L'architecture x86 est-elle sûre ?

Laurent Bloch [1]

En octobre 2015 la chercheuse en sécurité informatique Joanna Rutkowska a publié un article retentissant intitulé Intel x86 considered harmful [2]*, qui étudie principalement les mécanismes peu connus du démarrage du microprocesseur, avant le lancement du système d'exploitation, et qui fait l'inventaire des failles de sécurité potentielles exploitables par un attaquant.*

Le résultat est très impressionnant. Laurent Bloch propose, pour 1024, une analyse précise de cet article.

Lors de la mise sous tension d'un ordinateur il faut que lui soit donnée une information d'amorçage : que faire pour commencer ? Plus précisément : quelle est l'adresse de la première instruction à exécuter ? On imagine bien que la modification de cette adresse par un attaquant pourrait compromettre irrémédiablement tout le fonctionnement ultérieur de quelque logiciel que ce soit, et notamment de tout système d'exploitation, aussi sûr soit-il. Cette question du démarrage est donc cruciale, d'autant plus qu'elle est généralement mal documentée et mal connue. C'est tout le mérite de Joanna Rutkowska d'avoir appliqué ses capacités d'investigation et de critique à ce processus, et d'avoir ainsi montré qu'en dépit d'une complexité considérable ajoutée au cours des dernières années pour en améliorer la sécurité, le système est encore vulnérable. D'où le titre de l'article.

1. lb@laurentbloch.org

2. Joanna Rutkowska. *Intel x86 considered harmful*. The Invisible Things, October 2015, http://blog.invisiblethings.org/papers/2015/x86_harmful.pdf.

Au début était le BIOS

Lorsqu'IBM a lancé les premiers PC en 1981 sur une base de micro-processeurs Intel 8088, il en a publié les spécifications, ce qui a permis que les réalisations ultérieures, éventuellement par d'autres entreprises, respectent une architecture homogène. Selon ces règles architecturales, la carte mère de l'ordinateur reçoit un composant distinct du processeur [3] qui contient un programme de démarrage, le *Basic Input Output System* (BIOS). Le circuit de mise sous tension est câblé de façon à lancer l'exécution de la première instruction du BIOS. Le rôle du BIOS consiste à initialiser les composants de la carte mère, à identifier les dispositifs périphériques connectés à l'ordinateur (clavier, écran, souris, disque...), à les mettre en relation avec le processeur. Le BIOS avait été inventé en 1975 par Gary Kildall pour le système CP/M.

L'utilisateur peut accéder au menu de configuration du BIOS pour sélectionner certaines options, par exemple l'ordre selon lequel scruter les périphériques de démarrage possibles pour voir s'ils contiennent un logiciel d'amorçage et un système d'exploitation. Puis le BIOS va charger à partir d'un emplacement convenu sur le périphérique de démarrage choisi (disque dur, clé USB, CD-ROM...) le logiciel d'amorçage (par exemple GRUB dans le cas d'un périphérique de démarrage comportant Linux et éventuellement d'autres systèmes). Le logiciel d'amorçage va permettre à l'utilisateur de choisir le système d'exploitation à lancer s'il y en a plusieurs sur le périphérique de démarrage, et il va le faire démarrer effectivement. Inutile de dire que tout détournement malveillant d'une étape de ce processus peut corrompre toutes les opérations ultérieures, parce qu'à ce stade du démarrage aucun dispositif de protection matériel ou logiciel n'est activé, et qu'ainsi le BIOS a accès sans restriction en mode privilégié à toute la mémoire et à tous les périphériques. Il est par exemple concevable qu'un BIOS corrompu par un attaquant lance une version corrompue du système d'exploitation.

Depuis 2014 la plupart des ordinateurs sont livrés avec un programme de démarrage dit *Unified Extensible Firmware Interface* (UEFI), qui est essentiellement un BIOS écrit selon des principes plus modernes que ses prédécesseurs et doté de fonctions plus étendues, telles que la possibilité de disques de plus grande capacité avec un plus grand nombre de partitions. Un inconvénient majeur d'UEFI est la décision de Microsoft d'imposer aux vendeurs d'ordinateurs certifiés pour Windows 8 ou 10 l'obligation de livrer leur matériel configuré pour démarrer en mode dit *Secure Boot*, contrôlé par une clé de chiffrement privée détenue par Microsoft et utilisée pour signer le noyau du système. Cette situation rend nettement plus difficile l'installation d'un système d'exploitation libre sur ces machines.

3. Historiquement il s'agissait d'un élément de mémoire morte (*Read-Only Memory*, ROM), aujourd'hui il s'agit d'une mémoire flash analogue à celle des clés USB, qui peut être modifiée par logiciel, ce qui est pratique mais introduit un facteur de risque. Le premier virus BIOS, nommé « Chernobyl » ou CIH selon les initiales de son auteur Chen Ing Hau, date du 26 avril 1999.

Confiance et sécurité ?

Joanna Rutkowska nous avertit d'une confusion terminologique : le terme « *trusted* » (de confiance) apposé à un système informatique nous emplit d'un sentiment de sécurité, alors qu'au contraire la crainte devrait nous étreindre. Si celui qui met ce système entre nos mains nous dit qu'il est « de confiance » sans nous donner les moyens de le vérifier, donc en nous contraignant à une confiance *aveugle*, cela signifie que la défaillance de ce système peut détruire *absolument* sa sécurité et tout ce qui en dépend.

Il convient donc de se méfier de ce qui est *trusted*, c'est-à-dire de confiance obligée, et d'en réduire le périmètre au minimum. Ce périmètre se nomme « *trusted computing base* » (TCB), il faut en faire sortir le plus possible d'éléments techniques. Comme va nous le montrer Joanna Rutkowska, l'évolution récente de la technologie des processeurs Intel contribue au contraire à élargir démesurément le périmètre technique auquel nous devons faire confiance parce que nous ne disposons d'aucun moyen de savoir comment fonctionne ce qu'il renferme.

Le Bios à la source de toute confiance

Ainsi que nous l'avons signalé ci-dessus, sans confiance dans le BIOS on ne peut plus avoir confiance en rien dans l'ordinateur, son système d'exploitation (OS) et ses données. Joanna Rutkowska précise les raisons de cet état de fait :

— le BIOS est le premier programme à s'exécuter sur le processeur, ce qui lui permet de corrompre l'image de l'OS ou de l'hyperviseur chargée ultérieurement ;

— le BIOS a accès en mode privilégié à l'ensemble du matériel, ce qui lui permet de programmer des actions malfaisantes (par exemple les attaques dites DMA et *Evil Maid*) ;

— le BIOS fournit le code qui s'exécute selon le *System Management Mode* (SMM) ; or le code SMM peut s'exécuter pendant toute la période d'activité du système, d'où un risque majeur induit par sa corruption éventuelle ; Xeno Kovah et Corey Kallenberg ont prouvé la réalité de ce risque en écrivant le *rootkit* SMM LightEater [4] capable de voler des clés privées en mémoire vive pendant l'exécution du système Tails OS [5], pourtant spécialement conçu à partir d'un système Linux Debian pour la sécurité (ses communications réseau utilisent Tor, Edward Snowden l'utilise pour cette raison).

4. Xeno Kovah, Corey Kallenberg. *How Many Million BIOSes Would you Like to Infect*? LegbaCore, June 2015, `http://legbacore.com/Research_files/HowManyMillionBIOSesWouldYouLikeToInfect_Whitepaper_v1.pdf`.

5. *Tails : The amnesic incognito live system*. The TAILS Project, November 2015, `https://tails.boum.org/`.

Le BIOS est donc un élément ultra-sensible de tout système informatique à base de processeur x86, or comme nous l'avons écrit plus haut il est modifiable assez facilement, l'article de Joanna Rutkowska indique plusieurs procédés pour ce faire, et il n'existe à l'heure actuelle aucun moyen de s'affranchir de cette dépendance vis-à-vis du BIOS, en d'autres termes de faire en sorte qu'il puisse être considéré comme non-fiable sans que ce soit grave.

Au-delà de cette analyse des pouvoirs du BIOS, Joanna Rutkowska commence à nous faire prendre conscience de la présence d'une quantité considérable de logiciel actif pendant toute la vie du système et hors de tout contrôle du système d'exploitation. Il serait donc vital que ce logiciel soit vérifiable et sûr. Nous allons voir qu'il n'en est rien.

Multiples sous-systèmes en micro-code

Pour corriger certaines failles de sécurité telles que celles mentionnées ci-dessus, les ingénieurs d'Intel ont multiplié les dispositifs : *Trusted Platform Module* (TPM), *Trusted Execution Technology* (TXT), exécution du code SMM *(System Management Mode)* sous contrôle d'un hyperviseur, *Active Management Technology* (AMT), *Boot Guard*, *Secure Boot* pour UEFI, et pour couronner l'édifice *Intel Management Engine* (ME). Il s'agit en fait de systèmes implantés dans le micro-code du processeur, c'est-à-dire hors de portée de l'utilisateur même muni de tous les privilèges sur son système. L'article de notre auteure en fait une analyse approfondie.

Ce qui est à noter, c'est que des systèmes comme AMT et ME sont actifs en permanence, *y compris lorsque l'ordinateur est arrêté* ! C'est dire à quel niveau de contrôle arrivent ces techniques, dont la puissance est telle qu'il est à souhaiter qu'elles ne puissent pas être détournées par des malfaisants.

Intel Management Engine *(ME)*

Nous ne traiterons pas ici de tous ces dispositifs, pour lesquels nous renvoyons à l'article, et nous n'évoquerons que ME (*Intel Management Engine*), parce que c'est le dernier en date et qu'il englobe en quelque sorte tous les autres. ME est un petit ordinateur incorporé à *tous* les processeurs Intel contemporains. Il est impossible de l'enlever ou de le désactiver, et d'ailleurs, à supposer que l'on trouve un moyen de l'inhiber, le processeur deviendrait pratiquement inutilisable parce que beaucoup de ses fonctions dépendent de ME.

Plus qu'un simple microcontrôleur, ME est une infrastructure d'accueil complète pour toutes sortes de sous-systèmes, et de fait AMT est désormais implanté sur ME, ainsi que *Boot Guard*, PTT (*Platform Trust Technology*, la version pour ME de TPM), et d'autres à venir. ME procure bien sûr un hyperviseur qui permet par exemple l'exécution de code SMM en bac à sable (pour une récapitulation de tous

ces sigles on pourra se reporter au document *Intel Hardware-based Security Technologies for Intelligent Retail Devices* [6]).

ME partout et pour tout ?

Joanna Rutkowska souligne les effets pervers de cette prise de contrôle par ME de tous les aspects du fonctionnement du système.

D'abord, imposer à tous les utilisateurs de processeurs Intel une technologie fermée et opaque en prétendant qu'elle offrirait un niveau de sécurité inégalable, sans discussion possible, est une attitude arrogante et peu convaincante sur le fond.

Ensuite, si cette idée s'impose, et il semble que l'on n'ait guère le choix (AMD développe des technologies comparables sous le nom *Platform Security Processor*, PSP), cela conduira à ce qu'elle appelle la « zombification » des systèmes d'exploitation tels que Windows, Linux, OS-X, etc., réduits au rôle d'interfaces avec l'utilisateur pour balader des fenêtres, réagir aux déplacements de souris et jouer de la musique, cependant que les véritables traitements de données seront effectués derrière les portes closes (par des clés de chiffrement détenues par Intel) de ME, sans que l'utilisateur sache quels sont les algorithmes utilisés, et avec quel niveau de sécurité.

Comment savoir, par exemple, si Intel n'a pas décidé de confier le séquestre de ses clés de chiffrement à un tiers de confiance, par exemple une agence de sécurité gouvernementale américaine ? ME pourra-t-il filtrer et analyser nos messages électroniques, nos communications par Skype, nos recherches sur le Web ? Le générateur de nombres pseudo-aléatoires de ME comporte-t-il des faiblesses, voulues ou non ? Répondre à de telles questions est déjà difficile aujourd'hui, mais Joanna Rutkowska attire notre attention sur le fait qu'avec ME la rétro-ingénierie nécessaire à leur compréhension et à leur analyse sera d'une difficulté accrue d'un ordre de grandeur.

Et si ME était corrompu ?

C'est la question qu'il faut toujours se poser à propos d'un système tout-puissant : tant que ses actions sont bénéfiques tout va bien [7], mais s'il est détourné vers des actions maléfiques, intentionnellement ou par suite d'une attaque réussie, alors l'empire du mal risque d'être absolu.

Or, pour qui voudrait entreprendre une action maléfique, ME est l'infrastructure idéale, nous dit Joanna Rutkowska : un *rootkit* implanté dans ME aurait le contrôle total des traitements effectués par le système et des données traitées (y compris les clés privées en transit par la mémoire), et il serait pratiquement indétectable. Que ce *rootkit* soit implanté par la mafia ou par la NSA ne change rien au problème.

6. http://www.intel.com/content/www/us/en/intelligent-systems/shark-bay/security-technologies-4th-gen-core-retail-paper.html

7. Sous réserve d'un accord unanime pour déterminer ce qui est bénéfique et ce qui ne l'est pas, et l'on sait que ce n'est vrai que dans l'imagination des dictateurs totalitaires, c'est même la définition du totalitarisme.

Idées pour un système plus sûr

La démarche de Joanna Rutkowska lui permet d'élaborer les idées qui mènent à un système plus sûr, par exemple :

— utilisation intensive des techniques d'isolation des différents artefacts fonctionnels les uns par rapport aux autres : virtualisation et bac à sable (*sandboxing*) en particulier ;

— exécuter en mode non-privilégié tout ce qui peut l'être, et son article démontre que c'est possible pour presque tout ce qui a trait aux périphériques, ce qui annulerait les menaces de type *Evil Maid* et DMA par exemple.

Joanna Rutkowska ne s'est pas contentée de proférer des idées, qu'elle réunit sous le vocable compartimentation, elle les a mises en pratique en organisant le laboratoire *The Invisible Things* qui a créé le système d'exploitation Qubes OS [8] fondé sur ces principes. On remarquera que cette idée de décomposer les systèmes en sous-systèmes le plus possible indépendants les uns des autres et dotés de privilèges réglés au minimum rejoint le courant des micro-noyaux, une technologie un peu oubliée mais dont les qualités de modularité, de flexibilité et d'aptitude au calcul réparti devraient permettre le retour au premier plan.

Elle constate avec dépit que de façon générale les industriels et les éditeurs de systèmes d'exploitation ne suivent pas ces principes et continuent à produire des systèmes monolithiques et donc vulnérables.

Il n'en reste pas moins que même en supposant toutes ces idées mises en œuvre, la question du démarrage reste entière, parce qu'elle est entre les mains des industriels qui conçoivent et fabriquent processeurs et cartes mères. La multiplication par Intel de technologies concurrentes ou complémentaires ne fait que compliquer la question : *Trusted Platform Module* (TPM), *Trusted Execution Technology* (TXT), exécution du code SMM *(System Management Mode)* sous contrôle d'un hyperviseur, *Boot Guard*, *Secure Boot* pour UEFI, et pour couronner l'édifice *Intel Management Engine* (ME), qui est incorporé de façon irréversible à *tous* les processeurs Intel contemporains et qui est un véritable système d'exploitation implanté en micro-code et qui vit sous le système d'exploitation que l'utilisateur croit avoir choisi.

Avec *Intel Management Engine* le vrai système d'exploitation est dans les entrailles du processeur, et Windows, Linux ou OS-X ne sont plus que des systèmes de gestion de fenêtres, d'affichage de vidéo et de diffusion de musique. Ce qui est grave dans tout cela, c'est que non seulement l'utilisateur (et même le fabricant d'ordinateur) est privé de toute liberté de choix de son système, mais qu'en outre le système imposé n'offre pas et ne peut pas offrir les garanties de sécurité auxquelles il prétend.

8. `https://www.qubes-os.org/`