中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES

操作系统中的层级内隔离技术

武成岗

中国科学院计算技术研究所处理器芯片全国重点实验室

团队介绍

武成岗

中科院计算所研究员
中科院关键技术人才
CCF体系结构专委主任
CCF理事会理事

研究方向：

- 安全操作系统、操作系统抗攻击能力测评、漏洞挖掘和利用、程序分析、虚拟化、二进制翻译。

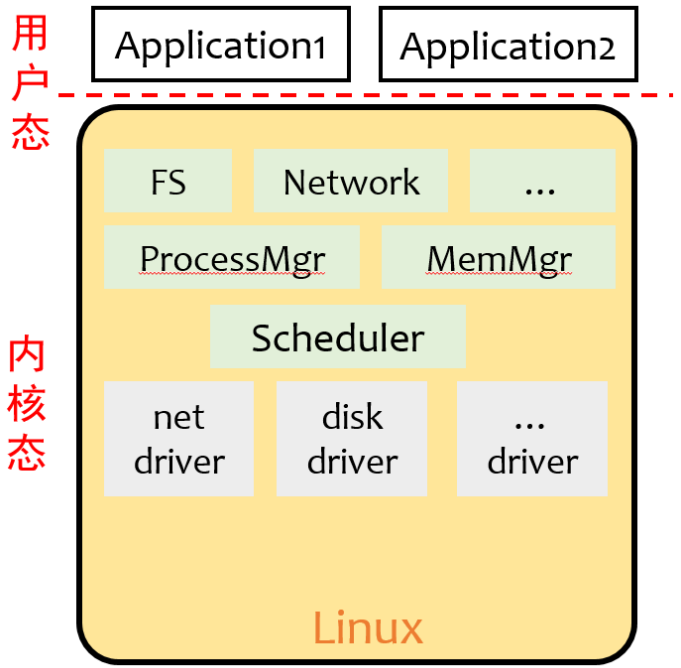
成果：

- 在S&P、USENIX Security、CCS、ATC、TDSC、TSE、ICSE、ASE、TPDS、TACO、TCAD、SIGMETRICS、PACT、CGO、VEE、DATE等发表论文30余篇
- **CCS'22 最佳论文提名奖、CCS'23 杰出论文奖**
- **获省部级科技成果二等奖2次**
- 获得授权专利30项、软件著作权5项

联系方式

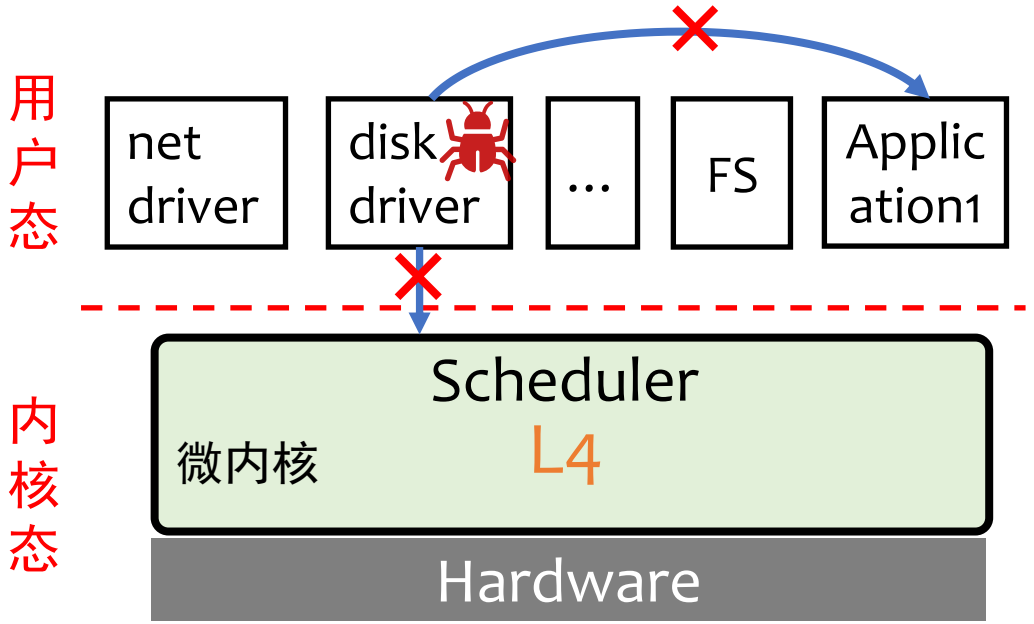
操作系统的安全性　宏内核 VS 微内核

宏内核



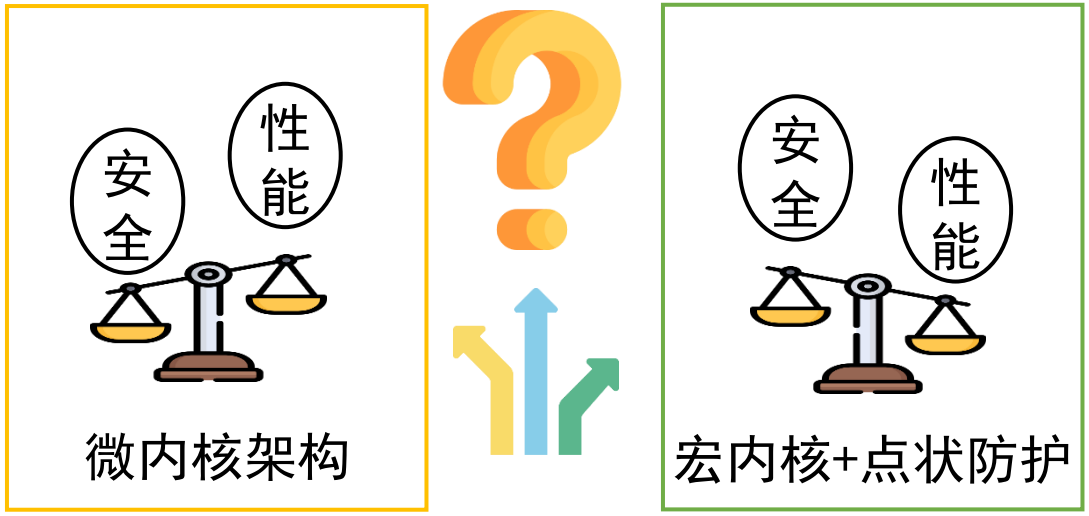
优点：模块间交互效率高
缺点：所有内核代码处于同一地址空间，单点缺陷就能够导致整个系统破防

微内核

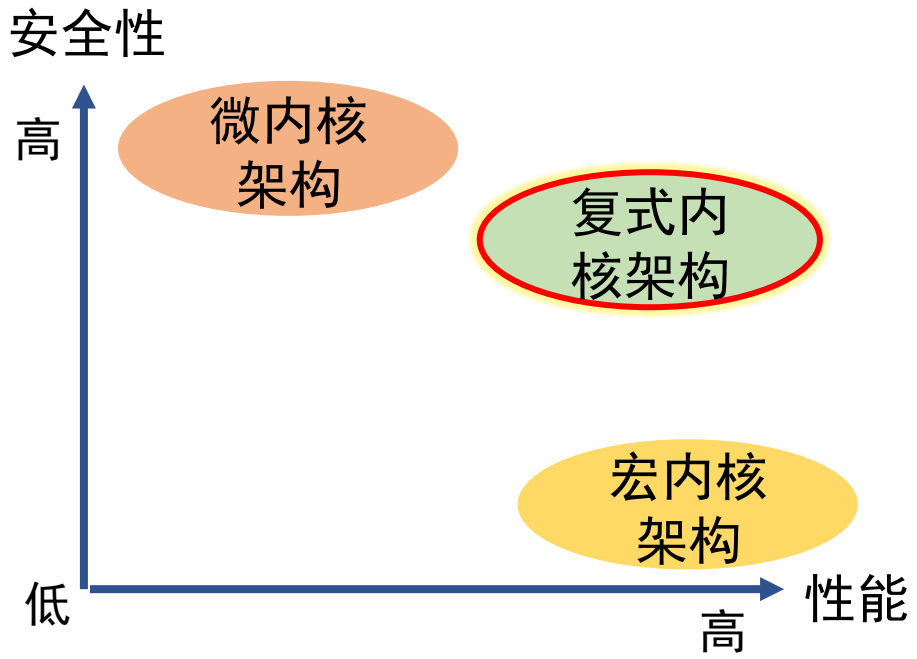


优点：模块间隔离提升了安全性
缺点：模块间交互性能开销大

目标：如何同时获得微内核的安全性和宏内核的高性能？

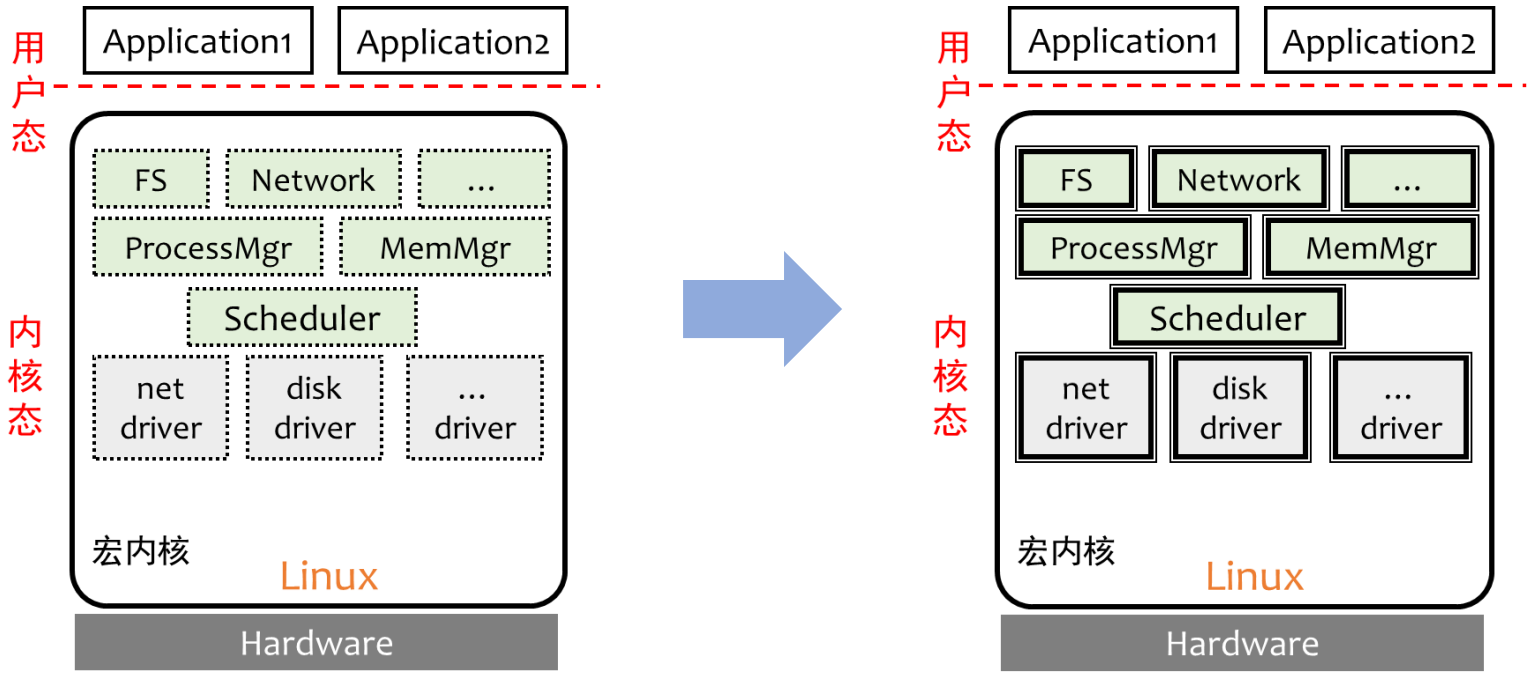


目标



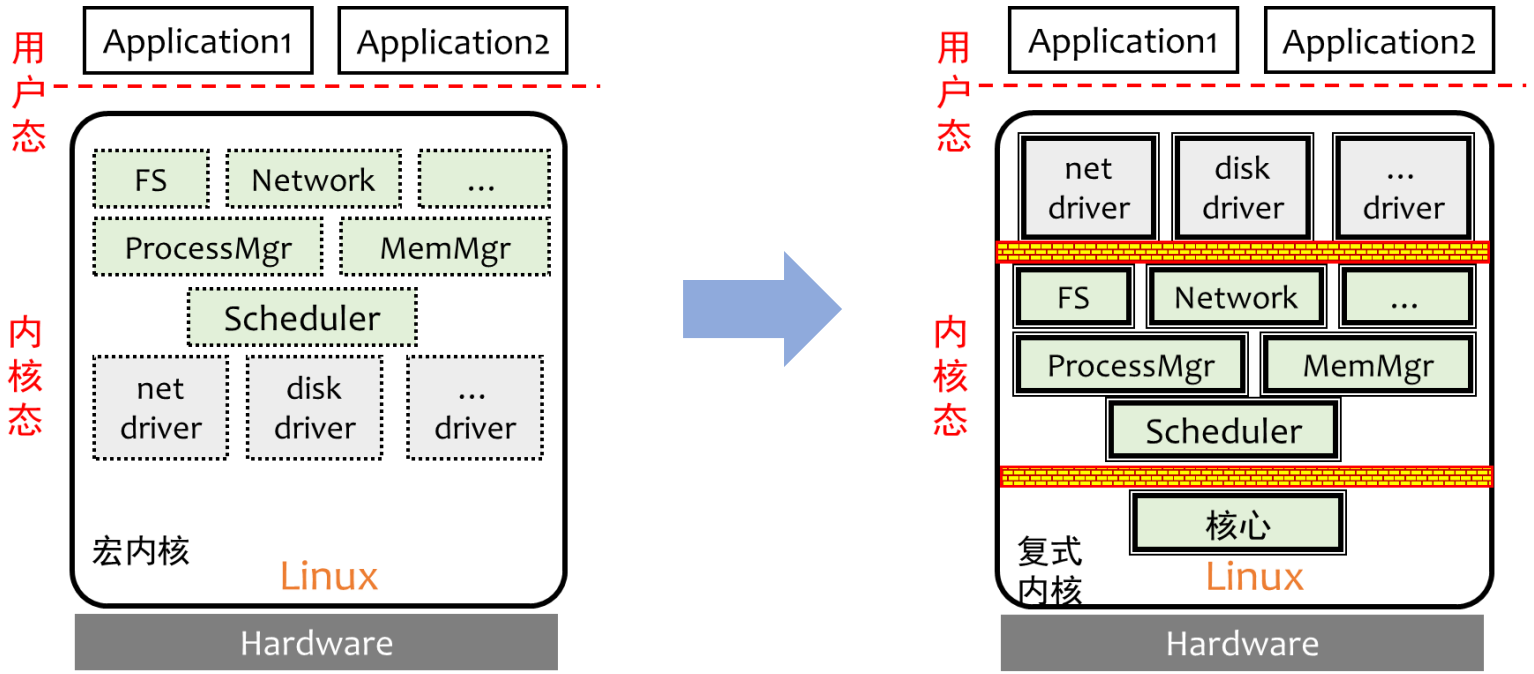
技术方案

基本思想：划分&最小权限（复式内核）



技术方案

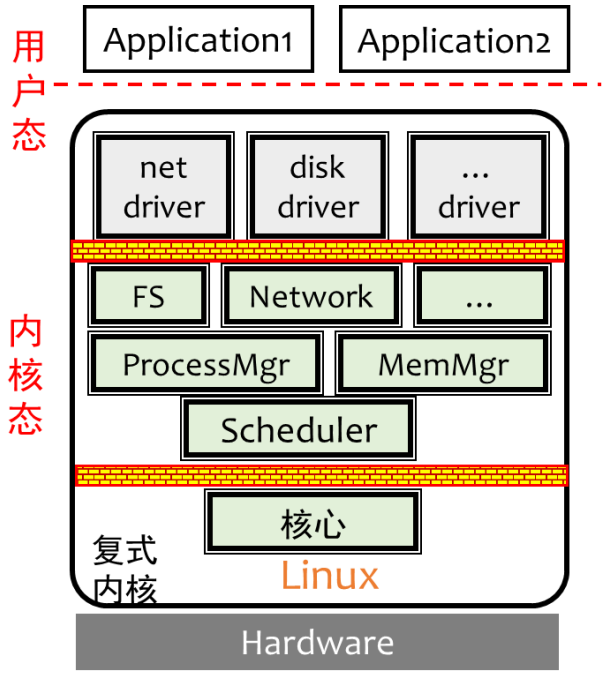
基本思想：划分&最小权限（复式内核）



技术方案

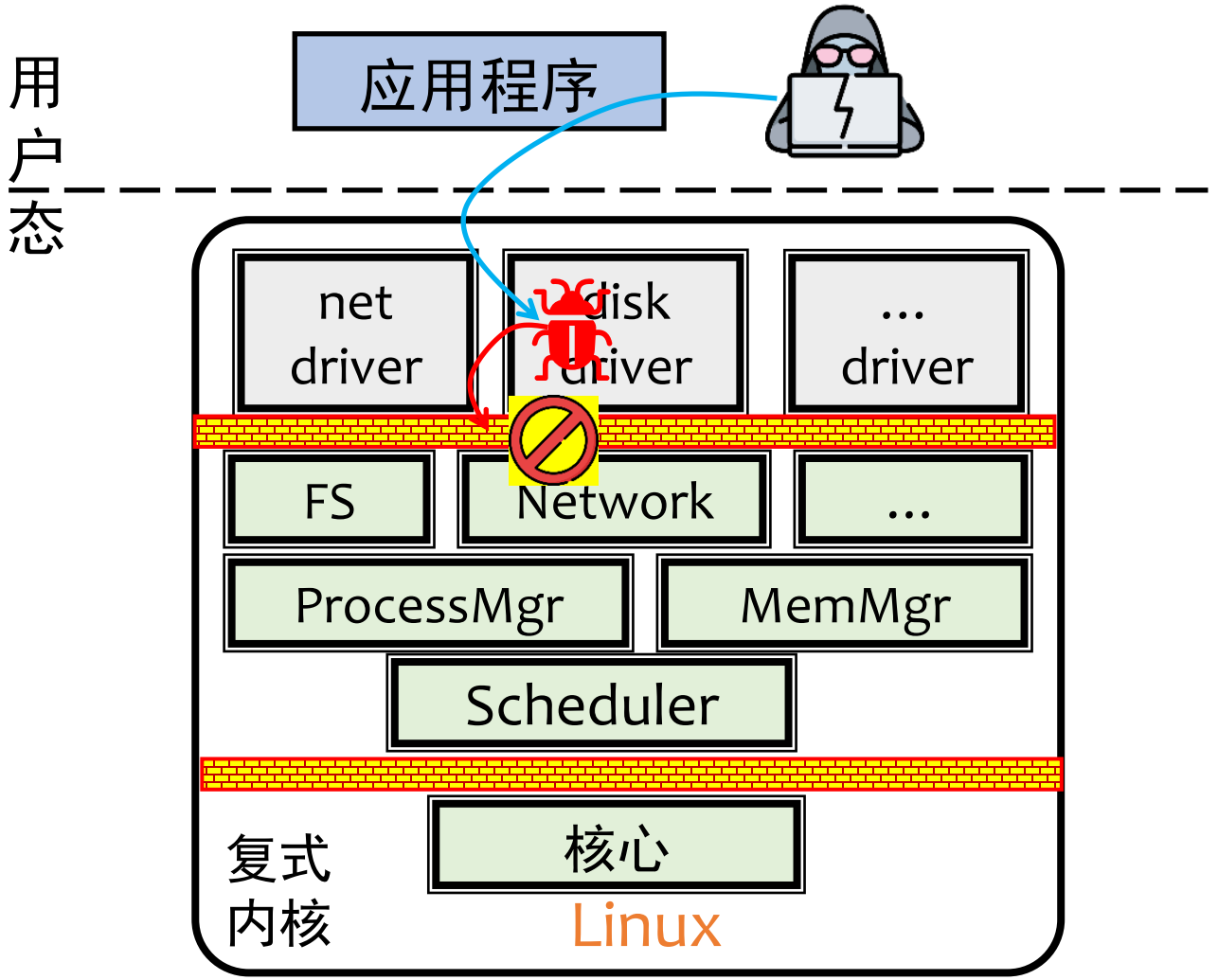
基本思想：划分&最小权限（复式内核）

特点：
1、内核内部再次分层
2、每个模块只能访问自己的数据
3、每个模块只能在内部跳转
4、跨内部层次跳转和访存需要过隔离门



防护能力

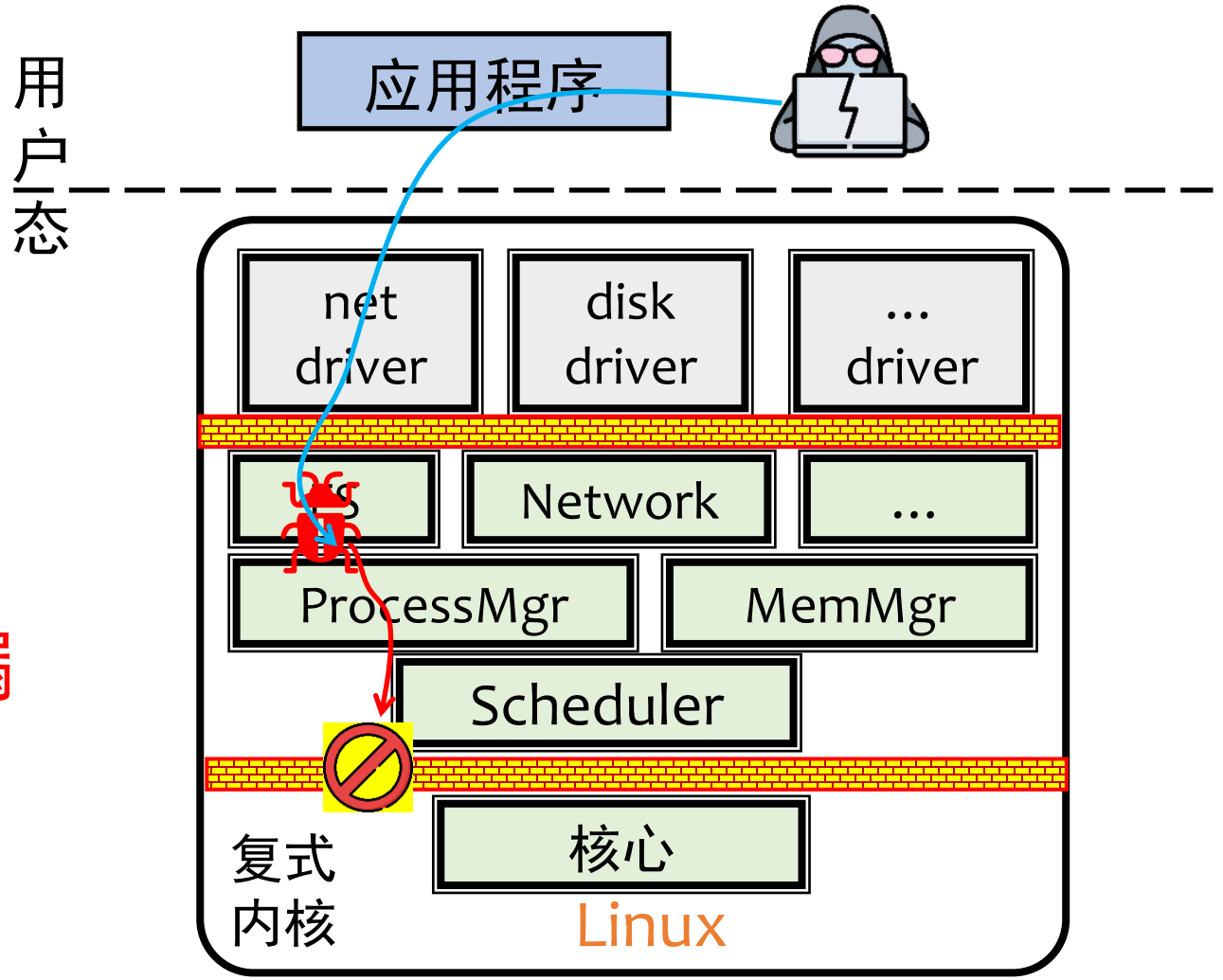
①阻止攻击者利用驱动漏洞发起攻击



防护能力

①阻止攻击者利用驱动漏洞发起攻击

②阻止攻击者利用core kernel模块的漏洞发起攻击

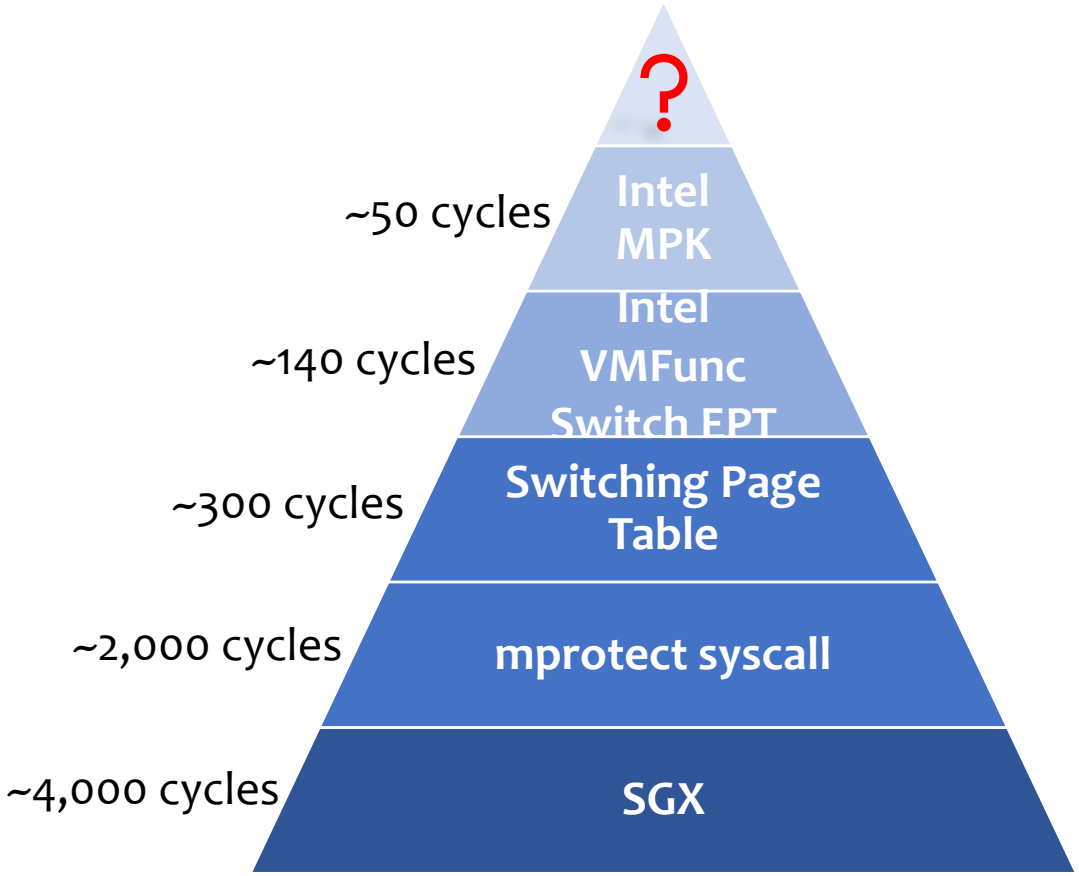


与DARPA不谋而合

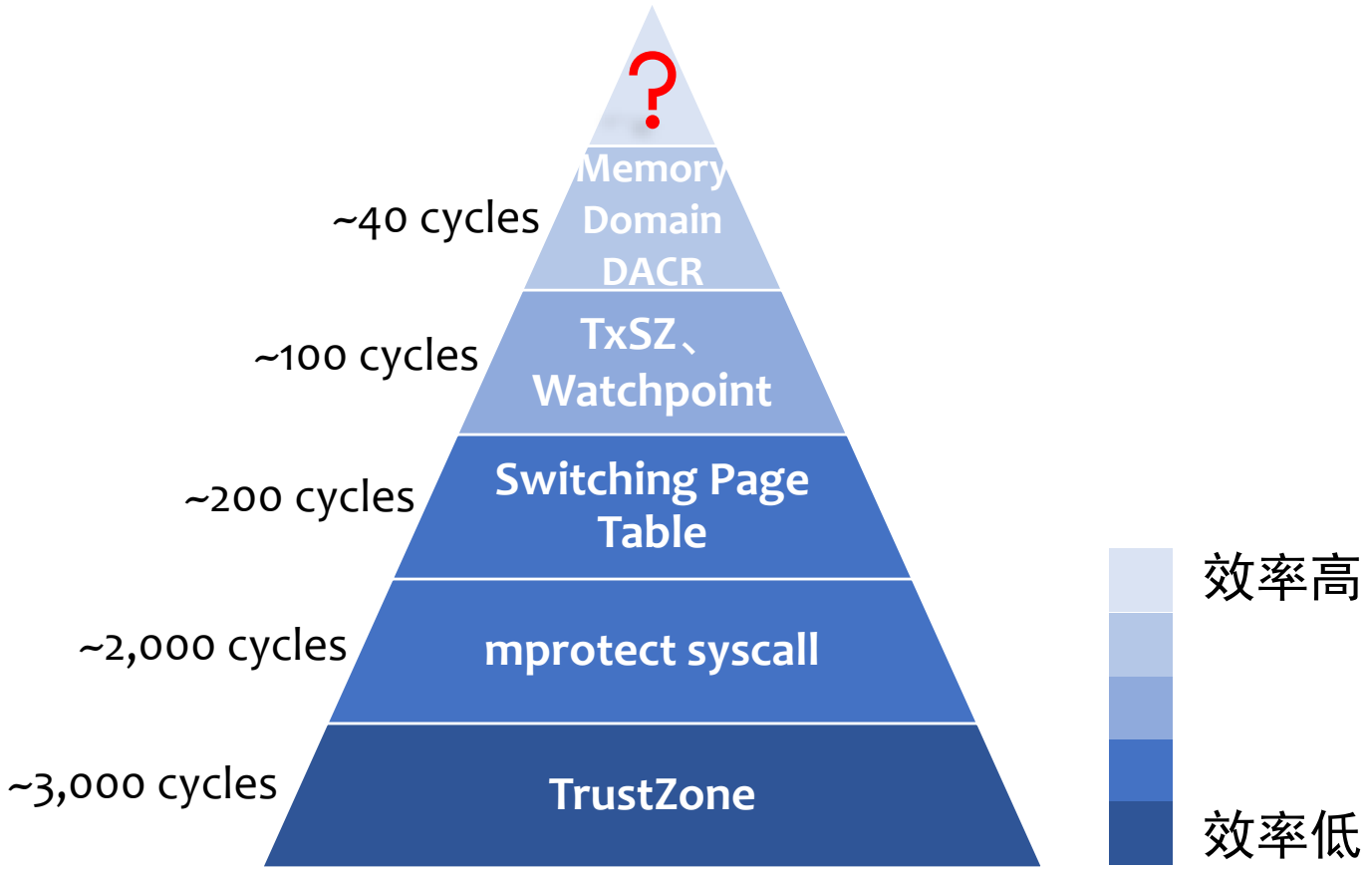
Federal Grant Title:	**Compartmentalization and Privilege Management (CPM)**
Federal Agency Name:	**DARPA Information Innovation Office (DOD-DARPA-I2O)**
Grant Categories:	**Science and Technology**
Type of Opportunity:	**Discretionary**
Funding Opportunity Number:	**HR001123S0028**
Type of Funding:	**Cooperative Agreement**
CFDA Numbers:	**12.910**
CFDA Descriptions:	**Information not provided**
Current Application Deadline:	**October 3rd, 2023**
Original Application Deadline:	**October 3rd, 2023**
Posted Date:	**April 4th, 2023**
Creation Date:	**April 4th, 2023**

现有处理器的层内隔离机制的效率总览

可用的层内隔离机制非常少，并且隔离效率（安全域切换）一般



X86层内隔离机制



ARM层内隔离机制

提升层级内隔离效率的手段

②挖掘现有硬件潜力
- 组合和巧用现有硬件
- 借用其他权级的硬件

①设计新的硬件支持
- 设计新型隔离原语

③基于随机的伪隔离
- 研究代码和数据随机化

提升层级内隔离效率的手段

②挖掘现有硬件潜力
- 组合和巧用现有硬件
- 借用其他权级的硬件

①设计新的硬件支持
- 设计新型隔离原语

③基于随机的伪隔离
- 研究代码和数据随机化

挖掘现有硬件潜力

X86硬件和应用场景

Intel MPK	用户态隔离、内核态隔离
Intel VMFunc	用户态隔离、内核态隔离
Intel WP	内核态隔离
Intel SMAP	**用户态隔离（S&P 2020）**
Intel CET	**用户态隔离（CCS 2022）**

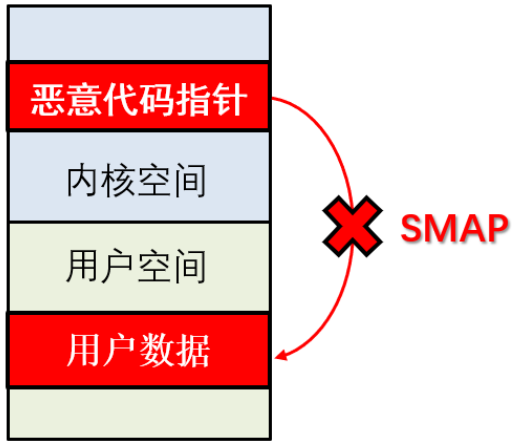
ARM硬件和应用场景

ARM Memory Domain	用户态隔离
ARM Watchpoint	用户态隔离
ARM TxSZ	内核态隔离
ARM TTBRx/ASID	内核态隔离
ARM PAN/LSU	**用户态隔离（CCS 2023）**

挖掘现有硬件潜力——Intel SMAP应用在进程内隔离

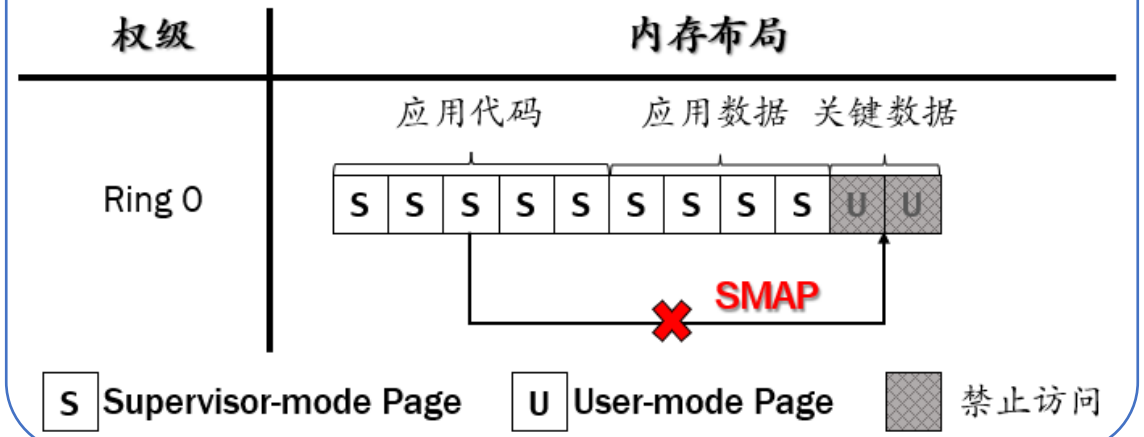
借用特权态硬件SMAP实现高效的进程内隔离——SEIMI

- SMAP机制主要用于阻止特权态代码访问用户页面（User-mode Page）。
- 开关SMAP机制的特权指令stac/clac只需8.6个时钟周期，比Intel MPK还要快。



Intel SMAP机制

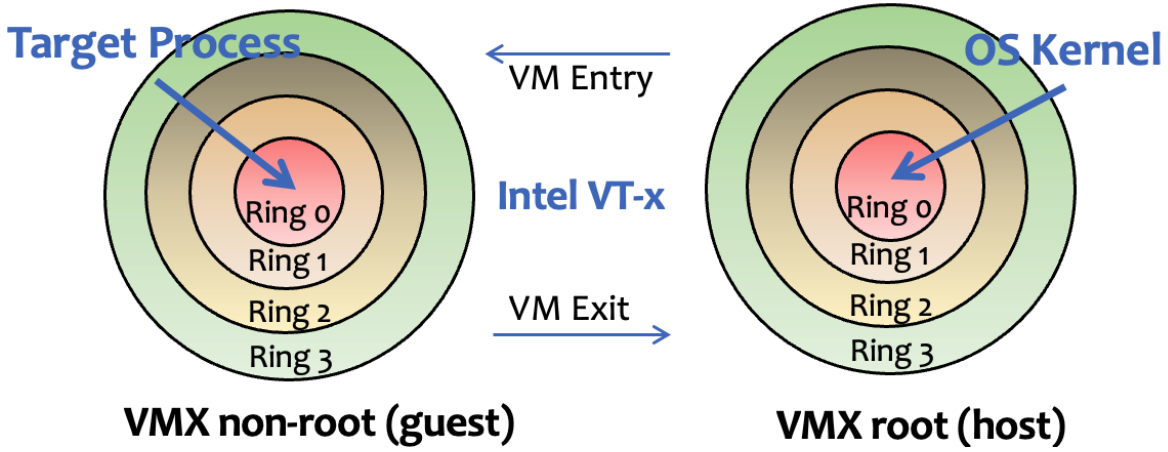
- 让进程直接运行在内核态，关键数据所在页面被设置成User-mode Page，其他页面设置为Supervisor-mode Page。
- 通过开关SMAP实现关键数据的隔离保护。

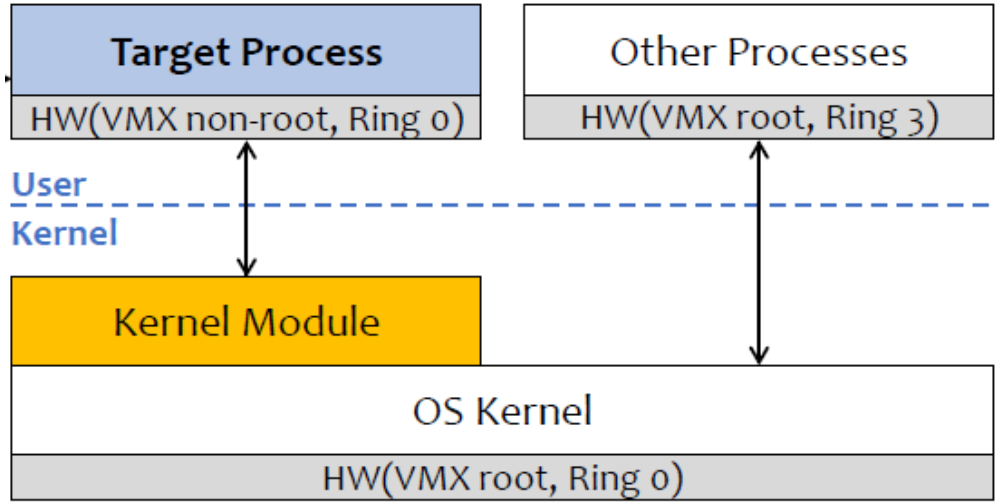


SEIMI内存布局

挖掘现有硬件潜力——Intel SMAP应用在进程内隔离

操作系统的保护：利用Intel VT-x技术激活硬件虚拟化，将目标进程运行在non-root模式的内核态，OS内核运行在root模式的内核态。

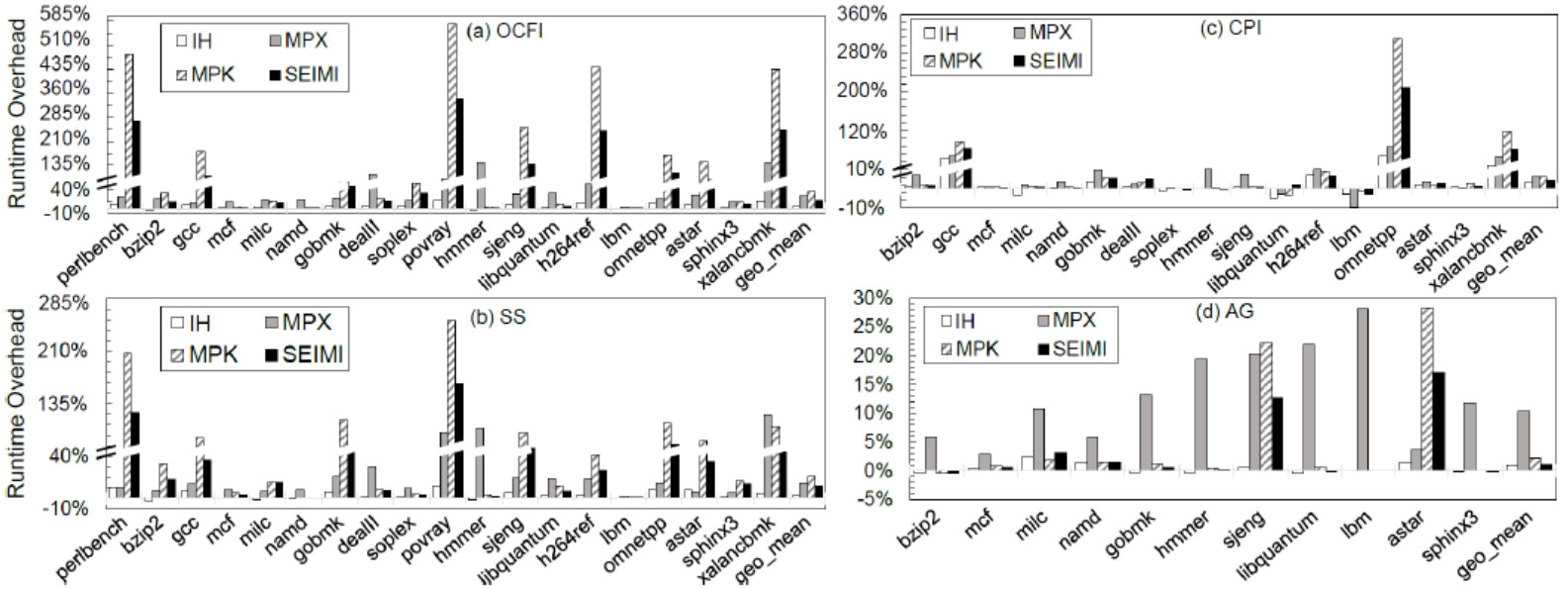




挖掘现有硬件潜力——Intel SMAP应用在进程内隔离

实验效果

- SPEC CPU 2006评测四种防护机制的关键数据隔离保护效率
  - SEIMI的平均性能开销低于MPX，平均降低33.97%。
  - SEIMI的性能开销均低于MPK，平均降低42.3%。

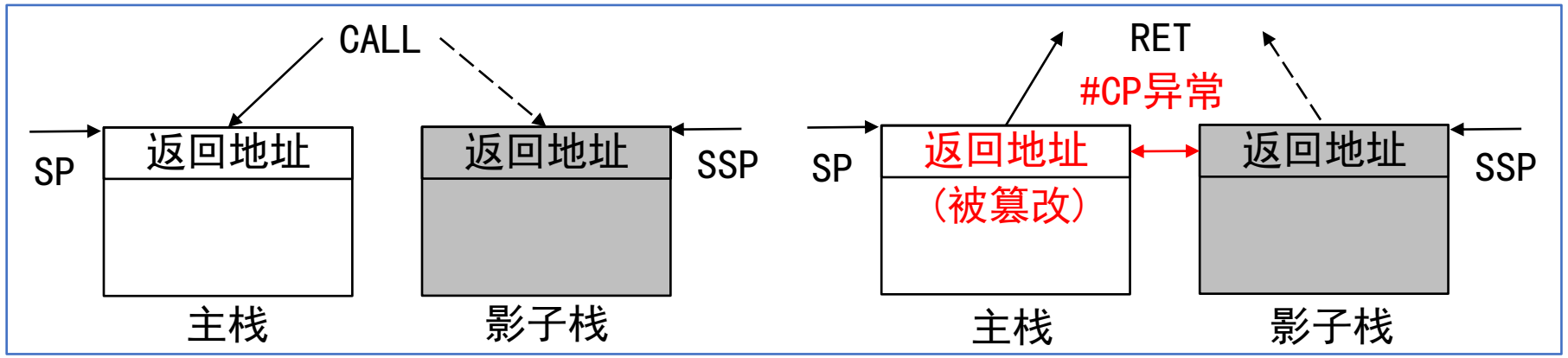


- 12款Real-world Applications评测（4个web server，4个database，4个JS引擎）
  - SEIMI在保护防护机制的关键数据时引入的性能开销明显低于MPX和MPK。

挖掘现有硬件潜力——Intel CET应用在进程内隔离

改造Intel CET的硬件影子栈用来隔离安全域

- Control-flow Enforcement Technology (CET)是Intel第十一代处理器为了阻止ROP攻击新增的硬件特性。CET中的SHSTK机制是一种硬件实现的影子栈（shadow stack）机制。
- 影子栈所在的页面被称为影子栈页（shstk page），普通的写指令不能修改其内容。
- 新增WRSSQ/WRSSD指令，分别将8字节/4字节内容写入影子栈页。

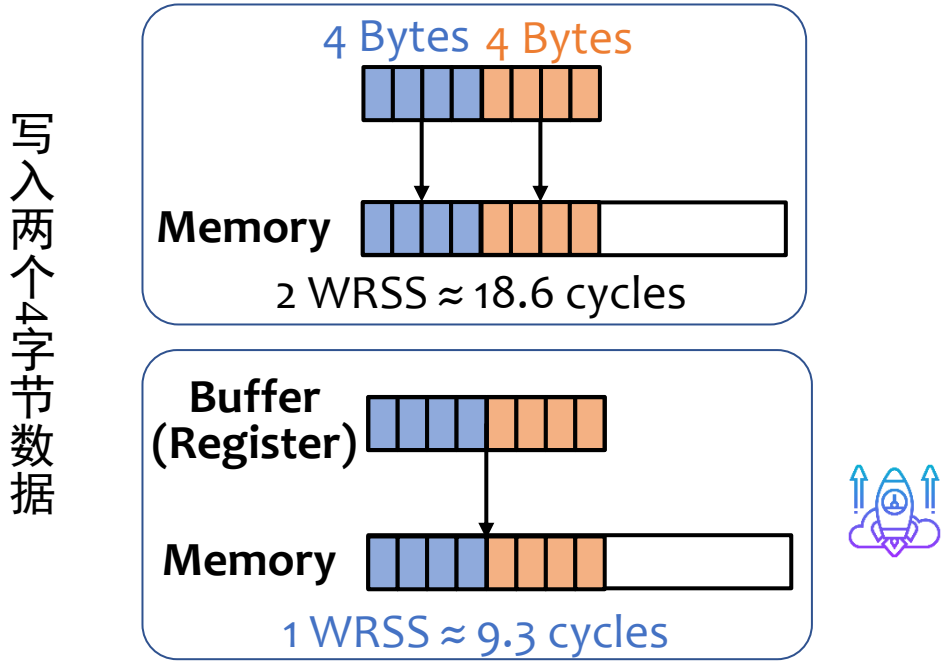


挖掘现有硬件潜力——Intel CET应用在进程内隔离

改造Intel CET的硬件影子栈用来隔离安全域

- **核心思想：**将影子栈上移，空出一个区域，用于存放敏感数据，利用CET保护机制对敏感数据进行隔离。
- **挑战：**WRSS指令有地址对齐和写入数据量固定的限制。

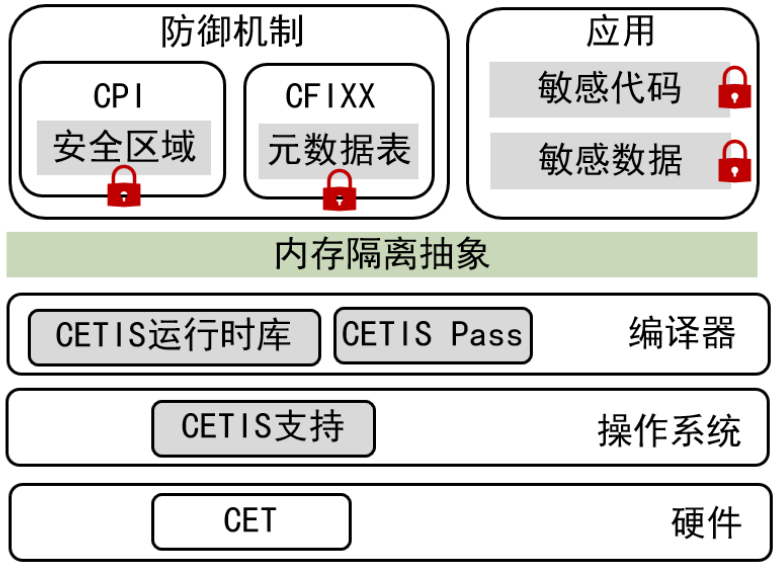
CETIS核心思想



挖掘现有硬件潜力——Intel CET应用在进程内隔离

CETIS架构

- CETIS提供了内存隔离抽象，支持对防御机制中关键数据以及应用中敏感数据/代码完整性的保护。



内存隔离抽象

- CETIS将敏感区域抽象为CETIS内存文件（CMFILE），提供APIs屏蔽了用户使用WRSS指令的限制。

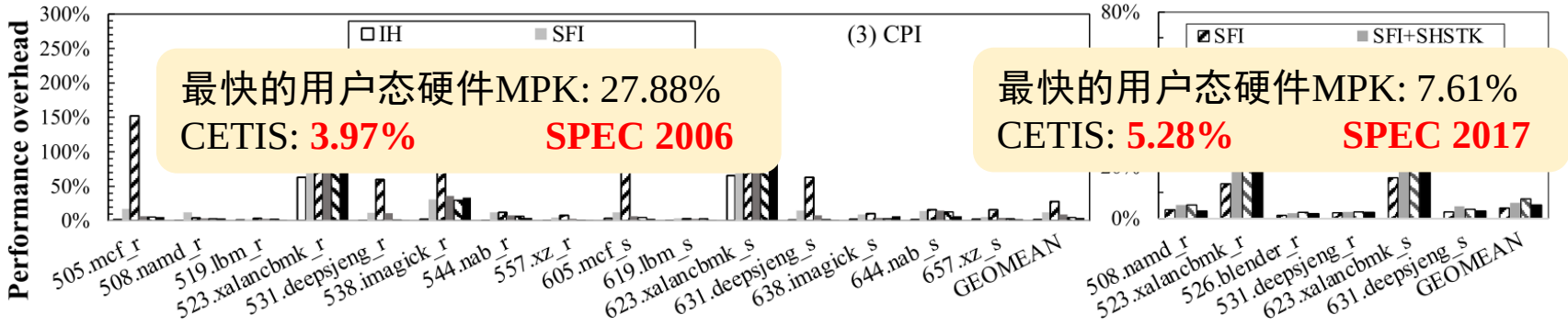
Cat.	APIs	Description	
1	int cetis_init (void) CMFILE *cmf_open (size_t len, bool is_exec) int cmf_close (CMFILE *cmfp)	CETIS initialization. Allocate a cmfile. Deallocate a cmfile.	Init.
2	FPI make_pos_ind (CMFILE *cmfp, off_t off) assume_pos_aligned (FPI fpi, size_t align) int cmf_write (FPI fpi, void *src, size_t len) int cmf_read (FPI fpi, void *dst, size_t len)	Construct a position indicator. Alignment hint for compiler. Write data without buffer. Read data without buffer.	read/write mode
3	int cmf_append_byte (uchar val) int cmf_append_word (ushort val) int cmf_append_dword (uint val) int cmf_append_qword (ulong val)	Append val w/ buffer at the append position of the current cmfile.	append mode
	int set_curr_append_pos (off_t off)	Set the current append position.	
	int cmf_sync_buf (bool is_flush)	If the argument is true, flush the buffer to make its content global visible; otherwise, reload the buffer.	
	int set_curr_cmf (CMFILE *cmfp)	Switch cmfile.	
	CMFILE *get_curr_cmf (void)	Obtain the current cmfile pointer.	

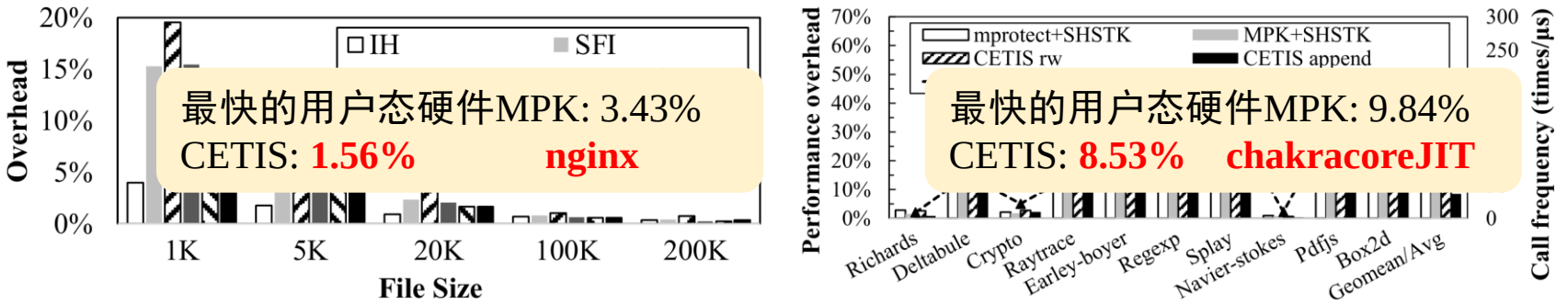
封装指令解决地址对齐

缓存数据解决性能问题

挖掘现有硬件潜力——Intel CET应用在进程内隔离

- 实验结果：性能开销低于基于MPK的域隔离方法
  - CETIS保护网络服务器Nginx的性能开销不到2%
  - SPEC CPU2017性能开销低于5.5%
  - 保护JIT编译器中的代码缓存，性能开销低于基于MPK的隔离方法





挖掘现有硬件潜力

X86硬件和应用场景

Intel MPK	用户态隔离、内核态隔离
Intel VMFunc	用户态隔离、内核态隔离
Intel WP	内核态隔离
Intel SMAP	**用户态隔离（S&P 2020）**
Intel CET	**用户态隔离（CCS 2022）**

ARM硬件和应用场景

ARM Memory Domain	用户态隔离
ARM Watchpoint	用户态隔离
ARM TxSZ	内核态隔离
ARM TTBRx/ASID	内核态隔离
ARM PAN/LSU	**用户态隔离（CCS 2023）**

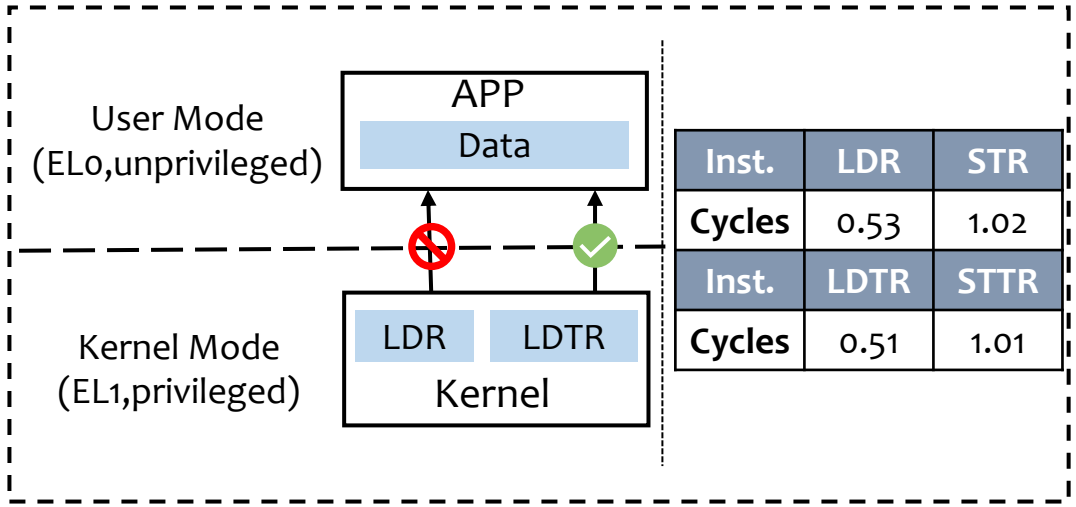
挖掘现有硬件潜力——ARM PAN应用在进程内隔离

利用特权态硬件PAN实现用户进程内内存隔离机制

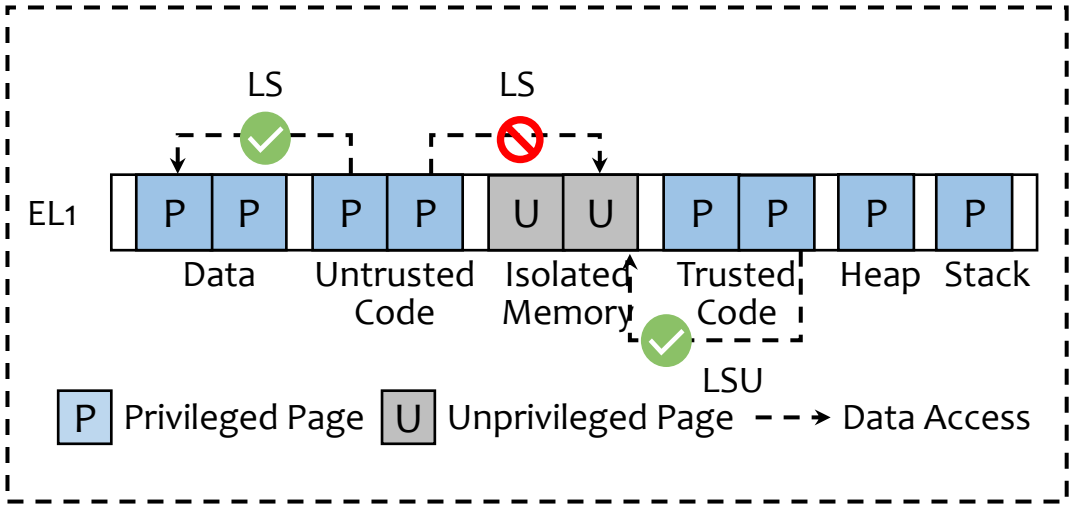
Privileged Access Never(PAN) + load/store unprivileged(LSU)

严格的隔离机制

高效的隔离内存访问指令



Inst.	LDR	STR
Cycles	0.53	1.02
Inst.	LDTR	STTR
Cycles	0.51	1.01



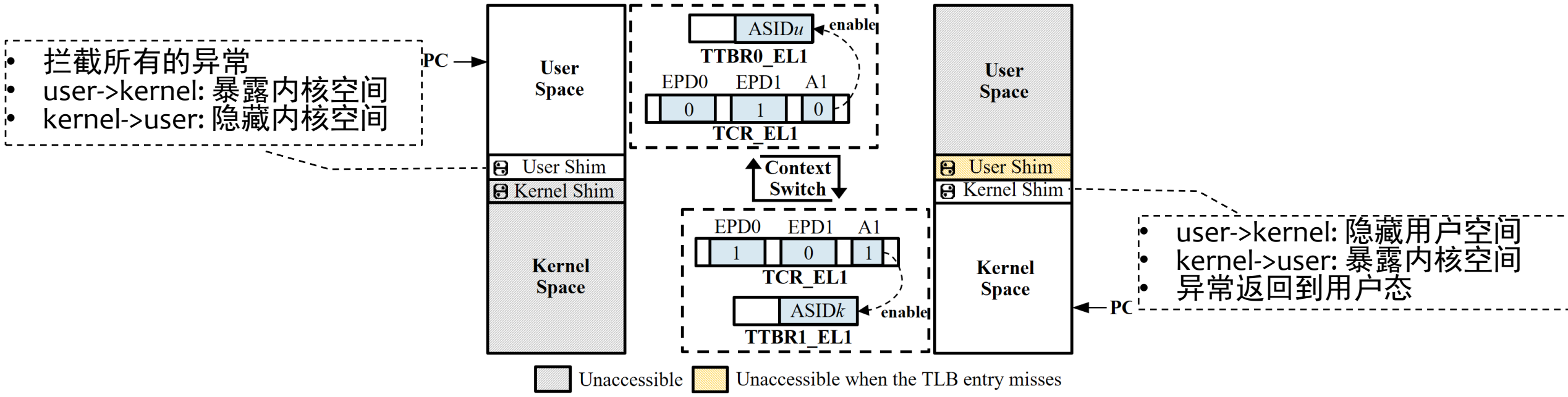
硬件特性：开启PAN并关闭UAO，访存指令的行为

PANIC基本思路

挖掘现有硬件潜力——ARM PAN应用在进程内隔离

阻止特权态进程访问内核

EPDn + separated ASID 可以打开/关闭用户或内核空间的访问权限

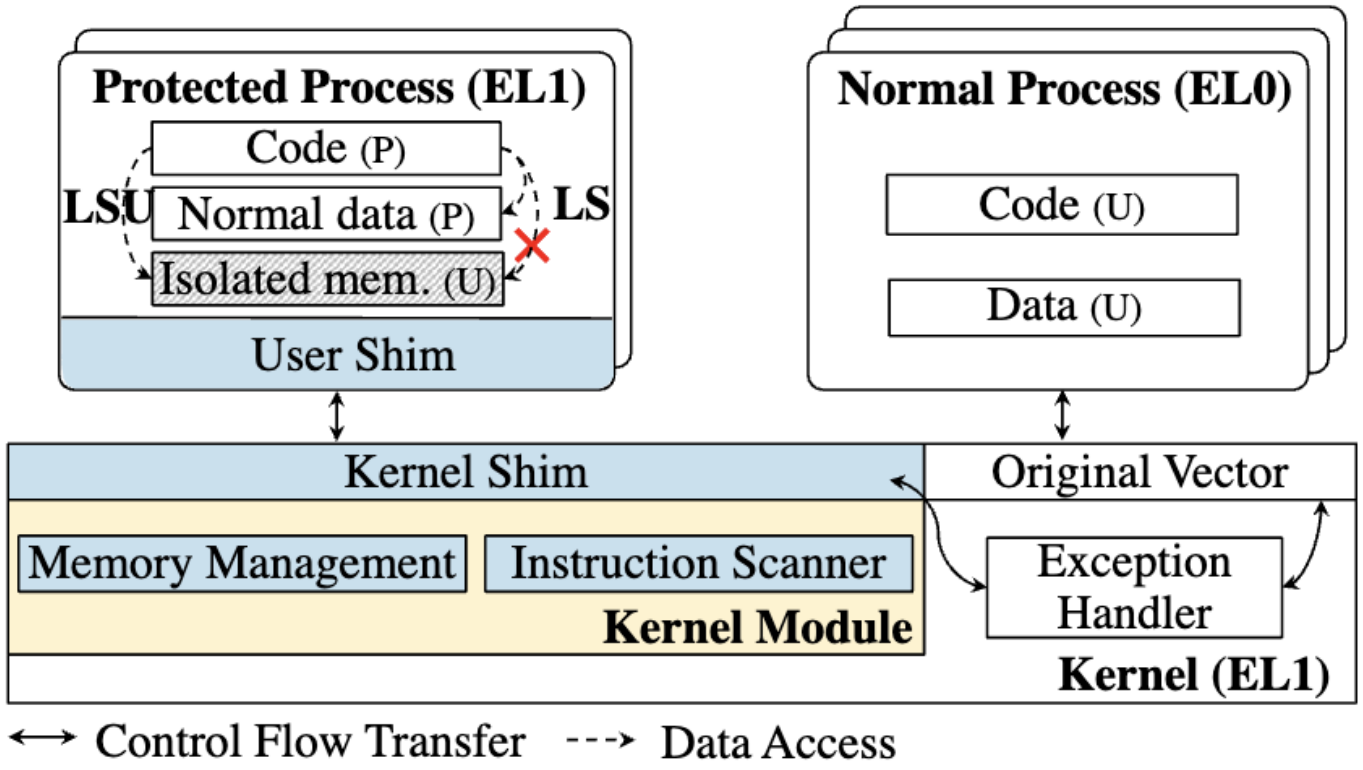


- **TCR_EL1.EPDn: 开启或关闭指定空间页表翻译过程**
  例如, EPD1=1意味着内核空间的页表翻译被关闭了，内核空间将无法通过虚拟地址访问

- **TCR_EL1.A1: 选择缓存在TLB中的working ASID**
  例如 A1=1，代表将TTBR1.EL1中的ASID域值缓存在TLB中

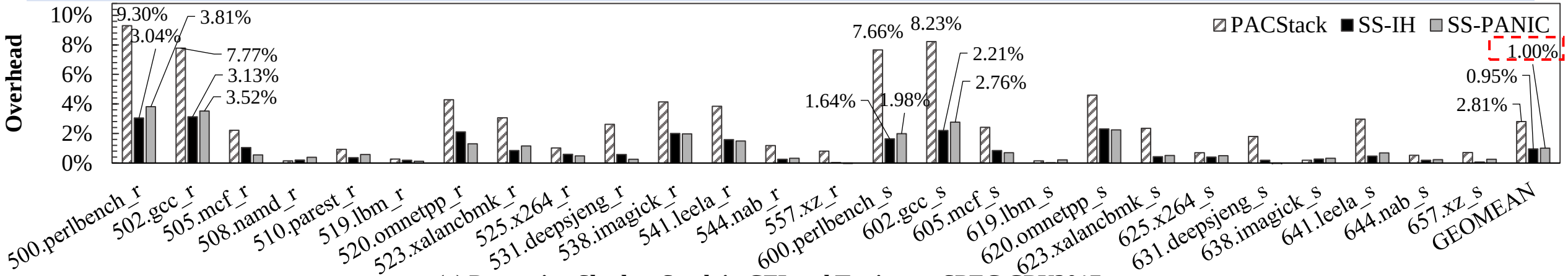
挖掘现有硬件潜力——ARM PAN应用在进程内隔离

PANIC系统架构：用户只需要装载PANIC内核模块并指定受保护进程

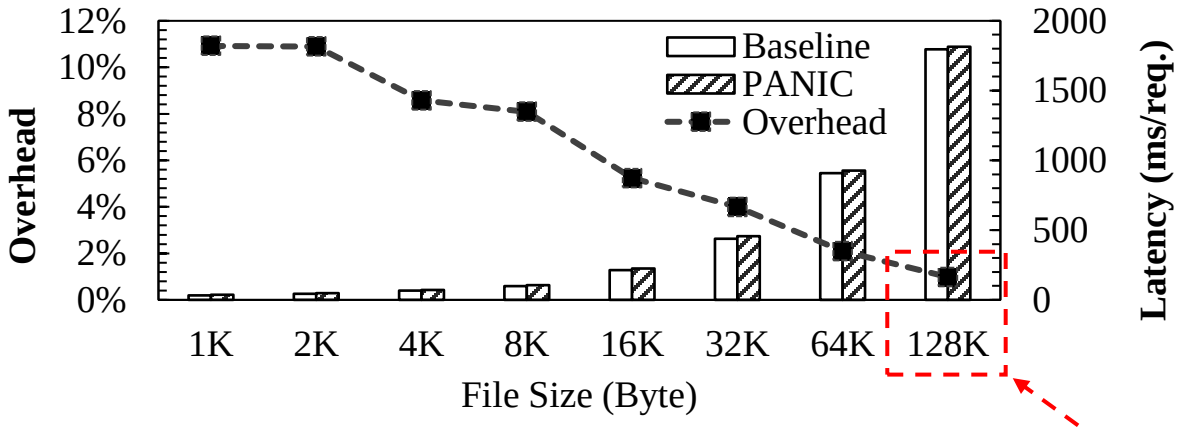


挖掘现有硬件潜力——ARM PAN应用在进程内隔离

实验评估: 在影子栈、OpenSSL的key和JITed Code的保护上，PANIC所引入的开销很低

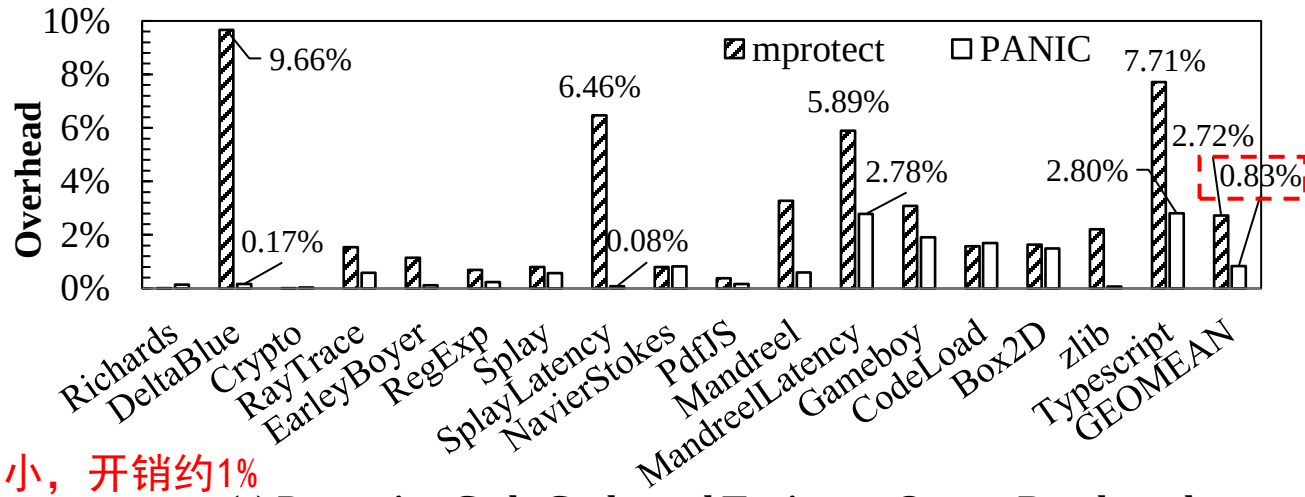


(a) Protecting Shadow Stack in CFI and Testing on SPEC CPU2017



(b) Protecting Session Keys in Nginx+OpenSSL

常用文件大小，开销约1%



(c) Protecting Code Cache and Testing on Octane Benchmark

谢谢

电子邮件

wucg@ict.ac.n

微信

中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES