处理器硬件漏洞研究综述

蓝泽如[①②] 邱朋飞*[①②④] 王春露[①②] 赵娅喧[①②③④] 金 宇[①②] 张志昊[①②] 汪东升

[①](北京邮电大学网络空间安全学院 北京 100876)
[②](北京邮电大学可信分布式计算与服务教育部重点实验室 北京 100876)
[③](清华大学计算机科学与技术系 北京 100084)
[④](中关村实验室 北京 100194)

摘 要：处理器安全是计算机安全的基石，然而，最近几年处理器硬件漏洞层出不穷，给计算机安全带来了严重的挑战，已成为一种新兴的安全威胁。该文对处理器硬件漏洞相关的研究进行综述与分析，首先介绍导致处理器硬件漏洞的性能优化技术，然后基于现有工作从漏洞发现、攻击实现和攻击利用3个角度对硬件漏洞总结建立3步攻击模型和全新的攻击场景模型，并总结提出4种常用的漏洞发现方法。随后，依据硬件漏洞的攻击路径与微架构依赖特征，将现有硬件漏洞进行系统性地分类，并对每一类别进一步细分与详述。此外，在现有 Cache侧信道攻击九项评估指标的基础上，该文总结形成针对侧信道类漏洞的八项评估指标，从攻击效果、隐蔽性、适用性等多个维度综合评估其潜在威胁，为漏洞的量化分析与对比研究提供指引。更重要的是，该文在多款来自不同厂商、涵盖多种微架构和代际的处理器平台上，对代表性的漏洞进行了实验验证，系统分析了各类漏洞在不同平台下的攻击效果与行为特征。然后，该文基于漏洞攻击流程中的3个关键步骤，整理了当前已提出的缓解方案的缓解思路，为后续防护机制的设计与优化提供了实证支撑与理论指导。最后，对当前硬件漏洞研究的进展与趋势进行了分析与前瞻性讨论。

关键词：硬件漏洞；计算机体系结构安全；侧信道；瞬态执行；故障注入

中图分类号：TN918; TP309 **文献标识码**：A **文章编号**：1009-5896(2025)00-0001-18

DOI: 10.11999/JEIT250357 **CSTR**: 32379.14.JEIT250357

1 引言

处理器安全是计算机系统安全的基础，为上层系统与应用提供可信的运行环境。然而，随着处理器微体系结构日趋复杂，为追求更高性能而广泛引入的各类优化机制，在缺乏充分安全考量的情况下，逐渐暴露出潜在的安全隐患。这些优化技术往往以性能和能效为优先设计目标，忽略了对潜在攻击面的系统性评估，从而引发了大量微架构层面的安全漏洞，成为当前体系结构安全研究的核心挑战之一。

近年来，随着 Spectre[1]和 Meltdown[2]等高影响力硬件漏洞的发现与公开，深刻动摇了人们长期以来对处理器硬件边界和隔离机制的信任假设。这些漏洞揭示了现代处理器在性能优化设计中所引入的微架构行为，可能成为攻击者利用的突破口，从而实现对敏感信息的越权访问与泄露。因此，学术界与工业界对硬件漏洞的攻击原理、漏洞发现方法以及防御机制展开了系统性和持续性的研究。

侧信道攻击是硬件安全领域的核心研究议题之一。早期的侧信道攻击研究主要集中在分析缓存访问时序差异，通过揭示资源共享机制中的隐蔽信息泄露风险，暴露了现代处理器在微架构设计中的安全弱点[3–10]。随后，研究者发现了多种基于转译后备缓冲器(Translation Lookaside Buffer, TLB)[11,12]、性能监控单元(Performance Monitoring Unit, PMU)[13,14]以及能耗特征[15,16]的侧信道攻击手法，拓展了侧信道攻击的攻击面。2018 年，硬件安全研究迎来重要转折， Meltdown漏洞与 Spectre漏洞的披露揭示了瞬态执行(transient execution)机制在现代处理器中的深层次安全隐患，引发了学术界与工业界对微架构安全的持续关注。这些漏洞利用了处理器引入的预测执行(speculative execution)、乱序执行(out-of-order execution)等优化机制。当预测或乱序执行错误时，瞬态执行在微架构层面留下的痕迹不会被处理器主动清除，从而可能被攻击者利用。这些在预测或乱序执行阶段被执行但最终

收稿日期：2025-05-06；改回日期：2025-xx-xx
*通信作者： 邱朋飞 qpf@bupt.edu.cn
基金项目：北京市自然科学基金(4242026)，国家自然科学基金(62372258)，中央高校基本科研业务费专项资金(2023RC71)
Foundation Items: Beijing Natural Science Foundation (4242026), The National Natural Science Foundation of China (62372258), The Fundamental Research Funds for the Central Universities (2023RC71)

未提交的指令被称为瞬态执行指令，利用这些指令进行的信息泄露攻击被称为瞬态执行攻击。此后，瞬态执行类硬件漏洞的研究迅速成为新的热点方向[17–21]。与此同时，针对故障注入类硬件漏洞的研究也在持续推进。2017年，硬件漏洞的研究开始扩展至电源管理域，攻击者利用动态电压与频率调整(Dynamic Voltage and Frequency Scaling, DVFS)过程中的安全缺陷，能够操控频率和电压来实现故障输入，进而突破处理器的安全防护[22–25]。

尽管近年来硬件漏洞相关研究取得了显著进展，但仍存在若干关键问题亟待解决。首先，当前硬件漏洞种类繁多、实现方式复杂，而现有文献在漏洞分类体系方面尚不完善，缺乏统一、系统的分类框架。其次，目前研究普遍侧重于漏洞的个体发现或攻击实现，缺乏对漏洞生命周期的整体建模与系统性分析，尚未形成从漏洞识别、攻击实例化到漏洞利用的完整研究链条。此外，与软件漏洞相比，硬件漏洞源于处理器微架构设计阶段的结构性缺陷，其修复难度更高，防御成本更大。现有缓解方案多呈现出针对性强且缺乏统一协调的特征，尚未形成针对多类型漏洞的综合防御体系。

因此，本文对 2010年以来发表在学术界主要体系结构安全会议和期刊上的处理器硬件漏洞进行了系统的归纳和分析。首先系统总结了4类主要的漏洞发现方法，并在现有工作基础上总结了一种硬件漏洞3步攻击模型[26,27]与全新的攻击场景模型。然后，按照攻击行为特征进一步将现有硬件漏洞进行分类和描述，并结合现有相关工作针对侧信道类漏洞总结了8项综合评估指标[28,29]。为了验证不同类型攻击的可行性与影响范围，本文选取了部分代表性漏洞，在多款处理器平台上进行了实验验证，并深入分析了实验结果。此外，还系统归纳了当前已有的硬件漏洞防御的缓解思路。最后，从攻击和防御两个方面展望了硬件漏洞研究的未来方向。

本文主要贡献如下：

(1)从漏洞发现、攻击实现以及漏洞利用3个维度出发，总结形成一个硬件漏洞3步攻击模型和全新的攻击场景模型，全面描述了漏洞从被发现到被实际利用的全过程，并总结归纳了主流硬件漏洞的四种发现方式，为后续研究提供理论基础与分析框架。

(2)依据硬件漏洞的攻击路径与微架构依赖特征，将现有硬件漏洞划分为3大类：侧信道类、瞬态执行类和故障注入类，并对每一类别进一步细分与详述。此外，结合现有高速缓存(Cache)侧信道攻击评估指标形成了针对侧信道类漏洞的8项评估指标，从攻击效果、隐蔽性、适用性等多个维度综合评估其潜在威胁，为漏洞量化分析与对比研究提供指引。

(3)在多款来自不同厂商、涵盖多种微架构和代际的处理器平台上，对具有代表性的漏洞样例进行了深入实验验证，系统分析了各类漏洞在不同平台下的表现差异与行为特征。同时，基于漏洞攻击流程中的3个关键阶段，评估了当前已提出缓解方案的缓解思路，为后续防护机制的设计与优化提供了实证支撑与理论指导。

2 背景知识

随着计算机技术的不断发展与普及，提升处理器性能成为处理器设计领域的核心研究方向之一。现代处理器已广泛采用高速缓存、乱序执行、预测执行等先进的性能优化技术，这些优化技术显著提升了处理器的计算效率和指令吞吐量。然而，在提高性能的同时，这些优化机制也可能引入新的安全隐患，使处理器在特定条件下容易受到侧信道攻击等安全威胁的影响[1,2]。

2.1 多级缓存架构

Cache是一种位于处理器与主存之间的小型高效存储器，主要负责存储处理器最近或频繁访问的指令和数据，使处理器能够减少对主存的访问，从而提升系统性能。现代处理器普遍采用多级层次架构的高速缓存体系，通常分为3级结构，即 L1 Cache, L2 Cache和 L3 Cache，其中 L3 Cache也被称为最后一级缓存(Last Level Cache, LLC)，这三者在容量和访问速度上存在显著差异。

2.2 乱序执行

早期的处理器采用顺序执行方式会导致流水线阻塞，并在指令等待期间产生较大的执行延迟，从而显著降低处理器的处理效率。为了解决这个问题，现代处理器引入了乱序执行优化技术[2]。乱序执行允许处理器根据数据依赖关系及计算资源的可用性，动态调整指令的执行顺序。通过减少流水线阻塞并提高指令级并行性，乱序执行能够有效提升处理器的计算性能与执行效率。

2.3 预测执行

为了解决指令流水线在分支决策完成前会处于阻塞状态的问题，现代处理器引入了预测执行技术[1,17–19]。在采用预测执行的处理器遇到可能引发流水线停滞的情况时，该处理器会基于分支预测的结果提前执行可能的指令路径，从而减少等待时间。若预测结果正确，则执行结果直接提交。若预测错误，则回滚错误路径上的计算，并重新执行正确的指令。

2.4 多线程同步

在传统的单线程处理模式下，每个物理核心在任意时间点只能执行1个线程。当线程因数据依赖等因素进入等待状态时，处理器的部分计算单元仍将处于空闲状态，导致计算资源未被充分利用，从而影响整体性能。为了解决这一问题，现代处理器引入了多线程同步技术(Simultaneous MultiThreading, SMT)[30]。SMT可以使多个线程能够在同一时间内共享计算单元，从而减少因计算单元闲置造成的性能损失。

2.5 动态电压与频率调节

为实现高效的能耗管理，现代处理器广泛采用DVFS技术[22–25]。DVFS通过动态调整处理器的工作频率和电压，使其能够根据当前计算负载在性能与能耗之间达到优化平衡。处理器会持续监控功耗和温度，一旦负载状态发生变化，系统会重新调整电压和频率，以确保性能与能耗的动态平衡。

3 攻击模型

在硬件漏洞研究中，攻击的全过程通常包括漏洞的发现、漏洞的实例化以及最终的漏洞利用，三者构成了完整的攻击链条，且每一环节均具有重要的研究价值。为了系统地分析硬件漏洞的影响及其演化路径，并进一步揭示第2节所述性能优化机制所引入的安全隐患，本文结合已有工作形成了一个通用的“3步攻击模型”，用于刻画硬件漏洞攻击的基本流程[26,27]。同时，构建了攻击场景模型，以描述漏洞在不同系统架构中所针对的攻击目标与可能造成的危害。此外，本文总结了主流硬件漏洞的4种发现方式，展示了当前研究在漏洞挖掘阶段的主要技术路线。

3.1 硬件漏洞3步攻击模型

为系统分析此类漏洞的攻击路径，本文在综合大量相关研究工作的基础上，形成了一个通用的硬件漏洞3步攻击模型。该模型将攻击过程划分为3个关键步骤：数据准备、数据传输和数据恢复，分别对应于攻击者初始化敏感数据访问、构建隐蔽传输通道，以及通过特定手段恢复私密信息的过程，本模型为理解和分析硬件漏洞提供了统一的理论框架。

第1步：数据准备。处理器的执行环境具有高度复杂性与动态性，非攻击性操作所带来的系统噪声可能严重干扰硬件漏洞的利用效果。因此，数据准备作为大多数硬件漏洞攻击流程中的第1步，往往涉及对攻击环境的初始化与控制。攻击者需预先初始化共享硬件组件中的存储状态，以消除对后续数据恢复对潜在干扰。

第2步：数据传输。在第2步数据传输过程中，攻击者通过精心构造的指令序列将目标私密数据加载至微架构组件中，并借助处理器的性能优化机制实现信息的隐蔽传输。在此过程中，攻击者会通过特定的数据访问操作将敏感信息加载到共享资源的特定位置，为后续的数据恢复创造条件。

第3步：数据恢复。硬件漏洞攻击的第3步是数据恢复。在这一过程中，攻击者需要通过特定的技术手段从微架构中恢复先前泄露的敏感信息。为了减小由于系统噪声或测量误差引发的不确定性，攻击者通常会采用统计分析的方法，通过对同一攻击流程的重复执行，并对每个候选数据进行频次统计，最终选取出现频率最高的值作为恢复结果。

综上所述，在数据准备、数据传输和数据恢复3个攻击步骤的有机协同下，攻击者能成功实现对处理器中私密信息的提取。硬件漏洞的3步攻击模型不仅验证了攻击路径的有效性，也凸显了微架构级安全防护中仍存在的挑战。

3.2 硬件漏洞攻击场景模型

仅完成3步攻击尚不足以构成完整的攻击链条。为了实现漏洞的有效利用，攻击者需在真实系统中应对多种复杂条件与防护机制。因此，本文在相关工作的基础上，进一步总结提出了一个全新的攻击场景模型，如图1所示。该模型归纳了6类典型攻击场景，并在此基础上细化为 27种具体情形，以全面覆盖不同硬件漏洞在实际利用中的可能路径与条件。表1汇总了本文对漏洞攻击场景的分类结果。

场景 1：跨线程场景。跨线程是现代处理器硬件漏洞研究中的一种关键攻击场景，广泛存在于采用 SMT的处理器中。在此类场景中，攻击者可以利用不同线程之间共享底层硬件资源的特性在逻辑线程间建立隐蔽信道，从而实现对其他线程私密信息的窃取[31]。

场景 2：跨进程场景。与跨线程场景相比，跨进程场景的适用范围更广，其攻击进程与受害进程在逻辑上相互独立，可以运行在不同的用户空间上下文中。尽管如此，攻击者仍可通过利用共享的微架构资源建立隐蔽通道，实现对敏感信息的窃取[32]。

场景 3：跨用户-内核边界场景。现代操作系统通常采用用户态与内核态分离的权限模型，以确保用户进程在常规运行过程中无法直接访问内核空间的数据。然而，处理器为提升指令执行效率而引入的性能优化机制在特定条件下能暴露内部状态信息，进而为攻击者构建跨权限的攻击通道提供了可能性。跨用户-内核边界攻击场景正是基于此特性，

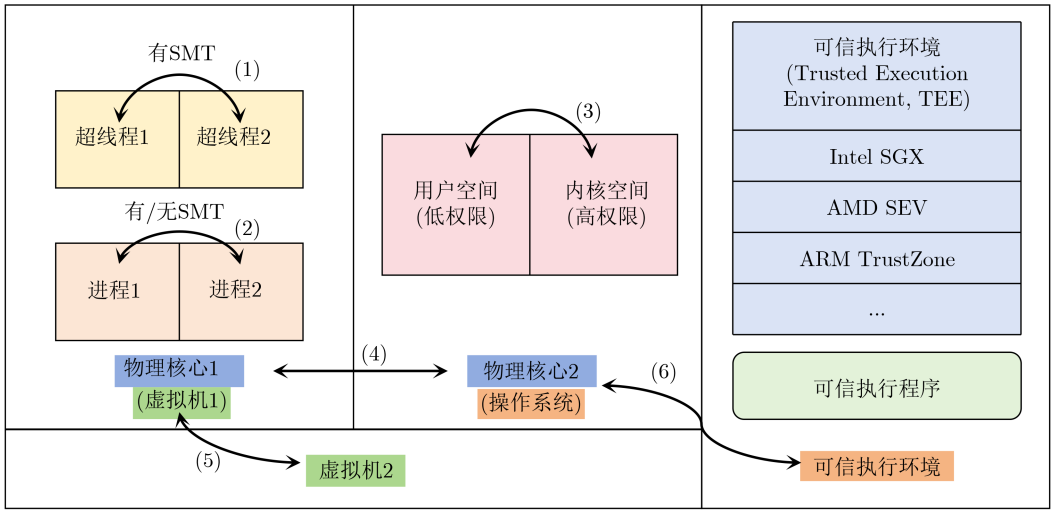


图 1 硬件漏洞攻击场景模型

表 1 漏洞攻击场景分类汇总

典型攻击场景	包含具体场景	具体场景		
(1) 跨线程	③⑦	① 攻击内核地址空间布局随机化(Kernel Address Space Layout Randomisation, KASLR)	⑩ 远程缓存服务器	⑲ 绕过Intel CAT、Intel TSX缓解措施
(2) 跨进程	②④⑤⑥⑦⑧⑩⑫⑭⑮㉖	② 恢复RSA密钥	⑪ 泄露私密数据	⑳ 针对KVM的攻击
		③ 恢复 (ED/EC)DSA密钥	⑫ 攻击OpenSSH公钥认证、攻击 apt-get	㉑ 推断CNN架构
(3) 跨用户-内核边界	①②④⑤⑥⑦⑧⑩⑫⑭⑮⑯⑰⑱㉒㉖	④ 恢复AES-NI密钥	⑬ 攻击JIT引擎	㉒ EPID 私人验证密钥
		⑤ 恢复AES密钥	⑭构建隐蔽信道	㉓攻击libjpeg库
(4) 跨核	②⑤③⑭㉕	⑥ 恢复DES密钥	⑮ 沙箱逃逸	㉔ 攻击 SMM、Ring-0 控制结构攻击、引导加载器/BIOS阶段攻击
(5) 跨虚拟机	②⑤⑫⑱⑳	⑦ 网站/端口指纹识别	⑯ 攻击后量子密码系统	㉕ 温度感知攻击
(6) 跨TEE	①②③④⑤㉓㉖	⑧ 探测击键时间	⑰ 转储物理内存	㉖ 跨TEE加载RSA自签名代码
		⑨ 攻击Web浏览器	⑱ 获取内核数据	㉗ 蓝牙事件，鼠标移动

通过侧信道攻击打破特权隔离，从而恢复内核态中的敏感信息[2]。

场景 4：跨物理核心场景。现代多核处理器虽然在每个核心中配置了相对独立的执行单元与资源，但多个核心之间可共享系统级的硬件资源，这些共享资源为攻击者提供了构建侧信道的可能路径。在跨物理核心场景下，攻击者可通过对共享资源访问行为的监测与分析，跨越物理核心窃取受害者敏感信息。

场景 5：跨虚拟机场景。在跨虚拟机攻击场景中，攻击者与受害者分别处于不同的虚拟机实例中，尽管二者在逻辑空间、操作系统层级乃至虚拟化管理权限上均实现了严格隔离，但由于底层硬件资源的共享特性，攻击者仍可通过构建微架构侧信道实现对受害虚拟机的越界信息窃取和执行行为推测。

场景 6：跨可信执行环境场景。尽管可信执行环境(Trusted Execution Environment, TEE)被设计为一种高度隔离的安全执行区域，广泛应用于密钥管理、加密计算等场景，并被认为具有较强的安全保障能力。但是攻击者仍可通过侧信道攻击等手段绕过 TEE的访问控制机制，窃取其内部的私密信息。

3.3 硬件漏洞4种发现方式

在第 3.1节与第 3.2节中，本文分别归纳了硬件漏洞的3步攻击模型与攻击场景模型，系统地描述了漏洞在利用过程中的行为链条与潜在的攻击路径。然而，仅从攻击利用的角度出发，仍不足以构建一个完整的硬件漏洞研究框架。漏洞的发现过程是硬件安全研究的关键起点，决定了研究的深度与广度。因此，本文进一步结合已有研究成果与典型案例，总结了当前主流的4种硬件漏洞发现方法。

方式 1：硬件资源共享泄露。处理器微架构设计中的重要优化策略之一是硬件资源共享，其核心目标在于提高硬件资源的利用效率，优化计算性

能。在现代处理器中，多个关键功能单元(如 Cache、执行单元、预取单元等)通常由不同的指令流或进程共享。然而，这种共享机制引入信息泄露风险。攻击者可以通过监测共享资源的访问时间等微架构特征，从而构建侧信道攻击。在某些优化机制下，原本用于隔离不同进程或用户的安全边界可能被削弱，导致私密数据暴露给恶意进程[1,2,4–7]。

方式 2：显性和隐性陈旧状态利用。为了提高计算性能，现代处理器通常会保留特定状态信息，而这些状态不会被及时清除，攻击者可利用这些残留状态实施安全攻击。例如，陈旧的缓存数据可能仍然存留于 Cache中，攻击者可以通过 Cache侧信道攻击推测受害进程先前访问的内容，从而窃取机密信息[1,2,17–19]。此外，微架构部件中残留的状态信息，也可能被恶意利用，导致权限绕过或数据泄露。

方式 3：低功耗故障注入测试。为有效应对能耗增加带来的挑战，DVFS已广泛应用于现代处理器架构中，旨在在保证系统性能的同时最大限度地降低功耗。处理器在一定的电压和频率范围内可保证稳定运行，但若运行条件超出安全边界，可能会引发数据延迟或计算错误，从而影响系统可靠性。基于此特性，攻击者可利用 DVFS机制发起故障注入攻击，即在特定时间窗口内对电压或频率进行瞬时扰动，诱导处理器产生计算错误，并通过分析错误结果推断敏感信息，进而威胁系统的机密性与完整性[22–25]。

方式 4：模糊测试与形式化验证。针对未知硬件漏洞的检测已成为保障系统安全的关键环节。在硬件漏洞检测中，研究人员通常采用动态分析方法(模糊测试)和静态分析方法(形式化验证)。在实际应用中，单一漏洞检测技术往往难以覆盖所有潜在威胁，因此现代硬件安全研究通常结合模糊测试与形式化验证，以提高漏洞检测的深度与准确性。

4 硬件漏洞分类

本文对现有的处理器硬件漏洞进行了系统性调研，并基于第 2节介绍的处理器优化技术及第 3节提出的攻击模型与漏洞发现方法，对这些硬件漏洞的特征进行了归纳与分类，分类结果如表2所示。本文将现有的处理器硬件漏洞分为侧信道类、瞬态执行类和故障注入类，这3类漏洞覆盖了第 3节中描述的攻击模型与漏洞发现方式。第 4.1～4.3节将分别对表2中所列的处理器硬件漏洞分类进行详细阐述，以进一步探讨其工作机制、影响范围及潜在安全威胁。现代处理器优化技术与伴生漏洞微架构分布如图2所示。

4.1 侧信道类硬件漏洞

在处理器运行过程中，数据处理时间、处理器能耗及温度等参数会发生数值变化。为了更精确地监测这些变化，处理器集成了专门的组件用于记录相关信息，并通过特定接口向开发者提供访问权限。尽管这些数据可用于优化处理器性能，但同时也引入了安全隐患。侧信道攻击是一种通过分析处理器在执行计算时泄露的物理信息来推测敏感数据的攻击方式。一般来说，处理器中的执行时间、能耗和温度等信息都可以被攻击者用作恢复敏感密钥的侧信道信息，利用这些侧信道信息攻击者可以恢复处理器中的私密信息。

4.1.1 Cache类侧信道漏洞

在第2.1节中，本文介绍了现代处理器的多级缓存架构，其中Cache的引入显著提升了处理器的数据访问速度和整体性能。然而随着性能的提升，Cache也带来了潜在的安全隐患。基于Cache的工作机制，当处理器所需的目标数据存储在Cache中时，数据访问时间较短。相反，当目标数据未存储在Cache中时，处理器需要从主存获取数据，导致访问延迟增加。由于这两种情况下的数据访问时间存在显著差异，攻击者可利用该时间差实施Cache侧信道攻击。

(1)Flush+Reload类 Cache侧信道攻击。

Flush+Reload类 Cache侧信道攻击的核心机制在于攻击者先初始化缓存行状态，当受害者访问特定缓存行后，攻击者通过测量相应内存地址或执行特殊指令的执行时间来推测受害者进程的访问行为，从而恢复敏感数据。Flush+Reload攻击是一种典型且广泛应用的 Cache侧信道攻击方式，能够通过分析 Cache访问时间的差异推测受害者进程的内存访问行为[4]。

(2)Prime+Probe类 Cache侧信道攻击。

除了Flush+Reload类攻击Cache侧信道攻击，Prime+Probe类也是影响广泛的Cache类侧信道攻击之一[7,8]。Prime+Probe通过针对特定的缓存集合，检测其他进程或操作系统对该缓存集合中任意地址的访问情况，来推测受害者进程的内存访问模式。Prime+Probe类攻击无需共享内存，而是通过竞争缓存资源来推测受害者进程的行为，从而适用于更广泛的攻击场景。

(3)不依赖时间测量的 Cache 侧信道攻击。

大多数侧信道攻击均依赖于攻击者频繁执行一系列精确的内存操作，以推测受害者进程是否访问了特定的缓存地址。这些攻击手法高度依赖于高精度计时，因此防御措施通常通过限制对高精度定时

表 2 硬件漏洞信息汇总

漏洞类型	子分类	漏洞名称	披露时间	泄露源部件	主要发现方式	攻击场景
侧信道类	Cache侧信道类	Flush+Reload[4]	2014	LLC	硬件资源共享泄露(Flush+Reload类)	②
		Flush+Flush[6]	2016			⑷⑤
		Prime+Probe[7,8]	2010	L1 Cache,LLC	硬件资源共享泄露(Prime+Probe类)	⑤
		Prime+Abort[10]	2017			⑭
		Evict+Reload[9,33]	2015			⑤
		Reload+Refresh[34]	2020	LLC	硬件资源共享泄露(Cache替换策略)	②⑤
		Xiong等人工作[35]	2020	L1 Cache		⑪⑭
		Cui等人工作[36]	2022	L1 Cache	资源共享(Dirty位状态)	⑭
		Yao等人工作[37]	2018	L1,L2,LLC	资源共享(一致性状态)	②⑧⑭
		EXAM[38]	2025	系统级 Cache	资源共享(Cache占用)	⑦
	TLB侧信道类	Peek-a-Walk[11]	2025	TLB、页表	硬件资源共享泄露(TLB)	⑯⑰
		TLBleed[12]	2018	TLB		②③⑭⑲
		SLAM[39]	2024			①⑱
	能耗类	Collide+Power[16]	2023	Cache和缓冲区	硬件资源共享泄露	⑭
		Platypus[40]	2021	运行平均功率限制接口		(3)(6)①②④
		ThermalBleed[15]	2022	热接口		①
	PMU类	PMU-Spill[13]	2022	PMU	硬件资源共享泄露(PMU)	⑤
		PMU-Leaker[14]	2023			⑤
	争用类	PortSmash[30]	2018	加载端口	硬件资源共享泄露(端口争用)	③⑭
		SMoTherSpectre[31]	2019	TLB		⑦
		MeshUp[41]	2022	网状互连	资源共享(互连路径)	②⑦
		LockedDown[42]	2022	PCIe总线带宽	资源共享(总线争用)	⑦⑧⑭
	预取类	Cachebleed[43]	2017	L1 Cache	硬件资源共享泄露(Cache Bank争用)	②
		Memjam[44]	2019	写后读错误依赖		⑤
		AfterImage[45]	2023	IP步进预取器	硬件资源共享泄露	②⑪⑭
		PrefetchX[46]	2024	扩展预测表预取器		②⑧⑭
	其他类	Controlled Preemption[47]	2025	完全公平调度器	硬件资源共享泄露	②⑤
		SegScope[48]	2024	段寄存器		①⑦⑪
瞬态执行类	乱序执行类	Meltdown[2]	2018	L1/L2 Cache,LLC	显性和隐性陈旧状态利用&硬件资源共享泄露	⑪
		ForeShadow[49]	2018	L1 Cache		(5)(6)④⑦㉒
		ForeShadow-NG[50]	2018			(2)(3)(6)①⑮⑱
		RIDL[51]	2018	LFB		(4)(6)③⑭㉒
		ZombieLoad[52]	2019			(2)(3)(5)(6)①②⑤
	预测执行类	Spectre-V1-PHB[1]	2018	条件分支预测器		⑪
		Spectre-V2-BTB[1,17]	2018	分支目标缓冲区		(1)(2)(3)(6)
		Spectre-V4-CTL[32]	2024	加载/存储单元		⑪⑭
		Spectre-V4-STL[18]	2019			⑨
		Spectre-V5-RSB[19]	2018	返回栈缓冲区		①⑨
		FLOP[20]	2025	加载值预测器		(2)(3)⑷(6)①㉑
		SLAP[21]	2025	加载地址预测器		⑦⑨⑪
		GadgetSpinner[53]	2024	循环流检测器		⑤⑦
故障注入类	指令缺陷类	(M)WAIT[54]	2023	umwait,umonitor	硬件资源共享泄露	②⑧
		AVX-Timing[55]	2023	Vmaskmov指令		(3)(6)⑤
		Adversarial Prefetch[56]	2022	PrefetchW		(6)②⑤
	高频率类	CLKScrew[22]	2017	DVFS	低功耗故障注入测试(超频)	(6)㉔
	低功耗类	VoltJockey[23]	2021	DVFS	低功耗故障注入测试(欠压)	(6)②④
		Voltpwn[24]	2020	DVFS,MSR		②
		Plunfervolt[25]	2020	MSR		⑩
	RowHarmmer类	ProbeHammer[57]	2025	DRAM	硬件资源共享泄露(DRAM)	⑪
		Go Go Gadget Hammer[58]	2024			⑪

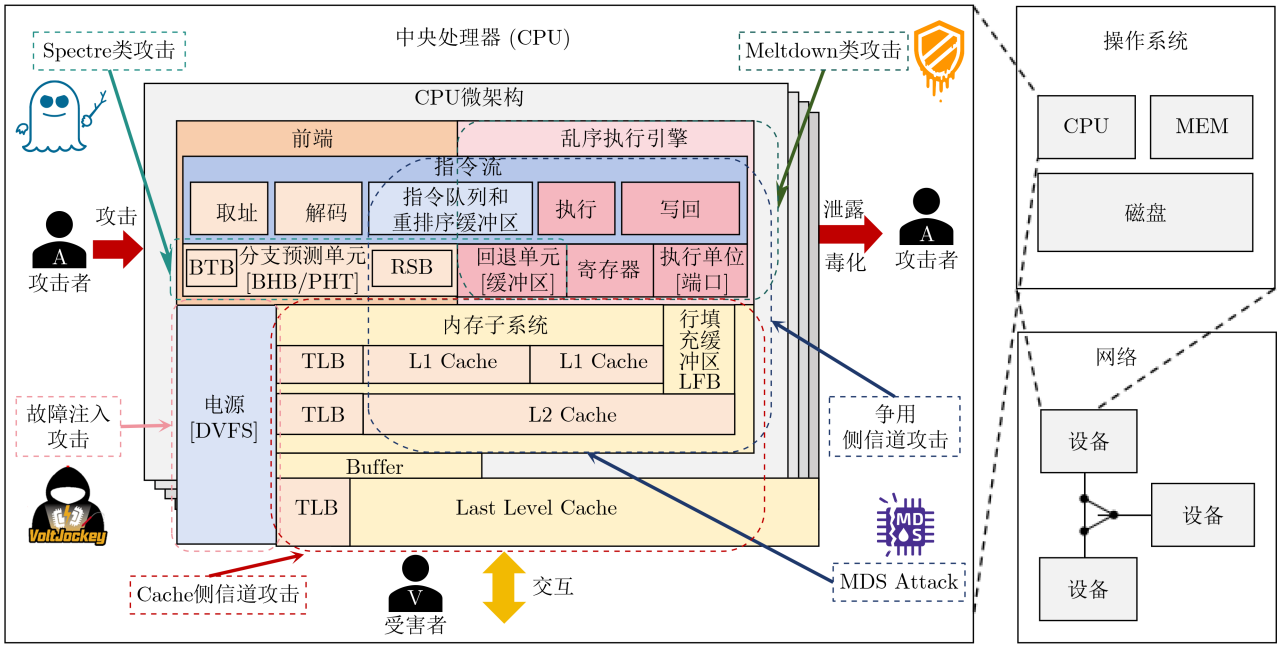


图 2 现代处理器优化技术与伴生漏洞微架构分布图

器的访问或引入时间噪声来干扰基于时序分析的攻击模型。然而，Disselkoen等人[10]结合 Intel的硬件事务内存技术提出了 Prime+Abort攻击，能够不依赖于传统时序差异分析实现数据恢复。

(4)其他 Cache侧信道攻击。

除了通过判断数据是否存储于 Cache中的状态来恢复信息外，Cache中还蕴含大量可被攻击者利用的侧信道信息，为 Cache侧信道攻击提供了多样化路径。例如，不同缓存替换策略在 Cache填充过程中的状态更新可被用作泄露载体[34,35]，缓存行的 Dirty 位状态可揭示写操作行为[36]，而 Cache一致性协议所维护的状态亦可能暴露处理器核心之间的通信模式[37]。此外，缓存占用率的变化也有可能间接反映受害者程序的访问特征，进而构成安全隐患[38]。

4.1.2 TLB类侧信道漏洞

作为处理器中关键的地址转换缓存结构，TLB具有高命中率和低延迟等性能优势，在多核处理器架构中，TLB状态往往被多个线程或进程共享。这种共享特性使攻击者可以通过分析页级别的访问模式而获取敏感信息，从而构成新的侧信道攻击面。

(1)基于页表访问状态实现的 TLB侧信道。

在现代处理器架构中，同一个物理核心上的超线程会共享TLB资源，以协同翻译其各自的代码和数据地址访问请求。基于此，Gras等人[12]提出了TLBleed漏洞，TLBleed是一种新颖的基于TLB的侧信道攻击方式。攻击者可借助TLBleed来获取TLB中的页表访问状态，对运行在同核心另一线程上的受害者进程进行监测与信息恢复。

(2)基于页表遍历过程中缓存行为实现的 TLB侧信道。

Peek-a-Walk是一种新型TLB侧信道，能够利用页表遍历过程中固有的缓存行为来恢复敏感信息[11]。Peek-a-Walk能通过诱导对秘密地址的解引用操作，将秘密的虚拟地址高位编码进页表缓存状态。随着对TLB的深入研究，Hertogh等人[39]提出了SLAM攻击。SLAM结合了L1 TLB层级的Evict+Reload操作，并引入规范性检查绕过策略，以提升攻击稳定性与适用性。

4.1.3 能耗类侧信道漏洞

处理器功耗作为一种可被观测的物理信号，蕴含了丰富的微架构执行信息。具体而言，处理器在处理不同类型的数据或执行不同指令路径时，其功耗会产生可测量的变化。攻击者可借助这些功耗波动，通过建立数据与功耗之间的统计相关性，推测并恢复加密密钥等敏感信息。

(1)基于温度的能耗侧信道漏洞。

温度作为一种低速、可感知但难以屏蔽的物理特征可以被用来构建侧信道以泄露信息[15]。现基于此，Taehun等人[15]提出了 ThermalBleed，这是一种基于温度的侧信道攻击。攻击者可利用 ThermalBleed来观察相邻核心的温度变化趋势，来恢复未在不同核心间共享的私密数据。此外，Masti等人[40]提出了一种温度侧信道攻击，该攻击可在协同执行的应用间建立通信通道，进而绕过传统的时间与空间隔离机制。

(2)基于功耗的能耗侧信道漏洞。

在处理器执行过程中，指令路径和所访问数据

的差异会引起其功耗特性的变化，这种功耗差异可被间接感知并加以利用。基于此，Andreas等人[16]提出了 Collide+Power漏洞。Collide+Power利用受害者程序在特定数据或路径下所引起的功耗波动，通过构造与其共享微架构状态的攻击路径来泄露敏感信息。Lipp等人[40]提出了一种名为 Platypus的新型功耗侧信道攻击，其利用 Intel处理器的运行平均功耗限制接口构建出一个低分辨率但可控的功耗侧信道，能够推断出应用程序的控制流与敏感数据内容。

4.1.4 PMU类侧信道漏洞

PMU作为现代处理器中关键的硬件模块，主要用于记录系统运行过程中产生的各类事件。但攻击者可借助PMU提供的事件计数器，分析目标程序在微架构层面上的行为差异，间接泄露敏感信息。PMU-Spill与PMU-Leaker是两种典型的PMU类侧信道漏洞，分别揭示了PMU在微架构层面信息泄露中的不同利用方式[13,14]。

4.1.5 基于争用的侧信道漏洞

基于争用的侧信道漏洞是现代处理器面临的重要安全威胁之一。攻击者利用执行单元争用引起的时延变化可泄露敏感信息[30,31]。现代处理器架构中，各计算单元可并发运行，导致资源共享程度显著增强。针对这一问题，可以实现基于互联路径争用的侧信道攻击[41]。当处理器和图形处理器共同占用带宽或容量受限的资源时，会发生资源竞争。基于此，可以实现基于总线争用的侧信道攻击，从而间接推测其行为信息[42]。此外，还可以利用 Cache Bank的争用实来现侧信道攻击，以推断出受害者的内存访问行为[43,44]。

4.1.6 基于预取的侧信道漏洞

预取指令未进行访问权限校验这一设计缺陷，使得无特权攻击者能够访问本不应获取的地址翻译信息。Chen等人[46]提出了一种利用扩展预测表预取器的跨核心攻击——PrefetchX。该攻击无需依赖共享内存，即可泄露用户进程中的敏感信息。

4.1.7 其他侧信道漏洞

为突破传统侧信道方法中时间精度低所带来的局限，Zhu等人[47]提出了一种称为 Controlled Preemption的进程抢占驱动型侧信道攻击。通过精细控制抢占间隔，攻击者可以对受害者执行路径或微架构状态进行高时间分辨率的观测。

中断是操作系统内核用于调度进程的重要硬件资源，其行为高度依赖系统活动特征，因此可作为侧信道攻击的切入点。Zhang等人[48]提出了一种名为 SegScope的新型侧信道攻击方法，该方法能够在不依赖任何计时器的前提下，实现对中断行为的细粒度观测。

4.1.8 侧信道漏洞评估体系

随着硬件侧信道攻击技术的不断演进，公开披露的攻击方式日益多样化，这对评估各类侧信道攻击的能力和影响提出了更高要求。如何系统、客观且可重复地量化侧信道在攻击性能、安全性威胁等方面的关键特征已成为当前硬件安全研究中的一个重要课题[28]。为此，本文在梳理现有文献的基础上，形成了用于评估侧信道攻击能力的八项核心指标[28,29]。

空间精度与拓扑范围。空间精度与拓扑范围是评估侧信道攻击能力与影响范围的两个核心维度，分别体现了攻击的精细化程度以及攻击者与受害者之间的空间关系。其中，空间精度用于衡量攻击者在空间维度上恢复私密信息的能力。另一方面，拓扑范围描述了攻击者能够实施攻击并成功恢复数据的空间范围。这两个指标在侧信道攻击的风险评估、攻击路径建模以及防御策略设计中具有重要意义。

攻击时间与盲点长度。在侧信道攻击的评估体系中，攻击时间与盲点长度是衡量攻击效率与可操作性的两个关键指标。攻击时间定义为从攻击流程启动到成功恢复私密信息所经历的总时间，它不仅直接影响攻击的实时性与隐蔽性，还决定了攻击所能实现的数据吞吐量。盲点长度则指攻击过程中攻击者无法感知或控制受害者执行行为的时间窗口，盲点长度的存在显著降低了攻击的成功概率与稳定性。

信道容量与抗噪能力。在侧信道攻击的评估体系中，信道容量与抗噪能力是衡量攻击信息传输质量与鲁棒性的重要指标。信道容量用于衡量攻击者在单位时间内能够通过侧信道获取的有效私密信息量，反映了攻击通道的最大信息传输能力。抗噪能力表征在存在系统干扰条件下，攻击通道维持稳定传输并成功恢复信息的能力。攻击者通常需在设计攻击方案时平衡这两个维度，以实现更高效的数据泄露。

可探测性。在侧信道攻击的评估体系中，可探测性是衡量攻击是否易被系统或安全工具识别的关键指标。该属性不仅体现了攻击的隐蔽性，还直接影响其在真实系统环境中的可部署性。在实际场景中，高可探测性的攻击手段虽然可能具有更强的攻击效果，但更适用于受控环境下的安全性验证与研究测试。

侧信道信息差值。在侧信道攻击的评估体系中，侧信道信息差值决定了攻击者能否准确识别并区分受害者行为所引发的微架构状态变化，进而影响攻击的成功率与恢复精度。侧信道信息差值是指

在访问硬件资源过程中，由于数据是否命中而引起的访问延迟差异，或者是硬件部件中产生的其他数值变化。这种差值越大，攻击者越容易做出判断，从而提高私密数据的恢复能力。

4.2 瞬态执行类硬件漏洞

当处理器正确预测或乱序执行指令时，能够加速指令执行。但是当预测或乱序执行错误时，处理器会在架构层面回滚至执行前的状态，并重新执行正确的指令。然而，在微架构层面，这些被暂时执行的指令可能已经影响了微架构的状态，而这些瞬态执行痕迹不会被处理器主动清除，从而可能被攻击者利用。在此期间，攻击者可以通过观察微架构状态的残留信息实施攻击。

4.2.1 乱序执行类硬件漏洞

在第 2.2节中提到，乱序执行机制通过解析指令并根据解码后的微指令之间的数据依赖关系优化执行顺序来提高处理效率。当指令提交检查失败时，处理器会回滚执行状态至该指令执行之前，并重新执行正确的指令。由于乱序执行的特性，未与回滚指令存在数据依赖的后续指令可能已经提前执行，并在微架构层面产生了可观察的执行痕迹。这些残留的微架构状态不会被回滚机制清除，从而可能被攻击者利用，借助瞬态执行攻击推测敏感数据。乱序执行类硬件漏洞攻击流程如图3(a)所示。

(1)跨内存隔离机制的乱序执行类漏洞。

Meltdown是由 Lipp等人[2]于 2018年公开的一种瞬态执行攻击技术，该漏洞揭示了现代处理器在处理异常控制流程时对内存隔离机制的破坏性影响。Meltdown允许非特权攻击者绕过用户态与内核态之间的内存访问边界，进而在用户空间中读取本应受保护的内核空间中的敏感信息。尽管该攻击最初主要影响Intel架构处理器，但后续研究发现部分 ARM架构处理器也同样存在该漏洞。

(2)跨可信执行环境的乱序执行类漏洞。

随着对 Meltdown漏洞研究的不断深入，Bulck等人[49]于同年提出了 ForeShadow漏洞。ForeShadow是一种针对 Intel处理器架构的攻击，其核心目标不仅限于泄露用户空间中的敏感数据，还可突破更为严格的隔离边界。2019年由 Schwarz等人[52]提出的 ZombieLoad 漏洞和 Schaik等人[51]提出的RIDL漏洞也可以利用行填充缓冲区(Line Fill Buffer, LFB)实现对内核空间乃至虚拟化环境中私密信息的恢复。

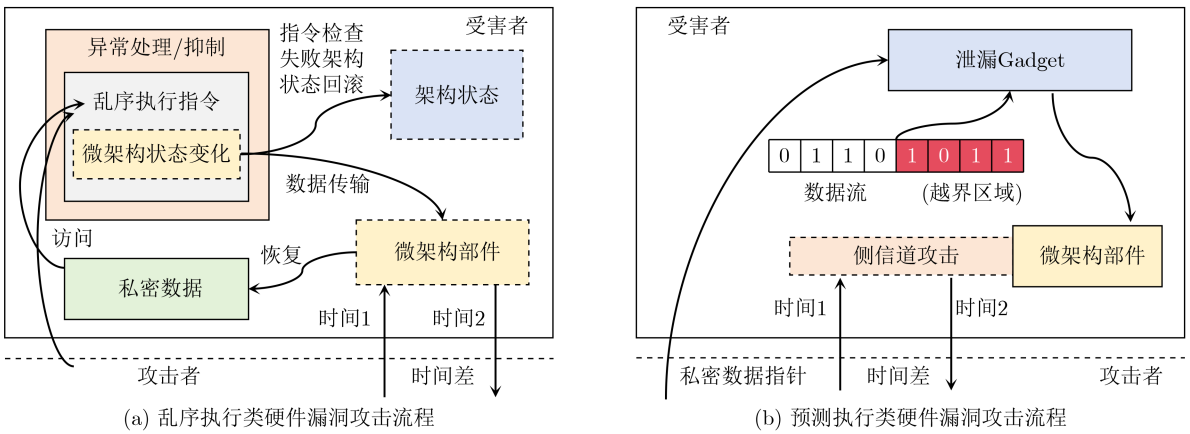
4.2.2 预测执行类硬件漏洞

如第 2.3节所述，当处理器在指令执行过程中面临导致流水线停滞的情况时，通常会采用预测执行机制来预测后续指令路径，并提前执行相应指令来减少指令等待时间。尽管预测执行在提高处理器性能方面发挥了显著作用，但也引入了严重的安全隐患。预测执行类硬件漏洞攻击流程如图3(b)所示。

(1)基于控制流预测的预测执行类漏洞。

Spectre漏洞是一类基于现代处理器预测执行机制缺陷的微架构级安全漏洞，攻击者可以利用Spectre在指令尚未被正式提交的阶段访问本应受限的敏感数据，并通过侧信道技术将其泄露[1]，图4展示了 Spectre的攻击流程。因其高度的普适性和灵活性，迅速发展出多个变种，持续对现代计算系统构成威胁[1,18,19,32]。Spectre几乎影响所有主流处理器架构，包括 Intel, AMD和 ARM，且具备跨进程、跨线程乃至跨虚拟机的信息窃取能力，打破了传统操作系统与虚拟化环境中所依赖的边界隔离假设。

在现代处理器中除了分支指令的预测执行机制存在安全风险外，循环结构的预测执行同样可能引发严重的信息泄露问题。2024年Chen等人[53]提出了GadgetSpinner，其利用循环流检测器在循环加



(a) 乱序执行类硬件漏洞攻击流程　　(b) 预测执行类硬件漏洞攻击流程

图 3 乱序执行类与预测执行类硬件漏洞攻击流程

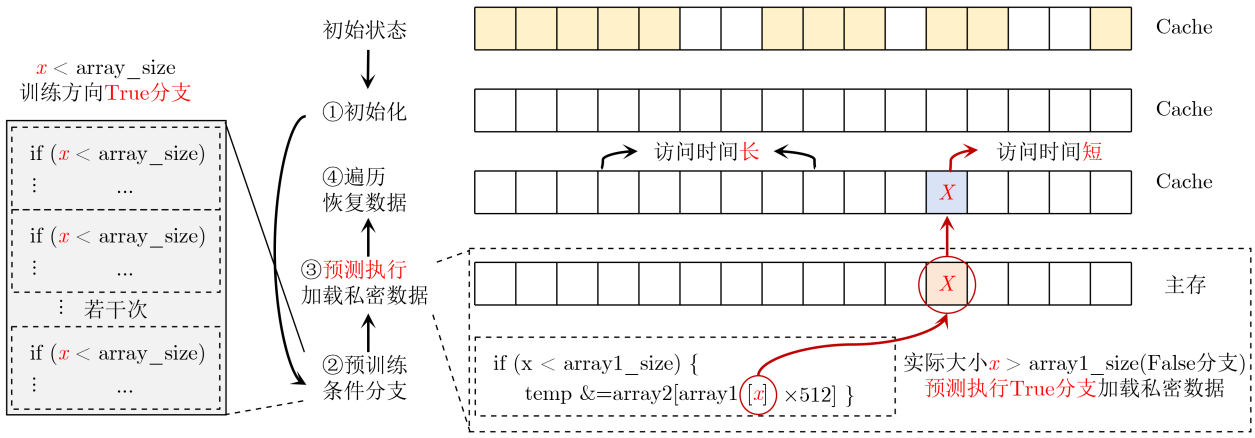


图 4 Spectre攻击流程

速过程中的预测执行行为，诱导处理器在正常控制流之外执行特定指令，从而访问并泄露私密数据。

(2)基于数据流预测的预测执行类漏洞。

为提升处理器性能，现代微架构中广泛引入了值预测机制，用于减少数据依赖引发的执行延迟。然而，Deng等人验证了值预测器存在信息泄露风险，提出了两种攻击模型。2025年，Kim等人[20]提出了FLOP攻击，揭示了Apple Silicon架构中加载值预测器在安全性方面的缺陷。

(3)基于地址预测的预测执行类漏洞。

随着微架构预测技术的不断发展，除了传统的基于控制流和数据流的预测执行机制带来的安全隐患外，基于地址的预测执行也逐渐暴露出安全风险。2025年，Kim等人[21]针对 Apple Silicon架构提出了 SLAP攻击，揭示了加载地址预测器在预测执行过程中的潜在安全漏洞。

4.2.3 指令缺陷类硬件漏洞

指令缺陷类硬件漏洞是指由于处理器在执行特定指令时，因其设计逻辑、实现细节或微架构优化策略存在缺陷，进而引发信息泄露。这类漏洞往往集中体现在特定指令或指令序列在边界条件下的异常行为或未定义状态处理不当。而且此类漏洞具有较强的平台依赖性，在不同的平台上可能会有完全不同的效果。

(1)Intel架构中的指令缺陷类漏洞。

Intel处理器的高级矢量扩展指令集中的部分指令在实现细节上存在安全隐患，可能被攻击者利用以发起侧信道攻击[55]。攻击者可据此实现AVX Timing攻击，通过对指令执行时间的精细测量，推断出系统的地址空间布局信息。此外，利用umonitor和umwait两条指令的特性，攻击者可以实现(M)Wait攻击，能够在无需依赖时间测量手段的情况下将微架构级状态变化转换为可由架构级逻辑直接观察的结果[54]。

(2)AMD架构中的指令缺陷类漏洞。

2024年，Zhang等人[4]针对 AMD提出的加密虚拟化技术，提出了一种名为 CacheWarp的新型软件层级故障注入攻击。CacheWarp利用了处理器在特定异常处理中可能出现的缓存状态回滚不一致问题，实现对虚拟机内部加密数据的篡改与回退。其核心在于通过控制虚拟机在特定时间点发生中断，攻击者能够使已经被修改的缓存行失效，而这些缓存内容并不会同步写回主存，从而恢复为之前的陈旧值。

(3)ARM架构中的指令缺陷类漏洞。

Gruss等人设计了一种利用 ARMv8-A微架构特性的攻击方法，攻击者可以使用无特权的 PRFM PLDL1KEEP指令从用户空间预取目标地址数据至L1 Cache，并结合无特权的 DC CIVAC指令对缓存行执行清除和刷新操作。这些指令的组合操作可以让攻击者观测 Cache状态，从而构建侧信道攻击。

(4)RISC-V架构中的指令缺陷类漏洞。

RISC-V作为一种开放且可扩展的指令集架构，其灵活性为多样化硬件实现提供了基础。然而，这种灵活性也带来了实现层面的安全隐患。针对这一问题，Thomas等人提出了名为 GhostWrite的漏洞，该漏洞允许攻击者将任意字节值写入特定物理地址，实现对物理内存的任意读写。此外，Gerlach等人提出了3种基于 RISC-V特定架构的攻击方法。通过运用这3种攻击，攻击者能够从处理器中恢复私密数据。

4.3 故障注入类硬件漏洞

现代硬件设备在工作过程中对频率、电压等物理参数十分敏感，攻击者可以通过外部手段操控这些参数实施故障注入攻击，诱导硬件系统在关键操作阶段发生异常行为。这些异常行为可能会泄露敏感信息。所谓故障注入类硬件漏洞，是指通过主动干扰硬件正常运行状态，使其产生暂态或永久性的

异常，从而触发系统在微架构层或逻辑层的非预期反应，进而实现攻击目的。

4.3.1 高频率故障注入

为满足高性能计算与能源效率的双重需求，现代处理器广泛引入了软硬件协同的能量管理机制。尽管DVFS在提升系统能效方面具有显著优势，但其同时暴露了潜在的安全风险。Tang等人[22]对 DVFS能源管理机制进行了系统性的安全分析并提出CLKScrew漏洞，揭示了现代能耗管理机制中存在的安全缺陷。CLKScrew攻击是通过内核驱动程序实现的，攻击流程如图5所示。

4.3.2 低电压故障注入

由于超频通常会使处理器运行于异常高的频率区间，该行为在系统层面相对容易被检测与防御。基于此，Qiu等人[23]提出了 VoltJockey漏洞，将关注点转向了 DVFS中的电压控制手段。与 CLKscrew相比，VoltJockey无需依赖处理器频率的调节机制。其次，VoltJockey的整个攻击过程均在DVFS的正常功能范围内完成，使得 VoltJockey更难防御。在此基础上，2020年 Kenjar等人[24]和Murdock等人[25]分别提出了 V0LTpwn和 Plundervolt攻击，进一步验证了现代处理器中电压控制机制的安全性缺陷。

4.3.3 RowHammer攻击

RowHammer漏洞是现代内存系统中一种典型的硬件级故障注入漏洞，其攻击原理基于DRAM硬件结构中的电荷干扰效应。基于此，Dio等人[57]提出了Preload+Time攻击，并设计了一种无需数据依赖的ProbeHammer攻击，ProbeHammer能够在增强稳定性的同时仍保留对目标行的攻击能力。此外，Tobah等人[58]开发了一种基于嵌套指针解引用的Rowhammer工具，成功实现了对受害者地址空间的任意读取。

5 实验评估

在第 4节中，本文系统地归纳了3类主要的硬件安全漏洞。为进一步验证相关攻击的可行性与影响，本文选取了其中具有代表性的漏洞开展实验研究。基于实验结果，本文对各类攻击的可行性、影响范围及其平台相关性进行了深入分析与总结。 表3展示了代表性处理器硬件漏洞的验证结果。

5.1 实验设置

本研究在6种不同型号的处理器平台上开展了系统性实验，覆盖了多家主流处理器供应商及其在多种操作系统与内核版本下的运行环境。实验主要聚焦于当前具有代表性的硬件安全漏洞类型，包括Meltdown系列攻击、Spectre系列攻击、Fallout, ZombieLoad, Foreshadow、侧信道攻击以及其他类型的已公开或变种攻击手段。通过对各类漏洞在不同微架构和软件栈下的可行性与攻击效果进行实证分析，为处理器漏洞跨平台表现的评估及其防御机制设计提供了重要实验依据。

5.2 实验结果

根据表3所示实验数据，尽管两款 i7-6700处理器内核版本存在差异，但其在漏洞攻击实验中的表现保持一致。此外，i7-7700处理器与 i7-6700在实验结果上也高度相似。在这两类处理器平台上，成功实施的攻击包括 Meltdown系列漏洞、Spectre-v1/v2/STL, Fallout, ZombieLoad, Foreshadow、侧信道攻击、CrossTalk以及 RIDL等多种类型，

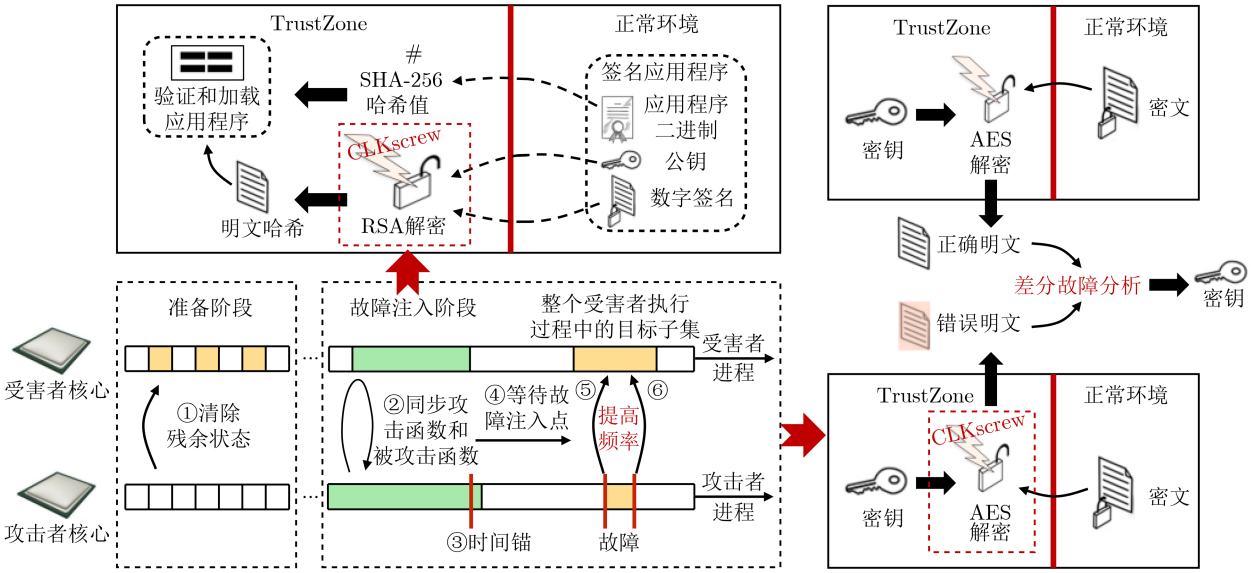


图 5 CLKScrew攻击流程

表 3 代表性处理器硬件漏洞的验证结果

实验处理器参数							
处理器型号	供应商	操作系统	内核	处理器型号	供应商	操作系统	内核
i7-6700-1	Intel	Ubuntu 16.04.1	4.15.0-142	i7-6700-2	Intel	Ubuntu 18.04.1	5.4.0-150
i7-7700	Intel	Ubuntu 18.04.1	5.4.0-150	i5-7300u	Intel	Ubuntu 18.04.1	5.4.0-150
i9-13900	Intel	Ubuntu 20.04.1	5.15.0-86	Ryzen 5 5600G	AMD	Ubuntu 20.04.1	5.15.0-76
实验结果							
漏洞	变种	i7-6700-1	i7-6700-2	i7-7700	i9-13900	i5-7300u	Ryzen 5 5600G
Meltdown[2]	Meltdown-US	√	√	√	×	×	×
	Meltdown-RW	√	√	√	√	√	√
	Meltdown-P	√	√	√	×	×	×
	Meltdown-DE	√	√	√	√	√	×
	Meltdown-Exploit	√	√	√	×	×	×
Spectre	Spectre-V1[1]	√	√	√	√	√	√
	Spectre-V2[1]	√	√	√	×	√	√
	Spectre-CTL[32]	×	×	×	×	×	√
	Spectre-STL[18]	√	√	√	√	√	√
	Spectre-V5[19]	×	×	×	×	×	×
Fallout[59]	Write Transient Forwarding	√	√	√	×	√	×
	Data Bounce	√	√	√	×	√	×
	Fetch Bounce	√	√	√	×	√	×
	Spec Bounce	√	√	√	×	√	×
ZombieLoad[52]	ZombieLoad-V1	√	√	√	×	×	×
	ZombieLoad-V2	√	√	√	×	×	×
	ZombieLoad-V4	√	√	√	×	×	×
ForeShadow[49,50]	Foreshadow	√	√	√	×	×	×
	Foreshadow-NG(OS)	×	×	×	×	×	×
	Foreshadow-NG (VMM)	×	×	×	×	×	×
侧信道攻击	Flush+Reload[4]	√	√	√	√	√	√
	Prime+Probe[7,8]	√	√	√	√	√	√
	PMU-Spill[13]	√	√	√	×	×	×
	PMU-Leaker[14]	√	√	√	√	√	√
其他	RIDL(MLPDS)[51]	√	√	√	×	×	×
	RIDL(MDSUM)[51]	√	√	√	×	×	×
	Zenbleed[60]	×	×	×	×	×	×
	CrossTalk[61,62]	√	√	√	×	√	×
	Inception[63]	×	×	×	×	×	√

√表示该漏洞可在处理器上成功实施，×表示该漏洞无法在处理器上实施。

表明这些处理器在多个微架构层面存在类似的漏洞利用路径。

与之对比，i9-13900处理器在实验中可成功实现的攻击类型明显减少，仅包括 Meltdown-RW, Meltdown-DE, Spectre-v1, Spectre-STL, Flush+Reload, Prime+Probe以及PMU-Leaker，反映出较新的微架构在部分已知漏洞方面具有更强的防护能力。对于 i5-7300U处理器，成功攻击的范围相对更广，包括 Meltdown-RW, Meltdown-DE, Spectre-v1/v2, Fallout, Flush+Reload, Prime+Probe, PMU-Leaker和 CrossTalk，说明其在多个安全机制上的防护能力较弱。

在AMD平台方面，Ryzen 5 5600G的实验结果表明，攻击者可成功实施的漏洞类型包括Meltdown-RW, Spectre-v1/v2/CTL/STL, Flush+Reload, Prime+Probe, PMU-Leaker以及Inception。该结

果进一步印证了某些漏洞(如Spectre-CTL和Inception)对 AMD架构具有特定攻击适应性，同时也突出了各厂商微架构差异对漏洞实现效果的显著影响。

5.3 实验结果分析

通过对实验汇总结果的分析，可以观察到若干具有代表性的现象。首先，部分漏洞呈现出显著的处理器厂商相关性。通过进一步审阅攻击实现代码与文档发现，造成此现象的主要原因在于上述漏洞各自依赖于特定厂商所实现的微架构组件，当目标处理器不具备相应组件或其实现机制存在差异时，攻击路径便无法建立，从而导致攻击失败。例如，Fallout, ZombieLoad, Foreshadow及 RIDL等漏洞仅可在 Intel处理器上成功实现，无法在 AMD处理器上实现，因为这些攻击依赖于 Intel处理器特有的微架构部件。

其次，实验结果还揭示了部分漏洞之间存在结构性关联。例如，ZombieLoad与 RIDL攻击在测试中呈现出完全相同的攻击效果，这一现象可归因于两者均依赖对处理器中的 LFB进行数据泄露。同样地，Meltdown-US, Foreshadow-type及PMU-Spill漏洞在测试中的表现亦高度一致。通过分析可知，Foreshadow-type与 PMU-Spill攻击均在以Meltdown-US为前提的攻击环境中完成，一旦处理器可被成功利用以实施Meltdown-US攻击，后续的Foreshadow-type与PMU-Spill攻击同样具备可行性。

此外，实验结果表明，同一供应商的处理器在不同代际之间，甚至在同一代际内的不同型号之间，对于特定漏洞的可利用性存在显著差异。以 Meltdown-US, Meltdown-P和 Meltdown-Exploit为例，在 Intel的第6代、第7代和第13代处理器中，其攻击效果呈现出不同表现。

硬件漏洞的跨平台差异不仅体现在宏观上的效果，其根本原因在于不同微架构设计对缓存行为、预测执行逻辑、异常处理机制与指令回滚策略的具体实现存在差异，这些底层实现细节直接决定了攻击路径的可构建性与漏洞的触发条件。因此，面向未来的漏洞评估与防御机制设计，必须综合考虑处理器微架构的异构性，构建具有通用性与适应性的分析与防护框架，以提升应对复杂硬件安全威胁的能力与鲁棒性。

6 缓解方案

随着处理器微架构设计的日益复杂，硬件层面的安全问题日益凸显。如何在不显著牺牲系统性能的前提下实现对硬件攻击的防护，已成为安全领域的核心研究方向之一。鉴于此，本节将从硬件漏洞攻击过程的三个关键步骤出发，系统归纳并分析当前已有的缓解策略。

6.1 针对数据准备步骤的缓解方案

在攻击的第1步数据准备时，大多数瞬态执行漏洞利用处理器的性能优化机制，以实现对攻击环境的初始化和私密数据的非法提取。为此，最直接且有效的缓解策略是禁用相关的微架构优化功能，然而该方法虽然在安全性上具有显著成效，却也会带来严重的性能退化。因此，针对本步骤的缓解思路应聚焦于在不禁用优化机制的基础上，对其内部实现逻辑进行细粒度的安全加固。

6.2 针对数据传输步骤的缓解方案

在第2步数据传输过程中，瞬态执行漏洞的核心特征在于通过构建隐蔽信道，将第1步加载的敏感数据以微妙方式编码，并在后续阶段中被攻击者提取利用。针对该步骤的防御策略应侧重于阻止敏感数据进入可被侧信道探测的微架构部件。为此，一种有效的缓解方案是引入“影子缓存”机制，即在权限验证完成之前，临时将加载的数据存储于安全的硬件缓冲区。另一种思路则是通过延迟微架构状态的更新，将数据写入缓存的时机后移至权限校验之后，从根本上阻断隐蔽信道的构建。因此，本过程的防御目标应聚焦于切断私密数据在微架构层级的传输路径，有效抑制隐蔽信道的形成。

6.3 针对数据恢复步骤的缓解方案

在最后一步数据恢复时，瞬态执行攻击通常依赖各种侧信道技术对第2步经隐蔽信道泄露的敏感信息进行还原。攻击者通过对侧信道信息进行精细测量与分析，可以间接重建原始的私密数据。因此，针对此步骤的防御策略主要聚焦于削弱攻击者对侧信道信号的观测精度。常见的缓解手段包括引入操作的时间随机化、注入系统性噪声，以及降低计时器等微架构计量工具的时间分辨率，以模糊潜在的数据信息特征。

总体而言，瞬态执行漏洞在每个攻击步骤均展现出独特的攻击特性，相应的防御策略也应围绕这些特点展开，形成分阶段、针对性强的安全机制。然而，仅依赖对单一漏洞的被动防护难以从根本上解决问题。面向未来的处理器设计应将安全性作为基础目标之一，在体系结构层面主动构建对瞬态攻击的天然抵抗能力，提升整体安全性与运行效率的协同水平。这一方向将成为处理器安全研究的重要发展趋势。

7 讨论与展望

本节将结合当前发展趋势，面向未来硬件漏洞

的攻击演化路径与防御体系构建，提出具有前瞻性的研究方向，为后续研究提供理论支撑与技术参考。

7.1 攻击方面

现代处理器大致可分为两类：一类是以 Intel, AMD为代表的主流商业处理器，其底层微架构设计和实现细节基本处于闭源状态。另一类则是以RISC-V为代表的开源处理器，其架构规范完全公开。在对闭源商业处理器的研究中，主要依赖逆向工程技术来发现潜在的漏洞并进行攻击验证。而在开源的 RISC-V处理器平台上，研究工作更多集中在利用模糊测试与形式化验证等手段探索潜在的硬件漏洞与安全缺陷。

7.1.1 逆向工程研究

为应对处理器内部设计未知问题，研究人员广泛采用逆向工程技术，通过精心设计的测试集、性能事件分析以及行为建模等方法，从处理器的外部可观测行为中推导其内部微架构结构与执行机制，从而识别潜在的信息泄露路径与安全风险。因此，逆向工程技术已经成为硬件漏洞研究中不可或缺的基础工具。

7.1.2 RISC-V处理器安全研究

面向未来，RISC-V平台上的硬件漏洞研究仍有诸多发展空间。首先，可构建标准化的漏洞测试基准平台，提升漏洞发现与验证的系统性与效率。其次，结合模糊测试与形式化验证等技术，实现对处理器安全属性的多维度建模与系统性检测，能够为未来高安全性处理器的设计提供理论与技术支撑。

7.2 防御方面

在硬件漏洞防御研究中，未来的发展趋势正逐步朝向跨层协同、高性能保障、可验证设计及智能化防御等多个方向深入拓展。因此，未来的研究应聚焦于构建统一的形式化建模与验证框架。同时，应开发用于验证缓存一致性、预测执行行为等微架构特性的分析工具链。此外，随着人工智能技术的迅速发展，将其引入硬件漏洞防御研究亦成为一项重要方向。

8 结论

本文对处理器中存在的硬件漏洞进行了系统性梳理与深入分析。本文从漏洞的发现、实现与利用3个维度出发，归纳了4种主要的漏洞发现方法，并在现有工作基础上总结形成了一种硬件漏洞3步攻击模型和全新的攻击场景模型。在此基础上，根据攻击行为的特征，本文将现有漏洞划分为侧信道类、瞬态执行类与故障注入类3大类，并在每个类别中进行了更细致的划分。此外，本文参考 Cache侧信道评估指标针对侧信道类漏洞形成了8项评估指标，用于从攻击效果、隐蔽性、适用性等多个维度综合评估其潜在威胁。为进一步验证各类攻击的可行性与影响范围，本文选取了部分具有代表性的漏洞，在多款处理器平台上进行了实验验证，分析总结了其在实际系统中的表现差异与行为特征。此外，本文还从硬件漏洞攻击过程中的三个关键阶段出发，系统归纳当前已有的缓解思路。最后，本文探讨了硬件漏洞研究的未来发展方向，包括处理器逆向工程技术的发展趋势以及基于 RISC-V处理器的攻击探索。同时，本文还分析了面向未来的硬件安全缓解方案的设计理念与研究路径。

参 考 文 献

[1] KOCHER P, HORN J, FOGH A, *et al*. Spectre attacks: Exploiting speculative execution[C]. Proceedings of the 2019 IEEE Symposium on Security and Privacy, San Francisco, USA, 2019: 1–19. doi: 10.1109/SP.2019.00002.

[2] LIPP M, SCHWARZ M, GRUSS D, *et al*. Meltdown: Reading kernel memory from user space[C]. Proceedings of the 27th USENIX Conference on Security Symposium, Baltimore, USA, 2018: 973–990.

[3] 王泉成, 唐明. 微架构安全漏洞攻击技术综述[J]. 密码学报, 2024, 11(6): 1199–1232. doi: 10.13868/j.cnki.jcr.000730.
WANG Quancheng and TANG Ming. Survey of attack techniques for microarchitecture security vulnerabilities[J]. *Journal of Cryptologic Research*, 2024, 11(6): 1199–1232. doi: 10.13868/j.cnki.jcr.000730.

[4] ZHANG Ruiyi, GERLACH L, WEBER D, *et al*. CacheWarp: Software-based fault injection using selective state reset[C]. Proceedings of the 33rd USENIX Conference on Security Symposium, Philadelphia, USA, 2024: 1135–1151.

[5] YAROM Y and FALKNER K. FLUSH+RELOAD: A high resolution, low noise, L3 cache side-channel attack[C]. Proceedings of the 23rd USENIX Conference on Security Symposium, San Diego, USA, 2014: 719–732.

[6] GRUSS D, MAURICE C, WAGNER K, *et al*. Flush+Flush: A fast and stealthy cache attack[C]. Proceedings of the 13th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, San Sebastián, Spain, 2016: 279–299. doi: 10.1007/978-3-319-40667-1_14.

[7] TROMER E, OSVIK D A, and SHAMIR A. Efficient cache attacks on AES, and countermeasures[J]. *Journal of Cryptology*, 2010, 23(1): 37–71. doi: 10.1007/s00145-009-9049-y.

[8] LIU Fangfei, YAROM Y, GE Qian, *et al*. Last-level cache side-channel attacks are practical[C]. Proceedings of the 2015 IEEE Symposium on Security and Privacy, San Jose,

USA, 2015: 605–622. doi: 10.1109/SP.2015.43.

[9] LIPP M, GRUSS D, SPREITZER R, *et al*. ARMageddon: Cache attacks on mobile devices[C]. Proceedings of the 25th USENIX Conference on Security Symposium, Austin, USA, 2016: 549–564. **(查阅网上资料, 请确认修改是否正确)**.

[10] DISSELKOEN C, KOHLBRENNER D, PORTER L, *et al*. Prime+abort: A timer-free high-precision L3 cache attack using intel TSX[C]. Proceedings of the 26th USENIX Conference on Security Symposium, Vancouver, Canada, 2017: 51–67.

[11] WANG Alan, CHEN Boru, WANG Yingchen, *et al*. Peek-a-walk: Leaking secrets via page walk side channels[C]. Proceedings of the 2025 IEEE Symposium on Security and Privacy, San Francisco, USA, 2025: 3534–3548. doi: 10.1109/SP61157.2025.00023.

[12] GRAS B, RAZAVI K, BOS H, *et al*. Translation leak-aside buffer: Defeating cache side-channel protections with TLB attacks[C]. Proceedings of the 27th USENIX Conference on Security Symposium, Baltimore, USA, 2018: 955–972.

[13] QIU Pengfei, GAO Qiang, LIU Chang, *et al*. PMU-spill: A new side channel for transient execution attacks[J]. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2023, 70(12): 5048–5059. doi: 10.1109/TCSI.2023.3298913.

[14] QIU Pengfei, GAO Qiang, WANG Dongsheng, *et al*. PMU-leaker: Performance monitor unit-based realization of cache side-channel attacks[C]. Proceedings of the 28th Asia and South Pacific Design Automation Conference, Tokyo, Japan, 2023: 1–6.

[15] KIM T and SHIN Y. ThermalBleed: A practical thermal side-channel attack[J]. *IEEE Access*, 2022, 10: 25718–25731. doi: 10.1109/ACCESS.2022.3156596.

[16] KOGLER A, JUFFINGER J, GINER L, *et al*. Collide+Power: Leaking inaccessible data with software-based power side channels[C]. Proceedings of the 32nd USENIX Conference on Security Symposium, Anaheim, USA, 2023: 7285–7302.

[17] 刘畅, 杨毅, 李昊儒, 等. 处理器分支预测攻击研究综述[J]. 计算机学报, 2022, 45(12): 2475–2509. doi: 10.11897/SP.J.1016.2022.02475.

LIU Chang, YANG Yi, LI Haoru, *et al*. A survey of branch prediction attacks on modern processors[J]. *Chinese Journal of Computers*, 2022, 45(12): 2475–2509. doi: 10.11897/SP.J.1016.2022.02475.

[18] CANELLA C, VAN BULCK J, SCHWARZ M, *et al*. A systematic evaluation of transient execution attacks and defenses[C]. Proceedings of the 28th USENIX Conference on Security Symposium, Santa Clara, USA, 2019: 249–266.

[19] KORUYEH E M, KHASAWNEH K N, SONG Chengyu, *et al*. Spectre returns! Speculation attacks using the return stack buffer[J]. *IEEE Design & Test*, 2024, 41(2): 47–55. doi: 10.1109/MDAT.2024.3352537.

[20] KIM J, CHUANG J, GENKIN D, *et al*. FLOP: Breaking the apple M3 CPU via false load output predictions[C]. Proceedings of the 34th USENIX Conference on Security Symposium, Seattle, USA, 2025.

[21] KIM J, GENKIN D, and YAROM Y. SLAP: Data speculation attacks via load address prediction on apple silicon[C]. 2025 IEEE Symposium on Security and Privacy, San Francisco, USA, 2025: 3549–3566. doi: 10.1109/SP61157.2025.00098.

[22] TANG A, SETHUMADHAVAN S, and STOLFO S. CLKSCREW: Exposing the perils of security-oblivious energy management[C]. Proceedings of the 26th USENIX Conference on Security Symposium, Vancouver, Canada, 2017: 1057–1074.

[23] QIU Pengfei, WANG Dongsheng, LYU Yongqiang, *et al*. Voltjockey: Breaching TrustZone by software-controlled voltage manipulation over multi-core frequencies[C]. Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, London, UK, 2019: 195–209. doi: 10.1145/3319535.3354201.

[24] KENJAR Z, FRASSETTO T, GENS D, *et al*. V0LTpwn: Attacking x86 processor integrity from software[C]. Proceedings of the 29th USENIX Conference on Security Symposium, 2020: 1445–1461. **(查阅网上资料, 未找到本条文献出版地信息, 请确认)**.

[25] MURDOCK K, OSWALD D, GARCIA F D, *et al*. Plundervolt: Software-based fault injection attacks against intel SGX[C]. 2020 IEEE Symposium on Security and Privacy, San Francisco, USA, 2020: 1466–1482. doi: 10.1109/SP40000.2020.00057.

[26] DENG Shuwen, XIONG Wenjie, and SZEFER J. Analysis of secure caches using a three-step model for timing-based attacks[J]. *Journal of Hardware and Systems Security*, 2019, 3(4): 397–425. doi: 10.1007/s41635-019-00075-9.

[27] XIONG Wenjie and SZEFER J. Survey of transient execution attacks and their mitigations[J]. *ACM Computing Surveys*, 2022, 54(3): 54. doi: 10.1145/3442479.

[28] RAUSCHER F, FIEDLER C, KOGLER A, *et al*. A systematic evaluation of novel and existing cache side channels[C]. Proceedings of the 32nd Annual Network and Distributed System Security Symposium, San Diego, USA, 2025.

[29] DENG Shuwen, XIONG Wenjie, and SZEFER J. A benchmark suite for evaluating caches' vulnerability to timing attacks[C]. Proceedings of the 25th International Conference on Architectural Support for Programming Languages and Operating Systems, Lausanne, Switzerland,

2020: 683–697. doi: 10.1145/3373376.3378510.

[30] ALDAYA A C, BRUMLEY B B, UL HASSAN S, *et al*. Port contention for fun and profit[C]. Proceedings of the 2019 IEEE Symposium on Security and Privacy, San Francisco, USA, 2019: 870–887. doi: 10.1109/SP.2019.00066.

[31] BHATTACHARYYA A, SANDULESCU A, NEUGSCHWANDTNER M, *et al*. SMoTherSpectre: Exploiting speculative execution through port contention[C]. Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, London, UK, 2019: 785–800. doi: 10.1145/3319535.3363194.

[32] LIU Chang, WANG Dongsheng, LYU Yongqiang, *et al*. Uncovering and exploiting AMD speculative memory access predictors for fun and profit[C]. 2024 IEEE International Symposium on High-Performance Computer Architecture, Edinburgh, UK, 2024: 31–45. doi: 10.1109/HPCA57654.2024.00014.

[33] 尹嘉伟, 李孟豪, 霍玮. 处理器微体系结构安全研究综述[J]. 信息安全学报, 2022, 7(4): 17–31. doi: 10.19363/J.cnki.cn10-1380/tn.2022.07.02.
YIN Jiawei, LI Menghao, and HUO Wei. Survey on security researches of processor's microarchitecture[J]. *Journal of Cyber Security*, 2022, 7(4): 17–31. doi: 10.19363/J.cnki.cn10-1380/tn.2022.07.02.

[34] BRIONGOS S, MALAGÓN P, MOYA J M, *et al*. RELOAD+REFRESH: Abusing cache replacement policies to perform stealthy cache attacks[C]. Proceedings of the 29th USENIX Conference on Security Symposium, Berkeley, USA, 2020: 1967–1984. **(查阅网上资料, 未找到本条文献出版地信息, 请确认)**.

[35] XIONG Wenjie and SZEFER J. Leaking information through cache LRU states[C]. 2020 IEEE International Symposium on High Performance Computer Architecture, San Diego, USA, 2020: 139–152. doi: 10.1109/HPCA47549.2020.00021.

[36] CUI Yujie, YANG Chun, and CHENG Xu. Abusing cache line dirty states to leak information in commercial processors[C]. 2022 IEEE International Symposium on High-Performance Computer Architecture, Seoul, Korea, Republic of, 2022: 82–97. doi: 10.1109/HPCA53966.2022.00015.

[37] YAO Fan, DOROSLOVACKI M, and VENKATARAMANI G. Are coherence protocol states vulnerable to information leakage?[C]. 2018 IEEE International Symposium on High Performance Computer Architecture, Vienna, Austria, 2018: 168–179. doi: 10.1109/HPCA.2018.00024.

[38] XU Tianhong, DING A A, and FEI Yunsi. EXAM: Exploiting exclusive system-level cache in apple M-series SoCs for enhanced cache occupancy attacks[EB/OL]. https://arxiv.org/abs/2504.13385, 2025.

[39] HERTOGH M, WIEBING S, and GIUFFRIDA C. Leaky address masking: Exploiting unmasked spectre gadgets with noncanonical address translation[C]. Proceedings of the 2024 IEEE Symposium on Security and Privacy, San Francisco, USA, 2024: 3773–3788. doi: 10.1109/SP54263.2024.00158.

[40] LIPP M, KOGLER A, OSWALD D, *et al*. PLATYPUS: Software-based power side-channel attacks on x86[C]. 2021 IEEE Symposium on Security and Privacy, San Francisco, USA, 2021: 355–371. doi: 10.1109/SP40001.2021.00063.

[41] WAN Junpeng, BI Yanxiang, ZHOU Zhe, *et al*. MeshUp: Stateless cache side-channel attack on CPU mesh[C]. 2022 IEEE Symposium on Security and Privacy, San Francisco, USA, 2022: 1506–1524. doi: 10.1109/SP46214.2022.9833794.

[42] SIDE M, YAO Fan, and ZHANG Zhenkai. LockedDown: Exploiting contention on Host-GPU PCIe bus for fun and profit[C]. 2022 IEEE 7th European Symposium on Security and Privacy, Genoa, Italy, 2022: 270–285. doi: 10.1109/EuroSP53844.2022.00025.

[43] YAROM Y, GENKIN D, and HENINGER N. CacheBleed: A timing attack on OpenSSL constant-time RSA[J]. *Journal of Cryptographic Engineering*, 2017, 7(2): 99–112. doi: 10.1007/s13389-017-0152-y.

[44] MOGHIMI A, WICHELMANN J, EISENBARTH T, *et al*. *MemJam*: A false dependency attack against constant-time crypto implementations[J]. *International Journal of Parallel Programming*, 2019, 47(4): 538–570. doi: 10.1007/s10766-018-0611-9.

[45] CHEN Yun, PEI Lingfeng, and CARLSON T E. AfterImage: Leaking control flow data and tracking load operations via the hardware prefetcher[C]. Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Vancouver, Canada, 2023: 16–32. doi: 10.1145/3575693.3575719.

[46] CHEN Yun, HAJIABADI A, PEI Lingfeng, *et al*. PREFETCHX: Cross-core cache-agnostic prefetcher-based side-channel attacks[C]. 2024 IEEE International Symposium on High-Performance Computer Architecture, Edinburgh, UK, 2024: 395–408. doi: 10.1109/HPCA57654.2024.00037.

[47] ZHU Yongye, CHEN Boru, ZHAO Z N, *et al*. Controlled preemption: Amplifying side-channel attacks from userspace[C]. Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Rotterdam, The Netherlands, 2025: 162–177. doi: 10.1145/3676641.3715985.

[48] ZHANG Xin, ZHANG Zhi, SHEN Qingni, *et al*. SegScope: Probing fine-grained interrupts via architectural

footprints[C]. 2024 IEEE International Symposium on High-Performance Computer Architecture, Edinburgh, UK, 2024: 424–438. doi: 10.1109/HPCA57654.2024.00039.

[49] VAN BULCK J, MINKIN M, WEISSE O, *et al*. Foreshadow: Extracting the keys to the intel SGX kingdom with transient out-of-order execution[C]. Proceedings of the 27th USENIX Conference on Security Symposium, Baltimore, USA, 2018: 991–1008.

[50] BRUNELLA M S, BIANCHI G, TURCO S, *et al*. Foreshadow-VMM: Feasibility and network perspective[C]. 2019 IEEE Conference on Network Softwarization, Paris, France, 2019: 257–259. doi: 10.1109/NETSOFT.2019.8806712.

[51] VAN SCHAIK S, MILBURN A, ÖSTERLUND S, *et al*. RIDL: Rogue in-flight data load[C]. Proceedings of the 2019 IEEE Symposium on Security and Privacy, San Francisco, USA, 2019: 88–105. doi: 10.1109/SP.2019.00087.

[52] SCHWARZ M, LIPP M, MOGHIMI D, *et al*. ZombieLoad: Cross-privilege-boundary data sampling[C]. Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, London, UK, 2019: 753–768. doi: 10.1145/3319535.3354252.

[53] CHEN Yun, HAJIABADI A, and CARLSON T E. GADGETSPINNER: A new transient execution primitive using the loop stream detector[C]. 2024 IEEE International Symposium on High-Performance Computer Architecture, Edinburgh, UK, 2024: 15–30. doi: 10.1109/HPCA57654.2024.00013.

[54] ZHANG Ruiyi, KIM T, WEBER D, *et al*. (M)WAIT for it: Bridging the gap between microarchitectural and architectural side channels[C]. Proceedings of the 32nd USENIX Conference on Security Symposium, Anaheim, USA, 2023: 7267–7284.

[55] CHOI H, KIM S, and SHIN S. AVX timing side-channel attacks against address space layout randomization[C]. 2023 60th ACM/IEEE Design Automation Conference, San Francisco, USA, 2023: 1–6. doi: 10.1109/DAC56929.2023.10247741.

[56] GUO Yanan, ZIGERELLI A, ZHANG Youtao, *et al*. Adversarial prefetch: New cross-core cache side channel attacks[C]. 2022 IEEE Symposium on Security and Privacy, San Francisco, USA, 2022: 1458–1473. doi: 10.1109/SP46214.2022.9833692.

[57] DI DIO A, HERTOGH M, and GIUFFRIDA C. Half spectre, full exploit: Hardening rowhammer attacks with half-spectre gadgets[C]. 2025 IEEE Symposium on Security and Privacy, San Francisco, USA, 2025: 3583–3598. doi: 10.1109/SP61157.2025.00207.

[58] TOBAH Y, KWONG A, KANG I, *et al*. Go go gadget hammer: Flipping nested pointers for arbitrary data leakage[C]. Proceedings of the 33rd USENIX Conference on Security Symposium, Philadelphia, USA, 2024: 1635–1650.

[59] CANELLA C, GENKIN D, GINER L, *et al*. Fallout: Leaking data on meltdown-resistant CPUs[C]. Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, London, UK, 2019: 769–784. doi: 10.1145/3319535.3363219.

[60] ORMANDY T. Zenbleed[EB/OL]. https://cmpxchg8b.com/zenbleed.html, 2023.

[61] RAGAB H, MILBURN A, RAZAVI K, *et al*. CrossTalk: Speculative data leaks across cores are real[C]. Proceedings of the 2021 IEEE Symposium on Security and Privacy, San Francisco, USA, 2021: 1852–1867. doi: 10.1109/SP40001.2021.00020.

[62] ZHANG Jiliang, CHEN Congcong, CUI Jinhua, *et al*. Timing side-channel attacks and countermeasures in CPU microarchitectures[J]. *ACM Computing Surveys*, 2024, 56(7): 178. doi: 10.1145/3645109.

[63] TRUJILLO D, WIKNER J, and RAZAVI K. INCEPTION: Exposing new attack surfaces with training in transient execution[C]. Proceedings of the 32nd USENIX Conference on Security Symposium, Anaheim, USA, 2023: 7303–7320.

蓝泽如：男，博士生，研究方向为处理器硬件安全、处理器微架构安全等.

邱朋飞：男，博士，副教授，博导，研究方向为计算机硬件安全、计算机微架构安全等.

王春露：女，硕士，教授，硕导，研究方向为计算机系统结构、智能计算等.

赵娅喧：女，硕士生，研究方向为处理器微架构安全等.

金　宇：男，硕士生，研究方向为处理器微架构安全等.

张志昊：男，硕士生，研究方向为处理器微架构安全等.

汪东升：男，博士，教授，博导，研究方向为计算机体系结构、神经网络处理器、硬件安全与大数据处理等.

责任编辑：余　蓉

A Survey of Processor Hardware Vulnerability

LAN Zeru[①②]　QIU Pengfei[①②④]　WANG Chunlu[①②]　ZHAO Yaxuan[①②]
JIN Yu[①②]　ZHANG Zhihao[①②]　WANG Dongsheng[③④]

①(*School of Cyberspace Security*, *Beijing University of Posts and Telecommunications*, *Beijing* 100876, *China*)
②(*Key Laboratory of Trustworthy Distributed Computing and Service* (*BUPT*),
Ministry of Education, *Beijing* 100876, *China*)
③(*Department of Computer Science and Technology*, *Tsinghua University*, *Beijing* 100084, *China*)
④(*Zhongguancun Laboratory*, *Beijing* 100194, *China*)

Abstract:

Significance Processor security is a cornerstone of computer system security, providing a trusted execution environment for upper-layer systems and applications. However, the increasing complexity of processor microarchitectures and the widespread integration of performance-driven optimization mechanisms have introduced significant security risks. These mechanisms, primarily designed to enhance performance and energy efficiency, often lack comprehensive security evaluation, thereby expanding the potential attack surface. Therefore, numerous microarchitectural security vulnerabilities have emerged, presenting critical challenges in architectural security research.

Progress Although recent years have witnessed notable progress in the study of hardware vulnerabilities, several key issues remain unresolved. First, the landscape of hardware vulnerabilities is both diverse and complex, yet existing literature lacks a consistent and systematic classification framework. This gap complicates researchers' efforts to understand, compare, and generalize vulnerability characteristics. Second, current studies predominantly focus on individual vulnerability discovery or specific attack implementations, with limited attention to modeling the full vulnerability lifecycle. A comprehensive research framework including vulnerability identification, attack instantiation, and exploitation is still lacking. One pressing challenge is how to efficiently and systematically convert potential vulnerabilities into practical, high-risk attack paths. In addition, unlike software vulnerabilities, hardware vulnerabilities are inherently more difficult to mitigate and impose higher defense costs. These characteristics highlight the need for a more structured and integrated approach to hardware vulnerability research.

Contributions This paper systematically reviews and analyzes processor hardware vulnerabilities reported in major architecture security conferences and academic journals since 2010. It first outlines four primary methods for discovering hardware vulnerabilities and, based on prior studies, proposes a three-step attack model and a novel attack scenario framework. The paper then categorizes and describes existing hardware vulnerabilities according to their behavioral characteristics and consolidates eight evaluation metrics for side-channel vulnerabilities derived from related research. To assess the feasibility and scope of various attack types, representative vulnerabilities are selected for experimental validation across multiple processor platforms, with in-depth analysis of the results. In addition, the study provides a systematic evaluation of current defense and mitigation mechanisms for hardware vulnerabilities. Finally, it discusses future research directions from both offensive and defensive perspectives.

Prospects Future research in processor hardware security is expected to focus on new attack surfaces introduced by increasingly diversified microarchitectural optimizations. Key areas will include the development of system-level collaborative defense mechanisms, automated verification tools, and integrated strategies to enhance awareness and precision in mitigating hardware-level information leakage risks.

Key words: Hardware vulnerabilities; Computer architecture security; Side channels; Transient execution; Fault injection