



MISC

Multi-System & Internet Security Cookbook

100 % SÉCURITÉ INFORMATIQUE

N° 76 NOV./DÉC. 2014

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



RÉSEAU DÉTECTION / BOTNET

Fast Flux :
comprendre son
fonctionnement et
bloquer les sites
malveillants p. 52



SYSTÈME SAN / STOCKAGE

**Convergence
des réseaux de
stockage sur IP
et Ethernet : quel
impact pour la
sécurité ?** p. 66



RÉSEAU SCAN / NETFILTER

**Analyse des brute
force SSH : qui
sont les attaquants,
quels sont leurs
réseaux ?** p. 60



APPLICATION TRACKING / CNIL



**Vie privée : fuite de données
personnelles dans les
applications web** p. 74

DOSSIER

SÉCURITÉ DU CLOUD : PEUT-ON CONFIER SON INFRASTRUCTURE À UN TIERS ? p. 28

- 1 - Les différents types de nuages
- 2 - Mutualisation et élasticité : un risque pour la disponibilité ?
- 3 - Le TPM au secours du Cloud
- 4 - Sortir du brouillard juridique



PENTEST CORNER

**Contribuer à
Metasploit en
développant vos
propres exploits** p. 04



MALWARE CORNER

**Prise de contrôle
à distance avec
le malware
Blame** p. 11



FORENSIC CORNER

**Le guide pas à pas
pour analyser une
image mémoire
linux avec Volatility,
seconde partie** p. 14



NOUVEAUX SERVEURS BUSINESS LINE

La confiance est un critère important pour le choix d'un serveur. Avec 13 années d'expertise serveur, plus de 6 000 collaborateurs dans 11 pays, 70 000 serveurs hébergés et 7 data centers haute performance qui garantissent la sécurité de vos données, nous sommes votre partenaire de confiance !



NOUVEAU !

Serveurs dédiés Business Line X8i et X10i avec Hardware Dell™ PowerEdge™ R630

- Nouveaux processeurs Intel® Xeon® E5-2600 V3 jusqu'à 10 cœurs HT, 2,3 GHz et 128 Go de RAM DDR4
- Jusqu'à 6 To HDD, Hardware RAID 6 et disque dur optionnel Intel® SSD
- Raccordement 1 Gbit/s avec trafic illimité
- Sécurité maximale grâce à des composants redondants

- Connexion facile avec votre infrastructure de serveur Dell™ existante via Dell™ OpenManage Essentials

Retrouvez notre gamme complète sur 1and1.fr : des serveurs pour démarrer à partir de 19,99 € HT/mois* aux serveurs professionnels pour les plus hautes exigences.

BUSINESS LINE by Dell™

À partir de

199,99 € HT/mois
(239,99 € TTC)*



☎ 0970 808 911
(appel non surtaxé)



1and1.fr

* Le serveur dédié X8i est à partir de 199,99 € HT/mois (239,99 € TTC) pour un engagement minimum de 6 mois. Le serveur dédié L2 est à partir de 19,99 € HT/mois (23,99 € TTC) pour un engagement minimum de 6 mois. À l'issue des 6 premiers mois, les prix habituels s'appliquent. Frais de mise en service : 99 € HT (118,80 € TTC) pour le serveur X8i, 29 € HT (34,80 € TTC) pour le serveur L2. Offres également disponibles avec une durée d'engagement minimum de 12 mois ou sans durée d'engagement minimum. Conditions détaillées sur 1and1.fr. Dell, le logo de Dell, l'emblème de Dell et PowerEdge sont des marques protégées appartenant à Dell.

Commençons par une banalité : l'affaire Snowden, qui nous envoie de *bons baisers de Russie*, a été un électrochoc. Ceci dit, quelques mois auparavant, l'intrusion de Bercy aussi. Tout comme celles de l'Élysée, de Belgacom, d'Areva, bref, la routine est faite d'*Opérations Tonnerre*. Elles ont toutes en commun d'être récentes. Difficile de trouver trace d'une grosse intrusion en France avant 2010.

Depuis, la gestion de crise se professionnalise, s'organise comme LE truc à la mode où tout le monde est expert (que tous les CERT en France lèvent le doigt), mais où concrètement, seule l'ANSSI chez nous, et 2-3 boîtes non françaises ont géré des crises de bout en bout. Si on s'y prend mal, on tape sur un malware, mais il revient et *meurt un autre jour...* quand il meurt.

Et justement, pas mal de choses bougent sous l'impulsion de l'ANSSI, bien que dans une position ambiguë :

- réglementation... qui conduit globalement les gens (même en interne) à se plaindre de la lourdeur administrative bien de chez nous. Vive la centralisation et cette habitude d'empiler des strates de hiérarchies plus ou moins compétentes pour assouplir le processus #ironie. S'installe alors le rythme administratif : *vivre et laisser mourir*.
- les « clients » de l'ANSSI : l'État et les OIV (Organisme d'Importance Vitale). Ça couvre du monde, mais hors de ça, que se passe-t-il : *owner n'est pas jouer* ? Sans oublier les nouvelles prérogatives liées à la LPM comme la contre-attaque.
- les actions de l'ANSSI vers l'extérieur : les avis du CERT FR (merci pour la planète, on ne risque pas la déforestation si on les imprime), des guides #nocomment et de temps en temps, un outil (oups, 6 sur le GitHub) ou CLIP, un Linux durci... mais qui n'est pas disponible à tous. N'oublions pas non plus ce site web so De Gaulle. Heureusement, il y a de très bonnes conférences #sincère (cf. ANSSITC il y a 2 ans), je rage de voir tant de compétences et si peu de résultats (ce qui ne veut pas dire qu'on y gèle).

Pour résumer, l'ANSSI cherche à contrôler tout ce qui se passe en sécurité en France, mais les retours vers la communauté de la sécurité ne sont pas à la hauteur des espoirs engendrés par l'Agence #Frustration.

En sécurité, on mesure couramment la légitimité de quelqu'un à ce qu'il produit. Or là, on ne voit quasiment rien de ce que crée l'ANSSI, car il n'y a pratiquement rien de public (*for your eyes only*). Du coup, après avoir recruté des juniors à tour de bras, l'ANSSI impose sa légitimité par la Loi. Cette légitimité législative donne une petite arrogance à certains (très peu pour être honnête) alors qu'ils ont encore du lait qui leur coule au coin des lèvres en termes de sécurité.

Il est quasi-sûr que, sans ce passage en force, les grandes entreprises et OIV ne bougeraient pas autant qu'il faudrait (la sécurité n'est que rarement la priorité, ça coûte de la money avec un budget d'un penny). Quand le sage montre le doigt, le fou regarde la Lune et raque (*moon raker* en anglais, évidemment). Alors, rien que pour ça, merci l'ANSSI.

Cette frénésie réglementaire est aussi liée à une obligation de neutralité : l'ANSSI, au service de l'État, ne peut (veut) pas dire que tel produit est pourri, ou telle société incompetente, elle n'a pas de *permis de tuer* en tant que juge. Pour compenser, la réglementation a vocation à éliminer les « mauvais », mais regardez les 8 CSPN et 13 PASSI, pas de surprise ! Du coup, à quoi servent ces usines à gaz ? Les clients continuent à jouer à la roulette pour choisir produits et prestataires comme au *casino royale*. Tous les CSPN/CESTI et PASSI ne sont pourtant pas équivalents. Pourquoi ne pas mettre une sorte de graduation par domaines d'expertise par exemple ?

L'Agence pourrait aussi révolutionner ses règles de diffusion (articles et outils) guidées par le politiquement correct. Les bulletins de sécurité fourniraient différents niveaux d'analyses des failles et une signature pour détecter les attaques utilisant ce vecteur. Offrez du vrai contenu ! Aujourd'hui, on récupère un bulletin, un rapport, et *on ne lit que deux fois* la prose de l'Agence.

La sécurité est un domaine avant tout humain. Pas technique, organisationnel, juridique ou je ne sais quoi : humain ! Et la sécurité marche parce que des personnes collaborent, discutent, se font confiance. Et ça, pas une réglementation ou un contrat ne pourra le forcer. Tout le monde se connaît. Les labos de l'ANSSI font des trucs impressionnants : ouvrez-les à la coopération, simple, sans cadre administratif, juste parce que des échanges naît le progrès.

La France est un petit pays avec de très nombreux talents, entre autres avec ses ingénieurs. Il serait peut-être temps de les libérer, de les laisser briller : *les diamants sont éternels*. Faites-leur confiance : *la France ne suffit pas*. Mais pour ça, il faudrait aussi arrêter de sans cesse se reposer sur l'État, et prendre ses responsabilités : c'est aux acteurs d'agir, pas d'attendre que ça vienne à eux.

N'en doutez pas, je suis fan de l'ANSSI. Ou plus exactement d'un paquet de monde qui y travaille et qui essaye de faire bouger la bête administrative et hiérarchique de l'intérieur. Mais je suis vraiment frustré de voir la sous-exploitation, et plus encore des jeunes déprimer devant la lourdeur interne et me sortir comme excuse « on n'y peut rien, c'est l'administration ». Des choses bougent, certes, mais tellement lentement si on suit le système.

Conclusion : pour faire un *Bond (James Bond)* en avant, bougez-vous le Q ! Ou pour être plus souverain, on ne manque pas d'*os, sans disette*, dans ce nid d'APT.

God save l'ANSSI

Fred RAYNAL (idéliste)
@fredraynal
@MISCRedac

PENTEST CORNER

[04-08] Contribuer à Metasploit : guide du débutant

MALWARE CORNER

[11-13] Analyse du malware Blame

FORENSIC CORNER

[14-26] Volatilisons Linux : partie 2

DOSSIER



SÉCURITÉ DU CLOUD :
PEUT-ON CONFIER SON
INFRASTRUCTURE À UN TIERS ?

[28] Préambule

[29-34] Appréhender le cloud, pour le meilleur et pour le pire

[36-40] Introduction à la sécurité de la gestion dynamique des ressources dans le cloud

[42-45] Mécanismes de sécurité au niveau processeur pour la virtualisation

[46-50] Le Cloud : cette nébuleuse juridique

RÉSEAU

[52-59] Génération automatique de règles Snort pour la détection des réseaux Fast-Flux

[60-65] Une étude des bruteforce SSH

SYSTÈME

[66-73] La sécurité des SAN (2ème partie) : impacts de la convergence LAN/SAN

APPLICATION

[74-82] Analyse d'une inscription en ligne : comment vos données fuient sur Internet...

ABONNEMENT

[09,10] Bon d'abonnement & Commandes

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
boutique.ed-diamond.com
IMPRIMÉ en Allemagne - PRINTED in Germany
Dépôt légal : A parution
N° ISSN : 1631-9036
Commission Paritaire : K 81190
Périodicité : Bimestrielle
Prix de vente : 8,90 Euros

LES ÉDITIONS
DIAMOND

Directeur de publication : Arnaud Metzler
Chef des rédactions : Denis Bodor
Rédacteur en chef : Cédric Foll
Secrétaire de rédaction : Aline Hof
Conception graphique : Kathrin Scali
Responsable publicité :
Black Mouse Communication
Tél. : 03 67 10 00 27
Service abonnement : Tél. : 03 67 10 00 20
Illustrations : www.fotolia.com
Impression : pva, Druck und Medien-Dienstleistungen GmbH, Landau, Allemagne
Distribution France : (uniquement pour les dépositaires de presse)
MLP Réassort :
Plate-forme de Saint-Barthélemy-d'Anjou. Tél. : 02 41 27 53 12
Plate-forme de Saint-Quentin-Fallavier. Tél. : 04 74 82 63 04
La rédaction n'est pas responsable des textes, illustrations et photos qui lui sont communiqués par 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.

boutique.ed-diamond.com

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



CONTRIBUER À METASPLOIT : GUIDE DU DÉBUTANT

Davy Douhine (@ddouhine) – Netcore

mots-clés : **METASPLOIT / CONTRIBUER / MODULES / EXPLOIT / EMC REPLICATION MANAGER / CVE-2011-0647 / ZDI-11-061**

Inutile de présenter le framework d'exploitation Metasploit, conçu par HD Moore et désormais maintenu par la société Rapid7. Devenu une trousse à outils incontournable pour les tests d'intrusion en quelques années, il est très largement utilisé par la communauté de la sécurité informatique.

C'est d'ailleurs probablement cette communauté qui est à l'origine de ce succès, car elle contribue grandement au développement de l'outil.

Mais comment peut-on contribuer au projet ? Est-ce à la portée de tout le monde ? En prenant un exemple concret de soumission d'exploit, nous allons essayer de répondre à ces questions.

1 Pas d'exploit pas de chocolat

Pour exploiter une vulnérabilité, il faut le plus souvent un *exploit*, aussi simple soit-il. C'est ce petit bout de code qui va déclencher une vulnérabilité pour ensuite exécuter sur la cible une charge utile qui réalisera les actions menant à l'objectif de l'attaquant. Les pentesteurs et autres curieux plus ou moins bienveillants utilisent ou écrivent régulièrement des *exploits*.

Mais écrire un exploit qui « marchotte » en Perl ou en Python et l'utiliser pour une mission est une chose. L'affûter et le rendre public en est une autre. Cette question : publier ou non, je me la suis posée il y a quelque temps lors d'un test d'intrusion et je suppose se la posent.

La petite histoire commence par un échec. En l'occurrence un test d'intrusion raté. Il faut dire que tous les ingrédients sont réunis : un sacré périmètre, peu de temps, une seule

vulnérabilité intéressante, mais aucun exploit connu. Alors surgit un dilemme : doit-on bâcler le test d'intrusion en utilisant la formule consacrée dans le rapport « un attaquant pourrait développer un exploit pour cette vulnérabilité » ou écrire cet exploit ? Dans un élan de courage et d'optimisme, j'ai préféré ne pas employer le conditionnel et j'ai donc opté pour la *seconde voie*.

Mais calmons d'emblée les ardeurs des aficionados d'OllyDbg, vous n'aurez pas le moindre opcode dans cet article. Nous allons présenter une vulnérabilité qui ne nécessite pas de manipuler la mémoire ou les registres.

La vulnérabilité en question, **CVE-2011-0647 [1]**, intitulée « EMC Replication Manager Command Execution »

6542/tcp	40849 - EMC Replication Manager irccd.exe RunProgram Message Handling Arbitrary Command Execution	[-/+]
Synopsis		
Arbitrary code can be executed on the remote host.		
Description		
The remote host is running EMC Replication Manager Client Control Daemon.		
The remote version of this software is affected by a remote command execution vulnerability.		
An unauthenticated attacker can exploit this flaw by sending a specially crafted packet to the remote host. Successful exploitation would result in remote code execution with the privileges of the daemon itself.		

Fig. 1 : Extrait du rapport Nessus mentionnant la vulnérabilité.

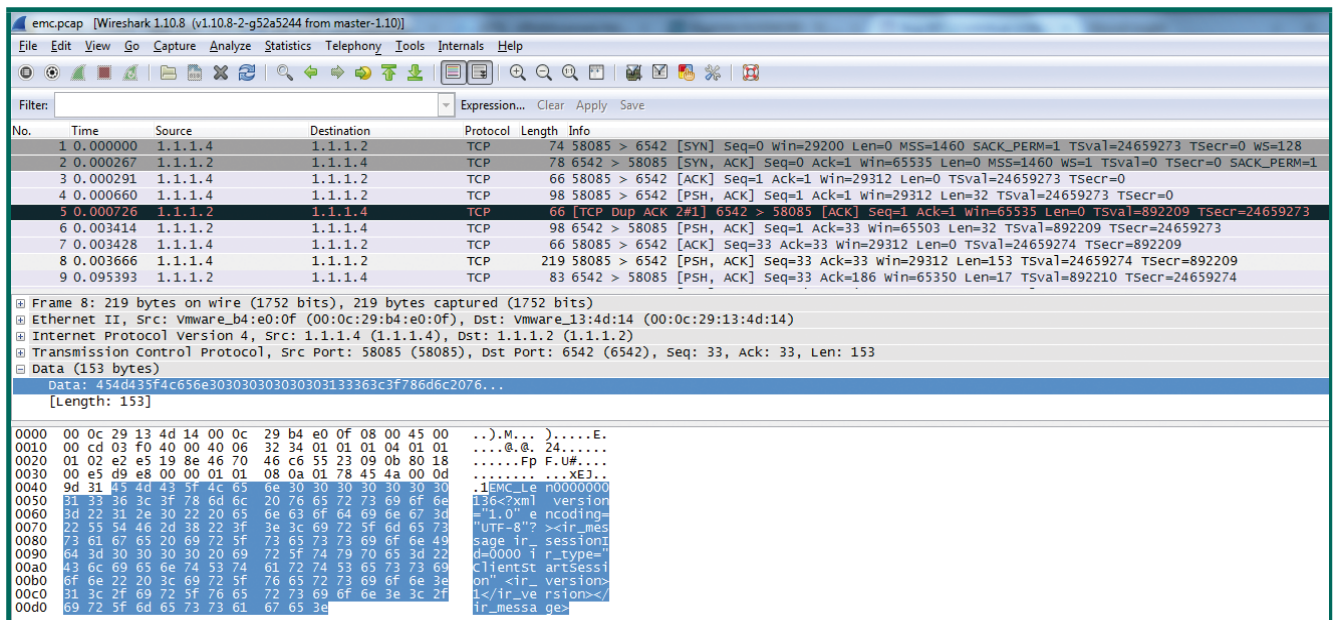


Fig. 2 : Extrait d'une communication avec le démon EMC Replication Manager Client Control.

impacte l'outil de réplication EMC Replication Manager **[2]**. Son score CVSS de 10 et sa belle couleur violette sous Nessus la rendent plutôt attrayante.

Le synopsis de Nessus dit que l'hôte distant exécute le démon EMC Replication Manager Client Control et que la version de ce logiciel est affectée par une vulnérabilité d'exécution distante de commande. Réussir à l'exploiter permettrait d'exécuter des commandes sur la cible et donc de rendre le rapport un peu plus illustré. Nessus précise même que c'est la gestion du message **RunProgram** qui pose problème. Mais comment envoyer ce type de message ?

Malheureusement, ce n'est pas en fouillant du côté des plugins de Nessus que nous aurons la réponse, car le plugin détectant la vulnérabilité fait partie de la fraction qui est compilée. Nous n'aurons donc pas accès facilement au code source en NASL.

Toutefois si Nessus a su détecter la vulnérabilité à distance et que le protocole n'utilise pas de chiffrement, une capture réseau devrait permettre d'identifier le fameux message. La chance est avec nous : pas de chiffrement. Les secrets du plugin ne resteront pas inviolés très longtemps. En effet, la capture permet de comprendre le protocole d'échange : on se dit bonjour, on initie une session, puis on demande le lancement d'un programme.

Après quelques tentatives, nous avons un exploit fonctionnel :

```
root@kali:/home/davy# perl emc_poc.pl --dst 1.1.1.2 --port 6542
--mode run --cmd c:\\windows\\system32\\calc.exe
Mode: run
[*] We send hello...
[*] We send a ping...
[*] We send the endstring to disconnect...
[*] We get back:
```

```
1RAWHELLO0000000000000000000000000000EMC_Len0000000240
<?xml version="1.0" encoding="UTF-8"?>
<ir_message ir_sessionId="00000" ir_requestId="00000" ir_
type="Status" ir_status="0">
<ir_statusType>final</ir_statusType>
<ir_statusTime>2014 07 26 11:49:50</ir_statusTime><errno>0</errno>
</ir_message>
```

1528	0.10	EMC Replication Manager Client C...	EMC Corporation	DEP
5840	0.10	Application Calculatrice de Windows	Microsoft Corporation	DEP

Fig. 3 : Calc.exe est bien exécuté sur la cible.

2 Exploit standalone VS module Metasploit

L'exploit, écrit en Perl, est suffisant pour apporter une conclusion différente à la mission. Nous ne parlerons pas au conditionnel dans le rapport et nous pourrions ajouter l'exploit en annexe. Mais devons-nous en rester là pour autant ?

Le pentester doit travailler vite et les frameworks d'exploitation permettent de gagner un temps fou, car non seulement ils rassemblent des centaines d'exploits fiabilisés et prêts à l'emploi, mais ils fournissent également des outils puissants pour automatiser les tâches de reconnaissance, d'attaques et surtout de post-exploitation.

Intégrer l'exploit sur Metasploit permettrait par exemple de bénéficier de la puissance de meterpreter. Le script Perl pourrait très bien délivrer le premier étage de la charge utile de meterpreter, mais une approche plus élégante serait de créer un exploit Metasploit. Bien sûr, il serait possible de faire ça dans son coin, mais ajouter



l'exploit au repository officiel Metasploit permettrait aux autres utilisateurs d'en bénéficier. En tant qu'utilisateur régulier de Metasploit, cela serait aussi une sorte de remerciement pour les autres contributeurs.

À noter que si dans ce cas l'*exploit* initial a été écrit en Perl, il aurait peut-être été plus simple de l'écrire directement en Ruby pour Metasploit. En effet, Metasploit propose de nombreuses fonctions qui peuvent faciliter la vie : que vous cherchiez à générer un fichier ou des requêtes HTTP, Metasploit saura faire.

Lors d'une discussion avec Juan Vazquez, un des contributeurs de Rapid7 les plus prolifiques, je décide de me lancer dans la soumission de cet exploit. Comme il le fait remarquer, même si l'exploit ne sera probablement pas très utile sur Internet, il pourrait l'être lors de tests d'intrusion internes.

3

Se construire un environnement de développement Metasploit

Les contributions au projet Metasploit se font par le biais de leur GitHub. Metasploit met à disposition un wiki [4] décrivant les étapes nécessaires à l'installation d'un environnement minimal permettant de contribuer. Un premier conseil : lisez-le. J'aurais vraiment aimé qu'il soit aussi complet lorsque je me suis intéressé au sujet pour la première fois. Un deuxième conseil : n'optez pas pour une Kali. Ce fut mon premier choix par facilité, mais à l'installation des prérequis j'ai vite compris qu'il faudrait revoir ce choix pour ne pas perdre trop de temps sur cette première étape. À ce moment, j'ai bien tenté de savoir s'il n'y avait pas d'autres moyens de soumettre un exploit. Réponse sur IRC [5] : « github is the right way ».

Après avoir installé une distribution Ubuntu comme conseillé, j'ai installé git puis créé un compte GitHub. Ensuite, il suffit de configurer git pour qu'il soit lié au compte GitHub. Toutes les étapes sont détaillées sur le wiki ; inutile de revenir dessus. Une fois les prérequis installés correctement on peut entreprendre le développement du module (voir encadré).

Les modules Metasploit !

Dans le langage Metasploit, un module est un script Ruby. Les modules *auxiliary* s'utilisent de manière autonome. Par exemple, le module *auxiliary/scanner/portscan/tcp* va effectuer un balayage de ports, afficher les résultats puis rendre la main. Les modules *exploit* s'utilisent conjointement avec d'autres modules. Par exemple, le module *exploit/windows/browser/ms14_012_textrange*, une fois la vulnérabilité exploitée, va utiliser un module *payload* (ex : *payload/windows/meterpreter/reverse_tcp*) pour délivrer une charge utile.

4

Écriture, soumission et correction

Chaque module est composé de deux parties. La première contient des éléments d'information (description, références, etc.) ainsi que des éléments de configuration (architecture processeur ciblée, taille maximale de la charge utile, port, etc.). Ils sont affichés dans **msfconsole** lorsque l'on tape **info**. Les éléments qui sont à la main de l'utilisateur sont affichés lorsque l'on tape **show options**. La seconde partie du module contient la partie exécutable qui est généralement découpée en fonctions.

Pour débiter l'écriture d'un nouveau module, il est conseillé de se baser sur un module existant fournissant des fonctionnalités similaires. Utilisez de préférence un module récent, car en utilisant un module vieux de plusieurs années vous allez peut-être utiliser des fonctions obsolètes et remplacées. Vous risquez donc de devoir réécrire une partie du module après sa relecture par les équipes Rapid7.

Si vous écrivez un module exploitant une vulnérabilité de type corruption de mémoire comme un débordement de tampon et que vous utilisez Immunity Debugger ou Windbg pour déboguer la vulnérabilité, vous pouvez utiliser mona [6] pour vous préparer un squelette d'exploit au format Metasploit en tapant : **!mona skeleton**.

Lorsque vous souhaitez tester une première version de votre module, vous pouvez déposer le fichier directement dans l'arborescence Metasploit. Dans notre cas, elle se trouve ici : **/home/davy/git/metasploit-framework/modules/**.

Notre exploit peut donc être déposé dans : **/home/davy/git/metasploit-framework/modules/exploits/windows/emc/replication_manager_exec.rb**.

À ce stade, vous aller enfin pouvoir tester votre module en grandeur nature. Il est temps de lancer **msfconsole**.

Vont s'en suivre des allers-retours (dont le nombre est inversement proportionnel à votre niveau de compréhension de Ruby, de Metasploit et de la vulnérabilité) entre votre éditeur de texte et **msfconsole**. Après chaque modification du module, au lieu de recharger Metasploit dans son ensemble, il suffit de taper **reload** pour que le fichier soit rechargé.

Lorsque le module est fonctionnel et stable, il faut vérifier sa conformité avec les critères d'écriture de Metasploit.

Il y a sûrement des problèmes d'indentation, des espaces en trop en fin de ligne, etc. L'outil **msftidy** va vous aider dans la détection de ces erreurs de style.

```
ruby /home/davy/git/metasploit-framework/tools/msftidy.rb /home/davy/git/metasploit-framework/modules/exploits/windows/emc/replication_manager_exec.rb
```

Des vérifications supplémentaires ont été ajoutées récemment [7] pour respecter le Ruby Style Guide [8]. L'objectif de tous ces contrôles est d'augmenter la maintenabilité de Metasploit en standardisant au maximum les modules.

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



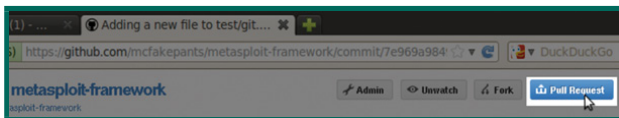


Attention, il est conseillé de mettre à jour son Metasploit avant de lancer **msftidy** pour être sûr d'avoir la dernière version de l'outil.

Par mesure de précaution, une des premières étapes qui est réalisée automatiquement par Rapid7 lors d'une *pull request* est un contrôle **msftidy**. Mais autant être propre dès le départ. Une fois que le module passe **msftidy** sans erreur, le moment est venu de le rendre public en effectuant une *pull request*.

```
ssh -T github
cd /home/davy/git/metasploit-framework
git push origin master
git checkout master
git checkout -b replication_manager_exec
git add modules/exploits/windows/emc/replication_manager_exec.rb
git commit -m "added aux module for EMC Replication Manager Command Execution"
git push origin replication_manager_exec
```

Finalement, après s'être connecté sur GitHub il suffit de cliquer sur le bouton *pull* pour que la *pull request* soit émise.



Le module est maintenant à la main des développeurs de Rapid7 qui vont effectuer leurs revues puis le tester.

Si l'application vulnérable est simple à trouver, ils testeront le module en conditions réelles. Si par contre, elle est payante et qu'ils ne peuvent pas se la procurer, ils demanderont des preuves à l'auteur du module (copie d'écran ou capture réseau).

S'ils n'ont rien à redire sur les fonctionnalités et le style du module, ils l'intégreront directement en « mergeant » la *pull request* dans la branche master. Le module sera instantanément disponible pour tous les utilisateurs.

Ou alors ils peuvent faire des remarques pour corriger ou améliorer certaines parties. Ce fut mon cas. Juan Vazquez de l'équipe Rapid7 m'a aidé à réécrire et à perfectionner ce module. J'ai initialement soumis un bête module *auxiliary* qui n'était rien de plus que la transformation en Ruby du script Perl qui avait été écrit pour le test d'intrusion. Mais, comme Juan me l'a judicieusement fait remarquer, pour les vulnérabilités de type exécution de commande système, il faut systématiquement essayer d'écrire un exploit de manière à bénéficier des sessions meterpreter et donc de tous les outils de post-exploitation. La plupart du temps, ce type de vulnérabilité permet d'aboutir sans encombre à un module de type exploit. Effectivement, en utilisant le mixin **CmdStagerVBS** pour délivrer la charge utile, notre session meterpreter s'établit à tous les coups.

La cinématique d'exploitation est la suivante : le module lance le *handler metasploit*, se connecte sur le service EMC, dit bonjour, initie une session EMC, puis demande le lancement de **cmd /c #{CmdStagerVBS}**. Cette commande écrit sur le système de fichier de la cible le premier étage de la charge sous la forme d'un fichier VBS qui est ensuite exécuté par **CScript**. Ce premier étage va se connecter au *handler* pour récupérer le restant de la charge utile.

Une fois les remarques prises en compte, il faudra effectuer une nouvelle *pull request*. Si cette fois le code est satisfaisant, le module est ajouté à la branche master par Rapid7 [9] et vous recevez une notification de « Land ». Champagne !

Quelques mois plus tôt, Juan avait presque intégralement réécrit un autre module que j'avais coécrit avec un collègue [10]. Il n'est donc pas avare de son temps et de ces conseils.

5 Comment contribuer autrement ?

Combien d'exploits s'ennuient à mourir sur des secteurs de disques durs ou de clés USB alors qu'ils pourraient passer du bon temps à satisfaire des palanquées de pentesteurs et à attiser la curiosité de quelques RSSI ?

Vous l'avez lu, intégrer un module est assez simple et rapide, alors partagez en soumettant les vôtres !

Et même sans disposer d'exploit sous la main, il est possible de contribuer. Car si les exploits sont la matière première de Metasploit, ils seraient inoffensifs s'il n'était pas possible de les combiner avec des charges utiles efficaces. Alors si vous êtes doués en écriture de shellcodes, vous pouvez apporter votre aide sur les modules de type payload. Par exemple, cette *pull request* [11] propose une nouvelle version de meterpreter pour OS X qui sera, à terme, compatible avec les iOS [12].

D'autres contributions, techniquement plus accessibles, peuvent être intéressantes aussi comme un module de post-exploitation ou un auxiliary.

Et finalement, même sans écrire la moindre ligne de code vous pouvez aussi participer en documentant le projet ou en réalisant des screencasts. ■

■ Références

- [1] <http://cvedetails.com/cve/2011-0647>
- [2] <http://france.emc.com/storage/replication-manager.htm>
- [3] <https://github.com/rapid7/metasploit-framework>
- [4] <https://github.com/rapid7/metasploit-framework/wiki/Setting-Up-a-Metasploit-Development-Environment>
- [5] <https://freenode.net/#metasploit>
- [6] <http://redmine.corelan.be/projects/mona>
- [7] <https://community.rapid7.com/community/metasploit/blog/2014/07/17/weekly-metasploit-update-embedded-device-attacks-and-automated-syntax-analysis>
- [8] <https://github.com/bbatsov/ruby-style-guide>
- [9] https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/windows/emc/replication_manager_exec.rb
- [10] https://www.rapid7.com/db/modules/exploit/unix/webapp/spip_connect_exec
- [11] <https://github.com/rapid7/metasploit-framework/pull/34827>
- [12] <http://sourceforge.net/p/metasploit/mailman/message/32514602/>

NOUVEAU ! NOUVEAU ! NOUVEAU ! NOUVEAU !

ABONNEZ-VOUS!

DÉCOUVREZ NOS NOUVELLES OFFRES !

LE CLASSIQUE !

VERSION

PAPIER



Retrouvez tous les 2 mois votre bimestriel favori en papier dans votre boîte à lettres !



Envie de lire tous les 2 mois votre magazine sur votre tablette ou votre ordinateur ? Voici l'abonnement PDF !

Attention, les offres mentionnées dans la page suivante sont réservées aux particuliers, professionnels n'hésitez pas à nous contacter à : abopro@ed-diamond.com

PDF

L'ÉLECTRONIQUE !

VERSION

PDF



LE NOUVEAU SERVICE !

ACCÈS À LA

BASE DOCUMENTAIRE



L'accès à la base documentaire en ligne de MISC et de ses Hors-séries vous permettra d'effectuer des recherches dans la majorité des articles parus, qui seront disponibles avec un décalage de 6 mois après leur parution en magazine.

Attention, les offres mentionnées dans la page suivante sont réservées aux particuliers, professionnels n'hésitez pas à nous contacter à : abopro@ed-diamond.com



boutique.ed-diamond.com

Découvrez une sélection d'offres page suivante



RETROUVEZ TOUTES LES AUTRES OFFRES D'ABONNEMENTS SUR : **boutique.ed-diamond.com !**



Merci de cocher ci-dessous l'offre souhaitée...

PAPIER	TARIF NORMAL	ÉCONOMIE	TARIF SPÉCIAL*
☐ MISC 6 ^{N°}	53,40	11,40	42,-€

PDF**	TARIF NORMAL	ÉCONOMIE	TARIF SPÉCIAL*
☐ MISC 6 ^{N°}	53,40	11,40	42,-€

BASE DOCUMENTAIRE**	TARIF SPÉCIAL*
☐ ACCÈS 1 AN	59,-€

PAPIER + PDF**	TARIF NORMAL	ÉCONOMIE	TARIF SPÉCIAL*
☐ MISC 6 ^{N°} + MISC 6 ^{N°}	106,80	44,80	62,-€

PAPIER + BASE DOCU**	TARIF NORMAL	ÉCONOMIE	TARIF SPÉCIAL*
☐ MISC 6 ^{N°} + ACCÈS 1 AN	112,40	23,40	89,-€

PAPIER + PDF** + BASE DOCUMENTAIRE**	TARIF NORMAL	ÉCONOMIE	TARIF SPÉCIAL*
☐ MISC 6 ^{N°} + MISC 6 ^{N°} + ACCÈS 1 AN	165,80	54,80	111,-€



Merci de cocher ci-dessous l'offre souhaitée...

PAPIER	TARIF NORMAL	ÉCONOMIE	TARIF SPÉCIAL*
☐ MISC 6 ^{N°} + MISC HS 2 ^{N°}	71,40	17,40	54,-€

PDF**	TARIF NORMAL	ÉCONOMIE	TARIF SPÉCIAL*
☐ MISC 6 ^{N°} + MISC HS 2 ^{N°}	71,40	17,40	54,-€

PAPIER + PDF**	TARIF NORMAL	ÉCONOMIE	TARIF SPÉCIAL*
☐ MISC 6 ^{N°} + MISC HS 2 ^{N°} + MISC 6 ^{N°} + MISC HS 2 ^{N°}	142,80	58,80	84,-€

PAPIER + BASE DOCU**	TARIF NORMAL	ÉCONOMIE	TARIF SPÉCIAL*
☐ MISC 6 ^{N°} + MISC HS 2 ^{N°} + ACCÈS 1 AN	130,40	32,40	98,-€

PAPIER + PDF** + BASE DOCUMENTAIRE**	TARIF NORMAL	ÉCONOMIE	TARIF SPÉCIAL*
☐ MISC 6 ^{N°} + MISC HS 2 ^{N°} + MISC 6 ^{N°} + MISC HS 2 ^{N°} + ACCÈS 1 AN	201,80	68,80	133,-€

Vous souhaitez également vous abonner à :



Rendez-vous sur notre boutique pour découvrir toutes nos offres de couplages !
boutique.ed-diamond.com

*Prix TTC France Métro. Pour les tarifs hors France Métro et étranger, veuillez consulter notre boutique en ligne !

**Offre réservée aux particuliers, professionnels contactez-nous par mail : abopro@ed-diamond.com

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 :

ANALYSE DU MALWARE BLAME

Paul Rascagneres – Senior Threat Researcher
G DATA SecurityLabs



mots-clés : APT / REVERSE ENGINEERING / LAME / MALWARE / RAT / OBFUSCATION

Cet article décrit le malware Blame. Ce malware est un Remote Administration Tool (RAT) qui permet de prendre la main sur les systèmes infectés. La plus ancienne version identifiée dans la nature date de 2012, et la plus récente de cette année. Ce malware est donc utilisé depuis plus deux ans. Cet article décrit le dropper, ainsi que l'astuce mise en place par le développeur pour tromper les analystes. Il traite également des systèmes d'obfuscation mis en place.

1 Le dropper

Le dropper dépose sur le système 4 fichiers :

- %TMP%\Jus1.tmp ;
- %TMP%\lts2.tmp ;
- %TMP%\Jus3.tmp ;
- %TMP%\lgt4.tmp.

Les 2 premiers fichiers sont en fait des « .reg ». Leur but est de modifier des valeurs de clé de registre. Le premier permet de supprimer la clé suivante : **HKCU\System\CurrentControlSet\Services\AppMgmt**.

Cette clé de registre est légitime, elle permet de gérer le service *AppMgmt* (gestion d'application). Cependant, le système fonctionnera sans problème sans ce service. Le second fichier a pour but de recréer le service *AppMgmt*. Voici un extrait du fichier **lts2.tmp** :

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\AppMgmt]
"Description"="Provides software installation services such as
Assign, Publish, and Remove."
"DisplayName"="Application Management"
"ErrorControl"=dword:00000000
"ImagePath"=hex(2):25,00,53,00,79,00,73,00,74,00,65,00,6d,00,52,00,
6f,00,6f,00,\
74,00,25,00,5c,00,73,00,79,00,73,00,74,00,65,00,6d,00,33,00,32,00,
5c,00,73,\
00,76,00,63,00,68,00,6f,00,73,00,74,00,2e,00,65,00,78,00,65,00,2
0,00,2d,00,\
6b,00,20,00,6e,00,65,00,74,00,73,00,76,00,63,00,73,00,00,00
"ObjectName"="LocalSystem"
"Start"=dword:00000002
```

```
"Type"=dword:00000020
"DelayedAutoStart"=dword:00000000

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\AppMgmt\
Parameters]
"ServiceDll"=hex(2):43,00,3a,00,5c,00,50,00,72,00,6f,00,67,00,72,00,
00,61,00,6d,\
00,20,00,46,00,69,00,6c,00,65,00,73,00,5c,00,43,00,6f,00,6d,00,6
d,00,6f,00,\
6e,00,20,00,46,00,69,00,6c,00,65,00,73,00,5c,00,4f,00,44,00,42,00,
00,43,00,5c,\
00,4f,00,44,00,42,00,43,00,30,00,5c,00,41,00,70,00,70,00,4d,00,6
7,00,6d,00,\
74,00,2e,00,64,00,6c,00,6c,00,00,00
"ServiceDllUnloadOnStop"=dword:00000001
```

Voici les contenus des deux clés de registre une fois convertis en ASCII :

- %SystemRoot%\system32\svchost.exe -k netsvcs ;
- C:\Program Files\Common Files\ODBC\ODBC0\AppMgmt.dll.

La première chaîne de caractères correspond à la définition du service et la deuxième correspond au chemin de la librairie à charger et exécuter par ce service. Comme vous vous en doutez, cette librairie est le malware lui-même.

Pour en revenir aux fichiers déposés par le dropper, le fichier **Jus3.tmp** est un fichier Cabinet (archive proposée par Microsoft). Nous pouvons extraire le contenu de l'archive sous Linux via la commande **cabextract**. Cette archive contient le fichier **AppMgmt.dll** qui sera déposé dans **C:\Program Files\Common Files\ODBC\ODBC0**. Le dernier fichier est **lgt4.tmp**. Ce fichier est un script **.bat** permettant de supprimer le dropper.

2 Le malware

Le malware dispose des fonctionnalités classiques d'un RAT, il permet de réaliser des captures d'écran, de manipuler des fichiers, des répertoires et bien évidemment d'exécuter des programmes sur la machine. Ces fonctionnalités ne seront pas décrites ici. Les explications porteront sur l'astuce utilisée par les développeurs pour compliquer l'analyse et également sur les techniques d'obfuscation utilisées pour cacher les chaînes de caractères ou l'utilisation de l'API.

2.1 LAME pour les lamers

Lors de l'analyse d'un malware et en particulier d'une librairie, une des premières étapes consiste à regarder la liste des fonctions exportées. Voici une capture de cette liste :

Name	Address	Ordinal
beInitStream	100742E0	1
beEncodeChunk	100750E0	2
beDeinitStream	100751B0	3
beCloseStream	10073D60	4
beVersion	10073210	5
beWriteVBRHeader	10074260	6
beEncodeChunkFloatS16NI	10075080	7
beFlushNoGap	10074280	8
beWriteInfoTag	100741C0	9
lame_set_out_sample	10075200	10
lame_init	1004D240	100
lame_close	1004D2A0	101
lame_init_params	1004D8B0	102
lame_encode_buffer_interleaved	1004FD30	110
lame_encode_flush	1004FFC0	120
lame_mp3_tags_fid	1004D6F0	130
lame_set_num_samples	10050C90	1000
lame_get_num_samples	10050C80	1001
lame_set_in_samplerate	10050C70	1002
lame_get_in_samplerate	10050C60	1003
lame_set_num_channels	10050C30	1004
lame_get_num_channels	10050C20	1005
lame_set_scale	10050C10	1006
lame_get_scale	10050C00	1007
lame_set_scale_left	10050BF0	1008
lame_get_scale_left	10050BE0	1009
lame_set_scale_right	10050BD0	1010

Fig. 1 : Liste des fonctions exportées par la librairie.

La liste semble correspondre aux exports de la célèbre librairie d'encodage MP3 libre LAME. Nous pouvons trouver la liste des exports de la librairie LAME dans le CSV du projet à l'adresse suivante : <http://lame.cvs.sourceforge.net/viewvc/lame/lame/Dll/BladeMP3EncDLL.def>.

Cependant, nous pouvons noter que 2 exports sont présents en plus dans notre sample :

- **lame_set_out_sample ;**
- **lame_get_out_sample.**

Ces deux exports n'existent pas dans le projet LAME. En fait, le développeur a utilisé la librairie LAME et a ajouté la partie malveillante, en ajoutant deux fonctions, tout en gardant le même type de convention pour les nommer.

Cette technique a été utilisée afin de noyer le code malveillant dans un code légitime afin d'en ralentir l'analyse, voire de la fausser en pensant que la librairie est celle de LAME.

2.2 Obfuscation des chaînes de caractères et de l'API

Le développeur de Blame a ajouté une couche d'obfuscation pour :

- les chaînes de caractères en Unicode ;
- les chaînes de caractères en ASCII ;
- les noms des fonctions utilisées.

Pour chacun des points ci-dessus, l'algorithme est le même : DES. Mais le malware utilise 3 clés différentes pour chaque point. Voici un exemple de clé pour l'obfuscation de l'API :

```
push 99h ; int
push offset unk_10089050 ; int
mov [ebp+230h+var_278], 42CC975Eh
mov [ebp+230h+var_274], 0CD0Eh ; clé \x5E\x97\xCC\x42\x8E\xCD\x80\x80
call DES_node_ECB
push [ebp+230h+var_270] ; void *
call ???3@VAXPAX0Z ; operator delete(void *)
lea eax, [ebp+230h+var_20C]
```

Fig. 2 : Clé DES pour le déchiffrement de l'API.

Nous pouvons voir dans la capture suivante le déchiffrement du nom des fonctions présentes dans **kernel32.dll** :

7295780C	ES	PUSH EBP	
7295780D	EB	MOV EBP, ESP	
7295780E	EB	MOV EBX, 72908A90	ASCII "CloseHandle"
7295780F	EB	PUSH EDI	
72957810	EB	CALL EBX	
72957811	EB	PUSH EDI	ASCII "CompareFileTime"
72957812	EB	CALL EBX	
72957813	EB	PUSH EDI	
72957814	EB	CALL EBX	ASCII "CopyFileW"
72957815	EB	PUSH EDI	
72957816	EB	CALL EBX	ASCII "CreateDirectoryW"
72957817	EB	PUSH EDI	
72957818	EB	CALL EBX	ASCII "CreateEventW"
72957819	EB	PUSH EDI	
7295781A	EB	CALL EBX	ASCII "CreateFileW"
7295781B	EB	PUSH EDI	
7295781C	EB	CALL EBX	ASCII "CreateFileW"
7295781D	EB	PUSH EDI	
7295781E	EB	CALL EBX	ASCII "CreateFileW"
7295781F	EB	PUSH EDI	
72957820	EB	CALL EBX	ASCII "CreateFileW"
72957821	EB	PUSH EDI	
72957822	EB	CALL EBX	ASCII "CreateFileW"
72957823	EB	PUSH EDI	
72957824	EB	CALL EBX	ASCII "CreateFileW"
72957825	EB	PUSH EDI	
72957826	EB	CALL EBX	ASCII "CreateFileW"
72957827	EB	PUSH EDI	
72957828	EB	CALL EBX	ASCII "CreateFileW"
72957829	EB	PUSH EDI	
7295782A	EB	CALL EBX	ASCII "CreateFileW"
7295782B	EB	PUSH EDI	
7295782C	EB	CALL EBX	ASCII "CreateFileW"
7295782D	EB	PUSH EDI	
7295782E	EB	CALL EBX	ASCII "CreateFileW"
7295782F	EB	PUSH EDI	
72957830	EB	CALL EBX	ASCII "CreateFileW"
72957831	EB	PUSH EDI	
72957832	EB	CALL EBX	ASCII "CreateFileW"
72957833	EB	PUSH EDI	
72957834	EB	CALL EBX	ASCII "CreateFileW"
72957835	EB	PUSH EDI	
72957836	EB	CALL EBX	ASCII "CreateFileW"
72957837	EB	PUSH EDI	
72957838	EB	CALL EBX	ASCII "CreateFileW"
72957839	EB	PUSH EDI	
7295783A	EB	CALL EBX	ASCII "CreateFileW"
7295783B	EB	PUSH EDI	
7295783C	EB	CALL EBX	ASCII "CreateFileW"
7295783D	EB	PUSH EDI	
7295783E	EB	CALL EBX	ASCII "CreateFileW"
7295783F	EB	PUSH EDI	
72957840	EB	CALL EBX	ASCII "CreateFileW"

Fig. 3 : Déchiffrement des fonctions présentes dans kernel32.dll.

Ces noms de fonctions sont ensuite stockés dans des variables. Ces variables sont utilisées par la fonction **GetProcAddress()** afin d'obtenir l'adresse correspondante à chaque nom de fonction décryptée.

Le malware utilise alors directement l'adresse pour exécuter ces fonctions, rendant l'analyse statique plus compliquée, car IDA Pro affiche alors « *call ebx* » et pas « *call functionName* », comme habituellement.

Voici un exemple de script python pour déchiffrer les noms des fonctions :

```
from Crypto.Cipher import DES
import sys
```

```
fp = open(sys.argv[1], 'r')
fp.seek(0x8C608)
i=0
while (i <= 12850):
    data = fp.read(0x38)
    key=(b"\x5B\x5F\xCB\x8D\xE5\xF5\x00\x00")
    obj = DES.new(key, 1)
    print obj.decrypt(data).split('\x00\x00')[0]
    i=i+0x38
fp.close
```

2.3 Obfuscation de l'adresse du serveur de commandes

Pour l'obfuscation de l'adresse du serveur de commandes, le développeur a choisi un algorithme beaucoup plus simple que précédemment. Voici une capture de la fonction qui décode l'adresse du serveur :

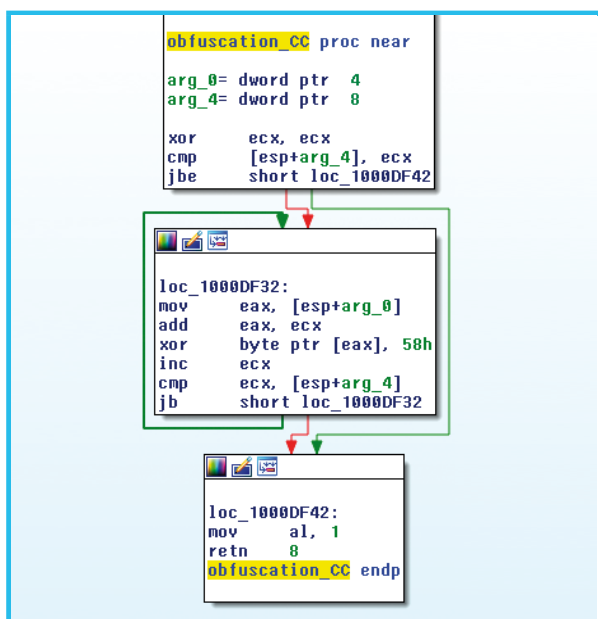


Fig. 4 : Fonction de décodage de l'adresse du serveur de commandes.

L'algorithme est simplement un xor 0x58 sur chaque caractère composant le nom de domaine.

Conclusion

Cet exemple nous a permis de voir une astuce permettant de cacher un malware dans un programme légitime. Un cas pratique de l'expression « trouver une aiguille dans une botte de foin ». Deux méthodes d'obfuscation des données ont aussi été expliquées : l'utilisation d'un algorithme de chiffrement connu et l'utilisation d'un simple encodage de caractères. ■

NE LE MANQUEZ PAS : OPEN SILICIUM N°12 !

FOCUS : PRENEZ EN MAIN LES TECHNOLOGIES RFID / NFC



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





VOLATILISONS LINUX : PARTIE 2

Frédéric Baguelin – frederic.baguelin@arxsys.fr – @udgover

mots-clés : **LINUX / MÉMOIRE / VOLATILE / ANALYSE / VOLATILITY / ROOTKIT / HOOKS / KERNEL / PROCESSUS / NETWORK**

Cet article fait suite au premier publié dans le numéro 72. La première partie présentait l'acquisition de la mémoire volatile d'un système GNU/Linux ainsi que les « internals » de Volatility. Cet article présente les commandes et les possibilités d'analyses associées avec un rappel sur les structures noyaux utilisées.

1 Introduction

Avant d'entamer l'analyse, je vais revenir rapidement sur l'acquisition. Dans la partie 2.2 de l'article précédent, un teaser présentait la phrase suivante : « Sachez que certaines personnes travaillent actuellement sur une méthode de chargement d'un module pré-compilé et ce quelle que soit la version du noyau ». Ce travail a finalement abouti et a été présenté à Amsterdam à la première édition européenne du DFRWS. Le but de l'article n'est pas de présenter le fonctionnement complet du projet. Brièvement, la première étape repose sur la possibilité de *parasiter* un module déjà présent sur le système grâce au fait qu'un module noyau soit un objet ELF et puisse donc être lié à d'autres objets. Le lecteur intéressé pourra se délecter du papier intitulé « Robust Linux memory acquisition with minimal target impact », coécrit par Johannes Stüttgen et Michael Cohen ou bien regarder la présentation, les deux étant disponibles sur le site de la conférence [DFRWS14]. Pour les plus impatients, le code est accessible sur Google code [LMAP].

Petit bémol cependant, car Volatility ne supporte pas actuellement le format de fichier résultant de cette acquisition qui n'est autre qu'un fichier de type ELF64. Je ne doute pas que les développeurs de Volatility fourniront prochainement le module gérant cet *address space* mais en attendant, l'acquisition est exploitable au travers de rekall, le fork de Volatility. Le projet ressemble beaucoup à Volatility, mais il existe de subtiles différences internes.

Revenons à nos moutons : l'analyse d'une acquisition mémoire GNU/Linux avec Volatility. L'objectif est de présenter les différentes commandes implémentées dans le projet et permettant de mener à bien une analyse,

que ce soit la récupération des différents processus, l'obtention des connexions réseaux créées ou, de la potentielle présence de *rootkits*. Afin de mettre en contexte les différentes commandes, l'article sera basé sur plusieurs codes permettant d'exploiter un système comme le ferait une personne malveillante.

2 Généralités

2.1 Les informations présentées

L'article précédent présentait les méthodes d'acquisition de la mémoire volatile d'un système GNU/Linux, la création d'un profil, son exploitation à l'aide de Volatility et la manière dont il était utilisé, pour présenter enfin le fonctionnement des *Address Space*. La suite de l'article va se concentrer sur les commandes utilisables sur une acquisition d'un système GNU/Linux en suivant la même catégorisation que la documentation en ligne sur le wiki du projet [VOLINUX] tout en expliquant sur quelles informations Volatility se base pour afficher ses résultats. Ainsi les informations suivantes seront présentées :

- les informations relatives au système ;
- l'analyse des processus ;
- les objets du noyau ;
- les informations relatives au réseau ;
- les détections de codes malicieux.

2.2 L'utilisation du profil et l'interprétation des données

La majorité des informations extraites par Volatility résulte de l'utilisation des symboles du noyau présents dans le fichier **System.map**. Si vous lisez le code source des différentes commandes, vous verrez que la majorité d'entre elles utilise la fonction **get_symbol()**, implémentée dans le fichier **volatility/plugins/overlays/linux/linux.py**, qui retourne l'adresse du symbole souhaité. Ce symbole est ensuite interprété par l'instanciation d'un objet **Object** utilisant la description de la structure au format dwarf comme présenté dans l'article précédent. Dans la suite de l'article, le symbole associé à une commande, et donc à l'information extraite, sera détaillé.

2.3 L'invocation des commandes

L'invocation des commandes relatives à l'analyse d'un système GNU/Linux est similaire à celle utilisée pour Windows. La seule différence est l'utilisation du préfixe **linux**. De manière générale, l'invocation d'une commande Volatility se fait comme suit lorsque l'on se trouve à la racine des sources du projet :

```
$ python vol.py -profile=ProfileDuSystème -f /chemin/vers/le/dump.  
bin <commande>
```

- **--profile** correspond au profil qui doit être utilisé par Volatility pour être en mesure d'interpréter les données. Il est possible de consulter la liste des profils en exécutant la commande **python vol.py --info | less** et de se rendre à la section **Profiles**.
- **-f** permet de fournir le chemin vers l'acquisition mémoire à analyser. Il peut être relatif ou absolu.
- **commande** correspond à la commande à utiliser. Il est possible de les lister en exécutant **python vol.py --info | grep linux_**

Des options supplémentaires peuvent être fournies pour, par exemple, activer le **debug**, utiliser un cache pour certaines informations récurrentes, paramétrer la **timezone**, etc. Le lecteur curieux se référera à la documentation générale **[VOLUSAGE]**.

2.4 Practice yourself

Après ce rappel, il est temps de présenter les différentes commandes proposées. Pour vous exercer, de l'acquisition à l'analyse, une machine virtuelle, créée avec VirtualBox, est mise à votre disposition **[VBOX]**. Celle-ci est basée sur un système Debian Wheezy (7.5) en 64 bits. Le mot de passe du compte root est **toor**. À la racine du compte utilisateur **misc** (mot de passe **misc**), se trouvent les répertoires et fichiers suivants :

- **suterusu** : qui correspond au code open source d'un rootkit permettant de cacher des processus ou des connexions réseaux ainsi que de poser des hooks syscall ou de faire du keylogging. Au début de la rédaction de l'article, le code source accessible sur GitHub **[SUTERUSU]** a été modifié dans la machine virtuelle afin de présenter les mécanismes de détection de Volatility version 2.3.1. En effet, les techniques de hooks employées par ce rootkit n'étaient pas détectées par la version 2.3.1. Depuis la finalisation de l'article, la version 2.4 est disponible et détecte désormais les mécanismes utilisés par **suterusu**. Le lecteur est donc invité à compiler lui-même la dernière version pour le constater.
- **profile** : contient le code permettant de générer le profil de manière automatisée. Il suffit d'exécuter la commande **make** dans ce répertoire.
- **lime-forensics-1.1-r17** : contient le code du module noyau pour faire l'acquisition comme présentée dans l'article précédent.

Pour accéder simplement à l'acquisition, le script **mount_sf.sh** permet de monter un répertoire partagé avec le système hôte. Avant de lancer le script, il faut ajouter le répertoire en question à travers VirtualBox et associer au chemin en question le nom **vbox** comme sur la figure 1. Le point de montage dans la machine virtuelle se situe dans le répertoire **share**.

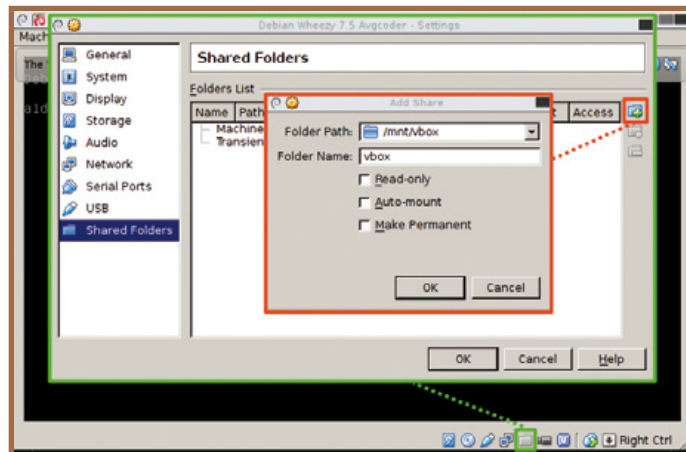


Figure 1 : Ajout d'un répertoire partagé à une machine virtuelle dans VirtualBox.

3 Informations du système

La première étape consiste à obtenir les informations de base sur le système. Cela permet de s'assurer que c'est la bonne machine qui est ciblée. La première commande à lancer affiche la version du système par le biais de la commande **linux_banner** qui récupère l'adresse du symbole éponyme et affiche la chaîne de caractères qui s'y trouve :



```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_banner
Volatility Foundation Volatility Framework 2.3.1
Linux version 3.2.0-4-amd64 (debian-kernel@lists.debian.org) (gcc
version 4.6.3 (Debian 4.6.3-14) ) #1 SMP Debian 3.2.57-3+deb7u1
```

L'obtention des informations relatives au CPU résulte de l'exécution de la commande **linux_cpuinfo**. Cette commande récupère le symbole **cpu_info**, puis selon le type d'architecture (mono ou multi cpu) va récupérer différentes structures pour extraire les informations :

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_cpuinfo
Volatility Foundation Volatility Framework 2.3.1
Processor      Vendor      Model
-----
0              AuthenticAMD  AMD Phenom(tm) II X6 1090T Processor
```

Avant de travailler sur les processus et les différentes zones mémoire, il peut être intéressant de regarder du côté des logs du noyau grâce à la commande **linux_dmesg**. Encore une fois, le symbole éponyme est utilisé. Cependant, il existe une différence depuis l'apparition des noyaux 3.0. En effet, ce n'est plus un tableau de type **char**, mais des structures consécutives de type **log** qui sont utilisées. L'exécution de cette commande permet, par exemple, d'obtenir les paramètres fournis au noyau ou encore de voir l'activité des points de montage physique en recherchant la chaîne **sda** dans le log grâce à **grep** comme dans le listing suivant :

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_dmesg | grep BOOT_IMAGE
Volatility Foundation Volatility Framework 2.3.1
<6>[ 0.000000] Command line: BOOT_IMAGE=/boot/vmlinuz-3.2.0-4-
amd64 root=UUID=2638027f-de2b-4f86-a668-5523f6ec9e37 ro quiet
<5>[ 0.000000] Kernel command line: BOOT_IMAGE=/boot/vmlinuz-
3.2.0-4-amd64 root=UUID=2638027f-de2b-4f86-a668-5523f6ec9e37 ro quiet
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_dmesg | grep sd
Volatility Foundation Volatility Framework 2.3.1
<5>[ 1.101424] sd 0:0:0:0: [sda] 16777216 512-byte logical blocks:
(8.58 GB/8.00 GiB)
[...]
<6>[ 1.792592] EXT4-fs (sda1): mounted filesystem with ordered
data mode. Opts: (null)
<6>[ 4.799232] Adding 392188k swap on /dev/sda5. Priority:-1
extents:1 across:392188k
```

4 Analyse des processus

La majorité des commandes relatives aux processus repose sur l'utilisation de la structure **task_struct**, définie dans **include/linux/sched.h**. Elle est équivalente à la structure **Eprocess** pour Windows. Cette structure de données représente un processus dans le noyau Linux et contient des informations simples comme son nom, son état actuel, son uid, gid et des informations plus complexes telles que les allocations mémoires ou les descripteurs de fichiers ouverts.

Chaque structure **task_struct** contient un champ **tasks** de type **list_head**. Ce dernier type est au cœur du mécanisme officiel de gestion des listes chaînées dans le noyau Linux. Cette structure de listes chaînées contient les champs **prev** et **next** qui, dans le contexte d'une **task_struct**, pointeront respectivement sur la **task_struct** précédente et suivante. La gestion de ces listes chaînées est directement prise en compte par Volatility avec une possibilité d'itération par boucle **for** sur les structures contenant une **list_head**.

4.1 Lister les processus

Comment récupérer la liste des processus à partir d'une acquisition mémoire ? Pour comprendre le cheminement, un rappel sur l'initialisation d'un système GNU/Linux s'impose. Au cours de l'initialisation du noyau, l'ancêtre de tous les processus, le processus 0 ou *idle process* aussi appelé *swapper process*, est créé. Contrairement à tous les autres processus, ses structures sont allouées de manière statique. Parmi ces structures, celle qui nous intéresse est accessible à travers la variable **init_task** qui correspond, vous l'aurez deviné, à une **task_struct**.

Volatility récupère l'adresse du symbole **init_task** pour ensuite accéder à la liste des processus. C'est précisément le mécanisme implémenté par la commande **linux_pslist**. L'extrait de code suivant est implémenté dans la méthode **calculate()** du fichier **volatility/plugins/linux/pslist.py**.

```
init_task_addr = self.addr_space.profile.get_symbol("init_task")
init_task = obj.Object("task_struct", vm = self.addr_space, offset
= init_task_addr)
for task in init_task.tasks:
    yield task
```

Le résultat de cette commande est le suivant :

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran.lime linux_pslist
Volatility Foundation Volatility Framework 2.3.1
Offset      Name      Pid      Uid      Gid      DTB      Start Time
-----
0xfffff8800f89e740 init      1        0        0        0x00000000fa540000 2014-08-04 15:21:45 UTC+0000
0xfffff8800f89e040 kthreadd 2        0        0        0x00000000c7f20000 2014-08-04 15:21:45 UTC+0000
[...]
0xfffff8800d64d140 exim4    2334    101     104    0x00000000c7bb0000 2014-08-04 15:21:52 UTC+0000
0xfffff8800db3c040 nginx    2359     0        0        0x00000000c7f20000 2014-08-04 15:21:52 UTC+0000
[...]
0xfffff8800c5b3140 sshd     2425    1000    1000    0x00000000c5ca0000 2014-08-04 15:22:14 UTC+0000
0xfffff8800f222740 bash     2426    1000    1000    0x00000000c8020000 2014-08-04 15:22:14 UTC+0000
[...]
0xfffff8800f1290c0 wget     2705    1000    1000    0x00000000f2510000 2014-08-04 15:28:54 UTC+0000
0xfffff8800f916040 bash     2706    1000    1000    0x00000000f0d10000 2014-08-04 15:28:59 UTC+0000
0xfffff8800f0d7740 sudo     2707     0        1000    0x00000000cd950000 2014-08-04 15:28:59 UTC+0000
0xfffff8800c5e2880 insmod   2708     0        0        0x00000000ec2f0000 2014-08-04 15:28:59 UTC+0000
```

Vous noterez l'absence du processus *swapper* (processus 0) qui est un choix de l'équipe du projet. La première colonne correspond à l'adresse logique de l'objet **task_struct**, suivie du nom, du pid, uid et gid du processus par l'utilisation des membres **comm**,



pid, **uid** et **gid** respectivement. La colonne DTB, pour *Directory Table Base*, correspond à l'adresse physique de la base de la table de répertoire des pages utilisée par le mécanisme de pagination pour traduire une adresse linéaire vers une adresse physique. La traduction d'adresse logique à linéaire repose sur le mécanisme de segmentation. C'est cette valeur qui est mise dans le registre de contrôle **CR3** de la CPU lors d'un changement de contexte de processus. Cette adresse est associée au membre **task_struct.mm.pgd**. Cette structure sera détaillée dans une autre section de l'article ayant trait à l'accès des zones mémoire.

Dans l'exemple précédent, vous aurez remarqué que la colonne DTB pour le processus **kthreadd** est vide. Cela vient du fait que c'est un thread noyau qui utilise donc l'espace d'adressage du noyau. Ce qu'il se passe en réalité, c'est qu'un thread noyau se voit réattribuer l'espace d'adressage du dernier processus actif étant donné que l'espace d'adressage noyau est commun à tous les processus.

La dernière colonne fournit l'information temporelle de démarrage du processus grâce au membre **start_time**.

Il est possible d'enrichir les résultats de la commande précédente en affichant les arguments fournis au processus grâce à la commande **linux_psaux** qui reproduit le comportement de la commande **ps aux**. Les arguments fournis à un processus sont stockés dans un tampon accessible à l'adresse linéaire **task_struct.mm.arg_start**, d'une taille de **task_struct.mm.arg_end**.

Pour finir, il est possible de présenter la relation des processus sous forme d'arborescence. La méthode consiste à utiliser les membres **children** pour traverser les fils et **sibling** pour itérer sur les processus de même niveau grâce à la commande **linux_pstree** :

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_pstree
Volatility Foundation Volatility Framework 2.3.1
Name      Pid      Uid
init      1        0
..udev    291      0
[...]
..exim4    2334     101
..nginx    2359     0
..nginx    2362     33
..nginx    2363     33
..nginx    2364     33
..nginx    2365     33
..sshd     2377     0
..sshd     2423     0
...sshd    2425     1000
....bash   2426     1000
.....nc    2570     1000
.....bash  2706     1000
.....sudo  2707     0
.....insmod 2708     0
..sshd     2586     0
..sshd     2588     1000
...bash    2589     1000
.....wget  2705     1000
[...]
```

4.2 Espace d'adressage

Les sections suivantes présentent les commandes et structures utilisées pour récupérer l'espace mémoire d'un processus utilisateur. Le noyau représente celles-ci avec une structure appelée *Memory Descriptor* contenant toutes les informations relatives à l'espace d'adressage. Cette structure est représentée par **mm_struct** déclarée dans **linux/sched.h**. Elle est directement accessible depuis la **task_struct** par le membre **mm**. Le lecteur attentif aura remarqué que cette structure est aussi utilisée par la commande **linux_psaux**.

4.2.1 Répertoire des tables de page

À partir du membre **mm**, il est possible d'accéder à l'adresse physique pointant à la base de la page globale de répertoire par le biais du membre **pgd** présenté en section 4.1. L'article précédent expliquait que Volatility implémente les différents systèmes de translation d'adresse et notamment celui de la pagination, utilisé par la commande **memmap**, qui permet d'obtenir les pages physiques propres au processus. La sortie de la commande présentée ci-après affiche pour chaque processus les adresses virtuelles, physiques et la taille de chaque page allouée. Il est possible de filtrer un processus donné en renseignant l'argument **-p** suivi du **PID** en question.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_memmap -p 2725
Volatility Foundation Volatility Framework 2.3.1
Task      Pid      Virtual      Physical      Size
-----
wget      2725     0x0000000000400000 0x00000000006401000 0x1000
wget      2725     0x0000000000401000 0x00000000006400000 0x1000
[...]
wget      2725     0x00007f0d21600000 0x000277b1cfe00000 0x200000
wget      2725     0x00007f0d21800000 0x00032ad0d6a00000 0x200000
[...]
wget      2725     0x00007f77c0000000 0x000fffffc0000000 0x40000000
wget      2725     0x00007f7800000000 0x000fffffc0000000 0x40000000
[...]
wget      2725     0x0000fffff5ff000 0x0000000001695000 0x1000
wget      2725     0x0000fffff600000 0x0000000001602000 0x1000
```

Il est possible d'obtenir des informations plus précises concernant les différentes allocations mémoire d'un processus en utilisant le mécanisme d'allocation des ressources mémoire, les *Virtual Memory Area* (VMA).

4.2.2 Virtual Memory Area

Une autre façon d'accéder aux différentes zones de l'espace d'adressage d'un processus consiste à utiliser le membre **mmap** qui pointe sur la tête de la liste chaînée d'une structure **VMA** et représentée par une structure **vm_area_struct** définie dans **linux/mm_types.h**. Celle-ci représente une zone de mémoire unique dans un espace d'adressage donné (un processus).



Une VMA peut représenter différents types de zones mémoire tels qu'un fichier mappé en mémoire ou la pile d'un processus utilisateur. Une zone est définie par l'intervalle représenté par les membres **vm_start** (inclusif) et **vm_end** (exclusif). C'est à partir de cette structure qu'il est également possible de connaître le chemin vers les fichiers mappés en mémoire en utilisant le membre **vm_file** de type **file**. Si ce pointeur n'est pas nul, il est possible d'accéder aux informations du fichier par l'intermédiaire du champ **f_path**.

Cette structure est utilisée par la commande **linux_proc_maps** qui affiche les informations pour chaque VMA d'un processus. Le résultat de la commande affiche le PID, l'adresse linéaire de début et de fin de la VMA, les différents flags qui lui sont associés (lecture, écriture, exécution). Si la région correspond à un fichier mappé en mémoire, le champ **vm_pgoff** correspond à l'offset dans le fichier en question. Sont ensuite affichés le numéro de *major* et *minor* du pilote, le numéro d'*inode* associé au fichier et enfin son nom. Lorsque la zone fait partie du tas ou de la pile, le nom associé sera respectivement **[heap]** et **[stack]**. Il est possible d'afficher uniquement les VMA pour un processus donné en passant le PID à l'argument **-p**.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran.lime linux_proc_maps -p 2359
Volatility Foundation Volatility Framework 2.3.1
Pid Start End Flags Pgoff Major Minor Inode File Path
-----
2359 0x00000000400000 0x000000004ba000 r-x 0x0 8 1 152299 /usr/sbin/nginx
2359 0x000000006b5000 0x000000006ba000 r-- 0xb9000 8 1 152299 /usr/sbin/nginx
2359 0x000000006ba000 0x000000006ce000 rw- 0xba000 8 1 152299 /usr/sbin/nginx
2359 0x000000006ce000 0x000000006dd000 rw- 0xd 0 0 0
2359 0x000000001140000 0x00000000119e000 rw- 0xd 0 0 0 [heap]
2359 0x00007f0e5ff34000 0x00007f0e5ff3f000 r-x 0xd 8 1 56 /lib/x86_64-linux-gnu/
libnss_files-2.13.so
2359 0x00007f0e5ff3f000 0x00007f0e6013e000 --- 0xb000 8 1 56 /lib/x86_64-linux-gnu/
libnss_files-2.13.so
[...]
2359 0x00007f0e61b1c000 0x00007f0e61b50000 r-x 0xd 8 1 148735 /usr/lib/libGeoIP.so.1.4.8
2359 0x00007f0e61b50000 0x00007f0e61d4f000 --- 0x34000 8 1 148735 /usr/lib/libGeoIP.so.1.4.8
[...]
2359 0x00007f0e62f5e000 0x00007f0e62fb4000 r-x 0xd 8 1 131514 /usr/lib/x86_64-linux-gnu/
libssl.so.1.0.0
2359 0x00007f0e62fb4000 0x00007f0e631b4000 --- 0x56000 8 1 131514 /usr/lib/x86_64-linux-gnu/
libssl.so.1.0.0
[...]
2359 0x00007fff959b9000 0x00007fff959db000 rw- 0xd 0 0 0 [stack]
2359 0x00007fff959db000 0x00007fff95a00000 r-x 0xd 0 0 0
```

Si vous souhaitez accéder au contenu d'une VMA, il faut utiliser la commande **linux_dump_map** qui prend en paramètre le numéro de processus avec l'argument **-p**, l'adresse de la VMA à récupérer, préalablement fournie par **linux_proc_map**, avec l'argument **-s** et enfin le nom du répertoire où le fichier sera extrait avec **-D**. Le fichier créé a pour nomenclature **task.<pid>.<vma_addr>.vma**.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran.lime linux_dump_map -p 2359 -s 0x00000000400000 -D /tmp
Volatility Foundation Volatility Framework 2.3.1
Task VM Start VM End Length Path
-----
```

```
2359 0x00000000400000 0x000000004ba000 0xba000 /tmp/
task.2359.0x400000.vma
$ file /tmp/task.2359.0x400000.vma
/tmp/task.2359.0x400000.vma: ELF 64-bit LSB executable, x86-64, version 1
(SYSV), dynamically linked (uses shared libs), stripped
```

4.3 Lister les fichiers ouverts

La liste des fichiers ouverts par tous les processus s'obtient par la commande **linux_lsof**. Elle utilise le membre **files** de la structure **task_struct**, de type **files_struct** permettant de pointer sur un tableau de *file descriptors* (**struct fdtable *fdt**). Chaque entrée de ce tableau pointe sur une structure de type **file**. La boucle est bouclée, il est maintenant possible de savoir à quel fichier correspond un *fd* ouvert ! Pour travailler uniquement sur un processus donné, l'argument **-p** suivi du PID peut être utilisé.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran.lime linux_lsof -p 2364
Volatility Foundation Volatility Framework 2.3.1
Pid FD Path
-----
2364 0 /dev/null
2364 1 /dev/null
2364 2 /var/log/nginx/error.log
2364 3 /
2364 4 /var/log/nginx/access.log
2364 5 /var/log/nginx/error.log
2364 6 /
2364 7 /
2364 8 /
2364 11 /
2364 12 /[eventpoll]
```

5 Modules noyau

Un module noyau est représenté par la structure **module** définie dans le fichier **linux/module.h**. Les deux commandes suivantes utilisent des champs de cette structure pour extraire les informations voulues.

5.1 Lister les modules

La liste des modules actuellement chargés dans l'espace mémoire noyau s'obtient en exécutant la commande **linux_lsmod**. Celle-ci récupère l'adresse du symbole **module**, donnant accès au premier module puis parcourt la liste chaînée des modules représentée par le membre **list** de la structure **module**. Cette commande peut recevoir deux arguments optionnels :

- **-S** pour afficher les différentes sections du module par l'utilisation de la structure détaillant les attributs de chacune d'entre elles : **module_sect_attrs** définie

dans **kernel/module.c**. Celle-ci contient un membre **nsections** représentant la taille d'un tableau de structures **module_sect_attr** nommé **attrs**.

- **-P** pour afficher les paramètres fournis au module.

```
$ python vol.py --profile=LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_lsmod -P -S
lime 17991
    .note.gnu.build-id      0xfffffffffa0262000
    .text                  0xfffffffffa0261000
    .rodata                0xfffffffffa0262024
    .rodata.str1.1         0xfffffffffa0262034
    __param                0xfffffffffa0262058
    .data                  0xfffffffffa0263000
    .gnu.linkonce.this_module 0xfffffffffa0263010
    .bss                   0xfffffffffa0263260
    .symtab                0xfffffffffa0036000
    .strtab                0xfffffffffa00363c0
    format=lime
    dio=Y
    path=aldebaran.lime
[...]
```

En omettant les paramètres **-P** et **-S**, le chiffre après le nom du module représente l'addition des champs **init_size** et **core_size**. **init_size** représente la taille de la section **.init** qui contient le code d'initialisation. **core_size** représente quant à elle la taille de la section **.text** contenant le code du module à proprement parler.

5.2 Extraire les modules

Dans le cadre d'une étude d'un module noyau, il est possible d'extraire son code grâce à la commande **linux_moddump**. Celle-ci repose sur l'utilisation du membre **module_core** qui représente l'adresse de début du module et le membre **core_size** représentant la taille de la section **core**. Il est nécessaire de fournir l'argument **-D** à la commande pour spécifier le chemin où sera extrait le module. Enfin, il est possible d'extraire un seul module en fournissant une expression régulière avec l'argument **-r**.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_moddump -r lime -D /tmp/
Volatility Foundation Volatility Framework 2.3.1
$ strings /tmp/lime.0xfffffffffa02af000.lkm
[...]
format
path
lime
padded
tcp:%d
System RAM
lime
setup_tcp
setup_disk
cleanup_module
init_module
write_vaddr_disk
write_vaddr_tcp
cleanup_tcp
cleanup_disk
```

Forensics Training
Red Team
Penetration Tests R&D
Reversing
Security audits **Code review**
Incident response
Exploits Secure hosting

Venez nous rencontrer à NoSuchCon '14
du 19 au 21 novembre 2014



 @synacktiv
www.synacktiv.com
contact@synacktiv.com

6 Réseau

Dans le cas d'une analyse de système corrompu, il est plus que probable que des connexions réseaux illégitimes aient été effectuées vers l'extérieur. Dans la version simple, ces connexions ne sont pas cachées. Les connexions dissimulées seront présentées en section 7.

6.1 Connexions

La première commande utile va être de lister le cache ARP en utilisant **linux_arp**. Cette commande récupère le symbole **neigh_tables** de type **neigh_table**, défini dans le fichier **net/neighbour.h**, puis parcourt les éléments de la liste chaînée (membre **next**). Chaque élément de cette liste contient un tableau de type **neighbour** permettant de pointer sur les informations finales telles que l'adresse IP (v4 ou v6), le nom du périphérique réseau par lequel transiter et la MAC.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran.lime linux_arp
Volatility Foundation Volatility Framework 2.3.1
[::] at 00:00:00:00:00:00 on lo
[0.0.0.0] at 00:00:00:00:00:00 on lo
[10.0.2.2] at 52:54:00:12:35:02 on eth0
```

Pour obtenir les informations relatives aux interfaces réseau actives, l'équivalent de la commande **ifconfig** est implémenté par **linux_ifconfig**. Sur les noyaux récents, le symbole **net_namespace_list** est utilisé, sur les versions plus anciennes, ce sera le symbole **dev_base**. Pour les deux types, les informations de chaque carte sont accessibles en parcourant la liste chaînée des structures **net_device**. Cette dernière fournit l'adresse MAC du périphérique réseau et indique si la carte est en mode *promiscuous* ou non. La liste des interfaces réseaux associées aux périphériques (gestion des alias) est obtenue en parcourant la liste **ifa_list** du membre **ip_ptr**.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran.lime linux_ifconfig
Volatility Foundation Volatility Framework 2.3.1
Interface IP Address MAC Address Promiscuous Mode
-----
lo 127.0.0.1 00:00:00:00:00:00 False
eth0 10.0.2.15 08:00:27:e5:6b:0a False
```

Pour les versions du noyau antérieures à 3.6, il est possible d'obtenir les informations contenues dans le cache de la table de routage. Pour les versions plus récentes, ce cache n'existe plus car cela pouvait être une source de déni de service [**ROUTE CACHE**]. Le symbole **rt_hash_table** est obtenu puis, pour chaque élément du tableau pointé, une nouvelle liste (**chain**) est parcourue pour fournir l'interface, la destination et la passerelle.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran.lime linux_route_cache
Volatility Foundation Volatility Framework 2.3.1
Interface Destination Gateway
-----
eth0 10.1.0.2 10.0.2.2
eth0 10.1.0.2 10.0.2.2
lo 10.0.2.15 10.0.2.15
lo 127.0.0.1 127.0.0.1
eth0 10.0.2.2 10.0.2.2
eth0 10.0.2.2 10.0.2.2
lo 127.0.0.1 127.0.0.1
lo 10.0.2.15 10.0.2.15
eth0 152.19.134.47 10.0.2.2
lo 10.0.2.15 10.0.2.15
eth0 152.19.134.47 10.0.2.2
```

Pour terminer sur les connexions réseaux, l'équivalent de la sortie de la commande **netstat** peut être obtenu en exécutant **linux_netstat**. Cette commande récupère la liste des fichiers ouverts pour chaque processus en utilisant les résultats de la commande **linux_lsof**. Puis, pour chaque fichier, la commande va tester si le membre **f_op** pointe sur une structure **socket_file_ops** ou si le membre **dentry.d_op** pointe sur une structure **sockfs_dentry_operations**. Si tel est le cas, le membre **dentry.d_inode** est traduit en une structure **socket** permettant d'accéder au membre **sk** représentant la structure **inet_sock**. Cette dernière permet d'accéder aux informations telles que le protocole : TCP ou UDP, dans le cas de socket de type IPv4 ou IPv6 : les adresses, ports sources et destinations ainsi que l'état de la connexion (fermée, établie, en écoute). Dans le cas d'une socket locale, le chemin vers celle-ci sera affiché. Finalement, le nom et le PID du processus sont affichés.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran.lime linux_netstat
Volatility Foundation Volatility Framework 2.3.1
UNIX /run/udev/control
[...]
UDP 0.0.0.0:68 0.0.0.0:0 dhclient/1734
UDP 0.0.0.0:46020 0.0.0.0:0 dhclient/1734
UDP :::61435 :::0 dhclient/1734
UNIX /dev/log
UNIX /var/run/acpid.socket
TCP 127.0.0.1:25 0.0.0.0:0 LISTEN exim4/2334
TCP :::1:25 :::0 LISTEN exim4/2334
TCP 0.0.0.0:80 0.0.0.0:0 LISTEN nginx/2359
[...]
TCP 0.0.0.0:22 0.0.0.0:0 LISTEN sshd/2377
TCP :::22 :::0 LISTEN sshd/2377
[...]
TCP 10.0.2.15:22 10.0.2.2:33518 ESTABLISHED sshd/2586
TCP 10.0.2.15:22 10.0.2.2:33518 ESTABLISHED sshd/2588
TCP 10.0.2.15:54675 152.19.134.47:80 ESTABLISHED wget/2705
```

6.2 Contenu des paquets

À partir d'une acquisition mémoire, il est possible d'obtenir dans certains cas le contenu des paquets réseaux.



La première technique repose sur la mise en *queue* des paquets, ce qui arrive dans deux cas de figure :

- lorsque le réseau n'est pas dimensionné pour tenir la cadence d'envoi d'une socket ;
- lorsque les paquets destinés à un processus utilisateur n'ont pas été traités.

La commande **linux_pkt_queues** extrait les paquets de cette queue avec la nomenclature suivante : **[receive|send].pid.fd**. Il est donc nécessaire de croiser les informations obtenues précédemment avec **pslist** et **lsof**.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_pkt_queues -D /tmp/
Volatility Foundation Volatility Framework 2.3.1
Wrote 1460 bytes to write.2362.11
```

La dernière commande relative à l'extraction des paquets réseaux, **linux_sk_buff_cache**, utilise le cache du noyau Linux, le SLAB, pour récupérer les structures **sk_buff** dans les caches **skbuff_head_cache** et **skbuff_fclone_cache**. Le système de cache du noyau sera décrit plus tard dans l'article. Les tampons utilisés par le noyau pour gérer les paquets réseaux sont représentés par cette structure **sk_buff**. Elle contient une multitude d'informations comme les périphériques utilisés, les informations des couches 1 à 3 du modèle OSI (membres **h**, **nh** et **mac**). Dans le cas présent, le membre **len** est utilisé pour connaître la taille du paquet contenant les données brutes pointées par le membre **data**.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_sk_buff_cache -D /tmp/
Volatility Foundation Volatility Framework 2.3.1
Wrote 28 bytes to ffff88000c607180
Wrote 20 bytes to ffff88000c607280
[...]
Wrote 1420 bytes to ffff88000f10e180
Wrote 3594 bytes to ffff88000f10e580
Wrote 1420 bytes to ffff88000f10e980
Wrote 28 bytes to ffff88000c5e50c0
[...]
Wrote 56 bytes to ffff88000cdf3dc0
Wrote 28 bytes to ffff88000cde11c0
```

Sachez que la structure **sk_buff** n'est pas pleinement exploitée par Volatility et qu'il pourrait être intéressant de faire un module tirant parti des différentes informations contenues dans celle-ci, notamment pour croiser les données...

7 Stratégie d'allocation mémoire

Afin d'optimiser les allocations et libérations de structure noyau (processus avec **task_struct** ou paquet réseau avec **sk_buff**), le noyau fournit un mécanisme de cache fourni par l'intermédiaire du mécanisme SLAB.

Cela consiste à allouer au démarrage du noyau un *pool* de structures définies. Au moment d'une demande de création de structure donnée, le noyau a uniquement besoin de retourner le pointeur vers une entrée libre du *pool*, évitant ainsi le coût d'une allocation et la fragmentation de la mémoire physique.

Ces fonctionnalités sont proposées par le noyau au travers d'une API permettant d'accéder à des fonctions génériques d'allocation mémoire (**kmalloc**, **kmalloc_node**, ...) ou à des caches spécifiques (**kmem_cache_alloc**, **kmem_cache_alloc_node**). Derrière ce mécanisme, il existe trois stratégies d'allocation différentes : le SLAB, le SLUB et le SLOB. Le SLAB est la première version développée et utilisée par le noyau, le *SLUB* étant sa version améliorée avec une gestion différente des allocations comme l'utilisation d'une liste au lieu d'un tableau et une meilleure gouvernance du cache. Le *SLOB* est principalement utilisé pour les systèmes embarqués disposant de peu de mémoire.

Il est possible d'obtenir le statut actuel du cache de l'alloueur SLAB sous GNU/Linux en exécutant la commande **slabtop**. C'est une commande équivalente à **top** en termes d'affichage et qui permet de connaître la taille de chaque cache et son taux d'occupation. L'équivalent sous Volatility est possible en exécutant la commande **linux_slabinfo** comme on peut le constater dans le listing suivant :

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran.lime linux_slabinfo
Volatility Foundation Volatility Framework 2.3.1
<name>          <active_objs> <num_objs> <objsize> <objperslab> <pagesperslab> <active_slabs> <num_slabs>
-----
[...]
arp_cache        2          8      448      8          1          1          1
PING             0          0      832      9          2          0          0
RAW              2          9      832      9          2          1          1
UDP              6          9      832      9          2          1          1
TCP             12         12     1728      4          2          3          3
[...]
skbuff_fclone_cache 21         21      512      7          1          3          3
skbuff_head_cache 2577573693 315      256     15          1         21         21
file_lock_cache   11         20      192     20          1          1          1
[...]
sysfs_dir_cache   8262       8262     144     27          1         306        306
mnt_cache         25         30      256     15          1          2          2
filp              570        570      256     15          1         38         38
inode_cache       1981       1981     552      7          1        283        283
dentry            4460       4460     192     20          1       223        223
names_cache       2           2     4096      1          1          2          2
[...]
vm_area_struct    2662       2662     176     22          1        121        121
mm_struct         48         48      960      4          1         12         12
fs_cache          37         59       64     59          1          1          1
files_cache       63         77      704     11          2          7          7
signal_cache      82         98     1088      7          2         14         14
sigand_cache      75         75     2112      3          2         25         25
task_xstate       73         80      512      8          1         10         10
task_struct       92         92     1792      2          1         46         46
[...]
```

La majorité des commandes ayant comme suffixe **_cache** utilise les informations du cache grâce aux fonctions définies dans **slab_info**. Il est également nécessaire de préciser que Volatility ne gère pas l'alloueur d'objets de type SLUB.

8 Montrez ce que je ne saurais voir

Les sections précédentes ont présenté les commandes utiles pour travailler sur la partie visible d'un système. Cette section est consacrée aux commandes permettant de mettre en évidence ce qu'un attaquant souhaite dissimuler. De la même manière que les sections précédentes, l'envers du décor des commandes est présenté afin de comprendre les mécanismes de détection utilisés.

8.1 Historique des commandes exécutées

Lorsqu'une machine compromise est analysée, une des premières sources d'information est la liste des commandes exécutées par l'attaquant. Cependant, ce dernier prend généralement soin d'effacer le fichier **.bash_history** à la racine du compte utilisateur. Même si ce fichier a été effacé de manière sécurisée, tout n'est pas perdu. Il est en effet possible de récupérer l'historique des différentes commandes exécutées en analysant la mémoire allouée dynamiquement par le binaire **bash**. Ce dernier maintient une liste des commandes exécutées qu'il est possible de récupérer par l'utilisation du module **linux_bash**.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran3.
lime linux_bash
Volatility Foundation Volatility Framework 2.3.1
```

Pid	Name	Command Time	Command
...	2589 bash	2014-08-04 15:27:25 UTC+0000	sudo nano /etc/nginx/sites-available/
2589 bash	2014-08-04 15:27:25 UTC+0000	sudo /etc/init.d/nginx restart	
2589 bash	2014-08-04 15:27:25 UTC+0000	./sock 1 3505	
2589 bash	2014-08-04 15:27:25 UTC+0000	sudo insmod suterusu.ko	
2589 bash	2014-08-04 15:27:25 UTC+0000	sudo pgrep -l nc	
2589 bash	2014-08-04 15:27:25 UTC+0000	./sock 5 22	
2589 bash	2014-08-04 15:27:25 UTC+0000	sudo wget http://www.gutenberg.org/ebooks/41622.txt.utf-8	
...	2589 bash	2014-08-04 15:27:25 UTC+0000	netstat -ltpn
2589 bash	2014-08-04 15:27:25 UTC+0000	sudo pgrep -l nc	
2589 bash	2014-08-04 15:27:25 UTC+0000	sudo netstat -ltpn	
2589 bash	2014-08-04 15:27:25 UTC+0000	make linux-x86_64	
KDIR=/lib/modules/\$(uname -r)/build	2589 bash	2014-08-04 15:27:25 UTC+0000	nc -l -p 4242&
2589 bash	2014-08-04 15:27:25 UTC+0000	cd /tmp/	
2589 bash	2014-08-04 15:27:25 UTC+0000	less sock.c	
...			

8.2 Hooking

Afin de rester sous le radar, la plupart des codes malveillants utilisent des techniques de dissimulation, dont celle du *hooking*. Elle consiste à intercepter les

appels aux fonctions permettant par exemple de lister des fichiers, des processus, des connexions réseau, etc. L'objectif du code malveillant est de faire en sorte que ses fonctions soient appelées au lieu de celles du système. De manière simpliste, cela consiste à modifier l'adresse de la fonction interceptée en la remplaçant par l'adresse de la fonction de *hook*. Lors d'un appel à la fonction en question, celle du code malveillant sera donc exécutée. Si le résultat attendu de cet appel ne contient pas d'informations à dissimuler, la fonction originale est appelée sinon la fonction faisant office de *hook* retournera un résultat maquillé. Le principe de détection repose donc sur la validation des adresses des fonctions du système. Les sections suivantes présentent les *hooks* habituellement utilisés par du code malveillant et qui sont détectables par Volatility.

8.2.1 Modification des structures opérations

Une technique couramment utilisée est d'intercepter les appels aux fonctions contenues dans une structure ayant pour suffixe **_operations**. Par exemple, celle relative aux fichiers se nomme **file_operations** et permet de lister le contenu d'un répertoire ou de lire le contenu d'un fichier. Cette structure est accessible par exemple depuis le membre **f_op** d'une structure de type **file**. Pourquoi modifier précisément ces informations ? Tout simplement parce que les outils habituels tels que **ps** ou **top** pour les processus ou **netstat** pour les connexions réseaux reposent sur l'utilisation du système de fichiers virtuel *procfs*.

Ce système de fichiers virtuel fournit les différentes informations relatives aux processus et au système de manière plus générale. Les répertoires nommés avec un nombre correspondent aux différents processus, le nombre représentant leur PID. Chaque répertoire contient les informations relatives au processus telles que ses arguments, ses fichiers mappés ou ses descripteurs de fichiers. Un code malveillant souhaitant dissimuler un processus à un utilisateur va donc *hooker* les routines de lecture du système de fichiers virtuel *procfs*.

La commande **linux_check_fop** permet de détecter aisément ce type d'usurpation en vérifiant que les pointeurs sur fonction pointent sur les adresses des symboles usuels définis dans **System.map**. La commande exécute cette vérification pour tous les fichiers ouverts fournis par la commande **linux_lsof**, sur les opérations associées à la structure **proc_root** accessible par l'obtention de l'adresse du symbole éponyme. Enfin le point de montage de *procfs* est également vérifié en récupérant le symbole **proc_mnt** représenté par la structure **vfsmount** (**linux/mount.h**).

Cette technique est également applicable pour dissimuler des connexions réseaux à une commande comme **netstat**. Ainsi, à l'instar de **linux_check_fop**, la commande **linux_check_afinfo** permet de rechercher ces modifications de pointeurs sur fonctions pour toutes les structures comportant des opérations d'entrées/sorties utilisées par les protocoles TCP et UDP. En plus



de **files_operations**, cette commande effectue aussi la vérification sur la structure **sequence_operations** pour la fonction **show** par exemple.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_check_afinfo
Volatility Foundation Volatility Framework 2.3.1
```

Symbol Name	Member	Address
tcp6_seq_afinfo	show	0xfffffffffa036d1ee
tcp4_seq_afinfo	show	0xfffffffffa036d293
udp6_seq_afinfo	show	0xfffffffffa036d0a4
udp4_seq_afinfo	show	0xfffffffffa036d149

8.2.2 Syscall

Une autre technique consiste à modifier la table des appels systèmes. Cette table contient les fonctions appelées par un processus demandant un service au noyau. Ainsi lorsqu'un programme ouvre un fichier et lit dans celui-ci, les appels systèmes **open()** et **read()** sont utilisés. Un rootkit comme Kbeast va ainsi durant son initialisation poser différents hooks d'appels système en remplaçant le pointeur de la fonction du système par la sienne. Afin de détecter ce comportement, la commande **linux_check_syscall** vérifie que chaque adresse contenue dans la table des appels système correspond à une adresse connue dans les symboles du profile. Si tel est le cas, le nom du symbole est affiché et **HOOKED** dans le cas contraire.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_check_syscall
Volatility Foundation Volatility Framework 2.3.1
```

Table Name	Index	Address	Symbol
64bit	0x0	0xfffffffffa02fc7cc	HOOKED
64bit	0x1	0xfffffffffa02fc8ae	HOOKED
64bit	0x2	0xfffffffffa0fa30c	sys_open
64bit	0x3	0xfffffffffa0f9788	sys_close
64bit	0x4	0xfffffffffa0fe3ca	sys_newstat
64bit	0x5	0xfffffffffa0fe44b	sys_newfstat
64bit	0x6	0xfffffffffa0fe3f5	sys_newlstat
64bit	0x7	0xfffffffffa0f9787	sys_poll
64bit	0x8	0xfffffffffa0fa860	sys_lseek

[...]

8.2.3 IDT

Une autre technique de *hooking* consiste à modifier l'*Interrupt Descriptor Table*. Cette table linéaire de 256 entrées contient un pointeur vers le gestionnaire d'interruptions. Brièvement, une interruption est un événement nécessitant une attention soudaine et qui va altérer la séquence d'instruction qui était en cours d'exécution par le processeur. Les sources d'une interruption peuvent provenir du matériel (Entrées/Sorties) ou peuvent être logicielles (par exemple avec l'instruction **int3** utilisée par les debuggers pour poser un *breakpoint*). Le lecteur intéressé par les techniques de *hooking* IDT peut se reporter à l'article *Handling Interrupt Descriptor Table for fun and profit* paru dans Phrack [PHRACK].

La commande permettant de vérifier l'intégrité de l'IDT se nomme **linux_check_idt** et fonctionne sur le même principe de comparaison d'adresse que la commande **linux_check_syscall**.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_check_idt
Volatility Foundation Volatility Framework 2.3.1
```

Index	Address	Symbol
0x0	0xfffffffffa0356ab0	divide_error
0x1	0xfffffffffa0356fb0	debug
0x2	0xfffffffffa0356390	nmi
0x3	0xfffffffffa0356ffc0	int3
0x4	0xfffffffffa0356ad0	overflow
0x5	0xfffffffffa0356af0	bounds
0x6	0xfffffffffa0356b10	invalid_op
0x7	0xfffffffffa0356b30	device_not_available
0x8	0xfffffffffa0356b50	double_fault
0x9	0xfffffffffa0356b80	coprocessor_segment_overrun
0xa	0xfffffffffa0356ba0	invalid_TSS
0xb	0xfffffffffa0356bd0	segment_not_present
0xc	0xfffffffffa0356000	stack_segment
0xd	0xfffffffffa03560a0	general_protection
0xe	0xfffffffffa03560b0	page_fault
0xf	0xfffffffffa0356c00	spurious_interrupt_bug
0x10	0xfffffffffa0356c20	coprocessor_error
0x11	0xfffffffffa0356c40	alignment_check
0x12	0xfffffffffa0356130	machine_check
0x13	0xfffffffffa0356c70	simd_coprocessor_error
0x80	0xfffffffffa03573f0	ia32_syscall

8.3 Le jeu des différences

Une autre technique utilisée pour la détection d'anomalie repose sur la corrélation d'informations entre différentes sources. Selon les informations concernées, différents types de structures auront un accès à celles-ci. Si le code malveillant ne modifie pas tous ces pointeurs, il devient possible de détecter ce qui est dissimulé par ce biais. De plus, l'objectif d'un code malveillant, comme tout processus, est de se voir allouer du temps processeur. Même si son objectif est de rester le plus furtif possible, il aura toujours besoin d'être accessible d'une manière ou d'une autre.

8.3.1 Processus

Il est possible de dissimuler un processus en modifiant les pointeurs **next** et **prev** des **task_struct** précédentes et suivantes respectivement. Bien que le processus ne soit plus accessible depuis la liste doublement chaînée des **task_struct**, cela ne l'empêche pas d'obtenir du temps CPU pour continuer son exécution. En effet, afin de gérer l'exécution des processus, le noyau utilise un gestionnaire de tâches ou *scheduler*. Sa structure essentielle se nomme **runqueue**. Sur une architecture multiprocesseur, chaque CPU se voit attribuer une structure **runqueue**. Le champ le plus intéressant de cette structure est **arrays** qui est un tableau contenant la liste des processus actifs ou endormis. Ces deux éléments du tableau représentent une structure de type **prio_array_t** qui inclut 140 listes



doublément chaînées représentant toutes les priorités existantes pouvant être attribuées à un processus. Le *scheduler* travaille directement à partir de ces listes pour gérer l'exécution des processus. Il faut savoir qu'au moment de la rédaction de l'article, il n'existait pas encore de module Volatility capable d'extraire les informations relatives aux **runqueues**.

Cependant d'autres structures du noyau permettent actuellement de mettre en avant des comportements potentiellement malicieux.

Comme vu précédemment, le noyau maintient un cache des objets alloués. À partir de celui-ci, il est donc possible d'obtenir des informations relatives aux allocations de la structure **task_struct** définissant un processus. C'est précisément la fonctionnalité fournie par la commande **linux_pslst_cache** qui va lister toutes les **task_struct** présentes dans le cache du mécanisme d'allocation SLAB. Par défaut, la commande va uniquement travailler sur les structures allouées, mais il est possible d'obtenir également une liste des éléments non alloués en fournissant l'argument **-u** à la commande. Il est ainsi possible de récupérer des informations relatives aux processus ayant pu être alloués précédemment.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran2.lime linux_pslst_cache
Volatility Foundation Volatility Framework 2.3.1
Offset(V)      Name      Pid      Uid      Gid      DTB      Start Time
-----
0xffff8000f0d7040 rs:main Q:Reg 1953      0         0         0x00000000c6db000 2014-08-04 15:21:51 UTC+0000
0xffff8000f0d7740 sudo      2723      0      1000      0x00000000c7f0000 2014-08-04 15:29:29 UTC+0000
0xffff8000c5e2180 acpid      1992      0         0         0x00000000c6fd000 2014-08-04 15:21:52 UTC+0000
0xffff8000c5e2800 bash       2722      1000      1000      0x00000000f318000 2014-08-04 15:29:29 UTC+0000
0xffff8000c62b0c0 rpc.statd  1593      102      65534     0x00000000fadf000 2014-08-04 15:21:51 UTC+0000
0xffff8000c62b7c0 insmod     2724      0         0         0x00000000c7e1000 2014-08-04 15:29:29 UTC+0000
0xffff8000c6d0140 wget       2725      1000      1000      0x00000000c7c9000 2014-08-04 15:29:31 UTC+0000
[...]
0xffff8000f0ab1c0 nc         2570      1000      1000      0x00000000db69000 2014-08-04 15:23:47 UTC+0000
[...]
0xffff8000c59a100 nginx      2362      33        33        0x00000000e0ef000 2014-08-04 15:21:52 UTC+0000
0xffff8000c59a800 rsyslogd  1955      0         0         0x00000000c6db000 2014-08-04 15:21:51 UTC+0000
0xffff8000f089180 vminfo    2407      0         0         0x00000000c847000 2014-08-04 15:21:53 UTC+0000
0xffff8000f089880 cron      2061      0         0         0x00000000e08d000 2014-08-04 15:21:52 UTC+0000
0xffff8000d62d100 dhclient  1734      0         0         0x00000000d2eb000 2014-08-04 15:21:51 UTC+0000
0xffff8000d62d800 getty     2419      0         0         0x00000000e003000 2014-08-04 15:21:53 UTC+0000
0xffff8000fae0180 nginx      2364      33        33        0x00000000e296000 2014-08-04 15:21:52 UTC+0000
```

Une autre technique possible repose sur l'analyse des tables de hash de PID obtenues au travers du symbole **pid_hash**. Il existe quatre tables de hash différentes dans le noyau Linux permettant de récupérer un descripteur de processus à partir de son PID. Un de ces mécanismes de traduction est par exemple utilisé lors de l'appel système **kill**. La commande **linux_pidhashtable** est en charge d'analyser ces informations.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran2.lime linux_pidhashtable
Volatility Foundation Volatility Framework 2.3.1
Offset      Name      Pid      Uid      Gid      DTB      Start Time
-----
0xffff8000f1dc140 nginx      2365      33        33        0x00000000c485000 2014-08-04 15:21:52 UTC+0000
0xffff8000d62d100 dhclient  1734      0         0         0x00000000d2eb000 2014-08-04 15:21:51 UTC+0000
[...]
0xffff8000f0ab1c0 nc         2570      1000      1000      0x00000000db69000 2014-08-04 15:23:47 UTC+0000
0xffff8000f222740 bash       2426      1000      1000      0x00000000c802000 2014-08-04 15:22:14 UTC+0000
0xffff8000c5b3840 rpcbind    1562      0         0         0x00000000f1ae000 2014-08-04 15:21:50 UTC+0000
[...]
```

Bien qu'il soit possible d'utiliser les deux commandes précédentes de manière séparée, il est plutôt recommandé de recourir à la commande **linux_psxview** qui croise les informations de **linux_pslst**, **linux_pslst_cache** et **linux_pidhashtable** et qui affiche le nom de chaque processus et leur présence dans chacune des trois informations obtenues.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran2.lime linux_psxview
Volatility Foundation Volatility Framework 2.3.1
Offset(V)      Name      PID pslst pid_hash kmem_cache
-----
0xffff8000f089e740 init          1 True   True   True
0xffff8000f089e040 kthreadd      2 True   True   True
[...]
0xffff8000c5b3140 sshd         2425 True   True   True
0xffff8000f222740 bash         2426 True   True   True
0xffff8000d975800 kthread      2568 True   True   True
0xffff8000c5b0780 dlexec       2569 True   True   True
0xffff8000d64d840 sshd         2586 True   True   True
0xffff8000c58d740 sshd         2588 True   True   True
0xffff8000c7f37c0 bash         2589 True   True   True
0xffff8000c5e2800 bash         2722 True   True   True
0xffff8000f0d7740 sudo        2723 True   True   True
0xffff8000c62b7c0 insmod       2724 True   True   True
0xffff8000c6d0140 wget         2725 True   True   True
[...]
0xffff8000f0ab1c0 nc           2570 False  True   True
[...]
```

8.3.2 Modules noyau

De la même manière qu'un processus peut être caché en étant retiré de la liste **task_struct**, il est possible d'en faire autant avec la liste des modules noyaux chargés telle que présentée par la commande habituellement utilisée : **lsmod**. À titre informatif, le fichier **/proc/modules** est basé sur l'export de cette liste chaînée en question et c'est sur ce fichier que repose l'implémentation de la commande habituelle **lsmod**. Ce que fait Volatility pour détecter cette dissimulation est de procéder à une comparaison entre cette liste chaînée et le contenu du répertoire **/sys/module** qui représente chaque module par un répertoire portant son nom. Ce dernier contient à son tour les informations du module en question. Cette représentation « système de fichiers » repose en fait sur l'utilisation d'une nouvelle structure. Pour y accéder, il faut que le noyau ait été compilé avec le support de **sysfs** ce qui permet d'obtenir le symbole **module_kset** de type **kset** contenant un champ **kobj**. Ce champ donne ensuite accès à la liste chaînée de type **kobject**. La commande **linux_check_modules** fait par la suite une différence entre ces deux listes chaînées pour mettre en avant les modules suspects.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f aldebaran2.lime linux_check_modules
Volatility Foundation Volatility Framework 2.3.1
Module Name
-----
suterusu
```

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 Infoforensique
Windows

FOR 508

Analyse Infoforensique 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





8.4 Keylogging

Comme mentionné sur le wiki du projet, Volatility permet de mettre en avant deux techniques utilisées pour *keylogger* et présentées dans le papier [KEYLOGGERS].

8.4.1 TTY

La première technique repose sur la pose d'un contournement de la fonction **receive_buf** du pilote *tty*. La commande Volatility en charge de détecter ce *hook* se nomme **linux_check_tty**. Elle va dans un premier temps récupérer et itérer sur la liste des pilotes *tty* obtenue par le symbole **tty_drivers**. À chaque pilote est associé un tableau contenant les périphériques *tty* et qui sont représentés par la structure **tty_struct**. Pour chacune d'entre elles, la commande accède ensuite au champ relatif à la structure définissant les opérations propres aux *tty line discipline* présentée dans le papier précédemment cité : **ldisc.ops**. Cet accès permet finalement de pointer sur la fonction **receive_buf**. L'adresse de cette fonction obtenue, il suffit de vérifier si celle-ci est associée à un symbole du noyau (**get_symbol_by_address("kernel", recv_buf)**). Si tel n'est pas le cas, la commande affichera *HOOKED* dans la colonne **Symbol**.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_check_tty
Volatility Foundation Volatility Framework 2.3.1
Name Address Symbol
-----
tty1 0xffffffff8122b296 n_tty_receive_buf
tty2 0xffffffff8122b296 n_tty_receive_buf
tty3 0xffffffff8122b296 n_tty_receive_buf
tty4 0xffffffff8122b296 n_tty_receive_buf
tty5 0xffffffff8122b296 n_tty_receive_buf
tty6 0xffffffff8122b296 n_tty_receive_buf
```

4.8.2 Keyboard notifier chains

Le côté intéressant de cette technique réside dans le fait qu'aucune modification de structure en mémoire n'est faite, mais qu'elle repose sur l'abus d'une API du noyau Linux existante depuis la version 1.1 : les *notifiers chains*. De manière simple, cette API permet d'associer à un événement système une liste chaînée de fonctions de rappel (callbacks). Lorsque l'événement en question se produit, la liste chaînée est parcourue et chaque fonction appelée. D'un point de vue programmation, ce mécanisme est équivalent au *design pattern Observer*. Vous l'aurez deviné, pour être informé d'un événement, il suffit d'ajouter sa *callback* à cette liste chaînée. Et depuis la version 2.6.24, il existe une chaîne de notification relative aux frappes clavier.

De la même manière que la commande **linux_check_tty**, la commande **linux_keyboard_notifier** va donc vérifier si les adresses de fonction de rappel enregistrées dans

la liste chaînée **keyboard_notifier_list** correspondent à l'adresse d'un symbole du noyau ou non.

```
$ python vol.py --profile LinuxDebianWheezy7_5-3_2_0-4-amd64x64 -f
aldebaran.lime linux_keyboard_notifier
Volatility Foundation Volatility Framework 2.3.1
Address Symbol
-----
0xfffffffffa035f664 HOOKED
```

Conclusion

Cet article a présenté la majorité des différentes commandes utilisables par Volatility sur un système GNU/Linux. Résumer toutes les informations utilisées en un article n'est pas chose aisée notamment quand des livres sont dédiés à un seul sous-système du noyau. J'en profite pour attirer l'attention sur la sortie en juillet dernier d'une bible de 900 pages rédigée par l'équipe de développement de Volatility intitulée *The Art of Memory Forensics* traitant à la fois les systèmes Windows, Linux et Mac. Enfin, certaines informations ne sont pas encore utilisées et des pans entiers de structures n'ont pas été présentés, donnant la possibilité de développer de nouveaux modules à qui le veut. ■

■ Remerciements

Merci à Cédric et Jean-François pour leur relecture, Samir pour ses remarques et conseils pertinents ainsi qu'à Guillaume pour son extrême patience...

■ Bibliographie

[DFRWS14] <http://www.dfrws.org/2014eu/program.shtml>

[LMAP] <https://code.google.com/p/rekall/source/browse/#git%2Ftools%2Flinux%2Fimap>

[VOLINUX] <http://code.google.com/p/volatility/wiki/LinuxCommandReference23>

[VOLUSAGE] <http://code.google.com/p/volatility/wiki/VolatilityUsage23>

[VBOX] <http://www.digital-forensic.org/media/vms/misc.zip>

[SUTERUSU] <https://github.com/mncoppola/suterusu>

[ROUTE CACHE] <https://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/commit/?id=89aef8921bfbac22f00e04f8450f6e447db13e42>

[PHRACK] <http://www.phrack.org/issues/59/4.html#article>

[KEYLOGGERS] <http://www.ieee-security.org/TC/SPW2012/proceedings/4740a097.pdf>

DÉCOUVREZ NOS NOUVELLES OFFRES D'ABONNEMENTS !

PRO OU PARTICULIER = CONNECTEZ-VOUS SUR :

boutique.ed-diamond.com



1

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.



EXEMPLE

6^{N°} MISC (PAPIER + PDF 1 LECTEUR)
+ ACCÈS 1 AN BASE DOCUMENTAIRE MISC (1 CONNEXION)

TARIF NORMAL
165,80 €

ÉCONOMIE
54,80 €

OFFRE SPÉCIALE
111,-€

2

LES COUPLAGES AVEC NOS AUTRES MAGAZINES :

EXEMPLE

6^{N°} MISC (PAPIER)
+ 11^{N°} GNU/LINUX MAGAZINE (PAPIER)

TARIF NORMAL
140,30 €

ÉCONOMIE
40,30 €

OFFRE SPÉCIALE
100,-€

EXEMPLE

6^{N°} LINUX PRATIQUE (PAPIER)
+ 6^{N°} MISC (PAPIER) + 11^{N°} GNU/LINUX MAGAZINE (PAPIER)
+ 6^{N°} HACKABLE (PAPIER) + 6^{N°} OPEN SILICIUM (PAPIER)

TARIF NORMAL
271,12 €

ÉCONOMIE
71,12 €

OFFRE SPÉCIALE
200,-€



IN THE CLOUD : DO ELECTRIC SHEEP JUMP OVER A VIRTUAL FENCE ?

Depuis le début du développement du marché du Cloud Français, je me dis que les paradigmes technologiques ça s'en va et ça revient, c'est fait de tout petits riens. En effet, le Cloud c'est juste une architecture client/serveur un peu évoluée, ça existe depuis les années cinquante [1], mais à l'époque ça s'appelait le *time-sharing*. Souvenez-vous, chacun se connectait avec un terminal à bas coût à une machine surpuissante partagée entre tous les utilisateurs [2]. Les principales différences étant qu'aujourd'hui :

- le client tient dans la poche et est plus puissant que le serveur de l'époque,
- la connexion n'est plus limitée à un réseau d'entreprise, mais peut se faire de (presque) n'importe où dans le monde,
- la virtualisation nous donne l'impression de ne pas partager notre serveur avec tout un tas d'autres utilisateurs.

Ces avancées permettent de nouveaux usages qui ont fait émerger un terme beau comme un clair de lune à Maubeuge : le Cloud. Brasseurs de vent professionnels, les commerciaux en ont même réussi à nous faire oublier l'existence des serveurs. Madame Michu s'imaginer [3] que ses données sont partout et nulle part et que personne ne sait où elles sont, et d'une certaine façon, elle a raison. Google a d'ailleurs trouvé une solution élégante au problème, posé par les différentes réglementations sur le respect de la vie privée, de recherche pour suppression de données périmées [4] : les données sont chiffrées avec une même clé, seule la clé est supprimée à la date imposée, les données chiffrées peuvent survivre jusqu'à ce qu'on ait besoin de la place qu'elles occupent.

Ces nouveaux usages, et l'ignorance des utilisateurs, posent de nouveaux problèmes de sécurité, et quand le Cloud se rate, on peut se retrouver dans une situation pas très belle (belle, belle [5]). Les récentes affaires de photos volées nous donnent l'impression qu'elles deviennent (les attaques dans le Cloud) monnaie courante.

Ces quelques articles vous permettront, je l'espère, de mieux appréhender le Cloud, d'un point de vue technique comme juridique. ■

Colas

[1] À peu près au moment où on a arrêté de s'intéresser aux films 3D.

[2] Personnellement, je ne m'en souviens pas, je ne suis pas si vieux...

[3] Véridique

[4] citation needed

[5] Attention, il y a un piège

AU SOMMAIRE DE CE DOSSIER :

- [29-34] Appréhender le cloud, pour le meilleur et pour le pire
- [36-40] Introduction à la sécurité de la gestion dynamique des ressources dans le cloud
- [42-45] Mécanismes de sécurité au niveau processeur pour la virtualisation
- [46-50] Le Cloud : cette nébuleuse juridique

APPRÉHENDER LE CLOUD, POUR LE MEILLEUR ET POUR LE PIRE

Matthieu Bouthors – matthieu.bouthors@outscale.com

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



mots-clés : CLOUD COMPUTING / IAAS / PAAS / SAAS

Cet article propose une introduction au fonctionnement des différentes solutions de « Cloud Computing ». Comme toute nouvelle technologie, nous verrons que ces nouvelles solutions arrivent de concert avec de nouvelles problématiques liées à la sécurité.

1 La tête dans les nuages

Ce n'est pas parce que le « Cloud Computing » est le dernier terme à la mode, que ce soit au niveau des DSI ou des éditeurs souhaitant fournir une version reliftée de leur produits que cette nouvelle manière de consommer de la ressource informatique est amenée à disparaître aussi vite qu'elle a pu apparaître. Malgré l'affaire Snowden, nous continuons d'utiliser de plus en plus de ces services adaptés à nos nouveaux modes de vie et à l'émergence de l'Internet des objets.

Afin de démystifier ce « Cloud Computing » qui fait désormais partie de notre quotidien, commençons par voir de quoi il se compose et ce qui se cache derrière ces différentes abréviations nébuleuses. Afin d'avancer dans cette direction, nous pouvons déterminer assez rapidement un certain nombre de caractéristiques propres aux différentes solutions estampillées « Cloud Computing ».

1.1 La forme des nuages

1.1.1 Accessible via Internet

Dans nos sociétés modernes hyper-connectées, l'accès Internet est possible quasiment à tout moment et en tout lieu, il est donc plus efficace et pratique d'accéder à ses applications/données/infrastructures par ce biais. Les différents mécanismes « classiques » d'isolation réseau sont alors difficiles à adapter et il devient désuet

de devoir s'astreindre à être physiquement au bureau pour accéder aux applications d'entreprise ou chez soi pour accéder à ses données personnelles.

1.1.2 Adaptable et automatisable dans les moindres détails

Corollaire du premier point, l'accès universel par Internet, mais via des terminaux hétérogènes nécessite énormément d'agilité. Ce qui est généralement possible par la fourniture d'APIs. Dans la plupart des solutions, l'API est le premier élément et les applications mobiles (iOS/Android), portails web et clients lourds ne sont que des interfaces graphiques consommant ces APIs. Il est alors plus intéressant de se focaliser sur la sécurité des APIs que sur la sécurité des interfaces web plus classiques [1].

1.1.3 Modulable et élastique

De par l'absence de bien matériel possédé par l'utilisateur et la modularité, l'utilisateur tend à n'utiliser/commander uniquement ce dont il a effectivement besoin. Nous avons donc généralement à faire à des solutions divisées en différentes fonctionnalités indépendantes. L'utilisateur n'utilise donc que les fonctionnalités dont il a besoin dans les quantités qui lui conviennent. Les modèles économiques du freemium (contenu gratuit financé par la publicité, puis payant pour les utilisateurs faisant état de besoins plus conséquents) et de la facturation à la demande se sont donc imposés vis-à-vis d'offres packagées en mode forfait un peu plus classiques.



Cela signifie qu'en cas d'augmentation ou diminution de la charge (par exemple sur un site web), de nouvelles ressources peuvent être ajoutées ou enlevées de manière quasi immédiate. L'impact sur la facturation est lui aussi immédiat, ce qui peut être assez désagréable dans le cas par exemple d'un déni de service distribué (DDoS).

1.1.4 Mutualisé et simplifié

Afin de fournir l'ensemble de ces services au plus grand nombre tout en restant attractives, les infrastructures sont habituellement mutualisées. Cela permet aux fournisseurs de services d'optimiser au mieux et de lisser les effets des consommations élastiques et modulés des différents utilisateurs.

La mutualisation permet par ailleurs de centraliser et d'unifier la maintenance du côté du fournisseur. La solution se retrouve alors simplifiée pour l'utilisateur débarrassé de ces opérations de maintenance et de maintien de la solution et des services. De notre point de vue de l'utilisateur à la recherche du meilleur niveau de sécurité, cela implique un nécessaire besoin de confiance et garanties vis-à-vis du fournisseur. Dans les petites structures plus limitées au niveau des compétences nécessaires au maintien des infrastructures, ce peut être un progrès, pour les autres, la situation peut s'avérer plus délicate.

1.2 Les familles de nuages

Au-delà de ces caractéristiques principales, nous pouvons distinguer différents types de solutions de « Cloud Computing » répondant à des besoins différents tout en restant complémentaires.

1.2.1 Infrastructure as a Service (IaaS)

Le premier type de solution qui peut être à bien des égards considéré comme la brique de base de l'écosystème « Cloud Computing » est l'*Infrastructure as a Service* (IaaS). L'objectif de ces solutions est de fournir des infrastructures matérielles : calcul (CPU/RAM), stockage (blocs, objets) et réseau (connectivité, filtrage, répartition de charge). Nous sommes donc sur une évolution des métiers de l'hébergement apportant plus de souplesse et d'instantanéité. Comme nous le verrons plus tard au sein de cet article, ces évolutions rendent souvent nécessaire l'utilisation de certaines solutions techniques telles que la virtualisation. Les risques liés à ces solutions sont donc sensiblement les mêmes que ceux liés à l'utilisation de prestations d'hébergement externes. Cependant, l'automatisation et la virtualisation augmentent les risques liés à un éventuel défaut d'isolation entre les différents utilisateurs.

Ce type de solution s'adresse avant tout aux administrateurs systèmes et réseaux qui gagnent en souplesse et efficacité en utilisant ces solutions en lieu et place de solutions internes ou d'hébergeurs externes plus « conventionnels ». Les solutions les plus connues

restent Amazon avec son offre AWS et OpenStack qui entend fournir une alternative open source crédible.

1.2.2 Platform as a Service (PaaS)

Clef de voûte et pourtant assez méconnue, le *Platform as a Service* consiste à ne plus fournir de l'infrastructure sous forme de ressources brutes, mais un middleware directement à même d'accueillir différentes applications. Le service est donc rendu principalement sous la forme de serveurs d'applications (PHP, Java, Python, Ruby...) et de moteurs de stockage et traitement des données (SQL, NoSQL, Message Queueing...). L'abstraction vis-à-vis du matériel est bien plus forte et ces solutions peuvent être comparées à des solutions d'hébergement de sites Internet mutualisés qui ont eu leur heure de gloire aux débuts des années 2000. Leur version PaaS est cependant gavée de fonctionnalités et place la barre quelques crans plus haut au niveau des performances. D'un point de vue risque, les solutions étant plus éloignées du système et du matériel, il est parfois plus compliqué de comprendre comment y protéger au mieux ses données.

Ce type de solution s'adresse plutôt aux développeurs qui y trouvent un outil des plus adaptés, car débarrassé des contraintes matérielles, systèmes et lié au déploiement de leurs solutions. Les solutions les plus utilisées en PaaS restent Heroku et Google App Engine.

1.2.3 Software as a Service (SaaS)

Bien plus connu du grand public et de plus en plus présent dans le monde de l'entreprise, le *Software as a Service* met directement à disposition des applications conservant les différentes caractéristiques du « Cloud Computing ». L'application est généralement utilisable via des portails Internet et des applications mobiles (iOS/Android).

Ce type de solution est utilisé directement par les utilisateurs finaux que ce soit pour communiquer (Gmail, Yahoo...), éditer des documents et contenus (Office 365, Adobe Creative Cloud...) ou stocker des données (DropBox, iCloud...).

1.2.4 anything as a Service (XaaS)

La nature ayant pour notre plus grand bonheur choisi de bannir le vide et la simplicité, cette liste n'est bien entendu pas exhaustive. Nous pouvons par exemple citer le *Desktop as a Service* (DaaS) poussé par VMware et Citrix qui permet de remplacer les postes de travail par des terminaux légers accédant aux « bureaux » virtualisés dans le Cloud. Dans le même registre, nous avons aussi le *Communication as a Services* (CaaS) qui remplace la téléphonie d'entreprise par des solutions assez proches du SaaS.

De manière plus globale, l'ensemble des ressources informatiques se retrouve avec une version cloudifiée qui au-delà de l'effet de mode s'appuie sur des solutions de type IaaS, PaaS et SaaS.



2 Les usines à nuages

Après notre petite tour d'horizon des différentes caractéristiques et solutions de « Cloud Computing », nous allons maintenant pouvoir nous intéresser un peu plus en détail à un paradigme régulièrement mis en œuvre sur ces solutions : le « software defined » qui consiste à redéfinir de manière logicielle des éléments d'infrastructures habituellement fournis par du matériel spécialisé.

2.1 Naviguer dans les nuages : Software Defined Networking (SDN)

De par la nécessité de fournir des ressources réseaux de manière élastique tout en conservant un niveau acceptable de performance et de sécurité, les solutions « conventionnelles » de commutation et filtrage réseau éprouvent leurs limites. De manière similaire aux serveurs physiques qui sont remplacés par des machines virtuelles, les switches, routeurs et autres firewalls sont généralement remplacés par divers éléments regroupés sous l'appellation *Software Defined Networking* (SDN).

2.1.1 OpenVSwitch

Afin de s'abstraire des limites physiques habituelles telles que le nombre de VLANs lors de l'encapsulation de multiples réseaux logiques au sein d'une même interface physique, une solution logicielle est généralement préférée.

OpenVSwitch est un des logiciels communément utilisés afin de permettre une encapsulation logicielle bien plus pratique lorsqu'il est question d'adapter la solution de commutation à de nouvelles contraintes en terme de fonctionnalité ou d'élasticité. Entre différentes machines virtuelles présentes sur le même serveur physique, OpenVSwitch remplace avec aisance les bridges Linux alors que pour les communications inter serveurs physiques l'encapsulation logicielle permettra de traverser avec un minimum de contraintes des liens réseaux classiques.

Afin de commuter au mieux les divers paquets réseaux de manière logicielle, OpenVSwitch va utiliser la notion de flux (« flow » dans la langue de Shakespeare). Un flux sera défini comme un groupe de paquets partageant un ensemble de caractéristiques issues des quatre premiers niveaux de la couche OSI :

- interface réseau par laquelle le flux est entré sur l'équipement ;
- adresses IP sources et destinations ;
- protocole IP utilisé (TCP, UDP, ICMP, ESP...) ;
- pour TCP et UDP les ports source et destination ;
- valeur du champ IP ToS (généralement utilisé dans le cadre de la mise en place de QoS sur un réseau).

Sans apporter la finesse d'un pare-feu qui fera un suivi complet des états (statefull) au niveau 4 et parfois au niveau 7 pour certains protocoles (SIP, FTP...), cette utilisation des flux apporte de la finesse dans les règles de commutation et peut donc généralement permettre la mise en place de règles spécifiques plus efficaces que sur des commutateurs matériels classiques.

2.1.2 OpenFlow & NetFlow

OpenVSwitch ayant uniquement le rôle de commuter les paquets au sein de l'infrastructure physique, il nous est nécessaire d'avoir un contrôleur garant de la bonne orchestration de ces commutateurs logiciels. Le protocole OpenFlow permet ce lien entre l'application de contrôle et d'orchestration et le commutateur de paquets tel qu'OpenVSwitch. Au niveau des contrôleurs compatibles OpenFlow, nous pouvons citer Quantum, utilisé pour orchestrer les différents éléments réseaux d'OpenStack.

De notre point de vue sécurité, cette séparation des rôles entre le contrôleur et le commutateur apporte une centralisation. Ce qui peut certes éventuellement engendrer des impacts négatifs sur la disponibilité du service si elle est mal implémentée, mais d'un autre côté facilite énormément la gestion de la cohérence et la mise en place de contre-mesures rapides en cas d'incident (blocage d'une attaque, par exemple).

Cependant, ces systèmes d'orchestration ne sont pas encore des plus matures. L'année dernière, une étude sur la sécurité des différents déploiements OpenFlow [2] montrait que la plupart des canaux de contrôle n'utilisaient pas de chiffrement TLS (les rendant donc vulnérables à une attaque relativement triviale en cas de Man-In-The-Middle). Par ailleurs, la gestion des changements d'état au niveau des règles de commutation n'étant pas déterministe, dans certaines situations particulières des abus sont possibles et il n'est pas possible de garantir à chaque instant la cohérence globale de l'ensemble du système. Ces vulnérabilités sont pour le moins inquiétantes sur le papier, mais restent quasiment impossibles à exploiter dès lors que les réseaux d'administration sont strictement isolés des réseaux utilisateurs et que les accès physiques aux datacenters sont correctement contrôlés. Ces points sont malheureusement difficilement auditable pour la plupart des utilisateurs.

De manière complémentaire, les bonnes pratiques du *Software Defined Network* encouragent vivement à la collecte d'informations permettant d'avoir plus de visibilité sur l'ensemble des flux échangés. C'est à ce moment précis que NetFlow devient intéressant en permettant la collecte, par échantillonnage pour d'évidentes raisons de performance, des différentes informations sur les flux échangés au sein d'une infrastructure réseau matérielle compatible ou SDN déployée par exemple en utilisant OpenVSwitch.

Le protocole NetFlow ayant déjà fait l'objet d'un précédent article [3], nous n'allons pas nous attarder sur son fonctionnement. L'apport de NetFlow dans le cadre de plateformes « Cloud Computing » se situe au niveau de la détection des anomalies. La taille et le



niveau d'automatisation des plateformes font qu'il n'est que difficilement possible d'effectuer manuellement ces actions, il est donc important d'industrialiser l'analyse du réseau avec un outil tel que NetFlow. Les comportements malicieux ou sortant de l'ordinaire peuvent ainsi générer des alertes qui permettront de déclencher une analyse manuelle plus poussée et/ou un blocage des flux concernés, par exemple en modifiant les règles de commutation des flux via OpenFlow.

2.2 Entreposer dans les nuages : Software Defined Storage

Afin d'entreposer les différentes données nécessaires à ce que l'on peut communément appeler le « Big Data », les différentes solutions proposent plusieurs solutions de stockage des données. Les deux principales étant le stockage en mode bloc (un périphérique se comportant comme un disque physique est mis à disposition de la VM) et le stockage en mode objet/fichier (les fichiers sont ajoutables/modifiables/supprimables individuellement par le biais de requêtes HTTP). Le premier mode est omniprésent dans les solutions IaaS alors que le second est plutôt destiné au PaaS et au SaaS. Nous allons nous attarder un peu plus en détail sur le stockage en mode bloc.

2.2.1 Formats spécifiques

De manière à mutualiser les systèmes de stockage entre les différents utilisateurs tout en garantissant un minimum d'isolation des données, le stockage en mode bloc se trouve lui aussi virtualisé. Par commodité et dans un souci d'apporter un maximum de fonctionnalités à l'utilisateur, ce sont généralement les formats de fichiers des disques virtuels des hyperviseurs qui sont retenus. Ce qui nous donne VMDK dans le cas de VMWare et QCOW dans le cas de Qemu/KVM. À noter que VMWare étant un des acteurs historiques de la virtualisation, il est courant que des fournisseurs IaaS supportent l'importation dans le format VMDK sans l'utiliser par ailleurs dans leurs infrastructures. Cette solution pratique pour les utilisateurs peut par contre poser des problèmes de sécurité supplémentaires comme nous l'aborderons d'ici peu.

2.2.2 Fonctionnalités

Nos deux formats QCOW et VMDK possèdent plusieurs similarités dans la conception. Tout d'abord, ils permettent de ne stocker que les données réellement utiles sans réservation préalable de l'espace (*thin-provisionning*) avec éventuellement une compression transparente de ces données. Ensuite, ces deux formats supportent le concept de fichier parent qui permet à un fichier de ne pas contenir l'ensemble des blocs du disque virtuel, mais pour certains des pointeurs vers d'autres fichiers.

2.2.3 Snapshots et vulnérabilités

Cette dernière fonctionnalité est utile pour permettre par exemple de fournir des snapshots de disques aux utilisateurs de la solution IaaS. Lors d'un snapshot, le fichier utilisé pour stocker le disque dur virtuel n'est pas modifié et un nouveau fichier utilisant ce premier fichier comme fichier parent est créé. Après le snapshot, les écritures sont effectuées dans le nouveau fichier créé lors de l'opération de snapshot. L'ancien fichier, n'étant plus modifié, correspond à une copie instantanée du disque. Ceci peut s'avérer intéressant pour l'utilisateur et même dans des cas plus particuliers comme pour une analyse forensique d'une machine virtuelle corrompue.

Cependant les fichiers parents pouvant aussi être des fichiers raw sur lesquels aucune vérification particulière n'est effectuée, il est possible de forger un fichier QCOW ou VMDK valide ayant pour fichier parent n'importe quel fichier du système d'exploitation de l'hyperviseur.

Forger un tel fichier est relativement simple, par exemple pour lire le fichier « `/etc/passwd` » depuis un fichier QCOW, il suffit de créer un fichier avec l'utilitaire standard « `qemu-img` » et les options suivantes :

```
$ qemu-img create -f qcow2 -b /etc/passwd my-nasty-disk.qed
```

Lors de l'importation de ce fichier au sein de l'infrastructure de l'IaaS, si des tests poussés ne sont pas effectués, des données sensibles pourront alors être accessibles tout simplement en lisant le contenu du disque virtuel depuis la VM. Ce qui peut être dangereux pour les infrastructures du fournisseur, mais surtout pour les autres utilisateurs de l'IaaS.

3 Zones de turbulences

Que ce soit dans le cas d'un service IaaS, PaaS ou SaaS, les caractéristiques partagées par ces différentes solutions « Cloud Computing », nous amènent à mettre en évidence des surfaces d'exposition similaires et donc des familles de vulnérabilités communes.

3.1 Services Web

La partie la plus exposée d'une solution « Cloud Computing » est bien entendu la partie web accessible directement sur Internet. Sur les solutions IaaS et PaaS, les web services sont généralement documentés, car ils sont mis en avant comme la manière la plus adaptée d'utiliser le service. Sur les applications SaaS, ce n'est pas toujours le cas, les éditeurs peuvent avoir tendance à vouloir contrôler leur utilisation en évitant de les documenter et en forçant l'utilisation de leurs applications propriétaires.

L'offuscation n'étant pas la solution la plus efficace pour garantir la sécurité d'une API, cette stratégie a déjà posé plusieurs soucis de sécurité, en particulier sur les applications mobiles (iOS/Android) dont l'ingénierie

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

A smaller version of the SANS logo, consisting of the word "SANS" in white on a dark blue background, is located in the bottom right area of the page.



reverse reste relativement aisée. Ces APIs finissent donc par être documentées de manière non officielle et certaines vulnérabilités peuvent être découvertes et exploitées comme ce fut le cas pour WhatsApp [4] par exemple.

3.2 Traitement des données

Une des autres familles de vulnérabilités particulièrement importantes est le mauvais traitement des données mutualisées. Nous venons de voir que le *Software Defined Storage* peut être une source de vulnérabilités lors de l'importation de machines virtuelles au sein d'un IaaS. Ce ne sont pas les seuls contenus importés : les plateformes PaaS doivent exécuter du code fourni par l'utilisateur et les logiciels SaaS doivent bien souvent importer et modifier différents contenus multimédias (images, vidéos...). Toutes ces opérations demandant de parser un grand nombre de formats différents sont autant de sources de vulnérabilités potentielles. Il n'y a qu'à regarder le nombre et la typologie des vulnérabilités publiées sur des bibliothèques standards prévues à cet effet comme LibPNG [5] ou LibXML2 [6], pour se rendre compte que parser des données dans de tels formats est source de dénis de services et d'exécution de code malicieux.

Une fois les données correctement importées sur le service de « Cloud Computing », nos ennuis ne sont pas terminés, il est bien souvent nécessaire de porter une attention particulière aux conditions générales d'utilisation et aux différents mécanismes de partage des contenus entre utilisateurs de la plateforme si nous ne souhaitons pas nous voir déposséder de nos précieuses données. Les services étant mis à jour régulièrement sans intervention de l'utilisateur, il n'est pas rare qu'un contenu considéré comme privé au moment de son importation devienne partiellement ou totalement public lors de la mise en place d'une nouvelle fonctionnalité par l'éditeur du service.

3.3 Mutualisation du matériel

La dernière grande famille de vulnérabilités est liée à la mutualisation du matériel. Même si les différentes technologies de virtualisation offrent un honorable niveau d'isolation entre les machines virtuelles, elles sont elles aussi composées de code source susceptible de contenir différentes vulnérabilités. Ceci est d'autant plus vrai au sein des hyperviseurs modernes (ESXi, Qemu/KVM, Hyper-V) qui, pour des raisons de performances, utilisent la paravirtualisation afin de fournir un accès direct à certains composants matériels.

La problématique n'est pas plus simple au niveau de la partie réseau elle aussi mutualisée. En effet, en plus des vulnérabilités éventuelles sur les différentes solutions d'isolation réseau, il ne faut pas oublier que les utilisateurs doivent eux-mêmes configurer les différents flux autorisés entre les machines virtuelles d'un même compte, mais aussi vis-à-vis des autres utilisateurs et de l'ensemble de l'Internet. Les mises en place de ces machines virtuelles étant la plupart du temps automatisées, il n'est pas rare

de voir des politiques d'accès assez laxistes, par facilité ou tout simplement à cause d'un bug présent dans les scripts d'automatisation.

4 Atterrissage et conclusion

Nous avons pu voir au cours de cette introduction à ce vaste domaine qu'est le « Cloud Computing » différents risques liés de près ou de loin à ces nouvelles technologies. Les acteurs du marché étant pour la plupart relativement jeunes et ces technologies étant encore en constante évolution, un travail de veille est indispensable à quiconque souhaitant allier « Cloud Computing » et sécurité.

De manière à rassurer leurs utilisateurs, la plupart des fournisseurs de solutions « Cloud Computing » adoptent des démarches de certification, que ce soit sur des standards existants comme ISO 27001 ou sur des référentiels spécifiques comme celui de la Cloud Security Alliance. Cette démarche est rassurante pour les utilisateurs, mais n'est pas forcément suffisante. Pour aller plus loin, il peut être tentant de favoriser un fournisseur de cloud « de proximité », plus accessible et plus facilement auditable.

Il peut aussi être intéressant, avant d'envisager toute migration, de réfléchir à comment diminuer la valeur des données confiées au fournisseur de la solution « Cloud Computing », par exemple en chiffrant localement les données qui sont stockées ensuite à distance. ■

■ Références

- [1] « La tête dans les nuages » (juin 2014), <http://www.slideshare.net/MatthieuBouthors/ndh2k14-la-tete-dans-les-nuages>
- [2] « Openflow vulnerability assessment » (2013), <http://conferences.sigcomm.org/sigcomm/2013/papers/hotson/p151.pdf>
- [3] « Surfez sur les réseaux avec Netflow », MISC n°69 (septembre/octobre 2013)
- [4] « Malicious Threats, Vulnerabilities and Defenses in WhatsApp and Mobile Instant Messaging Platforms », <http://www.slideshare.net/segofensiva/malicious-threats-vulnerabilities-and-defenses-in-whatsapp-and-mobile-instant-messaging-platforms>
- [5] LibPNG : 4 vulnérabilités publiées depuis le début de l'année : http://www.cvedetails.com/vulnerability-list/vendor_id-7294/Libpng.html
- [6] LibXML2 : 1 vulnérabilité publiée cette année et 3 l'année dernière : http://www.cvedetails.com/vulnerability-list/vendor_id-1962/product_id-3311/Xmsoft-Libxml2.html

NE MANQUEZ PAS MISC HORS-SÉRIE N°10 !

DOSSIER SPÉCIAL :

RÉAGIR À UNE INTRUSION

**LA CITADELLE EST
TOMBÉE...
...QUE DOIS-JE
FAIRE ?**



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





INTRODUCTION À LA SÉCURITÉ DE LA GESTION DYNAMIQUE DES RESSOURCES DANS LE CLOUD

Sylvie Laniepce et Kahina Lazri – Ingénieurs de recherche à Orange Labs

mots-clés : CLOUD / ÉLASTICITÉ / SÉCURITÉ / VULNÉRABILITÉS / DISTRIBUTED RESOURCE SCHEDULER (DRS)

Pour ce dossier consacré à la sécurité du cloud, nous livrons avec cet article une introduction à un nouveau domaine de vulnérabilités apparu avec la virtualisation et le partage des ressources qu'elle induit, celui de la gestion dynamique des ressources connue sous la terminologie « élasticité du cloud ».

1 Gestion dynamique des ressources

Une des perspectives les plus attractives du cloud réside dans le modèle de paiement à l'usage des ressources informatiques, associé à la propriété attendue d'élasticité du cloud définie comme suit : une infrastructure démontre une propriété d'élasticité autonome si elle a la capacité d'ajuster de manière dynamique et à tout moment les quantités de ressources allouées aux machines virtuelles en fonction des besoins fluctuants de ces dernières. Cela est rendu possible dans ces infrastructures virtualisées du cloud grâce à une couche d'indirection – plus précisément la couche de virtualisation (hyperviseur) – qui s'interpose entre les ressources matérielles et celles virtualisées présentées aux machines virtuelles. La couche de virtualisation permet de reconfigurer par logiciel – à la demande des utilisateurs (de leurs machines virtuelles) – les quantités de ressources allouées aux machines virtuelles.

de ressources consommées par les machines virtuelles sont aisément manipulables par une exécution logicielle sous le contrôle d'un attaquant, agissant avec ses propres machines virtuelles ou avec celles dont il a pris le contrôle localement ou via le réseau. Cette dépendance des décisions des systèmes de gestion dynamique des ressources aux quantités de ressources consommées par les machines virtuelles, constitue par nature une vulnérabilité potentielle qu'il convient de considérer.

Ce domaine des vulnérabilités liées à la gestion dynamique des ressources dans les plateformes partagées du cloud est nouveau. L'attaque par déni de service économique entre dans cette famille d'attaques. Le déni de service économique cherche à prendre avantage du redimensionnement dynamique des ressources pour faire consommer de manière indue des ressources allouées par l'infrastructure, du fait de requêtes inutiles forgées à cette fin par l'attaquant. Le déni de service économique est difficile à détecter puisque les requêtes sont conformes au protocole attendu et leur débit est adapté pour ne pas provoquer d'interruption de services.

1.1 Problématique sécurité

Dans l'état de l'art actuel, les processus de décision des algorithmes de gestion dynamique de ressources prennent en considération uniquement les quantités de ressources matérielles disponibles au sein du pool de ressources partagé ainsi que les quantités de ressources consommées par les machines virtuelles. Or, les quantités

1.2 Principes généraux de la gestion dynamique des ressources

Pour maîtriser et assurer l'équité du partage des quantités de ressources allouées aux différentes machines virtuelles co-résidentes d'un même pool de ressources



(par exemple, un serveur hôte), il est nécessaire de déterminer une valeur plafond de la quantité de ressources allouées à chaque machine virtuelle faute de quoi, une consommation excessive de ressources par une machine virtuelle risque de pénaliser une machine virtuelle co-résidente alors en manque de ressources. Le principal défi des systèmes de gestion de ressources autonomiques réside dans l'ajustement dynamique de cette valeur plafond au fil de la variation des besoins, en tenant compte du caractère fini de la quantité de ressources disponible au sein d'un pool de ressources à partager. Si on veut assurer un contexte d'exécution équitable pour l'ensemble des utilisateurs du cloud, la valeur plafond fixée pour une machine virtuelle est idéale si elle évite les contentions à la fois pour cette machine virtuelle, mais aussi pour les machines virtuelles co-résidentes.

Une sous-estimation de la valeur plafond de la quantité de ressources allouée à une machine virtuelle est susceptible de provoquer des violations des engagements de qualité de services passés entre un opérateur d'infrastructure cloud et ses clients (la dénomination anglaise est '*Service Level Agreement*', SLA) puisqu'il se peut alors, à un moment donné, que la machine virtuelle ne dispose pas des ressources dont elle a besoin. Une surestimation de la valeur plafond n'est pas souhaitable non plus puisqu'elle conduit à sous-utiliser les ressources de l'infrastructure d'hébergement, synonyme de sous-remplissage de l'infrastructure ; des ressources étant allouées à des machines virtuelles qui n'en ont pas l'usage. L'attractivité des offres cloud dépend donc de la capacité de l'opérateur à garantir le respect des SLAs tout en optimisant les taux d'utilisation des ressources garants d'une bonne performance économique.

Pour les offres cloud commerciales actuelles, les taux de consolidation des serveurs et d'allocation des ressources sont optimisés en recourant au sur-engagement maîtrisé des ressources pour définir correctement le point de fonctionnement des plateformes. Les ressources sont dites sur-engagées si les quantités de ressources configurées, toutes machines virtuelles confondues, totalisent une quantité supérieure à la capacité en ressources physiquement disponibles au niveau du pool de ressources partagé.

Ceci est possible, car, la plupart du temps, l'ensemble des machines virtuelles n'utilise pas au même moment la totalité des ressources qui leur sont allouées. De plus, si des contentions de ressources apparaissent, il existe des moyens automatiques de les résorber, en particulier la migration à chaud de machine virtuelle permettant de migrer une machine en cours d'exécution d'un serveur hôte source vers un serveur hôte destinataire libérant ainsi des ressources au niveau du serveur hôte source. Dans cette approche, on retient un principe de surestimation de la valeur plafond de la quantité de ressources allouées à la machine virtuelle évitant ainsi les contentions au niveau de la machine virtuelle, combiné à un sur-engagement maîtrisé des ressources au niveau serveur hôte pour optimiser les taux d'utilisation des ressources.

On voit apparaître d'autres principes possibles, étudiés dans les travaux de recherche actuels, qui recourent à l'ajustement automatique de la valeur plafond de la

quantité de ressources allouée aux machines virtuelles, avec deux approches possibles : la première consiste à prédéfinir des règles à seuils de consommation des ressources qui, lorsqu'elles sont vérifiées, déclenchent automatiquement des actions de réajustement de la valeur plafond ; le deuxième type d'approche repose sur des modèles adaptatifs prédictifs de la consommation en ressources des machines virtuelles, pour anticiper leur comportement de consommation à venir et ajuster pro-activement les valeurs plafond en conséquence.

Ce réajustement automatique de la valeur plafond reste à ce jour un challenge et, pour le relever, deux axes de propositions coexistent. D'une part, l'élasticité verticale (le terme anglais est le '*scaling-up*') où la machine virtuelle se voit ajuster, au sein du serveur hôte l'exécutant, la quantité de ressources qui lui est allouée. D'autre part, l'élasticité horizontale (le terme anglais est le '*scaling-out*') agissant par duplication (et suppression) de machines virtuelles sur différents serveurs hôtes, pour agréger de manière ajustable l'espace de ressources alloué à l'utilisateur.

1.3 Le mécanisme de migration de machine virtuelle

La migration à chaud d'une machine virtuelle consiste à transférer son état courant itérativement d'un serveur hôte source vers un serveur hôte destination, alors que cette machine est en cours d'exécution. Le mécanisme comprend deux phases :

- 1) une phase de pré-copie (phase 1) durant laquelle l'état courant de la VM est itérativement transféré : les pages modifiées avant la terminaison de cette phase de copie, sont copiées de nouveau pour ne conserver que les pages courantes ;
- 2) une phase de 'stop et copie' (phase 2) pour borner dans le temps la phase 1 de copie des pages modifiées : l'exécution de la machine virtuelle sur le serveur hôte source est suspendue pour éviter toute nouvelle modification de page et une ultime copie des pages modifiées est opérée avant de reprendre l'exécution de la machine virtuelle sur le serveur hôte destination.

Les performances de la machine virtuelle migrée peuvent être impactées par le processus de migration sous la forme d'une exécution ralentie pendant toute la durée du processus de migration (appelée en anglais '*migration time*' qui inclut les phases 1 et 2 décrites précédemment) et d'une exécution ponctuellement suspendue durant le laps de temps correspondant à la phase 2 (le terme anglais est le '*down time*'). La durée du processus de migration dépend du type des applications exécutées par la machine virtuelle, de la capacité de la machine virtuelle migrée et du niveau de contention des serveurs hôtes source et destination impliqués dans la migration.

Le recours à la migration à chaud de machines virtuelles en tant que mécanisme de gestion dynamique des ressources doit rester maîtrisé, car il induit des coûts :



- 1) pour l'infrastructure au niveau des ressources consommées par l'hyperviseur des serveurs hôtes source et destination pour exécuter la migration et au niveau de la bande passante réseau utilisée pour le transfert de l'état courant ;
- 2) pour la machine virtuelle migrée comme explicité ci-dessus ;
- 3) possiblement pour les machines virtuelles co-résidentes avec la machine virtuelle migrée, si la consommation de ressources générée par la migration s'ajoute à un état existant de contention au niveau des serveurs hôtes impliqués.

Les algorithmes de gestion dynamique de ressources recourent au mécanisme de migration dynamique de machines virtuelles pour différents objectifs :

- Soit la gestion des contentions : la migration de machines virtuelles permet d'alléger la charge d'un serveur hôte observant des contentions. Les contentions de ressources de niveau machine virtuelle traitées par l'opérateur de l'infrastructure sont celles qui résultent d'une quantité de ressources allouée insuffisante en regard de l'engagement de qualité de service ;
- Soit la consolidation des serveurs hôtes : la migration est utilisée dans un objectif d'optimisation du taux d'utilisation des ressources des plateformes cloud. La migration permet de réviser dynamiquement le placement des machines virtuelles sur les hôtes du pool de ressources en fonction de l'utilisation des ressources par les machines virtuelles, dans le but de minimiser le nombre d'hôtes 'allumés' et d'optimiser la consommation énergétique associée ;
- Soit l'équilibrage de la charge entre les hôtes : un tel équilibre consiste à optimiser dynamiquement le placement des machines virtuelles sur les différents hôtes grâce à la migration, de façon à minimiser la dispersion de charge entre les hôtes. Selon cette approche, on cherche à parfaire l'équilibre de la charge entre les hôtes, car cet état est celui qui correspond à l'état des moindres contentions. En effet, en évitant les situations de dispersion de la charge, on évite que des hôtes soient en surcharge – donc en contention - alors que des ressources sont disponibles par ailleurs sur des hôtes en sous-charge.

2 Attaque par migration intempestive de machine virtuelle

Pour démontrer les possibilités de vulnérabilités de la gestion dynamique des ressources discutées précédemment, nous examinons maintenant l'attaque par migration intempestive de machines virtuelles que nous avons identifiée et analysée pour un système de gestion

dynamique des ressources poursuivant une stratégie d'équilibre de la charge entre les hôtes (stratégie n° 3 détaillée précédemment).

Dans cette attaque, l'attaquant fait fluctuer délibérément et à des fins malveillantes la consommation en ressources de machines virtuelles sous son contrôle, dans le but de créer fréquemment suffisamment de déséquilibre de charge entre les hôtes afin d'amener le système de gestion dynamique des ressources à déclencher des migrations de machines virtuelles pour compenser le déséquilibre créé par l'attaquant et rétablir l'équilibre de charge perdu.

Le lecteur se souviendra de la présentation faite précédemment des coûts générés par les migrations de machines virtuelles pour apprécier que cette prise de contrôle de l'attaquant sur les décisions du système de gestion dynamique des ressources pour lui faire déclencher des migrations, n'est pas sans effet. On précise également que les migrations opérées par le système de gestion ne portent pas nécessairement sur la machine virtuelle manipulée, c'est-à-dire la machine sous le contrôle de l'attaquant et dont la consommation de ressources est à l'origine du déséquilibre de charge, mais au contraire sur la machine virtuelle qui rétablit au mieux l'équilibre de charges une fois migrée et qui, de ce fait, est celle qui subit une partie des effets de l'attaque (dégradation de performances).

2.1 Principe de l'attaque

L'attaquant ajuste la quantité de ressources consommées par ses machines virtuelles, entre des valeurs (exprimées en pourcentage des quantités configurées) très hautes et des valeurs très basses, avec une stratégie d'attaque simple : quand les machines virtuelles sous le contrôle de l'attaquant se mettent à consommer de faibles quantités de ressources, le serveur hôte les hébergeant est alors vu par le système de gestion dynamique des ressources comme étant en sous-charge et ce dernier tend à migrer des machines virtuelles vers ce serveur hôte pour rétablir l'équilibre de charge perdu : les machines virtuelles sont en quelque sorte « attirées » par ce serveur hôte en sous-charge. De manière similaire, lorsque les machines sous le contrôle de l'attaquant se mettent à consommer des quantités importantes de ressources, l'état du serveur hôte les hébergeant est alors considéré par le système de gestion dynamique des ressources comme étant surchargé et ce dernier tend à déclencher des migrations de machines virtuelles pour alléger la charge de ce serveur hôte et rétablir l'équilibre de charge entre les serveurs hôtes : les machines virtuelles sont en quelque sorte « repoussées » hors du serveur hôte.

L'effet de l'attaque peut être cumulatif si l'augmentation de la quantité de ressources consommées par les machines virtuelles sous le contrôle de l'attaquant fait suite à une diminution de la quantité de ressources consommées ayant conduit à l'attraction de machines virtuelles. En effet, l'effort de repoussement de machines virtuelles bénéficie dans ce cas d'un état favorable à l'attaque,



déjà chargé par l'attraction obtenue préalablement et donc plus proche de l'état de surcharge provoquant un déséquilibre suffisant pour amener le système à le compenser par migrations.

Pour renouveler l'état de déséquilibre de la charge compensé par les migrations opérées par le système de gestion dynamique des ressources et ainsi réussir à amener le système à une succession de migrations, l'attaquant fait fluctuer fréquemment entre valeurs basses et valeurs hautes les quantités de ressources consommées par ses machines virtuelles. À chaque création de déséquilibre de charge, le système entre dans une procédure de rétablissement de l'équilibre en opérant des migrations, dès lors que le déséquilibre créé par l'attaquant est supérieur au maximum de déséquilibre toléré par l'algorithme.

Pour que l'attaque menée produise des effets significatifs, c'est-à-dire un nombre élevé de migrations successives, l'attaquant doit pouvoir influencer le déséquilibre de charge avec la manipulation d'une quantité de ressources suffisante. Cela requiert que l'attaquant manipule de manière coordonnée les quantités de ressources consommées par plusieurs machines virtuelles, chacune d'entre elles produisant un déséquilibre cumulatif de ceux créés par les autres machines virtuelles également sous son contrôle.

En particulier, la capacité d'influence de l'attaquant sur le déséquilibre de charge est plus importante si l'attaquant contrôle la consommation en ressources de machines virtuelles réparties sur deux serveurs hôtes différents. Dans ce cas, il a la possibilité de coordonner les consommations pour qu'elles fluctuent de manière cyclique et en opposition de phase entre le premier serveur hôte et le deuxième serveur hôte. Si les machines virtuelles consomment des quantités de ressources élevées sur le premier serveur hôte quand celles sur le deuxième serveur hôte consomment de faibles quantités, le déséquilibre de charge créé entre les serveurs hôtes est plus conséquent et l'attaquant forge ainsi plus aisément des conditions de migration successives qui s'imposent au système de gestion dynamique de ressources.

2.2 Preuve de concept

Le lecteur intéressé peut se référer à une publication scientifique [1] des auteurs de cet article, présentée à SSTIC'14 [2], pour un exposé détaillé d'une preuve de concept de cette attaque réalisée à l'encontre de l'algorithme VMware DRS (*Distributed Resource Scheduler*) qui est l'algorithme de gestion dynamique des ressources de la plateforme virtualisée VMware. Les figures 1 et 2 donnent une représentation de cette preuve de concept en illustrant le déroulement de l'attaque coordonnée réalisée [3], produisant une série de migrations de machines virtuelles.

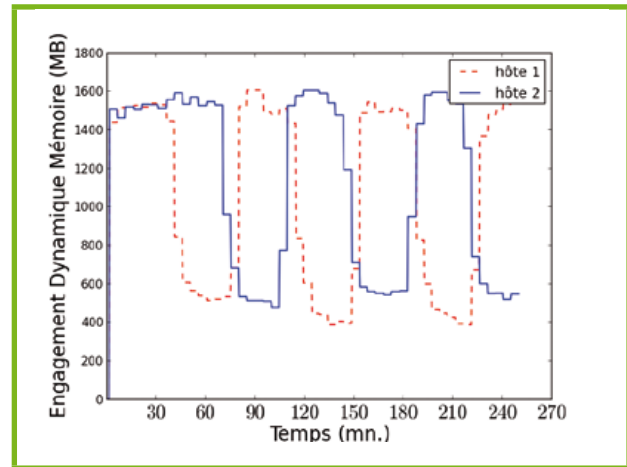


Fig. 1 : Supervision de l'attaque coordonnée, au niveau machine virtuelle.

La figure 1 illustre la variation dans le temps, cyclique et en opposition de phase, des quantités de ressources mémoire consommées par deux machines virtuelles sous le contrôle de l'attaquant, s'exécutant respectivement sur un premier serveur hôte et un deuxième serveur hôte, au travers du suivi de la métrique VMware d' 'Engagement Dynamique' (en anglais, 'Dynamic Entitlement') tout au



NSC

No Such Con 2014

"What you're about to watch is a nightmare..."

WHERE & WHEN
Espace Oscar Niemeyer
19-20-21 November 2014

CONTENT
Top notch keynoters & speakers

OPEN BAR
Free beers, coffee and soft drinks for 3 days
*Save your money from cheap beers
to buy your ticket online!*

SOCIAL EVENT
Free beverages (beers & soft drinks), free food and
party @La Rotonde [Place Stalingrad]

The bullshit-free conference www.nosuchcon.org

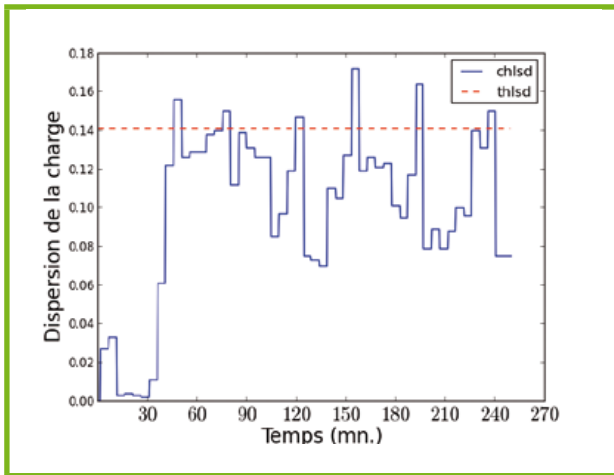


Fig. 2 : Supervision de l'attaque coordonnée, au niveau cluster.

long du déroulement de l'attaque. La figure 2 illustre la variation associée de la dispersion de la charge entre les hôtes du cluster, induite par la consommation des machines virtuelles du cluster (incluant celles sous contrôle de l'attaquant). La métrique VMware utilisée pour représenter la mesure de la dispersion courante de la charge est l'Écart type du chargement des hôtes courant' (en anglais, 'Current Host Load Standard Deviation' (chlsd)). Sur cette figure 2, la courbe horizontale apparaissant en pointillés représente le seuil de dispersion de charge maximum toléré au-delà duquel DRS entre dans un processus de décision pour évaluer le besoin et les possibilités de migrations de machines virtuelles pour rétablir l'équilibre de charge ; cette valeur seuil est donnée par la métrique 'Ecart type du chargement des hôtes cible' (en anglais, 'Target Host Load Standard Deviation' (thlsd)).

La figure 2 démontre l'occurrence de six migrations de machines virtuelles (aux points de dépassement du seuil 'thlsd') synchronisées dans le temps avec les fluctuations des profils de consommation des machines virtuelles sous le contrôle de l'attaquant.

3 Nouveaux défis de sécurité

Le type d'attaque détaillée précédemment présente la particularité de produire des effets au-delà de l'espace de ressources manipulé par l'attaquant : le système attaqué (i.e. les machines virtuelles sous le contrôle de l'attaquant, éventuellement ses propres machines virtuelles, dont l'espace de ressources est manipulé) n'est pas le système victime de l'attaque (la machine virtuelle migrée). Par conséquent, aucune intrusion n'est observée sur le système victime de l'attaque.

La plupart des méthodes classiques de détection d'attaques reposent sur la supervision de l'activité du système à protéger, dans le but de collecter des informations représentatives et significatives de l'utilisation des ressources de ce système. Si ces informations révèlent une activité

anormale - présence d'un code malveillant, réception de paquets malveillants par le réseau, augmentation de charge suspecte, erreurs d'authentification... - une alerte est levée puis analysée pour conclure (ou pas) à une attaque en fonction des politiques de sécurité applicables. Ces techniques centrées sur la cible sont donc inefficaces pour détecter les attaques décrites ci-dessus qui ciblent une machine virtuelle par manipulation des ressources d'une autre machine virtuelle.

Plusieurs types d'approches méritent d'être pris en compte pour faire face à ce type de vulnérabilités. Tout d'abord, il est nécessaire de concevoir des systèmes de gestion dynamique des ressources robustes qui offrent une forme de 'résistance' aux profils de consommation de ressources (tels que ceux présentés dans cet article) possiblement à l'origine d'anomalies de fonctionnement pour ces systèmes, tout en garantissant des taux optimaux d'utilisation des ressources des infrastructures. Par ailleurs, il faudrait considérer l'apport des politiques de placement des machines virtuelles sur les différents serveurs hôtes dans la maîtrise de la capacité d'influence d'une machine virtuelle au sein de l'espace de ressources qu'elle partage avec d'autres machines virtuelles. Enfin, il est nécessaire de superviser les systèmes de gestion dynamique des ressources pour détecter les anomalies de fonctionnement et les contraindre à adapter automatiquement leur fonctionnement aux menaces. Toutes ces voies partagent une même motivation, celle de prendre en compte des attributs de sécurité globaux - par opposition à des attributs restreints au seul contexte d'exécution de la machine virtuelle considérée - pour permettre à l'opérateur d'infrastructure cloud de traiter de la sécurité du partage dynamique et optimisé des ressources. ■

■ Références

- [1] K. Lazri, S. Laniepece, H. Zheng, J. Ben-Othman, *Sécurité de la gestion dynamique des ressources dans le cloud : prise de contrôle sur le déclenchement de migrations automatiques de machines virtuelles*, 12ième Symposium sur la Sécurité des Technologies de l'Information et des Communications (SSTIC'14), juin 2014, Rennes, France
- [2] SSTIC'14 : Symposium sur la Sécurité des Technologies de l'Information et des Communications, Rennes, juin 2014 https://www.sstic.org/2014/presentation/scurit_de_la_gestion_dynamiqu_cloud/
- [3] Preuve de Concept réalisée sur plateforme de laboratoire (cluster de 5 hôtes), avec les versions logicielles suivantes : hyperviseur VMware ESXi 4.1 et logiciel de management de cloud vCenter 4.1.



LE CLOUD GAULOIS, UNE RÉALITÉ ! VENEZ TESTER SA PUISSANCE

EXPRESS HOSTING

Cloud Public
Serveur Virtuel
Serveur Dédié
Nom de domaine
Hébergement Web

ENTERPRISE SERVICES

Cloud Privé
Infogérance
PRA/PCA
Haute disponibilité
Datacenter

EX10

Cloud Hybride
Exchange
Lync
Sharepoint
Plateforme Collaborative



sales@ikoula.com



01 84 01 02 66



express.ikoula.com



sales-ies@ikoula.com



01 78 76 35 58



ies.ikoula.com



sales@ex10.biz



01 84 01 02 53



www.ex10.biz



MÉCANISMES DE SÉCURITÉ AU NIVEAU PROCESSEUR POUR LA VIRTUALISATION

Loïc Guillois – contact@loicguillois.fr – Concepteur d'applications Web, Mobile & TV

mots-clés : MACHINE VIRTUELLE / AMD-V / INTEL VT-X

Cet article propose une présentation des technologies mises en œuvre au niveau matériel pour assurer la protection des machines virtuelles sur plateforme x86. Ces solutions sont notamment utilisées par certaines plateformes de « cloud computing ».

1 Cloud et virtualisation

1.1 Les bénéfices de la virtualisation

La virtualisation consiste à faire fonctionner plusieurs systèmes d'exploitation comme de simples logiciels, sur un ordinateur. Ces machines virtuelles sont généralement appelées serveur privé virtuel (VPS) par les hébergeurs. La virtualisation offre de nombreux avantages. Ses caractéristiques lui ont permis de se développer rapidement dans les salles serveurs et chez les hébergeurs. Les offres de cloud computing s'appuient directement sur ce type de solution pour concevoir leur plateforme à la demande « PaaS » (hébergement d'applications) ou infrastructure à la demande « IaaS » (serveur Linux ou Windows). L'utilisation des ressources est alors optimale grâce à une répartition des machines virtuelles sur les machines physiques en fonction des pics de charge. La flexibilité d'utilisation permet une administration plus productive tout en maîtrisant les coûts grâce à une mutualisation des ressources.

Les machines virtuelles permettent la mise en œuvre d'une sécurisation par l'isolation des différents utilisateurs simultanés d'une même machine. Cette isolation n'est cependant pas forcément parfaite et des vulnérabilités permettant de contourner les mécanismes d'isolations entre les machines virtuelles peuvent exister. Ce risque expose les machines virtuelles à des fuites d'informations et à des prises de contrôle du système hôte ou invité. Nous allons voir que les constructeurs ont mis en place des solutions matérielles permettant de renforcer significativement la sécurité des environnements virtuels.

1.2 Les solutions

Le projet Xen fait parti avec KVM des plateformes open source de virtualisation les plus utilisées chez les hébergeurs qui proposent des offres cloud dans le monde. On sait notamment que Xen est utilisé par Amazon, Rackspace ou encore Verizon. En France, c'est Gandi qui fait partie de ses plus grands utilisateurs. KVM est quant à lui utilisé par IBM et l'hébergeur américain The Planet de même que le nantais Clever Cloud.

Du côté des solutions propriétaires, Microsoft est présent avec Hyper-V. Cette solution est notamment utilisée pour le fonctionnement de son offre de « cloud computing » Windows Azure et par les hébergeurs français Ikoula, LinkByNet ou encore OVH. Enfin, VMWare est historiquement l'un des premiers à offrir une solution commerciale. Leurs produits possèdent une certaine maturité qui leur a permis de se positionner dans les datacenters de nombreux opérateurs tels que SFR Business Team, Orange Business ou encore OVH.

2 L'évolution du matériel

2.1 Les performances

L'avènement des processeurs multi-cœur et de l'*hyperthreading* a permis aux logiciels d'être plus réactifs en assurant de meilleurs temps de réponse. Les performances varient sensiblement selon les applications puisqu'il est nécessaire de les concevoir en prenant en



compte le parallélisme. En revanche, lorsqu'il s'agit d'exécuter plusieurs logiciels, la charge se trouve plus naturellement répartie entre des cœurs disponibles. Les performances des processeurs couplées à la quantité de mémoire vive disponible ont permis le développement de la virtualisation.

2.2 Les instructions spécifiques

Les constructeurs se sont naturellement adaptés à cette nouvelle tendance pour proposer le support de la virtualisation. Celle-ci peut être intégrée au processeur ou assistée par celui-ci, le matériel se chargeant, par exemple, de virtualiser les accès mémoire ou de protéger le processeur physique des accès les plus bas niveaux. Cela permet de simplifier la virtualisation logicielle et de réduire la dégradation de performances. L'objectif étant de se rapprocher du fonctionnement du système d'exploitation natif sur un CPU dédié.

On retrouve la virtualisation matérielle sur toutes les gammes de stations de travail et serveurs professionnels notamment IBM Power6, Sun et HP. En ce qui concerne les offres d'hébergement web et de « cloud computing » disponibles, les processeurs sont de type x86. Les principaux constructeurs sont donc AMD et Intel. Nous allons nous concentrer dans cet article sur cette architecture et notamment sur les instructions AMD-V et Intel VT.

2.3 Trusted Module Platform

Un nouveau composant est apparu sur les cartes mères de nos PC : le module TPM. Celui-ci est utilisé par les hyperviseurs au travers d'instructions processeurs spécifiques AMD et Intel. Son utilisation permet notamment de sécuriser la phase de démarrage du système d'exploitation invité. En effet, grâce au TPM les solutions de virtualisation peuvent effectuer des hashes des modules de démarrage (BIOS, MBR, bootloader, etc.) et ainsi déceler tout endommagement ou mises à jour inattendues ou non autorisées de ceux-ci.

Ce composant a vu le jour sur certains modèles d'ordinateurs portables, principalement dans les gammes professionnelles HP, Dell, Sony, Lenovo et Toshiba entre autres. Il s'agit d'un composant passif. Cela signifie qu'il ne peut pas donner d'ordre à l'ordinateur tel que bloquer le système ou surveiller l'exécution d'une application. En revanche, il permet de facilement stocker des secrets tels que des clefs de chiffrement ou des hashes de manière sécurisée.

Il est nécessaire d'activer cette fonctionnalité dans le BIOS de la carte mère. Il suffit d'accéder à la configuration de celui-ci et de sélectionner la valeur **Enabled** pour la propriété « **Enable TPM-Chip** ». Sous Linux, un module noyau est nécessaire : *tpm*.

Pour vérifier que votre système possède un TPM V1.2 :

```
$ modprobe tpm_tis force=1 interrupts=0
$ cat `find /sys -name pcrs`
```

Une liste de 24 PCRs (*Platform Configuration Registers*) doit s'afficher. Si vous n'en avez que 16, cela signifie que le TPM est en version 1.1b.

Le TPM s'exploite en assembleur et en langage C en général, mais il est possible de l'utiliser avec des bibliothèques de plus haut niveau. Avec Node.js, on peut notamment utiliser le module *node-tss* qui permet d'effectuer les manipulations en JavaScript. Voici un exemple :

Ici, on tente de récupérer les différents PCR.

```
var pcr = [];
for (var i=0; i<8; i++) {
  console.log("Reading pcr " + i + "...");
  pcr[i] = tssbridge.getPCR(i);
  assert.equal(pcr[i].length, 20, "PCR " + i + " correct length");
  if (i<8) {
    assert.notDeepEqual(pcr[i], [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0], "PCR " + i + " not empty");
  }
}
```

Ici, on tente d'étendre le PCR numéro 7 et on vérifie que la modification a été prise en compte.

```
tssbridge.extendPCR(7, [0x00,0x01,0x02,0x03,0x04,
  0x10,0x11,0x12,0x13,0x14,
  0x20,0x21,0x22,0x23,0x24,
  0x30,0x31,0x32,0x33,0x34]);
var pcrNewSeven = tssbridge.getPCR(7);
assert.notDeepEqual(pcrNewSeven,pcr[7], "PCR 7 hasn't changed post extend");
```

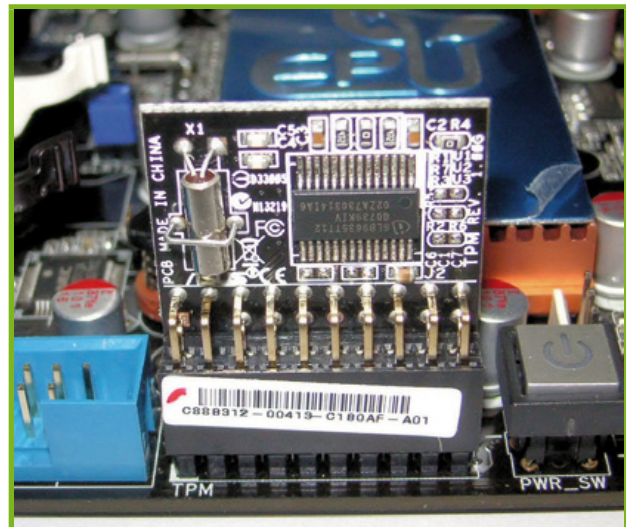


Fig. 3 : Trusted Platform Module installé sur une carte mère.

OSLO est un programme d'amorçage focalisé sur la sécurité, d'où son nom, acronyme « Open Secure Loader ». Il s'appuie sur les instructions spécifiques des processeurs AMD (SKINIT) afin de permettre un démarrage authentifié. Il repose sur environ 1000 lignes de code qui ont fait l'objet d'environ 4000 revues de code et correction. Ce qui en fait un élément important



dans l'implémentation d'une plateforme de confiance. Le projet a d'ailleurs été fondé par le consortium européen *Open Trusted Computing* (OpenTC).

OSLO est lancé comme un noyau à partir d'un programme d'amorçage multi-boot (par exemple GRUB). Il initialise le TPM (*Trusted Module Platform*) appelé SKINIT, hache tous les modules qu'il récupère depuis le programme d'amorçage parent avec SHA et enfin, lance le premier module en tant que nouveau noyau. Étant donné qu'OSLO a pour vocation d'être le plus sécurisé possible, ses fonctionnalités sont réduites au strict minimum. C'est pour cette raison qu'il ne contient pas de code permettant de charger une image depuis le réseau ou un disque dur.

Une extension du programme d'amorçage GRUB permet également la prise en charge de cette fonctionnalité : TrustedGRUB.

À l'image d'OSLO et de TrustedGRUB, l'ensemble des solutions du marché (Xen, KVM, Hyper-V, VMware

vSphere...) exploite le TPM. Si l'on s'intéresse en particulier à l'hyperviseur ESXi, on apprend que celui-ci effectue un hash sur l'intégralité de la pile de l'hyperviseur lors du démarrage du système et stocke ces hashes dans les registres PCR du TPM. Au niveau du code exécutable, le VMkernel, les modules du noyau, les pilotes et les applications de gestion natives sont ainsi authentifiés. ESXi effectue également un hash de l'ensemble des options de configuration du démarrage.

3 Les instructions processeur spécifiques

3.1 Zoom sur AMD (AMD-V)

Afin d'initialiser un environnement virtuel de confiance (DRTM, *Dynamic Root of Trust for Measurement*), les logiciels exécutés en ring 0 (niveau noyau) invoquent les instructions SKINIT. Celles-ci font partie de l'ensemble des solutions matérielles disponibles sur les processeurs AMD. Le constructeur fait référence à cette architecture grâce au nom *Secure Virtual Machine* (SVM). Il peut s'agir soit d'un programme d'amorçage, soit du système d'exploitation. Ces instructions prennent pour unique argument un bloc d'adresse mémoire physique appelé SLB (*Secure Loader Block*).

Afin de protéger le lancement du SLB contre toute attaque telle qu'une mise à jour non autorisée, le processeur inclut un certain nombre de protections matérielles. Lorsque le processeur exécute l'instruction SKINIT, cela va désactiver les accès directs à la mémoire (DMA) aux pages de mémoires physiques comprenant le SLB grâce au *Device Exclusion Vector* (DEV). Ces instructions vont également désactiver les interruptions matérielles afin d'éviter une prise de contrôle par le code exécuté. Le débogage sera également désactivé. Au final, le processeur entre en mode protégé 32-bit et va au point d'entrée spécifié.

AMD-V inclut également le support de l'attestation d'invocation du SLB. Avec les instructions SKINIT (*Security Kernel Initialization*), le processeur exploite le TPM afin d'obtenir un hash de celui-ci. Le TPM contient notamment les registres sécurisés (PCR). Les instructions SKINIT vont remettre à zéro les valeurs des PCRs allant du numéro 17 au numéro 23. Cela permet de mettre le TPM dans son état initial tel que le ferait un redémarrage de l'ordinateur. La valeur du SLB est envoyée au TPM et son hash. Dans le cas de TPM v1.0, ce sera SHA-1 qui sera utilisé. Cet algorithme ayant subi une attaque qui a été publiée, il n'est plus considéré comme fiable. Ce problème a été résolu à partir de TPM v2.0 qui utilise désormais SHA-2. La valeur du hash est donc stockée dans le registre 17 prévu à cet effet. La seule possibilité pour modifier cette valeur étant d'utiliser à nouveau une instruction SKINIT. Cette valeur sera utilisée pour authentifier le SLB chargé. AMD ne peut protéger qu'un SLB de 64Ko :

Virtualisation réseau

Le spécialiste des réseaux Cisco a travaillé avec VMware à la mise en œuvre d'un format d'encapsulation nommé VxLAN (*Virtual eXtensible LAN*). Ce format apporte des fonctionnalités semblables aux VLANs. Il permet de faire transiter des paquets de niveau 2 dans UDP (niveau 4). Cette propriété permet donc d'utiliser la segmentation de type VLAN au-delà d'un domaine Ethernet. Le fonctionnement de VxLAN reposant sur IP, celui-ci donne aux architectures l'utilisant l'avantage d'être scalable. NVGRE (*Network Virtualization using Generic Routing Encapsulation*) est une solution similaire portée essentiellement par Microsoft, mais également des fabricants carte réseau comme Broadcom, de processeurs comme Intel ou encore les assembleurs Dell et HP. Le principe repose sur *Generic Routing Encapsulation* (GRE) et permet d'envoyer les paquets de la couche niveau 2 par un tunnel sur la couche niveau 3. VMware et Red Hat soutiennent également ce projet.

D'autres formes de virtualisation existent comme VNT (*Virtual Network over TRILL*), implémentation réalisée par les équipes R&D de l'hébergeur Gandi. Cette technologie lui permet de proposer à ses clients une infrastructure performante et capable de monter en charge avec sérénité : la limite de 4096 VLANs a été repoussée à 16 millions de VLANs au sein d'un même réseau, tout en soulageant le matériel réseau. Pour y parvenir, TRILL utilise le protocole IS-IS pour échanger les tables MAC entre les switchs et supprime STP.



code exécutable et tas inclus. Cette authentification permet de détecter toute tentative de modification non autorisée du SLB en levant une exception.

AMD propose également par défaut un effacement automatique de la mémoire des machines virtuelles lors de l'arrêt de celles-ci. Cette fonctionnalité évite toute attaque visant à récupérer des données sensibles après l'utilisation d'une machine virtuelle.

Les exceptions sont produites lorsque le processeur rencontre une erreur. AMD a ajouté une exception pour couvrir le domaine de la sécurité (**#SX**). Celle-ci permettra notamment de provoquer un effacement des données et de réinitialiser la machine virtuelle. On peut noter qu'il existe également une exception pour la virtualisation (**#VE**). Cette exception sera exécutée lorsqu'une violation de page mémoire se produira. Ces exceptions sont bien sûr prises en charge par le système hôte et donc par le logiciel de virtualisation. Le système invité n'en ayant pas connaissance.

Pour savoir si votre processeur supporte cette technologie, sous GNU/Linux il suffit de regarder si **svm** est présent dans **/proc/cpuinfo**.

3.2 Zoom sur Intel (VT-x)

Intel Trusted Execution Technology (TXT) fait référence à l'ensemble des fonctionnalités de sécurisation matérielles mises à disposition par VT-x et notamment le *Safer Mode Extensions* (SMX). Les systèmes les plus récents incluent également Directed I/O (VT-d). SMX permet donc d'implémenter le DRTM et est en ce sens similaire à l'architecture SVM d'AMD. Nous allons donc nous concentrer sur ce qui distingue ces deux technologies. Au lieu de SKINIT, Intel a introduit une instruction nommée GET-SET. Celle-ci exécute différentes opérations en fonction de la valeur présente dans le registre EAX. Celui-ci est utilisé habituellement pour les opérations arithmétiques et le stockage de la valeur de retour des appels systèmes. L'équivalent de SKINIT est donc l'instruction **GETSEC[SENTER]**.

Un DRTM invoqué avec cette instruction sera constitué de deux phases. Dans un premier temps, le module appelé ACMod (*Authenticated Code Module*) doit être chargé en mémoire. Le chipset vérifiera la signature de l'ACMod en utilisant une clef publique intégrée. Ce système complète donc le hash réalisé par le TPM dans le registre PCR 17. Ensuite, le module ACMod est exécuté et mettra en place l'équivalent du SLB d'AMD (*Secure Loader Block*). Ce système équivaut sur processeur Intel porte le nom de MLE (*Measured, Launched Environment*). Tout comme AMD et sa technologie DEV, Intel protège les zones mémoires contenant le ACMod et le SLB de tout accès direct. Là encore, l'intérêt de cette authentification du code est d'en détecter toute modification non autorisée.

Intel TXT ne permet pas un effacement automatique de la mémoire. Ce sera à l'hyperviseur d'appeler l'instruction **VMCLEAR** une fois la machine virtuelle arrêtée.

Spécification SR-IOV

De la même manière qu'il existe des instructions spécifiques sur les processeurs modernes pour optimiser la virtualisation, les cartes réseaux ont également évolué pour apporter un gain de performance et de sécurité aux systèmes invités. SR-IOV est une spécification qui permet à une carte PCI express d'apparaître comme plusieurs cartes physiques. Pour fonctionner, il est nécessaire d'avoir un BIOS et chipset compatibles, mais également un système hôte qui prend en charge cette fonctionnalité. La majorité des plateformes de para-virtualisations sont compatibles.

Pour savoir si votre processeur supporte cette technologie, sous GNU/Linux, il suffit de regarder si **smx** et **vmx** sont présents dans **/proc/cpuinfo**. Sous Mac OS, on pourra vérifier avec la commande suivante :

```
$ sysctl -n machdep.cpu.features
```

Conclusion

Fin 2011, des travaux ont été menés afin d'apporter plus de sécurité aux plateformes de cloud computing. Cette initiative portant le nom d'un framework en devenir : « CertiCloud ». Un prototype a été réalisé autour de Nimbus, une plateforme cloud open source. Ce prototype a montré la faisabilité et le faible coût permettant la mise en place d'une plateforme cloud « IaaS » sécurisée. Les ressources matérielles qu'apportent les *Trusted Platform Module* (TPM) offrent ainsi à l'utilisateur un environnement virtuel sécurisé. Aujourd'hui, de plus en plus de plateformes proposent leur implémentation : XenServer, KVM, VMware... L'ensemble de ces solutions est donc capable de fournir une attestation de l'image de l'hyperviseur sur une base matérielle de confiance. Les instructions spécifiques à la sécurité que nous avons pu traiter dans cet article et notamment Intel TXT et SKINIT sont également exploitées et les plateformes cloud en profitent donc également. ■

■ Références

- [1] « *The Definitive Guide to the Xen Hypervisor* » (novembre 2007)
- [2] « *Attacking Intel® Trusted Execution Technology* » (février 2009), <http://digitalpiglet.org/teaching/stonybrook/CSE509.2012.F/Attacking.Intel.TXT-slides.pdf>
- [3] « *CertiCloud : a Novel TPM-based Approach to Ensure Cloud IaaS Security* », <http://certicloud.gforge.uni.lu/techrep/tccr.pdf>
- [4] « *Le Cloud à la française !* », <http://blog.octo.com/le-cloud-a-la-francaise> (juillet 2011)



LE CLOUD : CETTE NÉBULEUSE JURIDIQUE

Tris Acatrinei

@tris_acatrinei – www.hackersrepublic.org

Consultante pour FAIR-Security

mots-clés : DROIT DES CONTRATS / SABU / VIE PRIVÉE

Terme devenu à la mode pour désigner une réalité technique finalement assez ancienne, le Cloud Computing soulève certaines questions juridiques, mais également éthiques.

1 Un cumulus contractualisé

Cassons immédiatement tous les a priori concernant le Cloud et son régime juridique. La technique n'est pas nouvelle et son environnement juridique n'est absolument pas novateur puisqu'en France, il dérive du droit des contrats, de la loi de 1978 dite Informatique et Libertés et de quelques directives communautaires.

Nous n'allons pas revenir sur la définition du Cloud Computing : les lecteurs de MISC en connaissent les principales caractéristiques, mais nous allons nous attarder quelques instants sur sa définition juridique. Le Cloud est à la fois une prestation de service et une fourniture matérielle. Une entreprise française va offrir à un client un espace de stockage en France – donc un matériel ainsi qu'une prestation de service, pour la mise en place des accès du client au dit matériel. L'avantage pour le client reste le coût très réduit de la prestation et pour l'entreprise, la mutualisation du service : il peut offrir un même espace à différents clients. On est donc dans un schéma très classique du droit des contrats. On peut éventuellement considérer qu'il y a une chaîne si l'entreprise offrant un espace de stockage externalise également une partie des prestations, toujours en France. Auquel cas, si le client subit une panne, il se retournera contre l'entreprise qui elle-même se retournera contre le sous-traitant. Sur le plan de la responsabilité contractuelle, il n'y a rien de novateur : les chaînes de contrats sont un classique que l'on rencontre assez fréquemment dans le BTP.

Pour le prestataire, il va s'agir de répondre à certaines obligations légales. Il devra être en accord avec la loi de 1978 dite Informatique et Libertés pour le traitement des données à caractère personnel, il devra se conformer

aux exigences de la loi pour la confiance dans l'économie numérique (LCEN), selon la nature des données qu'il prétend stocker/héberger et il devra être certifié et respecter la réglementation sectorielle. Le cadre est strict et peut même sembler trop étriqué. Pourtant, l'ensemble de ces dispositions a été fait pour protéger les clients, notamment pour empêcher les hébergeurs de vendre des données ne leur appartenant pas.

En effet, lorsqu'une personne va externaliser ses données, elle va perdre la maîtrise sur la gestion quotidienne, la maintenance, cela revient à confier les clefs de son domicile à un(e) inconnu(e) sans être sûr(e) que la personne ne va s'en aller avec l'argenterie [1]. L'ensemble de ce dispositif donne une forme de garantie. Cette garantie s'exprime dans le contrat que la personne va former avec le prestataire : toutes les clauses y sont importantes, car ce sont elles qui définiront la responsabilité des cocontractants. Savoir où est physiquement le serveur peut avoir son importance, de même que les logiciels utilisés, la localisation du siège, l'obligation de moyens ou de résultats, etc. Il n'existe pas de contrat-type : il y a autant de contrats possibles qu'il existe de scénarios dans l'externalisation des données. Quelles peuvent être les questions à se poser avant de signer avec un prestataire ?

- Quel est le siège social de l'entreprise ?
- Dans quel État se situe le siège social ?
- Appartient-elle à un consortium ou à un groupe d'entreprises ?
- Où seront physiquement stockées les données ?
- Qui seront les différents intervenants ?
- Quel est le lien entre les différents intervenants avec l'entreprise avec laquelle on envisage de signer ?
- Sous quel droit le contrat est-il conclu ?



- L'obligation de fourniture de service est-elle de moyens ou de résultat ?
- Quelles sont les clauses d'exonération de responsabilité du prestataire [2] ?
- Quels sont les logiciels utilisés ?
- Le support est-il compris dans la prestation [3] ?
- Existe-il un plan d'assurance sécurité ?
- Quels sont les indicateurs de performance ?

Cette liste est évidemment non-exhaustive, mais peut servir de base à une mise en concurrence efficace des différents prestataires.

Pourquoi un tel niveau d'exigence ? Nous l'avons déjà dit, le client héberge ses données ailleurs, ce qui veut dire qu'il n'a pas la maîtrise de l'environnement technique ni matériel et qu'il ne peut pas contrôler qui a accès aux dites données.

Même quand le prestataire est Français, que son personnel est régi sous contrat de droit français, que son datacenter est sur le territoire Français, l'intégrité du matériel ne peut pas être garantie ni les aléas extérieurs comme une panne d'électricité, une inondation ou un tremblement de terre. Ces aléas sont qualifiés en droit : c'est la force majeure, mais pour que l'exonération de responsabilité contenue dans l'article 1148 du Code Civil puisse être actionnée, il faut que le fait ayant entraîné le dommage soit imprévisible et irrésistible, les deux conditions étant cumulatives.

Cas concret : une météorite, que personne n'a vue venir, tombe directement sur un datacenter. Comme elle a échappé aux satellites à cause de sa petite taille, les scientifiques ne l'ont pas détecté avant son point d'impact. Le caractère imprévisible est rempli. Et comme on ne sait toujours pas arrêter les météorites, on peut également considérer que le caractère irrésistible est rempli [4].

À l'inverse, une coupure d'électricité due à une période de grand froid n'est pas un cas de force majeure. Ainsi, dans un arrêt du 11 mars 1986, le Tribunal de Grande Instance d'Angers a énoncé que la responsabilité d'EDF doit être retenue du fait des interruptions de courant constatées dès lors qu'elle ne justifie pas de ce qu'il lui était impossible de parer à une période de grand froid inhabituelle. Le client peut se retourner contre le prestataire qui va lui-même se retourner contre le fournisseur d'électricité, en faisant jouer une action récursoire. Le fournisseur d'électricité qui ne peut pas justifier qu'il lui était impossible de parer à une période de grand froid verra sa responsabilité engagée [5].

Les cas d'exonération de responsabilité contractuelle sont textuellement limités et concrètement illustrés. Le Code Civil – toujours sous l'article 1148 – donne de nombreuses situations et réponses de la jurisprudence.

À notre connaissance, en matière de fournisseur de Cloud, il n'y a pas de jurisprudence spécifique relative à la responsabilité contractuelle. À défaut, on peut toujours

se référer à cet arrêt de la Première Chambre Civile de la Cour de Cassation du 19 novembre 2009, n°08-21-645 qui énonce – au sujet d'un fournisseur d'accès à Internet – que ce dernier est tenu à une obligation de résultat quant aux services offerts et qu'il ne peut s'exonérer de sa responsabilité à l'égard de son client en raison d'une défaillance technique, hormis le cas de force majeure.

2

Une sécurité en cirrocumulus

Tous les prestataires de Cloud ne se valent pas et tous ne disent pas leurs noms. Or, ne pas connaître l'identité exacte de son interlocuteur est une faille de sécurité exploitable et la triste histoire d'iCloud est assez révélatrice. Il y a fort à parier qu'une partie des utilisateurs des systèmes fournis par Apple Inc. ne savent pas que leurs données sont quelque part et ne remplissent donc pas les caractéristiques classiques de la sécurité.

En 2012, la CNIL a défini le Cloud de la façon suivante [6] : « *Le Cloud computing est la forme la plus évoluée d'externalisation, dans laquelle le client ou l'utilisateur dispose d'un service en ligne dont l'administration et la gestion opérationnelle sont effectuées par un sous-traitant (externe ou interne). Le Cloud se caractérise également par une facturation à la demande et une disponibilité quasi-immédiate des ressources. Dans de nombreux cas, le client ne connaît pas la localisation des données. La simplicité et le faible coût des solutions « Cloud » intéressent de nombreuses entreprises, mais peut poser des problèmes au regard de la protection des données personnelles et notamment sur les questions de responsabilité du sous-traitant, de sécurisation du traitement de données personnelles ou de transferts de données vers des pays ne disposant d'une législation protectrice des données personnelles.* »

Le premier point mettant en danger la sécurité d'une personne ayant recours à un service de Cloud est la localisation. Au-delà des aspects soulignés dans la définition précédente, se pose la question de la confidentialité des informations par rapport à une entité étatique, qui va être fonction du système juridique dont le prestataire technique va dépendre, le client n'étant plus qu'un pion dans une bataille qui va opposer le prestataire au système judiciaire. L'exemple le plus significatif est le cas de Microsoft.

Microsoft s'est vu demander le 4 décembre 2013 une demande d'accès à une boîte mail fournie par cette dernière [7], dans le cadre d'une enquête sur un trafic de stupéfiants. La société de Redmond a commencé par vérifier la localisation de cette boîte mail et plus précisément où était le serveur sur lequel était hébergée la boîte mail contenant les informations intéressant les fédéraux. Découvrant qu'elle était à Dublin, en Irlande [8], Microsoft a présenté une



motion de rejet, arguant que le pouvoir judiciaire et extra-judiciaire des fédéraux s'arrêtait aux frontières américaines. Elle explique également que n'importe qui peut donner n'importe quelle information, qu'elle n'a pas à regarder ce que font les utilisateurs et qu'avant toute action de ce type, une convention de réciprocité entre les États doit exister. Le 25 avril 2014, le juge James Francis de la Cour du District Sud de New York répond qu'en vertu du *Stored Communication Act* (SCA), l'entreprise devait s'exécuter, peu importe la localisation.

Le *Stored Communication Act* (1986) est une loi fédérale de l'administration Reagan. À la base, ce texte se voulait protecteur [9] des communications électroniques, car il interdisait formellement à toute personne de révéler le contenu des dites communications à des entités non-gouvernementales. À l'origine, cette disposition s'appliquait surtout aux communications téléphoniques et transmissions radio avant d'être étendue aux fournisseurs de services Internet (ISP dans la langue pour *Internet Service Provider*).

Le point principal de ce texte réside dans la notion de contrôle qui induit nécessairement la possession. Pour le législateur, puis pour les juges – comme en attestent différentes affaires – l'entité qui contrôle techniquement les communications, possède les informations. Ce raisonnement n'est pas délirant dans le monde matériel et tangible : nous avons l'équivalent en droit français avec l'article 2276 du Code Civil qui énonce « en fait de meubles [10], la possession vaut titre. »

Mais dans le cadre des communications électroniques, on ne peut pas affirmer aussi fermement qu'on est dans du matériel, ni que l'on est dans de l'immatériel. De la même manière, tous les techniciens savent que ce n'est pas parce que l'on contrôle un bout de réseau qu'on le possède réellement. Je dispose d'un bout de serveur chez un hébergeur, qui en a le contrôle technique. Mais si j'héberge des éléments pouvant tomber sous le coup de la loi, j'en suis la seule responsable.

Il faut bien comprendre que le SCA permet aux autorités judiciaires et extra-judiciaires d'avoir accès à ce qu'elles cherchent sans avoir besoin d'en demander le consentement à la personne concernée. Dans le cas de Microsoft, la personne suspectée de trafics de stupéfiants [11].

Malgré l'argument de la compétence territoriale – car le SCA ne comporte aucune clause de compétence extraterritoriale – et du respect du quatrième amendement de la Constitution Américaine [12], la motion de rejet de Microsoft est rejetée.

Elle fait appel de la décision et l'affaire est réexaminée par un juge fédéral cette fois-ci : Loretta A. Preska [13].

La personnalité de cette magistrate n'est pas étrangère à la décision et pour comprendre son cheminement intellectuel, il faut revenir sur son parcours et sur certaines affaires emblématiques. Comme vous le savez certainement, la Cour Suprême américaine est au sommet du pouvoir judiciaire. Si on devait faire

une analogie, on pourrait dire qu'elle est à la fois une Cour de Cassation, car elle juge en dernier ressort des affaires judiciaires, un Conseil Constitutionnel, car elle a un pouvoir de contrôle constitutionnel et un Conseil d'État, car elle surveille les scrutins et donne le dernier mot en matière d'élections, notamment présidentielles [14]. Les magistrats y siégeant sont nommés par le POTUS [15], avec confirmation du Sénat. En 1992, Georges Senior Bush nomme Loretta A. Preska à la tête de la Cour du District Sud de New York. Fervente républicaine, la famille Bush essaiera de lui renvoyer la politesse en la nommant à la Cour Suprême en 2008, mais la nomination ne sera jamais inscrite à l'ordre du jour du Sénat, devenant ainsi caduque. Le moins que l'on puisse dire, en examinant les quelques affaires dont elle a eu la charge et qui ont été médiatisés, est qu'elle a une lecture parfois littérale de la Constitution Américaine. Elle autorise les groupes religieux à utiliser les écoles – sans contreparties financières – pour l'organisation des groupes de parole et de travail. Mais on la connaît surtout pour son travail sur l'affaire Strarfor. Alors que son époux travaille pour l'un des cabinets qui se trouve être plaignant et partie civile dans cette affaire d'intrusion et de compromission informatique, elle refuse de se récuser. Bien que ce type d'élément est tout à fait de nature à permettre un appel devant la Cour Suprême, elle incite, via le Procureur Général, Jeremy Hammond, membre d'AntiSec « balancé » par Hector Monsegur alias Sabu, à plaider coupable. Or, dans la procédure du plaider coupable américain, en échange d'une peine plus légère que si un procès avec un jury populaire s'était tenu, le coupable renonce à son droit de faire appel [16]. Elle négocie également avec Sabu : en échange d'une condamnation très symbolique de sept mois d'emprisonnement et l'abandon de toutes les autres charges retenues contre lui, il devra travailler en totale coopération avec les services fédéraux. La peine est ici qualifiée de symbolique, car elle correspond aux sept mois déjà effectués au titre de la détention préventive. Dans la mesure où l'accord bénéficie à l'accusé, là encore, il n'y a pas d'appel.

Rendons à César ce qui est à César, la suprématie de la Constitution Américaine sur toutes les autres normes, y compris internationales (dans l'esprit de la juge Preska), peut avoir de bons côtés comme dans l'affaire Bloomberg c/la Réserve Fédérale Américaine. Elle a donné raison à Bloomberg et vivement encouragé à la Réserve Fédérale à délivrer les informations demandées par le journal, en vertu du premier amendement [17]. Revenons à Microsoft et à son bras de fer avec Preska.

Sans réelle surprise, la juge Preska valide la décision du juge Francis le 31 juillet 2014. L'un des points les plus intéressants de son argumentation est le suivant : ce n'est pas la localisation des données qui est importante, mais qui peut techniquement les contrôler. Dans la mesure où l'entité de contrôle est Microsoft et qu'il s'agit d'une société américaine, la question de la territorialité ne se pose pas puisque la



souveraineté de l'État Irlandais n'est pas remise en question. En ce sens, elle suit une jurisprudence déjà bien établie que nous avons évoquée précédemment.

Pour Microsoft, cette partie de poker a un enjeu de taille : la restauration de la confiance des utilisateurs qu'elle a peu à peu perdue.

Outre les questions de sécurité des applications qu'elle propose, les révélations d'Edward Snowden ont fait très mal aux géants de l'informatique. Si Apple déclenche les passions et peut se permettre de ne pas s'en préoccuper outre mesure, ce n'est pas le cas de la firme de Redmond. Au-delà de la question de la confiance se pose la question des condamnations. Si Microsoft obtempère, rien n'indique qu'elle ne fera pas l'objet de poursuites en Irlande. Si elle refuse de s'exécuter, elle risque des poursuites aux États-Unis.

Dans la mesure où la demande d'accès aux informations est vraisemblablement un élément de preuve déterminant dans la mise en accusation des personnes concernées et donc d'une éventuelle demande d'extradition – que la République d'Irlande, si tant est que la personne concernée y soit effectivement, il n'est pas possible de référer aux conventions bilatérales d'entraide policière entre les États-Unis et la République d'Irlande. Quand bien même la personne

se trouverait sur le sol Européen, elle ne serait pas forcément extradée si elle risque la peine capitale [18].

Microsoft se retrouve donc, à son corps défendant, au milieu d'un problème de droit international assez intéressant et dans une position de protecteur de données personnelles, ce qui ne manque pas de faire sourire certains.

Au moment où ces lignes sont écrites, Microsoft persiste dans son refus et a clairement fait savoir qu'elle ferait appel de la décision. On peut parier sur le fait que la firme cherchera à épuiser toutes les voies de recours avant de s'exécuter [19].

Mais elle n'est pas seule dans ce combat. Un autre géant du Web a rencontré un problème similaire : Yahoo ! En 2007, la société s'est vu menacer d'une amende de 250 000\$/jour si elle refusait de fournir les informations demandées aux autorités américaines. Malgré les recours intentés, Yahoo ! n'a pas eu d'autre choix que de livrer les informations demandées [20], d'autant que les tribunaux ont estimé qu'il n'y avait aucune violation du quatrième amendement dans la mesure où les personnes concernées n'étaient pas des citoyens américains.

Une chose est sûre : sur le plan juridique, la bataille promet d'être intéressante.



INGÉNIEUR SÉCURITÉ (H/F)

ASSUREZ LA SÉCURITÉ INFORMATIQUE DU LEADER FRANÇAIS DU POKER !

- DÉJOUER LES TENTATIVES D'INTRUSION
- TESTEZ ET CHOISISSEZ LES DISPOSITIFS TECHNIQUES DE SÉCURITÉ LES PLUS APPROPRIÉS
- RÉALISEZ EN CONTINU DES AUDITS DE L'INFRASTRUCTURE ET DES APPLICATIFS
- PRÉVEZ LES FUITES D'INFORMATION
- ADAPTEZ LA POLITIQUE SSI EN FONCTION DE L'ÉVOLUTION DES MENACES

RELEVEZ DES DÉFIS TECHNIQUES VARIÉS

- METTEZ EN PLACE UN NOUVEAU SYSTÈME DE DÉTECTION EN TEMPS RÉEL DES ATTAQUES
- AMÉLIOREZ LES SYSTÈMES EXISTANTS D'AUTHENTIFICATION ET DE CONTRÔLE DES ACCÈS
- AUTOMATISEZ LES AUDITS

VOTRE PROFIL

- BAC +5 - SÉCURITÉ/SYSTÈMES/RÉSEAUX - 1 AN D'EXPÉRIENCE MINIMUM
- PASSION POUR LA DÉCOUVERTE DE FAILLES DE SÉCURITÉ ET LEUR RÉSOLUTION
- EXPERTISE DANS LES ENVIRONNEMENTS LINUX ET WINDOWS
- À L'AISE AVEC **OWASP/CRYPTO/RE/FW/GPO/RADIUS/IDS**

POURQUOI NOUS REJOINDRE ?

CHALLENGE - JEUX D'ARGENT - SÉCURITÉ - ENJEU CRITIQUE
OPÉRATIONNEL - **TECHNIQUE** - **FULL STACK** - POLYVALENCE - DÉCISIF
PARIS - SALLE DE SPORT - **BUREAUX DE OUF** - FREE DRINKS

RETROUVEZ NOUS AUSSI À LA NSC
DU 19 AU 21 NOVEMBRE 2014, AU W OPEN BAR !

JE SUIS
CHAUD

jobs@winamax.fr





3 Et s'il était déjà trop tard ?

Dans un monde idéal, l'accès aux informations électronique serait restreint à un personnel spécifique et le client en serait informé. Dans la réalité, personne ne peut savoir qui, quand, comment et pourquoi une personne se retrouve à explorer ces informations. C'est le nœud du problème : il n'existe aucune garantie de sécurité, au sens classique du terme. Un client ne peut se voir assurer que les informations qu'il va mettre dans son coin de nuage resteront intègres, disponibles et confidentielles. Le mauvais épisode de l'iCloud en a été une très bonne démonstration : à partir du moment où un utilisateur ne contrôle pas d'un bout à l'autre la chaîne, les données peuvent tomber entre de mauvaises mains.

À ce stade s'opère une sélection naturelle : ne rien mettre dans le Cloud (quel qu'il soit), s'auto-héberger, ne pas avoir de smartphones ni de tablettes. Rares sont les personnes qui ont le temps de le faire, les compétences pour y arriver ainsi que les ressources nécessaires et la possibilité professionnelle. Imaginez un cadre qui refuserait un smartphone, qui demanderait à sa DSI la liste complète des prestataires informatiques, qui imposerait des solutions de chiffrement à l'ensemble de ses clients. On peut aussi essayer d'en mettre le moins possible de façon à ce que lorsqu'un incident de sécurité intervient, les données compromises ne se transforment pas en désastre pour l'utilisateur. Là encore, cela demande une certaine gymnastique intellectuelle à laquelle même les plus aguerri(e)s ne se livrent pas toujours. Enfin, il reste une dernière catégorie : celle des personnes qui vont continuer à utiliser des services de Cloud Computing avec une conscience plus ou moins prononcée des risques qu'elles encourent. La véritable fracture numérique ne réside pas tant dans le matériel que l'on a à sa disposition, mais dans la prise de conscience de l'emprise du dit matériel sur notre propre individualité et intimité. ■

■ Références

- [1] On aura compris que l'argenterie représente les données stockées dans cette métaphore.
- [2] Ou plus simplement : dans quel cas le client ne peut pas se retourner contre le prestataire si ce dernier fait une erreur ?
- [3] On a déjà des supports payants.
- [4] On notera la passion de l'auteur pour les films catastrophes de séries B/Z.
- [5] Voir TGI Angers du 11 mars 1998 sur la responsabilité d'EDF.
- [6] Voir la référence en question : <http://www.cnil.fr/les-themes/technologies/cloud-computing/>

- [7] La décision du 25 avril 2014 peut être consultée ici : <https://www.documentcloud.org/documents/1149373-in-re-matter-of-warrant.html>
- [8] Comme un grand nombre de villes européennes, la capitale irlandaise ayant un homonyme aux États-Unis, dans l'état de l'Ohio, la précision n'est pas superflue.
- [9] Le terme de protecteur peut paraître curieux, mais il convient de ne pas oublier les interprétations des Républicains concernant les libertés fondamentales, surtout en période de guerre froide.
- [10] Le meuble n'est pas nécessairement le buffet de la grand-mère. En droit, le terme de meuble désigne tout objet qui n'est pas fixé durablement au sol.
- [11] Une analyse assez intéressante est disponible ici : <http://www.fedbar.org/Federal-Lawyer-Magazine/2009/The-Federal-Lawyer-November-and-December-2009/Features/Consent-and-Discovery-Under-the-Stored-Communications-Act.aspx?FT=.pdf>
- [12] « Le droit des citoyens d'être garantis dans leur personne, domicile, papiers et effets, contre les perquisitions et saisies non motivées ne sera pas violé, et aucun mandat ne sera délivré, si ce n'est sur présomption sérieuse, corroborée par serment ou affirmation, ni sans qu'il décrive particulièrement le lieu à fouiller et les personnes ou les choses à saisir. »
- [13] Sa fiche Wikipedia est disponible ici : http://en.wikipedia.org/wiki/Loretta_A._Preska
- [14] Voir la première élection de Georges W. Bush.
- [15] POTUS : *President Of The United States*.
- [16] Il peut y avoir une réouverture du procès et abandon des charges si de nouvelles preuves peuvent disculper une personne condamnée. En l'espèce, cela ne semble pas pouvoir être le cas.
- [17] Voir un résumé de l'affaire ici : http://en.wikipedia.org/wiki/Bloomberg_L.P._v._Board_of_Governors_of_the_Federal_Reserve_System
- [18] Voir la convention d'extradition entre les États-Unis et l'Union Européenne : http://europa.eu/legislation_summaries/justice_freedom_security/judicial_cooperation_in_criminal_matters/jl0053_en.htm
- [19] Voir cet article par exemple : http://www.channelregister.co.uk/2014/09/10/microsoft_contempt_of_court/
- [20] Lire : http://www.washingtonpost.com/business/technology/us-threatened-massive-fine-to-force-yahoo-to-release-data/2014/09/11/38a7f69e-39e8-11e4-9c9f-ebb47272e40e_story.html

NE MANQUEZ PAS GNU/LINUX MAGAZINE N°176 !

FOCUS :

QUEL EST LE LANGAGE LE PLUS PERFORMANT ?



**OU COMMENT
LE CHOIX DU
LANGAGE VA
VRAIMENT
IMPACTER VOS
ALGORITHMES !**

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





GÉNÉRATION AUTOMATIQUE DE RÈGLES SNORT POUR LA DÉTECTION DES RÉSEAUX FAST-FLUX

Jean-Philippe Monteuis – jeanphilippe.monteuis@gmail.com

Julien Huor – julien.huor@gmail.com

Marc Hofmann – 1marc.hofmann@gmail.com

Valentin Beauvisage – valentin.beauvisage@gmail.com

Université de Technologie de Troyes

Rida Khatoun – rida.khatoun@telecom-paristech.fr

Telecom ParisTech

mots-clés : FAST-FLUX / RÉSEAU / SÉCURITÉ / BOTNET / DÉTECTION / SNORT

Les réseaux Fast-Flux sont une menace parfois méconnue, mais malgré tout présente pour les utilisateurs de l'Internet. Si elle est rare et à faible impact au début de sa croissance, elle reste une nuisance qui peut s'avérer désastreuse par la suite comme le phishing, les malwares ou autres attaques de grandes ampleurs sur des cibles de choix. Toutes ces raisons font que les réseaux Fast-Flux doivent être pris au sérieux. Cependant, il est très difficile à l'heure actuelle de trouver des solutions efficaces pour différencier un domaine Fast-Flux d'un domaine sain. C'est pourquoi nous proposons une méthode de détection dans le cas d'une connexion d'un utilisateur vers un domaine Fast-Flux.

1 Qu'est-ce qu'un réseau Fast-Flux ?

1.1 Définition

Le Fast-Flux (FF) est le détournement d'une technique DNS légitime, le *load-balancing*. La technique des FF a pour vocation d'optimiser la durée de vie de certains types de réseaux. En effet, elle est utilisée par des sites Internet malveillants pour masquer leurs activités malicieuses (phishing, spamming, *malware delivering*, pédopornographie...).

Les réseaux FF se basent sur l'utilisation de machines infectées appelées botnets, pouvant être des ordinateurs de particuliers, d'entreprises ou même des serveurs. Ces botnets sont regroupés en un réseau dont le contrôle est assuré par un serveur central appelé *Command and Control* (C&C). Un tel réseau sera appelé *Fast-Flux Service Network* (FFSN).

Les botnets étaient à l'origine utilisés sur des serveurs *Internet Relay Chat* (IRC). Un script en cours d'exécution sur une machine infectée permet de prendre le contrôle d'un salon via une présence continue, d'y diffuser du contenu et de récupérer des données sur les clients de ce dernier. L'automatisation des actions et la rapidité de propagation des données par les botnets IRC ont donné lieu au concept de machines zombies, utilisées actuellement dans les FFSN.

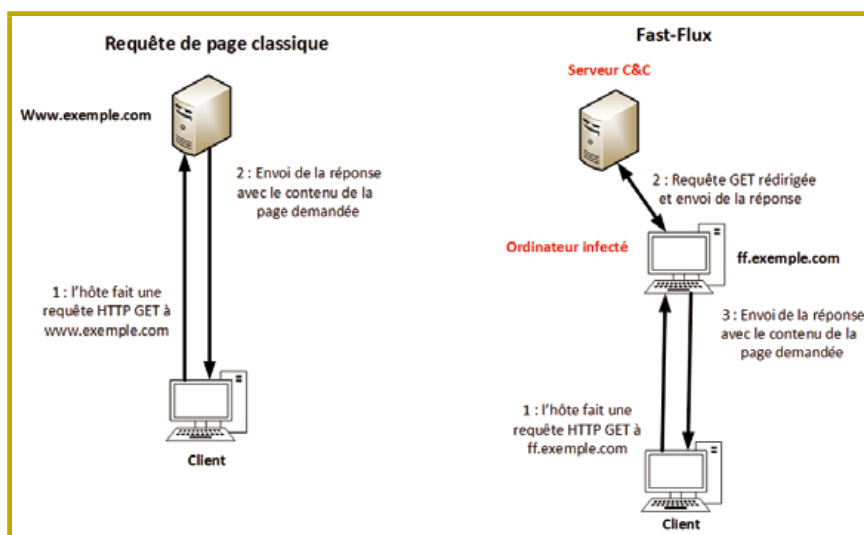


Fig.1 : Schéma de fonctionnement d'un Fast-Flux.

1.2 Single Flux et Double Flux

Il existe deux types de FFNS : ceux basés sur un mécanisme Single Flux et ceux basés sur un mécanisme Double Flux.

Dans un FFNS Single Flux, les botnets servent de « proxy inversés » au site malveillant (seule l'adresse IP du serveur C&C est connue). Un serveur DNS, contrôlé via un serveur C&C, redirige les clients vers les bots, ce qui permet également de répartir la charge entre ces derniers.

Un FFNS Double Flux, en plus d'utiliser des bots comme proxies pour des sites malveillants, utilise certains de ces derniers comme proxies DNS. Les requêtes envoyées vers ces « faux » DNS sont transmises au(x) serveur(s) C&C, qui choisit alors les bots vers lesquels rediriger les trafics. Un FFNS Double Flux possède donc, en plus du Single Flux, les caractéristiques suivantes :

- Un DNS à plusieurs niveaux ;
- De changer fréquemment l'association NS-IP auprès du registraire du nom de domaine.

De par sa topologie et ses caractéristiques, il est donc plus difficile de détecter le pirate d'un réseau Fast-Flux. Enfin, certains registraires de domaines prêtent peu d'attention aux activités de leurs clients ce qui ne facilite pas la lutte contre les réseaux Fast-Flux.

1.3 Mode de déploiement

Deux modèles d'architecture existent pour les FFSN : centralisé ou décentralisé.

Dans un FFSN centralisé, tous les botnets sont contrôlés au moyen d'un serveur C&C. Le contrôle se fait en direct ou à l'aide d'un nœud supérieur dans la hiérarchie, qui servira de relais au serveur C&C. Historiquement, le protocole HTTP sert de vecteur de communication, mais on trouve à présent des FFSN basés sur le protocole HTTPS et utilisant le réseau TOR (*The Onion Router*). Ce changement est autant lié aux faiblesses du premier, comme l'absence de chiffrement qui

facilite la détection, qu'aux forces des suivants, par exemple le chiffrement ou encore « l'anonymisation ». Le modèle centralisé présente un *single point of failure* au travers du serveur C&C, mais cette faiblesse est quelque peu compensée par l'anonymat apporté par le réseau TOR.

Les FFSN décentralisés sont eux basés sur des protocoles P2P (*Peer-to-Peer*), ce qui les rend très robustes. Chaque nœud pouvant jouer le rôle d'un serveur C&C et/ou d'un bot, la destruction de l'un d'entre eux n'affecte pas le fonctionnement global d'un FFSN. La tendance à bloquer le protocole P2P en entreprise, principalement utilisé pour réaliser des téléchargements illégaux, remet cependant en cause certaines utilisations de ce modèle.

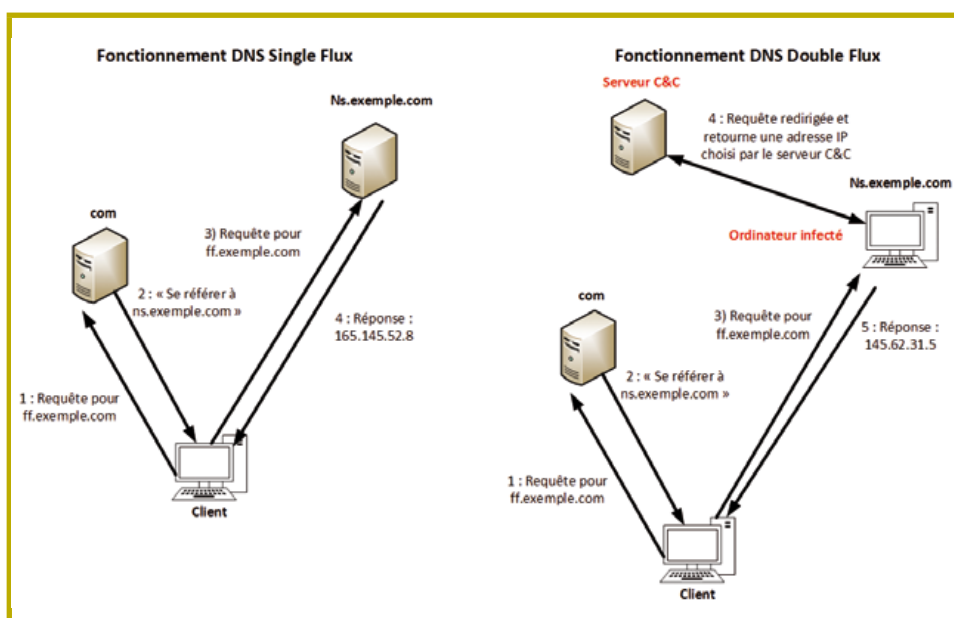


Fig.2 : Fonctionnement DNS du Single Flux et du Double Flux.

1.4 Utilisation des Fast-Flux

Les FFSN sont principalement utilisés pour le spam, le phishing et la distribution de malwares au travers de sites malicieux. Ils exploitent des techniques comme le *Round-Robin DNS* et s'inspirent du modèle CDN (*Content Delivery Network*), pour passer inaperçus.

La durée de vie de tels réseaux dépendra de leur vocation. L'article traitant des comportements des FFSN [13], montre dans une étude que les FFSN utilisés dans le cadre du spam sont les plus pérennes, à l'inverse de ceux consacrés à la distribution de malwares par exemple. La taille joue également un rôle important dans la durée de vie des FFSN. Une taille importante permet de garantir une certaine persistance face à la destruction des botnets, mais aussi d'être moins détectable en disposant de plus de sources différentes. On notera cependant, toujours selon cette étude, que l'augmentation de la durée de vie par rapport à la taille atteint une asymptote. Le facteur taille ne permet pas de prolonger à l'infini la vie d'un FFSN.

2 Solutions théoriques

Dans cette partie, nous allons présenter plusieurs études de recherche dont le but était de détecter des domaines FF.

2.1 Détection via caractéristiques bot et DNS

La première solution de détection en temps réel des FFSN est proposée par Martinez-Bea et al [8]. L'objectif de cette solution est la détection en temps réel de services de FFSN. Elle se base sur deux méthodes existantes en les reprenant et en les combinant afin d'améliorer les résultats.

La première méthode [9] exploite les caractéristiques d'une requête DNS à un domaine (nombre d'IP pour un

domaine, nombre de pays différents...) et les compare à des valeurs seuils typiques des domaines FF. La seconde méthode [6] se base sur les facteurs intrinsèques aux botnets (latence du réseau, temps de chargement d'une page...) et les compare également à des valeurs seuils pour indiquer s'il s'agit d'un domaine FF.

Chacune de ces méthodes présentait des résultats avec soit beaucoup de faux négatifs, soit beaucoup de faux positifs. Ici, la solution combine ces deux méthodes pour outrepasser leurs limites.

Un test a été fait en comparant la détection de domaines FF des trois méthodes, voir le résultat dans Tableau 1.

Comme nous pouvons le voir, en combinant les deux méthodes, il y a une réduction non négligeable du nombre de faux positifs et faux négatifs. Cependant, on observe que la méthode n'est pas optimale avec la présence d'un faux positif.

Finalement, la force d'une des méthodes comblant la faiblesse de l'autre et vice versa, c'est par la combinaison de ces deux recherches que les auteurs ont créé un moyen de détection fiable des domaines FF.

2.2 Détection des Fast-Flux en temps réel

La seconde solution [12] est aussi une solution en temps réel proposée par le groupe de recherche Milcord.

Sa méthodologie repose sur 5 capteurs :

- 3 capteurs actifs :

- TTL (*Time to Live*) ;

Répertoire d'activité Fast-Flux ;

Répertoire d'empreintes.

- 2 capteurs passifs :

- Capteur DNS qui exploite les données DNS ;

- Détecteur analytique qui exploite des blacklists externes de FF.

	Proposition de Mc Grath	Proposition de Hsu	Solution 1 Martinez-Bea et son équipe
Vrai Positif	78	63	81
Faux Positif	4	9	1
Vrai Positif	77	72	80
Faux Positif	3	18	0

Tableau 1 : Résultats des tests.



La partie intéressante est au niveau de la partie active de la solution. La détection du TTL permet un premier filtrage grossier des noms de domaines. Un site sain a, en général, un TTL plus élevé qu'un FFSN par exemple, 300 pour Google jusqu'à 3600 pour Microsoft. Cependant certains FFSN peuvent avoir un TTL supérieur à 300.

Le répertoire d'activité FF est un capteur qui va enregistrer l'activité d'un FF pendant 24h et va permettre d'affiner le résultat de la détection par TTL.

Le capteur d'activité va réagir lorsque les FFSN arrêtent de fonctionner pendant quelques heures ou jours afin d'éviter d'être détectés. Ce comportement n'est pas visible sur un nom de domaine sain tel que google.com, car il doit être tout le temps disponible. C'est cette absence d'activité qui peut être un facteur indiquant la présence d'un FF.

Le répertoire d'empreintes associe les adresses IP d'un domaine avec l'*Autonomous System Number* (ASN), caractéristique dépendant du Fournisseur d'Accès à Internet. Un domaine FF se disperse plus facilement qu'un domaine sain, car il possède les adresses de machines contaminées partout dans le monde. C'est en calculant l'indice de dispersion et en le comparant à un seuil, choisi arbitrairement, que le domaine sera répertorié sain ou non.

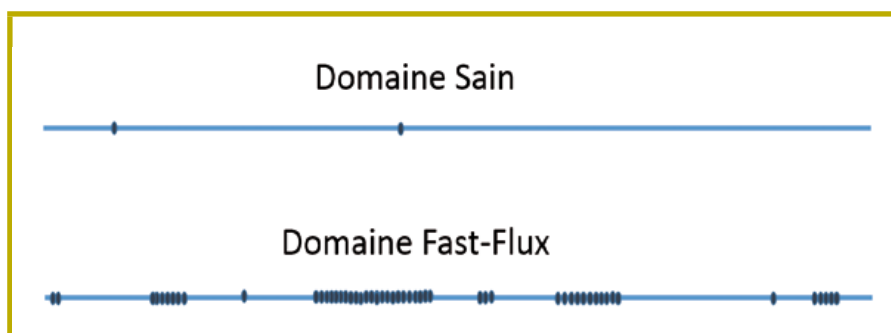


Fig.3 : Empreintes d'un domaine Fast-Flux et d'un domaine sain.

Pour conclure, cette solution est capable de montrer un taux de détection moyen de 96% sur une fenêtre de tests de 10 minutes et peut atteindre 100% sur des durées supérieures à 10 minutes.

2.3 Détection par data-mining

Nous avons étudié un article de recherche [11] comparant trois techniques de data-mining, utilisées dans le cadre de la détection de FFSN, à savoir :

- Régression linéaire ;
- Classification naïve bayésienne ;
- Méthode des K plus proches voisins.

Chacune de ces techniques a été testée dans un environnement réaliste (60 000 domaines sains pour

100 domaines FF). Il en ressort que la technique la plus efficace est celle des K plus proches voisins. Elle possède en moyenne des résultats supérieurs à ceux des deux autres techniques.

Il faudra cependant noter que cette solution n'est pas une méthode de détection a priori de FF, mais a posteriori. Ce qui signifie que l'utilisateur doit se connecter sur un domaine pour que la solution puisse donner son verdict. De plus, cette option possède certaines limites :

- La similitude des caractéristiques d'un domaine sain par rapport à un jeune domaine FF ;
- L'évolution des technologies qui nécessite une mise à jour constante de la solution.

2.4 Détection par géolocalisation

Cette méthode repose sur les principes de la géolocalisation pour détecter les domaines basés sur des FFSN. Nous avons étudié deux travaux de recherche [7] [10] proposant une implémentation concrète :

- *Spatial Snapshot Mechanism for Delay-free Detection* ;

- *Localized Spatial Geolocation Detection*.

Ces solutions sont « Delay-free ». Elles se basent sur une seule réponse DNS pour classer un domaine. Elles utilisent la position géographique des bots, diverses métriques spatiales et des bases de données externes pour fonctionner.

Les principales métriques spatiales employées sont :

- La dispersion : la présence dans différents pays ;
- Le degré de relation spatiale des services : distance entre un hôte et le fournisseur de service.

Par exemple, un réseau FF aura tendance à avoir des bots présents dans différents pays, éloignés de leur fournisseur de services et dans des *Autonomous System* (AS) très différents.

La détection se décompose en quatre étapes :

- Extraction des données d'une réponse DNS ;
- Récupération des informations spatiales ;
- Génération des métriques ;
- Classification du domaine analysé (FFSN ou non).

En plus de la rapidité de détection, cette méthode présente l'avantage de ne pas se laisser tromper par les FFSN se camouflant en réseaux bénins. En effet,

elle se base peu sur les caractéristiques réseaux des FFSN (TTL, nombre de bots, nombre d'adresses IP...) et les FFSN ne peuvent fausser leurs caractéristiques géographiques.

Cependant, en l'absence de données spatiales associées aux domaines scannés, cette méthode se trouve dans l'incapacité totale de fonctionner.

2.5 Limites des solutions de détections actuelles

Les FFSN ont tendance à s'adapter aux techniques de détection, notamment ceux en temps réel, afin de maximiser leur durée de vie et leur efficacité. Les réseaux malveillants « copient » les caractéristiques de réseaux légitimes, exploitant notamment les techniques du RR-DNS et des CDN citées précédemment. Les moyens de détections deviennent ainsi moins précis et peuvent voir leur période de collecte d'informations sensiblement rallongée. Il est important de noter que, même si en théorie il est possible de se rapprocher de valeurs seuils pour se faire passer pour un domaine sain, en pratique cela reste extrêmement difficile et complexe à réaliser.

Certaines caractéristiques propres aux FF restent cependant difficilement camouflables. Par exemple, ces réseaux souffrent d'une dispersion géographique élevée en raison de la nature même des botnets. On peut aussi remarquer un important turnover au niveau des adresses IP liées à la suppression de nœuds, ainsi que des adresses largement réparties et non successives.

3 Approche proposée pour la détection via les blacklists

3.1 Principe

Notre outil a pour objectif de détecter des connexions d'utilisateurs d'un réseau à des domaines FF. Cet outil est implémentable rapidement et facilement afin de permettre son utilisation par toutes les entreprises, contrairement aux solutions théoriques vues précédemment. Son objectif diffère cependant de ces dernières solutions et leur est complémentaire. En effet, il vise la détection des connexions sortantes vers des sites hébergés sur des réseaux FF, plutôt que leur détection en soit.

Pour fonctionner, notre outil est composé de trois modules :

- L'IDS/IPS Snort ;

- Un script Bash ;
- Une blacklist.

L'IDS Snort est au cœur de la solution de détection. Cet outil de détection d'intrusion a été retenu, car il est librement accessible à toutes les entreprises (distribué sous licence GPL). Il jouit également d'une communauté active et d'une certaine popularité, ce qui se traduit par une documentation facilement accessible sur Internet.

Le script joue un rôle central dans l'outil, c'est lui qui va permettre de créer la blacklist et de l'intégrer à l'IDS de manière automatique. Le langage Bash fut choisi, car il permet l'exploitation du script par un grand nombre de distributions Linux.

La blacklist se compose, quant à elle, de domaines reconnus comme étant malveillants (FFSN, distribution de malwares...) par des sites spécialisés de confiance. Ces derniers distribuent des listes de domaines et d'adresses identifiées pour leurs activités frauduleuses, dangereuses ou comme infectées par des botnets. Les sites Internet choisis sont Malwaresdomains, Palevotracker, Spyeyetracker et Zeustracker.

L'installation de cette solution a été réalisée sur un Raspberry Pi utilisé lors du développement et des tests présentés plus tard dans cet article.

3.2 Algorithme

3.2.1 Le script

Pour chacune des bases de données, nous obtenons une liste de domaines malveillants, incluant des domaines FF.

Le script Bash va tout d'abord extraire ces données afin d'obtenir les URLs de ces domaines FF en « parsant » chacune des listes des sites précédents.

Les listes des domaines sont récupérées via la commande **wget** et sont ajoutées dans un fichier temporaire. Ensuite, la fonction **addDomain** ajoute le contenu de ce fichier temporaire dans un autre fichier comportant l'ensemble des domaines trouvés sur les quatre sites cités précédemment, et ce uniquement si le domaine ne s'y trouve pas déjà.

```
addDomain() { #fichier liste domaine d'un tracker en particulier
  local RESULT=""
  while read line
  do
    RESULT=`grep "$line" /$REP_TMP/$EXT_TMP"liste_domaines_TOT`
    if [ "$RESULT" == "" ]
    then
      echo "$line" >> /$REP_TMP/$EXT_TMP"liste_domaines_TOT"
    fi
  done < "$1"
}
```

Une fois la liste de ces domaines établie, le script génère une règle Snort par domaine et place ces dernières



dans un fichier **.rules** dédié. Une sauvegarde de l'ancien fichier de règles est archivée en cas de problèmes.

```
create_rules() {
    local fic_rules="$REP_SNORT_RULES/black-liste_fast-flux.rules"
    local rule_id=$First_Rule_number
    if [ -f $fic_rules ]
    then
        cp $fic_rules $fic_rules.back
    fi
    echo "# File automatically generated by slowing_flux" > $fic_rules
    echo "# last updated : " $(date -u) >> $fic_rules
    while read line
    do
        echo 'alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS
(msg:"Fast-flux    domain : '$line'; content:"Host: "; content:
"$line"|0d 0a|"; nocase;    content:"GET"; http_method;
classtype:fast-flux; sid:'$rule_id';rev:1;)' >>    $fic_rules
        rule_id=$((rule_id+1))
    done < /$REP_TMP/$EXT_TMP/"liste_domaines_TOT"
}
```

Une fois l'opération terminée, le script se charge de relancer Snort afin que les nouvelles règles puissent être prises en compte. Pour cela, le script retrouve le PID du processus Snort en cours d'exécution et le réinitialise à l'aide de la commande suivante, qui oblige la relecture de la configuration :

```
kill -SIGHUP [PID_SNORT]
```

Enfin, à l'aide de la commande **crontab** permettant l'exécution automatique de tâches, nous pouvons paramétrer le script afin qu'il se lance régulièrement de manière autonome afin de garder les bases à jour.

3.2.2 La règle Snort

L'objectif de la règle est, qu'une fois en place, Snort puisse détecter la connexion d'utilisateurs de notre réseau à des domaines FF connus et générer une alerte. Voici un exemple de règle dont on se servira pour la suite pour expliquer comment cela fonctionne avec le domaine factice FF **xyz.com** :

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS
(msg:"Fast-Flux domain : xyz.com"; content:"Host: ";
content:"xyz.com|0d 0a|"; nocase; content:"GET"; http_method;
classtype:fast-flux; sid:90000; rev:1;)
```

3.2.2.1 Variables d'entrées

Les variables d'entrées **\$HOME_NET**, **\$EXTERNAL** et **\$HTTP_PORTS** sont à définir au préalable dans le fichier de configuration de Snort pour le fonctionnement de l'IDS.

La variable **\$HOME_NET** doit contenir l'adresse réseau de l'entreprise ou le sous-réseau que l'on souhaite surveiller. La variable **\$EXTERNAL** doit

contenir l'adresse des réseaux de destination des flux que l'on souhaite également surveiller, étant donné que les domaines FF sont présents sur Internet, il est préférable de la configurer en mettant l'adresse 0.0.0.0 afin d'indiquer à Snort que l'on souhaite surveiller tous les paquets sortants. Pour terminer, la variable **\$HTTP_PORTS** détermine les ports de destination des flux que l'on souhaite surveiller. Les flux ici étant des flux web, il faut la définir pour monitorer les protocoles HTTP et HTTPS.

3.2.2.2 L'objet recherché

Les champs de la règle permettant à Snort de détecter la connexion d'un utilisateur à un domaine FF sont **content** et **http_method**. En effet, **content:"GET"** et **http_method** permettent de spécifier à Snort de rechercher dans les paquets toutes les requêtes de type HTTP GET parmi celles sortantes du réseau. Les deux autres instructions **content** (« **Host** » et « **mondomain.ru** ») sont présentes afin de rechercher le champ comportant l'adresse URL dans un paquet HTTP GET sortant émis par un client. Pour mieux comprendre la règle, voici une capture d'écran d'une requête HTTP GET sur Wireshark (voir Figure, page suivante).

Un événement

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN

Normandie Université

UNIVERSITÉ DE ROUEN

Master SSI

Sponsor

MISC

Multi-System & Information Security Cookbook

4^e édition

JOURNÉE SÉCURITÉ DES SYSTÈMES D'INFORMATIONS

13 novembre 2014

à l'Université de Rouen, Campus du Madrillet
76800 SAINT-ÉTIENNE-DU-ROUVRAY

Conférences ouvertes à tous
Entrée gratuite
Inscription en ligne

<http://jssl.insa-rouen.fr>

Cybersécurité, cloud, contrôle d'accès, logiciels malveillants, DoS, cryptographie

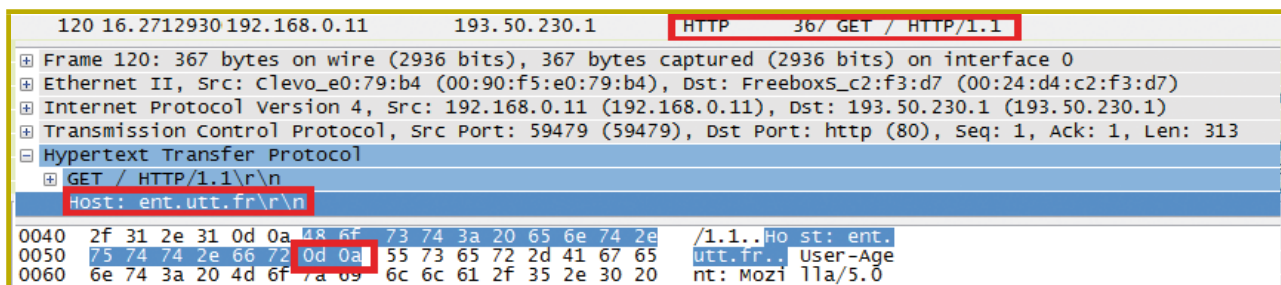


Fig.4 : Requête HTTP GET sur Wireshark.

Comme nous pouvons le voir, le paquet HTTP GET contient le champ suivant : **Host: domaine\r\n**. C'est cela qu'exploite Snort pour générer les alertes avec les mots clés indiqués précédemment. À noter que **\r\n** correspond à **|0d 0a|** en hexadécimal, ce qui correspond bien à l'une des instructions **content** que nous avons. Enfin, nous avons séparé le **content "Host"** et le **content** de l'URL afin d'éviter l'apparition de faux positifs, c'est-à-dire un domaine légitime qui a été identifié par Snort en tant que domaine FF. Cela peut être, par exemple, une recherche Google avec l'URL d'un domaine FF.

3.2.2.3 Les champs msg et classtype

Le champ **msg** de la règle permet l'affichage d'informations concernant cette dernière dans les logs et la console. Notre script remplit automatiquement ce champ avec la valeur **Fast-Flux domain : <nom du domaine>** en se basant sur la blacklist créée précédemment.

De plus, nous avons créé au préalable dans Snort un type de classe particulier pour les alertes Fast-Flux afin d'améliorer leur visibilité. Ce type est référencé dans la règle via le champ **classtype: fast-flux**. C'est également dans cette classe que l'on définit le niveau de criticité d'une alerte. Les FF étant potentiellement dangereux, nous avons décidé de donner à cette classe une priorité 1, c'est-à-dire la plus élevée.

Ces deux champs permettent de retrouver plus facilement des informations de détection de FF dans les fichiers de logs ou dans une éventuelle Interface graphique.

3.2.2.4 Conclusion

Voici un exemple d'alerte que Snort génère suite à la connexion d'un utilisateur au domaine FF factice **xyz.com** :

```
08/08-17:06:25.599829 [**] [1:90055:1] Fast-flux domain : xyz.
com[**] [Classification: fast-flux domain detection] [Priority: 1]
{TCP} 10.23.0.6:35653 -> 50.28.88.109:80
```

Nous avons donc un outil capable de récupérer une liste de domaines FF sur plusieurs bases de données connues, de mettre à jour les règles de Snort et de les appliquer en le réinitialisant de manière automatique.

3.3 Expérimentations et résultats

Afin de tester l'outil, nous avons utilisé une liste de sites Internet regroupant deux types de sujets. Le premier type correspond à 1000 domaines sains, c'est-à-dire des sites Internet tout à fait légitimes comme Google. Ces derniers ont été récupérés sur le site Alexa **[14]** qui référence les sites les plus visités sur Internet. Le second type correspond aux domaines FF. Lors de l'expérimentation, nous avons récupéré une liste de 177 domaines FF obtenus sur les quatre sites fournissant les bases de données, cités précédemment.

	Fast-Flux	Sain
Nombres de domaines	177	1000

Tableau 2 : Population utilisée pour l'expérimentation.

Nous avons également utilisé un poste client qui va se connecter aux sites Internet de la liste. Il est branché à un Hub sur lequel est connecté notre IDS Snort, afin que ce dernier puisse analyser les connexions de l'utilisateur.

Enfin, nous avons utilisé la commande **wget** dans un script pour automatiser les connexions utilisateurs vers les sites Internet.

À l'issue de cette étape, nous obtenons les résultats dans le fichier « alert » généré par Snort. Dans ce fichier sont référencées les alertes émises par Snort lorsqu'un paquet HTTP GET a été identifié positivement par une règle Snort. À l'aide d'un autre script, nous avons pu déterminer quels sites ont été reconnus ou non dans les alertes

À l'issue de plusieurs tests, nous avons obtenu toujours les mêmes résultats, à savoir : voir tableau ci-dessus.

	Vrai positif	Vrai négatif	Faux positif	Faux négatif
Domaines détectés	177	1000	0	0

Tableau 3 : Résultats de l'expérimentation.



Cette expérience démontre deux choses : que les règles sont bien toutes créées automatiquement, mais aussi que ces dernières ont relevé tous les sites malveillants, sans logger de connexions à destination d'un site légitime.

Conclusion

Nous avons pu constater que les solutions apportées par les chercheurs sont difficiles à mettre en œuvre par une entreprise et peu nombreuses.

De notre côté, nous voulions créer une solution simple, économique et rapide à mettre en place. Son champ d'action dépend de bases de données, sur lesquelles nous n'avons aucun contrôle, mais cet outil garantit une détection fiable des connexions vers les domaines FF blacklistés.

Cette solution permet donc, après détection, d'alerter l'administrateur afin qu'il puisse réagir rapidement pour effectuer toutes les actions nécessaires, comme isoler les machines potentiellement infectées du réseau.

En revanche, seules les communications réalisées par le biais du protocole HTTP sont prises en charge. Pour les communications en HTTPS, il est possible de se tourner vers une solution de blacklist basée sur un proxy. Le proxy sera placé en coupure pour détecter quel est le domaine visé par la requête d'un client.

Enfin, cette solution est complémentaire et ne remplace en aucun lieu les autres équipements (FW, AV, Proxy...) ni les procédures mises en place pour la sécurisation d'un réseau et de ses équipements. ■

Bibliographie

- [1] **Malwaredomains.** <http://malwaredomains.lehigh.edu/files/domains.txt>
- [2] **MaxMind - GeoIP | Base de données de localisation d'adresses IP.** [https://www.maxmind.com/fr/geolocation landing](https://www.maxmind.com/fr/geolocation%20landing)
- [3] **Palevotracker.** <https://palevotracker.abuse.ch/blocklists.php?download=domainblocklist>
- [4] **Spyeyetracker.** <https://spyeyetracker.abuse.ch/blocklist.php?download=domainblocklist>
- [5] **Zeustracker.** <https://zeustracker.abuse.ch/blocklist.php?download=domainblocklist>
- [6] **Ching-Hsiang Hsu, Chun-Ying Huang, and Kuan-Ta Chen.** *Fast-flux bot detection in real time.* In Somesh Jha, Robin Sommer, and Christian Kreibich, editors, *Recent Advances in Intrusion Detection*, number 6307 in *Lecture Notes in Computer Science*, pages 464–483. Springer Berlin Heidelberg, 2010.

- [7] **Si-Yu Huang, Ching-Hao Mao, and Hahn-Ming Lee.** *Fast-flux service network detection based on spatial snapshot mechanism for delay-free detection.* In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, ASIACCS '10*, page 101–111. ACM, 2010.
- [8] **S. Martinez-Bea, S. Castillo-Perez, and J. Garcia-Alfaro.** *Real-time malicious fast-flux detection using DNS and bot related features.* In *2013 Eleventh Annual International Conference on Privacy, Security and Trust (PST)*, pages 369–372.
- [9] **D.K. McGrath, A. Kalafut, and M. Gupta.** *Phishing infrastructure fluxes all the way.* 7(5) :21–28, sept.-oct. 2009.
- [10] **Horng-Tzer Wang, Ching-Hao Mao, Kuo-Ping Wu, and Hahn-Ming Lee.** *Real-time fast-flux identification via localized spatial geolocation detection.* In *Computer Software and Applications Conference (COMPSAC), 2012 IEEE 36th Annual*, pages 244–252.
- [11] **Jiayan Wu, Liwei Zhang, Jian Liang, Sheng Qu, and Zhiqiang Ni.** *A comparative study for fast-flux service networks detection.* In *2010 Sixth International Conference on Networked Computing and Advanced Information Management (NCM)*, pages 346–350.
- [12] **Caglayan A., Toothaker M., Drapeau D., Burke D., Eaton G.** « *Real-Time Detection of Fast Flux Service Networks* ». In : *Conference For Homeland Security, 2009. CATCH '09. Cybersecurity Applications Technology [En ligne]. Conference For Homeland Security, 2009. CATCH '09. Cybersecurity Applications Technology.* [s.l.] : [s.n.], 2009. p. 285–292. Disponible sur : <http://dx.doi.org/10.1109/CATCH.2009.44>
- [13] **Alper Caglayan, Mike Toothaker, Dan Drapeau, Dustin Burke, and Gerry Eaton.** *Behavioral Patterns of Fast Flux Service Networks.* In *Proceedings of the 2010 43rd Hawaii International Conference on System Sciences (HICSS '10)*. IEEE Computer Society, Washington, DC, USA, 1–9. DOI=10.1109/HICSS.2010.81 <http://dx.doi.org/10.1109/HICSS.2010.81>
- [14] **Alexa Top Global Sites.** <http://www.alexa.com/topsites>

UNE ÉTUDE DES BRUTEFORCE SSH

Éric Leblond – eleblond@stamus-networks.com

Co-fondateur de Stamus Networks, core développeur Suricata

mots-clés : SSH / SCAN / SURICATA / NETFILTER / ELASTICSEARCH

Bien que les attaques en bruteforce des SSH fassent partie du bruit blanc du trafic réseau des systèmes connectés, leur étude révèle des profils d'attaquants à l'hétérogénéité intrigante. Leur analyse approfondie grâce à un pot de miel dédié révèle des informations intéressantes sur les agresseurs.

1 Qu'est-ce que c'est ?

Les attaques dites « bruteforce SSH » sont des tentatives de connexions SSH effectuant une succession d'essais pour découvrir un couple utilisateur/mot de passe valide afin de prendre le contrôle de la machine. Il s'agit d'une attaque très répandue et toute machine exposée sur Internet se verra attaquer plusieurs fois par jour.

La fréquence de ces attaques a entraîné l'apparition de diverses méthodes de protection telle que l'utilisation d'un port non standard pour le serveur SSH. Une autre méthode très répandue est l'utilisation d'outils comme Fail2ban [**FAIL2BAN**] qui analyse les journaux SSH pour trouver les adresses IP responsables de trop nombreux échecs de connexions et modifie la configuration du pare-feu pour bloquer les tentatives ultérieures.

2 Collecte des informations

2.1 Présentation du point de collecte

Cet article présente l'analyse de données collectées sur un ADSL résidentiel français dont l'adresse IPv4 n'est associable à aucun service public. Il s'agit donc d'une adresse anonyme perdue au milieu des 2³² autres. Des redirections de ports sont utilisées sur le modem routeur pour envoyer le trafic de certains ports vers la machine cible.

2.2 Journaux systèmes

Les premières sources d'information pour récupérer des données sur les bruteforce SSH sont bien entendu les journaux syslog et en particulier ceux générés par le démon SSH :

```
Aug 24 19:55:10 tiger2 sshd[2525]: Failed password for miscmag from 1.2.3.4 port 48459 ssh2
```

Ces journaux permettent de récupérer le nom d'utilisateur, l'adresse IP et le port utilisés pour la tentative de connexion. Si cela fournit déjà quelques informations intéressantes, il manque cependant celles de bas niveaux (niveau TCP) et de haut niveau (niveau applicatif). Il est possible d'obtenir plus de détails dans les journaux en modifiant le niveau de détail des journaux SSH, mais le côté intrusif de la démarche n'est pas très satisfaisant et elle ne permet pas d'obtenir les informations pour un réseau complet.

2.3 Outils d'analyse réseaux

LIDPS Suricata [**SURICATA**] présenté dans MISC n°66 est une sonde réseau de détection d'intrusion open source. Elle propose les fonctionnalités d'un IDS basé sur des signatures combiné à des fonctions de surveillance réseau orienté sécurité (*Network Security Monitoring*) qui ont été renforcées avec la version 2.0 parue en début d'année 2014. De nouveaux protocoles ont été ajoutés et Suricata peut maintenant extraire et journaliser des éléments des protocoles HTTP, TLS, DNS et SSH. Le tout peut être enregistré dans un fichier au format JSON qui est extensible et s'intègre facilement avec des outils d'analyse de données récents comme Splunk [**SPLUNK**] ou Elasticsearch [**ELK**]. Cette dernière solution sera présentée et utilisée dans la suite de l'article.

Pour récupérer des informations de bas niveau, c'est ulogd2 [ULOGD], le démon de journalisation de Netfilter qui a été utilisé. Depuis la version 2.0.4, un export JSON des informations est disponible ce qui rend facile l'intégration avec Elasticsearch. La configuration pare-feu mise en place journalise les connexions entrantes sur le système en générant un message pour chaque paquet SYN.

par l'utilisateur. La philosophie de l'interface est une approche en entonnoir avec une création de filtres lorsque l'utilisateur clique sur un élément et un rafraîchissement des différents éléments du tableau de bord (Fig. 1).

L'ensemble de ces outils est open source et, est, il convient de le noter, très facile à installer.

2.4 Analyse des données

Les données collectées sont enregistrées et analysées grâce au trio Elasticsearch, Logstash, Kibana communément appelé ELK. Elasticsearch est un moteur de recherche et d'indexation basé sur Lucene de la fondation Apache. Son architecture est distribuée de manière à assurer le passage à l'échelle lors de l'augmentation de la quantité de données par ajout de nœuds. Elasticsearch peut être vu comme une base de données NoSQL qui utilise la méthode REST pour les requêtes.

Logstash est un logiciel qui convertit les données d'une série de greffons d'entrée vers différents greffons de sortie. Les greffons d'entrée vont de l'acquisition de fichiers simples au traitement d'informations de redis ou rabbitmq en passant par la lecture de Twitter. Les sorties sont également très variées et elles adressent notamment Jira ou IRC même si l'export vers Elasticsearch est le plus utilisé. Un système de filtres est intercalé entre les greffons d'entrée et de sortie pour pouvoir réaliser des modifications (normalisation par exemple) et des enrichissements (ajout des données geoip par exemple).

Kibana est un visualiseur de données qui est basé sur des tableaux de bord construits dynamiquement

3 Première analyse

3.1 Volumétrie

Le point de collecte a subi plus de 10000 tentatives d'authentification SSH par jour. Des pointes à 300 essais par minute ont été observées. Dans la plupart des cas, de tels maximums sont liés aux tentatives de bruteforce effectuées par un attaquant unique dont le script est particulièrement agressif.

3.2 Géographie des attaques

La répartition géographique des attaques correspond à ce qui est généralement observé par les utilisateurs de Fail2ban avec plus de 90% provenant de Chine (Fig. 2, page suivante).

Cependant, la distribution des adresses chinoises est intéressante puisque, au moment de l'écriture de cet article, elles sont concentrées sur les réseaux 61.174.15.0/24 et 116.10.191.0/24 et elles semblent

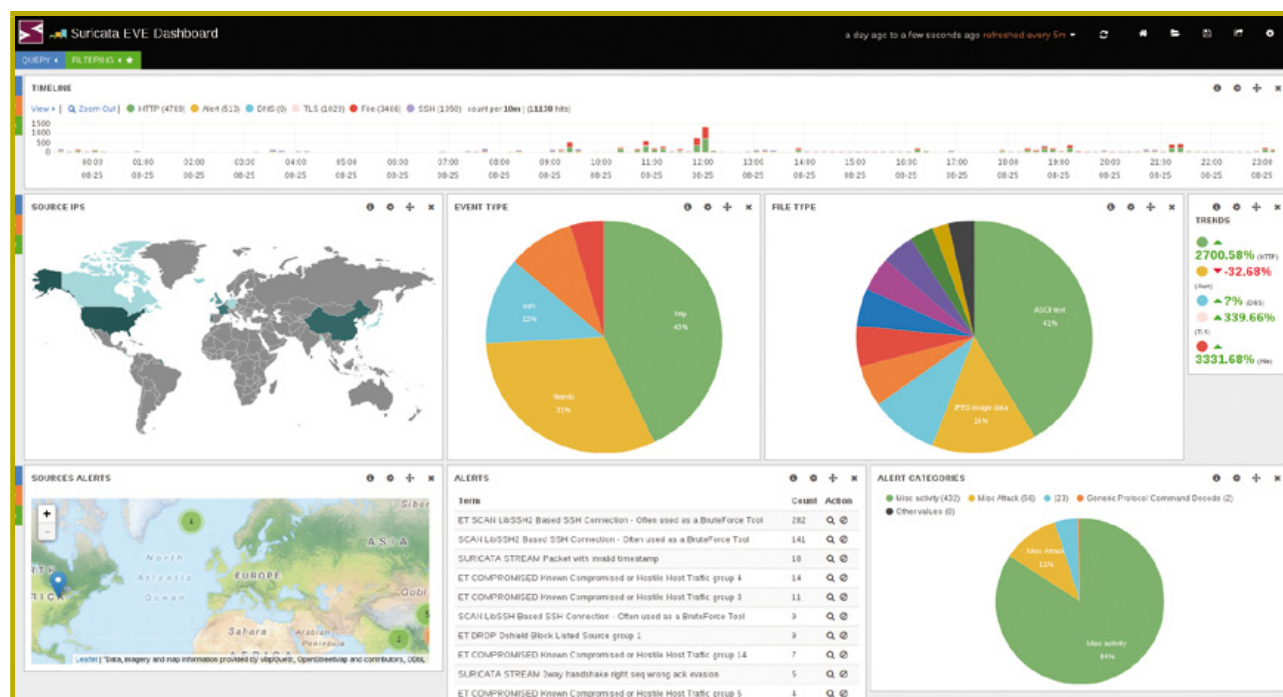


Figure 1

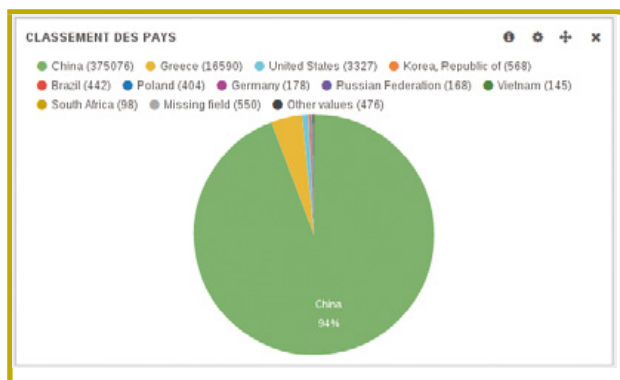


Figure 2

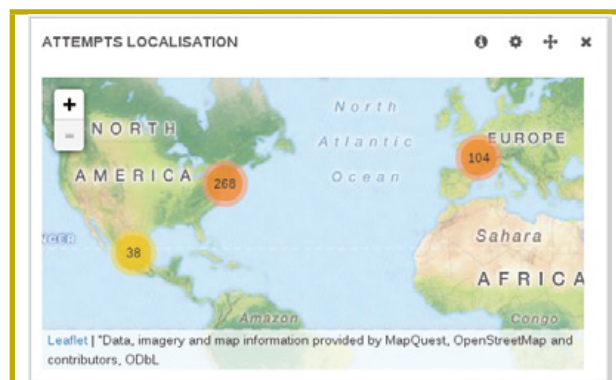


Figure 4

toutes être utilisées pour des attaques par brute force. Il est donc imaginable que les serveurs de ces réseaux soient dédiés aux scans et aux brute force sur certains protocoles comme SSH et qu'il s'agisse d'une véritable infrastructure d'attaque.

3.3 Technologies utilisées

Suricata journalise la version des logiciels clients et serveurs pour toutes les connexions et il est donc possible de connaître les technologies utilisées. Il s'agit majoritairement de libssh [LIBSSH] et libssh2 [LIBSSH2]. Le tableau suivant montre la répartition des événements SSH sur 1 mois environ :

Term	Count	Action
libssh-0.1	46147	Q ☒
libssh2_1.4.2	43680	Q ☒
libssh-0.4.8	6104	Q ☒
libssh2_1.4.3	1539	Q ☒
libssh-0.2	988	Q ☒
JSCH-0.1.51	504	Q ☒
libssh2_1.4.0	327	Q ☒
libssh-0.11	100	Q ☒
Pipfs-1.0	36	Q ☒
libssh-0.5.5	29	Q ☒
Other values	114	

Figure 3

Parmi les solutions plus exotiques, une des plus fréquentes est « JSCH-0.1.51 » [JSCH] qui est une implémentation en Java de SSH. La répartition géographique de l'utilisation de JSCH pour des brute force SSH est bien différente et concerne essentiellement les États-Unis, le Mexique et la France, comme le montre l'image suivante : voir Figure 4.

La majeure partie des attaques semble être isolée, une même adresse, scannant et se connectant pour effectuer le brute force. Cependant, certaines solutions

exotiques utilisent des méthodes plus avancées. C'est par exemple le cas d'une attaque réalisée avec un client nommé « Ruby/Net::SSH_2.9.1 x86_64-linux ». En moins de 20 secondes, 20 essais ont été réalisés depuis 5 pays différents. L'image suivante montre la répartition : voir Figure 5 ci-contre.

Réaliser une telle séquence suppose vraisemblablement un serveur central distribuant des tâches à des esclaves. La répartition géographique exotique de ces derniers fait penser à l'utilisation d'un botnet. Cette dernière attaque montre bien que les tentatives de brute force SSH reposent sur des solutions variées allant des scripts artisanaux à des solutions technologiquement complexes.

4 Une attaque spécifique

4.1 Pour les fenêtres, c'est la taille qui compte

Les données collectées par ulogd contiennent l'intégralité des entêtes des paquets et sont une source d'informations potentielles sur les technologies utilisées par les attaquants. Il existe en effet de nombreuses techniques pour déterminer le système d'exploitation d'une machine en analysant certaines des caractéristiques des paquets qu'elle a émis.

Si la plupart des techniques nécessitent des opérations mathématiques non accessibles dans Kibana, il est une donnée accessible directement qui en dit déjà beaucoup sur une machine. Il s'agit de la taille de fenêtre TCP à l'ouverture des sessions. La taille de fenêtre est un champ du paquet TCP définissant la quantité de données pouvant être envoyées sans acquittement. L'évolution de la taille de fenêtre au cours de la vie d'une connexion est définie de manière précise dans la RFC 1323 et c'est d'ailleurs un des paramètres utilisés par les pare-feux pour détecter des tentatives d'injection de données. Cependant, la RFC ne définit pas la taille de fenêtre au démarrage de la session et les systèmes d'exploitation ont donc dû choisir « arbitrairement » les valeurs.

La liste suivante montre les valeurs pour différents systèmes :

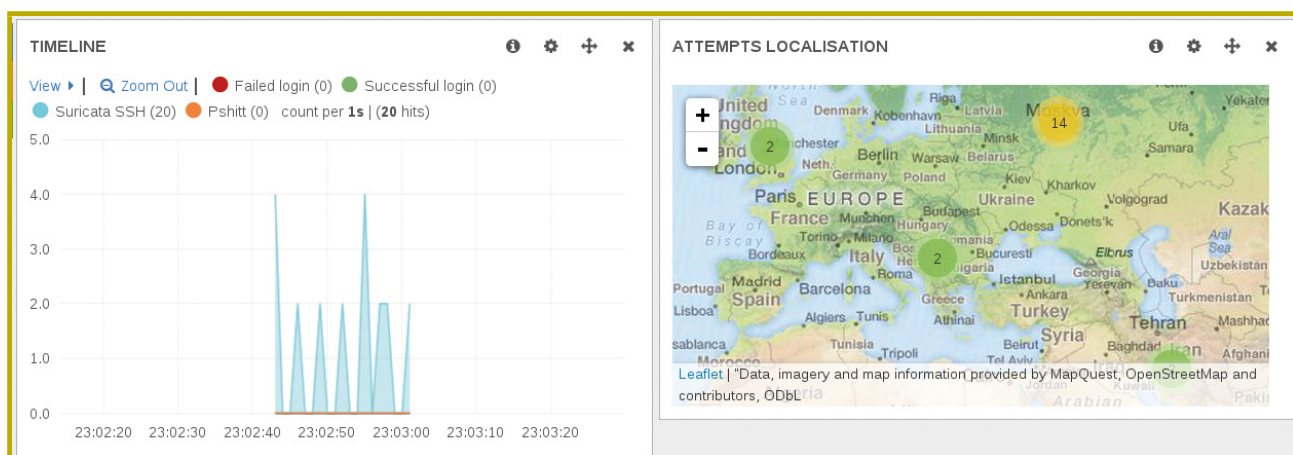


Figure 5

- 8192: Windows 7 SP1 ;
- 65535: Mac OS X 10.2 – 10.7, Windows XP, Windows 2003, FreeBSD ;
- 14600: Linux ;
- 5840: D'autres Linux.

La valeur de ce champ à l'initialisation de la connexion est donc un indicateur du système d'exploitation et s'agissant d'un des champs collectés par ulogd, il est intéressant de le tracer dans Kibana. Pour cela, il suffit de visualiser les valeurs de la fenêtre TCP pour l'ensemble des paquets ayant le flag SYN présent.

En demandant un « term » sur la valeur de fenêtre, on obtient le résultat suivant pour une période de 24h sur le système de collecte :

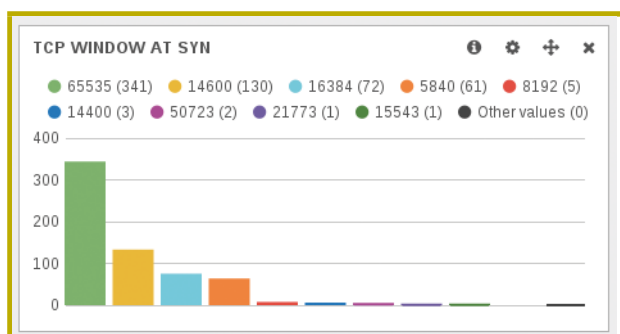


Figure 6

On identifie donc les valeurs dans notre précédent tableau pour les deux plus fréquentes fenêtres TCP contrairement à la troisième qui ne s'y trouve pas. Une étude des références sur les valeurs classiquement rencontrées confirme que cette valeur n'est utilisée par aucun système d'exploitation référencé et qu'elle mérite d'être approfondie.

Les barres du graphe dans l'interface de Kibana sont cliquables et le résultat est l'ajout d'un filtre sur la valeur. La mise à jour se fait sur l'intégralité des éléments de la page et la mise en place de ce filtre explicite des informations significatives sur la carte géographique des sources :



Figure 7

Cette taille de fenêtre est uniquement présente en Chine et une étude sur une plage de temps plus longue montre que cette répartition géographique est confirmée. On trouve toutefois trace de cette taille de fenêtre TCP dans quelque pays d'Amérique, mais si l'on limite l'étude au port 22 on exclut alors ces populations qui sont en fait à la recherche de relais mails ou serveurs mandataires ouverts.

4.2 Analyse en profondeur

Le tableau de présentation des événements individuels est le suivant :

@timestamp	src_ip	src_port	dest_port
2014-02-02T12:58:11.735Z	61.174.51.219	6000	22
2014-02-02T12:55:24.699Z	222.186.62.20	6000	22
2014-02-02T12:49:04.621Z	222.186.62.42	6000	22
2014-02-02T12:28:28.150Z	222.186.62.53	6000	22
2014-02-02T12:26:02.045Z	61.160.195.250	6000	22
2014-02-02T12:21:00.961Z	61.160.215.5	6000	22
2014-02-02T11:45:40.916Z	61.174.51.201	6000	22
2014-02-02T11:44:09.874Z	115.230.126.87	6000	22

Figure 8

On remarque immédiatement que les ports sources sont tous égaux à 6000. La probabilité d'un tel événement est quasiment nulle, car si les connexions étaient ouvertes

@timestamp	src_ip	src_port	dest_port	geoip.country_name	tcp.window
2014-01-31T08:11:15.314Z	61.160.223.102	6000	22	China	16384
2014-01-31T08:19:16.371Z	61.160.223.102	4585	22	China	65535
2014-01-31T08:20:08.378Z	61.160.223.102	1901	22	China	65535
2014-01-31T08:20:35.381Z	61.160.223.102	2363	22	China	65535
2014-01-31T08:20:44.383Z	61.160.223.102	2919	22	China	65535
2014-01-31T08:20:57.385Z	61.160.223.102	1208	22	China	65535
2014-01-31T08:21:07.387Z	61.160.223.102	4382	22	China	65535
2014-01-31T08:21:30.390Z	61.160.223.102	4519	22	China	65535
2014-01-31T08:21:51.393Z	61.160.223.102	4219	22	China	65535
2014-01-31T08:22:13.396Z	61.160.223.102	3548	22	China	65535
2014-01-31T08:22:33.399Z	61.160.223.102	1798	22	China	65535
2014-01-31T08:22:55.402Z	61.160.223.102	1275	22	China	65535
2014-02-02T10:56:04.435Z	61.160.223.102	6000	22	China	16384
2014-02-02T11:04:29.575Z	61.160.223.102	4075	22	China	65535
2014-02-02T11:04:52.582Z	61.160.223.102	4793	22	China	65535

Figure 9

par les systèmes d'exploitation des hôtes concernés, les ports sources seraient alors choisis de manière « aléatoire ». Ce choix du port source combiné à la valeur non référencée de la taille de fenêtre indique qu'il s'agit de paquets générés par une socket raw. Une analyse d'un échange montre que l'attaquant envoie deux paquets par séquence, un SYN suivi d'un RST en réponse au SYN ACK. C'est donc un scanner de port utilisant une socket raw et utilisant un modèle de paquet peu conventionnel en ce qui concerne la taille de la fenêtre TCP.

En se focalisant sur une IP donnée, on obtient le profil de l'attaque. Le tableau suivant présente les données reçues dans l'ordre chronologique : Figure 9.

On constate donc un scan (port 6000) suivi d'une série de tentatives de connexions SSH. Comme l'indique le port source aléatoire et la taille de fenêtre, il s'agit là de connexions ouvertes par le système d'exploitation et utilisées pour l'attaque bruteforce à proprement dit.

Si l'on regarde au niveau des événements SSH analysés par Suricata on s'aperçoit que le client SSH utilisé est libssh2_1.4.2 pour toutes les adresses IP utilisant la taille de fenêtre 16384.

Au niveau des alertes générées, les machines sont souvent déjà connues et elles apparaissent dans les règles identifiant les machines compromises ou offensives :

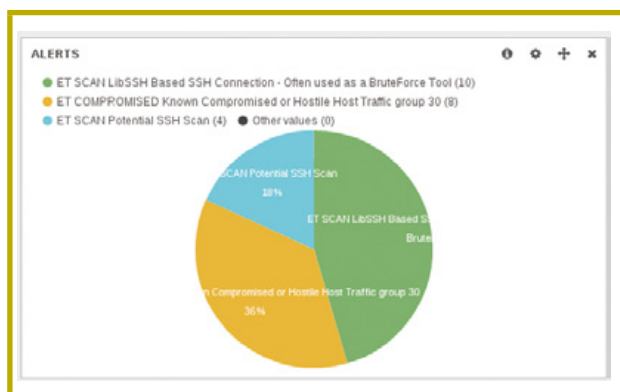


Figure 10

Les machines des réseaux 61.174.15.0/24 et 116.10.191.0/24 précédemment mentionnés utilisent toutes cette technique. Cette homogénéité tend à confirmer le côté industriel [1] des attaques effectuées à partir de ces deux réseaux.

L'ensemble de ces traces réseaux a donc permis de caractériser une plate-forme d'attaque par bruteforce qui n'est déployée qu'en Chine.

5 Un honeypot pour collecter les mots de passe

Les données collectées sont très intéressantes, mais il manque l'un des éléments les plus significatifs d'un bruteforce à savoir les couples utilisateur/mot de passe utilisés.

Il existe des solutions complètes comme Kippo [KIPPO] pour simuler une session SSH et ainsi récupérer les commandes lancées par le client, mais comme cela semble beaucoup pour faire une simple récupération des informations d'authentification, le logiciel pshitt [PSHITT] [2] a été développé dans le cadre de cette étude.

Il s'agit d'un serveur SSH basé sur paramiko [PARAMIKO] qui journalise les mots de passe dans un fichier au format JSON et refuse les connexions dans tous les cas.

Pshitt prend la place du serveur SSH et il est donc souhaitable de ne rediriger que le trafic des adresses utilisant des clients SSH non classiques (non OpenSSH dans le cas de l'étude) pour ne pas perdre la possibilité d'accéder en SSH au système. Pour détecter les IP avec des clients suspects, il suffit d'étudier les informations extraites par Suricata. On peut alors utiliser cette information pour mettre les adresses dans une liste exploitée avec Netfilter pour rediriger les tentatives de connexions vers pshitt.

Cette tâche est réalisée par l'outil « Deny On Monitoring » [3] [DOM] développé aussi dans le cadre de cette étude.

DOM s'appuie sur ipset **[IPSET]** qui est un système de gestion d'ensembles développé pour Netfilter. Ipset utilise des structures adaptées afin d'offrir des temps de recherche très courts et de permettre la mise à jour rapide de ses listes. Dans notre cas, nous avons uniquement besoin d'une table d'adresses IP. Et nous allons rediriger l'ensemble des adresses de la liste vers le port 2200 :

```
# création de l'ensemble
ipset create sofitel hash:ip
# redirige les IP de l'ensemble vers le port 2200 où écoute pshitt
iptables -A PREROUTING -t nat -m set --match-set sofitel src \
-i eth0 -p tcp -m tcp --dport 22 -j REDIRECT --to-ports 2200
```

Il suffit alors d'exécuter DOM en le configurant afin de rajouter à la liste les adresses n'utilisant pas (-i) un agent OpenSSH. :

```
./dom -f /usr/local/var/log/suricata/eve.json -i -m OpenSSH -s sofitel
```

Le code de DOM est un exemple d'utilisation des journaux JSON de Suricata. Sa simplicité montre à quel point il est facile de développer ses propres tâches utilisant les informations extraites par Suricata.

5.1 Résultats

Les comptes massivement attaqués sont root et admin et on trouve au niveau des mots de passe les combinaisons bien connues, comme l'illustre le tableau suivant qui reprend les données sur sept jours :

BRUTEFORCE USERNAME			PASSWORDS		
Term	Count	Action	Term	Count	Action
root	69075	Q	admin	422	Q
admin	13019	Q	password	63	Q
oracle	20	Q	zaq12wax	56	Q
cisco	16	Q	12345678	54	Q
huawei	13	Q	qwewasd	52	Q
nagios	12	Q	test	48	Q
administrator	10	Q	qwe123	48	Q
postgres	6	Q	789789	48	Q
lihan	6	Q	1234	48	Q
ftp	6	Q	root123	47	Q
Other values	165		Other values	81237	

Figure 11

La récupération des mots de passe a aussi conduit à observer quelques attaques surprenantes comme par exemple ce bruteforce violent basé sur libssh : Figure 12.

Il semble que l'attaquant a inversé le champ mot de passe et utilisateur dans son dictionnaire. Le cas présenté provient d'une adresse chinoise, mais une attaque similaire venant de Grèce a aussi été observée.

Conclusion

La mise en place d'une infrastructure de monitoring basée sur Suricata et Ulogd, couplée à un pot de miel dédié a permis de mieux classer les différentes attaques

« src_ip »	« username »	« password »
61.143.139.11	greatwallchina	root
61.143.139.11	greatwall	root
61.143.139.11	greatbritain	root
61.143.139.11	grasiele	root
61.143.139.11	grace123	root
61.143.139.11	grace	root
61.143.139.11	gr3atwall	root
61.143.139.11	gr3atbritain	root
61.143.139.11	gowno12	root
61.143.139.11	government	root

Figure 12

par bruteforce SSH. L'analyse menée facilement grâce à Elastisearch montre que la variété des attaques est considérable. Les moyens vont du script artisanal déployé sur une unique machine à l'utilisation d'une ferme de serveurs provisionnés automatiquement. Une fois le bruit constitué par les nombreux bruteforce, majoritairement chinois, analysé et nettoyé, les attaques technologiquement plus intéressantes comme ces rares cas distribués géographiquement apparaissent et pourraient être approfondis. ■

■ Notes

- [1] Le lecteur peu soucieux du politiquement correct peut s'autoriser à lire étatique plutôt qu'industriel.
- [2] Nommé d'après une citation de Jacques Chirac : [les attaques] c'est qu'elles font « pschitt ».
- [3] DOM, grand bloqueur de scans, a été nommé officieusement ainsi en référence à un homme politique français : « Dom, y nique trop de scans ».

■ Bibliographie

- [FAIL2BAN] <http://www.fail2ban.org/>
[SURICATA] Suricata : <http://suricata-ids.org/>
[SPLUNK] Site de la société Splunk : <http://www.splunk.com/>
[ELK] <http://www.elasticsearch.org/>
[Ulogd] <http://www.netfilter.org/projects/ulogd>
[LIBSSH] <http://www.libssh.org/>
[LIBSSH2] <http://www.libssh2.org/>
[JSCH] <http://www.jcraft.com/jsch/>
[KIPPO] <https://github.com/desaster/kippo>
[PSHITT] <https://github.com/regit/pshitt>
[PARAMIKO] <https://github.com/paramiko/paramiko>
[DOM] <https://github.com/regit/DOM>



LA SÉCURITÉ DES SAN (2ÈME PARTIE) : IMPACTS DE LA CONVERGENCE LAN/SAN

Pierre-Charles Wagrez – Cabinet Solucom

mots-clés : SÉCURITÉ / STOCKAGE / SAN / FCOE / NFS / SMB / ISCSI

La convergence LAN/SAN s'introduit lentement mais sûrement dans les réseaux des entreprises, même si cela se fait sous des formes différentes de celles prévues. Avec ces technologies viennent évidemment de nouvelles failles, mais également une plus grande exposition qui, via une mutualisation de fait des réseaux, ne permet plus de considérer le réseau de stockage avec la même légèreté qu'un SAN Fibre Channel isolé au fond du Datacenter. Dans la foulée de la première partie [1], focalisée sur les SAN Fibre Channel classiques, cet article aborde les différentes technologies de convergence LAN/SAN sur le marché, puis, après quelques explications sur leur fonctionnement, présente les risques et contre-mesures spécifiques à chacun.

1

Quel impact de la convergence LAN/SAN ?

La convergence LAN/SAN peut prendre de nombreuses formes :

- Transporter les flux de stockage sur Ethernet : Fcoe, voire AoE (ATA Over Ethernet) ;
- Transporter les flux de stockage sur IP : iSCSI ;
- Basculer vers un NAS, en travaillant en mode fichiers et non plus en mode blocs : NFS, CIFS/SMB...

Note : le lecteur qui voudra approfondir les mécanismes et différences entre eux pourra se référer à quelques articles en ligne sur le sujet [2].

La figure 1 permet de repositionner les différents protocoles les uns par rapport aux autres.

Toutes ces technologies, aussi différentes soient-elles, partagent un point commun : l'utilisation du réseau LAN Ethernet, et pour certaines également IP. Ces technologies peuvent donc déjà profiter des mécanismes

de sécurité des LAN Ethernet : VLAN, ACL, 802.1x, voire IP pour les protocoles basés sur IP : ARP Inspection, IP Source Guard...

Certaines normes de convergence intègrent d'ailleurs nativement ces mécanismes, comme la norme FC-BB-5 [3] pour le FCoE qui intègre du filtrage (ACL MAC dynamiques).

2 Zoom sur AoE

2.1 Présentation

AoE (ATA over Ethernet) est une tentative de fournir une couche de transport de flux de stockage la plus simple et la plus rapide sur Ethernet. Poussé en mode open source par Coraid [4], qui est à l'origine du protocole et distribue des baies de stockage AoE, cette solution reste pour l'instant une solution de niche qui ne semble pas mûre pour l'environnement grande entreprise : manque de sécurité ou de mécanismes de qualité de service.

AoE a toutefois atteint son objectif sur un point : la simplicité.

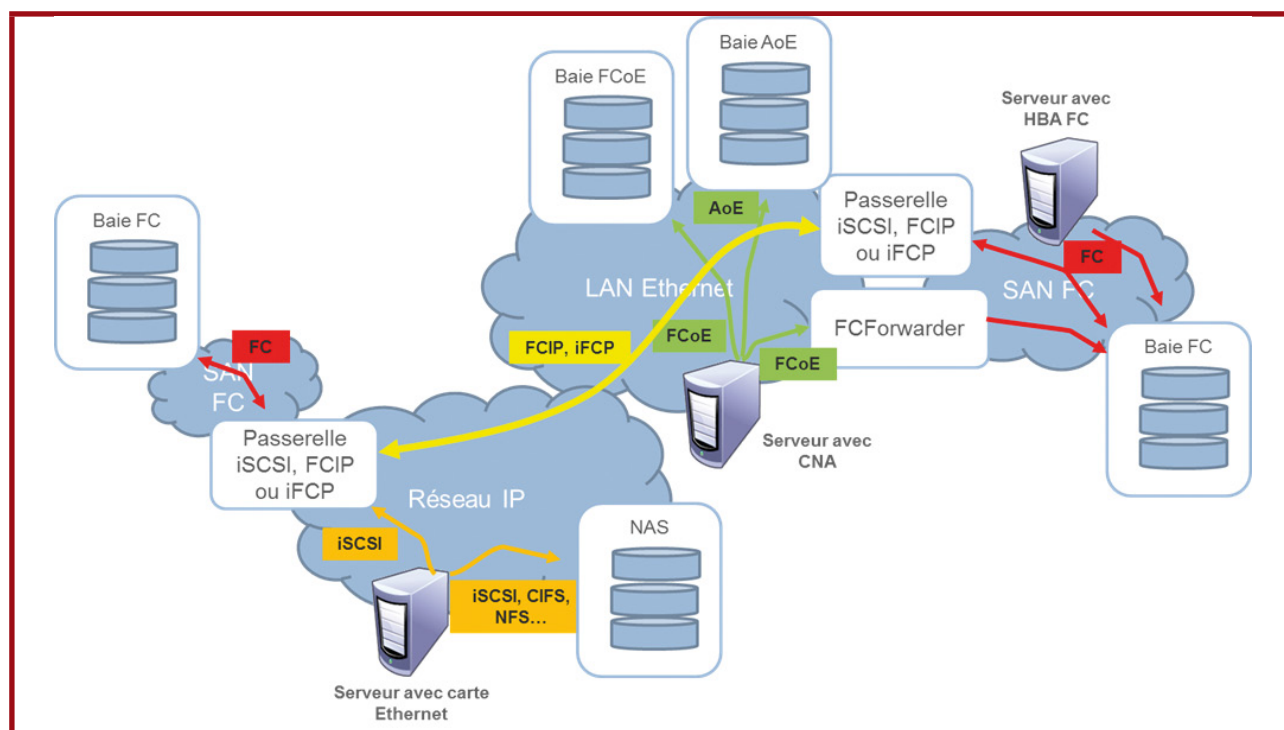


Fig. 1 : Positionnement des protocoles sur le réseau.

2.2 Sécurité

AoE a été montré du doigt à maintes reprises pour son absence de sécurité intrinsèque [5][6] puisqu'en effet l'argument principal qui est donné par Coraid est que AoE est « naturellement sécurisé, car non routable », ce qui est toutefois tout à fait en ligne avec la « conception de la sécurité » des administrateurs du stockage. De ce fait, AoE implémente une sécurité minimale, plus destinée à protéger d'erreurs de manipulation :

- Filtrage MAC sur les matériels Coraid, simulant le LUN Masking des SAN FC ;
- utilisation optionnelle d'une « configuration string » transmise en clair dans les requêtes. Limitée à 1024 octets, elle est écrite sur la cible par le client pour indiquer qui « possède » le LUN et contient par défaut le nom d'hôte et le nom de l'utilisateur propriétaire du LUN.

2.3 Risques spécifiques et contre-mesures

Le fonctionnement de AoE le rend naturellement vulnérable à de nombreuses attaques :

- Usurpation d'identité via MAC spoofing pour accéder aux données, puisqu'il est aisé de modifier son adresse MAC ;
- Interception de données par capture du flux, soit via positionnement en coupure du flux, soit via attaque

de MAC flooding pour provoquer une saturation de la table d'adresses MAC du switch et provoquer ainsi un flooding systématique des trames. L'attaquant pourra recourir à d'anciens outils comme Macof [7], puis simplement lancer un outil de capture réseau comme Wireshark. Cette attaque permet notamment d'obtenir la « configuration string » utilisable par AoE comme « authentification », qui passe en clair ;

- Attaques de rejeu pour modifier une donnée, après capture du flux initial comme indiqué précédemment ;
- Déni de service, en accédant au disque et en le reformatant par exemple, ou bien sur les serveurs par spoofing de l'adresse MAC du disque et envoyant des réponses erronées, susceptibles de provoquer un crash du serveur ou de l'application.

La plupart de ces attaques sont réalisables assez simplement à l'aide de Scapy [8] ou d'autres outils existants comme cela a déjà été montré [9].

Il est toutefois possible d'apporter un niveau de sécurité acceptable en s'appuyant sur les fonctionnalités de sécurité apportées par les switches évolués généralement présents en data center :

- VLAN dédié au réseau AoE, un peu à la manière du Zoning Fibre Channel, pour limiter l'exposition, l'attaque devant alors d'abord passer par la prise de contrôle d'un serveur raccordé à ce VLAN. On pourra utiliser plusieurs VLAN AoE pour protéger des ressources sensibles vis-à-vis de serveurs plus exposés pour éviter une attaque par rebond.

- Port-security, pour contrôler finement les adresses MAC et leur localisation et ainsi empêcher aussi bien les attaques de MAC spoofing que les attaques de CAM Table Overflow lorsqu'il est appliqué sur tous les ports du switch.

```
switch(config)# interface Gi0/0/5
switch(config-if)# switchport port-security
switch(config-if)# switchport port-security maximum 1
switch(config-if)# switchport port-security violation restrict
```

Ces serveurs étant rarement déplacés, on pourra mémoriser l'adresse MAC sans trop d'impacts en exploitation :

```
switch(config-if)# switchport port-security mac-address sticky
```

Attention !

S'il n'est pas possible de l'appliquer partout (ex. : liens inter switches) on pourra compléter avec du « Port Blocking » qui empêche le flooding, sur le port configuré, des trames dont l'adresse MAC destinataire n'est pas en table.

- éventuellement du filtrage MAC sur les ports, ou le VLAN, si les machines autorisées à échanger sont bien identifiées, mais ceci n'est pas viable en exploitation au-delà de quelques machines.

On pourra ajouter éventuellement l'utilisation de chiffrement au niveau du système de fichiers afin d'assurer la confidentialité des données, aussi bien en transit que sur les disques.

2.4 Zoom sur FCoE

2.4.1 Présentation

FCoE permet à des nœuds FC natifs de communiquer de manière transparente avec des nœuds raccordés à un réseau Ethernet. L'élément qui fait office de passerelle entre les deux mondes est un « Fibre Channel Forwarder » (FCF).

Sur Ethernet, la découverte et l'initialisation entre les nœuds et le FCF s'effectuent via *Fibre Channel Initialization Protocol* (FIP) :

- Découverte du FCF ;
- Fabric Login (FLOGI) ;
- Attribution du FC-ID et relation @Mac – FC-ID ;
- Établissement du lien FC virtuel au-dessus du nuage Ethernet.

La figure 2 repositionne les différents éléments.

- Utilisation de cartes CNA (*Converged Network Adapter*) 10 G Ethernet ;
- Support minimal des baby jumbo frames par les switchs Ethernet (2K) ;
- Support de DCBX, protocole de négociation des paramètres FCoE, par les switchs Ethernet de raccordement.

2.4.2 Sécurité

En FC natif, les nœuds FC et commutateurs sont connectés par des liens point à point FC. En FCoE, les « Enodes » et les « FCF » (*FC Forwarders*) sont connectés par des liens virtuels au-dessus d'un nuage Ethernet sans perte : il existe donc des opportunités d'attaques à travers le nuage Ethernet qui sont moins probables sur des liens point à point. Le challenge est donc d'appliquer une isolation sur le lien virtuel proche de celle d'un lien physique en utilisant un média partagé.

Les annexes C et D de la norme FC-BB-5 [3] prévoient un certain nombre de mécanismes pour réduire ou supprimer ces opportunités d'attaques supplémentaires liées au transport sur un nuage Ethernet, mais ce ne sont que des recommandations.

Ces recommandations prévoient donc la mise en œuvre d'ACL très précises au niveau Ethernet, qui limitent très strictement le type de flux, leur sens et les adresses MAC utilisées, et dont la définition est heureusement effectuée dynamiquement :

- Naturellement par les équipements comportant un FCF ;
- En interprétant les échanges FIP les traversant (« FIP snooping ») pour les autres... mais avec des limites du fait des possibilités de chemins multiples. Seuls les switchs « edge » peuvent réellement implémenter du filtrage et les filtres ne peuvent concerner que les équipements directement attachés.

2.4.3 Risques spécifiques et contre-mesures

Les mécanismes prévus par la norme permettent d'assurer un haut niveau de protection face aux attaques de niveau Ethernet. Si l'infrastructure implémente ces mécanismes, une attaque nécessitera la désactivation préalable de ces mécanismes - via une attaque sur l'administration ou via l'exploitation d'une éventuelle faille (aucune faille n'ayant été documentée publiquement à l'heure actuelle) - ou le contournement physique des équipements déployés par insertion d'un TAP ou d'un switch non sécurisé.

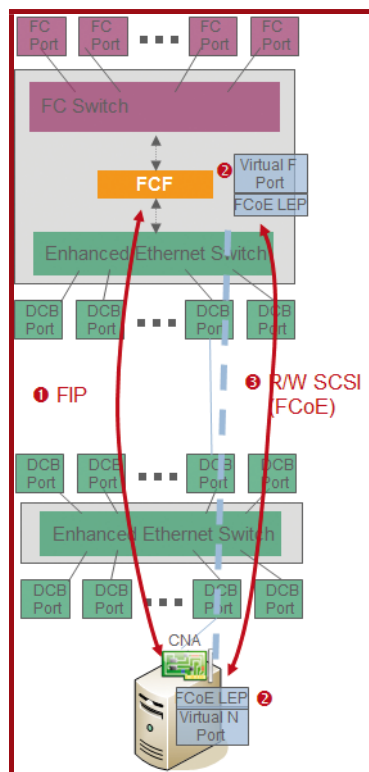


Fig. 2 : Fonctionnement de FCoE. FCoE transporte donc FC dans Ethernet, mais avec un certain nombre de prérequis

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

**+ DE
300
PAGES !**



À PARTIR DE

39€*

uniquement disponible
EN VERSION PDF

* TARIF

particulier 1 lecteur : 39,- € TTC

professionnel 1 à 5 lecteurs : 78,- € TTC



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 :
<http://www.gnulinuxmag.com>



Pour lire la suite, rendez-vous sur :
<http://www.ed-diamond.com>

Les switches non « FCoE aware » ou n'intégrant pas le FIP snooping n'appliquent pas ces recommandations et exposent les flux FCoE à toute attaque Ethernet classique.

À défaut de pouvoir utiliser du chiffrement, non supporté par FCoE, on pourra s'appuyer sur les mécanismes de sécurité des switches : VLAN dédiés et limités aux seuls endroits nécessaires et Port Security, voire également filtrage des adresses MAC par ACL.

Le FCoE n'introduit donc pas de nouvelle faille, dès lors que des switches compatibles sont utilisés, mais il ne sécurise pas non plus le SAN FC qui se trouve derrière : la partie FCoE n'est que le raccordement à un SAN FC et toutes les attaques sur un SAN FC restent valides.

2.5 ISCSI

2.5.1 Présentation

iSCSI transporte les trames SCSI dans TCP (port 3260) et, au contraire du FCoE, n'impose pas de prérequis forts sur le réseau de transport. Il peut donc être utilisé sur un WAN ou sur un réseau à faible débit en laissant le protocole TCP prendre en charge la fragmentation et la réémission en cas de perte. Néanmoins pour des performances se rapprochant du FC, il est nécessaire de disposer d'un réseau irréprochable.

iSCSI utilise des WWUI (*World Wide Unique Identifier*) pour désigner une entité unique : la couche iSCSI d'un nœud. Cet identifiant est utilisé uniquement au login

ou pendant la découverte. iSCSI utilise également des adresses, comprenant le WWUI : l'IQN (*iSCSI Qualified Name*), ou bien au format EUI-64.

iSCSI peut s'appuyer sur un serveur iSNS (*Internet Storage Name Service*) pour la découverte des Targets disponibles et éviter la déclaration manuelle sur chaque initiator. Chaque Target ou Initiator s'enregistre auprès du serveur iSNS, qui inclut des « Discovery Domains », comparables à du Soft Zoning dans le monde FC, comme illustré par la figure 3. Note : iSNS est également utilisé par iFCP.

2.5.2 Sécurité

iSCSI intègre nativement des mécanismes de sécurité [10][11], qui sont toutefois rarement mis en œuvre, bien qu'ils soient expressément recommandés par la RFC dans des environnements exposés :

- iSCSI supporte plusieurs méthodes d'authentification entre initiateur et cible : Kerberos v5, SPKM1, SPKM2, SRP et CHAP qui est le plus couramment supporté par les équipements ;
- la norme prévoit le support du chiffrement en IPsec et pour illustration l'initiateur intégré à Microsoft Windows le supporte nativement ;
- Les baies iSCSI implémentent généralement le LUN Masking sur l'IQN de l'initiator.

iSCSI peut par ailleurs s'appuyer sur des mécanismes liés à IP :

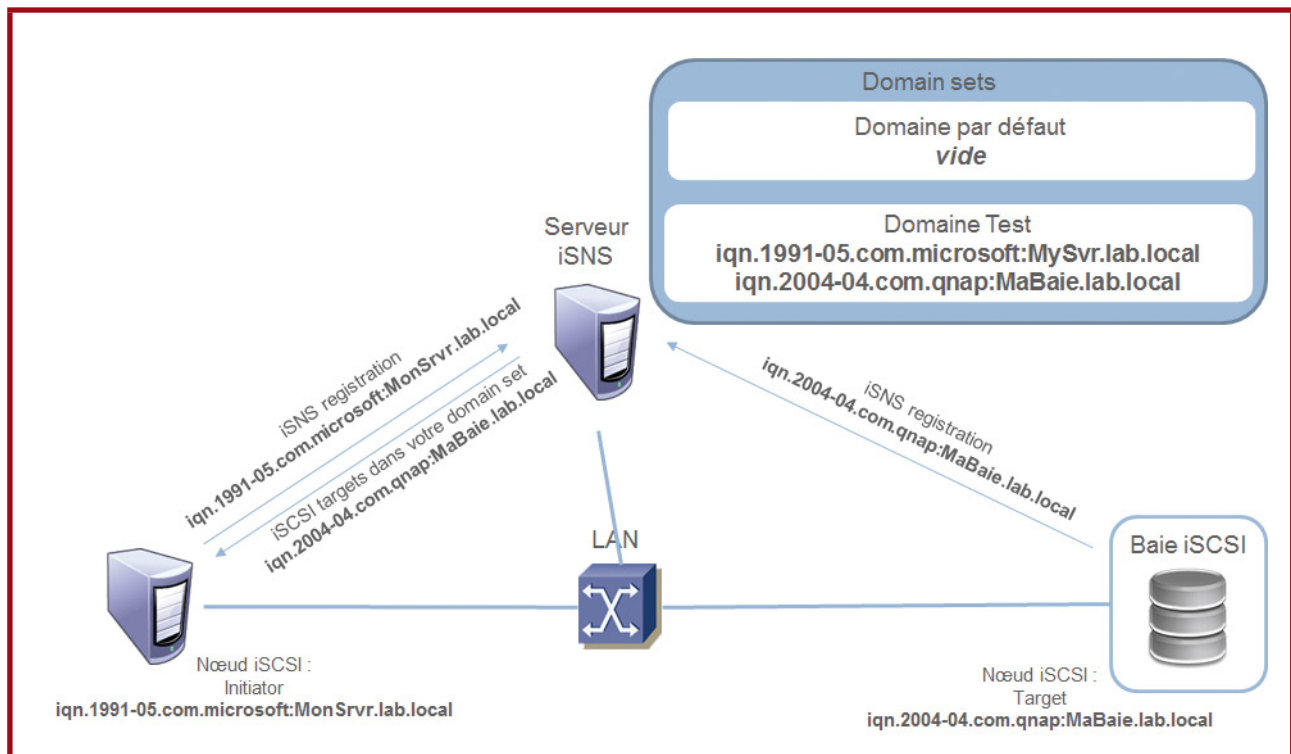


Fig. 3 : Fonctionnement de iSNS et des Domain Sets.



- iSCSI n'intègre pas de chiffrement natif, mais peut s'appuyer sur une connexion IPsec mise en œuvre au niveau de l'OS du serveur. Par exemple, l'initiateur iSCSI intégré à Windows supporte nativement IPsec. Le support côté baie dépendra du constructeur.
- iSCSI peut bénéficier de l'isolation logique apportée par les VLAN, dont on prendra soin de contrôler l'étendue aux seuls switchs nécessaires.

2.5.3 Risques spécifiques et contre-mesures

Les attaques possibles sur iSCSI sont nombreuses :

- Toutes celles liées à Ethernet et IP (capture du flux, attaque Man in The Middle...) ;
- IQN Spoofing, en général par simple configuration de l'initiateur iSCSI comme démontré dans la figure 4 ;
- Attaque brute force sur l'authentification iSCSI ;
- iSNS Domain Hopping, en général via attaque man-in-the-middle sur le serveur iSNS.

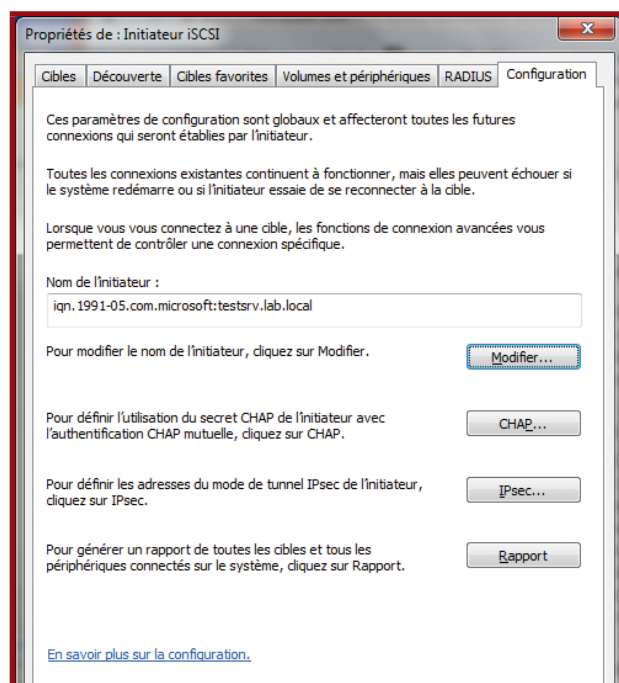


Fig. 4 : Modification de l'IQN via l'initiateur Windows.



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



Une attaque d'accès aux données sur iSCSI peut s'effectuer la plupart du temps en 3 étapes seulement : capturer un flux (interception préalable via ARP spoofing, par exemple), Spoofer l'IQN, monter le LUN.

Si un serveur iSNS est utilisé, une attaque man-in-the-middle sur le serveur (par ARP spoofing, par exemple) permettra d'intervenir sur toute la segmentation liée aux Domain sets et ainsi interdire l'accès à un disque de la part du serveur légitime pour provoquer un déni de service sur ce serveur et laisser libre accès à un serveur malveillant.

Les contre-mesures sont les suivantes, classées des plus simples aux plus complexes à mettre en œuvre :

- isoler des flux dans des VLAN dédiés, éventuellement non routés ;
- activer la fonction « port-security » sur les Switchs pour lutter contre les attaques de débordement de table CAM et protéger les sessions iSCSI contre la capture de données ;
- activer l'authentification CHAP mutuelle pour bloquer une attaque man-in-the-middle ou d'IP Spoofing. Cette méthode d'authentification a ses imperfections, mais limite néanmoins le risque ;
- utiliser un chiffrement IPsec pour garantir la confidentialité des échanges et ajouter une couche d'authentification mutuelle lors de l'initialisation du tunnel IPsec ;
- à défaut de chiffrement, on pourra activer « ARP Inspection » sur les switchs pour lutter contre les attaques d'ARP spoofing et donc l'interception, mais les correspondances adresses MAC – adresses IP devant être entrées manuellement dans ce contexte, l'exploitation peut être lourde en cas de gros parc et de changements réguliers.

3 Les NAS

Un NAS (*Network Attached Storage*) n'est ni plus ni moins qu'un serveur de fichiers. Les protocoles supportés embarquent leurs propres fonctionnalités d'authentification et de contrôle d'accès, voire de chiffrement. De nombreux articles ont été rédigés sur la sécurité des deux protocoles les plus rencontrés, SMB et NFS, et cet article ne s'attache qu'à aborder les aspects permettant de les comparer aux protocoles en mode bloc évoqués dans la première partie.

3.1 SMB

SMB (*Server Message Block*) de Microsoft supporte les différentes méthodes d'authentification disponibles dans le monde Microsoft et effectue un contrôle d'accès au niveau partage ainsi qu'au niveau fichiers.

La sécurité de SMB a beaucoup évolué avec les versions 2 et 3 [13] et notamment :

- la signature des paquets, afin de détecter la modification des données en transit, qui peut être imposée via la clef de registre **HKLM\System\CurrentControlSet\Services\LanManServer\Parameters\RequireSecuritySignature**. SMBv2 utilise HMAC-256 alors que SMBv3 utilise AES-CMAC.
- le chiffrement des données en transit à partir de SMBv3, via AES-CCM. Note : par défaut, un accès SMB2 n'est pas autorisé sur un partage marqué pour le chiffrement, mais SMBv1 oui pour des raisons de rétrocompatibilité. L'utilisation de SMBv1 permet de contourner le chiffrement et celui-ci est donc à désactiver. Ceci peut s'effectuer sur un serveur Windows 2012 en PowerShell via la commande suivante :

```
Set-SmbServerConfiguration -EnableSMB1Protocol $false
```

- « secure dialect negotiation », à partir de SMBv3 : la protection contre une attaque man-in-the-middle lors de la négociation, visant par exemple à désactiver le chiffrement en négociant une version de SMB inférieure, se fait via la signature des échanges. Il est possible de rendre cette sécurisation obligatoire sur les serveurs Windows 2012 via la commande PowerShell suivante :

```
Set-SmbClientConfiguration -RequireSecureNegotiate 1
```

L'authentification employée pour SMB a moins évolué et s'appuie sur Kerberos depuis Windows server 2003, voire sur NTLM dans certains cas [14]. Un certain nombre de failles ont été trouvées sur NTLM au fil du temps et ont été corrigées par Microsoft, comme celle des « weak nonce » [15].

Finalement, SMBv3 est un protocole qui apporte nativement un haut niveau de sécurité comparé aux protocoles SAN, même si quelques réglages restent nécessaires pour corriger certaines ouvertures laissées pour des raisons de rétrocompatibilité, et notamment la désactivation du support de SMBv1.

Nécessitant toutefois Windows Server 2012 et Windows 8, SMBv3 est de ce fait encore loin d'être généralisé en entreprise pour l'accès utilisateurs aux services de fichiers, mais peut se déployer aisément entre des serveurs 2012 et des baies NAS puisqu'aussi bien EMC que Netapp supportent SMBv3 sur leurs têtes NAS.

L'activation du chiffrement SMBv3 sur une tête NAS VNX d'EMC s'effectue via la ligne de commandes suivante :

```
[nasadmin@NASSRV ~]$ server_export server_1 -name SMB3_FS -o type=Encrypted /SMB3_FS
```

3.2 NFS

De la même manière que SMB, NFS (*Network Filesystem*) a beaucoup évolué et notamment sur les aspects sécurité dans sa dernière version.



NFSv2 et NFSv3 apportaient une sécurité toute relative où le droit de monter un partage était lié à l'identité de la machine cliente... qu'il était possible d'usurper, que le client soit identifié par son adresse IP ou son nom d'hôte. Sans compter qu'une fois le système de fichiers monté, il est possible d'accéder à n'importe quel fichier puisque le contrôle d'accès est lié à l'ID utilisateur et est effectué localement par la machine cliente. Une fois l'export NFS monté depuis une machine malveillante, un attaquant a donc accès aisément à l'intégralité des fichiers.

NFS avait donc un retard conséquent par rapport à SMB, y compris SMBv1, et NFSv4 [16] corrige en revoyant en profondeur le modèle de sécurité de NFS :

- une réelle authentification utilisateur, basée sur Kerberos ;
- une vérification du serveur, basée sur LIPKEY ;
- le support du chiffrement, basé sur RPCSEC_GSS et le plus souvent Kerberos v5 ;
- l'utilisation de TCP seul, assurant ainsi nativement la sécurisation contre la perte et l'adaptation à la bande passante disponible.

La version 4 de NFS est donc la première version à pouvoir se hisser au niveau de SMBv3 avec des fonctionnalités de chiffrement et d'authentification ... qui restent toutefois à être éprouvées par une analyse de la communauté. On bannira de préférence les versions précédentes, mais cela entraînera malheureusement l'exclusion d'un certain nombre de plateformes, dont VMWare VSphere qui reste limité à NFSv3.

Il sera donc souvent nécessaire de compléter la sécurité pour les implémentations inférieures à la v4 :

- utilisation de VLAN dédiés, et non routés ou bien filtrés, pour protéger les échanges NFS de ces environnements ;
- implémenter les mécanismes de sécurité du LAN déjà évoqués précédemment (port-security et ARP inspection).

Conclusion

La convergence LAN / SAN introduit bien évidemment des risques accrus, mais, à l'exception d'AoE, les protocoles utilisés font également preuve d'une plus grande maturité en matière de sécurité que Fibre Channel. Combinés avec quelques mécanismes complémentaires au niveau du LAN, tous permettent ainsi d'atteindre un niveau de sécurité correct, moyennant les toutes dernières évolutions pour certains, pour peu que les sécurités soient mises en œuvre : ici encore on se heurte à une méconnaissance du risque de la part des administrateurs du stockage, quand bien même les normes insistent sur la nécessité de mettre de la sécurité en place. ■

■ Remerciements

Merci aux relecteurs Ary Kokos et Jessica Gallante.

■ Bibliographie

- [1] **La sécurité des SAN (1ère partie)**, MISC 75, Pierre-Charles Wagrez (septembre 2014)
- [2] **Convergence LAN/SAN quel bilan ?**, Pierre-Charles Wagrez (2013), <http://www.it-expertise.com/convergence-lan-san-quel-bilan/>
- [3] **Norme FC-BB-5 Amendment 1, Standard ANSI 462-2012 (2012)**, voir <ftp://ftp.t10.org/t11/document.11/11-403v1.pdf>
- [4] www.coraid.com
- [5] **Security Assessment (2006)**, cf http://www.security-assessment.com/files/whitepapers/Insecurities_in_AoE.pdf
- [6] **Ivan Pepelnjak (2010)**, cf <http://searchnetworking.techtarget.com/ATA-over-Ethernet-for-converged-data-center-networks-No-way>
- [7] **Macof est un des outils du package dSniff écrit par Dug Song**, <http://www.monkey.org/~dugsong/dsniff/>
- [8] **Scapy est un générateur de paquets développé en Python par Philippe Biondi**, <http://www.secdev.org/projects/scapy/>
- [9] **Insecurities in AoE, Morgan Marquis-Boire (2006)**, cf http://www.security-assessment.com/files/documents/presentations/2006_MMB_Access%20over%20Ethernet-%20Insecurities%20in%20AoE.pdf
- [10] **Sécurité iSCSI, IETF**, <http://tools.ietf.org/html/rfc3720#page-99>
- [11] **iSNS, IETF**, <http://tools.ietf.org/html/rfc4171>
- [12] **Securing Block Storage Protocols** <http://tools.ietf.org/html/rfc3723>
- [13] <http://blogs.technet.com/b/filecab/archive/2012/05/03/smb-3-security-enhancements-in-windows-server-2012.aspx>
- [14] **Kerberos n'est pas utilisé en cas d'accès aux Share via l'adresse IP, Microsoft**, obtenu de <http://support.microsoft.com/kb/322979>
- [15] **Hernan Ochoa et Agustin Azubel (2010)**, https://media.blackhat.com/bh-us-10/presentations/Ochoa_Azubel/BlackHat-USA-2010-Ochoa-Azubel-NTLM-Weak-Nonce-slides.pdf
- [16] **NFSv4 RFC**, <http://www.ietf.org/rfc/rfc3530.txt>



ANALYSE D'UNE INSCRIPTION EN LIGNE : COMMENT VOS DONNÉES FUIENT SUR INTERNET...

Stéphane Labarthe - slabarthe@cnil.fr - misc@labarthe.es

Benjamin Vialle - bvialle@cnil.fr - misc@vialle.io

Auditeurs des systèmes d'information au service des contrôles de la CNIL

mots-clés : WEB / CDN / HTTPS / CHIFFREMENT / TRACKING / REFERER / CNIL

Lorsqu'un internaute visite un site web ou s'y inscrit, il laisse certaines de ses données personnelles au site. Mais des acteurs tiers peuvent aussi en être destinataires... Peu visibles de l'internaute, ils agissent parfois à l'insu du site web lui-même. Ce sont d'abord des acteurs situés au niveau applicatif, dont le code source s'exécute directement dans la page web (régies publicitaires, services de mesure d'audience, plateformes RTB). Ce sont aussi les CDN, qui se positionnent comme des intermédiaires au niveau réseau. Nous vous proposons l'analyse de deux méthodes utilisées pour détecter ces fuites de données. Les exemples présentés dans cet article sont fictifs, mais fondés sur des situations réelles rencontrées lors de contrôles de la CNIL.

L'essentiel est invisible pour les yeux. (Antoine de Saint-Exupéry)

1

Méthodologie et configuration de l'environnement proposées

1.1

Description générale de la méthodologie

L'un des objectifs de cet article est de proposer au lecteur de MISC une méthodologie reproductible afin de lui permettre d'analyser une partie des flux de données mis en jeu lors d'une inscription sur un site web (de e-commerce, de jeu en ligne, service de messagerie électronique, réseau social...). Le lecteur pourra alors se rendre compte par lui-même que, dans de nombreux cas, certaines des données personnelles transmises de son navigateur vers les serveurs du site sont également envoyées, souvent à son insu, vers des tiers qu'il ne connaît pas forcément.

La méthodologie se veut simple : elle consiste à réaliser une inscription sur le site web choisi puis, éventuellement, un achat avec paiement. On capturera alors d'une part les flux HTTP/HTTPS entre le navigateur et les serveurs web distants et d'autre part les flux réseaux entre la carte réseau de l'ordinateur de l'internaute et ces mêmes serveurs. Par ailleurs, l'examen du code source de la page web (y compris les éventuelles fonctions JavaScript externes) pourra permettre d'aborder le problème sous un autre angle.

1.2

Une configuration basée sur des logiciels libres

La configuration proposée pour l'analyse repose sur un ordinateur directement connecté à Internet (c'est-à-dire sans proxy) afin de visualiser immédiatement, dans le flux réseau, l'adresse IP de l'hôte destinataire et de s'affranchir de toute modification par le proxy des réponses renvoyées par les serveurs web.



Ensuite, afin de faciliter la reproductibilité de l'expérience, notre article s'appuie sur des logiciels libres :

- Pour la navigation : Mozilla Firefox augmenté de Live HTTP Headers **[LIVE_HTTP_HEADERS]**. Cette extension capture les entêtes HTTP.
- Pour la capture du réseau, WireShark : Ce logiciel permet la capture des flux émis et reçus par la carte réseau de l'ordinateur de l'internaute.

À noter que lorsque la communication est chiffrée (HTTPS), par son positionnement, Live HTTP Headers permet d'accéder au contenu des données échangées, ce qui n'est pas possible avec Wireshark (qui ne permet alors même pas d'accéder aux URL).

Un autre point important concerne le « nettoyage » de la configuration du navigateur, afin que celui-ci ne bloque ni ne produise de requêtes parasites. Pour cela, nous recommandons notamment de ne pas avoir d'extension produisant ou bloquant des requêtes (par exemple, les outils tels que NoScript qui bloquent l'exécution de code JavaScript), de supprimer tout l'historique, de désactiver certains paramètres pouvant générer des requêtes automatiques du navigateur (comme les mises à jour de sécurité) et de désactiver le blocage des cookies tiers.

À cette fin, une machine virtuelle ou un profil Firefox dédié (via la commande « **firefox -p** ») pourrait être une aide utile.

Enfin, il est indispensable d'avoir une horloge interne synchronisée sur un serveur de temps.

Remarque : Une autre approche est possible : celle du proxy local en rupture de chiffrement qui permet une recherche directe des contenus dans les flux chiffrés. Des solutions libres de ce type existent également **[ZAP]**. Cependant, nous avons préféré ici l'usage de WireShark, que le lecteur connaîtra probablement et qui réalise une écoute totalement passive.

2

Ces acteurs qui agissent au cœur de la page web (analyse de la capture HTTP/HTTPS)

2.1

« Cachons ces partenaires que vous ne sauriez voir » : comment les pages web communiquent avec des tiers

Lorsque l'on navigue sur un site web, notre navigateur communique évidemment avec les serveurs web de ce site mais également, dans la majorité

des cas, avec une multitude d'autres serveurs web, dits « tiers » ou « partenaires ». Certains peuvent se matérialiser de manière concrète, par exemple par le biais d'un bouton Facebook ou d'un bandeau publicitaire, mais la plupart sont invisibles « à l'œil nu ». Parmi les plus répandus, on peut citer les acteurs publicitaires (régies publicitaires, plateformes d'enchères publicitaires ou RTB **[RTB]**, etc.), les réseaux sociaux, les plugins vidéos et les solutions de mesure d'audience (« analytics »).

Ces tiers ou partenaires ont fourni à l'éditeur du site web un morceau de code source (appelé *tag*) que l'éditeur du site a inclus dans le code HTML de la page web.

Ce *tag* fait en général appel à une ressource externe (image ou fonction JavaScript) située sur le serveur web du tiers. C'est cet appel qui initie la communication avec le serveur web tiers.

Un examen du code source de la page pourra faire apparaître ce type de *tags*. Dans notre cas :

```
<script type="text/javascript" src="http://www.ma-regie-pub.com/pagead/conversion.js">
```

On peut remarquer que ce type de fonction n'est pas directement intégré dans le code HTML de la page renvoyée par le serveur : elle peut donc être modifiée par le « partenaire » sans que l'éditeur du site web ne s'en aperçoive. En s'exécutant à l'intérieur de la page web, cette fonction a accès à l'intégralité de son contenu. Des questions en termes de sécurité se posent donc. De plus, même un examen du code source « appelé » ne révélera que très difficilement ce qui se passe réellement à l'exécution, car il a souvent été « obfusqué » — c'est l'une des raisons pour lesquelles nous avons privilégié une analyse de capture réseau plutôt que du code source.

Alors comment garantir la sécurité d'un site web si l'on n'est pas capable de savoir ce que fait le code source qui s'y exécute ? (La même question se pose de manière accrue s'il est inconnu ou susceptible d'être modifié de manière invisible et sans préavis.) Nous y reviendrons dans la suite.

2.2 Inscription en ligne

Imaginons une inscription en deux temps sur ma-boutique-en-ligne.org : un formulaire de saisie de notre adresse électronique, suivi d'un autre formulaire permettant de saisir les autres données personnelles nécessaires à notre inscription. Nous le complétons avec les données suivantes (et le mot de passe « azerty78 »), voir Figure, page suivante.

Dans notre cas, les deux formulaires sont situés sur des pages ayant une URL de type HTTPS. Nous en déduisons donc que les données sont transmises directement aux serveurs de ma-boutique-en-ligne.org de manière chiffrée. L'authentification de l'unique destinataire et la confidentialité de la transmission semblent donc assurées.

Vos identifiants

Email*

Mot de passe*

Confirmation Mot de passe*

Votre identité

Civilité* ☒ M. ☐ Mme ☐ Mlle

Nom*

Prénom*

Date de naissance*

2.3 Analyse HTTP de la requête d'envoi de données : jusqu'ici tout va bien...

La première recherche que nous allons effectuer dans notre fichier de captures HTTP/HTTPS est la requête qui a servi à envoyer les données du formulaire d'inscription. Pour cela, nous recherchons des chaînes de caractères complétées dans le formulaire, par exemple avec le mot de passe « azerty78 ».

La recherche nous renvoie une seule réponse qui nous amène à la requête HTTPS suivante :

```
https://www.ma-boutique-en-ligne.org/fr/secure/subscription_s.html
POST /fr/secure/subscription_s.html HTTP/1.1
Host: www.ma-boutique-en-ligne.org
[...]
Referer: https://www.ma-boutique-en-ligne.org/fr/secure/
inscription_client/index.html?Email=slabarthe@cnil.fr
[...]
Email=slabarthe@cnil.fr&password=azerty78&gender=M&name=St%EF9phane&
lastname=LABARTHE &birth_day=06&birth_month=01&birth_year=1978
```

Cette requête montre que l'ensemble des données saisies est envoyé en HTTPS et avec la méthode POST vers une URL de [ma-boutique-en-ligne.org](https://www.ma-boutique-en-ligne.org) (ces données apparaissent ci-dessus en rouge). C'est donc un bon point.

Un champ de l'en-tête HTTPS attire cependant notre attention : il s'agit du champ *Referer*, mentionné plus haut. Il contient ici l'URL principale de la page précédente. On peut observer que l'adresse électronique y est passée en paramètre à travers la chaîne : « ?**Email=slabarthe@cnil.fr** ».

2.4 Mais où va donc mon adresse électronique ? (Transmission directe)

Mais la suite nous réserve une surprise. La recherche de « slabarthe@cnil.fr » dans le fichier de capture fait

apparaître beaucoup d'autres requêtes dont certaines destinées à des URL externes au site web. La première est la suivante :

```
https://ssl.mesuredaudience.com/image.gif?version=5.4.0&id=127283
4227&site=www.ma-boutique-en-ligne.org&mid=42572054&zmp=%2Ffr%2Fs
ecure%2Finscription_client%2Findex.html%3FEmail%3Dslabarthe@cnil.
fr&idclient=2545080-2
```

L'URL contient de nombreuses informations en paramètres dont notre adresse électronique. Elle est à destination du service de mesure d'audience du site et son envoi a été déclenché par une fonction JavaScript de celui-ci (un *tag*), qui s'est exécutée dans le contexte de la page.

On peut également remarquer l'utilisation par le service de mesure d'audience du site du protocole HTTPS afin de ne pas soulever d'alerte de sécurité au niveau du navigateur.

Les fonctions JavaScript des prestataires qui s'exécutent dans une page HTTP ou HTTPS ont accès au contenu et à l'URL de la page et peuvent communiquer de manière invisible avec les serveurs du prestataire. L'usage du protocole HTTPS n'offre donc, ici, aucune protection contre ce type de fuite de données.

2.5 Mais où va donc mon adresse électronique ? (Transmission par Referer)

Indépendamment des fonctions JavaScript « partenaires » qui peuvent aller chercher la donnée dans l'URL, la transmission de données personnelles par la méthode GET, dans le cas du protocole HTTP comme HTTPS, pose un autre problème, celui du *Referer*.

Lorsqu'une requête HTTP(S) à destination d'une URL externe est initiée par un *tag* tiers, le champ *Referer* de cette requête contiendra l'URL de la page web.

Or, si cette URL contient, comme dans notre exemple, une adresse électronique (ou d'autres données personnelles) dans ses paramètres, alors toutes les requêtes partant de la page web vers des tiers auront cette donnée dans leur champ *Referer*. Ainsi, les données personnelles seront effectivement transmises à tous les partenaires dont des *tags* s'exécutent dans la page. Dans notre cas, on s'aperçoit que l'adresse électronique d'inscription a été transmise par le biais du *Referer* à une dizaine de tiers, eux-mêmes susceptibles de la retransmettre en cascade (cas du *Real Time Bidding* notamment). L'exemple suivant l'illustre :

```
https://ma-regie-pub.com/ots/js-3.2/313349/9iE1AeL080a9AUke0nTadZ1u
ap0c3DmgNKH7q0f2JN0iGQGV1E2mCFLHqkasZd4Dx1wm4vC-q684Rbvwm6hCwbI8PnL
ZbATqC399jckM229ANxVYBquNnstVrk0YLNzn/1364290796051-132/1/2
GET /ots/js-3.2/313349/9iE1AeL080a9AUke0nTadZ1uap0c3DmgNKH7q0f2JN0i
GQGV1E2mCFLHqkasZd4Dx1wm4vC--q684Rbvwm6hCwbI8PnLZbATqC399jckM229ANx
VYBquNnstVrk0YLNzn/1364290796051-132/1/2 HTTP/1.1
Host: ma-regie-pub.com [...]
Referer: https://www.ma-boutique-en-ligne.org/fr/secure/
inscription_client/index.html?Email=slabarthe@cnil.fr
```



2.6 Une conséquence : la rupture d'« anonymat »

Une des conséquences de ce type de transmission est la rupture d'« anonymat » (en réalité de pseudonymat **[PSEUDO]**) par les régies publicitaires et autres acteurs tiers. En effet, ces acteurs reconnaissent en général les internautes grâce à des numéros identifiants (qu'ils disent « anonymes ») contenus dans des cookies. La transmission conjointe de l'identifiant du cookie et d'une donnée personnelle directement identifiante dans une même requête HTTP ou HTTPS (cf. la requête précédente) peut lui permettre d'associer ces données au numéro d'identification contenu dans le cookie. Le pseudonymat est ainsi rompu.

La tentation d'exploiter ces masses de données devient d'autant plus grande pour ces acteurs, même dans le cas des pure players (qui ne font par exemple que régie publicitaire ou que service de mesure d'audience). L'heure est aujourd'hui en effet à l'orientation vers le *cross device*, qui consiste à reconnaître un même utilisateur sur ses différents terminaux. Or, un traçage par cookies ou même par *fingerprinting* ne permet pas à une régie publicitaire de savoir que c'est le même internaute qui se connecte depuis son ordinateur personnel, son ordinateur professionnel, son smartphone... ; la régie verra trois cookies ou trois empreintes distinctes. Dès lors, utiliser l'adresse électronique discrètement récoltée pour lier ces trois identifiants devient tentant...

2.7 Aspect organisationnel de la problématique : pourquoi le service informatique ne devrait pas obéir au service marketing

Dans la pratique, ces *tags partenaires*, qui sont insérés dans le code source de la page d'un site, sont souvent fournis par le service marketing de la société éditrice au service informatique qui se contente alors de les positionner dans la page.

De plus, les prestations associées ne font en général pas l'objet d'un vrai contrat, avec des clauses de confidentialité **[CLAUSES]**, mais d'une simple facture, non connue du service informatique. Il en résulte souvent que, même lorsque la prestation est terminée avec le partenaire, le *tag* n'est pas retiré et continue de s'exécuter dans la page et donc de générer le dépôt de cookies aux utilisateurs du site et la transmission des données au prestataire, cela sans finalité et en dehors de tout encadrement contractuel. Ce problème, très répandu, a une expression pour le désigner : les *tags obsolètes*.

Lors d'un contrôle de la CNIL sur un important site français d'e-commerce, le même type de transmission que celui décrit dans la partie 2.4 (transmission directe de l'adresse électronique en paramètre de

e

EN PARTENARIAT AVEC

QUARKSLAB
 INNOVATIVE SECURITY

PROPOSE 2 BADGES, FORMATIONS SUR 7 MOIS SUR

REVERSE ENGINEERING - SÉCURITÉ OFFENSIVE

BADGE REVERSE ENGINEERING

Un BADGE pour être capable d'étudier tous les programmes,

- Analyse de codes malveillants
- Reverse et reconstruction de protocoles
- Protections logiciels et unpacking
- Analyse d'implémentations de cryptographie

BADGE SÉCURITÉ OFFENSIVE

Un BADGE pour trouver, exploiter, corriger les vulnérabilités dans un système :

- Détournement des protocoles réseaux non sécurisés
- Exploitation des corruptions mémoires et vulnérabilités web
- Escalade de privilèges sur un système compromis
- Intrusion, progression et prise de contrôle d'un réseau

e

www.esiea.fr/badges
badges@esiea.fr

www.quarkslab.com/fr-badges
badges@quarkslab.com



l'URL) avait été identifié par l'analyse de la capture HTTP. Parmi les prestataires destinataires de l'adresse électronique, il y en avait un qui était totalement inconnu du responsable *front-office* de la société ainsi que du responsable développement. Finalement, il aura fallu interroger l'un des plus anciens développeurs de la société pour se rendre compte que le prestataire en question était une société de *A/B testing* dont le contrat était échu depuis plusieurs années. Le responsable *front-office* et le directeur informatique nous indiqueront même qu'avec le grand nombre de *tags* partenaires (certains obsolètes) s'exécutant dans le code source du site, son fonctionnement en était considérablement ralenti !

Il est vrai que le nombre de serveurs tiers (et donc de noms de domaine) appelés depuis un site de e-commerce sur un simple parcours « inscription-achat » dépasse régulièrement la vingtaine...

2.8 Premières recommandations à destination des éditeurs de sites

Si le risque de fuite de données depuis un site web ne peut être réduit à zéro dès lors que celui-ci inclut des *tags* partenaires, on peut néanmoins effectuer les recommandations suivantes :

1. Le service juridique et le service informatique doivent être impliqués du début à la fin de toute nouvelle prestation impliquant l'insertion d'un *tag* dans le code source de la page (publicité, mesure d'audience, plugin vidéo ou de réseau social, A/B testing, etc.). Cette collaboration devra conduire en particulier à la suppression des *tags* obsolètes en fin de prestation.
2. Pour chacun des *tags* partenaires, le rapport risque/bénéfice doit être évalué en prenant en compte son positionnement (page d'accueil, formulaire de saisie de données, etc.).
3. L'envoi des données de l'ensemble des formulaires (pré-inscription, inscription, authentification, commande, paiement avec saisie de données bancaires, etc.) doit se faire en utilisant le protocole HTTPS et la méthode POST.
4. Des tests sur la façon dont fonctionne le site web et les destinataires des données doivent être régulièrement effectués. La méthode décrite dans cet article est facilement reproductible.

Pour aller plus loin et ne pas se contenter d'analyser uniquement les échanges HTTP/HTTPS entre le navigateur de l'internaute et les serveurs web distants, on peut étendre cette analyse et s'intéresser plus précisément aux serveurs destinataires des données à caractère personnel. On s'aperçoit alors que d'autres acteurs en sont destinataires.

3 Quand les CDN entrent en jeu (analyse WireShark)

3.1 Mais à qui est cette adresse IP ?

Reprenons l'inscription réalisée précédemment sur le site de e-commerce ma-boutique-en-ligne.org et observons la capture réseau effectuée à l'aide du logiciel WireShark.

Celle-ci a été faite en parallèle de la capture des entêtes HTTP/HTTPS. Par souci de simplification, nous allons traiter le cas d'une requête HTTP (donc lisible en clair dans WireShark). Le cas d'une requête HTTPS est en effet un peu plus complexe, puisqu'il faut faire correspondre les entêtes capturés d'une part et les captures réseaux WireShark d'autre part, ou bien utiliser un outil tel que OWASP — ce que nous ne faisons pas dans ce type de constat.

Cherchons d'abord à récupérer la (ou les) trame(s) ayant transmis notre adresse électronique :

```
tshark -r capture.pcap -T text -Tfields -e frame.number -e ip.proto -e ip.dst -e http.host -e text | grep cnil
```

```
9      6      203.0.113.62      [...]Timestamps:
TSval 5587, TSecr 1204208032, POST fr/secure/subscription_s.
html HTTP/1.1\r\n, Accept-Charset: ISO-8859-1, utf-8;q=0.7,*;q=0.7\r\n,
Keep-Alive: 115\r\n, \r\n, HTTP request 1/1, [truncated]
Email=slabarthe@cnil.fr&password=azerty78&gender=M&name=St%E9phane&
lastname=LABARTHE &birth_day=06&birth_month=01&birth_year=1978
```

La trame n° 9 de la capture nous indique que le serveur contacté pour la requête contenant l'adresse électronique et les autres données d'inscription a pour adresse IP 203.0.113.62.

Une requête *whois* sur cette adresse IP renvoie vers un FAI étranger. Le site ma-boutique-en-ligne.org a pourtant tous ses serveurs hébergés en France.

L'explication est la suivante : cette adresse IP est celle d'un serveur du CDN (Content Delivery Network) situé chez un FAI étranger. Il s'avère en effet que les sociétés fournissant des services de CDN possèdent des serveurs le plus souvent placés dans les datacenters des FAI afin d'être au plus près des utilisateurs finaux.

3.2 Mais par où transitent nos coordonnées bancaires ?

Le site ma-boutique-en-ligne.org propose également de stocker les coordonnées bancaires de ses clients, pour ne plus avoir à les leur demander lors de leur achat suivant.

En se rendant sur la page d'enregistrement des coordonnées bancaires, on peut vérifier que l'on se trouve bien sur une page HTTPS : les échanges de données entre le premier serveur de ma-boutique-en-ligne.org et le



navigateur sont protégés (chiffrement) et le serveur du site [ma-boutique-en-ligne.org](https://secure.ma-boutique-en-ligne.org) est bien celui qu'il prétend être si l'on en croit le certificat (authentification).

Nous allons de nouveau rechercher la requête ayant soumis le formulaire. Contrairement à l'exemple précédent, les flux sont cette fois chiffrés (HTTPS).

Voici la requête telle que capturée par Live HTTP Header, directement dans Firefox :

```
https://secure.ma-boutique-en-ligne.org/fr/secure/enregCoordBancaires
POST /fr/secure/enregCoordBancaires/ HTTP/1.1

Host: secure.ma-boutique-en-ligne.org
[...]
Referer: https://www.ma-boutique-en-ligne.org/fr/secure/
envoiCoordBancaires
[...]
Date: Wed, 09 Oct 2013 13:27:45 GMT
Card_number=0123456789012345&ExpDate=06/01/78&CCV=007
```

On constate que la méthode utilisée est POST. Nos coordonnées bancaires seront donc envoyées en POST à l'adresse <https://secure.ma-boutique-en-ligne.org>. Un envoi bel et bien sécurisé, donc : authentification, chiffrement et envoi direct vers [ma-boutique-en-ligne.org](https://secure.ma-boutique-en-ligne.org), seule à pouvoir déchiffrer les données.

Comme les flux sont chiffrés, nous ne pouvons pas utiliser WireShark pour effectuer des recherches de texte et ainsi retrouver la requête TLS correspondante. Procédons autrement : cherchons, dans la capture réseau, la résolution DNS. Celle-ci a dû avoir lieu au même moment puisque c'est la première fois que nous nous rendons sur le domaine secure.ma-boutique-en-ligne.org :

```
tshark -r capture.pcap -T text -Tfields -e frame.number -e
ip.proto -e ip.dst -e dns -e text -e frame.time | grep 13:27

1002 17 198.51.100.1 Queries,secure.ma-boutique-en-ligne.org:
type A, class IN,Answers,secure.ma-boutique-en-ligne.org: type CNAME, class
IN, cname secure.ma-boutique-en-ligne.org.mon-cdn.net,secure.ma-boutique-en-
ligne.org.moncdn.net: type CNAME, class IN, cname e678.mon-cdn.net: type A,
class IN, addr 203.0.113.62 Oct 9, 2013 13:27:14.526743000
```

La résolution DNS nous indique ici que secure.ma-boutique-en-ligne.org est un alias qui pointe vers [secure.ma-boutique-en-ligne.org.mon-cdn.net](https://secure.ma-boutique-en-ligne.org) qui, lui, pointe vers e678.mon-cdn.net !

Le reste de la capture montre qu'une connexion TLS s'établit entre l'adresse IP du CDN (203.0.113.62) et celle de l'utilisateur : nos coordonnées bancaires sont donc envoyées, dans un premier temps, au CDN.

Par ailleurs, on constate que, pour le site de ma-boutique-en-ligne.org, c'est l'intégralité des requêtes qui transite par l'adresse IP 203.0.113.62. Ainsi, le CDN est destinataire de l'ensemble des informations.

3.3 L'intermédiaire (presque) invisible

Dans la pratique, les serveurs de CDN sont très souvent destinataires du contenu des requêtes HTTP

ou HTTPS des internautes, requêtes susceptibles de contenir des données à caractère personnel. Autrement dit, le CDN peut-être vu comme un intermédiaire quasi invisible, par lequel transitent non seulement les contenus statiques des pages internet, mais également les données personnelles échangées entre le serveur web et l'internaute.

Notons que, dans presque tous les cas rencontrés par la CNIL lors des contrôles, l'intégralité des flux HTTP ou HTTPS transitait par les serveurs du CDN.

3.4 La question des certificats cryptographiques

3.4.1 Explications

Les CDN proposent deux options dans le cas de flux chiffrés :

- En général, le CDN se voit fournir les certificats et clefs privées des sites web. C'est le cas dans l'exemple précédent. Les contenus peuvent être ainsi directement délivrés en étant chiffrés au moyen du certificat du site web sur lequel l'internaute navigue.
- Le CDN peut également utiliser ses propres certificats, soit sur l'ensemble des pages, soit pour certains contenus.

Dans les deux cas, les flux sont accessibles en clair pour les CDN. Lors des contrôles, ces faits étaient le plus souvent ignorés du DSI et, s'il y en avait un, du RSSI.

3.4.2 L'accès aux certificats cryptographiques par le CDN : aspect contractuel

La société *mon-cdn*, dans le cadre de son service de CDN, se place dans le premier cas mentionné dans la partie précédente. La société a accès au certificat SSL de ses clients, cela faisant partie intégrante du contrat. Ce dernier peut prévoir que le CDN est autorisé à héberger et utiliser les certificats au nom du site éditeur sur ses propres serveurs. Par exemple, la société AKAMAI, leader du marché, précise dans un document accessible sur son site web : « *If an Akamai customer wants to deliver entire SSL pages, they do so by CNAMEing their Web site over to the Akamai network. The Akamai network serves SSL pages over a secure connection on behalf of the customer using a customer-provided SSL certificate.* » **[AKAMAI_CNAME]**.

Si nous revenons à notre exemple, un examen de la requête suivant la requête « SERVER HELLO » du protocole TLS fait d'ailleurs apparaître le certificat, dans lequel nous pouvons observer un champ « *Common Name* » et un champ « *Organisational Unit Name* » faisant respectivement référence à ma-boutique-en-ligne.org et à *Mon-CDN*.

3.5 Une contre-mesure : le déROUTAGE des flux porteurs de données sensibles

Il existe un moyen de s'assurer que les flux porteurs de données « sensibles » ne sont pas accessibles par le CDN : ces flux doivent être « déROUTés » et aller directement au serveur web depuis le navigateur du client, sans transiter par un CDN.

S'agissant de l'exemple précédent, ma-boutique-en-ligne.org pourrait utiliser, *a minima*, deux noms de domaine supplémentaires :

- images.ma-boutique-en-ligne, redirigeant (CNAME) vers les serveurs du CDN, pour les contenus statiques, ou du moins ne contenant pas de données sensibles ;
- secure.ma-boutique-en-ligne.org, pointant directement vers les serveurs web de ma-boutique-en-ligne.org, pour les flux porteurs de données sensibles.

Dans tous les cas, il faut également prendre en compte les remarques de la partie précédente s'agissant du *Referer*. En effet, cette technique du déROUTAGE des flux porteurs de données sensibles n'a été rencontrée qu'une seule fois lors des contrôles de la CNIL. Malheureusement, parmi les requêtes qui transitaient par le CDN, certaines avaient pour *Referer* une URL contenant l'adresse électronique de l'internaute renseignée à l'inscription.

Attention !

Attention ici aux termes « données sensibles » : l'usage du terme « données sensibles » n'est pas employé dans cet article exclusivement au sens de l'article 8 de la loi Informatique et Libertés, qui concerne les données relatives aux origines raciales ou ethniques, aux opinions politiques, philosophiques ou religieuses ou à l'appartenance syndicale des personnes, ou qui sont relatives à la santé ou à la vie sexuelle. Ici, le terme « données sensibles », à prendre dans un sens très large, regroupe toute donnée que le lecteur de MISC juge comme ne devant pas être communiquée à des tiers (les secrets commerciaux et industriels, et les données à caractère personnel en général, dont les données bancaires et données de santé, etc.).

3.6 Les CDN et l'informatique dans les nuages : exemple des réseaux sociaux

Le recours à des CDN peut occasionner des fuites de données lorsque les serveurs hébergent des données personnelles non publiques : c'est le cas de certains réseaux sociaux. Ainsi, les photos ou vidéos privées servies par un réseau social peuvent être stockées sur les serveurs de son CDN. Lorsque l'internaute affiche

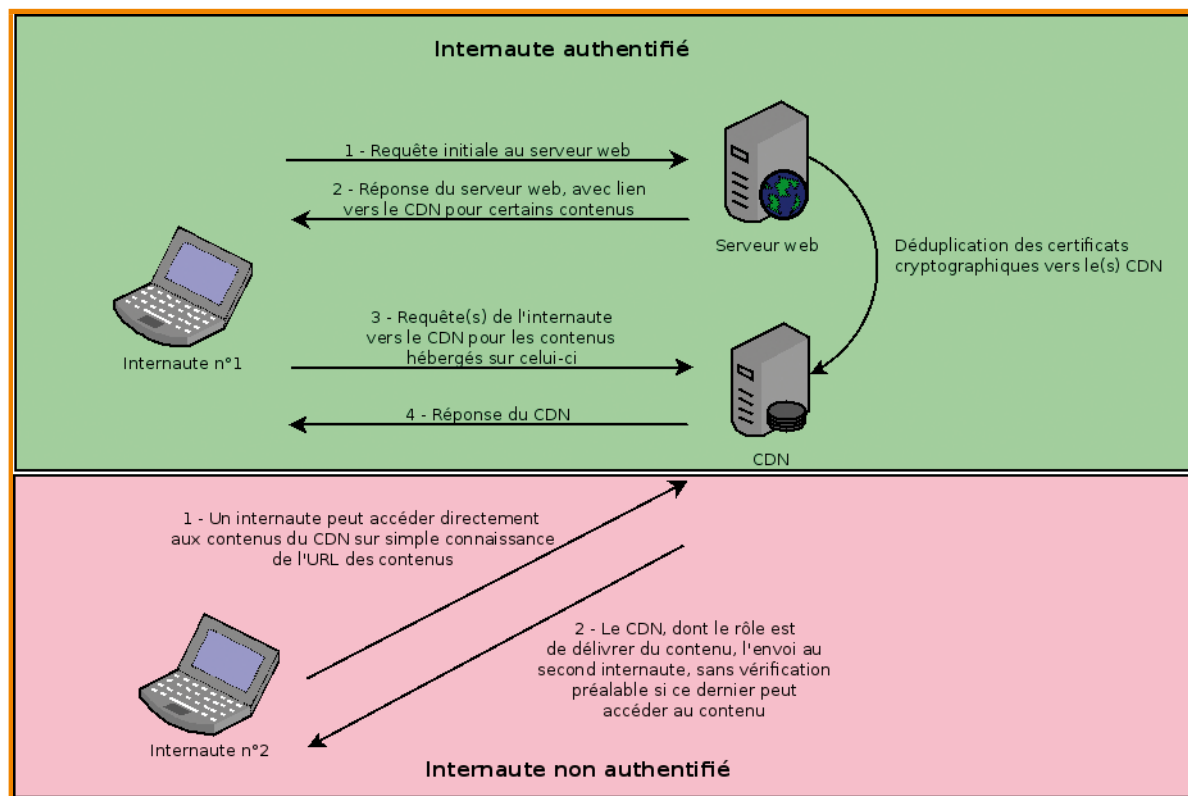


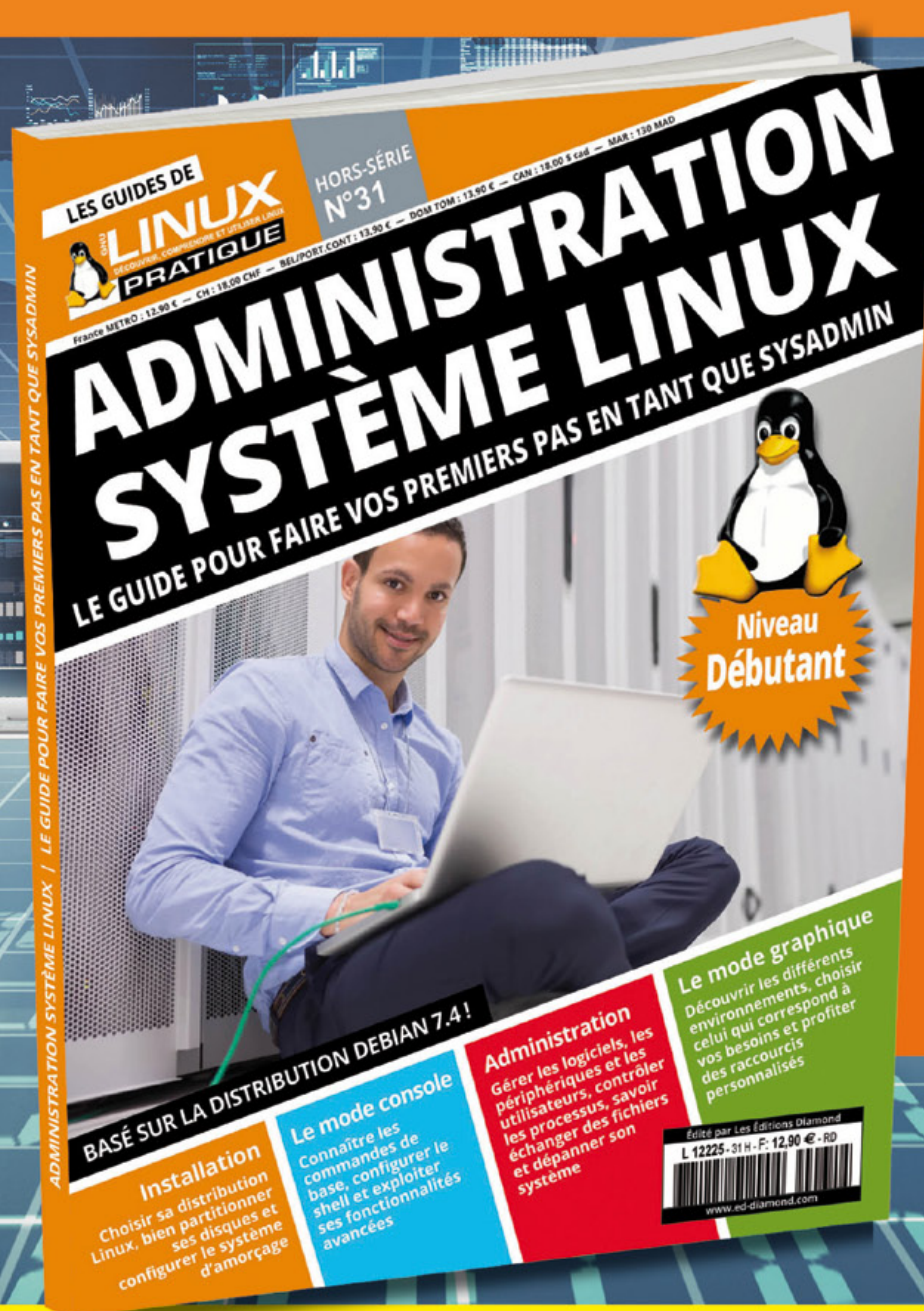
Fig. 2 : Schéma simplifié illustrant le cas donné en exemple.

NE MANQUEZ PAS LINUX PRATIQUE HORS-SÉRIE N°31 !

DOSSIER SPÉCIAL :

ADMINISTRATION SYSTÈME LINUX

NIVEAU DÉBUTANT



LE GUIDE
POUR
FAIRE VOS
PREMIERS
PAS EN
TANT QUE
SYSADMIN !

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



Sous réserve de toutes modifications.



une page du site web avec ces contenus, ceux-ci sont récupérés depuis le CDN et intégrés dans la page web. Or, on constate que ces contenus peuvent être accessibles directement depuis les serveurs du CDN, via leurs URL, sans authentification. Ainsi, il est donc possible à un utilisateur d'accéder à du contenu non public du site sans être authentifié, directement sur les serveurs du CDN, sur simple connaissance de l'URL dudit contenu, comme l'illustre la figure n° 2, page précédente.

3.7 Les CDN et l'informatique dans les nuages : exemple d'une application SaaS professionnelle

Voici un exemple de cas réel rencontré lors d'un contrôle de la CNIL : la société contrôlée fournissait une application en ligne (*cloud computing* de type *SaaS*) à destination de professionnels qui pouvaient y charger leurs bases de données. Les données transitant par le CDN étaient déchiffrées par celui-ci après un éventuel ajout de contenu statique. Lorsque la société a compris que toutes les communications entre ses clients et ses serveurs étaient accessibles par le CDN et qu'elle allait donc devoir, sur le fondement de la loi informatique et libertés, en informer ses clients, elle a préféré rompre le contrat avec le CDN en question...

4 CDN, Big Data et NSA

À l'ère du Big Data, les CDN s'intéressent au contenu des myriades de données qui transitent par leurs réseaux.

Aujourd'hui, 50 % du trafic Internet mondial transiterait par les réseaux des CDN **[TRAFFIC CDN]** dont 20 % à 30 % pour le seul acteur AKAMAI, leader du marché et qui posséderait pour cela plus de 125 000 serveurs répartis dans environ 80 pays **[AKAMAI]**.

Par conséquent, au vu de la formidable masse de données accessibles déchiffrées par les CDN, dont les acteurs majeurs sont américains (AKAMAI, AMAZON CLOUDFRONT, CLOUDFLARE, etc.), on ne peut pas exclure le risque d'un espionnage massif par ce biais, en complément du risque d'usage détourné à des fins publicitaires.

La vulnérabilité évoquée dans la partie 3.6 a d'ailleurs effectivement été exploitée par un programme d'espionnage opéré par le GCHQ dénommé « SPRING BISHOP » et permettant d'accéder aux photos privées de Facebook, sans authentification, directement sur les serveurs d'AKAMAI **[SPRING BISHOP]**.

5 Les limites de notre analyse

Les exemples mis en évidence dans la partie 2 de cet article ne montrent pas les méthodes de traçage

indirect que peuvent mettre en place les sites web. Certaines sont connues et utilisées depuis maintenant de nombreuses années, comme les cookies, tandis que d'autres font encore l'objet de travaux de recherche et de développement, comme le *fingerprinting*.

De plus, les exemples qui illustrent cet article présentent systématiquement des données à caractère personnel « lisibles » au niveau du protocole HTTP. Or, des contrôles ont mis en évidence certaines techniques de camouflage par encodage, hachage ou chiffrement des données lors de leur transmission à un tiers. Ainsi, l'internaute n'est pas toujours en mesure de savoir que son adresse électronique ou toute autre donnée est transmise et utilisée pour l'identifier sur d'autres sites que celui sur lequel il s'est inscrit.

Enfin, notre analyse ne permet pas de mettre en évidence les communications de serveurs à serveurs, notamment s'agissant des données transmises pour la publicité ciblée, entre acteurs publicitaires.

Conclusion : à vous de jouer !

Aujourd'hui, l'internaute n'est pas clairement et suffisamment informé de la transmission à des tiers d'une partie de ses données à caractère personnel. Cependant beaucoup de ces transmissions, parfois qualifiables de « fuites de données », sont facilement détectables. En cas de détection, il est d'ailleurs possible de saisir la CNIL qui s'est vue octroyée, depuis cette année, le droit d'effectuer des contrôles en ligne. ■

Note

Le rapport annuel de la CNIL contient, chaque année, la liste des sociétés contrôlées. Les rapports annuels des années 2012 et 2013, contiennent un certain nombre d'acteurs importants de l'Internet français et mondial. Ils sont disponibles en ligne : http://www.cnil.fr/fileadmin/documents/La_CNIL/publications/CNIL_RA2012_web.pdf et <http://www.cnil.fr/institution/actualite/article/article/bilan-2013-la-protection-des-donnees-une-preoccupation-croissante-des-particuliers/>.

■ Remerciements

Nous tenons à remercier notre institution et plus particulièrement son Secrétaire Général, Édouard GEFFRAY, ainsi que Florence FOURETS et Christophe VIVENT de nous avoir soutenus dans la rédaction de cet article. Merci également à Thierry CARDONA pour ses relectures actives.

Retrouvez les références de cet article sur le blog de MISC : www.miscmag.com

IT EXPO

**LE RENDEZ-VOUS
DES BUSINESS TECHNOLOGIES**

**PARIS PORTE DE VERSAILLES
PAVILLON 5-1
18 & 19 NOVEMBRE 2014**

**LE SALON RÉFÉRENT
DES DSI, DES
DIRECTIONS MÉTIERS
ET DES DIRECTIONS
GÉNÉRALES**

5 UNIVERS POUR UNE VISION GLOBALE :

CLOUD COMPUTING ET DATACENTER SOLUTIONS HARDWARE ET INNOVATIONS DE SERVICES
COLLABORATIVE BUSINESS SOLUTIONS PARTAGÉES POUR LA CRÉATION DE VALEUR
EXPLOITATION DES DONNÉES BIG DATA ET ALIGNEMENT STRATÉGIQUE SUR LA DONNÉE
MOBILITÉ ET DÉMATÉRIALISATION TRANSFORMATION NUMÉRIQUE DES ORGANISATIONS
SÉCURITÉ PROTECTION DES ACTIFS ET PRÉVENTION DES RISQUES

WWW.IT-EXPO.FR

Platinum sponsor



Gold sponsor



Silver sponsors



DSI Business Hub Sponsors



Partenaire officiel



Partenaires premium



Partenaires presse



Un événement



Communication et direction éditoriale



7^{ème} Forum International de la Cybersécurité

20 ET 21 JANVIER 2015

LILLE GRAND PALAIS

PLUS D'INFORMATIONS SUR
www.forum-fic.com



**L'événement européen de référence
sur la cybersécurité**



**Cybersécurité
et Transformation Numérique**

contact@forum-fic.com

Un événement co-financé par



RÉGION
Nord-Pas de Calais

et organisé par



CEIS

