Méthodes formelles pour l'exécution sécurisée sur plate-forme matérielle hostile

Florian Stosse, ESIEA Paris – Bureau Veritas

25 mai 2018

Les architectures matérielles modernes font aujourd'hui l'objet d'un certain nombre de vulnérabilités critiques, comme les récentes Spectre[1] et Meltdown[2] (ainsi que d'autres variantes comme SGXPectre[3] ou Spectre-NG). Ces failles sont la résultante de choix architecturaux faits par les fondeurs de microprocesseurs, tels que l'exécution d'instructions dans le désordre ou l'exécution spéculative (notamment utilisée pour faire de la prédiction de branchement).

Avant elles, l'introduction de firmwares ou de coprocesseur séparés (tels que l'Intel Management Engine[4]) s'exécutant avec des niveaux de privilèges élevés (ring -3 pour l'Intel ME) et offrant une surface d'attaque conséquente (présence d'une stack TCP/IP et d'un serveur web...), a également fait l'objet d'attaques[5] et de recherches visant à réduire leur emprise sur le processeur principal[6][7].

D'autres attaques matérielles, telles que le martèlement de mémoire ("rowhammer"), les attaques par canaux cachés ou auxiliaires, ou encore les risques de présence de chevaux de Troie matériels[8] nous incitent donc à considérer les plate-formes matérielles modernes comme hostiles à l'exécution sécurisée de programmes, et ce malgré l'ajout de fonctions de sécurité matérielles aux processeurs récents (enclaves sécurisées, fonctions de virtualisation pour la sécurité, etc.).

Avec un modèle de menace considérant le matériel comme hostile, la seule réponse possible est nécessairement attendue au niveau des contre-mesures logicielles, notamment via l'usage des méthodes formelles. Elles peuvent être introduites à différentes étapes du cycle de développement ou de compilation :

— Annotation du code par les développeurs
— Analyse statique du code avant compilation : détection des constructions faisant appel aux instructions ou fonctions matérielles hostiles
— Analyse statique du code intermédiaire : instrumentation du code (démarche similaire à PICON)
— Compilation : introduction de contremesures (retpoline, etc.)
— Analyse du binaire généré
— Analyse dynamique

Références

[1] Kocher, Paul, et al. "Spectre Attacks : Exploiting Speculative Execution." arXiv preprint arXiv :1801.01203 (2018).

[2] Lipp, Moritz, et al. "Meltdown. arXiv preprint." arXiv preprint arXiv :1801.01207 (2018).

[3] Chen, Guoxing, et al. "SGXPECTRE Attacks : Leaking Enclave Secrets via Speculative Execution." arXiv preprint arXiv :1802.09085 (2018).

[4] Joanna Rutkowska, "Intel x86 considered harmful" (2015)

[5] Ermolov, Mark, et al. "How to hack a tuned-off computer, or running unsigned code in Intel ME", BlackHat Europe (2017)

[6] Igor Skochinsky et Nicola Corna, "Intel ME : Myths and reality", 34C3 (2017)

[7] Joanna Rutkowska, "State considered harmful - A proposal for a stateless laptop" (2015)

[8] Adee, Sally. "The hunt for the kill switch." iEEE SpEctrum 45.5 (2008) : 34-39.

[9] Fontaine, Arnaud et al. "PICON : Control Flow Integrity on LLVM IR", SSTIC 2015