Tackling Hardware Trojans via Hardware-based Methodologies

Alessandro Palumbo*

CentraleSupélec, Inria, Univ Rennes, CNRS, IRISA

Abstract

Hardware Trojans (HTs) represent significant challenges to the security and reliability of modern microprocessor-based systems. This manuscript introduces two complementary approaches to enhance hardware security. First, programmable Hardware Security Modules are proposed to detect Hardware Trojan Horses by monitoring instruction-fetch activities, identifying malicious interferences, and preventing software-exploitable Hardware Trojan Horse activations. Second, a methodology based on side-channel analysis is proposed to verify the integrity of FPGA bitstreams, allowing the identification of tampered configurations through the extraction and classification of both high- and low-level features.

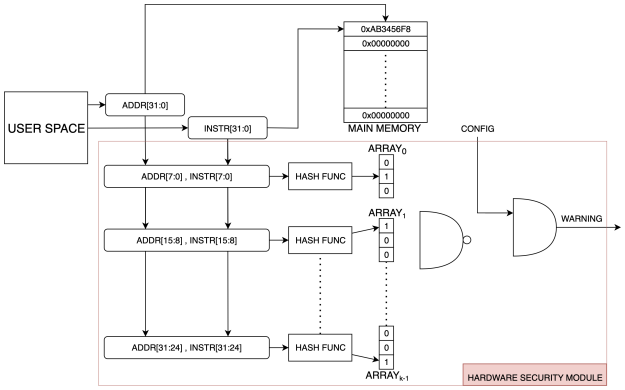

Figure 1: *Configuration phase of the HSC [1]*

Introduction & Motivation

Software-exploitable Hardware Trojan Horses (HTHs) represent a severe threat to modern microprocessor-based systems. These malicious modifications or additions to circuit elements allow attackers to execute their own software, modify running software, or acquire root privileges. A real-world example is the *Rosenbridge* backdoor, discovered in a commercial Via Technologies C3 processor. This backdoor could be activated by a specific sequence of instructions, enabling the attacker to gain supervisor mode privileges [2]. This manuscript addresses the growing need for robust hardware-based security mechanisms to protect microprocessors and FPGA-based systems against emerging threats such as HTHs. Integrating Hardware Security Checkers (HSCs) and side-channel analysis techniques is proposed to tackle these challenges.

In addition to runtime protections, side-channel analysis provides a powerful tool for verifying the integrity of FPGA bitstreams. By analyzing high- and low-level features, such as timing information, power consumption, and hardware resource utilization, side-channel analysis enables the detection of tampered configurations that could introduce malicious behavior into FPGA designs.

The Hardware Security Checker against Hardware Trojan Horses

The HSC is a component integrated between the Core and the Main Memory [1]. In parallel with the safe program's installation into the Microprocessor Main Memory, the HSC works in *configure* mode (Figure 1), taking information about legitimate instructions and their corresponding addresses (from the User-space side). During the program(s) execution, at runtime, the HSC switches into *query* mode (Figure 2) to monitor the microprocessor's fetching activity and check whether fetched addresses and instructions match the previously configured data. The HSC stores legitimate instruction-address pairs during the configuration phase by hashing these tuples. This data is then fragmented into smaller components and stored in a series of bit arrays. The query phase compares the fetched instruction-address tuples against the configured data, raising an alert if an anomaly is detected. To further enhance detection capabilities, the HSC can incorporate the Hamming computation module into its architecture (as reported in [3]). The checking bits are computed during the configuration phase for each legitimate instruction. The fetched instruction's Hamming code is compared to the pre-computed values stored in a dedicated Hamming memory at runtime.

A strategy to mitigate the activation of software-exploitable HTHs and to protect sensitive data involves implementing a built-in code obfuscation methodology within the microprocessor's execution pipeline [4]. A dedicated hardware module manages the obfuscation

*Corresponding author: alessandro.palumbo@inria.fr

technique, referred to as the Hardware Code Obfuscator (HCO), placed between the decode and execute stages of the pipeline. This module dynamically obfuscates the executed software at runtime, altering the sequence of operations performed by the microprocessor without changing the final outcome of the program. By doing so, the HCO minimizes the exposure of sensitive information to HTHs. It also sabotages their activation mechanisms (the *Rosenbridge* backdoor could be activated by a specific sequence of instructions [2]. The obfuscation strategies include: i) **Garbage Code Insertion:** Random instructions inserted to mask patterns and inject noise; ii) textbf-Variable Encryption/Decryption: Data are encrypted in registers and decrypted when needed; iii) **Register Scrambling:** The data are dynamically moved between registers.

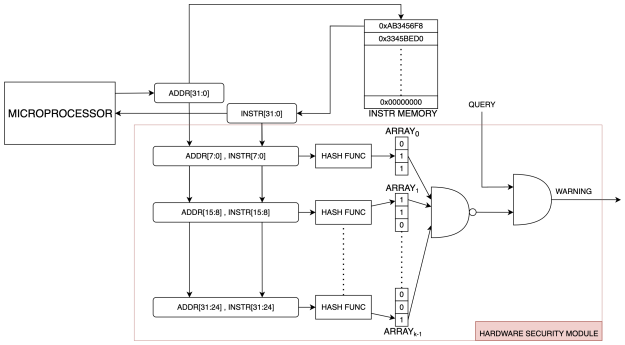

Figure 2: *Query phase of the HSC [1]*

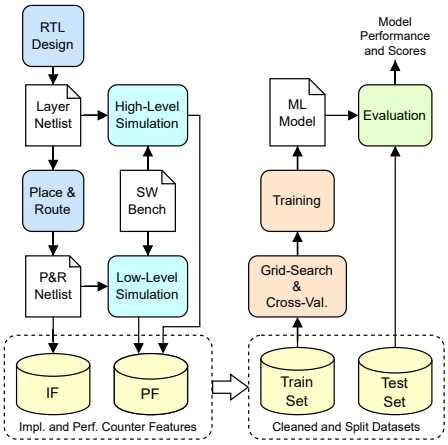

Figure 3: *The workflow for detecting HTHs in FPGA [5]*

Side-Channel Analysis for FPGA Tampering

The works presented in [6, 5] propose an innovative ML-based framework for detecting and classifying HTHs in RISC-V cores implemented on FPGA platforms. A central aspect of these works is the comprehensive feature extraction process, which combines performance features, such as runtime characteristics collected through hardware performance counters, with implementation features derived from FPGA synthesis and implementation reports. Performance features include metrics like the number of executed instructions, waiting cycles, and memory access patterns. In contrast, implementation features focus on hardware utilization (e.g., LUT and FF usage), power consumption, and timing parameters, such as the worst negative slack. The workflow is reported in Figure 3.

Table 1: *Best rates and resource usage of the proposals*

Ref.	Acc.	FP	FN	#LUT	#FF	#BRAM	#LUTRAM
[1]	100%	0%	0%	75	31	8	0
[3]	100%	0%	0%	82	31	8.5	0
[4]	NA	NA	NA	2,640	1,498	0	24
[6]	100%	0%	0%	NA	NA	NA	NA
[5]	100%	0%	0%	NA	NA	NA	NA

Conclusion

This paper presented two complementary methodologies for detecting and mitigating Hardware Trojan Horses in microprocessor-based systems. In the first approach, hardware security modules provide a programmable solution to monitor and verify executed instructions or obfuscate elaborated data. The second methodology leverages side-channel analysis with Machine Learning techniques, enabling the detection of compromised FPGA bitstreams by extracting and classifying both behavioral and structural features. Table 1 reports the best rates and resource usage details of the proposed solutions.

References

[1] A. Palumbo et al. "A lightweight security checking module to protect microprocessors against hardware trojan horses". In: *2021 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*. IEEE. 2021, pp. 1–6.

[2] D. Christopher. *Hardware Backdoors in x86 CPUs*. https://i.blackhat.com/us-18/Thu-August-9/us-18-Domas-God-Mode-Unlocked-Hardware-Backdoors-In-x86-CPUs-wp.pdf. 2018.

[3] A. Palumbo et al. "Improving the Detection of Hardware Trojan Horses in Microprocessors via Hamming Codes". In: *2023 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*. IEEE. 2023, pp. 1–6.

[4] A. Palumbo et al. "Built-in Software Obfuscation for Protecting Microprocessors against Hardware Trojan Horses". In: *2023 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*. IEEE. 2023, pp. 1–6.

[5] S. Ribes et al. "Machine Learning-Based Classification of Hardware Trojans in FPGAs Implementing RISC-V Cores." In: *ICISSP*. 2024, pp. 717–724.

[6] A. Palumbo et al. "Is your FPGA bitstream Hardware Trojan-free? Machine learning can provide an answer". In: *Journal of Systems Architecture* 128 (2022), p. 102543.