Retour d'expérience sur des missions de *forensics* OSSIR

Samuel Dralet (s.dralet@lexfo.fr) - LEXFO

Lundi 8 Octobre 2007

- société créée en Avril 2007 ;
- site: http://www.lexfo.fr ;
- blog (pour être à la mode): http://www.lexfo.fr/blog/ ;
- objectif de devenir une grande multi-nationale ... la route sera longue :)

Expertise en sécurité informatique :

- tests d'intrusion et audits sécurité ;

  L'article "Test d'intrusion grandeur nature" dans le hors-série Pentest de MISC (Octobre 2007) décrit un test d'intrusion effectué chez un client. Il démontre clairement que des outils automatiques ne suffisent pas à réaliser un pentest.

- audits d'application et de code source.

- recherche et développement ;

  Développement en cours pour Orange Labs de Meterpretux, un meterpreter Posix pour Metasploit : http://meterpretux.s34l.org

- réponse aux incidents de sécurité ;

- formations très spécialisées :
  - exploitation de vulnérabilité sous Windows ;
  - forensics sous Windows ;
  - sécurité Bluetooth et WiFi ;
  - rootkit et backdoor en environnement Windows ;
  - reverse engineering sous Windows.

Action d'acquérir, de recouvrer, de préserver et de présenter à la fois des informations traitées par le système d'information ainsi que des données stockées sur des supports informatiques.

Plus clairement :

- récupération de preuves informatiques sur supports informatiques (disques durs, disques amovibles, PDA, clés USB, cartes SIM, etc) ;
- récupération de données effacées sur supports informatiques ;
- détection de compromission sur des systèmes d'information.

D'après les différentes missions effectuées, on peut classifier les analyses forensics en deux catégories :

1. l'analyse est spécifique au support informatique et/ou au système d'information et à la problématique du client ;
2. l'analyse est "générique" puisqu'une seule problématique existe : la détection de compromission sur un système d'information.

De cette classification dépendent les outils à utiliser, les procédures à mettre en oeuvre et les problèmes rencontrés lors des analyses forensics.

Quatre cas concrets d'analyses forensics pour expliquer plus clairement cette classification (c'est bon, le dernier métro est à 1h :)

- analyse spécifique ;
  $\longrightarrow$ récupération d'*event logs* effacés
  $\longrightarrow$ récupération de preuves informatiques via la technique dite de "file carving"
  $\longrightarrow$ récupération de SMS effacés
- analyse générique ;
  $\longrightarrow$ détection de compromission à l'aide de l'outil **Forensics Live Tool** de LEXFO

Le contexte

Une tierce personne a volontairement détruit les *events logs* d'un serveur Windows 2000 en effaçant chaque enregistrement à l'aide de l'observateur d'évènements.

$\Rightarrow$ Les fichiers sont toujours présents, seuls leurs contenus ont été effacés.

Objectif: récupérer les *events logs* et prouver que cette personne s'est connectée un Dimanche entre 14h et 17h.

Processus de l'analyse

1. Copie bit à bit de la partition principale C: du serveur vers un disque externe branché via le port USB sur le serveur. L'intégrité des données récupérées est vérifiée automatiquement lors de la copie grâce à une signature MD5.

2. Récupération des *events logs* à partir de l'image obtenue précédemment.

La copie bit à bit

Permet une copie intégrale de l'information présente sur un support d'information (disque dur ou disque amovible), y compris les espaces non utilisés et les espaces non alloués.

La commande utilisée dans notre cas de figure:

```
dd.exe if=\\.\PhysicalDrive0 of=d:\images\PhysicalDrive0.img
--md5sum --verifymd5 --md5out=d:\images\PhysicalDrive0.img.md5
```

- *dd.exe* est une version modifiée de Windows de l'outil Unix dd disponible à l'adresse **http://users.erols.com/gmgarner/forensics/** ;
- `if=\\.\PhysicalDrive0` désigne la source de données à copier ;
- `of=d:\images\PhysicalDrive0.img` désigne l'emplacement vers lequel les données sont copiées ;
- les autres options permettent de vérifier l'intégrité des données copiées.

Le format des *event logs* 1/3

Un évènement affiché dans l'application *Event Viewer* regroupe en fait plusieurs sources de données:

- l'*event log record* enregistré dans un fichier `.evt` présent dans le répertoire `C:\windows\system32\config`.

  → `AppEvent.evt` pour le journal des applications ;
  → `SecEvent.evt` pour la sécurité ;
  → `SysEvent.evt` pour le système.

Le format des *event logs* 2/3

L'*event log record* a le format suivant:

typedef struct _EVENTLOGRECORD {
DWORD Length; // taille exacte de l'*event record*
DWORD Reserved; // valeur `ELF_LOG_SIGNATURE (0x654C664c)`, soit en ASCII "eLfl"
DWORD RecordNumber;
DWORD EventID;
DWORD TimeGenerated;
DWORD TimeWritten;
DWORD EventID;
[...]
} EVENTLOGRECORD

Détail de la structure `_EVENTLOGRECORD` à l'adresse

http://msdn2.microsoft.com/en-us/library/aa363646.aspx

Le format des *event logs* 3/3

- les clés de registre
  `HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\Eventlog\`

  Elles fournissent comme informations les fichiers (des fichiers .dll) contenant les messages associés aux *event records* .

- les *message files*

  Le message affiché va dépendre de l'*event id*. Certains messages contiennent des variables. Les valeurs sont présentes à la suite de la structure `EVENTLOGRECORD` et définies à l'adresse http://msdn2.microsoft.com/en-us/library/aa363646.aspx

L'*event viewer* fait la correspondance entre l'*event record*, l'*event id* et les messages pour afficher ce que vous avez l'habitude de voir dans l'observateur d'évènements de votre système.

Récupération des *event logs* effacés 1/3

La procédure consiste à rechercher à partir de l'image disque précédemment obtenue toutes les occurrences de la chaîne de caractères **"LfLe"**.

La taille de l'event log est présente à (offset de "LfLe" - 4)

$\Rightarrow$ il est facile de reconstruire l'évènement à chaque occurence de la chaîne "LfLe" trouvée.

Récupération des *event logs* effacés 2/3

```
$ hexdump -s 0x00000834 -n 680 -C disk.img
00000834 a8 02 00 00 4c 66 4c 65 0a 00 00 00 91 71 c0 45 ....LfLe.....qÀE
00000844 91 71 c0 45 e8 03 00 00 04 00 01 00 00 00 00 00 .qÀEè...........
00000854 00 00 00 00 7e 00 00 00 00 00 00 00 7e 00 00 00 ....~.......~...
00000864 00 00 00 00 a0 02 00 00 56 00 4d 00 77 00 61 00 .... ...V.M.w.a.
00000874 72 00 65 00 20 00 4e 00 41 00 54 00 20 00 53 00 r.e. .N.A.T. .S.
00000884 65 00 72 00 76 00 69 00 63 00 65 00 00 00 43 00 e.r.v.i.c.e...C.
[...]
```

A partir de cette information et connaissant le format d'un *event record*, il est possible de reconstruire l'*event log*:

- offset 00000834: a8 02 00 00 représente la taille de l'évènement soit 680
- offset 00000838: 4c 66 4c 65, notre chaîne de caractères "LfLe"
- offset 0000083c: 0a 00 00 00, le *record number* soit 10
- offset 00000840: 91 71 c0 45, le *time generated* soit Wed Jan 31 11:38:09 2007
- offset 00000844: 91 71 c0 45, le *time written identique* au champ précédent
- offset 00000848: e8 03 00 00, *l'event id* soit 1000
- etc

Récupération des *event logs* effacés 3/3

L'*event id* est égal à 1000.

Il suffit alors d'interroger le site http://www.eventid.net pour connaitre la signification exacte de l'évènement par rapport à son *event id*.

Evènement obtenu:
N d'évènement: 10
Date de création : Wed Jan 31 11:38:09 2007
Source: VMware NAT Service
Ordinateur: COMPAQ-T20VHPQV
Type: Information
Description Using configuration file: C:\Documents and Settings\All Users\Application Data\VMware\vwnetnat.conf
IP address: 192.168.85.2
Subnet: 255.255.255.0
External IP address: 0.0.0.0
Device: vmnet8
MAC address: 00:50:56:F1:8B:DD

Ignoring host MAC address: 00:50:56:C0:00:08

Conclusion

Cette procédure a permis de récupérer les *event logs* et de remplir l'objectif défini plus tôt.

Mais:

- le serveur était en production, donc aucun droit à l'erreur (j'ai perdu 10kg ... je sais ca se voit pas :) ;
- le disque dur n'était pas endommagé ;
- la capacité du disque ne faisait **que** 60 Go ;
- un port USB était disponible sur le serveur ;
- la connaissance du format d'un *event log* était nécessaire (le format des *event log* vient de changer sur Windows Vista) ;
- aucun outil de récupération de données ne répondaient à nos besoins, l'analyse fût donc longue et fastidieuse.

Résumé

→ Analyse qui consiste à faire une recherche sur l'image disque par rapport au type de fichiers (http://www.forensicswiki.org/wiki/Carving).

→ Les outils disponibles :

- scalpel (http://www.digitalforensicssolutions.com/Scalpel/) ;
- photorec (http://www.cgsecurity.org/wiki/PhotoRec_FR).

Le contexte

→ Un employé est soupconné d'envoyer des lettres injurieuses en interne.
→ Les lettres contiennent des mots spécifiques et une photo modifiée qui vont servir à l'analyse.
→ Cet incident est vieux de **4 mois**.

Objectif: Apporter la preuve en analysant son poste de travail que l'employé est bien l'auteur de ces lettres.

Analyse forensics: recherche de mots clés (1/2)

- copie bit à bit des partitions C: et D: du poste de travail concerné vers un disque externe branché via le port USB avec vérification de l'intégrité des données récupérées ;
- recherche des mots clés injurieux dans les images disques :
  $\longrightarrow$ récupération de toutes les chaînes de caractères
  $\longrightarrow$ recherche des mots dans la liste des chaînes de caractères
  $\longrightarrow$ aucun résultat et beaucoup de **faux positifs**

Analyse forensics: recherche de la photo (2/2)

→ utilisation de la technique dite de "file carving" à l'aide de l'outil *scalpel* ;
→ l'original de la photo a été récupérée:

"la copie de la partition D a une taille de 18 073 124 Ko, la photo jpeg nommée 00002046.jpg (sur un total de 5226 images jpeg) est présente à l'octet 1172427140 et a une taille égale à 8054 octets."

Conclusion

L'objectif défini plus tôt a été atteint (l'employé a été suspendu !).

Mais:

- la capacité du disque ne faisait **que** 40 Go ;
- un port USB était disponible sur le serveur ;
- la recherche des mots clés fût longue et fastidieuse: beaucoup de faux positifs ;
- le client n'a pas réagi assez tôt.

Le contexte

→ Une personne a récupéré le portable de son mari avec tous les SMS effacés.

→ L'affaire est délicate puisque cette personne souhaite prouver grâce à ces SMS que son mari a été poussé à se suicider !!

Objectif: Récupérer des SMS effacés sur une carte SIM.

Définition et caractéristiques 1/2

La carte SIM (**Subscriber Identity Module**) est une SmartCard.

- elle authentifie les connexions vers les réseaux GSM ;
- elle contient en général de 16 à 64 Ko de mémoire non volatile, un processeur et un système d'exploitation ;
- son contenu est d'ordinaire protégé par un code PIN (Personal Identification Number) et un code PUK (Personal Unblocking Code) ;
- elle contient d'autres informations comme les paramétres de l'utilisateur (langue, réseau préféré), son IMSI (International Mobile Subscriber Identity) et sa clé secrète Ki.

Définition et caractéristiques 2/2

Ces informations sont organisées sous forme de système de fichiers (seul le répertoire TELECOM est représenté):

7F10 TELECOM

- 6F3A Directory
- 6F3B Fixed directory
- 6F3C SMS
- 6F40 Last calls
- 6F42 SMS pointer
- 6F43 SMS status
- 6F44 Dialing numbers
- 6F4A Extension 1
- 6F4B Extension 2

Comment lire les données sur une carte SIM ?

→ un lecteur de carte à puce

- équipé de préférence d'un connecteur *SmartMouse* ;
- de type *Phoenix* ;
- une connection série est préférée à une connection USB.

→ un outil pour envoyer des commandes et recevoir les résultats

- *Chipit* par exemple disponible à l'adresse http://home.scarlet.be/chipit/Chipit.html

Format du SMS

Le SMS a une longueur de 176 octets dont le format est le suivant :

- le premier octet correspond au statut du message ;
- le reste correspond aux données (appele aussi TPDU pour Transport Protocol Data Unit) qui contient entre autre le numéro émetteur du message, la date et le texte.

Le statut d'un SMS peut prendre les valeurs suivantes :

- 0x00: SMS inutilisé ;
- 0x01: SMS lu ;
- 0x03: SMS non lu ;
- 0x05: SMS envoyé ;
- 0x07: SMS non envoyé.

Un SMS est effacé

→ **Selon le terminal GSM, seul le statut du SMS est modifié.**

Le contenu est toujours présent ... jusqu'à ce qu'un nouveau SMS vienne écraser celui marqué comme effacé ! [1]

[1] certains terminaux GSM peuvent effacer les 176 octets ou les remplacer par des FF plutôt que simplement changer le statut du SMS.

Récupérer un SMS effacé 1/2

Il est possible de lire tous les SMS sur une carte SIM quelque soit leurs statuts.
⇒ les SMS effacés compris

Il suffit de lire les *records* (enregistrements) à l'adresse de base `7F10:6F3C`.

```
00079133860940000F0040B913336476867F10000703032302230401 2C16030180C0
683C16030180C0683C120FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
```

- le premier octet ici 00 signifie que le SMS est inutilisé ;
- le TPDU (le reste des octets) est en fait encodé en ASCII 7 bits avec un en-tête

  ⇒ un décodeur est donc nécessaire (par exemple *decodesms* de LEXFO à l'adresse http://www.lexfo.fr/stuff/decodesms.tgz)

Récupérer un SMS effacé 2/2

PDU LENGTH IS 175 BYTES
ADDRESS OF DELIVERING SMSC
NUMBER IS : +33689004000
TYPE OF NR. : International
NPI : ISDN/Telephone (E.164/163)
MESSAGE HEADER FLAGS
MESSAGE TYPE : SMS DELIVER
MSGS WAITING IN SC : NO
SEND STATUS REPORT : NO
USER DATA HEADER : NO UDH
REPLY PATH : NO
ORIGINATING ADDRESS
NUMBER IS : +33612345678
TYPE OF NR. : International
NPI : ISDN/Telephone (E.164/163)
PROTOCOL IDENTIFIER (0x00)
MESSAGE ENTITIES : SME-to-SME
PROTOCOL USED : Implicit / SC-specific
DATA CODING SCHEME (0x00)
AUTO-DELETION : OFF
COMPRESSION : OFF
MESSAGE CLASS : NONE
ALPHABET USED : 7bit default
SMSC TIMESTAMP : 23/03/07 03:22 GMT+1,00
USER DATA PART OF SM
USER DATA LENGTH : 18 septets (should be 169)

USER DATA (TEXT) : AAAAAAAAAAAAAAAAAA

Analyse de la carte SIM

- récupération de la carte SIM et du code PIN ;
- utilisation du lecteur et de l'outil Chipit pour récupérer d'éventuels SMS effacés mais impossible de lire la carte SIM ;

  ⇒ Chipit n'envoit pas les bonnes commandes à la carte SIM.
  ⇒ Un autre outil plus "professionnel" (Smart Access de Atmel) est nécessaire pour envoyer ses propres commandes à la carte SIM.
- lecture de tous les enregistrements de la carte SIM.

  ⇒ récupèration uniquement des SMS visibles avec le terminal GSM.
  ⇒ les autres enregistrements contiennent tous des FF.

Conclusion

- la capacité à pouvoir récupèrer des SMS effacés sur une carte SIM dépend fortement du modèle du terminal GSM ;
- l'outil utilisé au départ ne répondait pas du tout aux besoins ;
- il aurait fallu pouvoir analyser aussi la mémoire flash du terminal GSM mais c'est une autre histoire...

La difficulté des analyses forensics dépend uniquement :

- du type de support informatique à analyser ;
- de la taille des données à récupérer ;
- de la méthode pour les récupérer ;
- des outils d'analyses disponibles.

Un client nous appelle complètement alarmé: "J'ai des fichiers bizarres sur un de mes serveurs !"

⇒ panique général, les femmes et les enfants d'abord, on récupère tant bien que mal des outils sur Internet, on analyse comme on peut le système.

Conclusion: nous ne sommes pas préparés !

Objectif: quelque soit le système informatique, détecter autant que possible toute forme d'intrusion.

Cahier des charges

- il est multi-architectures (Windows, Linux, Solaris);
- il est modulaire et donc facilement évolutif ;
- il répond à la règle élémentaire d'une analyse forensics: aucune modification du système ;
- il analyse le système en cours d'exécution donc aucun arrêt de services ;
  $\Rightarrow$ **live forensics**
- il peut comparer 2 analyses à un instant t et t+1 pour vérifier périodiquement que le système est vierge de toute intrusion ;
- l'analyse peut être automatisée et programmée ;
- il n'est pas destiné aux expertises judiciaires.

$\rightarrow$ LEXFO propose uniquement de la prestation autour de ce produit ;

$\rightarrow$ http://www.lexfo.fr/doc/plaquette_lexfo.pdf

Scénarios d'utilisation

1. scénario catastrophe : le client est certain qu'il y a eu compromission d'une machine ;

   $\Rightarrow$ Intervention d'urgence.

2. scénario préventif : le client souhaite s'assurer du "bon" état de ses machines ;

   $\Rightarrow$ Intervention programmée.

Architecture logicielle

Découpée en 2 parties :

- **le noyau** : il est développé avec le langage *Python* utilisable sur tous les systèmes supportant ce langage.

  Avantage : ne dépend d'aucun système d'exploitation.

- **les modules** :

  → propres à chaque système ;
  → utilisent l'API de l'outil **Forensics Live Tool** ;
  → utilisent des outils externes.

  Avantage : grande souplesse pour ajouter ou améliorer les techniques d'analyse.

Les modules

Ils sont regroupés par fonctionnalités ou "stages":

→ stg_fs: analyse du système de fichiers (NTFS, Ext2/3, etc) ;
→ stg_shell: récupération d'informations à l'aide de commandes natives au système ;
→ stg_process: analyse des processus en cours d'exécution (détection de processus cachés, code injecté, etc);
→ stg_bin: analyse des fichiers du système au niveau du format de binaire du système (PE, ELF);
→ stg_kernel: analyse du système au niveau de l'espace noyau.

⇒ L'ensemble des éléments du système sont analysés.

Template d'un module

```
from XXX import *

class Module(XXX):
def __init__(self, args):
self.args = args
self.log = ""

def run(self):
[YOUR CODE]
return self.log
```

Evolution de l'outil Forensics Live Tool

A la demande d'un client, l'outil Forensics Live Tool va devenir une appliance :

- mode client-serveur ;
- possibilité de configurer l'analyse forensics d'un système à partir du serveur ;
- programmation des analyses depuis le serveur ;
- centralisation sur le serveur des résultats des analyses ;

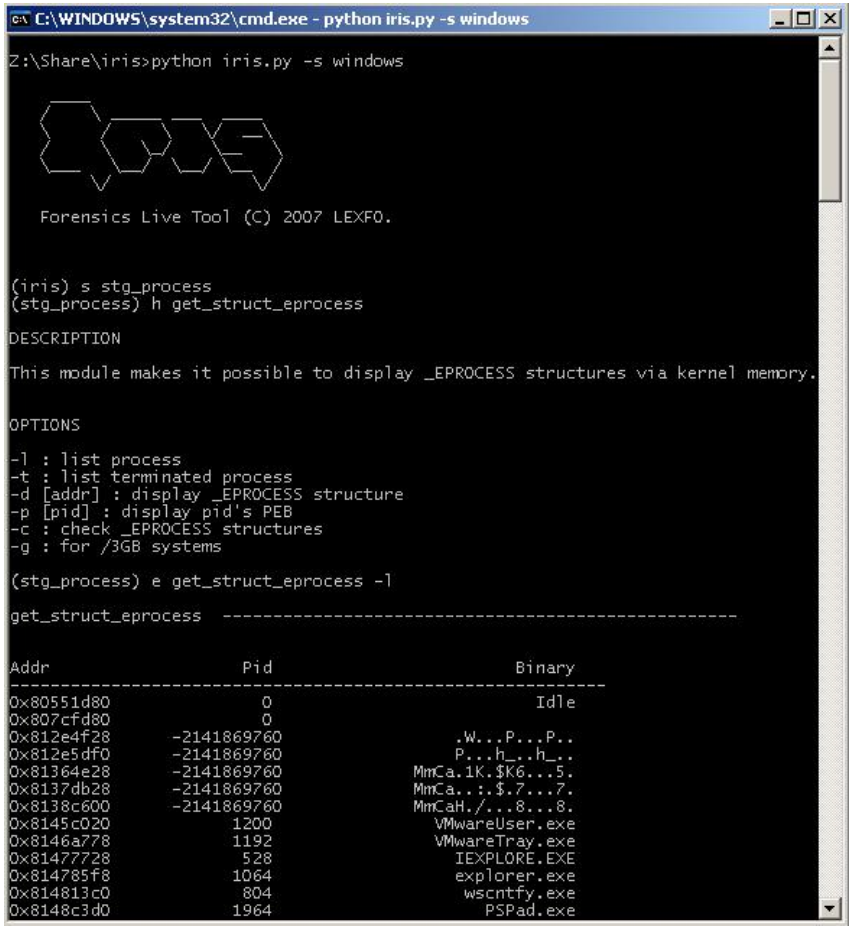
Exemple d'un module sous Windows (1/3)

Récupération des structures _EPROCESS sur un système en cours d'exécution (*live forensics*) :

- lit la mémoire à l'aide de l'API ZwSystemDebugControl() ;
- parse de la même manière que PTFinder [2] pour retrouver les structures _EPROCESS

[2] http://computer.forensikblog.de/files/ptfinder/ptfinder-current.zip

Exemple d'un module sous Windows (2/3)



Exemple d'un module sous Windows (3/3)

Difficulté à lire la mémoire d'un système Windows [3] :

→ L'API ZwSystemDebugControl() n'est plus disponible sur les serveurs Windows 2003 SP1 !!

→ Besoin de développer une nouvelle technique [4]

[3] http://actes.sstic.org/SSTIC07/Forensics_Memoire_Windows/

[4] http://recon.cx/en/f/aionescu-subverting-w2k3-kernel-integrity-protection.ppt

En résumé

Pour cet exemple, avant même le travail de détection de compromission, nous rencontrons des problèmes :

- l'évolution sans cesse des systèmes d'exploitation sans compatibilité descendante ;
- la difficulté à lire la mémoire (la présentation de Nicolas Ruff au SSTIC cette année le démontre bien :).

Merci. Des questions ?

s.dralet@lexfo.fr