Strategies for Securing a Memory Hierarchy Against Software Side-Channel Attacks

Amine Jaamoum[1], Thomas Hiscock[1] (Supervior), Giorgio Di Natale[2] (Director)
[1] *Univ. Grenoble Alpes, CEA-LETI, Minatech Campus, F-38054 Grenoble, France*
[2] *Univ. Grenoble Alpes, CNRS, INP, TIMA, 38000 Grenoble, France*

Summary

For many years, microprocessor designers implemented complex architectures to maximize the number of instructions executed per clock cycle. Over the years, architectures evolved from simple finite state machines to very complex out-of-order machines capable of processing hundreds of instructions simultaneously. These processor architectures dedicate a lot of power and chip area to bridge the wide gap between CPU and main memory speeds, known as the "memory wall" [1]. However, security has largely been ignored until recently in the processor microarchitecture of most consumer systems. This popularity attracts attackers, who take advantage of the lack of security features in these system designs.

The beginning of 2018 was marked by the highly publicized revelation of the Spectre and Meltdown attacks. These attacks primarily exploit the predictive behaviors of some modern processor microarchitecture components. The essential factor to the success of these attacks is the ability to leak information through cache memories. Security researchers have known this problem for more than a decade. A first cache timing attack was performed by Bernstein [2] on an AES in 2005. Over the years, many attacks on caches have developed, giving rise to numerous exploitation techniques such as Evict+Time [2], Prime+Probe [3], or Flush+Reload [4].

This thesis aims to design a memory hierarchy that prevents information leakage through cache memories. The challenge is high because it is clear that attacks like Spectre or Meltdown would not be feasible with a robust memory hierarchy against software side-channel attacks (SCAs). A promising solution against information leakage through the cache is to randomize the address-to-cache mapping for each process. However, it is essential for security that the mapping can frequently change, introducing coherence issues (data can be in several locations simultaneously) and increasing cache-miss rate. Both cases degrade the performance.

The first contribution of this thesis was to propose a secure cache architecture for the first level of cache. This work was published at the DATE conference. Our Scramble cache [5] architecture uses dynamic randomization mapping to place data in the cache in a very efficient way. Unlike other architectures, the Scramble Cache employs a lightweight permutation function that requires only a few logic gates, making it ideal for small-embedded devices. Furthermore, with the permutation changing every 10000 accesses, the overhead on the execution time is below 4%, and we have already observed security improvements against cache side-channel attacks. At the moment, we study the behavior of some techniques used to perform some cache side-channel attacks. Furthermore, this study aims to evaluate the security of the memory hierarchy with a randomized lower-level cache (e.g., L1 cache). In addition, this thesis aims to secure also the last-level cache, which is shared among all cores and can leak sensible information.

References

[1] W. A.Wulf and S. A. McKee. Hitting the memory wall: implications of the obvious. ACM SIGARCH.

[2] Tromer, E., Osvik, D. A., & Shamir, A. (2010). Efficient cache attacks on AES, and countermeasures. *Journal of Cryptology*, *23*(1), 37-71.

[3] LIU, Fangfei, YAROM, Yuval, GE, Qian, *et al*. Last-level cache side-channel attacks are practical. In: *2015 IEEE symposium on security and privacy*. IEEE.

[4] Yarom, Yuval, and Katrina Falkner. "FLUSH+ RELOAD: A high resolution, low noise, L3 cache side-channel attack." *23rd USENIX Security Symposium*, 2014.

[5] Jaamoum, Amine, Thomas Hiscock, and Giorgio Di Natale. "Scramble Cache: An Efficient Cache Architecture for Randomized Set Permutation." *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2021.