Model Validation in BINSEC for Fault Injection Attacks on Binary Code

Yanis Sellami
CEA, List

Sébastien Bardin
CEA, List

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

Automated program analysis is a research topic that tries to design tools and analyses to detect bugs in programs. Existing techniques typically target memory access vulnerabilities (*e.g.* buffer overflows) and numerical vulnerabilities (*e.g.* zero division) that can corrupt program data or crash the exec ution. Yet, even when one can prove that a program contain no such bug, an attacker can still target the hardware on which the program is running to modify the actual execution. An attacker can then bypass security protections that were present in the original code.

However, automated detection procedures for fault injection vulnerabilities are expensive as each fault increases the size of the code to analyze. The binary code analysis platform BINSEC [1] developped at CEA possesses a state of the art analysis for fault attacks detection [2]. Still, while this technique permits detecting potential fault attacks on binary code, it currently does not have the ability verify the solutions it returns. Similarly, when we want to compare the results of different tools that find fault injection attacks, they may return different solutions that may all be valid, making the comparison difficult.

The goal of this project is to extend the capabilities of the fault injection analysis of BINSEC to validate a given fault injection attack. This will require the definition of an interface to describe fault models and an update of the fault injection analyzer to replay a given solution. The technique will then be evaluated with an experimental evaluation, and will have the possibility to be extended to additional fault models.

Logistics. The internship will be homed at Nano-INNOV in Saclay, co-supervised by Yanis Sellami and Sébastien Bardin.

Candidate. To candidate or obtain additional information, please contact the supervisors by email.

Références

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

[2] S. Ducousso, S. Bardin, and M.-L. Potet, "Adversarial reachability for program-level security analysis," in *Programming Languages and Systems* (T. Wies, ed.), (Cham), pp. 59–89, Springer Nature Switzerland, 2023.

[3] Y. Kim, R. Daly, J. Kim, C. Fallin, J. H. Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu, "Flipping bits in memory without accessing them : an experimental study of dram disturbance errors," in *Proceeding of the 41st Annual International Symposium on Computer Architecuture*, ISCA '14, p. 361–372, IEEE Press, 2014.