Side-channel attack hardware detection module added to RISC-V core

Juliette Pottier [1]*, Bertrand Le Gal[2], Maria Méndez Real [3] and Sébastien Pillement [1]

[1]Nantes Université, CNRS, IETR-UMR 6164, F-44000 Nantes, France
[2]Université de Rennes, IRISA/INRIA lab., CNRS UMR 6074, 35042 Rennes, France
[3]Université de Bretagne Sud, Lab-STICC, CNRS UMR 6285, 56100 Lorient, France

Abstract

Processor performance optimizations such as Out-of-Order or speculative execution are known to be exploited by attackers for malicious purposes. Numerous side-channel attacks have been developed over decades and more recently transient attacks are considered as serious threats. As a countermeasure, previous works offered detection methods monitoring hardware performance counters (HPC). In this work we propose to take advantage of dynamic instructions insertion to monitor HPCs and detect side-channel attacks. This approach offers flexibility as it allows to dynamically adapt the events monitored by HPCs to the state of the system. We present our light-weight hardware micro-decoding unit used to insert monitoring instructions in the execution flow of a RISC-V core and detect side-channel attacks.

Cache-based side channel attacks (CSCAs)

CSCAs exploit microarchitectural features of cache memories in order to leak sensitive information. The ability to observe cache contention by measuring memory access time undermines the secure execution of encryption algorithms, making it possible for sensitive data, such as encryption keys, to be stolen by such attacks. Over the last decades, several CSCAs have been developed including Prime+Probe [1], and Flush+Reload [2]. These attacks can also be used as an intentional means of communication for a malicious application and are then referred to as covert channels. This is the case for the so-called transient attacks, i.e. attacks that leverage instructions that are not yet committed. Spectre [3] and Meltdown [4] are examples of most notable of these attacks. Both harness branch prediction and out-of-order execution to ensure secret information encoding in cache memory. Then, with the help of a covert channel, the malicious process can retrieve sensitive data by cache accesses time measurements.

Attack detection methods using HPCs

Hardware performance counters are embedded in modern CPUs since their introduction in processor families such as Intel Pentium or AMD Athlon [5]. Their initial purpose was to record hardware events activity to profile processor's execution in terms of performance to help software and hardware designers in their tasks. Monitored hardware events are of different kinds such as program characterization, memory accesses, pipeline stalls, branch prediction, or resource utilization [6]. Because HPCs allow to watch over microarchitectural events such as cache accesses or misses at a particular cache level, a whole section of microarchitectural attacks detection methods rely on the use of HPCs.

Micro-decoding Unit

The originality of the presented detection module lies in the use of a micro-decoding unit [7] to detect side-channel attacks at runtime. We added a micro-decoding unit in the CVA6[1] core pipeline in order to dynamically insert monitoring instructions in the execution flow. In order to avoid impacting the critical path, the decision was made to circumscribe modifications of the architecture at the interface between Instruction Decode stage (ID) and Issue stage, by adding a dedicated pipeline stage. The micro-decoding unit sees the flow of decoded instructions and enables the insertion of micro-instructions each time a decoded instruction is recognized as a macro-instruction. The micro-decoding unit is composed of (i) a FWFT-type FIFO (First-Word-Fall-Through) to handle synchronization with the Issue stage, (ii) a FSM to trigger a micro-decoding state each time a macro-instruction has been decoded, and (iii) a ROM containing the sequences of inserted micro-instructions.

*Corresponding author: juliette.pottier@univ-nantes.fr

[1] https://github.com/openhwgroup/cva6

Dynamic CSCAs detection

To take advantage of the insertion of monitoring instructions to detect side-channel attacks, we developed two distinct sequences of micro-instructions: one dedicated to the configuration and reset of HPCs, and the other to read HPCs and compare values to adapted thresholds. This approach allows a high degree of flexibility with dynamic selection of micro-architectural events to monitor, dynamic fitting of thresholds, and dynamic evaluation of the context.

In order to detect Prime+Probe and Flush+Reload attacks two different detection criteria C_1 and C_2 have been established. For each, acceptable value thresholds, T_1 and T_2 respectively are defined based on benchmarks HPCs profiling.

- *Criteria 1 C_1: Number of data-cache misses/Number of data-cache accesses $>T_1$*
- *Criteria 2 C_2: Number of data-cache misses/Number of load accesses $>T_2$*
- *Alarm trigger A: $A = C_1 \vee C_2$*

Detection accuracy

We evaluated our detection module effectiveness for the detection of Prime+Probe and Flush+Reload attacks under different system load conditions, and we compared HPCs values with about twenty different benchmarks. The enhanced core has been implemented on a Genesys II FPGA board and tests were done with a Linux OS. To evaluate the reliability of detection, attacks were performed under different load conditions: 1, 2 and 4 other applications running simultaneously. We obtained 2.73% false negative in the worst case and 0.40% false positive for 2000 executions of each benchmark, as Table 1 shows.

Table 1: *Detection rate evaluation.*

Load conditions	FP	FN	Accuracy
No noise	0.09%	0%	**100%**
1 noisy app	0.18%	0%	**100%**
2 noisy apps	1.52%	0.01%	**99.99%**
4 noisy apps	2.73%	0.40%	**99.60%**

Hardware costs evaluation

The hardware costs induced by the addition of our detection module to the CVA6 pipeline has also been evaluated. Table 2 presents the results obtained comparing the reports synthesized from the CVA6 core without the detection module as a baseline and the core enhanced with the addition of the detection module. With 2 sequences respectively of 13 and 12 micro-instructions contained in the ROM of our detection module, we observe a slight increase of the hardware complexity regarding the number of LUTs +4.3% and the number of flip-flops +3.5%. In fact, this overhead comes mainly from the introduction of the FSM, the addition of 5 registers in the register file to contain temporary data used by the detection module and the added FIFO with a width of 364 bit and a depth of 2 at the interface of the detection module and the issue stage. The increase in the number of BRAM cells corresponds to the memory cost of our micro-decoding unit. The use of BRAM technology guarantees the feasibility of a microcode upgrade, which will be studied in future work.

Table 2: *Hardware complexity comparison.*

	LUT	SRL	FF	BRAM36
Baseline	47453	0	24764	36
Enhanced core	49518	0	25636	38
	+4.3%	-	**+3.5%**	**+5.5%**

Conclusion

In this work, we demonstrate the possibility to take advantage of a micro-decoding unit to detect side-channel attacks by inserting dynamically monitoring instructions in the instruction flow. Additional efforts will be devoted to develop a control unit in charge of the interface between the OS and the hardware detection module and to deploy dynamic protection of the cache memory.

References

[1] D. A. Osvik, A. Shamir, and E. Tromer. "Cache Attacks and Countermeasures: The Case of AES". In: *Topics in Cryptology – CT-RSA 2006*. 2006.

[2] Y. Yarom and K. Falkner. "FLUSH+RELOAD: a High Resolution, Low Noise, L3 Cache Side-Channel Attack". In: *USENIX security symposium* (2014).

[3] P. Kocher and al. "Spectre Attacks: Exploiting Speculative Execution". In: *IEEE Symposium on Security and Privacy*. 2019.

[4] M. Lipp et al. "Meltdown: reading kernel memory from user space". In: *Communications of the ACM* (2020).

[5] S. Carnà and al. "Fight Hardware with Hardware: Systemwide Detection and Mitigation of Side-channel Attacks Using Performance Counters". In: *Digital Threats: Research and Practice* (2023).

[6] B. Sprunt. "The basics of performance-monitoring hardware". In: *IEEE Micro* (2002).

[7] J. Pottier et al. "RISC-V processor enhanced with a dynamic micro-decoder unit". In: *ICECS*. 2024.