Introduction à l'informatique confidentielle

Posted on **28/02/2023** by **Fabien Petitcolas**

On considère généralement que les données peuvent être dans trois états. Celles stockées, par exemple sur un disque dur ou dans une base de données, sont dites « au repos », celles envoyées d'un ordinateur à un autre, par exemple via un réseau, sont « en mouvement, » et les données traitées par le microprocesseur sont « en cours d'utilisation ».

Aujourd'hui, des primitives cryptographiques sont largement déployées pour protéger les données dans les deux premiers états, assurant intégrité et confidentialité. Cependant, le renforcement des réglementations sur la protection des données et de la vie privée et l'augmentation des cyber-menaces de longue durée a fait naître le besoin de protéger les données en cours d'utilisation.

Malheureusement cette protection des données en cours d'utilisation reste difficile, en particulier dans le contexte de **l'informatique en nuage** : il s'agit en effet d'exécuter un logiciel avec certaines garanties de sécurité sur un ordinateur distant qui peut être détenu et géré par une partie non fiable voire malintentionnée.

De nombreuses attaques

Les attaques possibles sont nombreuses.

Au **niveau logiciel** on compte toutes les attaques permettant de compromettre l'intégrité du système, que ce soit le système d'exploitation, l'hyperviseur ou le micrologiciel (« *firmware* »). Celles-ci sont rendues possibles via divers types de rançongiciels ou d'outils de piratage, via des pilotes de périphériques compromis, via l'exploitation d'une vulnérabilité du jour zéro, de l'injection d'erreurs logiques, ou encore, à cause d'employés malveillants ou de mauvaises pratiques de fabrication, etc. [1]–[8]. On compte également l'accès aux données à l'aide de canaux auxiliaires (« *side-channel attacks* »), ou celui dû à des erreurs de configuration ou de paramétrage du système. Enfin, plus rares, les attaques cryptographiques, suite à une analyse poussée d'un algorithme ou à l'exploitation d'une percée mathématique sont possibles.

Au **niveau physique**, on peut envisager une administratrice système ou un prestataire externe accédant sans autorisation au matériel, pouvant alors se connecter directement au circuit imprimé et copier des informations par accès direct à la mémoire vive (DIMM DMA), espionner le lien à débit de données double (DDR), surveiller le bus informatique et du cache, etc.

Technologies principales

L'informatique confidentielle (« *confidential computing* ») vise à apporter une solution technologique à ces problèmes à travers trois méthodes principales :

- Les **environnements d'exécution de confiance** (« *trusted execution environments* » – TEE) s'appuient sur une combinaison de composants matériels sécurisés, de mécanismes d'attestation et de logiciels conçus pour utiliser ces fonctionnalités matérielles[1].
- Le **chiffrement homomorphe** (« *homomorphic encryption* » – HE), partiel ou total, concerne le calcul direct sur des données chiffrées [11] et peut aider pour certaines classes de problèmes, mais engendre un coût de par la surcharge de calcul liée au chiffrement spécifique [12].
- Le **calcul multipartite sécurisé** [13] (« *secure multiparty computation* » - MPC) est une classe de méthodes cryptographiques qui permet à des parties mutuellement méfiantes de calculer conjointement des fonctions sur leurs données d'entrée sans révéler celles-ci à d'autres parties. Ce type de calcul a déjà été décrit sur cet article de blog.

Bien entendu, il est possible de combiner ces technologies comme par exemple les environnements d'exécution de confiance avec le chiffrement homomorphe (par ex. [14]).

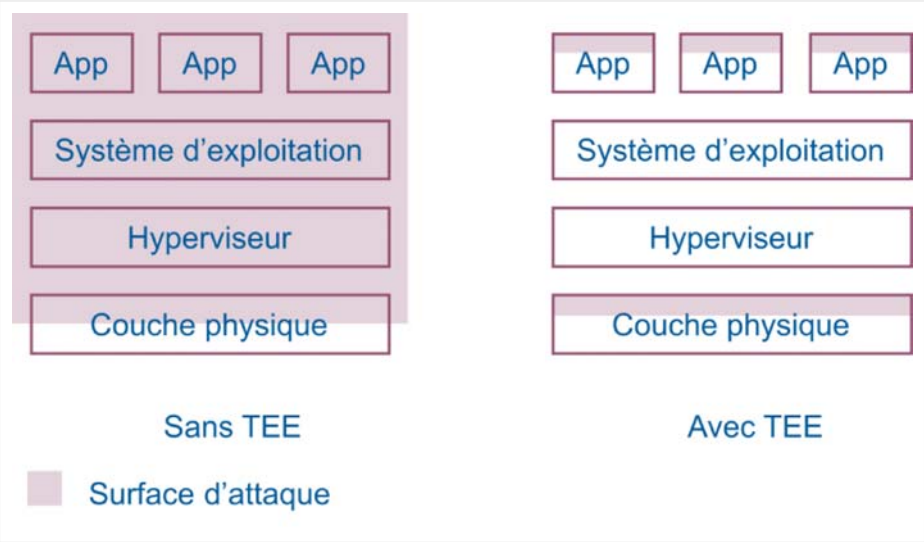
Dans la suite nous nous concentrerons sur le premier type d'informatique confidentielle : les environnements d'exécution de confiance.

Réduction de la surface d'attaque

En règle générale, la pile complète d'une plate-forme informatique est composée de plusieurs technologies provenant de nombreux fournisseurs différents, ce qui entraîne un manque de mise en œuvre cohérente des contrôles de sécurité.

Cependant, la couche physique matérielle sur laquelle l'ensemble du système est construit reste la première couche de protection. Par conséquent une façon d'améliorer la sécurité globale d'un système – y compris en empêchant l'accès aux données – est d'augmenter la sécurité de la partie physique.

C'est sur cet argument que se basent les TEE qui visent à protéger l'exécution du code des applications contre les attaques de code privilégié et certaines attaques physiques en réduisant la base d'information de confiance et la surface d'attaque (Figure 1).



— Figure 1 - Surface d'attaque typique d'un système sans (à gauche) et avec TEE (à droite).

Dans certains cas, la petitesse du TEE permet l'application de méthodes formelles (basée sur des constructions mathématiques) et la vérification de la conformité de la mise en œuvre par rapport aux spécifications. Cela permet de renforcer le niveau de confiance placé dans le TEE.

Les TEE

Les TEE sont des systèmes de sécurité qui s'exécutent parallèlement aux environnements d'exécution réguliers (y compris le système d'exploitation comme Linux ou Windows). Ils fournissent une zone sûre pour protéger les actifs (par ex., les secrets sensibles, les chaînes d'authentification) et ont également pour but d'assurer que les opérations effectuées en leur sein et les données associées ne peuvent pas être visualisées à l'extérieur, pas même par un logiciel privilégié (par ex., hyperviseur, système d'exploitation) ou un débogueur. Combinés à un mécanisme d'attestation (voir ci-dessous) ils assurent que le code qu'ils exécutent ne peut pas être modifié ou remplacé sans l'autorisation de l'utilisateur.

Un TEE peut être utilisé pour fournir une isolation de mémoire, empêchant les applications partageant le même matériel sous-jacent de lire la mémoire allouée aux autres. Certains TEE permettent d'isoler des applications entières dans leurs propres enclaves plutôt que de protéger uniquement des opérations ou de la mémoire spécifique.

L'application des propriétés mentionnées ci-dessus est basée sur un matériel physique spécifique, en intégrant tout ou partie des ressources nécessaires au bon fonctionnement du TEE dans un même boitier, ou en utilisant des ressources partagées par les autres composants disponibles sur le circuit imprimé de la machine (par ex. la mémoire vive). Dans ce dernier cas une isolation par chiffrement est nécessaire.

Tous les TEE ne sont pas identiques et différents fabricants peuvent proposer différentes mise en œuvre de TEE avec différentes propriétés de sécurité, différentes fonctionnalités et différents mécanismes de contrôle pour fonctionner sur les application sécurisées[2]. En effet certains TEE s'adressent plutôt à des téléphones (par ex. la technologie TrustZone d'ARM) tandis que d'autres à des ordinateurs de bureau ou des ordinateurs formant une infrastructure en nuage (par ex. les technologies SGX d'Intel SG ou SEV d'AMD).

Plusieurs fournisseurs d'infrastructure informatique en nuage offrent la possibilité d'utiliser ces technologies à celles ou ceux ne disposant pas de ce matériel. Par exemple, Microsoft a le produit Azure Confidential Computing offrant SGX et SEV. Amazon, met en œuvre sa propre technologie TEE appelée Nitro sur son infrastructure AWS.

Exigences de sécurité

Les TEE sont conçus pour résister à un large éventail de menaces logicielles du système d'exploitation standard, y compris les menaces au niveau du noyau (par ex. [16]) provenant d'un périphérique rooté ainsi qu'à certaines attaques physiques. Les TEE font l'objet d'une analyse de sécurité rigoureuse dans le cadre des « Critères communs » [17], [18], mais ils manquent encore généralement de défenses contre les attaques matérielles complexes. Par exemple, les attaques qui « *nécessitent généralement un accès à long terme et/ou invasif au matériel, y compris les techniques de grattage de puces et les sondes de microscope électronique* » sont explicitement hors du champ d'application des exigences du *Confidential Computing Consortium* [19].

Les exigences de sécurité d'un TEE se résument ainsi (voir [9] pour plus de détails) :

- L'objectif principal d'un TEE est de **protéger ses actifs** contre les attaques à travers le reste de l'environnement d'exécution. Ceci est réalisé grâce à des mécani

environnements ne peuvent pas contrôler.

- Le TEE offre une **protection contre certaines attaques physiques**. Les attaques intrusives qui brisent physiquement la structure physique du circuit intégré ne sont pas considérées.
- L'environnement d'exécution du système d'exploitation de confiance est démarré à partir d'une **racine de confiance** (RoT) à l'intérieur du TEE via un processus de démarrage sécurisé.
- Le TEE fournit un **stockage sécurisé** des données et des clés cryptographiques. Ce stockage est lié à ce TEE particulier sur un appareil particulier.
- Les logiciels extérieurs au TEE ne peuvent pas appeler directement les fonctionnalités exposées par les interfaces internes du TEE et doivent passer par des **protocoles de communication spécifiques** permettant au système d'exploitation de confiance de vérifier l'acceptabilité de l'opération non-TEE demandée.

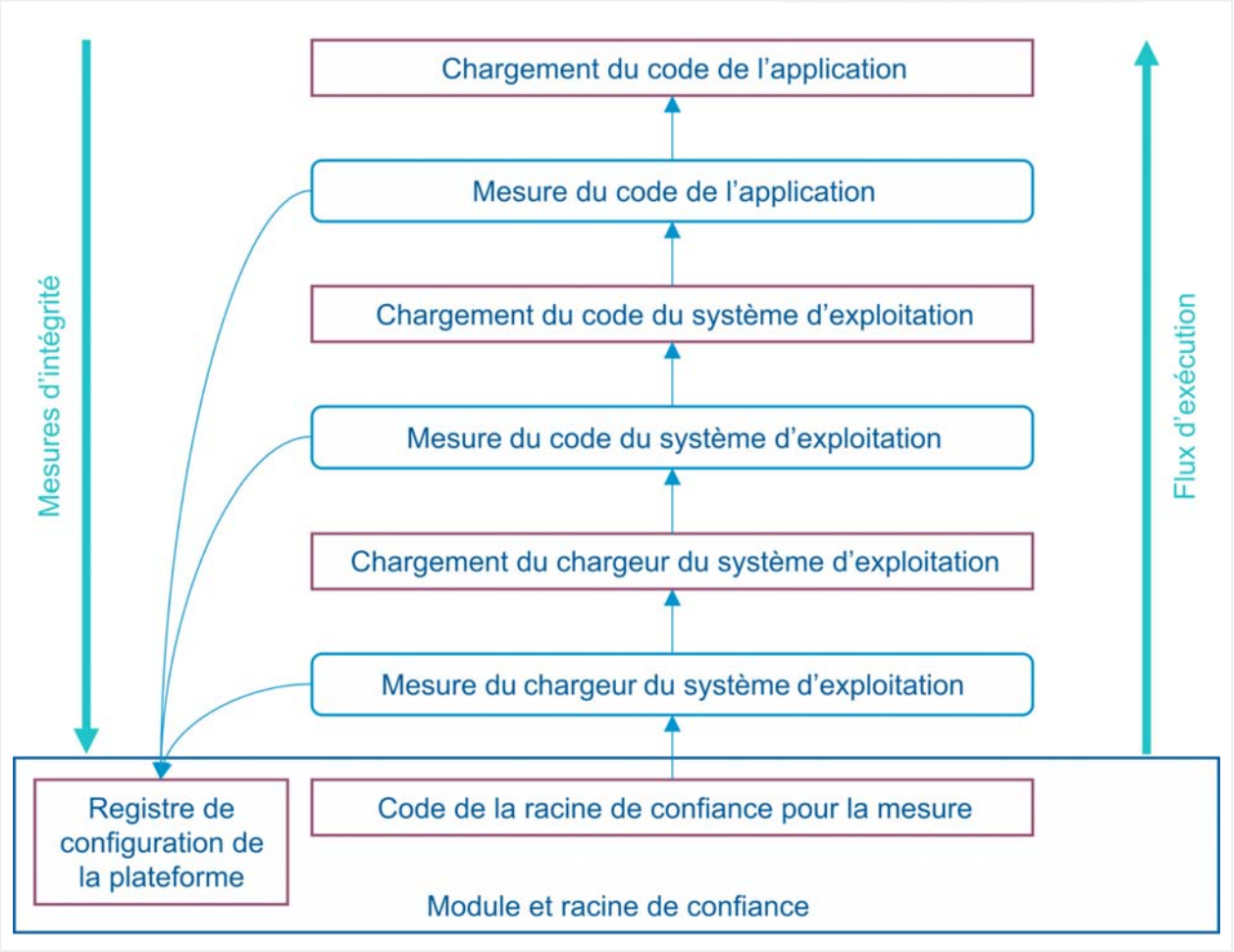
Vérification de l'intégrité

Un concept clé des TEE est la capacité de l'utilisateur à vérifier l'intégrité de la plate-forme sous-jacente. Cela est généralement réalisé grâce à deux composantes : le hachage des différents modules logiciels et la vérification du micrologiciel. Dans le cadre de l'informatique en nuage, il est nécessaire de pouvoir effectuer cette vérification à distance.

Hachage et signature des modules

Les hachages cryptographiques (également souvent appelés « mesures » dans le contexte des TEE) du logiciel, du micrologiciel et des fichiers de configuration associés, sous-tendent la création d'une chaîne de confiance. Chacune de ces chaînes de confiance commence par un module racine de confiance qui doit être aussi petit que possible. Il peut prendre la forme d'un bloc de démarrage implicitement fiable (par ex., stocké en mémoire à lecture seule) du système élémentaire d'entrée/sortie (BIOS) qui mesure d'abord sa propre intégrité, puis étend la mesure à l'ensemble du BIOS. Ensuite, chaque micrologiciel ou logiciel supplémentaire à exécuter est d'abord mesuré avant d'être exécuté et cela va jusqu'au système d'exploitation en suivant un processus de démarrage classique [20].

Ce mécanisme de *chaîne de confiance* résumé dans la Figure 2 fournit une vue fiable de l'état actuel du système. Toute bibliothèque, module, application, partie du système d'exploitation ou non, a une mesure d'intégrité stockée dans un stockage matériel sécurisé. En principe ces mesures sont signées afin d'empêcher les attaques par relecture ou par l'homme du milieu (par ex. [21]).



— Figure 2 – Séquence de démarrage d'une plateforme sécurisée (adapté de [22]).

Les protections matérielles garantissent qu'une fois qu'une enclave est lancée, aucune modification de son code ou de ses données statiques ne peut être effectuée depuis l'extérieur de l'enclave. Une fois lancés, les processus extérieurs à l'enclave peuvent recevoir une attestation cryptographique qui inclut une signature du code à l'intérieur de l'enclave, de sorte que le processus extérieur peut être assuré de l'exactitude de tout le code pouvant s'exécuter dans l'enclave.

Vérification du micrologiciel et de sa configuration

L'attestation locale ou à distance implique, comme indiqué ci-dessus, la génération d'une signature des mesures, et en particulier celle de la racine de confiance pour la mesur[3] (CRTM) avec des clés d'attestation spécifiques. Le mécanisme permet une utilisation à distance pour vérifier si le micrologiciel, le système d'exploitation et l'application souhaités sont réellement en cours d'exécution. Il permet également l'application de politiques de sécurité spécifiques.

Il convient de souligner que dans une architecture moderne complexe, le fabricant du CRTM peut ne pas rester longtemps l'autorité de confiance dans le processus de démarrage et il faut vérifier la chaîne de valeurs de hachage enregistrées pour déterminer l'autorité de l'état actuel du système. En outre, dans un centre de données typique, il peut y avoir des milliers de machines avec leur propre CRTM, ce qui rend leur suivi une tâche complexe, de sorte que des services d'attestation à distance dédiés sont souvent mis à la disposition de l'utilisateur de l'infrastructure d'informatique en nuage.

Notons enfin qu'une construction de racine de confiance avec des protections matérielles physiques sera plus difficile à changer qu'une construction uniquement dans le micrologicie

de racine

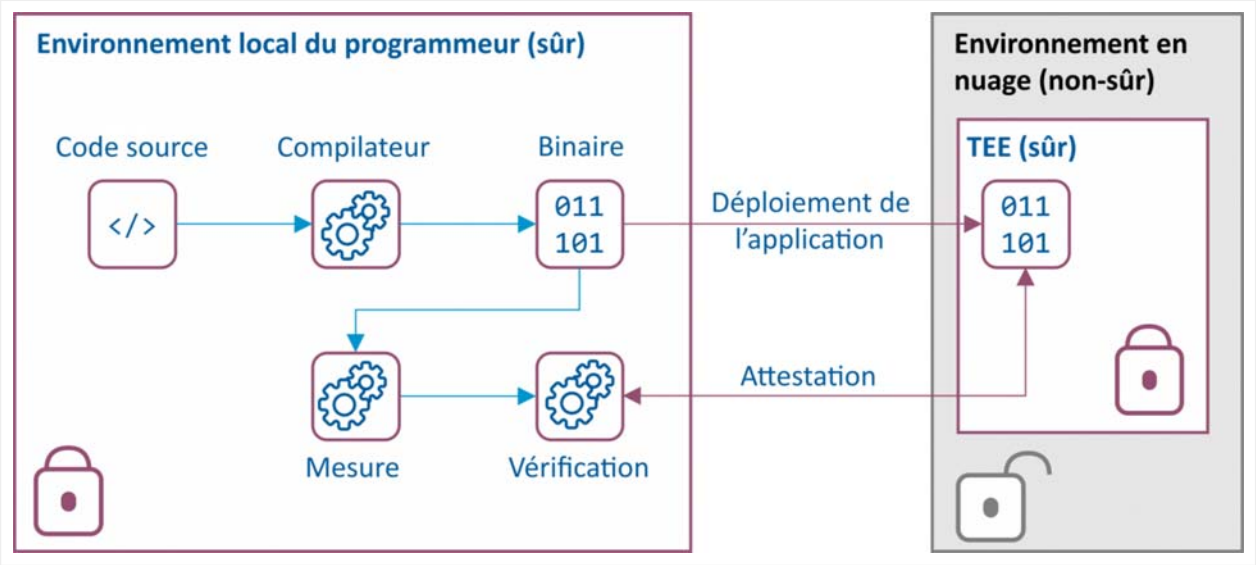
de confiance ne peut pas être mis à jour.

Attestation à distance

L'attestation est un élément fondamental pour vérifier la fiabilité d'un TEE et ensuite décider de lui faire confiance ou pas. Il permet à un environnement logiciel de prouver qu'un programme spécifique est exécuté par un matériel spécifique. Une attestation locale peut être utilisée entre deux environnements logiciels partageant le même matériel, mais ce qui intéresse la plupart des utilisateurs de systèmes avec TEE, est l'attestation à distance où les deux environnements logiciels fonctionnent sur du matériel différent.

En général cette attestation se base sur un module de plateforme fiable (« *trusted platform module* » – TPM), mais d'autres méthodes existent comme par exemple l'utilisation de fonctions physiques non clonables (« *physical unclonable function* » – PUF) ou encore des secrets fusionnés dans le circuit intégré.

La plupart des TEE peuvent générer des preuves cryptographiques vérifiables, qui peuvent être envoyées à une partie utilisatrice (par exemple, l'appareil utilisateur, le fournisseur de services), qui peut ensuite valider la signature de la preuve. Si la signature est valide, la partie de confiance conclut que le code attendu s'exécute dans un TEE authentique. La preuve est basée sur la mesure du TEE et peut également inclure une clé publique du TEE.



— Figure 3 – Diagramme générique pour le déploiement et l'attestation d'applications dans un TEE.

La Figure 3 présente un processus d'attestation générique. La partie utilisatrice du TEE doit connaître la mesure attendue, non seulement de l'application qu'elle souhaite déployer mais également du micrologiciel du TEE. La vérification peut s'effectuer grâce à un mécanisme de défi-réponse. De cette façon l'attestation permet de convaincre l'utilisateur que l'attestation a bien été produite par un logiciel spécifique à l'intérieur d'un matériel spécifique sans interférences extérieures. Plus de détails ainsi que des variations sont donnés dans [23].

Le système d'attestation doit permettre de pouvoir révoquer le TEE dans le cas où il s'avère que la sécurité de ce dernier est compromise.

Conclusion

La combinaison de composants matériels sécurisés et de techniques d'attestation à distance forme l'assise des environnements d'exécution de confiance qui ont pour but de protéger les données sensibles ou secrètes ainsi que le code exécuté contre des attaques de plus en plus fréquentes visant des données en cours de calcul.

Ces environnements sont *a priori* prometteurs dans le cadre de l'informatique en nuage. D'abord ils permettent d'augmenter le niveau de sécurité d'une infrastructure en nuage en isolant mieux les applications les unes des autres, empêchant par exemple une application compromise d'accéder aux données d'une autre. Ensuite, ils pourraient permettre d'accroître le niveau de confiance des utilisateurs de cette infrastructure pour le traitement de certaines données, pour autant que ces utilisateurs puissent garder le contrôle de la gestion des clés de chiffrement et déchiffrement utilisées au sein de l'environnement de confiance.

Pour l'utilisateur final, l'évaluation des risques liés à l'utilisation du système est simplifiée en ce sens que la confiance est placée dans un composant matériel spécifique et un micrologiciel vérifiable à distance, tous deux avec un comportement attendu, plutôt que dans l'ensemble de l'ordinateur. Mais d'autres risquent sont introduits, comme, par exemple, la possibilité d'un enferment dans une technologie TEE particulière.

Dans un prochain article nous regarderons plus en détails le fonctionnement de certaines mises en œuvre commerciales.

Références

[1] A. Greenberg, « A Peek Into the Toolkit of the Dangerous Triton Hackers », *Wired*, 10 avril 2019. Consulté le: 4 janvier 2023. [En ligne]. Disponible sur: https://www.wired.com/story/triton-hacker-toolkit-fireeye/

[2] K. Zetter, « Inside the Cunning, Unprecedented Hack of Ukraine's Power Grid », *Wired*, 3 mars 2016. Consulté le: 4 janvier 2023. [En ligne]. Disponible sur: https://www.wired.com/2016/03/inside-cunning-unprecedented-hack-ukraines-power-grid/

[3] R. Lakshmanan, « New Malware Families Found Targeting VMware ESXi Hypervisors », *The Hacker News*, 30 septembre 2022. Consulté le: 9 janvier 2023. [En ligne]. Disponible sur: https://thehackernews.com/2022/09/new-malware-families-found-targeting.html

[4] R. Naraine, « Stuxnet attackers used 4 Windows zero-day exploits », *ZDNET*, 14 septembre 2010. Consulté le: 9 janvier 2023. [En ligne]. Disponible sur: https://www.zdnet.com/article/stuxnet-attackers-used-4-windows-zero-day-exploits/

[5] C. Zhao, « SolarWinds, Probably Hacked by Russia, Serves White House, Pentagon, NASA », *Newsweek*, 14 décembre 2020. Consulté le: 9 janvier 2023. [En ligne]. Disponible sur: https://www.newsweek.com/solar-winds-probably-hacked-russia-serves-white-house-pentagon-nasa-1554447

[6] B. Barrett, « Russia's Elite Hackers Have a Clever New Trick That's Very Hard to Fix », *Wired*, 27 septembre 2018. Consulté le: 4 janvier 2023. [En ligne]. Disponible sur: https://www.wired.com/story/fancy-bear-hackers-uefi-rootkit/

[7] T. Bletsch, X. Jiang, V. W. Freeh, et Z. Liang, « Jump-oriented programming: a new class of code-reuse attack », in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, Hong Kong China, mars 2011, p. 30-40. doi:10.1145/1966913.1966919.

[8] T. Claburn, « MSI motherboards found to have insecure Secure Boot », *The Register*, 17 janvier 2023. Consulté le: 18 janvier 2023. [En ligne]. Disponible sur: https://www.theregister.com/2023/01/17/msi_motherboards_secure_boot/

[9] « TEE System Architecture v1.3 ». GlobalPlatform, mai 2022. [En ligne]. Disponible sur: https://globalplatform.org/specs-library/tee-system-architecture/

[10] M. Sabt, M. Achemlal, et A. Bouabdallah, « Trusted Execution Environment: What It is, and What It is Not », in *2015 IEEE Trustcom/BigDataSE/ISPA*, Helsinki, Finland, août 2015, p. 57-64. doi: 10.1109/Trustcom.2015.357.

[11] C. Gentry, « Fully homomorphic encryption using ideal lattices », in *Proceedings of the 41st annual ACM symposium on Theory of computing*, Bethesda MD USA, mai 2009, p. 169-178. doi: 10.1145/1536414.1536440.

[12] M. Naehrig, K. Lauter, et V. Vaikuntanathan, « Can homomorphic encryption be practical? », in *Proceedings of the 3rd ACM workshop on Cloud computing security workshop - CCSW '11*, Chicago, Illinois, USA, 2011, p. 113. doi: 10.1145/2046660.2046682.

[13] A. C. Yao, « Protocols for Secure Computations », présenté à 23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, nov. 1982. doi: 10.1109/SFCS.1982.38.

[14] N. Drucker et S. Gueron, « Combining Homomorphic Encryption with Trusted Execution Environment: A Demonstration with Paillier Encryption and SGX », in *Proceedings of the 2017 International Workshop on Managing Insider Security Threats*, Dallas Texas USA, oct. 2017, p. 85-88. doi: 10.1145/3139923.3139933.

[15] M. Pei, H. Tschofenig, D. Thaler, et D. Wheeler, « Trusted Execution Environment Provisioning (TEEP) Architecture », Internet Engineering Task Force, Internet Draft draft-ietf-teep-architecture-19, oct. 2022. Consulté le: 30 janvier 2023. [En ligne]. Disponible sur: https://datatracker.ietf.org/doc/draft-ietf-teep-architecture-19

[16] S. Checkoway et H. Shacham, « Iago Attacks: Why the System Call API is a Bad Untrusted RPC Interface ».

[17] « Common Criteria for Information Technology Security Evaluation ». novembre 2022. [En ligne]. Disponible sur: https://www.commoncriteriaportal.org/cc/

[18] « TEE Protection Profile - Version 1.3 ». GlobalPlatform, 4 juin 2020.

[19] « A Technical Analysis of Confidential Computing ». Confidential Computing Consortium, octobre 2021. [En ligne]. Disponible sur: https://confidentialcomputing.io/wp-content/uploads/sites/85/2022/11/CCC-A-Technical-Analysis-of-Confidential-Computing-v1.2_updated_2022-11-02.pdf

[20] D. A. Cooper, W. T. Polk, A. R. Regenscheid, et M. P. Souppaya, « BIOS protection guidelines », National Institute of Standards and Technology, Gaithersburg, MD, NIST SP 800-147, 2011. doi: 10.6028/NIST.SP.800-147.

[21] D. Challener et K. Goldman, « Trusted Platform Module (TPM) », *Trusted Computing Group (TCG)*, 2019. https://trustedcomputinggroup.org/work-groups/trusted-platform-

[22] C. Shepherd *et al.*, « Secure and Trusted Execution: Past, Present, and Future - A Critical Review in the Context of the Internet of Things and Cyber-Physical Systems », in *2016 IEEE Trustcom/BigDataSE/ISPA*, Tianjin, China, août 2016, p. 168-177. doi: 10.1109/TrustCom.2016.0060.

[23] H. Birkholz, D. Thaler, M. Richardson, N. Smith, et W. Pan, « Remote ATtestation procedureS (RATS) Architecture », Internet Engineering Task Force, Request for Comments RFC 9334, janv. 2023. doi: 10.17487/RFC9334.

Notes

[1] Plusieurs définitions des environnements d'exécution (TEE) de confiance ou des environnements d'exécution sécurisés (SEE) ont été données, parfois avec des contradictions entre elles. Dans cet article nous nous basons principalement sur les spécifications du *Confidential Computing Consortium* et de la *GlobalPlatform* [9]. Différentes définitions sont comparées et une description formelle de ces environnements est proposée dans [10].

[2] Il y a des propositions pour simplifier le travail des programmeurs d'applications sécurisées avec, par exemple, un protocole interopérable de gestion de telles applications fonctionnant sur différents TEE [15].

[3] Le « *Core Root of Trust for Measurement* » est le tout premier code du BIOS qui est exécuté dans le microprocesseur principal au moment de l'allumage. Il est souvent stocké dans un module de plateforme fiable.

This entry was posted in **Security** and tagged **environnement d'exécution de confiance**, **informatique confidentielle**, **TEE** by **Fabien Petitcolas**. Bookmark the **permalink [https://www.smalsresearch.be/introduction-a-l-informatique-confidentielle/]** .