文章编号：2095-2163(2021)10-0001-04 中图分类号：TP309 文献标志码：A

缓存侧信道攻击下硬件事件特征分析

王 楷[1]，郭 涛[2]，季振洲[1]
（1 哈尔滨工业大学 计算机科学与技术学院，哈尔滨 150001；2 哈尔滨工业大学 网络与信息中心，哈尔滨 150001）

摘 要：基于硬件事件的异常检测是当前防御缓存侧信道攻击的主要手段之一。然而，现有防御机制普遍未考虑攻击者主动隐藏特征时的检测准确度。本文指出可行的侧信道攻击需要满足不可或缺的驱逐操作和严格的攻击频率这两个先决条件，并发现这些攻击约束会导致被攻击的缓存组访问量急剧增加。实验结果表明，相比于 SPEC 基准程序，侧信道攻击至少会导致 2.61 倍的访问量。

关键词：系统结构；硬件安全；缓存；侧信道攻击；异常检测

Analysis of hardware event characteristics under cache side channel attacks

WANG Kai[1], GUO Tao[2], JI Zhenzhou[1]
(1 School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China;
2 Network and Information Center, Harbin Institute of Technology, Harbin 150001, China)

【Abstract】 Abnormal detection based on hardware events is one of the main means of defending against cache side channel attacks. However, existing defense mechanisms generally do not take into account the detection veracity when attackers actively hide features. This paper points out that feasible side channel attacks require meeting two prerequisites of indispensable eviction operations and strict attack frequency, and finds that these constraints lead to a sharp increase in accesses to the attacked cache set. Compared to the SPEC benchmarks, experiments show that side channel attacks result in at least 2.61 times set accesses.

【Key words】 system architecture; hardware security; cache; side channel attack; anomaly detection

0 引 言

缓存是当前商用处理器芯片中的核心部件之一，有效弥补了较快的处理器运算速度与较慢的内存访问速度的差距。然而，缓存时间侧信道攻击却利用了不同缓存层次和内存间的访问时间差作为信道，推断受害者程序运行过程中的访问模式，进而从系统中提取诸如密钥、浏览记录、用户输入等未授权的敏感信息[1-2]。不仅如此，以“幽灵”为代表的瞬态攻击也将缓存侧信道攻击视为信息泄露链中的重要一环[3]。不断增加的安全威胁表明探索有效的防御机制的迫切性。

异常检测是防御缓存侧信道攻击的主流思想之一。该方法认为攻击者在窃取缓存访问模式的过程中，需要针对目标缓存行构建特定的交互方式，而这会在硬件微架构事件中产生与普通程序存在显著差异的特征。识别这些可疑特征即可有效地定位攻击行为，进而采取低成本的针对性保护操作。目前，学术界已对指令数、缓存缺失和命中率、CPU 周期数等许多硬件事件进行分析，并陆续提出了基于阈值、概率分布、机器学习的探测手段。尽管这些方法在一定程度上缓解了侧信道攻击风险，但普遍未考虑攻击者调整攻击细节，伪造信息流的情况下的检测准确度。在第二节，本文将详细讨论当前典型检测方法的优缺点。

本文指出当前可行的缓存侧信道攻击在每轮探测中需要驱逐和重访操作，并且仅针对被攻击的缓存组，这导致了目标缓存组的访问量增加。另一方面，攻击者还受到了严格的频率约束(500~20 000周期)，这进一步加剧了访问流量。在时钟精准的全系统模拟器 gem5 上，实验结果表明，侧信道攻击至少会导致相当于 SPEC 基准程序 2.61 倍的访问量。

1 缓存侧信道攻击

缓存侧信道攻击能够利用不同缓存层级和内存间的访问延迟感知受害者程序的访问行为，进而窃

基金项目：国家重点研发计划“工控系统安全可信关键技术及应用”重点专项(2020YFB2009500)。

作者简介：王 楷（1992- ），男，博士研究生，主要研究方向：硬件安全、缓存侧信道；郭 涛（1973- ），女，硕士，工程师，主要研究方向：计算机网络；季振洲（1965- ），男，博士，教授，博士生导师，主要研究方向：计算机系统结构。

通讯作者：季振洲 Email：jizhenzhou@ hit.edu.cn

收稿日期：2021-05-24

取敏感信息。最初,侧信道攻击的观测目标是一级私有缓存,但这种攻击场景由于需要攻击者和受害者同驻在相同物理核,具有很大的局限性。直到 2014 年,Yarom 等人提出了针对末级缓存的跨核缓存攻击。由于跨核攻击具备噪声低、易于发起、带宽高的优势,自此之后,有关缓存侧信道的攻防研究重心转移到了跨核攻击。

依据攻击过程中驱逐手段的差异,跨核缓存攻击广义上可划分为基于竞争的攻击和基于冲刷的攻击。目前,学术界已证明不同缓存架构(包含型和非包含型)都会受到上述两类攻击的威胁,只是攻击细节存在差异。本文以包含缓存架构作为基准系统,分析和验证侧信道攻击下存在的显著特征。在未来工作中,将进一步分析非包含缓存架构中的硬件事件特征。

1.1 基于冲刷的攻击

图 1 描述了 Flush + Reload,这一典型的基于冲刷的攻击实例。这类攻击能够确定特定时间窗口内攻击者和受害者共享的缓存行的访问模式。由图 1 可知,其攻击过程包含 3 个步骤:

(1)驱逐:在驱逐阶段,攻击者使用 clflush(x86 指令集下)或其他等效指令将目标缓存行从缓存清除。该指令能够在任何特权级使用,并且能保证被操作的缓存行在各级缓存中都被清除。

(2)等待:在攻击者等待阶段,如果受害者访问了目标地址,该缓存行会迁移进入缓存。否则,缓存不会记录该数据。

(3)重访:攻击者最后重访目标缓存行并计算延迟,若出现缓存命中(延迟较短)则表明受害者曾经访问过目标缓存行。否则,攻击者能够确定受害者未访问对应数据。

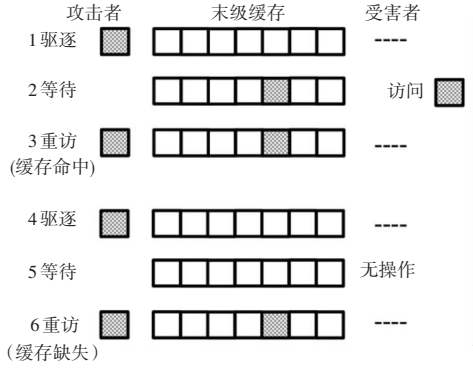

图 1 Flush + Reload 攻击

Fig. 1 Flush + Reload attacks

1.2 基于竞争的攻击

图 2 给出了 Prime+Probe,这一典型的基于竞争的攻击实例。这类攻击可以观测特定时间窗口内目标缓存组的访问模式。相比于基于冲刷的攻击,该攻击克服了共享内存的约束,因而更为可行,也更具威胁。拟对此展开研究分述如下。

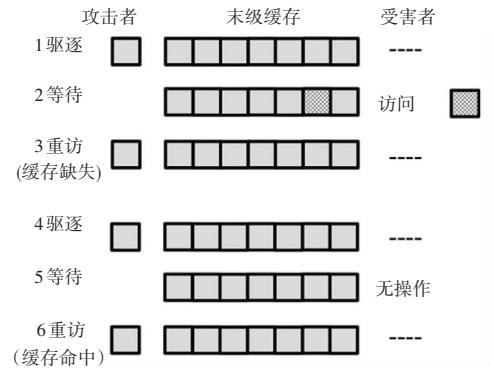

图 2 Prime+Probe 攻击

Fig. 2 Prime+Probe attacks

(1)驱逐:在驱逐阶段,攻击者使用预先准备好的驱逐集合占据目标地址所在的缓存组,利用组冲突将目标缓存行清除。为实现该目标,驱逐集合应包含与末级缓存路数相等的元素,并且每个元素都应与目标地址映射到相同缓存组。

(2)等待:在等待阶段,如果受害者访问了目标地址。无论缓存采用何种替换策略,该操作必然会移除驱逐集中的一个元素。否则,对应缓存组依旧存储完整的驱逐集。

(3)探测:攻击者测量重访驱逐集的延迟。如果延迟较长表明重访行为中出现了缓存缺失事件,进而可以推断受害者访问了目标缓存组。

2 基于异常检测的侧信道防御方案

异常检测最初采用阈值比对的方法识别缓存攻击。CacheShield 认为程序在未被攻击时缓存缺失数不会出现显著变化,并提出基于变点检测理论的检测算法分析该指标是否达到阈值[4]。Biscuit 指出被攻击程序在运行时会比未受到攻击时增加 5 倍的缓存缺失数。基于这一观测,Biscuit 使用编译器分析程序中每个循环产生的缓存缺失数量,并将该值插入到程序中。在程序后续运行时,若实际产生的缺失数量超过期望值,则认为可能遭受攻击[5]。然而,基于阈值比对的方法对运行环境敏感,可能会产生较多误报,并且容易被绕过。

Chiappetta 等人[6]扩大了采样范围,使用总指令数、CPU 周期数、二级缓存命中、末级缓存命中和缺失数五个评价指标。同时,该方法使用均值和方差而不是简单的数值对比识别攻击。

Fortuneteller 认为程序运行过程中微架构事件具有时间先后特性，因此使用门控循环单元和长短期记忆网络这两个循环神经网络预测选定的微架构事件（一级指令缓存命中数、一级指令缓存缺失数和末级缓存缺失数）在下一时刻的数值。如果在设定的滑动窗口内超过阈值次数过多，则认为存在恶意程序[7]。此外，Fortuneteller 从 37 个硬件事件中自动化地筛选出了上述三个最具关联性指标。该方案在检测缓存侧信道攻击的同时，还能够覆盖幽灵[3]、rowhammer[8]等典型的硬件漏洞。

对于常规的缓存侧信道攻击，上述机制在准确度和降低性能开销维度方面已取得了不错的进展。例如，Fortuneteller 的 $F-score$ 已达到 0.997 并且仅对 Phoronix 测试程序造成平均 0.12% 的性能损失。然而，这些检测方法并未考虑如何应对自适应攻击，即攻击者在感知系统中部署了防御机制后，伪装信息流，避免触发指标异常的攻击场景。本文的目标是分析并验证侧信道攻击中无法隐藏的硬件事件及其特征，启发研究人员探索更有效的防御方案。

3　攻击行为分析

通过分析攻击流程，本文发现无论是基于冲刷的攻击还是基于竞争的攻击在进行探测时都需要满足如下 2 个先决条件：

（1）不可避免的驱逐操作。攻击者在每轮发起攻击前需要确保被探测的缓存行已从缓存清除。否则，受害者随后的访问行为会出现缓存命中事件。由于缓存状态未发生改变，攻击者将无法感知到受害者的行为。

（2）严格的攻击频率。在每轮迭代中，攻击者仅能确定目标缓存行或缓存组是否被受害者访问，这一有限的信息量并不足以引起敏感信息泄露。为获得完整信息，攻击者需要重复完成多次攻击。例如，在 Liu 等人[2]提出的 Prime + Probe 攻击中，攻击者实施了上万次探测才能复原出 3 072 位密钥信息。当需要发动多轮攻击时，攻击者就需要考虑攻击频率的问题。一方面，攻击间隔过长可能导致受害者在此期间发生了多次与敏感信息相关的操作，导致错失信息。另一方面，攻击间隔过短会增加攻击者的驱逐和重访等操作与受害者的访问行为发生冲突的可能，引入新的噪声。通常，现有攻击的间隔为 500~20 000 周期。

4　筛选硬件事件

异常检测需要选择对攻击程序和正常程序有良好区分度的硬件事件作为识别器。如前所述，已有许多研究使用系统运行中的统计事件，尤其是缓存相关事件（缓存命中、缺失数量[9]等）识别侧信道攻击。然而，攻击者可能会优化攻击，避免上述指标出现异常。在侧信道攻击的 3 步流程中，攻击者会参与其中的第一和第三步。本节将从中挖掘基于冲刷和基于竞争的攻击存在的共性特征。具体论述如下。

（1）基于冲刷的攻击。基于冲刷的攻击的缓存事件集中在重访操作。由于受害者行为的差异，攻击者的重访操作必然会导致一次缓存命中或缺失。

（2）基于竞争的攻击。基于竞争的攻击的缓存事件集中在驱逐操作。本节将在 LRU（Least Recently Used）替换策略下进行分析。对于其他类型的缓存替换策略，已有研究证明相应策略将会在特殊的攻击手段下退化为 LRU[8]。

假定 $e_1e_2e_3...e_w$ 是目标缓存组的驱逐集而 v 是目标缓存行。在开启下一轮探测前，假定缓存组存储的元素分别是 $e_1e_2e_3...e_{w-1}v$，该存储顺序同时也代表了 LRU 状态，即发生缓存替换时，下一个被移除的元素是 e_1。在驱逐阶段，攻击者迭代访问整个驱逐集以确保目标地址 v 被移除。尽管访问驱逐集的顺序不会改变驱逐的效果但会影响微架构的统计信息。如果访问序列是 $e_1e_2e_3...e_w$，这会导致 $w-1$ 次缓存命中和一次缓存缺失。翻转访问顺序则会带来 w 次访问缺失。

从上述分析可以发现，攻击行为必然会导致缓存命中或缺失事件，并由于严格的攻击频率使得出现次数急剧增加。但是，单独的缓存命中或缺失事件无法完全覆盖攻击者的行为。此外，攻击者还可以引入无关的访存操作动态改变缺失率和命中率。

尽管攻击者有可能影响缓存命中和缺失的数量，但缓存访问量（缓存命中和缺失事件总和）是其无法隐藏的特征。由于物理地址到缓存的映射关系固定，目标地址每次都会落入到同一个缓存组，这也使得这些访问量必然集中在对应的缓存组。因此，本文建议使用末级缓存组访问量作为异常检测的关键指标。

5　实验评估

5.1　实验方法

本节在时钟精准的全系统模拟器 gem5 中统计了 SPEC CPU2006 基准程序和 2 类跨核缓存攻击在末级缓存组的访问量。该模拟器配置了一个使用 3 层包含缓存和 x86 指令集的系统。其中，一级缓存

和二级缓存为每个核私有,逻辑共享的末级缓存在物理上划分为与核数相等的分片。表 1 列出了详细的参数配置。

表 1 参数配置

Tab. 1 Parameter configurations

参数	配置
主频	2.0 GHz
指令集	x86
一致性协议	MESI
私有一级缓存	32 KB/core, 4 ways, 2 cycles
私有二级缓存	256 KB/core, 8 ways, 18 cycles
共享三级缓存(包含属性)	1 MB/slice, 16 ways, 35 cycles

本节使用 reference 输入运行 SPEC 测试集。对于每个程序,gem5 采样其有代表性的十亿条指令,并统计末级缓存组的访问量。

本节还构建了 Flush+Reload 和Prime+Probe,并统计攻击行为下各缓存组访问量。上述攻击会以 20 000周期为间隔从 square-and-multiply 算法中窃取密钥。其中,Flush+Reload 攻击依据图 1 所述流程监控目标缓存行;Prime+Probe 攻击依据图 2 所述流程监控目标缓存组。

5.2 实验结果

图 3 统计了测试集和攻击程序每百万周期中末级缓存组平均出现的访问量。目标缓存组在Prime+Probe 和 Flush+Reload 攻击中分别会出现约403.1和73.2 次访问操作。在相同攻击频率下,以 Prime + Probe 为代表的基于竞争的攻击会带来更显著的访问量。这一现象主要由于基于竞争的攻击在驱逐阶段需要利用缓存组冲突和替换策略清除目标缓存行。此外,由于攻击行为仅聚焦目标缓存组,因此其他组的访问量趋向 0。

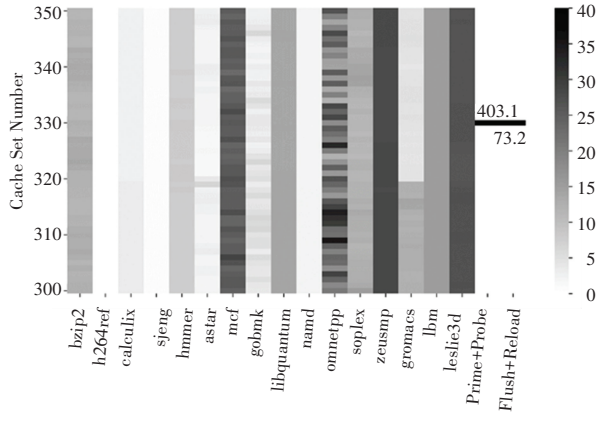

图 3 每百万周期末级缓存组(300~350)访问量

Fig. 3 Access count per one million cycles for last level cache sets (300~350)

测试集程序则不存在这种异常现象,且在每个缓存组的平均访问量都小于 30。其中,zeusmp 在所有测试集中缓存组访问量最多,约为 28.06。Prime + Probe 和 Flush + Reload 的访问量分别是该值的 14.37 倍和 2.61 倍。因此,末级缓存组访问量是一种对恶意程序和常规程序具有良好区分度的评价指标,可以用于识别侧信道攻击。

6 结束语

本文指出当前基于硬件事件的缓存侧信道攻击检测机制并未考虑攻击者隐藏特征时的准确度。本文发现侧信道攻击需要保证每轮的驱逐操作和严格的攻击频率,而这两个约束条件会导致目标缓存组的访问量不可避免地增加。实验结果表明侧信道攻击至少会产生相当于 SPEC 基准程序 2.61 倍的缓存组访问量。这说明缓存组访问量是一种对恶意程序和常规程序具有良好区分度的评价指标。在后续工作中,本研究将探索利用该指标检测和防御侧信道攻击的方案。

参考文献

[1] YAROM Y, FALKNER K. FLUSH+ RELOAD: A high resolution, low noise, L3 cache side-channel attack[C]//23rd USENIX Conference on Security Symposium (SEC'14). Berkeley, CA, USA: ACM, 2014: 719-732.

[2] LIU Fangfei, YAROM Y, GE Qian, et al. Last-level cache side-channel attacks are practical[C]//2015 IEEE Symposium on Security and Privacy. San Jose, CA, USA: IEEE, 2015: 605-622.

[3] KOCHER P, HORN J, FOGH A, et al. Spectre attacks: Exploiting speculative execution[C]//2019 IEEE Symposium on Security and Privacy (SP). San Francisco, CA, USA: IEEE, 2019: 1-19.

[4] BRIONGOS S, IRAZOQUI G, MALAGÓN P, et al. Cacheshield: Detecting cache attacks through self-observation[C]//Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy. New York, USA: ACM, 2018: 224-235.

[5] KHAN S, MRURU G, PANDE S. A Compiler assisted scheduler for detecting and mitigating cache-based side channel attacks[J]. arXiv preprint arXiv: 2003.03850, 2020.

[6] CHIAPPETTA M, SAVAS E, YILMAZ C. Real time detection of cache-based side-channel attacks using hardware performance counters[J]. Applied Soft Computing, 2016, 49(C): 1162-1174.

[7] GULMEZOGLU B, MOGHIMI A, EISENBARTH T, et al. Fortuneteller: Predicting microarchitectural attacks via unsupervised deep learning[J]. arXiv preprint arXiv: 1907.03651, 2019.

[8] GRUSS D, MAURICE C, MANGARD S. Rowhammer. js: A remote software-induced fault attack in javascript[C]// Proceedings of the 13th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Cham: Springer, 2016, 9721: 300-321.

[9] KULAH Y, DINCER B, YILMAZ C, et al. SpyDetector: An approach for detecting side-channel attacks at runtime[J]. International Journal of Information Security, 2019, 18(4): 393-422.