Is your Microprocessor attacked by the design tool? Machine Learning Can Provide the Answer

Alessandro Palumbo
alessandro.palumbo@centralesupelec.fr

Context

Software-exploitable Hardware Trojan Horses can be inserted into Microprocessors allowing the attackers to run their own software or to gain unauthorized privileges. It has been demonstrated that by observing some features of the Microprocessor (apparently not related to its program run), it is possible to gain information about Microprocessor running operations.

Hardware Trojan Horses (HTHs) consist of malicious, undesired circuit modifications. They have always been considered more an academic issue because of the difficulty of insertion in real-world systems, leading to reduced advantages for the attacker. Recently, it has been demonstrated that complex *software-exploitable HTHs* can be inserted in real-world commercial microprocessors. Such HTHs allow the attacker to execute his/her malicious software, modify the running software, or acquire root privileges [1]–[3]. In 2018, the *Rosenbridge* backdoor has been found in a commercial Via Technologies C3 processor [4, 5]. A specific sequence of instructions allowed the attacker to activate the Rosenbridge backdoor and enter supervisor mode[1].

Recently, a new security-related menace was raised: HTHs introduced in the designed circuit by the employed CAD tool [6, 7]. In [8, 9] the don't care of the design are exploited to insert HTHs both in the RTL code or gate-level netlist. In [10] a black-hat high-level synthesis tool has been presented: starting from a high-level specification, i.e., a C/C++/SystemC, of the desired functionality the tool produces an HTH-infested hardware implementation of the corresponding IP core. The authors also demonstrated that several types of HTHs could be introduced in the produced IP core: HTHs downgrading performance, changing the implemented functionality and draining the system's battery. Finally, in [11] the authors demonstrate that all electronic CAD tools, i.e., high-level synthesis, logic synthesis, physical design, verification, test, and post-silicon validation, are potential threat vectors to different degrees. Similar considerations can also be made when looking at the FPGA scenario

[1] Via Technologies officially commented that this behavior was due to an undocumented feature meant for debugging

instead of the ASIC one. It has indeed been demonstrated that CAD tools may seriously threaten the security and trust of FPGA-based systems [12, 13, 14]. In particular, it has been demonstrated that malicious CAD tools may tamper the produced bitstream before FPGA configuration to introduce HTHs in the system [15, 16]. Given this discussion, it is crucial to provide designers with effective tools to detect malicious modifications introduced in the system by the employed CAD tool before sending the design to the foundry (in the case of an ASIC design) or before integrating it in the final system (in the case of an FPGA-based design)

In [17] the authors demonstrate that has been possible to detect HTHs implemented in RISC-V ISA softcores. By looking at some features (i.e., power consumption and temperature traces, execution times, performance counter values, etc.) of the FPGA running, they detect some kind of HTHs via Machine Learning (ML) techniques. What about different HTH types? Are the features to detect HTHs always the same? Which are the best ML models to such detection?

Internship

The challenge of this internship is to emulate the insertion of different type of HTHs on FPGA in order to evaluate if it is possible to detect attacked bitstream via ML computasions. Many HTH models are reported in TrustHub repository [15].

Here are the main steps of the internship:

1. Get familiar with the Vivado tool for HDL design and FPGA bitstream implementation phases;
2. Get familiar with the RISC-V processor and its toolchain. Ibex repository [16] provides the code of the core and its toolchain;
3. Get familiar with the ML computations.

Required skills or interests

- Hardware Design Languages (e.g. VHDL, Verilog, Systemverilog)
- Software & Application Design Languages (e.g. C, C++, Python, Matlab)

Institute

The internship will take place at CentraleSupélec in Rennes, France, in the SUSHI Inria team[2].This team is part of the IRISA laboratory[3]

Practical aspects

[2] https://team.inria.fr/sushi/
[3] https://www.irisa.fr/en

References

[1] Y. Jin, M. Maniatakos, and Y. Makris, “Exposing vulnerabilities of untrusted computing platforms,” in *Proc. Int. Conf. Computer Design*, pp. 131–134, 2012.

[2] N. G. Tsoutsos and M. Maniatakos, “Fabrication attacks: Zero-overhead malicious modifications enabling modern microprocessor privilege es- calation,” *IEEE Trans. Emerging Topics in Computing*, vol. 2, no. 1, pp. 81–93, 2014.

[3] X. Wang, T. Mal-Sarkar, A. Krishna, S. Narasimhan, and S. Bhunia, “Software exploitable hardware trojans in embedded processor,” in *2012 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, pp. 55–58, IEEE, 2012.

[4] C.Domas,“Hardwarebackdoorsinx86cpus.”https://i.blackhat.com/us- 18/Thu-August-9/us-18-Domas-God-Mode-Unlocked-Hardware- Backdoors-In-x86-CPUs-wp.pdf, 2018.

[5] project:rosenbridge, last access Feb. 2022. URL: https://github. com/xoreaxeaxeax/rosenbridge.

[6] J. A. Roy, F. Koushanfar, I. L. Markov, Extended abstract: Circuit cad tools as a security threat, in: 2008 IEEE International Workshop on Hardware-Oriented Security and Trust, 2008.

[7] M. Potkonjak, Synthesis of trustable ics using untrusted cad tools, in: Proceedings of the 47th Design Automation Conference, 2010, pp. 633– 634.

[8] N. Fern, S. Kulkarni, K.-T. T. Cheng, Hardware trojans hidden in rtl don’t cares — automated insertion and prevention methodologies, in: 2015 IEEE International Test Conference (ITC), 2015.

[9] W. Hu, L. Zhang, A. Ardeshiricham, J. Blackstone, B. Hou, Y. Tai, R. Kastner, Why you should care about don’t cares: Exploiting internal don’t care conditions for hardware trojans, in: 2017 IEEE/ACM Interna- tional Conference on Computer-Aided Design (ICCAD), 2017.

[10] C. Pilato, K. Basu, F. Regazzoni, R. Karri, Black-hat high-level synthesis: Myth or reality?, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 27 (2018) 913–926.

[11] K. Basu, S. M. Saeed, C. Pilato, M. Ashraf, M. T. Nabeel, K. Chakrabarty, R. Karri, Cad-base: An attack vector into the electronics supply chain, ACM Transactions on Design Automation of Electronic Systems (TO- DAES) 24 (2019) 1–30.

[12] S. Sunkavilli, Z. Zhang, Q. Yu, New security threats on fpgas: From fpga design tools perspective, in: 2021 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), 2021, pp. 278–283.

[13] J. Zhang, G. Qu, Recent attacks and defenses on fpga-based systems 12 (2019).

[14] A. Duncan, F. Rahman, A. Lukefahr, F. Farahmandi, M. Tehranipoor, Fpga bitstream security: A day in the life, in: 2019 IEEE International Test Conference (ITC), 2019, pp. 1–10.

[15] B. Shakya, T. He, H. Salmani, D. Forte, S. Bhunia, M. Tehranipoor, Benchmarking of hardware trojans and maliciously affected circuits, Journal of Hardware and Systems Security 1 (2017) 85–102.

[16] IbexRISC-VCore,lastaccessFeb.2022.URL:https://github.com/lowRISC/ibex/.

[17] A. Palumbo, L. Cassano, B. Luzzi, J. A. Hernández, P. Reviriego, G. Bianchi, and M. Ottavi. “Is your FPGA bitstream HardwareTrojan-free? Machine learning can provide an answer”. In: Journal of Systems Architecture (2022), p. 102543.