基于动态分支过滤的SMT执行端口侧信道安全防护

岳晓萌[1,2],杨秋松[1],李明树[1]
(1. 中国科学院软件研究所基础软件国家工程研究中心,北京 100190; 2. 中国科学院大学,北京 100049)

摘 要: 同时多线程(Simultaneous Multi-Threading,SMT)技术是提升线程级并行度的重要微架构优化技术之一,以SMoTherSpectre为代表的利用SMT环境下共享分支预测器和执行端口的时间侧信道攻击表明SMT技术在提升性能的同时也存在显著的安全隐患. 基于记录分支预测错误刷新及调整执行端口资源使用策略,提出了一种SMT环境下执行端口时间信道攻击防护方法. 该方法实现了分支过滤和动态资源使用策略修改组件,在防护有效性上可以达到关闭SMT技术的防护效果,性能开销仅为关闭SMT技术的22%,硬件开销可控.

关键词: 同时多线程; 时间信道; 侧信道; 执行端口; 安全防护
中图分类号: TP309.2 **文献标识码:** A **文章编号:** 0372-2112(2022)07-1594-06
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20210210

SMT Port Side Channel Attack Defending Method Based on Dynamic Branch Filter

YUE Xiao-meng[1,2], YANG Qiu-song[1], LI Ming-shu[1]
(1. *National Engineering Research Center for Fundamental Software, Institute of Software, Chinese Academy of Sciences, Beijing* 100190, *China*;
2. *University of Chinese Academy of Sciences, Beijing* 100049, *China*)

Abstract: Simultaneous multi-threading(SMT) is one of the important micro-architecture optimization technologies to improve thread-level parallelism. The timing channel attack represented by SMoTherSpectre using shared branch predictors and execution ports in SMT environment shows that SMT technology has significant security risks as well as performance improvements. Based on recording branch misprediction refresh and dynamically adjusting the execution port resource utilization strategy, this paper proposes an approach for defending a timing channel attack on execution port in SMT environment. The approach implements a branch filter and a dynamic resource editor. This approach can achieve the same protection effect of turning off SMT technology, and the performance cost is only 22%, meanwhile, the hardware cost is controllable.

Key words: simultaneous multi-threading; timing channel; side channel; execution port; security defense

1 引言

同时多线程通过增加少量硬件资源,把1个物理核映射成多个逻辑核,同时运行的线程可共享处理器的硬件资源. 因为线程执行总有空闲或者等待的时间,SMT环境下当1个线程进入空闲或等待,另外的线程可以继续执行,从而更加合理的使用处理器资源. Intel最早于2002年的Pentium 4处理器上使用超线程(Hyper-Threading,HT)技术[1]. Intel HT技术在1个物理核内实现了2个逻辑核,本文中的SMT技术均以Intel提出的超线程技术作为主要参考.

SMT技术增加了很多处理器微架构安全问题利用的场景和机会,因此有研究人员评价SMT技术是"廉价的硬件并行意味着廉价的安全性"[2]. 2018年,Ge等人[3]总结了2002年到2018年期间各个系统层级由于资源共享导致的时间信道安全问题及缓解措施,提出基于SMT技术产生的硬件线程级时间信道安全问题相对于跨核和跨处理器的时间信道安全问题是更难应对和防护的,其将利用SMT技术的攻击称为"简单攻击

收稿日期:2021-02-03;修回日期:2021-12-22;责任编辑:李勇锋
基金项目:"核高基"国家科技重大专项基金(No.2014ZX01029101-002);中国科学院战略性先导科技专项(No.XDA-Y01-01);中国科学院战略性先导科技专项(No.XDC05020200)

(Easy Attacks)",而SMT环境下时间信道安全问题的防护或缓解方法大部分集中在缓存结构上. 普林斯顿的He等人在2017年发表了在侧信道攻击下的缓存是否安全的文章[4],通过分析攻击成功概率来总结已有防护型缓存架构的安全性.

2006年,Wang等人[5]首次使用执行单元的资源竞争构建了隐蔽信道. 2018年,Aldaya等人[6]在Intel Skylake和Kabylake微架构上利用其SMT技术开启后执行端口竞争问题提出了PortSmash攻击,该攻击通过构建不同执行端口的指令冲突场景,经过一定规模的时间信道收集和降噪后获取受害者信息. 2019年,IBM的研究团队提出了一种叫SMoTherSpectre[7]的新型"Spectre"类型攻击,其在PortSmash攻击基础上同时利用了SMT环境下分支预测器和执行端口共享的特征,通过构建投机代码重用的攻击场景提高执行端口冲突产生时间信道的准确性.

针对上述SMT环境下共享执行端口或执行单元的时间侧信道攻击,研究人员也提出了一些防护方法. Percival等人[8]在早期提出可以禁用SMT来防范此类攻击,其可以彻底解决执行单元或执行端口双线程共享产生的时间侧信道问题. Hu[9]提出通过将噪声添加到与进程相关的所有时间信息中,达到降低时序信道带宽的目的进行防护. 2019年,Zhang等人[10]提出了名为DDM(Demand-based Dynamic Mitigation)的防护方法,DDM通过软件手段动态关闭SMT技术来达到防护目的,尚未有从处理器微架构角度防护的案例.

SMoTherSpectre[7]在利用冲突指令时,仅仅使用了非常小的冲突窗口,增加了防护难度. 本文提出了一种针对现代处理器SMT环境下SMoTherSpectre执行端口时间信道攻击的防护方法. 主要贡献有如下两点:

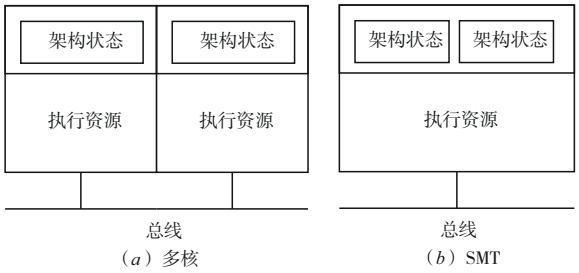
(1)首次使用TSG模型完成SMoTherSpectre攻击原理的建模并推导对应的防护路径,针对投机代码瞬态执行窗口的分支刷新特性基于分支指令执行状态设计动态调整执行端口资源分配策略的防护方案;

(2)提出并设计了一种基于记录分支预测错误刷新并动态调整执行端口资源使用策略的SMT环境下执行端口时间信道防护方法,可以有效防护以SMotherSpectre为代表的组合使用诱导分支预测投机执行及执行端口冲突的时间信道攻击.

2　背景知识

2.1　SMT微架构特征

如图1(*a*)所示[1],多核架构下2个线程的架构状态和执行资源是分开的,2个线程只通过线程间的总线进行交互. 如图1(*b*)所示,SMT技术开启后,2个线程的架构状态依然是分开的,但是其会共享执行资源,SMT技术的优势是可以灵活分配处理器内部资源,使得大部分数据结构可以被多个线程共享且同一周期可以调度多个线程的操作同时执行,最大化得提高资源利用率.



(*a*) 多核　　(*b*) SMT

图1　SMT技术特征

2.2　SMT技术的多端口调度算法

在超标量处理器微架构设计中,指令集架构中描述的宏指令会经过取指和译码的流程形成对应的微操作. 微操作在进行指令执行前,会通过指令解析完成对应执行端口的绑定. 在关闭SMT技术的情况下,处理器流水线内部只有1个线程,完成端口绑定的微操作会通过对应的端口进入执行单元;在开启SMT技术的情况下,处理器解码系统和乱序执行系统交互过程中,因为在重命名和分配微架构设计中约束了同一时刻只能处理来自于同一个线程的微操作,因此,每个周期也只有1个线程的微操作可以进入调度器. 在调度器微架构设计中为了更快地调度和分发微操作,其并不关心微操作的线程信息,只考虑其是否就绪("就绪"指的是某条微操作已满足被选择并发射到执行端口的条件),如果有同时就绪的微操作则年龄更老的优先发射,因此,调度器可以同一周期向不同执行端口发射2个线程的微操作.

如果2个线程在调度器内的微操作指向同一个端口,那么由于分配时的先后关系,2个线程的微操作会自然携带不同的年龄信息. 调度器中有2个关键数据结构,一个是年龄矩阵,另一个是就绪矩阵,虽然2个线程的微操作即使都处于就绪状态,依赖于年龄矩阵的先后关系,调度器会先发射更"老"的线程的微操作进入相应端口,那么已经就绪但由于较"年轻"导致无法发射到相同端口的另外线程的微操作就产生了延迟,此时就出现了线程间的端口冲突.

因此,某线程在特定的执行端口下执行指令并且度量其执行程序的时间就可以推断同一端口或执行单元下另一线程的执行情况.

2.3　TSG模型

TSG模型是2020年由普林斯顿大学He等人[11]针对处理器微架构侧信道安全问题提出的一种新的攻击图模型,是一种由顶点和顶点之间的边来形成序列的

有向无环图,示例如图2.

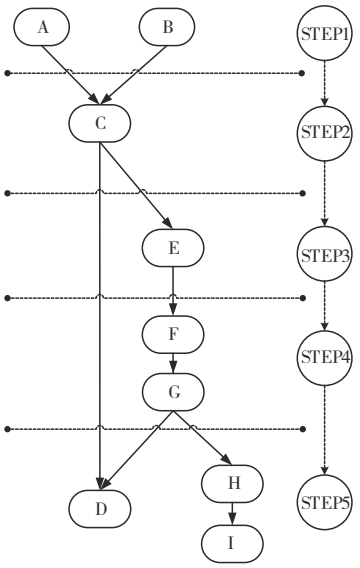


图2 TSG示例

TSG中的顶点表示操作,例如访问缓存行、刷新缓存行等,如示例中的顶点A,B,C,…,I;

TSG中的有向边(示例中带方向箭头的线)表示2个顶点间的依赖,如果2个顶点u和v之间有u到v的有向边,表示u在v之前发生,如示例中顶点A即在顶点C之前发生;

TSG中的路径表示连接顶点间的连续有向边,如示例中顶点A到顶点D的路径经过2个有向边A->C,C->D;

TSG中顶点的序列可以由一个带有序列信息的顶点集合表示,示例中的其中一个有效序列S=(A,B,C,D,E,F,G,H,I),顶点间的顺序需要精确描述顶点间的依赖关系;

TSG条件竞争指的是TSG模型中的顶点u和v之间,如果有2个不同的有效序列S_1和S_2,在S_1中,u在v之前,在S_2中u在v之后,TSG模型中存在条件竞争意味着缺失安全依赖性,示例中顶点D和E就存在条件竞争,因为存在S_1=(A,B,C,D,E,F,G,H,I),也存在S_2=(A,B,C,E,D,F,G,H,I).

TSG模型遵循如下定理:

对于任意一对顶点u和v,当且仅当有一条有向路径连接u和v这2个顶点时,顶点u和v则没有条件竞争.

3 防护目标及防护方法推导

SMoTherSpectre[7]在冲突构建阶段使用了Spectre[12]分支诱导的方式来加强冲突构建的成功率.其使用了分支预测器投机代码重用的攻击场景,即混合了分支预测器资源冲突和执行端口资源冲突这2个微架构组件的SMT技术特征,具有更强的适应性和攻击准确性.基于2.3节的TSG模型建立规则,利用SMT环境下执行端口及执行单元进行时间信道攻击的TSG模型建立如下:

TSG模型顶点包括:

①连续执行端口P/执行单元U指令;

②踢出BTB预测项;

③条件/间接分支指令执行;

④执行私密操作O;

⑤调度/执行资源冲突;

⑥分支刷新(授权检查);

⑦度量连续执行端口P/执行单元U指令执行总时间T.

基于TSG模型定理,要保证没有条件竞争,需要切断涉及条件竞争顶点的有效路径,即上述序列1到序列4中的顶点路径.

当前已有的防护手段分别是采用硬件隔离、时间隔离和增加全局时间噪声三种方式,对应典型的防护方法有SMT技术禁用/DDM及修改时间度量指令,分别应用于图3的①②③位置.除了已有的防护手段外,可以从顶点-执行私密操作O(④)、顶点-调度/执行资源冲突(⑤)和顶点-分支刷新(授权检查)(⑥)进行防护机制设计.

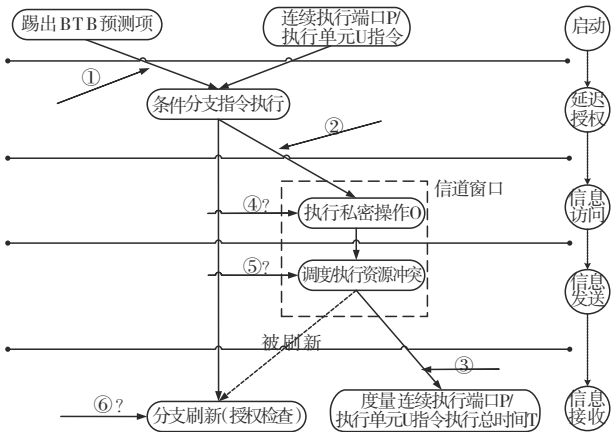


图3 SMoTherSpectre攻击TSG模型防护方法推导

基于TSG模型推导,可以利用顶点-调度/执行资源冲突和顶点-分支刷新(授权检查)的执行路径,来形成防护方法,涉及顶点及相关防护策略如下:

④作用于顶点-执行私密操作O,目的是当带有私密信息的分支指令执行时,相对于该分支指令更年轻的操作无法产生时间信道.

⑤作用于顶点-调度/执行资源冲突,目的是当检测到分支指令时,在调度算法和端口绑定策略上进行双线程执行端口和执行资源的隔离,消除资源冲突.

⑥作用于顶点-分支刷新(授权检查),目的是使得SMT环境下当一个线程调度分支指令时,首先将分支指令信息记录下来,然后使能调度/执行资源隔离,当触发分支刷新时,记录该分支刷新的线程到对应分支指令记录表中;当没有分支刷新时,删除该分支指令记录表,分配对应分支指令过滤表,分支指令过滤表维持可过滤掉的分支指令类型,命中分支指令过滤表的操作则无需使能调度/执行资源隔离机制,提升执行资源的利用率.

4　防护设计与实现

本防护设计微架构实现共包含2个子模块,分别是分支过滤器及策略修改器. 分支过滤器中包含2个核心数据结构,分别是分支过滤表和分支刷新表,分别记录可过滤的分支操作和触发刷新的分支操作,分支过滤器会产生隔离使能信号和优先级向量传递给策略修改器,最后通过策略修改器进行处理器分配单元及调度器选择逻辑的算法策略修改,消除由于执行端口冲突产生的时间信道. 详细的微架构示意图如图4所示.

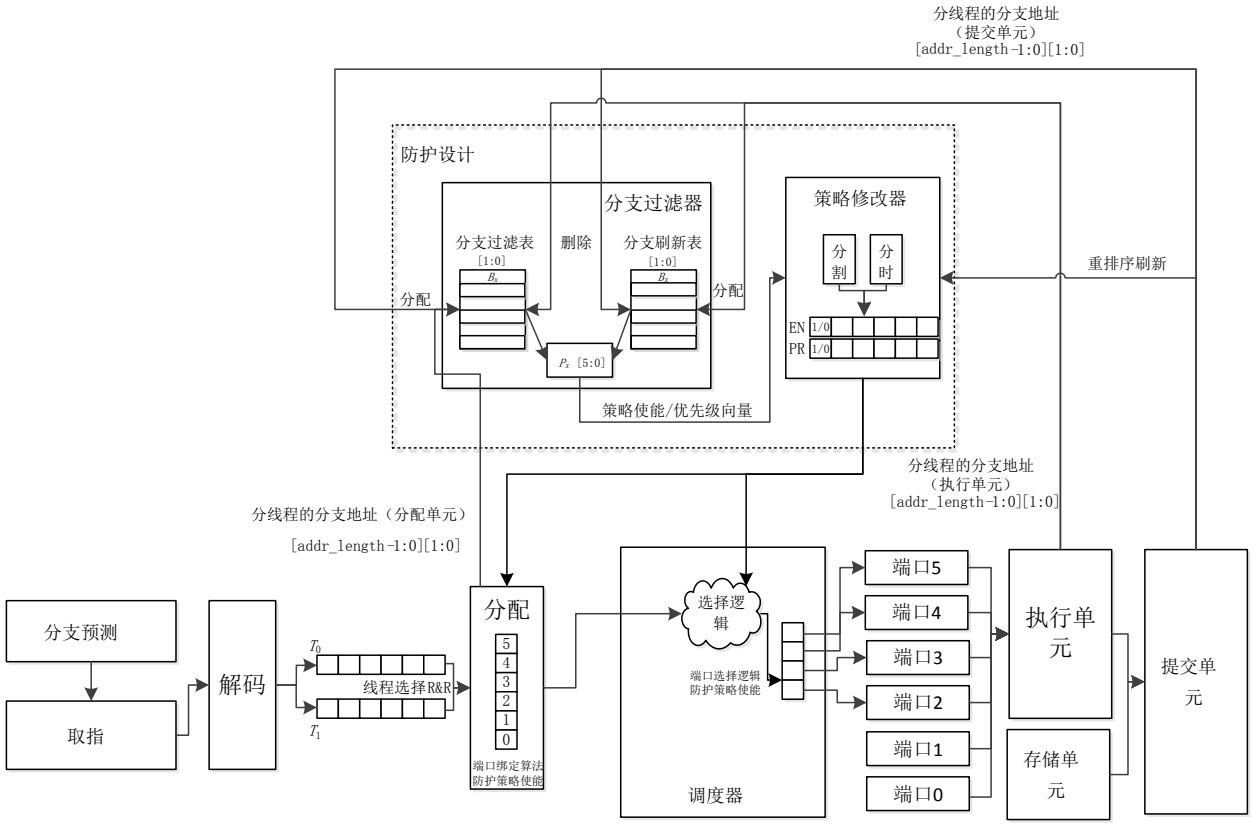


图4　防护设计微架构图

5　评估

5.1　评估方法

本文使用Gem5[13]模拟一个高性能多执行端口的超标量处理器作为本文防护方法的防护有效性和性能开销的评估平台.

对于防护有效性评估,选用SMoTherSpectre攻击中的POC代码片段,在不同冲突指令的规模量级下进行度量时间的比较.

对于性能评估,选用SPEC CPU 2006 INT测试集[14]的7个测试程序作为性能评估的参考主要评估防护方法对性能的影响以及同关闭SMT技术产生的性能影响进行对比(注:测试集共12个测试程序,有5个测试程序在Gem5模拟器中运行异常).

对于硬件开销的评估,使用华力HLMC 40GP工艺进行防护方法的硬件开销评估,主要包含面积及时序.

5.2　防护有效性评估

图5中的SMTwC和SMTwoC分别代表了SMT环境下出现指令端口冲突的执行时间和无指令端口冲突的执行时间,呈现非常明显的差别且随着量级增长呈线性趋势. 图中的SMTwDwC和SMTwDwoC分别代表了使能防护机制的SMT环境下出现指令端口冲突的执行时间和无指令端口冲突的执行时间,基于动态分支过滤的SMT执行端口时间信道防护方法可以达到禁用或动态关闭SMT技术的防护效果.

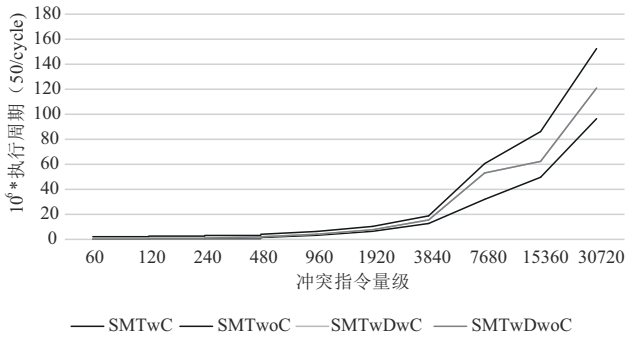


图5 防护效果评估

5.3 性能开销评估

由图6看出，使用本防护设计后SPEC CPU 2006 INT测试集测试出的执行性能相比于开启SMT技术平均下降8.6%，仅为关闭SMT技术性能下降比例的22%. 其中性能下降最明显的是471，相比开启SMT技术性能下降比例为50.86%，不包括471的其他6个SPEC CPU 2006 INT测试程序平均下降比例为1.6%. 在471的程序特征下，开启防护机制后性能比关闭SMT技术要差，其他程序均优于关闭SMT技术，且以462和473为代表的测试程序在增加防护设计后性能没有下降，基本和开启SMT技术持平.

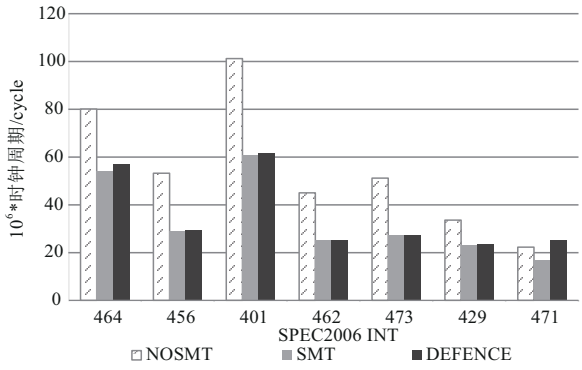


图6 性能对比

5.4 硬件开销评估

本防护机制使用Verilog语言进行RTL实现. 然后使用HLMC 40GP工艺和ASIC设计流程和工具进行面积和时序的评估.

使用DC-Compiler工具，设置时钟约束为2 GHz，clk_uncertainty参数为100 ps. 本防护设计的时序和面积数据如表1所示.

表1 硬件开销评估

模块	面积开销/μm^2	时序开销(reg2reg)/ps	时序开销(in2reg)/ps	时序开销(reg2out)/ps
smt_smother_defend	74 947	368	369	300
branch_filter	74 852	无	无	无
policy_modifier	95	无	无	无
int exec(对比)	302 979	无	无	无

为了进行面积和时序的对比，使用DC-Compiler工具在相同的设置和约束下分析了6端口整型执行单元的整体面积约为302 979 μm^2，本防护设计单元的面积为其24.7%，开销可控.

6 结论

本文使用TSG模型对SMoTherSpectre攻击路径和特征进行描述，通过建模进一步推导SMT执行端口共享时间信道攻击的微架构防护手段可以为防护方案的设计提供有效的理论依据.

本文提出的基于动态分支过滤并进行资源调整的SMT执行端口时间信道安全防护方法通过在处理器微架构级别对端口绑定和调度算法进行防护设计，记录分支预测错误刷新以及正常执行状态实时调整执行端口资源使用策略，达到防护SMoTherSpectre攻击的目的.

参考文献

[1] MARR D T, BINNS F, HILL D L. Hyper-threading technology architecture and microarchitecture[J]. Intel Technology Journal, 2002, 6(1): 1-12.

[2] ACIÇMEZ O, SEIFERT J P. Cheap hardware parallelism implies cheap security[C]//Proc of the Workshop on Fault Diagnosis & Tolerance in Cryptography. Vienna: IEEE, 2007: 80-91.

[3] GE QIAN, YAROM Y, COCK D, et al. A survey of microarchitectural timing attacks and countermeasures on contemporary hardware[J]. Journal of Cryptographic Engineering. 2016, 8(1): 1-27.

[4] HE Z, LEE R B. How secure is your cache against side-channel attacks?[C]//50th Annual IEEE/ACM International Symposium. Cambridge: ACM, 2017: 341-353.

[5] WANG ZHENGHONG, LEE R. Covert and side channels due to processor architecture[C]//Proc of the 22nd Annual Computer Security Applications Conference. Miami Beach: IEEE, 2006: 473-482.

[6] ALDAYA A, BRUMLEY B, HASSAN S U, et al. Port contention for fun and profit[C]//Proc of the Symposium

on Security and Privacy. San Francisco: IEEE, 2019: 19-23.

[7] BHATTACHARYYA A, SANDULESCU A, NEUGSCHWANDTNER M, et al. SMoTherSpectre: Exploiting speculative execution through port contention[C]//Proc of the 2019 ACM SIGSAC Conf on Computer and Communications Security. London: ACM, 2019: 785-800.

[8] PERCIVAL C. Cache missing for fun and profit[J/OL]. (2019-12-16) [2021-12-22] http://www. daemonology. net/papers/htt.pdf.

[9] HU WEIMING. Reducing timing channels with fuzzy time [C]//Proc of the 1991 IEEE Computer Society Symposium on Research in Security and Privacy. Oakland, CA: IEEE, 1991: 8-20.

[10] ZHANG YUE, ZHU ZIYUAN, MENG DAN. DDM: A demand-based dynamic mitigation for SMT transient channels[C]//IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking. Xiamen: IEEE, 2019: 614-621.

[11] HE Z, et al. New models for understanding and reasoning about speculative execution attacks[C]//IEEE International Symposium on High-Performance Computer Architecture(HPCA). Seoul: IEEE, 2021:40-53.

[12] KOCHER P, GENKIN D, GRUSS D, et al. Spectre attacks: Exploiting speculative execution[J]. Communications of the ACM, 2020, 63(7):93-101.

[13] BINKERT N, BECKMANN B, BLACK G, et al. The gem5 simulator[J]. ACM SIGARCH Computer Architecture News, 2011, 39(2): 1-7.

[14] HENNING J L. SPEC CPU2006 benchmark descriptions [J]. ACM SIGARCH Computer Architecture News, 2006, 4(34): 1-17.

作者简介

岳晓萌　男,1989年12月生,山东青州人.2021年毕业于中国科学院大学,计算机软件与理论博士,主要研究方向为操作系统、计算机架构和系统安全.

E-mail: xiaomeng@iscas.ac.cn

杨秋松　男,1977年生,博士,教授、博士生导师.主要研究方向为操作系统、软件工程和系统安全.

E-mail: qiusong@iscas.ac.cn

李明树　男,1966年生,博士,教授、博士生导师.主要研究方向为操作系统、软件工程和分布式系统.

E-mail: mingshu@iscas.ac.cn