ON THE EFFECTIVENESS OF INTEL'S CAT AS A SIDE-CHANNEL MITIGATION TECHNOLOGY

Pawel Wieczorkiewicz, Rodrigo Branco and Ben Lee
Open Source Security Inc. and Oregon State University
wipawel@grsecurity.net, rodrigo@kernelhacking.com, ben.lee@oregonstate.edu

FULL PAPER

The full paper is available at:
https://langsec.org/spw24/papers/Pawel_LangSec24.pdf

Abstract

Side-channels are created because contemporary systems share hardware components (e.g., caches) between different users and/or privilege domains. The paper by Yan *et al.* [8] introduced a novel side-channel attack against cache directory structures. They also discussed potential mitigations against their novel attack and suggested that a technique similar to Intel's Cache Allocation Technology (CAT) could be used to partition the directory structure, which hints that CAT as-is might not be enough to mitigate their proposed attack. The objective of this paper is to definitively answer the following questions: Is CAT partitioning an effective barrier against the cache directory attacks? And, are the Cache Directory Attacks possible in more constrained/realistic scenarios? To answer these questions, the work from [8] is reproduced, extended, and improved to run in a virtualized environment as a guest virtual machine (VM) attacking another guest (VM). This paper demonstrates that Intel's CAT is indeed not an effective mitigation against cache-based side-channels.

Introduction

Cache side-channel attacks belong to a more general class of microarchitectural attacks [2]. The basic idea is to exploit the fact that the cache is a shared resource that can be used to leak information. This is typically done by measuring and profiling cache activity, i.e., cache hits and misses. By analyzing cache activity patterns, an attacker can infer secret-dependent code paths or sensitive access patterns of other processes. This information can then be used by an attacker to circumvent higher-level security mechanisms such as privilege separation. Attacks have been demonstrated across many security boundaries, such as cross-cores, cross-VM, cross-users, and cross browser tabs. Therefore, cache side-channel attacks is a powerful threat.

CAT was suggested as a potential cache-based side-channel mitigation in a paper by Intel researchers (with others) [6]. Meanwhile, the authors in [1] suggested that CAT is not an effective mitigation against some cache-based side-channels. They mentioned the work by Yan *et al.* [8] as the reason why CAT is not effective, which introduces a novel side-channel attack that leverages the directory design of Intel systems. Yan *et al.* also discussed mitigations against their novel attack and suggested that a technique similar to CAT could be used to partition the directory structure, which hints that CAT as-is might not be enough to mitigate their proposed attack (since CAT by itself does not partition the directory structure). However, another paper by the same authors, Yan *et al.* [9] included the proposal discussed in [6] (which is purely based on CAT) as a mitigation and pointed out that the main downside is that it requires a predefined split between protected/secured and non-secured domains. In another work, [3], the authors use other data structures (the TLB) as an attack vector (instead of the cache structure) because they claim that finding alternatives to the traditional caches is necessary because CAT is a mitigation to the traditional cache-based side-channels. That demonstrates that experts are, at least, unsure about the effectiveness of CAT.

This work reproduces and expands the previous cache directory attack in an environment with a guest virtual machine running on top of an open-source hypervisor (Xen). This setup is used to demonstrate that CAT is ineffective against such attacks. Our findings have been shared with Intel and a confirmation of our conclusion has been received.

Background

In Figure 1, Core 0 is the attacking core while Core 1 is the victim core. Again, all the green cells represent cache lines brought into the cache hierarchy via Core 0 initiated memory accesses, while pink cells represent memory accesses of Core 1. Shared cache line accesses are omitted from this discussion because the focus is on the hardest to evict cache lines (i.e., the non-shared ones). Empty white cells in the private caches of Core 0 represent cache lines that will be filled with data from the newly accessed addresses. To better understand the memory accesses in Core 0 and the corresponding evictions that occur in the Cache Directory, the affected cache lines are shaded with a gradient color representing the corresponding congruent cache line in the Cache Directory. As an example, a white cell with partial green gradient fill represents a pending memory access to a physical address is congruent with physical address of a cache line private to Core 0, and thus is tracked by the Cache Directory. Similarly, a green cell with a partial pink gradient fill represents a physical address of memory access that brought in a cache line to the private cache of Core 0 was congruent with physical address of cache line in the Cache Directory tracked for Core 1. Figure 1 shows three different states of the private and shared caches of the cores depending on the memory accesses performed by Core 0

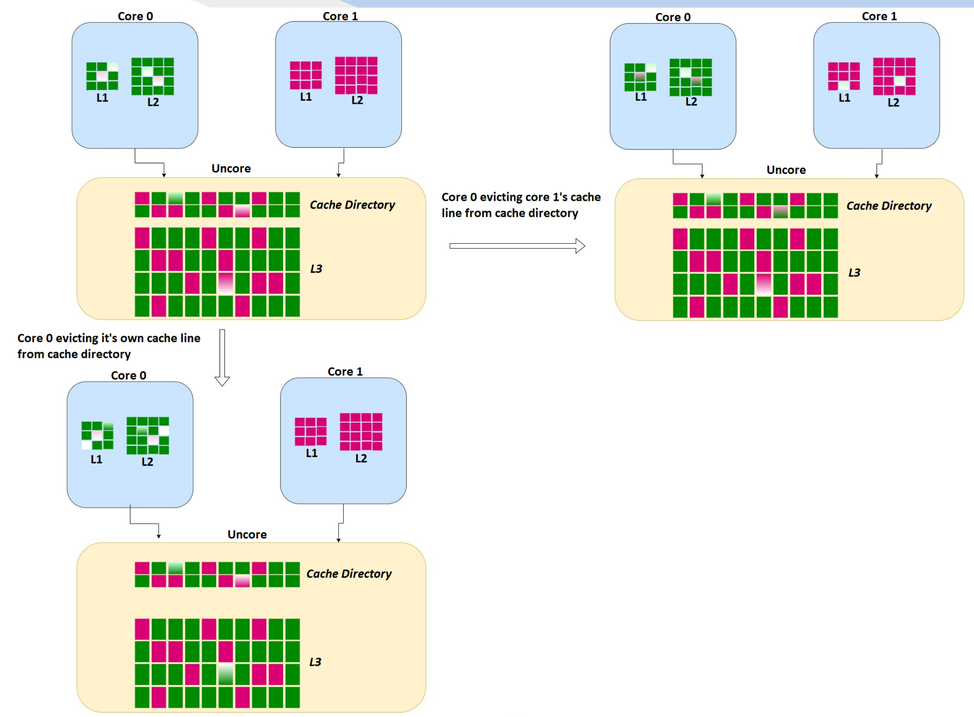

Figure 1: Non-inclusive caches with cache directory.

Evaluation of Intel CAT Against the Cache Directory Attack

Yan *et al.* discussed a few potential countermeasures to the cache directory attack [8]. One solution is to completely eliminate cache directory conflicts by extending its associativity. This is a simple solution, but it is also impractical due to its hardware real estate cost and unknown performance implications. Thus, it is unlikely that the future designs would implement this approach. As an alternative, they also suggested the centralization of the cache directory at the cost of reduced performance and scalability. The last option is to revert back from the directory snooping scheme to the source snooping scheme. However, to the best of our knowledge, such an option is not available in any of the currently implemented designs. The authors further stated that another potential solution would be to partition the cache directory entries among the cores in a manner similar to the way Intel CAT partitions the cache [8]. The same method was also proposed by academics working with Intel researchers in the CATalyst paper [6] against LLC side-channels. In other words, Intel CAT can be used to partition the cache and achieve isolation for the conflicting workloads [5].

Intel CAT is Not a Security Feature

This subsection demonstrates that Intel CAT is *not* an effective mitigation against the cache directory attack.

The cache directory attack was reproduced on a system with the CAT feature enabled and configured. In order to accomplish this, a few extra steps are added to the example mentioned earlier to enable CAT and partition the LLC into two distinct sets of ways – one assigned to the receiver (RX) and the other to the transmitter (TX).

Intel CAT partitioning was applied using the Xen's hypervisor feature Platform Shared Resource (PSR) Monitoring/Control.

Conclusions

The cache directory attack was considered novel, but had some hard constraints on it. In addition, it was not tried against platforms that used Intel's CAT (cache partitioning) as a mitigation. In this work, the original cache directory attack is extended to cover a more practical scenario, considers realistic constraints to the setup, and CAT is used by the hypervisor in order to demonstrate that even a state-of-the-art cloud-based platform would still be vulnerable if it depended only on CAT as a cache-based side-channel mitigation between instances.

In light of the demonstrated attack, the CAT feature can no longer be considered as a viable lonely cache-based side-channel mitigation option, even on systems with non-inclusive cache designs.

A general protection mechanism against cache directory attack is still a subject of research. Protection options to defend secrets at known locations do exist though (such as, marking areas that contain secrets un-cacheable [7]).

As new attack classes are discovered, past mitigations have to be re-visited with the enlightenment of the new attack possibilities/capabilities. Sharing attack code, environments and methods are the ways for the 'offense to drive defense' [4].