela peut être par exemple par jeu, par appât du gain (les informations d'un réseau peuvent être monnayables) ou par besoin de posséder une passerelle pour lancer de nouvelles attaques ou un déni de service réparti (voir la Section 4.2.4). De l'estimation de cette motivation, on peut facilement déduire les efforts de sécurité à effectuer.

La motivation et la nature des pirates informatiques sont très souvent mal connues et mésestimées. Aujourd'hui, les outils d'intrusion se sont démocratisés (des outils permettant d'exploiter des failles connues sont librement téléchargeables sur le web), les failles informatiques sont connues de tous et il n'est plus nécessaire de posséder de compétences extrêmement avancées pour opérer une intrusion dans un système informatique. De même, les motivations des pirates ne sont pas forcément tournées contre le système qu'ils pénètrent, dans beaucoup de cas, les pirates sondent des systèmes au hasard pour exploiter les failles le cas échéant et utiliser les ressources acquises dans le but de commettre des actes illégaux (piratage via une passerelle, déni de service, etc.). Ainsi, un système informatique qui présenterait des failles évidentes ou connues serait extrêmement vulnérable.

Le filtre de paquets en première ligne

Comme dans la vie réelle, la méthode la plus simple pour protéger un réseau est d'en contrôler les accès en entrée et en sortie. Dans un réseau informatique, le transport des informations étant effectué par TCP/IP, c'est à ce niveau que la majorité des contrôles se fera. Il est donc nécessaire de mettre en place un mécanisme de contrôle des flux TCP/IP, ce qui est notamment effectué par le filtre de paquets.

1 Principes du filtre de paquets

Un filtre de paquets (ou pare-feu ou *firewall*) est une machine capable de prendre la décision d'accepter ou non un paquet transitant par une de ses interfaces réseau. On l'appelle *filtre de paquets* car cette fonctionnalité permet de définir des filtres qui isolent les accès indésirables des accès autorisés.

Le filtrage des paquets intervient lors du traitement de paquets transitant par le filtre. Le filtre de paquets analyse les paramètres de chaque paquet et prend une décision sur le devenir du paquet en fonction de ces paramètres et de la politique de sécurité. Il est capable de combiner ces filtres et de les factoriser, ce qui est formalisé dans des *règles*. Lorsqu'un paquet est traité, le filtre de paquets applique l'une après l'autre les règles qu'on lui a configurées, jusqu'à ce que l'une corresponde au paquet analysé. Pour un paquet donné, il peut par exemple analyser :

- le protocole utilisé (TCP, UDP, ICMP, ...) ;
- l'adresse IP de la machine émettrice ou l'adresse IP de la machine de destination ;
- pour TCP ou UDP, le service du paquet (HTTP, SMTP, SSH, domain...) ;
- pour ICMP, le type ICMP (*destination-unreachable*, *echo-reply*, ...).

Une fois un paquet analysé, le filtre de paquets prend la décision de l'accepter ou non, ce qui signifie exécuter une action parmi les suivantes :

- continuer à le traiter ;
- le rejeter, ce qui implique d'envoyer un paquet ICMP d'erreur à l'émetteur ;
- l'ignorer, ce qui signifie ne pas le traiter sans même en avertir l'émetteur.

Sur ce principe, le filtre de paquets peut par exemple décider d'ignorer tous les flux en provenance d'un réseau particulier, hormis le trafic HTTP et SMTP. Ignorer un

paquet sans émettre de message d'erreur peut être une bonne solution pour limiter les inspections de ports en provenance de l'Internet.

Un filtre de paquets peut être installé sur une machine pour la protéger des attaques de l'Internet ou des machines du réseau local si elle est particulièrement sensible. Mais le rôle principal du filtre de paquets est de contrôler les paquets qui le traversent pour passer d'un réseau à l'autre. Ainsi, le filtre de paquets peut protéger des machines placées derrière lui, même si ces machines ne comportent pas de mécanisme de filtre de paquets.

Les filtres de paquets modernes sont capables de tirer partie des mécanismes de session TCP pour factoriser le filtrage. Ainsi, les filtres de paquets peuvent se contenter de vérifier si le premier paquet d'une connexion TCP est autorisé, mémoriser cette décision et appliquer la même aux paquets suivants. La configuration du filtre de paquets est ainsi simplifiée, est plus sûre car elle limite le *spoofing*[1] et est plus rapide. De même, un filtre de paquets est capable de factoriser plusieurs flux en une seule session pour les protocoles non plats comme FTP[2].

2 Netfilter

Netfilter est un filtre de paquets intégré aux noyaux Linux de la série 2.4 ou supérieure. *Netfilter* possède des fonctionnalités avancées de filtrage de paquets et de traduction d'adresses et de ports (*NAT*).

Netfilter étant intégré au noyau Linux, il permet de manipuler les paquets le plus tôt possible dans la chaîne de traitement. Ainsi, il est capable de les accepter, de les refuser ou même de les modifier avant que la couche applicative ne les traite. Le fonctionnement en mode noyau augmente pareillement la sécurité du processus et sa performance.

La configuration de Netfilter s'effectue par le biais de la commande *iptables*, dont la syntaxe est proche de celle de la commande *ipchains*, le système de filtrage de paquets du noyau Linux 2.2. Il existe des outils de configuration de netfilter, comme fwbuilder (http ://fwbuilder.sourceforge.net/), knetfilter (http ://expansa.sns.it :8080/knetfilter/) ou firehol (http ://firehol.sourceforge.net/). De la même manière, plusieurs outils d'analyse de logs permettent d'effectuer des rapports efficaces de l'activité du filtre de paquets, comme wflog.

La principale innovation de Netfilter sur ses prédécesseurs est l'introduction de la notion de *session connection tracking* (c'est-à-dire le traitement de l'état d'une

1. Le *spoofing* est une technique d'attaque courante qui consiste à contrefaire une adresse IP pour obtenir des privilèges particuliers.
2. Le protocole FTP nécessite une connexion de commande et une connexion de données établie dans la connexion de commande. Cette connexion de données peut être établie en provenance du client vers le serveur ou en provenance du serveur vers le client, au choix du client.

connexion), ce qui permet par exemple d'autoriser uniquement le passage de paquets relatifs à une connexion valide et déjà établie.

3 Placer un filtre de paquets

Un filtre de paquets doit se placer comme on placerait un poste de douane entre deux régions : au point de passage obligé. En effet, pour que l'action filtrante soit réellement efficace, il est nécessaire que des attaquants ne puissent pas faire passer des paquets d'un point protégé à un point non protégé sans passer par le filtre de paquets.

3.1 Principe du découpage

La méthode la plus efficace est de diviser son réseau en sous-réseaux logiques, n'ayant pas de contacts directs entre eux. En chaque réseau, un filtre de paquets se charge de contrôler les flux. Un filtre de paquets sert à décomposer les risques, ainsi des machines particulièrement exposées doivent être isolées des autres, grâce à la découpe en sous-réseaux.

Le filtre de paquets doit être positionné comme point d'entrée (passerelle) de chaque réseau, ce qui garantit que tout paquet entrant ou sortant est analysé par le filtre. Bien entendu, des mesures physiques doivent être prises pour qu'il n'existe pas de passage alternatif (découpe des réseaux par *switchs*, etc.).

En outre, il est naturel d'allier les capacités de filtrage des filtres de paquets à la traduction d'adresses (*network address translation*). En effet, une passerelle est une machine chargée de réécrire les adresses des paquets qui transitent par elle, permettant de relier deux réseaux entre eux sans que l'un ou l'autre connaisse l'adresse IP réelle d'une des deux parties. Un des avantages est la possibilité de définir des réseaux privés avec un plan d'adressage connu uniquement de la passerelle et donc forcer le trafic entrant et sortant de ce réseau à passer par la passerelle. La passerelle étant un point de passage obligée, elle peut ainsi décider seule des flux à laisser passer ou à bloquer sans que les émetteurs n'aient la possibilité de passer outre.

3.2 Le découpage en DMZ

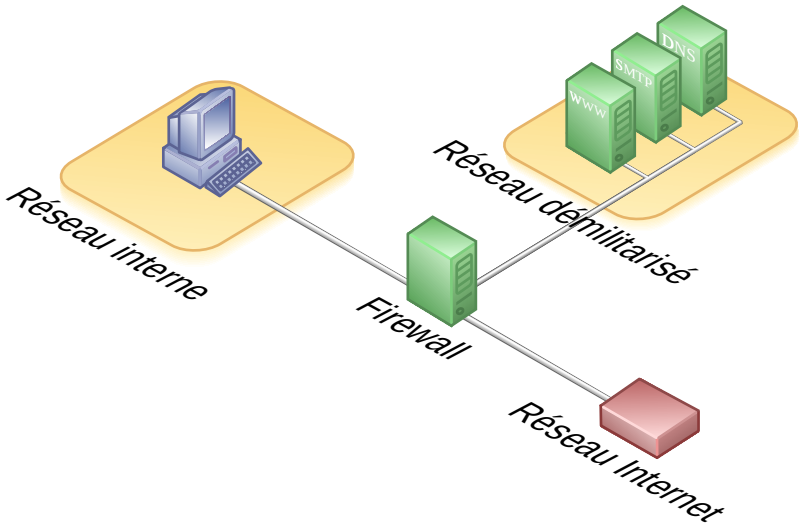
Le découpage en *DMZ* (pour *zone démilitarisée*) est un cas particulier de découpage en réseaux communément utilisée pour les réseaux ayant des besoins mixtes de sécurité interne et de visibilité de l'extérieur.

3.2.1 Principe

Le principe du découpage en DMZ (pour *DeMilitarised Zone* ou *zone démilitarisée*) est de séparer les réseaux en trois sous-réseaux ou plus, selon qu'il s'agisse :

- d'un réseau *interne*, qui contient des données privées qui ne doivent pas être consultées de l'extérieur. C'est le réseau contenant les postes clients. Ce réseau peut en revanche avoir besoin d'accéder à l'extérieur, par exemple au web, via un proxy. Il s'agit du *réseau privé* ;
- d'un réseau *externe*. Par exemple l'Internet, qu'on considère peuplé de volontés malfaisantes. Ce réseau ne doit pas avoir le droit d'accéder au réseau interne et idéalement ne doit même pas le voir (grâce à la traduction d'adresses) ;
- d'un réseau avec un besoin mixte d'être visible du réseau interne et du réseau externe. C'est par exemple le cas d'un réseau contenant un serveur web, qui doit pouvoir être visible de l'Internet mais aussi des utilisateurs internes. Une passerelle SMTP doit également être accessible des postes clients mais aussi des serveurs SMTP de l'Internet. Il s'agit de la *DMZ*.

Figure 1. Schéma classique de DMZ



Le mécanisme de la DMZ permet de séparer clairement les relations entre les différents réseaux et de minimiser les risques en cas de compromission des machines visibles de l'Internet. La DMZ permet de découper efficacement son réseau en zones de sensibilité différente. Le gain est évident en terme de sécurité car le cloisonnement des réseaux permet de stratifier une stratégie de défense en profondeur : un attaquant obtenant un accès sur une machine visible publiquement par l'exploitation d'une faille de sécurité peut attaquer à partir de là les machines de la DMZ, mais pas celles du ré-

seau interne. De la même manière, le découpage en DMZ permet de limiter les accès aux serveurs de la DMZ et ainsi les protéger d'un utilisateur malveillant du réseau interne (employé mécontent, etc.)

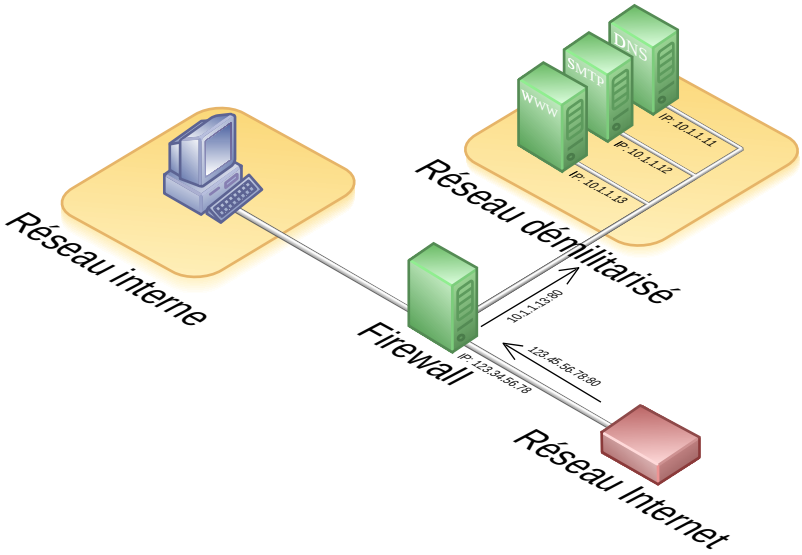
3.2.2 L'adressage

Il existe plusieurs solutions pour résoudre le problème de l'adressage des serveurs de la DMZ. En effet, il convient de séparer les différents réseaux en terme d'adressage IP, mais également d'être économe en terme d'adresses IP publiques. Plusieurs solutions se dégagent.

La première solution consiste à couper la plage d'adresses IP publiques dédiées en deux parties (pas forcément égales). La première moitié sera dédiée à l'interface publique du filtre de paquets et la deuxième aux machines de la DMZ. Ainsi, le filtre de paquets est capable de séparer les deux réseaux en se basant sur leur masque de réseau. Cette solution peut cependant se révéler coûteuse en termes d'adresses IP.

La deuxième solution consiste à utiliser les fonctionnalités de traduction d'adresses (*Network Address Translation*) du filtre de paquets et à lui affecter toutes les adresses IP publiques sur son interface publique (ou *patte publique*). Des règles de traduction d'adresses permettent au filtre de paquets de transmettre tout le trafic qu'il recevra sur une interface réseau donnée à une machine donnée de la DMZ (des distinctions peuvent également être faites par port). Ces serveurs de la DMZ auront quant à eux des adresses privées. Ainsi, seul le filtre de paquets est visible de l'Internet et il est possible de placer plusieurs serveurs dans la DMZ avec une seule adresse IP visible, la distinction entre un service et un autre se faisant par l'examen du port de destination. Il est à noter que cette solution n'est pas satisfaisante si les services de la DMZ sont importants en nombre ou en volume, car cette technique entraîne une montée de la charge du filtre de paquets.

Figure 2. DMZ avec adressage privé



3.2.3 Placement des services

Si le placement de certains services est évident (postes clients, serveurs web publics, etc.), certains services ont un rôle mixte et nécessitent une réflexion plus approfondie.

C'est le cas du serveur SMTP. Fonctionnellement, il a à la fois un rôle de messagerie interne, d'envoi de courriers vers l'Internet et enfin de réception de courriers en provenance de l'Internet. Deux schémas sont à envisager :

- le serveur de messagerie est placé en DMZ. Il reçoit et envoie les courriers directement de l'Internet. Les postes clients lisent leur courrier via POP3 ou IMAP directement sur ce serveur. La messagerie interne passe également par ce serveur. Ainsi, les messages électroniques internes sont stockés sur ce serveur, ce qui a pour conséquence de placer des données confidentielles sur un serveur potentiellement accessible de l'Internet ;
- le serveur de messagerie est dupliqué: un serveur de messagerie sur le réseau interne se charge de délivrer les courriers internes et de conserver les messages. Il relaie les courriers à destination de l'Internet à un serveur de messagerie placé sur la DMZ. Ce dernier est le seul à communiquer avec l'Internet et relaie les courriers en provenance de l'Internet au serveur du réseau interne. Les postes clients communiquent via POP3 ou IMAP avec le serveur de messagerie du réseau Interne. Ainsi, aucun courrier interne ne passe par la DMZ et les mots de passe nécessaires à l'authentification POP3/IMAP ne transitent pas par la DMZ. Il est alors par exemple possible de les stocker dans un annuaire LDAP interne.

Le serveur de nom (*Domain Name Server*) est soumis au même principe. Il doit effectuer à la fois le travail de résolution des noms de machines internes et le travail de résolution des noms des machines Internet. De même, le serveur DNS doit répondre aux requêtes des machines de l'Internet quant aux noms des machines disponibles publiquement de la DMZ (par exemple, *www.mondomaine.com*). La solution la plus simple consiste à placer un serveur DNS sur la DMZ pour répondre aux requêtes de l'Internet et un autre serveur DNS dans le réseau Interne, qui est l'autorité pour les machines internes et qui passe par le DNS public pour résoudre les noms de machines de l'Internet.

3.2.4 Les DMZ multiples

Certains schémas de partitionnement de réseau peuvent faire appel à plusieurs DMZ. C'est le cas lorsqu'il est nécessaire de rendre publics certains serveurs mais qu'il n'est pas possible en terme de sécurité de les faire cohabiter sur le même réseau. Notamment, cela prend du sens si un service est plus sensible que d'autres et qu'il n'est pas désirable de l'exposer aux attaques dans l'hypothèse où un autre service de la DMZ serait compromis.

De la même manière, il est parfois souhaitable de placer les services du réseau interne sur une DMZ « interne », où se trouveraient les serveurs de messagerie interne, le DNS interne, l'annuaire LDAP s'il en existe, etc.

4 Défendre le filtre de paquets

Le filtre de paquets est, on l'a vu, une pièce essentielle du dispositif de défense d'un réseau. Il contrôle et valide les flux entre les différents points du réseau. Ainsi, si le filtre de paquets est compromis, c'est toute la stratégie de défense du réseau qui est anéantie : les machines sans protection particulière se retrouvent exposées aux attaques de l'extérieur. Le filtre de paquets doit être, avec des serveurs de bases de données critiques, le serveur le mieux protégé des attaques.

4.1 La protection du filtre de paquets

La règle d'or pour défendre le filtre de paquets est la règle du *small is beautiful*. Cette règle est motivée par le constat qu'aucun logiciel n'est à l'abri d'une faille de sécurité. Ainsi, tous les services non nécessaires doivent être supprimés ou désactivés : le filtre de paquets ne doit pas contenir de serveur web, de serveur de messagerie, etc. S'il doit envoyer des courriers électroniques, il doit le faire directement par SMTP sans passer par un agent de transport. En suivant ce principe, les règles doivent être le plus simple

possible afin d'éviter les effets de bord. Idéalement, le filtre de paquets ne doit pas comporter de serveur d'accès à distance et n'être accessible que de la console. Pour des besoins pratiques, un serveur d'accès à distance chiffré comme SSH est acceptable à condition d'être lui-même protégé par filtrage de paquets et restreint à quelques adresses identifiées.

Des systèmes de détection d'intrusions peuvent compléter le dispositif de sécurité afin de protéger le filtre de paquets. Ces systèmes peuvent, soit auditer le système de fichiers du filtre de paquets, comme c'est le cas de *tripwire*, soit auditer les flux réseau en provenance de l'extérieur, comme c'est le cas de *snort*. Il est à noter que les systèmes de détection d'intrusions sont des programmes comme les autres et sont donc susceptibles de comporter des failles de sécurité[3], ils doivent donc être placés de manière à ce que leur compromission éventuelle n'affecte pas la sécurité du réseau. Voir la Section 1 pour plus d'informations sur les systèmes de détection d'intrusions.

Les systèmes de détection d'intrusions fonctionnant sur le filtre de paquets sont également potentiellement attaquables et ne doivent pas être considérés comme un système de sécurité absolue. En effet, ils se contentent de *surveiller* le filtre de paquets. Une fois ce dernier compromis, rien ne dit que les systèmes de détection d'intrusions ne soient pas eux aussi compromis. Ainsi, ils peuvent continuer à fonctionner apparemment normalement et couvrir les agissements d'un attaquant. Ils ont donc un rôle purement *indicatif*, qui n'a d'intérêt que si l'administrateur examine régulièrement leurs rapports ainsi que ceux du filtre de paquet lui-même.

De même, les fichiers de rapport du filtre de paquets ne doivent pas être conservés sur la machine locale, faute de quoi un attaquant ayant gagné un accès privilégié sur la machine a la possibilité de les effacer. L'idéal est d'utiliser les fonctionnalités de *syslog* pour envoyer les données du système à distance, sur une machine particulière qu'un pirate devra aussi attaquer s'il veut effacer ses traces.

Il est à noter que certains projets issus de l'informatique libre ont pour objectif de tirer partie de l'extraordinaire souplesse du système GNU/Linux et de mettre en place des systèmes de filtre de paquets sur CDROM ou en mémoire. L'intérêt d'un filtre de paquets sur CDROM est l'impossibilité de compromission physique du système. Un exemple de tel projet est le projet Gibraltar Firewall (http ://www.gibraltar.at/), basé sur Debian GNU/Linux.

4.2 Types d'attaques

Le filtre de paquets étant en premier ligne, il va subir un ensemble d'attaques, à sa destination propre ou à destination des machines du réseau qu'il protège.

3. Ainsi, Snort a récemment été victime d'une faille de sécurité permettant l'exécution de commandes arbitraires à distance, comme le témoigne une alerte publiée sur le site de Snort. (http ://www.snort.org/advisories/snort-2003-04-16-1.txt)

4.2.1 Attaques applicatives

Les attaques applicatives « classiques » tentent d'exploiter des failles dans les services réseaux activés sur le filtre de paquets ou sur les machines visibles derrière lui. Le filtre de paquets ne peut pas prévenir ces attaques si elles concernent des services autorisés. En revanche, il peut limiter le nombre de services autorisés. De même, il n'est pas censé être lui-même accessible de l'extérieur via un service réseau.

Pour autant, il est essentiel de faire profiter le filtre de paquets des mises à jour de sécurité fournies par la distribution avec laquelle il est installé. Une distribution comme Debian GNU/Linux fournit des paquets correctifs (http ://www.debian.org/security/) signés pour toutes les alertes de sécurité reportées.

4.2.2 IP spoofing

Le *spoofing* est une attaque où une machine tente de se faire passer pour une autre en contrefaisant son adresse IP. Le but de cette attaque est de tenter de faire passer des paquets pourtant invalides à travers le filtre de paquets. Un filtre de paquets comme *netfilter* est capable de parer ces attaques grâce au mécanisme du *connection tracking*, c'est-à-dire par l'analyse des connexions TCP en cours et par la mise en place de quelques règles de filtrage spécifiques. L'adjonction d'outils de surveillance ARP (comme *arpwatch*) rend le filtre de paquets plus résistant à de telles attaques.

4.2.3 Déni de service

Si la détection et l'exploitation de failles demande des connaissances particulières, un autre type d'attaque permet de gêner le réseau cible sans pour autant causer d'intrusion : le *déni de service* ou *Denial Of Service* (DOS). Il s'agit non pas de s'introduire dans le système mais de le bloquer par la congestion.

Ainsi, les utilisateurs de ce système ne pourront y accéder, ce qui peut causer des pertes de productivité. Un déni de service classique consiste à envoyer des demandes répétées à un service sans attendre de retour[4].

Un filtre de paquets arrive normalement trop loin dans la chaîne pour éliminer totalement les dénis de service, mais il peut limiter les dégâts dans le système qu'il protège. Notamment, il peut être capable de faire de la *qualité de service*[5]pour conserver un service minimal sur les flux critiques. En outre, si l'émetteur du déni de service est

4. Par exemple, cela peut passer par l'envoi de paquets ICMP (*ping flood*) ou de requêtes HTTP. La réponse à une requête HTTP étant la plupart du temps plus coûteuse et plus grande que la requête elle-même, ce type de déni de service est très efficace.
5. La qualité de service appliquée aux flux réseau est un mécanisme de comptabilisation des paquets transitant par le filtre de paquets et de mise en place de quotas pour chacun des flux identifiés. Il est par exemple possible d'affecter une fraction minimale de la bande passante au flux SMTP et d'affecter le reste de la bande passante aux autres flux ou même d'identifier des réseaux partenaires à qui on affecte une fraction maximale ou minimale de bande passante.

identifié, le filtre de paquets peut interdire tous les flux en sa provenance et ignorer les paquets en question (par *portsentry*, par exemple).

4.2.4 Déni de service réparti

Une catégorie particulière de dénis de service bien plus dévastatrice émerge : le déni de service réparti. Dans ce cas particulier, le déni de service n'est pas conduit d'un point unique ou défini, mais au contraire, il est conduit par un ensemble important de machines à travers l'Internet. Dans la plupart des cas, les machines participant au déni de service réparti essayent de reproduire un comportement normal d'utilisation afin d'éviter d'être repérées. Ainsi, il devient difficile de séparer le bon grain de l'ivraie.

Les dénis de service répartis se basent la plupart du temps sur l'exploitation semi-automatique d'une faille de sécurité connue sur des serveurs relais à travers le monde. Cela peut être par l'infection d'un virus (c'est le cas de *Blaster*) ou par l'exploitation d'une faille distante (*Trinoo*, *TFN*, *Stacheldraht*, *Shaft*, etc.). Les pirates installent sur les machines vulnérables un logiciel serveur (un *rootkit*) qui leur permettra le moment venu de lancer une attaque simultanée sur un site particulier.

Il n'existe pas encore d'outils libres permettant de répondre efficacement aux dénis de service répartis, mais un ensemble d'outils et de pratiques permet de limiter leur portée. En particulier, l'administrateur doit prévoir une qualité de service permettant de faire fonctionner les services critiques en mode non dégradé, détecter les hôtes potentiellement infectés de son réseau, prévoir une architecture de répartition de charge, etc. Le *CERT* (http ://www.cert.org/) a notamment publié (http ://www.cert.org/archive/pdf/Managing_DoS.pdf) un livre blanc sur les méthodologies de défense contre les dénis de service répartis ou non répartis.

5 Haute disponibilité

Le filtre de paquets est ainsi un point névralgique dans le réseau qu'il protège, d'une part du point de vue de la sécurité et d'autre part du point de vue de la disponibilité de service. Un filtre de paquets qui tombe en panne peut isoler de l'Internet la totatité du réseau d'une entreprise. Ainsi, il convient de mettre en place des mécanismes de reprise d'erreur.

L'architecture la plus simple consiste à mettre en place une duplication du serveur faisant office de filtre de paquets avec une surveillance mutuelle entre le serveur et son double. Un serveur est actif et l'autre est une machine de secours. Lorsque le filtre de paquets actif tombe en panne, le serveur de secours prend automatiquement la main et peut continuer le service.

Les deux serveurs se surveillent en testant régulièrement la présence de l'autre en vérifiant que l'autre répond aux requêtes ICMP, ARP ou que l'autre est présent via une ligne série. Si le serveur actif ne répond pas après une période déterminée, la machine de secours le considère comme hors service et devient à son tour le serveur actif. Dans le cas d'un filtre de paquets, le serveur de secours prend la main en usurpant l'adresse IP du serveur actif et en envoyant une annonce ARP forgée. Une fois la panne résolue, le serveur de secours peut soit rendre automatiquement la main au serveur principal, soit devenir à son tour serveur principal jusqu'à ce qu'il tombe en panne ou que l'administrateur fasse la bascule manuellement (*nice failback*).

Une telle architecture peut se mettre en place grâce à plusieurs outils libres, comme *mon*, *fake* ou *heartbeat*. Elle garantie une bonne reprise sur panne et permet de garantir une interruption de service minimale, de l'ordre de quelques secondes. Elle peut cependant demander quelques adaptations de l'architecture réseau existante, comme par exemple le doublement des fournisseurs d'accès pour garantir une continuité de service maximum.

La sécurité en pelures d'oignons

Nous l'avons vu, le filtre de paquets est une pièce maîtresse de la stratégie de défense d'un réseau. Pour autant, si le filtre de paquets assure le contrôle des flux réseaux en inspectant les en-têtes des paquets, il n'est pas capable d'analyser leur contenu ni de comprendre les protocoles applicatifs sous-jacents. La sécurité d'un réseau découle de la planification d'une stratégie traitant de tous les aspects de l'utilisation des systèmes informatiques.

1 L'analyse des failles protocolaires

Les attaquants profitent la plupart du temps de failles de sécurité connues, présentes dans certaines implémentations de protocoles. Des méthodes d'exploitation de ces failles sont disponibles sur l'Internet et sont donc utilisables par tous. Il n'est pas rare qu'un réseau soit attaqué au hasard par un pirate potentiel qui cherche à profiter d'une faille sans savoir si son attaque aboutira.

Des outils de détection d'intrusion existent afin d'analyser les attaques à destination de son réseau, de manière à y réagir, à découvrir les points faibles, à localiser d'éventuelles machines compromises ou tout simplement à conserver des preuves de tentatives d'intrusions. L'outil libre le plus connu est *Snort*. Snort est un *sniffeur* réseau, c'est-à-dire un programme qui va écouter les trames passant sur un réseau et analyser leur contenu. En s'appuyant sur une base de données de motifs, il est capable de reconnaître la signature d'une attaque donnée et de l'enregistrer. Ainsi, cet outil de détection d'intrusion utilise les mêmes techniques d'écoute que les pirates, mais pour la bonne cause.

Les outils de détection d'intrusion se basent toujours sur une base de signatures qu'il convient de mettre à jour. Faute de quoi, ils deviennent inefficaces car ils ne détectent pas les attaques les plus récentes, qui sont les plus susceptibles de causer la compromission des serveurs.

Figure 3. Extrait d'un rapport produit par Snort

```
Percentage and number of attacks from a host to a
destination
=============================================================
        #  of
  %    attacks   from              to
=============================================================
14.66    4479     123.57.140.121     123.57.160.225
 0.44    134      220.69.81.193     123.57.172.326
 0.40    122      220.69.81.129     123.57.172.325
 0.40    122      220.69.81.129     123.57.172.327
```

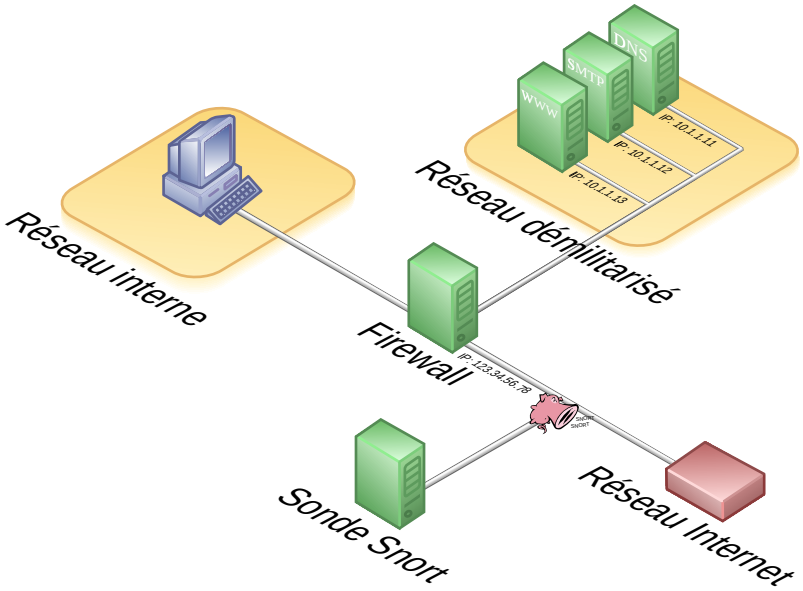
```
 0.40    121      220.69.81.129     123.57.160.225


Percentage and number of attacks from one host to any
with same method
=============================================================
        #  of
  %    attacks   from              method
=============================================================
14.66    4479    123.57.140.121    ICMP Large ICMP Packet : {ICMP}
 1.40    428     195.8.131.40      ICMP PING CyberKit 2.2 Windows : {ICMP}
 1.34    408     220.69.81.129     ICMP PING CyberKit 2.2 Windows : {ICMP}
 1.22    372     195.7.10.52       ICMP PING CyberKit 2.2 Windows : {ICMP}


Percentage and number of attacks to one certain host
=============================================================
        #  of
  %    attacks   to                method
=============================================================
31.39    9592    123.57.160.225    ICMP PING CyberKit 2.2 Windows : {ICMP}
26.27    8026    123.57.172.327    ICMP PING CyberKit 2.2 Windows : {ICMP}
14.68    4485    123.57.160.225    ICMP Large ICMP Packet : {ICMP}
 1.05    320     123.57.172.325    WEB-PHP content-disposition : {TCP}
 0.82    252     123.57.172.325    MS-SQL Worm propagation attempt : {UDP}
```

Il est possible de tirer partie des capacités de Snort pour mettre en place une sonde « discrète. » Il s'agit de placer une machine sans adresse IP sur le réseau, par exemple avant le filtre de paquets. Ainsi, elle ne pourra pas être attaquée par les méthodes classiques et ne sera pas repérée par les intrus car elle ne répondra à aucune requête, même aux *broadcasts* ARP. En revanche, Snort est capable de fonctionner même sans interface possédant une adresse IP, il écoutera tout simplement les trames et les analysera. Il est possible pour des besoins administratifs de placer une deuxième interface réseau sur la sonde snort et de la relier au filtre de paquets. Il conviendra de considérer ce mini-réseau comme un réseau de type DMZ, sans privilège particulier.

Figure 4. Déploiement d'une sonde snort en mode discret



2 Le contrôle applicatif

Les filtres de paquets et les outils de détection d'intrusions de type *Snort* sont capables de détecter les attaques les plus courantes basées sur le transport (TCP/IP par exemple) ou sur le protocole, mais il est également nécessaire d'effectuer un contrôle applicatif. Le contrôle applicatif nécessite de contrôler les données manipulées par les applications des utilisateurs et de garantir leur innocuité.

Si les postes clients sont composés de systèmes propriétaires de type MS-Windows, ils sont vulnérables aux attaques de virus. Les risques de telles attaques ont été particulièrement mis en avant récemment, pourtant ils peuvent être contrecarrés par l'instauration d'une politique de sécurité simple et le déploiement de quelques outils réseau. Notamment, un anti-virus peut être installé facilement sur une passerelle SMTP de manière à bloquer la propagation des virus « sociaux » transitant par le courrier électronique. Un contrôle similaire est également possible sur le proxy web. Une vaste gamme d'antivirus est disponible, dont plusieurs propriétaires mais aussi des produits libres comme *clamav*.

3 Le maillon faible

Un système de sécurité est comme une chaîne, sa résistance est celle du maillon le plus faible. Ainsi, la mise en place d'une politique de sécurité rigide n'a de sens que si elle

s'applique à la totalité de la chaîne de transmission de l'information, de l'utilisateur à la sortie du réseau. Il convient de déterminer le point qui lâchera en premier.

Ainsi, il est illusoire de croire qu'un système de sécurité soit sûr si les utilisateurs ne sont pas formés et ne sont pas conscients des enjeux de sécurité. Un filtre de paquets ne sert à rien si les règles définies sont trop permissives et un système de détection d'intrusions est inutile s'il n'est pas mis à jour. De la même manière, une machine ayant des privilèges sur une autre affaiblit la sécurité de cette dernière : si une machine peut accéder sans mot de passe à une autre par des clefs SSH ou via RSH, alors la compromission de la première compromet également la deuxième.

4 La sécurité par l'obscurité

Lorsqu'un système est conçu pour résister à des attaques, deux écoles s'affrontent généralement : celle de la sécurité par l'obscurité et celle de la sécurité par l'ouverture. La sécurité par l'obscurité consiste à cacher tous les détails d'une opération, notamment ses faiblesses. La sécurité par l'ouverture consiste à montrer les détails d'une opération mais aussi à se reposer sur une partie secrète, comme les mots de passe, pour garantir la sécurité globale. La sécurité par l'ouverture est généralement considérée comme meilleure dans bien des cas, notamment car la sécurité par l'obscurité donne un faux sentiment de sécurité alors que la sécurité par l'ouverture force à utiliser des algorithmes et des méthodes sûres et éprouvées.

Pourtant, dans le cas de la sécurité d'un réseau, il convient de mélanger les deux méthodes. De la même manière qu'un avion furtif réduit sa signature radar de manière à offrir moins de prise aux systèmes de DCA, un réseau doit libérer le moins d'informations possibles à un potentiel attaquant de manière à le ralentir et à l'empêcher d'utiliser immédiatement les failles de sécurité connues potentiellement présentes sur le réseau. Ainsi, les services installés doivent être configurés pour laisser transparaître le moins d'informations possible sur leur configuration ou leur version et les signatures des serveurs doivent être limitées, notamment par la mise en place de règles de filtrage de paquets limitant la communication entre les serveurs et les clients aux flux strictement nécessaires.

La sécurité par l'obscurité ne doit cependant pas être une fin en soi et n'est en aucun cas une garantie de sécurité. Ce serait comme camoufler sa porte mais ne pas la fermer à clef : un attaquant qui arriverait à la trouver pourrait entrer chez vous sans problème. Ainsi, une vraie politique de sécurité doit être menée sur l'ensemble du réseau en imaginant que le plan complet de votre réseau serait connu par un attaquant potentiel. Ainsi, les informations sensibles doivent être chiffrées lorsque c'est possible et aucune information ne doit être donnée à qui n'est pas strictement identifié.

5 Trouver un compromis utilisation et sécurité

Si les utilisateurs ont besoin de fonctionnalités et veulent un minimum de contraintes dans leur travail, le responsable de la sécurité a en revanche besoin de restreindre les accès et de limiter les fonctionnalités. Il est ainsi nécessaire de définir un compromis entre la sécurité et l'utilisabilité, qui fournisse suffisamment de fonctionnalités et en même temps qui ne compromette pas la sécurité globale. Une approche possible à la définition d'une politique de sécurité pourrait être la suivante :

- identifier les services qu'il est nécessaire de rendre disponibles aux utilisateurs et pourquoi (par exemple : messagerie, accès au web, accès en écriture à une base de données, etc.) ;
- dégager les groupes d'utilisateurs auxquels il est nécessaire de rendre disponibles des services (par exemple : employés, direction, agences, opérateurs de saisie, administrateurs, prestataires, etc.) ;
- identifier les services auxquels chaque groupe doit avoir accès (par exemple, les opérateurs de saisie doivent avoir un accès en écriture aux bases des données, les employés doivent avoir accès à leur messagerie, les administrateurs doivent avoir un accès distant aux serveurs de production, etc.) ;
- formaliser la politique de sécurité pour chaque service, identifier les risques potentiels (par exemple, les administateurs ont un accès SSH privilégié sur les serveurs de production à partir de tel poste uniquement, etc.) ;
- considérer tout autre accès à une ressource comme une violation de la politique de sécurité, être capable d'identifier de telles violations et de les remonter.

Réagir à une intrusion

Aucun système de sécurité n'est parfait. Malgré toutes les précautions qui pourraient être prises, il n'est pas impossible qu'un serveur administré soit un jour compromis. Pour dommageable qu'une telle éventualité soit, ses effets seront énormément amoindris si l'administrateur prend les mesures nécessaires.

1 Être préparé

Le plus gros travail se situe avant l'intrusion et consiste à s'y préparer. Mettre en place des outils de détection d'intrusion ou remontée d'alerte (*logcheck*, par exemple), bien sûr, mais également surveiller régulièrement l'activité des machines pour détecter tout comportement anormal (les outils de supervision comme *nagios* peuvent se révéler utiles). En effet, une intrusion est d'autant plus grave qu'elle est étalée sur le temps. Si le pirate reste discret, il n'est pas rare qu'une machine soit compromise et que l'administrateur ne s'en rende compte que plusieurs semaines après.

L'administrateur doit également être prêt à réinstaller une machine à partir de zéro car une fois compromise, il est impossible de déterminer si elle a été totalement nettoyée. Ainsi, des sauvegardes des données à restaurer doivent être faites régulièrement et l'interruption de service dûe à la réinstallation prise en compte. De même, des outils de restauration de systèmes complets, comme *mkCDrec*, permettent de réinstaller rapidement une machine fonctionnelle.

2 Conserver les fichiers de rapport

Lorsqu'une machine est déclarée compromise, une des choses les plus urgentes à faire une fois la machine coupée du réseau est de sauvegarder les fichiers de rapport s'ils sont conservés en local sur la machine. Ces informations sont cruciales à plusieurs titres : d'une part, pour déterminer l'ampleur de l'intrusion, d'autre part pour déterminer la méthode utilisée par le pirate pour s'introduire dans le réseau et l'éliminer. En outre, ces informations peuvent servir de preuve en vue d'une riposte juridique.

3 Rechercher les rootkits

Lorsqu'une machine est compromise, le pirate responsable y installe généralement un programme particulier qui leur permettra plus tard de prendre contrôle de la machine

sans avoir à exploiter à nouveau de faille de sécurité. Ce programme s'appelle *root-kit*, car il donne un accès *root*. Un tel programme est difficilement détectable car les pirates le camoufle en modifiant les commandes de type *ps*, *netstat* ou *ls*. Il est cependant possible de le détecter soit par l'utilisation d'outils de détection d'intrusion comme *tripwire*[6], soit par des outils spécifiques comme *chkrootkit*, *DDS* ou *GAG*, soit manuellement par l'utilisation d'outils de scan de ports comme *nmap* ou *nessus*.

Dans tous les cas, la recherche de rootkits est utile dans un but documentaire, mais il ne faut absolument pas considérer leur élimination comme le nettoyage complet d'une machine compromise. Une machine compromise *doit* être réinstallée à partir de zéro, faute de quoi il est toujours possible qu'un pirate y ait conservé un accès non détecté par l'administrateur.

6. Cependant, ces outils peuvent également être compromis, bien que *tripwire* comprenne des fonctionnalités de chiffrement limitées.

Conclusion

Le filtre de paquets est le pilier incontournable de l'interconnexion des réseaux. Étant en première ligne, il nécessite une attention toute particulière et un personnel qualifié. Pourtant, ce n'est qu'une des pièces du système de défense d'un réseau, il doit pas donner un sentiment illusoire de sécurité. La politique de sécurité doit être globale, l'établissement de bastions ne sert à rien si d'autres aspects du système d'information recèlent des failles.

La sécurité informatique ne se réduit pas à une simple considération technique. Elle est avant tout un problème politique et social, mettant en oeuvre non seulement l'équipe informatique, mais également la totalité du personnel de l'entreprise. Elle est une affaire d'éducation, de responsabilisation et de respect d'une charte d'utilisation des ressources, qu'elles soient critiques ou non. Les logiciels de sécurité, pour perfectionnés et sûrs qu'ils soient, ne sont que des outils qui entre de mauvaises mains peuvent se révéler inutiles ou même nocifs. Avant tout, la sécurité d'un système ne sera garantie que par l'expertise des techniciens et par la reconnaissance que cette expertise aura au sein des équipes.

Logiciels évoqués dans ce document

Cette liste décrit les logiciels évoqués tout au long de ce document.

Arpwatch (ftp ://ftp.ee.lbl.gov/arpwatch.tar.gz)

Arpwatch permet de mettre en correspondance les adresses MAC et les adresses IP véhiculant sur un segment ethernet et de notifier par courrier électronique ou de répertorier les changements intervenant. Arpwatch se révèle vite utile pour surveiller au quotidien les réseaux et il aide à la détection de l'*IP spoofing*.

ClamAV (http ://clamav.elektrapro.com/)

Clam AntiVirus (ou ClamAV) est un anti-virus pour UNIX. Il est facilement interfaçable avec les serveurs de messagerie pour analyser les attachements. Il est composé d'un service capable de traiter plusieurs fichiers en parallèle, d'outils en mode ligne de commande et d'un mécanisme de mise à jour par Internet. Il est également composé d'une bibliothèque réutilisable. La base de données de virus est mise à jour régulièrement.

DDS, GAG (http ://staff.washington.edu/dittrich/misc/ddos), chkrootkit (http ://www.chkrootkit.org/)

DDS, GAG et chkrootkit sont des outils vérifiant la présence de rootkits sur la machine où ils sont exécutés. DDS et GAG testent des rootkits particuliers, chkrootkit est capable d'en détecter une cinquantaine.

Fake (http ://www.vergenet.net/linux/fake/)

Fake permet à un serveur de secours de prendre la main sur un réseau local. Fake « vole » l'adresse IP d'une autre machine en créant une interface réseau supplémentaire et en utilisant l'*ARP spoofing.*

firehol (http ://firehol.sourceforge.net/)

Firehol prend une approche différente et définit un langage de description de règle de filtrage de paquets. Il suffit de « compiler » le fichier de description pour obtenir des règles netfilter.

fwbuilder (http ://fwbuilder.sourceforge.net/)

Fwbuilder est une interface graphique de construction de règles de filtre de paquets. Il permet de mettre en place visuellement des règles de filtrage grâce au *drag & drop*.

Heartbeat (http ://www.linux-ha.org/)

Heartbeat surveille simplement des machines mises en grappe et est capable d'exécuter des actions en cas de panne. Notamment, Heartbeat utilise le code de Fake pour prendre l'adresse IP d'une machine en panne. Bien que Heartbeat soit conçu pour les serveurs en grappe, il fonctionne parfaitement pour deux machines se surveillant mutuellement.

knetfilter (http ://expansa.sns.it :8080/knetfilter/)

Knetfilter est également une interface graphique de construction de règles pour KDE, spécifique à netfilter.

mkCDrec

MkCDrec est un outil de restauration de système complet, qui permet de cloner et de réinstaller un système GNU/Linux à partir d'un CD amorçable, créé avec une simple ligne de commande. Voir les *Secrets d'Alcôve (http ://www.alcove.com/fr/alcove/newsletter/2002/20020909)* d'août 2002 pour un article d'introduction à mkCDrec.

Mon (http ://www.kernel.org/software/mon/)

Mon est un ordonnanceur généraliste doublé d'un outil de gestion d'alertes utilisé pour surveiller la disponibilité de services et pour exécuter des alertes ou des tâches quelconques en cas de panne. Mon est conçu pour être étendu car il offre une interface commune pour y brancher des programmes écrits dans un langage arbitraire.

Nagios (http ://www.nagios.org/)

Nagios (anciennement netsaint) est un programme conçu pour surveiller des services et des machines distantes. Lorsqu'un problème arrive ou est résolu, il peut avertir l'administrateur par exemple par courrier électronique ou par SMS. La surveillance est accomplie par des « plugins, » ce qui rend facile l'écriture de procédures de surveillance dans le langage de son choix. Il est possible de consulter l'état actuel de son réseau par le web et une interface WAP permet à l'administrateur de confirmer la prise en compte de problèmes.

Nessus (http ://www.nessus.org/)

Nessus est un outil d'audit de sécurité pour les réseaux locaux. Il a pour but d'identifier les failles de sécurité potentiellement vulnérables. Contrairement à de nombreux logiciels d'audit de sécurité, Nessus ne considère rien comme acquis : il ne considérera pas qu'un service fonctionne sur un port donné (il est par exemple capable d'auditer un serveur web qui tournerait sur le port TCP 1234). Nessus ne s'arrête pas à la détection de numéros de versions de services pour connaître les failles connues, il est capable d'exploiter *réellement* les vulnérabilités afin de tester leur présence ou non.

Netfilter (http ://www.netfilter.org/)

Netfilter est un filtre de paquets intégré aux noyaux Linux de la série 2.4 ou supérieure. Il possède également des fonctionnalités avancées de filtrage de paquets et de traduction d'adresses et de ports. Voir les *Secrets d'Alcôve (http ://www.alcove.com/fr/alcove/newsletter/2001/20010215)* de février 2001 pour un article d'introduction à Netfilter.

Nmap (http ://www.insecure.org/nmap/)

Nmap (pour *Network Mapper*) est un outil d'audit et d'exploration de réseaux qui permet d'auditer rapidement des réseaux importants. Nmap scanne les ports des machines cibles et en déduit les systèmes d'exploitation utilisés (ainsi que leurs versions), quels types de filtres de paquets sont utilisés, etc.

PortSentry (http ://sourceforge.net/projects/sentrytools/)

Portsentry est un outil permettant de détecter les scans de ports (y compris les scans de ports discrets) sur un serveur. Il est capable de réagir de manière proactive et de bloquer l'adresse IP des attaquants. Ce type de comportement doit cependant n'être activé qu'avec précaution.

Snort (http ://www.snort.org/)

Snort est un système de détection d'intrusions. Il fonctionne en écoutant les trames transitant sur le réseau auquel il est relié et en mettant en correspondance leur contenu avec une base de signatures d'attaques connues. Voir les *Secrets d'Alcôve (http ://www.alcove.com/fr/alcove/newsletter/2002/20020128)* de janvier 2002 pour un article d'introduction à Snort.

Tripwire (http ://www.tripwire.org/)

Tripwire est un outil de détection d'intrusions par analyse de systèmes de fichiers. Tripwire construit une base de données de l'état du système de fichiers (« photographie ») et compare à la demande cette base de données avec l'état actuel des fichiers surveillés. Afin de garantir une sécurité optimale et afin d'éviter la corruption de la base de données par un pirate, Tripwire utilise des techniques limitées de cryptographie à clef publique pour protéger ses fichiers.