Extension d'une analyse de vulnérabilité de code binaire à des attaques par canaux auxiliaires

Yanis Sellami
CEA, List

Sébastien Bardin
CEA, List

yanis.sellami@cea.fr, sebastien.bardin@cea.fr

L'analyse automatique de programmes est une discipline qui vise à construire des analyses et des outils pour détecter des bugs dans le code des programmes. Les techniques développées permettent notamment de détecter les vulnérabilités mémoire (*e.g. buffer overflows*) ou numériques (*e.g.* division par zéro) qui peuvent corrompre les données ou stopper l'exécution. Pourtant, même lorsqu'on peut prouver qu'un programme est exempt de tels bugs, un attaquant peut encore exploiter certaines fonctionnalités de l'exécution du programme (temps d'exécution, cache, ...) pour comprendre son fonctionnement et extraire des informations secrètes. On appelle cela des attaques par canaux auxiliaires.

La plateforme d'analyse de code binaire BINSEC [1], développée au CEA, permet d'effectuer une analyse relationnelle [2] pour détecter certaines vulnérabilités à des attaques par canaux auxiliaires, notamment lorsque le code ne s'exécute pas en temps constant. Cette analyse peut alors servir de base au développement d'extensions qui permettent de détecter des vulnérabilités à des attaques plus complexes.

L'objectif de ce stage est d'étendre l'analyse relationnelle à de nouvelles vulnérabilités. En particulier, on pourra s'intéresser à l'analyse de vulnérabilité aux attaques par consommation d'énergie [3, 4], par *port contention*, ou par *ciphertext* [5]. Il sera attendu un état de l'art sur ces vulnérabilités aux attaques par canaux auxiliaires et une prise en main de la plateforme BINSEC. L'étudiant pourra ensuite proposer une extension de l'analyse relationnelle de BINSEC pour un nouveau type de vulnérabilité et éventuellement proposer une implémentation.

Éléments logistiques. Le stage sera hébergé sur le site Nano-INNOV à Saclay. Le stage sera co-encadré par Yanis Sellami et Sébastien Bardin

Postuler. Pour candidater ou pour obtenir plus d'informations, merci de prendre contact par mail avec les encadrants.

Références

[1] "Plateforme binsec." `https://binsec.github.io/`.

[2] L.-A. Daniel, S. Bardin, and T. Rezk, "Binsec/rel : Symbolic binary analyzer for security with applications to constant-time and secret-erasure," *ACM Trans. Priv. Secur.*, vol. 26, apr 2023.

[3] R. Tsoupidi, R. Lozano, E. Troubitsyna, and P. Papadimitratos, "Securing optimized code against power side channels," in *2023 IEEE 36th Computer Security Foundations Symposium (CSF)*, (Los Alamitos, CA, USA), pp. 340–355, IEEE Computer Society, jul 2023.

[4] Y. Wang, R. Paccagnella, E. T. He, H. Shacham, C. W. Fletcher, and D. Kohlbrenner, "Hertzbleed : Turning power Side-Channel attacks into remote timing attacks on x86," in *31st USENIX Security Symposium (USENIX Security 22)*, (Boston, MA), pp. 679–697, USENIX Association, Aug. 2022.

[5] S. Deng, M. Li, Y. Tang, S. Wang, S. Yan, and Y. Zhang, "CipherH : Automated detection of ciphertext side-channel vulnerabilities in cryptographic implementations," in *32nd USENIX Security Symposium (USENIX Security 23)*, (Anaheim, CA), pp. 6843–6860, USENIX Association, Aug. 2023.