BadUSB,

une faille imparable ?

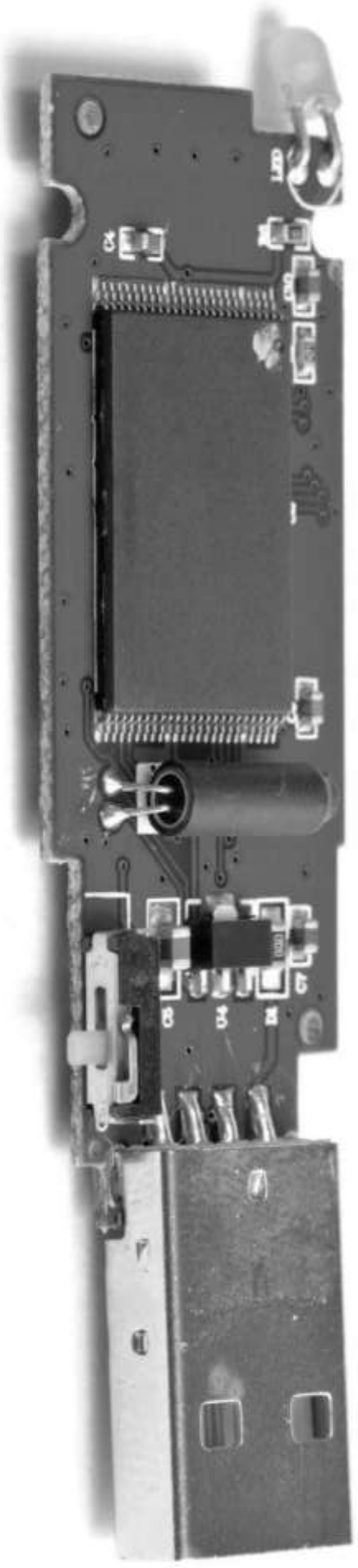
V 1.0 // Novembre 2014.

Bertin IT, Division de Bertin Technologies.
10 bis avenue Ampère, FR – 78180 Montigny

T. +33(0)1 39 30 62 50
E. contact@bertin-it.com

@Bertin_IT

www.bertin-it.com



Introduction

Depuis l'annonce de sa découverte en août 2014, la faille de sécurité BadUSB ne cesse d'agiter la communauté IT.

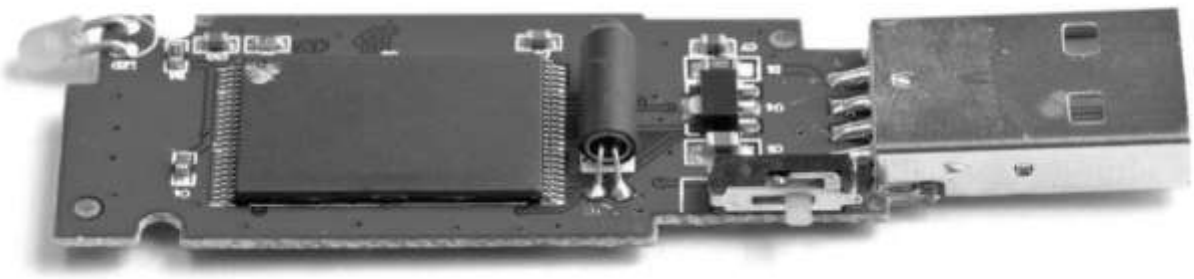
Indétectable et 'impatchable', elle se terre dans les confins du hardware et place sous la menace des milliards de périphériques USB. Au centre des inquiétudes, la très usitée clé USB.

En octobre, le vent de panique est devenu tempête avec la révélation du code source permettant d'exploiter cette vulnérabilité sur un certain type de matériel.

Le 12 novembre, une nouvelle étude indiquait que la faille n'affecterait que la moitié des microcontrôleurs USB diffusés sur le marché. Mais en l'absence de précision sur la marque et le modèle de puce qu'il contient, il demeure impossible de déterminer si un périphérique est vulnérable ou non, à moins de le démonter…

Auteur
Stéphanie BLANCHET

Relecteurs
Erwan LE DISEZ
David BOUCHER
Benoît POULOT-CAZAJOUS



BadUSB,
faille planétaire.

Les clés USB sont bien connues pour être des vecteurs d'infections potentielles par le biais de fichiers malveillants qu'elles peuvent contenir. Un scan d'antivirus ou un reformatage constituent généralement des parades efficaces. Dans le cas de BadUSB, la menace est indécelable, car elle ne réside pas dans la mémoire flash du support mais au cœur même du firmware qui contrôle son fonctionnement. Et la faille ne se limite pas aux simples clés : tout périphérique USB peut théoriquement être corrompu.

■ De la vulnérabilité naturelle de l'USB

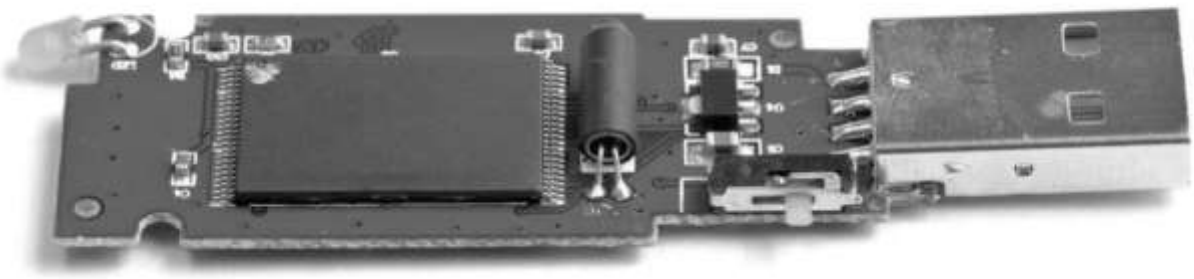
Les menaces informatiques liées à l'utilisation des périphériques USB (Universal Serial Bus) ne sont pas nouvelles. Elles sont inhérentes à la capacité de ces appareils aujourd'hui pléthoriques (clés, disques durs externes, appareils photo, téléphones mobiles, tablettes, souris, claviers, imprimantes, webcams, microphones, adaptateurs, etc.) de se brancher sur n'importe quel ordinateur, communiquer avec celui-ci et y introduire potentiellement un contenu malicieux. Cette versatilité de l'USB est à la fois la raison de son succès et son point faible. La très populaire clé USB constitue naturellement le vecteur d'infection le plus commun. Ce petit objet si banal qu'on le croirait volontiers inoffensif a pu servir d'agent de transmission à deux vers particulièrement redoutables : Conficker[1] (2008) et Stuxnet[2] (2010).

Les attaques spécifiques aux clés USB[3] tirent généralement parti de propriétés facilitatrices de Windows. L'exécution automatique (autorun) à l'insertion de la clé peut tout aussi bien provoquer l'installation d'un pilote que celle d'un logiciel malveillant contenu dans l'appareil. De même, la lecture automatique (autoplay) peut lancer sur le poste de l'utilisateur l'application nécessaire à l'ouverture d'un type de fichier stocké sur la clé, mais aussi permettre à un virus également présent d'exploiter une vulnérabilité de cette application (p.ex. exploit d'une faille d'Abode Reader à la lecture d'un PDF).

La faille BadUSB exposée lors du Black Hat 2014 par les chercheurs de Security Research Labs (SRLabs), Karsten Nohl et Jacob Lell[4], aurait ceci d'original qu'elle ne procède pas d'un fichier malveillant chargé sur le périphérique USB, mais d'une reprogrammation du firmware même de ce périphérique.

■ Universal Serial… Killer

Pendant deux mois, Karsten Nohl et Jacob Lell passent au crible de la rétroingénierie le microcontrôleur qui autorise un dispositif USB à communiquer avec un ordinateur et permet à l'utilisateur de charger et transférer des fichiers. C'est ainsi qu'ils découvrent que ce firmware – ici, de la marque Phison – peut être reprogrammé de manière à receler un code d'attaque. Ceci en profitant simplement d'une lacune commune à une large majorité de périphériques USB : l'absence de protection qui garantirait que tout nouveau code ajouté possède bien la signature cryptographique infalsifiable de son fabricant. Ainsi, tout périphérique ayant la capacité de mettre à jour son firmware de manière non sécurisée peut être corrompu, quelle que soit sa classe (d'interface, comme un clavier ou une souris ; de stockage, comme une clé USB, …).



Son firmware modifié, l'appareil malicieux peut se faire passer pour n'importe quel autre (clavier, disque dur externe, etc.) et prendre le contrôle de l'ordinateur, installer un virus qui se propagera aux autres périphériques USB, exfiltrer des données, espionner l'utilisateur… L'éventail d'actions possibles grâce à cette faille est large (cf. fig.1) : escamotage de fichiers sur une clé ou un disque dur externe, réécriture de données à la volée pour ajouter des virus à des fichiers nouvellement stockés, usurpation d'un écran pour accéder aux informations de sécurité (p.ex. captchas, codes PIN aléatoires), …

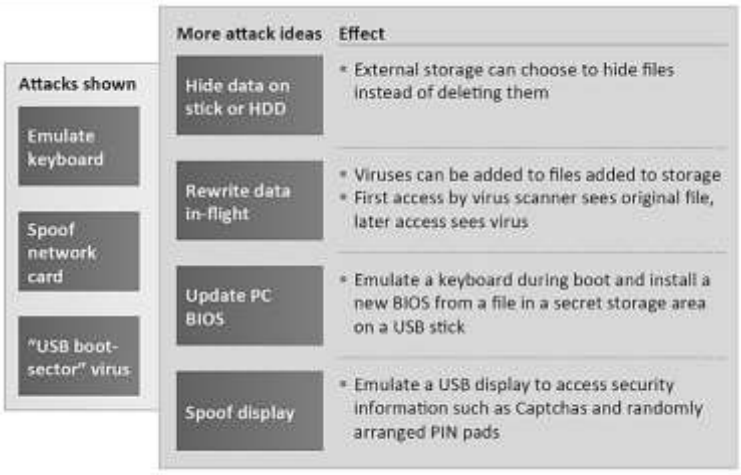


Figure 1 / K. Nohl, S. Krißler, J. Lell, 2014. p.18

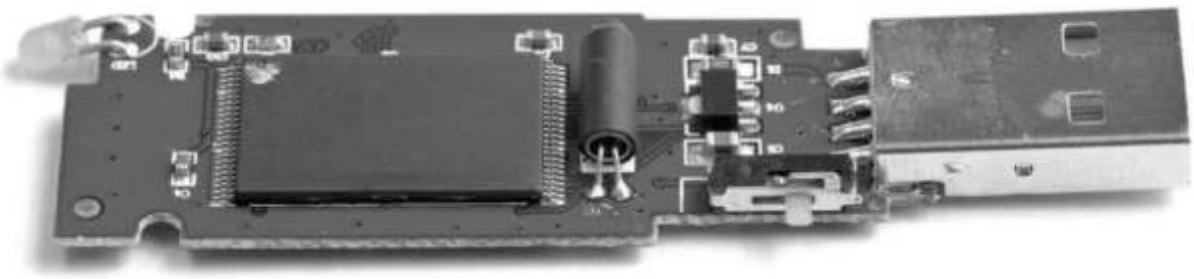
Lors de leur intervention au Black Hat 2014, Karsten Nohl et Jacob Lell font une première démonstration de simulation d'un clavier mettant en œuvre une attaque permettant d'intercepter des mots de passe et de s'approprier les privilèges de l'utilisateur connecté. Ils exposent ensuite un scénario de détournement totalement transparent du trafic Internet sur une machine Windows grâce à l'usurpation d'une carte réseau par un smartphone sous Android, présenté comme « *la plateforme d'attaque USB la plus simple* ». C'est là le seul cas pour lequel ils fournissent une preuve de concept. Les deux chercheurs font observer qu'en simulant un clavier, le smartphone infecté peut aussi compromettre l'authentification forte sur laquelle repose la sécurisation des transactions bancaires en ligne. Enfin, ils présentent une attaque par lancement d'un virus à la mise en marche de l'ordinateur (boot-sector virus) pour infecter le BIOS.

■ Pas vu, pas pris.

Les attaques de type BadUSB seraient non perceptibles par les défenses traditionnelles, la plupart des antivirus étant capables de détecter l'injection d'un malware classique via une clé USB, mais non d'accéder au firmware de cette clé et reconnaître que celui-ci a été dénaturé. De surcroît, un reset du périphérique ou le formatage de la clef USB ne parviennent pas à supprimer le firmware qui conserve donc toute sa nocivité. « *Une fois infectés, les ordinateurs et leurs périphériques USB ne peuvent plus jamais être dignes de confiance* », peut-on lire sur le site web de SRLabs[5]. En somme, corriger cette faille demanderait à repenser totalement la façon dont sont conçus les périphériques USB.

■ Quoi de neuf ?

Le caractère de nouveauté de BadUSB n'a pas manqué d'être contesté[6]. Des exploits de failles dans l'USB ont déjà fait parler d'eux avant la très médiatisée conférence de Karsten Nohl et Jacob Lell. Si celle-ci a suscité une telle effervescence,



c'est vraisemblablement en raison de l'ampleur que la famille d'attaques BadUSB peut prendre, notamment dans un contexte de boom des objets connectés et d'émergence de nouvelles menaces.

En 2010, Adrian Crenshaw[7] (TrustedSec) présentait lors de la DEF CON Hacking Conference, le PHUKD (Programmable HID USB Keystroke Dongle), petit appareil contenant une carte électronique à microcontrôleur Teensy programmé pour émuler des frappes clavier et des mouvements de souris sans que l'utilisateur ne s'en aperçoive et ainsi lancer des programmes malveillants. On retrouve ce dongle dans le Social-Engineer Toolkit[8]. La même année, le jailbreak de la PlayStation 3[9] sur simple clé USB avait fait la joie des gamers et le malheur de Sony. Le dispositif malicieux pouvait créer artificiellement un hub 6 ports USB actifs sur la console afin d'exploiter une faille rendant possible de générer un buffer overflow et autoriser par la suite l'exécution de logiciels non autorisés, homebrews et autres jeux piratés…

L'on peut même remonter à 2005, sur les lieux même du Black Hat, pour trouver une démonstration par David Dewey et Darrin Barrall (SPI Dynamics) d'une attaque contre les drivers USB de Windows XP permettant de prendre le contrôle de l'OS à l'aide d'une clé USB reprogrammée en Trojan (hardware-based Trojan)[10]. Plus récemment, lors du Black Hat 2011, Angelos Stavrou et Zhaohui Wang[11], chercheurs à l'Université George Mason, parvenaient à faire passer un smartphone (Android) pour un clavier afin d'agir directement sur la session de l'utilisateur et entrer des commandes hostiles sur son ordinateur. « *Il est possible d'(ab)user du protocole USB pour connecter n'importe quel appareil sur une plateforme informatique, sans authentification* [nécessaire] », faisaient-ils observer.

Attaques avec dispositifs USB malicieux [12]

ATTAQUES SUR LES DRIVERS USB

En libérant les restrictions de l'OS (*jailbreak*) grâce à un dispositif malicieux (p.ex. PS3 jailbreak), il est possible de modifier les droits en lecture et écriture de ce même OS. L'exécution de code non signé peut ainsi être autorisée.

ATTAQUES VIA HID

En émulant un clavier et/ou une souris à l'insu de l'utilisateur, un dispositif USB corrompu (type PHUKD) peut lancer l'exécution automatique d'un programme ou s'approprier les droits de l'opérateur (potentiellement un administrateur). Il peut également ouvrir un fichier texte, écrire une charge virale codée en base64 et le sauvegarder sur la machine-cible…

ATTAQUES VIA USB MASS STORAGE

En reprogrammant le firmware d'un périphérique USB, un attaquant peut modifier à la volée le contenu d'une partition ou de fichiers. Le principe est de forcer le système à relire un fichier après qu'il ait vérifié sa signature : cette deuxième lecture ne renverra pas les mêmes informations que la première et permettra l'installation du code non autorisé.

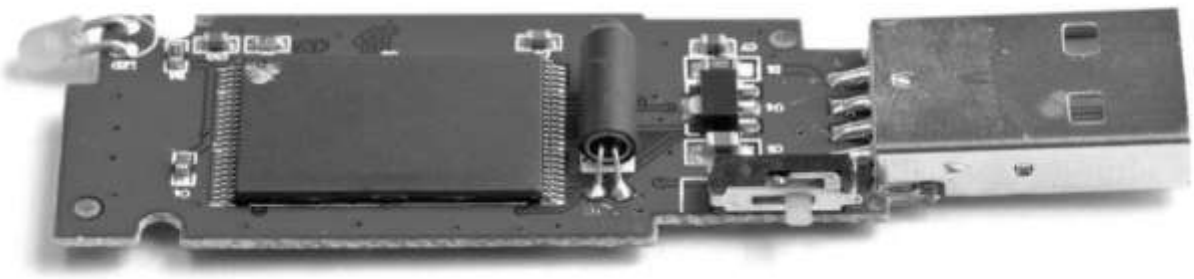
ACQUISITION D'INFORMATION SUR LE SYSTEME HOTE

En fonction de la manière dont l'OS lit son descripteur, une clef USB malicieuse peut découvrir quel est l'OS hôte et ainsi adapter son attaque aux vulnérabilités connues dans chacun des systèmes qu'elle souhaite attaquer.

ATTAQUES DMA & ECOUTE SUR LE BUS

Avec les périphériques USB On-The-Go, capables d'être vus comme un périphérique ou comme un hôte USB, les attaques DMA (Direct Memory Access) seraient devenues possibles. De plus, un périphérique malicieux peut facilement intercepter les informations que reçoivent tous les périphériques connectés sur un contrôleur USB hôte.

Nous reprenons ici les principaux types répertoriés par Benoît Badrigans (2012).



■ '*Fool*' Disclosure ?

Deux mois après l'annonce de la faille BadUSB, Adam Caudill et Brandon Wilson[13] communiquent ce que Karsten Nohl et Jacob Lell avaient préféré taire : le moyen d'exploiter la vulnérabilité.

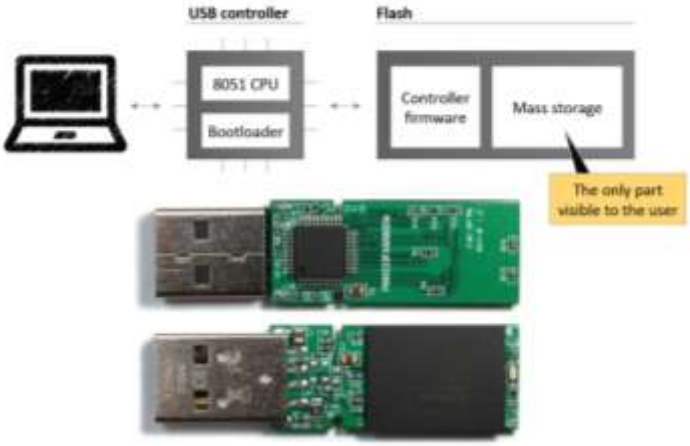


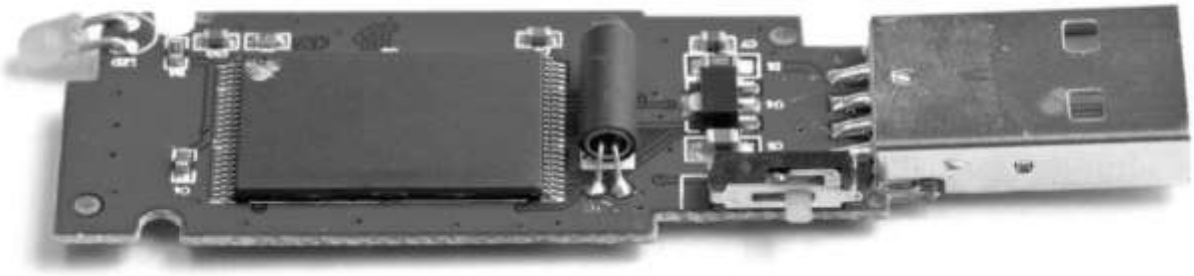
Figure 2 / K. Nohl, S. Krißler, J. Lell, 2014. p.4

Lors de la DerbyCon Hacker Conference 2014, ces deux chercheurs en sécurité informatique (dont l'employeur est quant à lui tenu secret) exposent la méthode complète, depuis la rétroingénierie jusqu'à la mise à jour illicite du firmware, par laquelle ils parviennent à modifier une clé USB 3.0 basée sur un microcontrôleur de la marque Phison Electronics, l'une des plus répandues au monde, également utilisée par Karsten Nohl et Jacob Lell dans leurs expérimentations (cf. fig.2). Comment personnaliser le firmware de manière à le convertir en clavier à l'instar de l'USB Rubber Ducky[14], créer une partition cachée dans le microcontrôleur de la clé qui apparaitra vide (mesure utile à l'exfiltration de données), ou encore modifier le mécanisme de protection par mot de passe de la clé ? Ils y répondent à travers trois démonstrations et publient sur Github[15] le code permettant de les mettre en œuvre.

Objectif de cette divulgation : contraindre les fabricants d'USB à renforcer leur politique de sécurité et appeler les utilisateurs à la prudence. « *Si les seules personnes capables d'utiliser cette faille sont les entités disposant de budgets importants, le problème ne sera jamais corrigé* » arguent Adam Caudill et Brandon Wilson. Mais c'est aussi une porte ouverte au déploiement de ce type d'attaques. « *Grâce à ce code, une grosse partie du travail a désormais été mâchée. Le développement qui reste à faire pour créer une attaque n'est pas si compliqué. Des millions d'informaticiens en seraient capables* », juge Karsten Nohl, interviewé par 01net[16] le 9 octobre 2014.

Bernie Thompson, fondateur de Plugable Technologies (fabricant de périphériques USB), tempère cependant les craintes[17]. Cet ex-Microsoft souligne que pour pirater un ordinateur via un dispositif USB, il faut que celui-ci possède un firmware qui doit pouvoir être mis à jour de manière logicielle (la ROM doit être effaçable pour être réécrite), laquelle doit être non sécurisée : ce n'est pas le cas de tous les périphériques, selon lui. Mais surtout, le code BadUSB doit être conçu spécifiquement pour le microcontrôleur de l'appareil. Aussi, celui publié par Adam Caudill et Brian Wilson vaudrait uniquement pour les périphériques basés sur un microcontrôleur Phison 2251-03. Security Now![18] liste notamment :

- Patriot 8GB Supersonic Xpress,
- Patriot Stellar 64 Gb Phison,
- Kingston DataTraveler 3.0 T111 8GB,
- Silicon power marvel M60 64GB,
- Toshiba TransMemory-MX™ Black 16 GB.



Une faille imparable ?

A la fin de leur conférence, Karsten Nohl et Jacob Lell passent en revue différentes éventualités de défense, tout en pointant leurs limites (cf. fig.3). La seule qui vaille, « simple et efficace » selon eux, serait de désactiver les mises à jour des firmwares. Avis aux fabricants ! Avec WhiteN®, Bertin IT propose une solution de neutralisation des menaces issues de supports amovibles, indépendamment de leur firmware, capable de déjouer les attaques décrites par les deux chercheurs.

■ Implémentation de listes blanches et blocage de périphériques USB

Karsten Nohl et Jacob Lell citent ces possibilités de défense mais font aussitôt observer que les OS ne sont pas encore dotés de mécanismes de liste blanche. C'est pourtant le cas de WhiteN®.

Les mécanismes de liste blanche sont implémentés au cœur de la pile USB pour n'autoriser que certains périphériques USB, préalablement identifiés (un seul clavier, une seule souris, ...). Cette identification est réalisée à partir d'un ensemble d'éléments dont la classe du périphérique, l'identifiant du vendeur et le numéro de série. Corollairement, tout dispositif USB non explicitement autorisé par la politique de sécurité sera bloqué (p.ex. un périphérique d'interface réseau ou une webcam, dont l'opérateur ne devrait pas avoir l'usage).

WhiteN® possède une pile USB minimaliste. Seules trois classes de périphériques sont supportées :

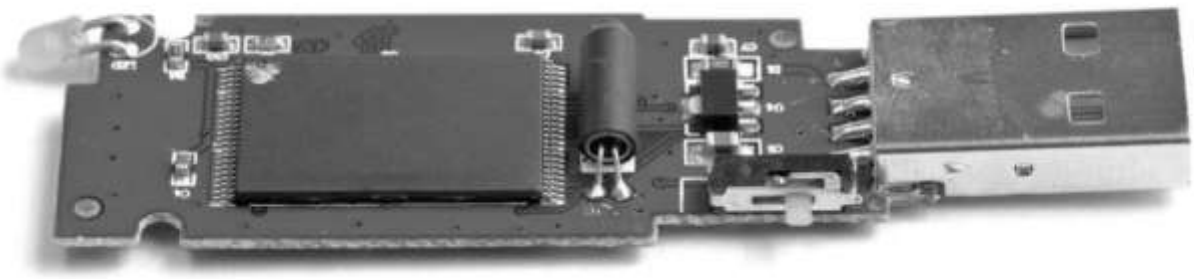
- HID, *Human Interface Device* (p.ex. clavier)
- CCID, *Chip/Smart Card Interface Device* (p.ex. lecteurs de carte à puce)
- MSC, *Mass Storage Class* (p.ex. clé USB)

Aussi, le scénario d'usurpation de carte réseau proposé par Karsten Nohl et Jacob Lell est non jouable sur WhiteN®, car cette classe de périphérique critique n'est pas autorisée. Aucune information ne sera envoyée au dispositif illicite, tout simplement ignoré et par là même neutralisé.

Dans les cas d'émulation de clavier ou d'usurpation d'écran, sans les informations nécessaires à l'autorisation des périphériques en question, l'attaque est impossible. Dans l'hypothèse où l'attaquant se serait procuré par ingénierie sociale les numéros de série et identifiants du vendeur afin d'usurper l'identité d'un appareil autorisé, son champ de nuisance serait fortement limité par les mécanismes de cloisonnement (cf. infra) assurés par WhiteN®.

Protection idea	Limitation
Whitelist USB devices	• USB devices do not always have a unique serial number • OS's don't (yet) have whitelist mechanisms
Block critical device classes, block USB completely	• Obvious usability impact • Very basic device classes can be used for abuse; not much is left of USB when these are blocked
Scan peripheral firmware for malware	• The firmware of a USB device can typically only be read back with the help of that firmware (if at all): A malicious firmware can spoof a legitimate one
Use code signing for firmware updates	• Implementation errors may still allow installing unauthorized firmware upgrades • Secure cryptography is hard to implement on small microcontrollers • Billions of existing devices stay vulnerable
Disable firmware updates in hardware	• **Simple and effective**

Figure 3 / K. Nohl, S. Krißler, J. Lell, 2014. p.20



Par ailleurs, la finesse de paramétrage de WhiteN® permet de mettre en œuvre des heuristiques avancées, telles que le blocage automatique d'un second clavier ou l'autorisation explicite par l'utilisateur de chaque périphérique détecté.

Le blocage complet de l'USB est de plus en plus employé par les entreprises pour limiter les risques liés aux périphériques USB non maîtrisés. Une mesure radicale et non sans impact sur l'utilisabilité, comme le soulignent les deux chercheurs de SRLabs, bien que les périphériques de type PS/2 (clavier et souris) continuent de fonctionner. Dans cette configuration hermétique, WhiteN® maintient la possibilité d'inputs mais de façon totalement contrôlée, grâce à des sas d'entrée sécurisés dans le système d'information qui filtrent toutes les données issues de périphériques USB.

■ Vérification de l'intégrité des firmwares

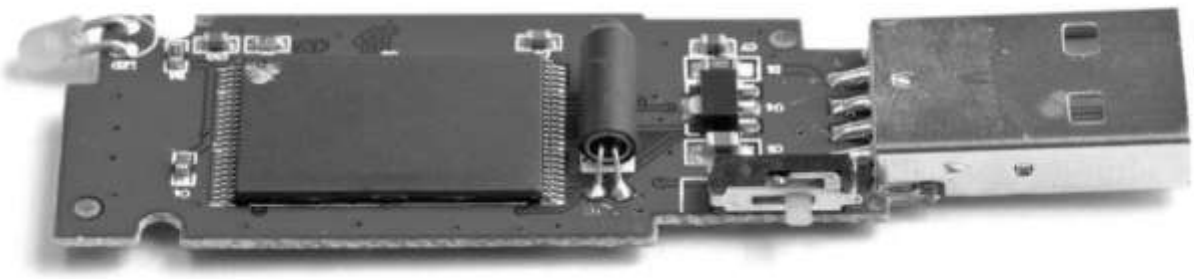
WhiteN® n'opère pas le scan des firmwares des périphériques. Du reste, Karsten Nohl et Jacob Lell font remarquer que le firmware d'un appareil ne peut être relu qu'à l'aide de ce même firmware, lequel peut être malicieux et usurper un firmware approuvé… Autant demander à un menteur s'il ment !

Mais, s'agissant de la reprogrammation des firmwares des périphériques USB intégrés aux postes de travail (p.ex. clavier, touchpad, webcam, ...), une solution consisterait à vérifier l'intégrité de la plateforme en incluant l'ensemble des firmwares. Cette problématique est couverte par les spécifications du Trusted Computing Group[19] (TCG).

Membre contributeur de ce consortium, Bertin IT a réalisé des preuves de concept démontrant la capacité du socle logiciel de WhiteN® à détecter des modifications dans les firmwares de certains périphériques.

■ Contrôle d'accès à base de rôles

Le contrôle d'accès à base de rôles (RBAC : *Role-Based Access Control*) permet d'appliquer des stratégies de sécurité spécifiques, en fonction du profil de droits de l'utilisateur (p.ex. utilisateur, administrateur système, …). Ainsi, un utilisateur disposera des privilèges nécessaires et suffisants à l'exécution d'un travail, ni plus ni moins.



L'architecture de WhiteN® assure cette séparation stricte des rôles mais aussi des environnements utilisateurs et administrateurs. Un périphérique malicieux n'aurait pas plus de droit que l'utilisateur lui-même et ne pourrait escalader les privilèges ou les environnements. Dans le scénario de simulation de clavier, par exemple, ce cloisonnement permet de circonscrire le périmètre de l'attaque.

Mesures côté fabricant

■ Signature de code pour la mise à jour

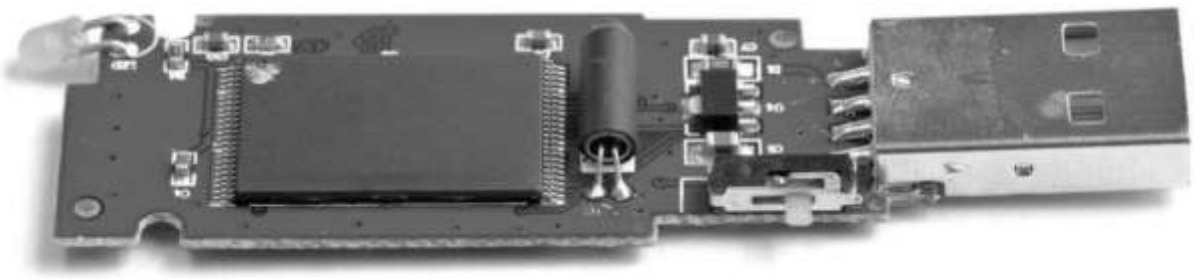
BadUSB s'appuie sur le fait que la très grande majorité des appareils USB n'exige pas de code signé pour autoriser la mise à jour du firmware. Si tel était le cas, un appareil dont le firmware aurait été modifié n'authentifierait pas ce firmware qui ne pourrait alors pas fonctionner.

Certains fabricants, tels que IronKey[20], n'ont pas manqué de signaler que leurs périphériques sont bien dotés d'une protection cryptographique interdisant toute reprogrammation illicite.

■ Désactivation de la mise à jour

Cette mesure, pour « simple et efficace » qu'elle soit selon Karsten Nohl et Jacob Lell, n'est pas satisfaisante pour une personne en charge de la sécurité informatique d'une entreprise car celle-ci ne saurait contrôler toutes les clés USB utilisées par les employés. Elle ne l'est pas davantage pour le grand public qui ne dispose pas des compétences techniques pour la mettre en œuvre.

Aussi, la désactivation de la mise à jour des firmwares se trouve sous la responsabilité du fabricant.



■ Vers un label 'firmware sécurisé' ?

Le 12 novembre dernier, lors de la conférence PacSec à Tokyo, Karsten Nohl dévoilait les résultats d'une étude de vulnérabilité[21] menée sur différents périphériques USB présents sur le marché. Il a d'abord analysé, avec ses confrères de SRLabs, les datasheets des microcontrôleurs diffusés par les huit plus grands vendeurs mondiaux (Microchip, Cypress, Alcor, Renesas, Genesys Logic, ASMedia, Phison, FTDI). Puis, il a examiné les hardwares de 33 périphériques de six classes différentes (hub, interface, webcam, adaptateurs SD et SATA). Cette phase n'a pas toujours permis d'identifier la marque du composant (en particulier pour les périphériques de type HID), certains ne comportant aucune référence.

Il ressort de ces deux analyses (cf. fig.4) que près de la moitié des firmwares, toutes classes de périphériques confondues parmi les dispositifs testés, sont reprogrammables et donc sujets à la faille BadUSB. L'on pourrait se tranquilliser quelque peu à l'idée qu'une « bonne moitié » existe. Mais, « *le plus effrayant est que nous ne pouvons fournir une liste des appareils sûrs* », confie Karsten Nohl à Wired[22]. Non seulement des disparités s'observent au sein d'une même marque (p.ex. certaines puces de Genesys Logic sont sécurisées, d'autres non), mais les fabricants de périphériques ont tendance à changer de fournisseurs de composants électroniques, d'un modèle à un autre, voire pour un même produit, au gré de l'offre et de la demande.

C'est ce que révèle une étude de Richard Harman[23] présentée à la conférence Shmoocon de janvier 2014. On y découvre, par exemple, que le fabricant de clés USB Kingston Digital utilise les microcontrôleurs de six fournisseurs différents. On en compte quatre chez Silicon Power, trois chez Trend Micro… Il s'avère donc impossible de déterminer a priori si le firmware contenu dans un appareil appartient à la catégorie non vulnérable, à moins de le démonter.

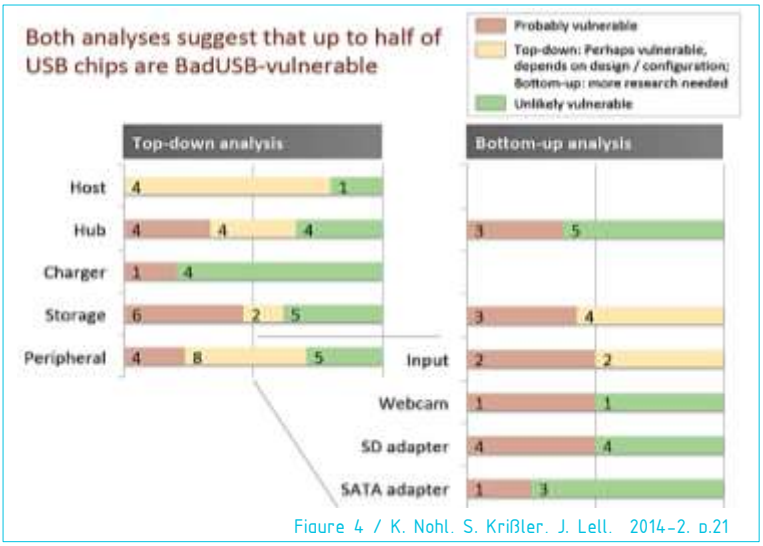
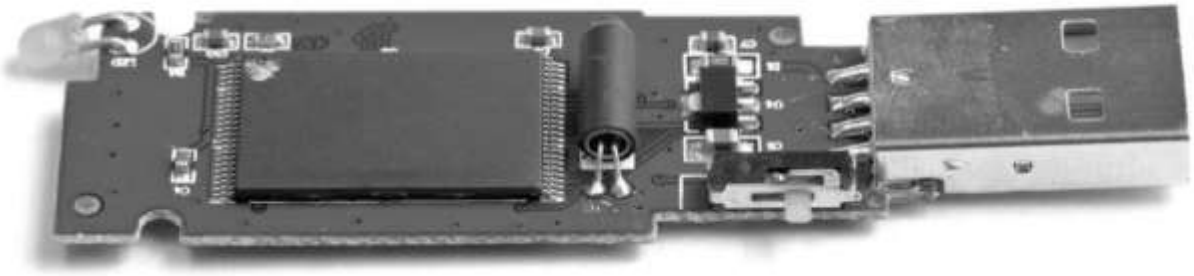


Figure 4 / K. Nohl, S. Krißler, J. Lell, 2014-2, p.21

On le voit, la faille BadUSB ne jette pas seulement le doute sur la sécurité de milliards de périphériques, mais aussi sur les pratiques industrielles de leurs fabricants. Malheureusement, un label 'firmware sécurisé' n'est pas pour demain. Certains l'ont bien compris : l'OS FreeBSD[24] s'est renforcé avec une option de désactivation de l'énumération USB (un périphérique nouvellement connecté ne peut pas s'identifier auprès de l'hôte) et G DATA[25] propose désormais un logiciel permettant de contrôler l'accès d'un nouveau clavier sur un système afin de parer l'attaque par émulation de frappes. Deux types de protection d'ores et déjà assurés par WhiteN®, la station de neutralisation des menaces USB développée par Bertin IT.



Références

1- Porras, Phillip, Saidi, Hassen, Yegneswaran Vinod. *An analysis of Conficker's logic and rendez-vous points.* SRI International Technical Report, 2009. ▸ http://mtc.sri.com/Conficker/

2- *Stuxnet*. Wiki. ▸ http://en.wikipedia.org/wiki/Stuxnet/

3- Pour une revue des risques associés aux clefs USB : Vallée, Luc. *Clef USB : pratiques mais risquées*. Magazine Sécurité de l'Information, 2011, n°11, p. 2-4. ▸ http://www.dgdr.cnrs.fr/fsd/securite-systemes/revues-pdf/Si11.pdf
-CERTA, Centre d'Expertise gouvernemental de Réponse et de Traitement des Attaques informatiques. *Risques associés aux clés* USB. Première version : 2006. Dernière version : 2009 ▸ http://www.cert.ssi.gouv.fr/site/CERTA-2006-INF-006/

4- Nohl, Karsten, Krißler, Sascha, Lell, Jacob. SRLabs. *BadUSB – On accessories that turn evil*. Black Hat, 2014.
▸ https://srlabs.de/blog/wp-content/uploads/2014/07/SRLabs-BadUSB-BlackHat-v1.pdf
A lire aussi : *Why the security of USB is fundamentally broken*. Wired, 2014. ▸ http://www.wired.com/2014/07/usb-security/

5- Security Research Labs (SRLabs). *Turning USB peripherals into BadUSB*. 2014. ▸ https://srlabs.de/badusb/

6- Co-écrit avec Tristan Vanel, Bitdefender. *BadUSB : beaucoup de bruit pour presque rien ?* D4v1d, 2014.
▸ http://d4v1d.me/badusb-beaucoup-de-bruit-pour-presque-rien/

7- Crenshaw, Adrian. TrustedSec. *Programmable HID USB Keystroke Dongle: Using the Teensy as a pen testing device*. DEF CON, 2010.
▸ https://www.defcon.org/images/defcon-18/dc-18-presentations/Crenshaw/DEFCON-18-Crenshaw-PHID-USB-Device.pdf
8- *Social-Engineer Toolkit v0.6.1 Teensy USB HID Attack Vector*. TrustedSec, 2010.
▸ https://www.trustedsec.com/august-2010/social-engineer-toolkit-v0-6-1-teensy-usb-hid-attack-vector/

9- *PSJailbreak Exploit Reverse Engineering*. PS3 Wiki. ▸
http://www.psdevwiki.com/ps3/PSJailbreak_Exploit_Payload_Reverse_Engineering/
10- Dewey, David, Barrall, Darrin. SPI Dynamics. *Plug and Root: The USB Key to the Kingdom*. Black Hat, 2005.
▸ http://www.blackhat.com/presentations/bh-usa-05/BH_US_05-Barrall-Dewey.pdf.

11- Stavrou, Angelos, Wang, Zhaohui. *Exploiting Smart-Phone USB Connectivity For Fun And Profit.* Black Hat, 2011. ▸ https://media.blackhat.com/bh-dc-11/Stavrou-Wang/BlackHat_DC_2011_Stavrou_Zhaohui_USB_exploits-Slides.pdf
12- Badrigans, Benoît. *Attaques applicatives via périphériques USB modifiés : infection virale et fuites d'informations.* SSTIC, 2013.
▸ sstic.org/2013/presentation/Attaques_applicatives_via_peripheriques_USB_modifies_infection_virale_et_fuites_d_informations/

13- Caudill , Adam, Wilson, Brandon. *Making BadUSB Work For You*. Derbycon, 2014. ▸ http://fr.slideshare.net/adam_caudill/derby-con2014presentation/

14- *USB Rubber Ducky - The Original Keystroke Injection Tool*. ▸ www.usbrubberducky.com

15- Caudill , Adam, Wilson, Brandon. *Phison 2251-03 (2303) Custom Firmware & Existing Firmware Patches.* Github, 2014.
▸ https://github.com/adamcaudill/Psychson/

16- Kallenborn, Gilbert. *Les attaques par USB, désormais à la portée de « millions de développeurs »*. 01Net, 2014.
▸ http://www.01net.com/editorial/628392/les-attaques-par-usb-desormais-a-la-portee-de-millions-de-developpeurs/

17- Thompson, Bernie. *What BadUSB Is and Isn't*. Plugable, 2014. ▸ http://plugable.com/2014/10/06/badusb/

18- *BadUSB returns.* Security Now! #476 - 10-07-14 Q&A #198, 2014. ▸ https://www.grc.com/sn/SN-476-Notes.pdf

19- Trusted Computing Group – TCG. ▸ http://www.trustedcomputinggroup.org

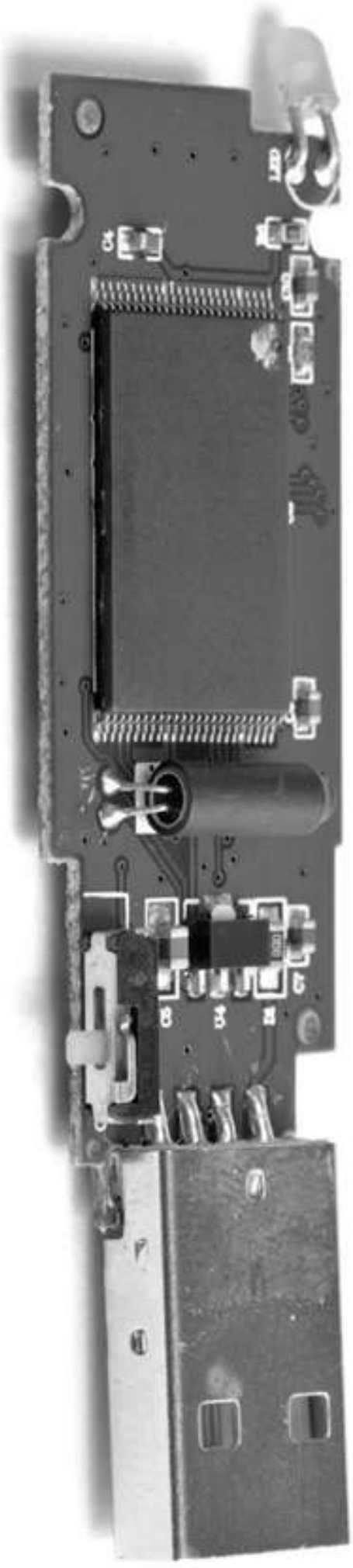
20- *Ironkey™ Secure USB Devices* ▸ http://www.ironkey.com/en-US/solutions/protect-against-badusb.html

21- Nohl, Karsten, Krißler, Sascha, Lell, Jacob. SRLabs. *BadUSB – On accessories that turn evil*. PacSec, 2014.
▸ https://srlabs.de/blog/wp-content/uploads/2014/11/SRLabs-BadUSB-Pacsec-v2.pdf - Résultats détaillés de l'étude ▸ https://opensource.srlabs.de/projects/badusb
22- Greenberg, Andy. *Only Half of USB Devices Have an Unpatchable Flaw, But No One Knows Which Half*. Wired, 2014.
▸ http://www.wired.com/2014/11/badusb-only-affects-half-of-usbs/

23- Harman, Richard. *Controlling USB Flash Drive Controllers: Exposé of hidden features*. Shmoocon, 2014. ▸
http://fr.slideshare.net/xabean/controlling-usb-flash-drive-controllers-expose-of-hidden-features/

24- FreeBSD ▸ https://www.freebsd.org/fr/

25- *Sécurisé contre les attaques USB*. G DATA ▸ https://www.gdata.fr/fr-usb-keyboard-guard.html



Dirigé par **Béatrice Bacconnet**, Bertin IT (Bertin Technologies - Groupe CNIM) est expert et fournisseur de solutions logicielles de pointe pour la cybersécurité, la cyber intelligence et la veille en gestion de crise. Son offre, tournée vers l'anticipation et la protection, recouvre la Sécurité des Systèmes d'Information sensibles et infrastructures critiques, et l'analyse en profondeur des sources ouvertes multimédias multilingues (Web, TV, Radio) à des fins de détection des menaces et de vigilance situationnelle.

A travers la filiale Vecsys, spécialiste de la reconnaissance vocale, Bertin IT propose également des solutions et services dédiés à la transcription automatique multilingue de sources audio & vidéo, à la production de ressources linguistiques et à la commande vocale embarquée.

Ce document est téléchargeable
sur notre site web www.bertin-it.com

Vous avez aimé cet article ? Partagez-le !

www.bertin-it.com