DISS. ETH NO. 30245

SOFTWARE-INSPIRED TECHNIQUES FOR DIGITAL HARDWARE SECURITY

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES
(Dr. sc. ETH Zurich)

presented by

FLAVIEN SOLT

Ingénieur diplômé de l'École polytechnique
MSc ETH ITET, ETH Zürich

born on May 16th, 1997

accepted on the recommendation of

Prof. Dr. Kaveh Razavi (Advisor)
Prof. Dr. Mathias Payer
Prof. Dr. Marco Guarnieri
Prof. Dr. Luca Benini

2024

ABSTRACT

We entered an era where new hardware flourishes at an unprecedented pace and with unseen diversity. We are also living in an era where security and safety are paramount, and where the potential impact of a single bug can be catastrophic. Hence, we urgently need foundations to detect as many hardware bugs as possible before their deployment.

Hardware validation is universally recognized as complex, expensive and tedious. Despite genuine best efforts, the last decade has shown that the industry is incapable of producing non-trivial bug-free hardware. What will then happen with the rise of open-source hardware? Without effective and easy-to-adopt solutions for validation, it is hard to believe that the open-source hardware community will be able to produce safe and secure hardware, despite its best intentions.

Interestingly, the exact same situation occurred in the software world some decades ago. Software was plagued with myriads of bugs and security issues, after what the software community developed a formidable set of tools and methodologies to detect bugs and security issues. **Could we adapt some of these tools and methodologies to hardware?**

To answer this question, our plan is to first observe many CPU errata, deduce the most promising techniques from software security, and adapt them. To understand contemporary CPU bugs, we build the *RemembERR* database based on thousands of errata. We deduce two techniques inspired by software security that are particularly promising for hardware: dynamic information flow tracking and fuzzing. We introduce *CellIFT*, the first scalable hardware dynamic information flow tracking mechanism and showcase 4 new architectural or microarchitectural security applications. We then introduce *Cascade*, a black-box CPU fuzzer that found dozens of new bugs and outperforms other fuzzers' coverage. We finally demonstrate *MiRTL*, a new class of hardware attacks that relies on EDA software bugs, and propose *TransFuzz*, a fuzzer that produces complex hardware descriptions to find such bugs in popular open-source EDA software.

All these contributions demonstrate that when properly adapted, software security techniques can provide effective and easy-to-adopt solutions that will empower safer and more secure hardware.

RÉSUMÉ

Nous entrons dans une ère où le matériel (hardware) prospère à un rythme sans précédent et avec une diversité inédite. À une époque où la sécurité est cruciale, la détection précoce des bugs matériels devient impérative.

La validation matérielle est reconnue comme complexe, coûteuse et fastidieuse. Malgré des efforts sincères, la dernière décennie a révélé l'incapacité de l'industrie à produire un matériel sans bugs significatifs. Face à l'émergence du matériel open source, sans solutions de validation efficaces et faciles à adopter, il paraît invraisemblable que la communauté puisse garantir un matériel sûr et sécurisé malgré ses bonnes intentions.

Une situation similaire s'est produite dans le domaine du logiciel il y a quelques décennies. L'univers logiciel était affecté par de nombreux bugs et problèmes de sécurité, mais la communauté a développé des outils efficaces pour les détecter. **Pourrait-on adapter ces outils et méthodes au matériel ?**

Pour répondre à cette question, notre plan est d'observer de nombreuses erreurs CPU, déduire les techniques les plus prometteuses provenant de la sécurité logicielle pour le matériel, et les adapter. Pour comprendre les bugs CPU actuels, nous introduisons la base de données *RemembERR* basée sur des milliers d'erratas. Nous en déduisons deux techniques particulièrement prometteuses pour le matériel, issues du logiciel: le suivi dynamique de flux d'informations et le fuzzing. Nous présentons *CellIFT*, le premier mécanisme de suivi dynamique de flux d'informations matériel scalable, avec 4 nouvelles applications de sécurité architecturale ou microarchitecturale. Ensuite, nous présentons *Cascade*, un fuzzer de CPU boîte noire qui a permis de découvrir de nombreux bugs et excède la couverture des autres fuzzers. Enfin, nous présentons *MiRTL*, une nouvelle classe d'attaques matérielles basées sur des bugs logiciels EDA permettant de corrompre du hardware apparemment sain, ainsi que *TransFuzz*, un fuzzer qui identifie ces bugs dans des logiciels EDA open source populaires.

Toutes ces contributions démontrent que, lorsqu'elles sont correctement adaptées, les techniques de sécurité logicielle peuvent fournir des solutions efficaces et faciles à adopter qui renforceront la sécurité du matériel.