Decoding Attack Behaviors by Analyzing Patterns in Instruction-Based Attacks using Gem5

Muhammad Awais*, Maria Mushtaq*,
Lirida Naviner*, Florent Bruguier†,
Jawad Haj Yahya‡

* *IMT, Telecom-Paris, 19 Pl. Marguerite Perey, 91120 Palaiseau, France*

† *LIRMM, CNRS - University of Montpellier, 34090 Montpellier, France*

‡ *Rivos Inc, 3315 Scott Boulevard, Santa Clara, USA*

ABSTRACT

The diversity of Instruction Set Architectures (ISAs), each with unique limitations and optimization strategies, presents both opportunities and challenges in processor design. Modern processor vendors leverage these ISAs to enhance security, reliability, and performance. Recent security vulnerabilities, notably Spectre and Meltdown, have underscored the importance of robust hardware security measures. The recent discovery of attacks such as Specter and Meltdown had a high impact on the vendors regarding hardware security. Processor micro-architectures are susceptible to side-channel attacks, which exploit information leakage to identify vulnerabilities. Techniques such as speculative execution and branch prediction, commonly employed by processors from AMD, Intel, and ARM, while beneficial for performance optimization, inadvertently create avenues for such attacks. Additionally, the practice of out-of-order execution, designed to maximize efficiency, can be manipulated to form side channels, further compromising security. Additionally, shared memory resources, particularly cache memory, are another vector for attack. By analyzing access patterns to shared caches, attackers can construct cache-based side channels, facilitating sophisticated attacks like FLUSH+Reload and Prime+Probe.

In response to these threats, this work proposes a comprehensive mechanism for securing processor micro-architectures against side-channel attacks. Our methodology comprises five stages: (1) identifying and developing attack vectors, (2) compiling these attacks across various architectures, (3) scripting simulations using the Gem5 tool, (4) running these simulations, and (5) analyzing the resultant attack traces to understand and mitigate vulnerabilities. These stages are detailed in the next paragraphs.

KEYWORDS: ACACES; Security and privacy; Hardware attacks and countermeasures; Hardware Security implementations and Micro-architecture security

[1]E-mail: {muhammmad.awais,maria.mushtaq,lirida.naviner} @telecom-paris.fr
[2]E-mail: {florent.bruguier} @lirmm.fr

1 Introduction

To analyze the behavior of these attacks, we run these attacks using the state-of-the-art tool named Gem5 [LPAA+20]. It is a cycle-accurate simulator, which means that it simulates each cycle of the hardware. There are two types of simulation modes that are offered by Gem5: System Emulation (SE) mode, and Full-System (FS) mode. FS mode simulates the entire hardware and software stack like the Operating System (OS). On the other hand, the System-Emulation (SE) mode uses only the hardware components such as processors, cache memory, etc., it operates at the application level, not requiring any OS to boot up.

Our methodology involves five stages. Fig.1 shows all the stages that involve decoding the behaviors of the attacks. Stage ① we analyze the attacks, in stage ② we compile them for different architectures by using the cross-compilers, in stage ③ we make the simulation scripts and in stage ④ involves the simulation. At last, we analyze the traces of the attacks in stage ⑤.

Gem5 is the state-of-the-art tool for architectural research [LPAA+20]. In the first stage ①, we reproduce the attack vector, which includes the Spectre attack for x86 and ARM architectures in SE mode and FS Mode. Additionally, we reproduce attacks involving cache-based timing attacks such as Flush+Reload [YF14]. For this, we reproduced it for the x86 ISA [Dom17], and for ARM, we use Aarch64 for cross-compiling [GS05].

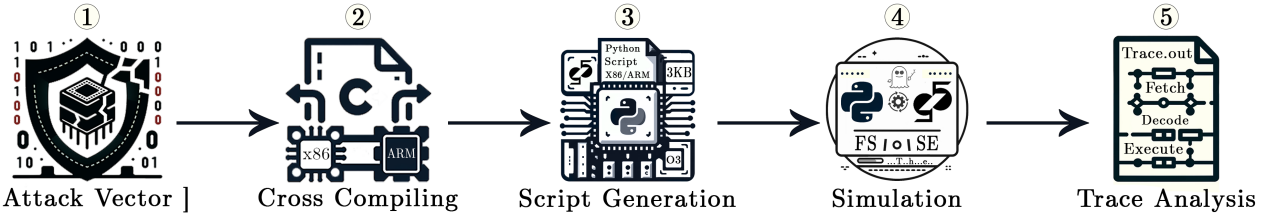

Figure 1: Methodology: Reproducing attack library, cross-compile for Aarch64 ARM, generate Python scripts, run Gem5 simulation, and analyze traces.

In stage ②, we use the (`aarch64-linux-gnu-gcc`) cross-compilers for the attack vector to make a binary executable file. After the compiling stage, we make the configuration scripts in stage ③, this script in Python is for Gem5 to set the interconnects between the cache and appropriate processors, use out-of-order (O3) processors [LTS+08] with the branch-predictor [WX22](`LTAGE`). To see the behavior of the attacks, we also set two levels of the cache hierarchy; Level-one data-cache of size 256kb with level-one instruction-cache of size 128kb and level-two cache size of 512kb.

In stage ④ we simulate setting up all the scripts with the appropriate binaries. We first simulate SE mode, in the simulation mode, we only simulate x86 ISA and ARM as for the transient execution attacks, we use the O3 processors with the branch predictor, our simulation takes 34178552000 ticks for the successful completion of the Specter attack, and we also run the attack in FS mode [LPAA+20].

In the last stage ⑤. off our methodology, we dump the traces of the simulation into a separate file named trace.out file, which includes all the details of what is happening on each

tick and also gives the details about the cache hits, cache misses, which cache line is accessed, and when it is flushed from the cache. Analyzing these traces, we analyze the behavior of the attacks and their patterns. We also use a tool named Konata for the graphical visualization of each trace entry in the trace file that gives us the pipeline visualization. By using this tool we can decode the behavior of the cache-based side channel attacks[YF14] and the transient execution attacks[KHF+20]. For further work, we will use this decoded pattern and analysis of attack traces to make the tool for detecting these attacks automatically.

References

[Dom17] Christopher Domas. Breaking the x86 isa. *Black Hat*, 1:1–6, 2017.

[GS05] John Goodacre and Andrew N Sloss. Parallelism and the arm instruction set architecture. *Computer*, 38(7):42–50, 2005.

[KHF+20] Paul Kocher, Jann Horn, Anders Fogh, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, et al. Spectre attacks: Exploiting speculative execution. *Communications of the ACM*, 63(7):93–101, 2020.

[LPAA+20] Jason Lowe-Power, Abdul Mutaal Ahmad, Ayaz Akram, Mohammad Alian, Rico Amslinger, Matteo Andreozzi, Adrià Armejach, Nils Asmussen, Brad Beckmann, Srikant Bharadwaj, et al. The gem5 simulator: Version 20.0+. *arXiv preprint arXiv:2007.03152*, 2020.

[LTS+08] Jin Li, Kristin Tufte, Vladislav Shkapenyuk, Vassilis Papadimos, Theodore Johnson, and David Maier. Out-of-order processing: a new architecture for high-performance stream systems. *Proceedings of the VLDB Endowment*, 1(1):274–288, 2008.

[WX22] Nan Wu and Yuan Xie. A survey of machine learning for computer architecture and systems. *ACM Computing Surveys (CSUR)*, 55(3):1–39, 2022.

[YF14] Yuval Yarom and Katrina Falkner. {FLUSH+ RELOAD}: A high resolution, low noise, l3 cache {Side-Channel} attack. In *23rd USENIX security symposium (USENIX security 14)*, pages 719–732, 2014.