Trust Under Siege: Exploiting and Mitigating Interface-Based Attacks on TEEs

Jo Van Bulck

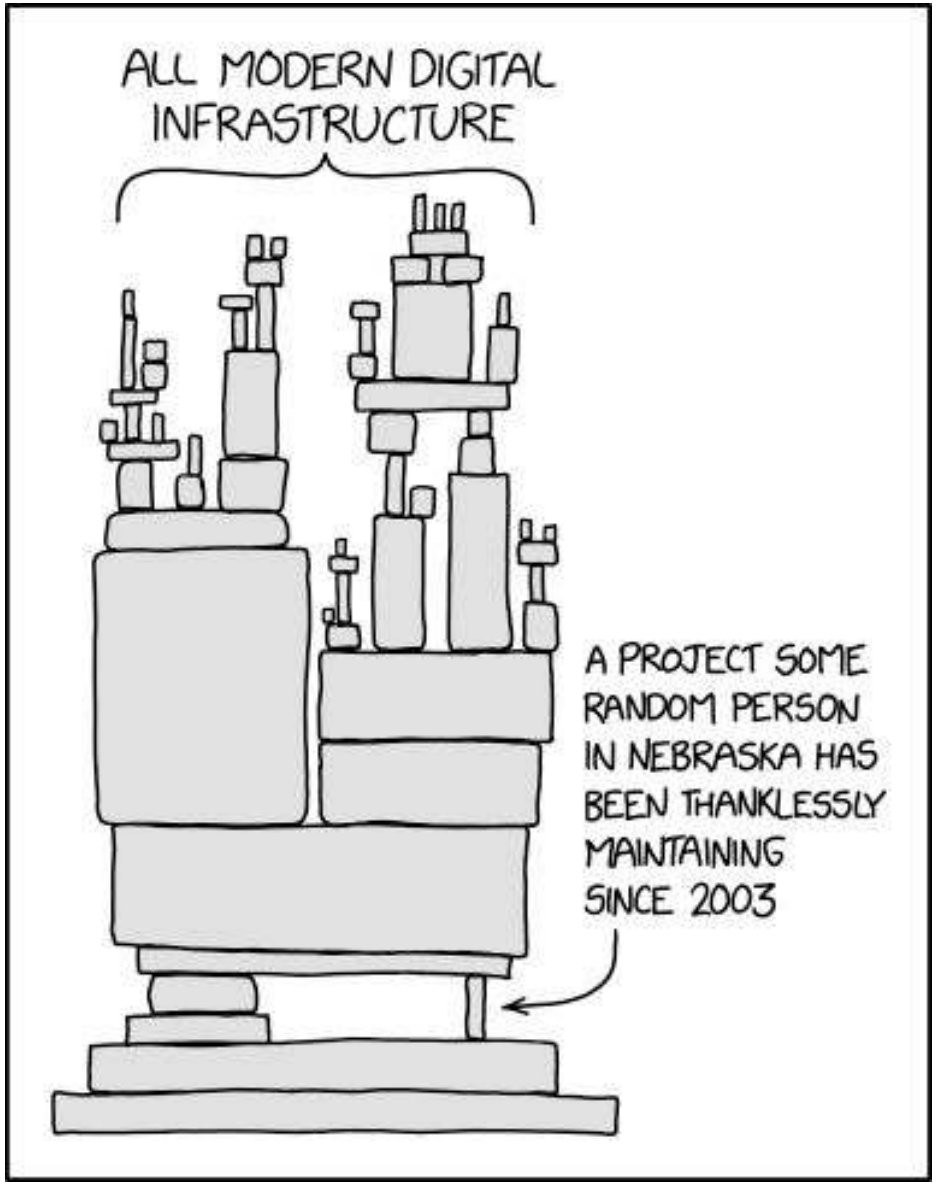
DistriNet, KU Leuven, Belgium jo.vanbulck@cs.kuleuven.be vanbulck.net

Graz Security Week, Sept 4, 2025

DistriNet

KU LEUVEN

Trust?

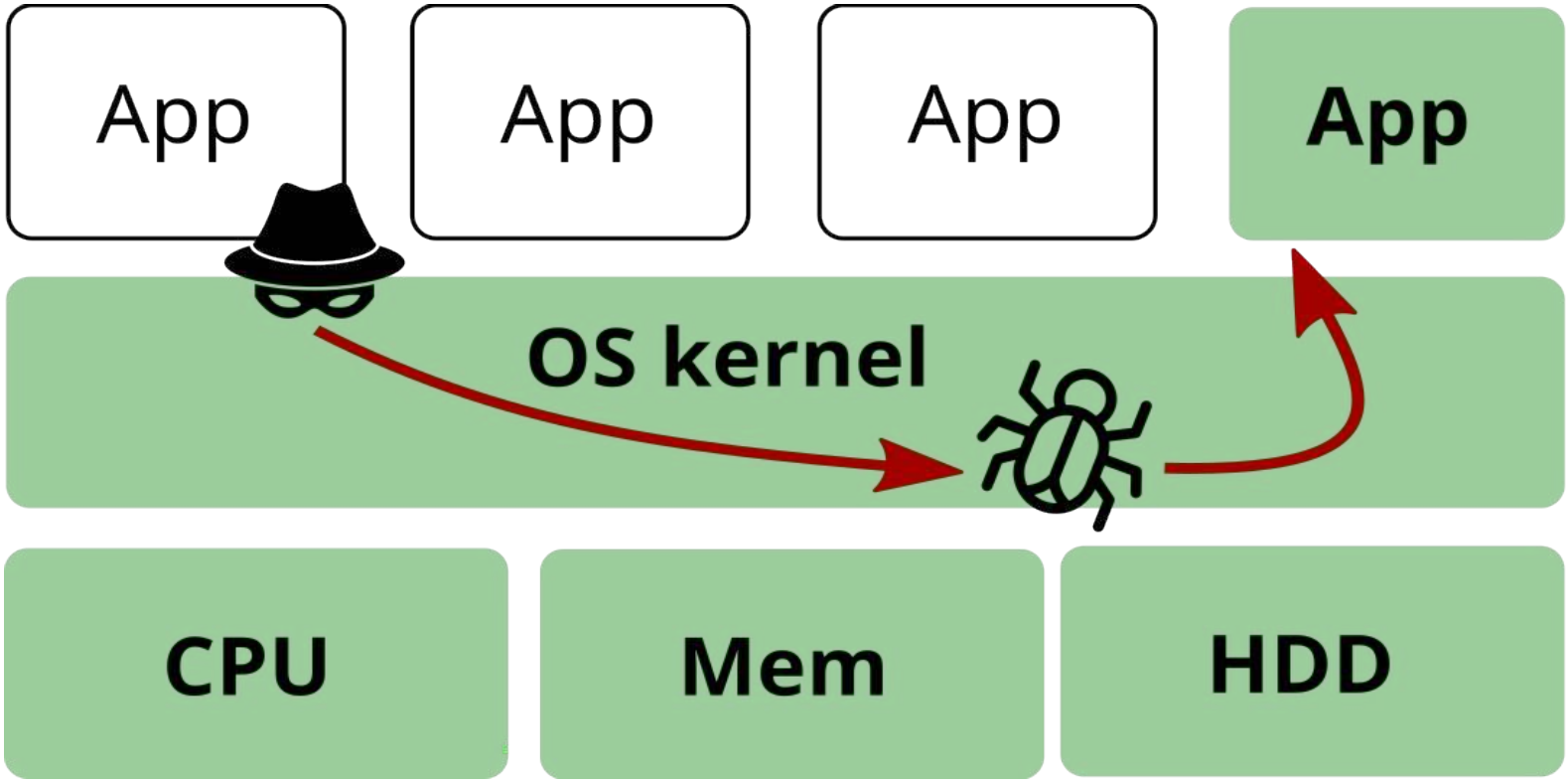


https://xkcd.com/2347/

Complexity?

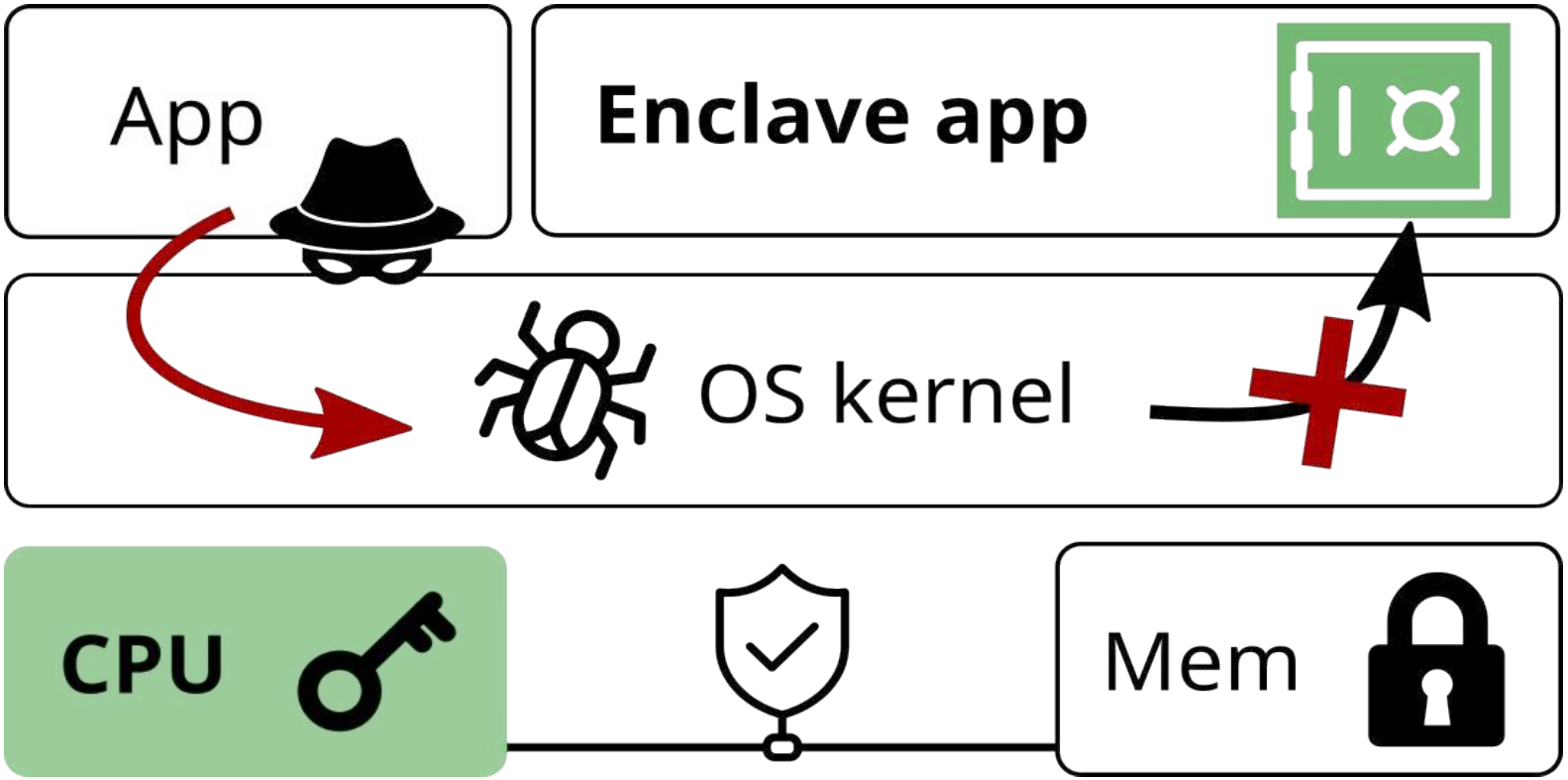
https://xkcd.com/676/

Confidential Computing: Reducing Attack Surface



Traditional layered designs: Large trusted computing base

Confidential Computing: Reducing Attack Surface



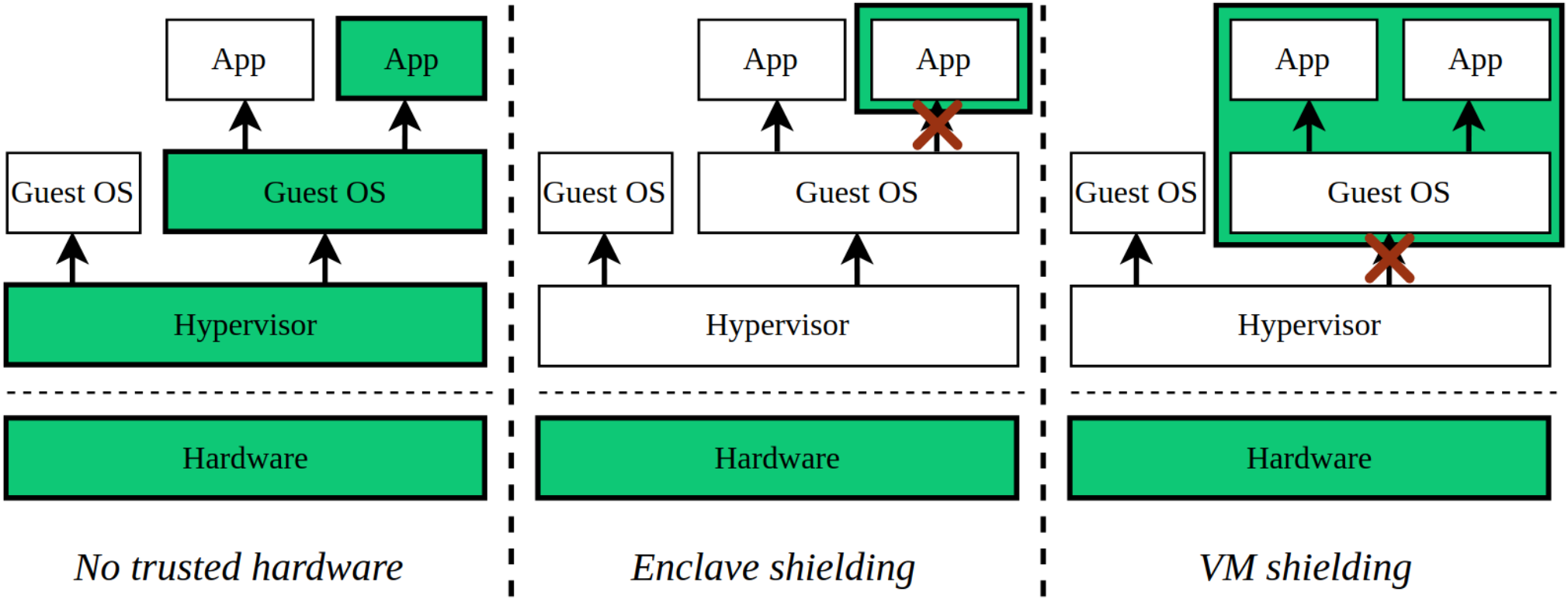
Trusted execution: Hardware-level isolation and attestation

The Rise of Trusted Execution Environments (TEEs)

- 2004: ARM TrustZone
- 2015: **Intel Software Guard Extensions (SGX)**
- 2016: AMD Secure Encrypted Virtualization (SEV)
- 2018: IBM Protected Execution Facility (PEF)
- 2020: AMD SEV with Secure Nested Paging (SEV-SNP)
- 2022: Intel Trust Domain Extensions (TDX)
- 2023: ARM Confidential Compute Architecture (CCA)
- 2024: NVIDIA Confidential Computing

TEEs are here to stay...

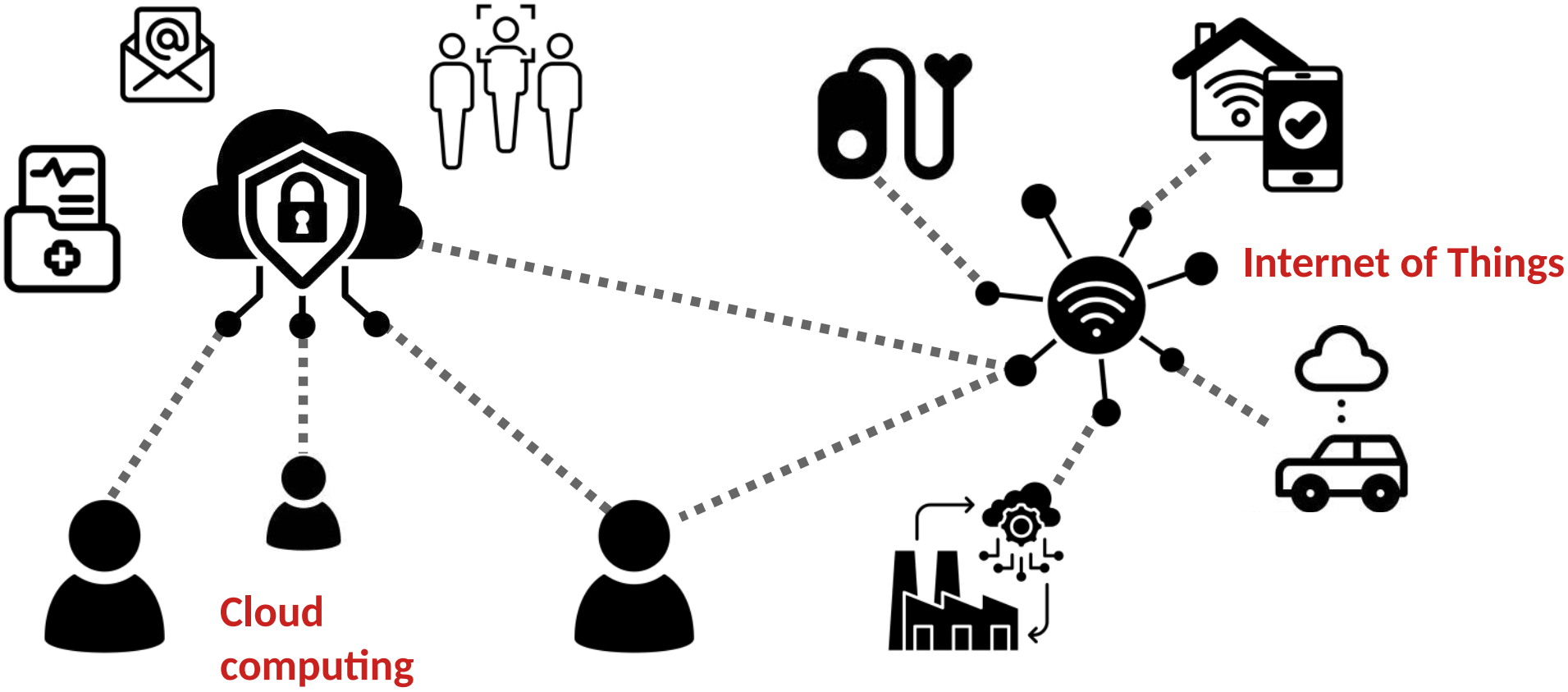
Confidential Computing Isolation Paradigms



Intel SGX, Arm TrustZone-M

AMD SEV, Intel TDX, Arm CCA
Arm TrustZone-A

“Confidential Computing Today, Just Computing Tomorrow” *



* *Mark Russinovich, CTO Microsoft Azure*

Motivation: Why Research TEE/SGX Security?

intel

Overview | About Intel | News & Events | Financial Info | Stock Info | Filings & Reports | Board & Governance | ESG

Overview | Press Releases | IR Calendar | Annual Stockholders' Meeting | Investor Meeting | Email Alerts | Presentations

Data Protection across the Compute Stack

Technologies such as disk- and network-traffic encryption protect data in storage and during transmission, but data can be vulnerable to interception and tampering while in use in memory. "Confidential computing" is a rapidly emerging usage category that protects data while it is in use in a Trusted Execution Environment (TEE). Intel SGX is the most researched, updated and battle-tested TEE for data center confidential computing, with the smallest attack surface within the system. It enables application isolation in private memory regions, called enclaves, to help protect up to 1 terabyte of code and data while in use.

https://www.intc.com/news-events/press-releases/detail/1423/intel-xeon-scalable-platform-built-for-most-sensitive

TEE Attack Research Leads the Way . . .

TEE Attack Research Leads the Way . . .

- Privileged TEE attacker models sets the bar!
- Idealized execution environment for attack research
- **Generalizations:** e.g., Foreshadow-NG, branch prediction, address translation, etc.

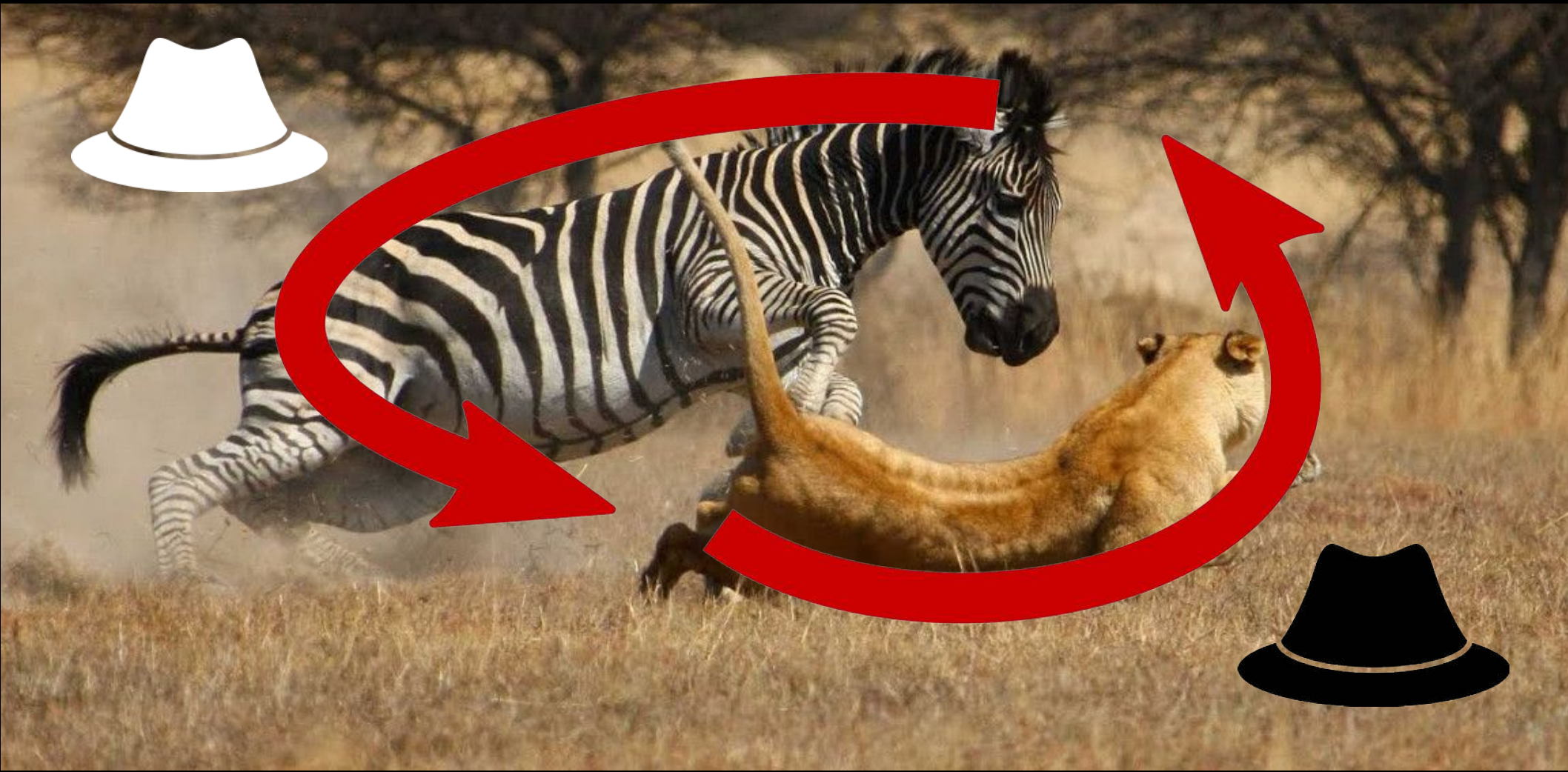
How Trusted Execution Environments Fuel Research on Microarchitectural Attacks

Michael Schwarz and Daniel Gruss | Graz University of Technology

https://ieeexplore.ieee.org/document/9107096

Scientific Understanding Driven by Attacker-Defender Race...

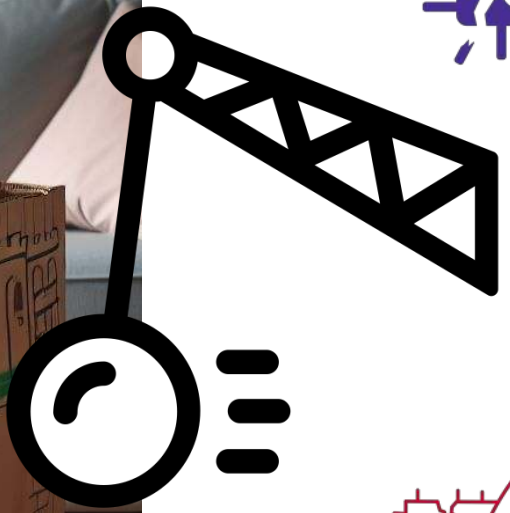
Scientific Understanding Driven by Attacker-Defender Race...



Trust under Siege?

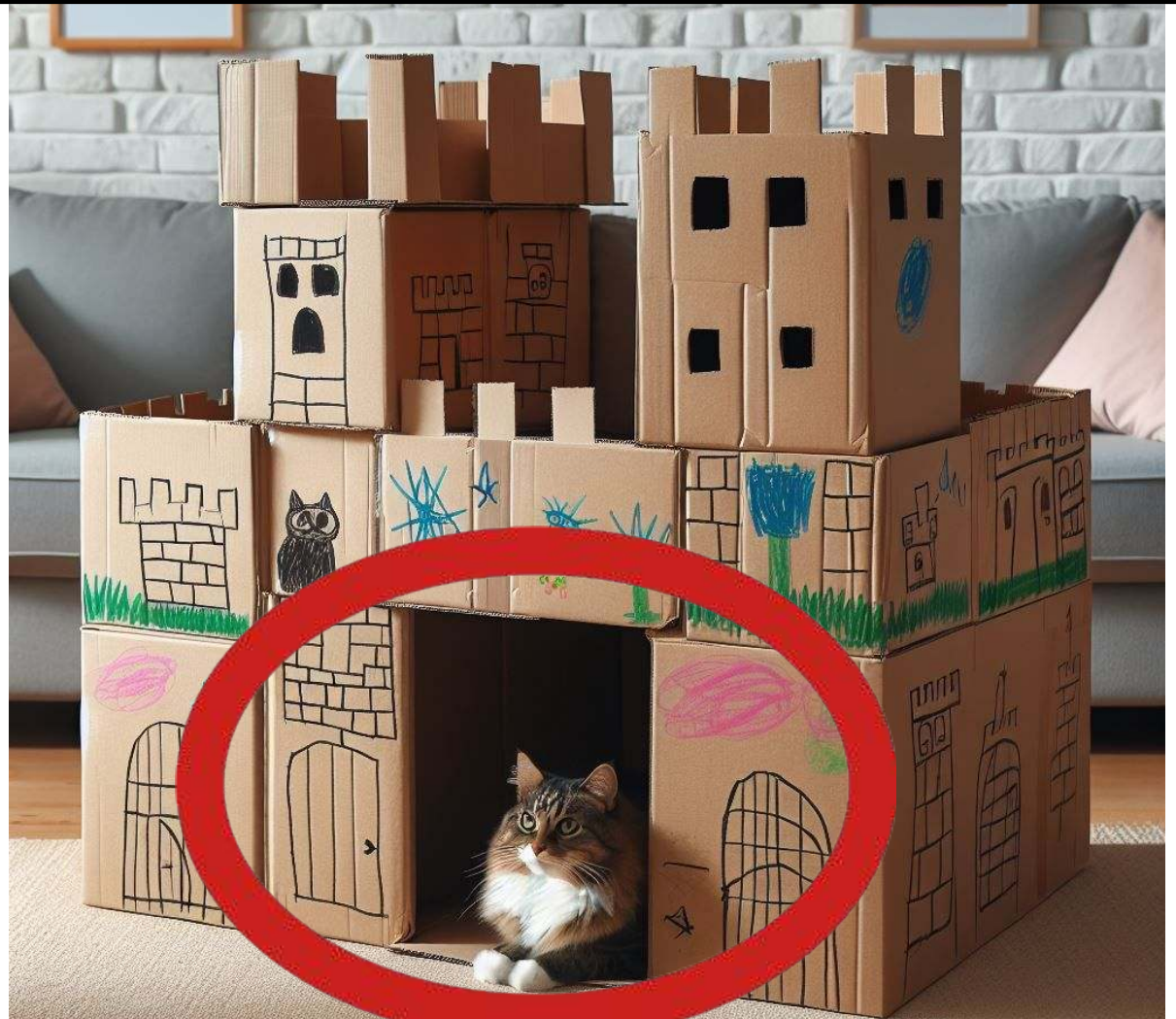
DALL·E 3

Trust under Siege: Transient-Execution Attacks (Not Today)



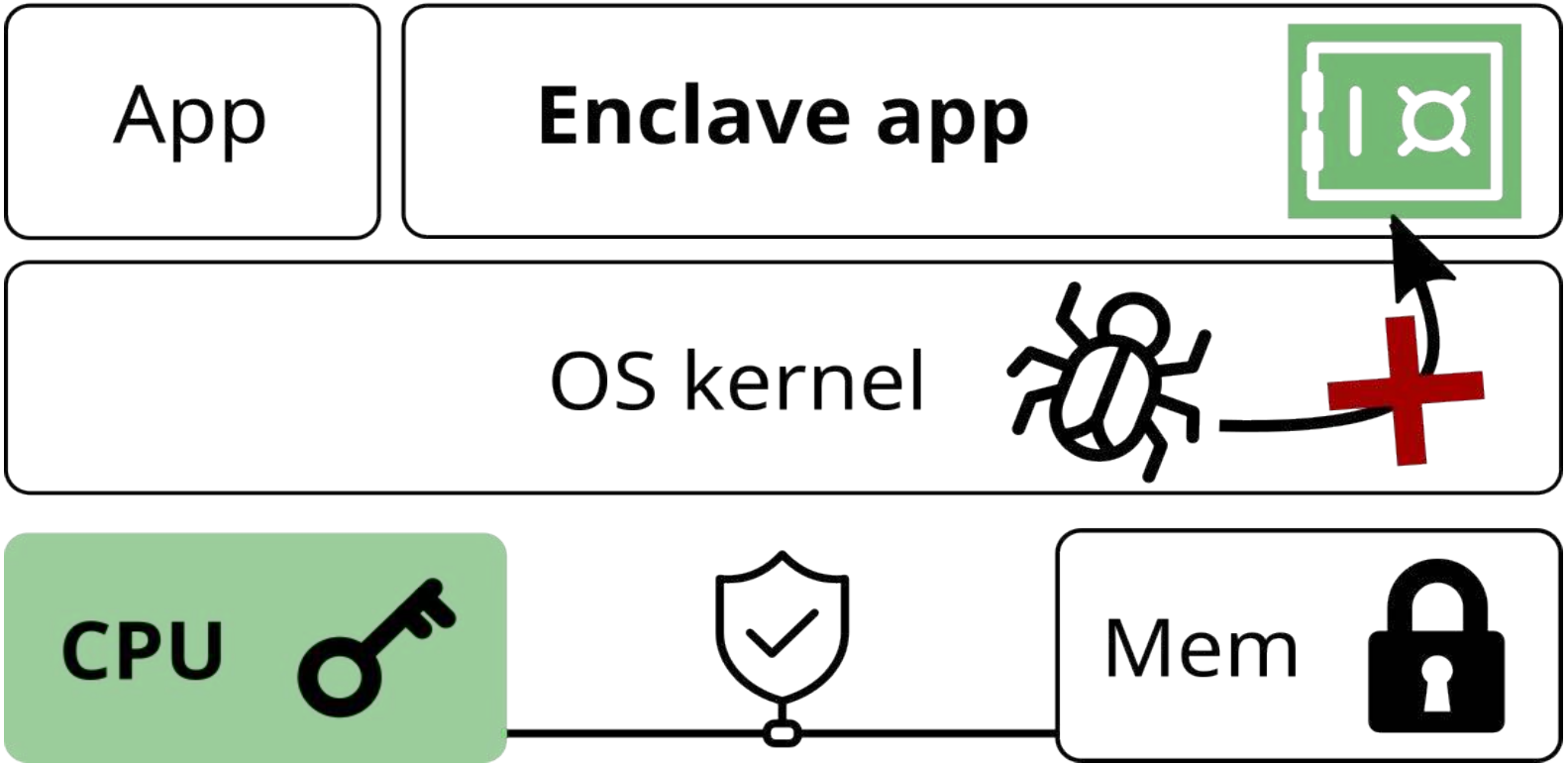
DALL·E 3

Trust under Siege: Interface-Based Attacks on TEEs (Today)

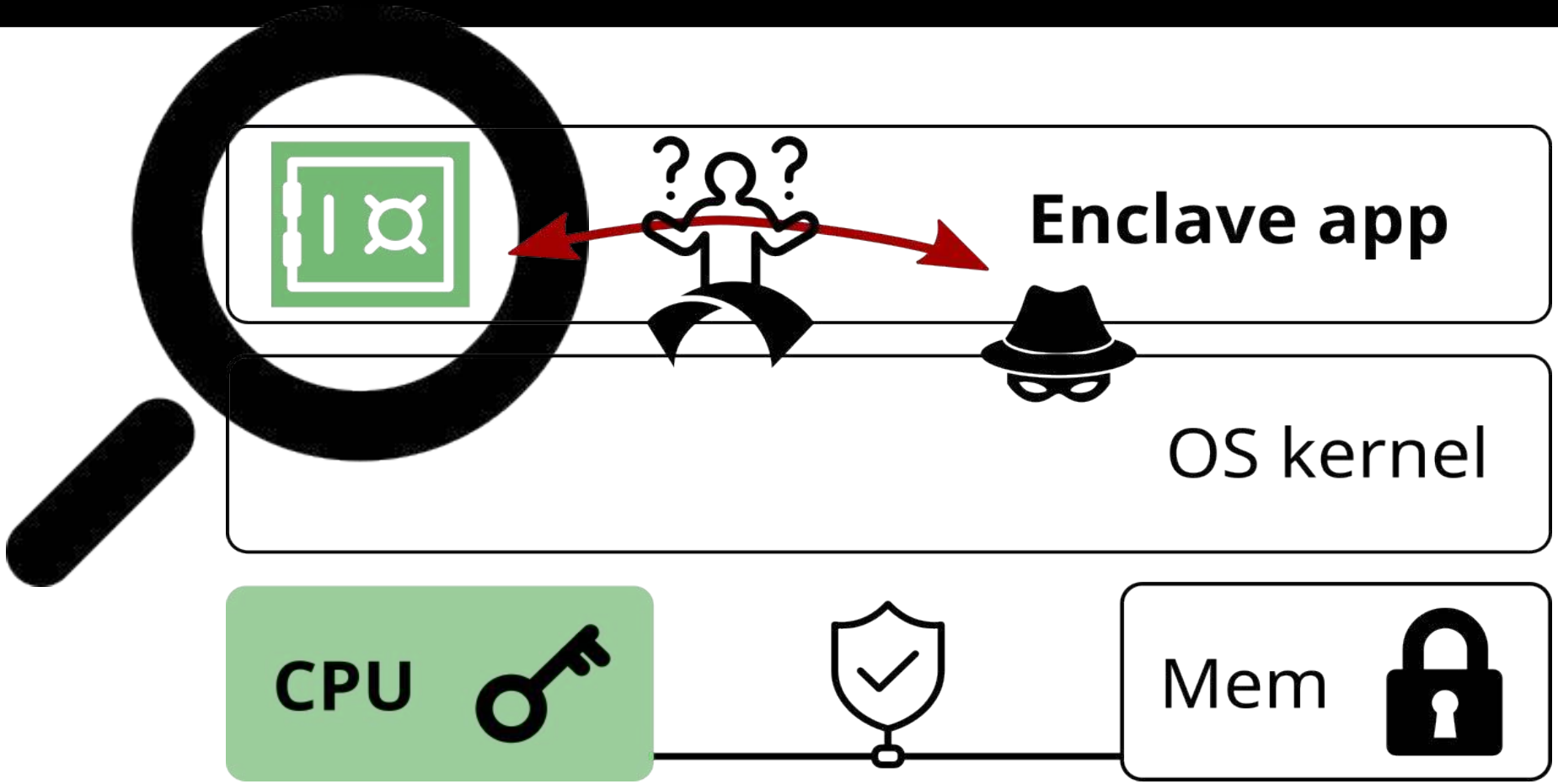


DALL·E 3

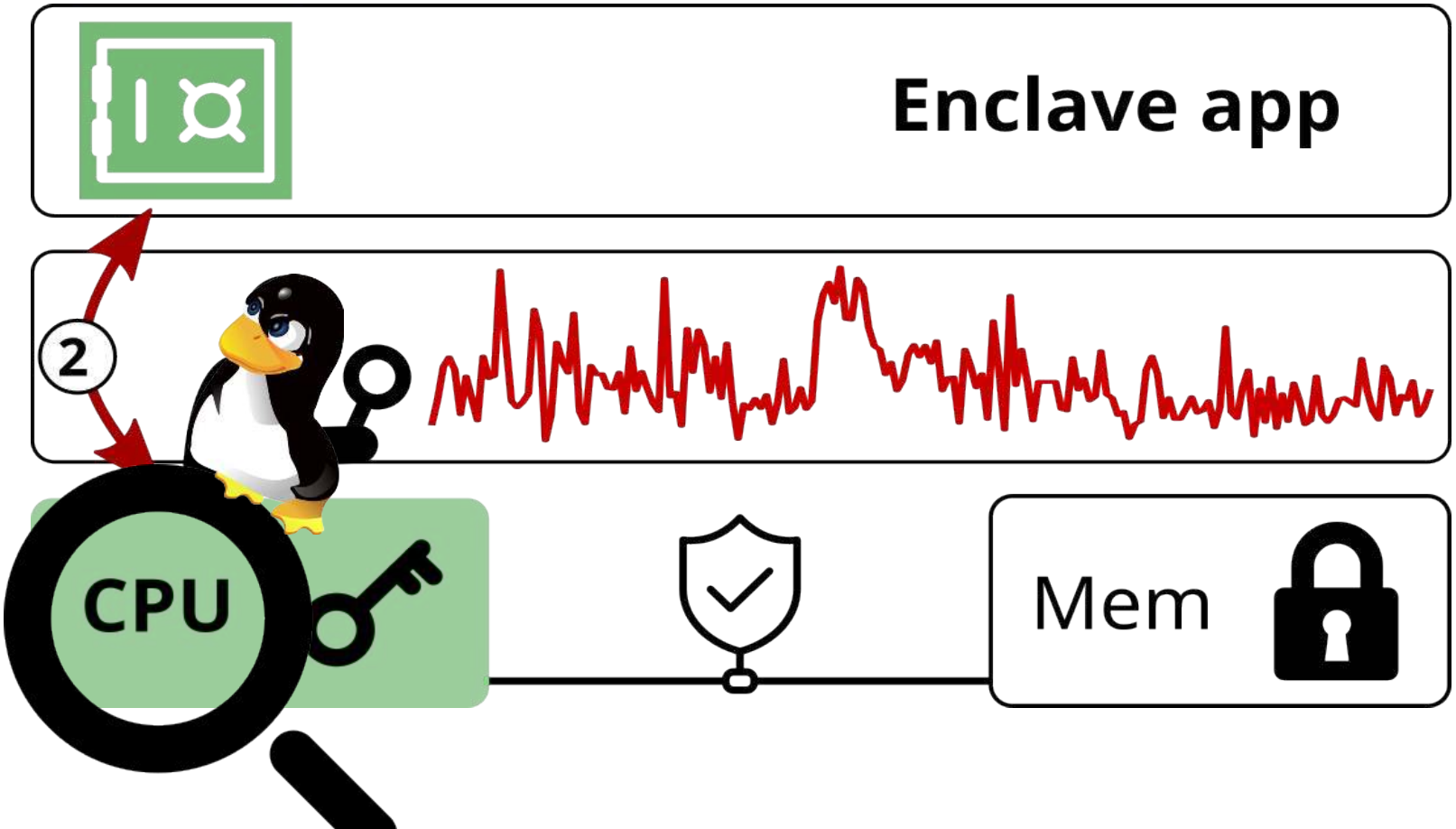
Lecture Overview: Interface-Based Attacks?



Lecture Overview: Enclave Software Interface *(Part 1)*

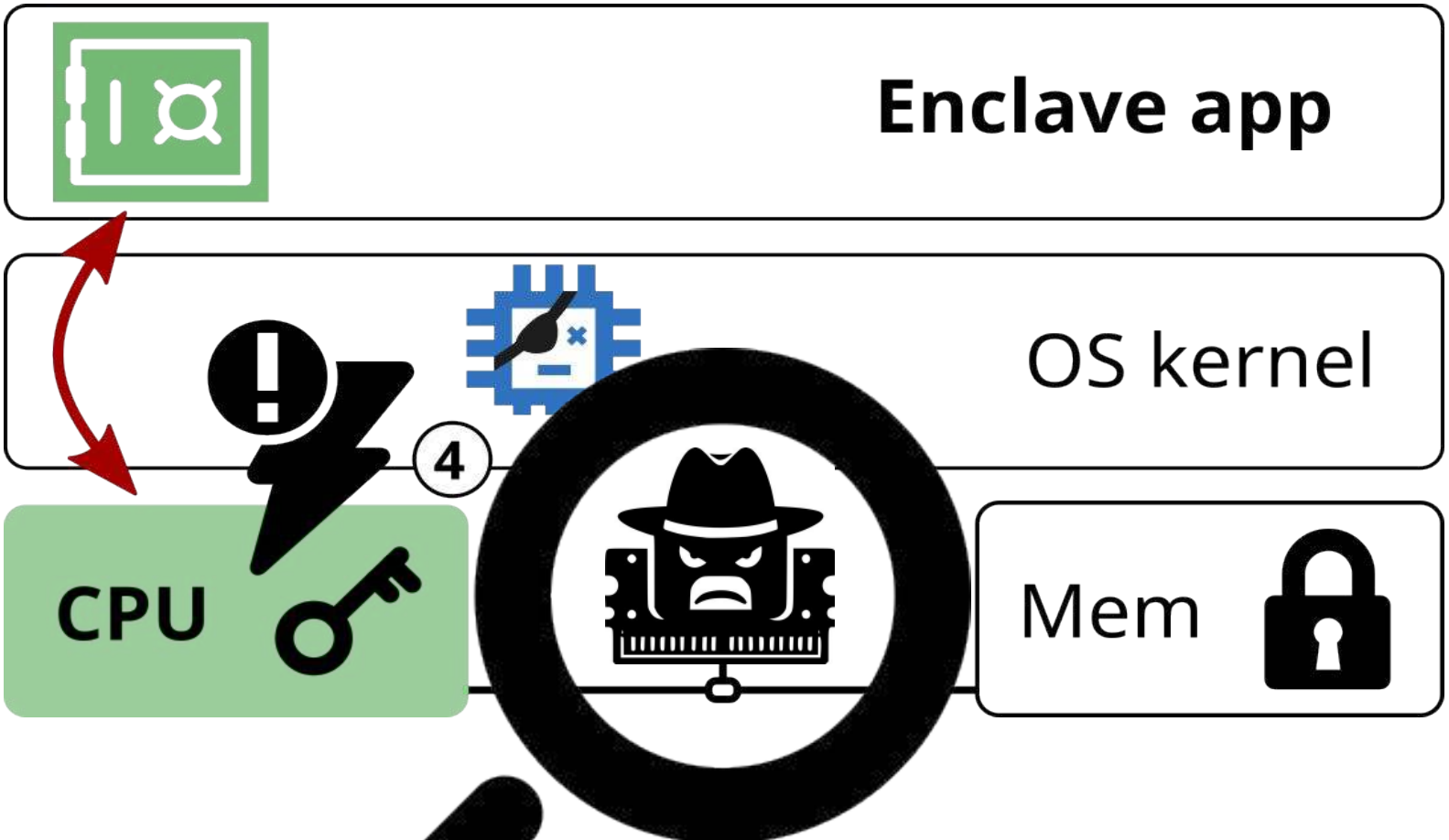


Lecture Overview: Privileged Software Interface *(Part 2)*



Not Today: Hardware Interface

(See David Oswald, Friday!)



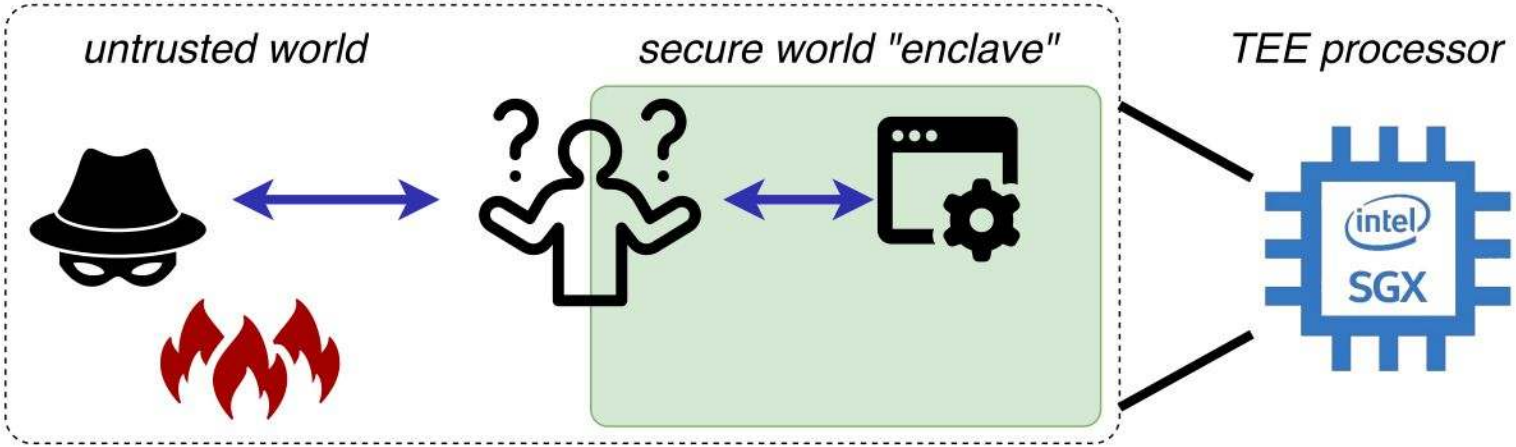
Part #1: Enclave Software Interface

Context: Writing “Secure” Enclave Software is Hard…

Improper sanitization of MXCSR and RFLAGS GHSA-5gfr-m6mx-p5w4 published on Jul 17, 2023 by radhikaj	Moderate
Intel Processor Stale Data Read from Legacy xAPIC GHSA-v3vm-9h66-wm76 published on Aug 13, 2022 by radhikaj	Moderate
Intel Processor MMIO Stale Data Vulnerabilities GHSA-wm9w-8857-8fgj published on Jun 14, 2022 by radhikaj	Moderate
Open Enclave SDK Elevation of Privilege Vulnerability GHSA-mj87-466f-jq42 published on Jul 13, 2021 by radhikaj	Moderate
Socket syscalls can leak enclave memory contents GHSA-525h-wxcc-f66m published on Oct 12, 2020 by radhikaj	Moderate
x87 FPU operations in enclaves are vulnerable to ABI poisoning GHSA-7wjx-wcwg-w999 published on Jul 14, 2020 by CodeMonkeyLeet	Low
Intel SGX Load Value Injection (LVI) vulnerability GHSA-8934-g2pr-x6cg published on Mar 12, 2020 by radhikaj	Moderate
Enclave heap memory disclosure vulnerability GHSA-mg2p-657r-46cj published on Oct 8, 2019 by CodeMonkeyLeet	Moderate

https://github.com/openenclave/openenclave/security

Why Isolation is Not Enough...



- TEE promise: enclave == "secure oasis" in a **hostile environment**
- ...but application writers and compilers are largely unaware of **isolation boundaries**

01 INTEL OPEN SOURCE .org

PROJECTS 01 COMMUNITY ABOUT

Intel® Software Guard Extensions

INTEL® SOFTWARE GUARD EXTENSIONS SDK FOR LINUX*

Open Enclave SDK

https://openenclave.io/sdk/

Open Enclave SDK

Build Trusted Execution Environment ba
with an open source SDK that provides
technologies as well as all platforms fror

Enarx | Enarx

https://enarx.dev

Enarx

Star 476

Enarx

WebAssembly + Confidential Computing

Enarx Introduction - 10min

GRAMINE

Gramine - a Library OS for Unmodified Applications

Open-Source community project driven by a core team of contributors.
Previously Graphene

SGX-LKL | LSDS - Large-Sc

https://lsds.doc

LSDS

Large-Scale Data & Systems Group

SGX-LKL: Linux Binaries in SGX Enclaves

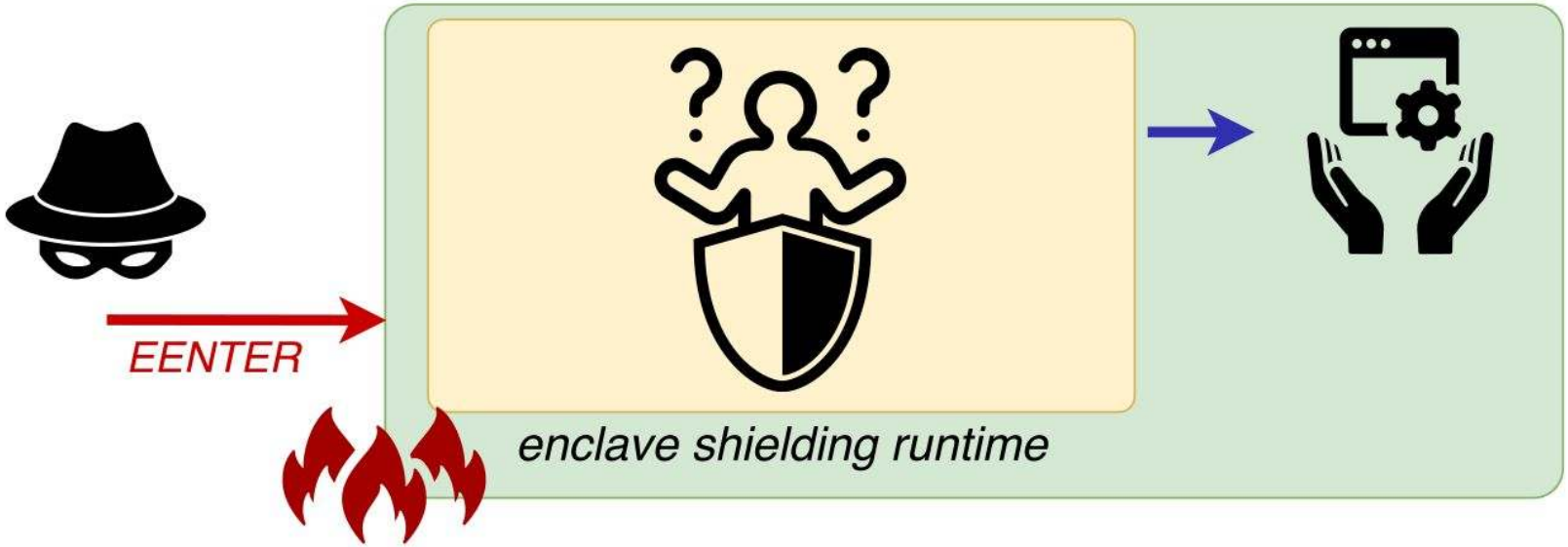
Fortanix EDP

ENCLAVE DEVELOPMENT PLATFORM

The Fortanix EDP is the preferred way for writing Intel® SGX applications from scratch.

Enclave Shielding Responsibilities

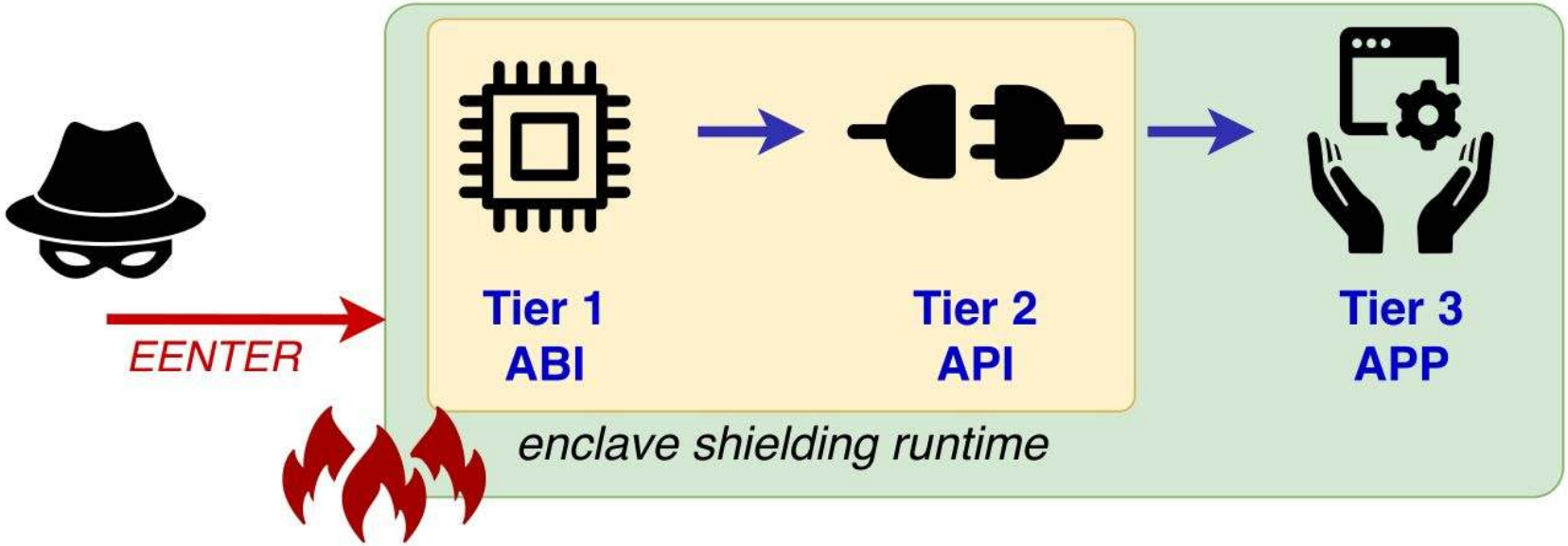
Key questions: how to securely bootstrap from the untrusted world to the enclaved application binary (and back)? Which sanitizations to apply?



Van Bulck et al. “A Tale of Two Worlds: Assessing the Vulnerability of Enclave Shielding Runtimes”, CCS 2019.

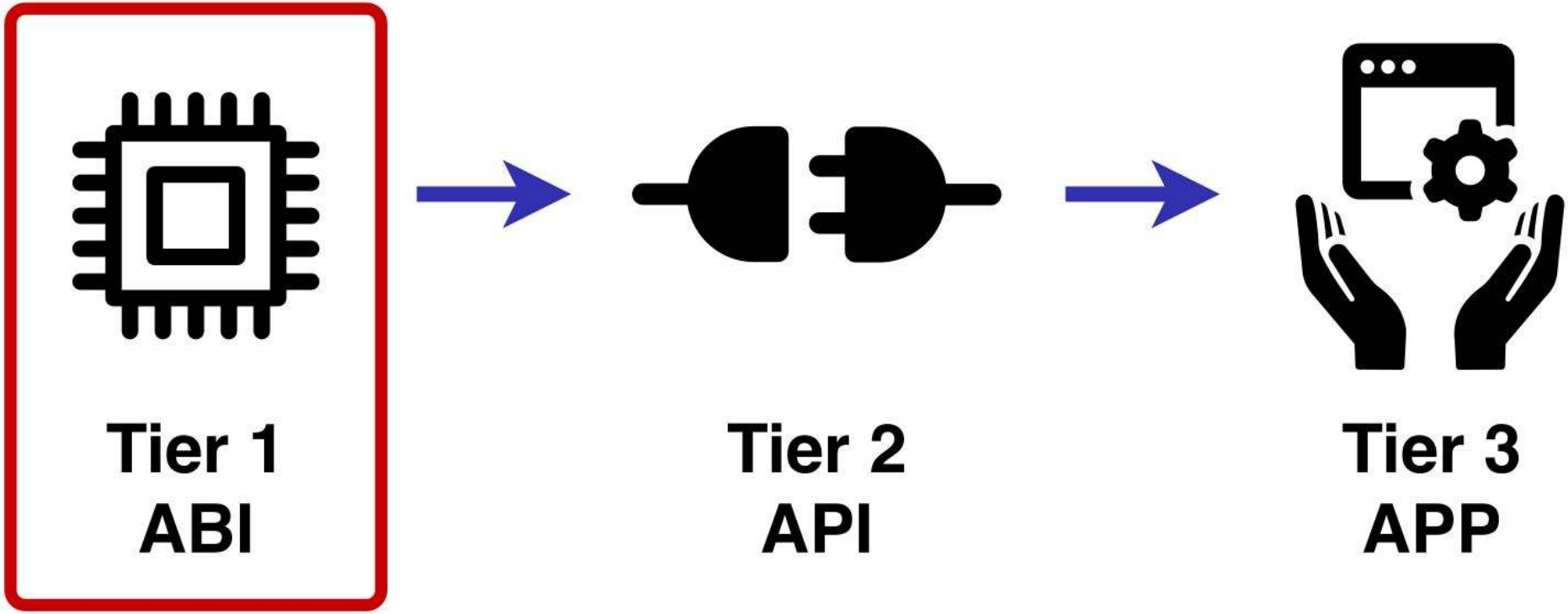
Enclave Shielding Responsibilities

Key insight: split sanitization responsibilities across the ABI and API tiers: *machine state* vs. higher-level *programming language interface*



Van Bulck et al. "A Tale of Two Worlds: Assessing the Vulnerability of Enclave Shielding Runtimes", CCS 2019.

Tier 1: Establishing a Trustworthy Enclave ABI



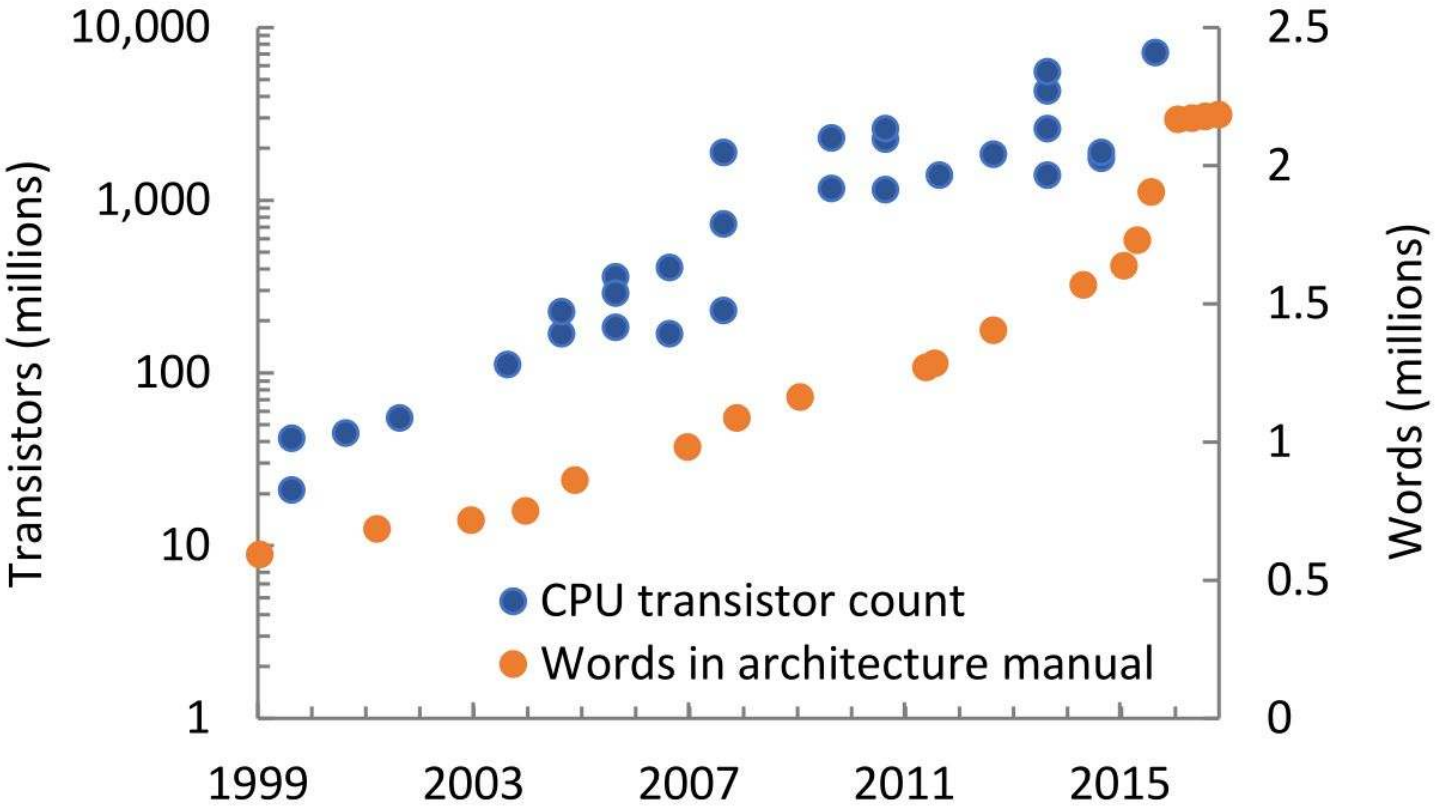
Tier 1: Establishing a Trustworthy Enclave ABI

$\rightsquigarrow$ Attacker controls CPU register contents on enclave entry/exit

$\leftrightarrow$ Compiler expects well-behaved **calling convention** (e.g., stack)

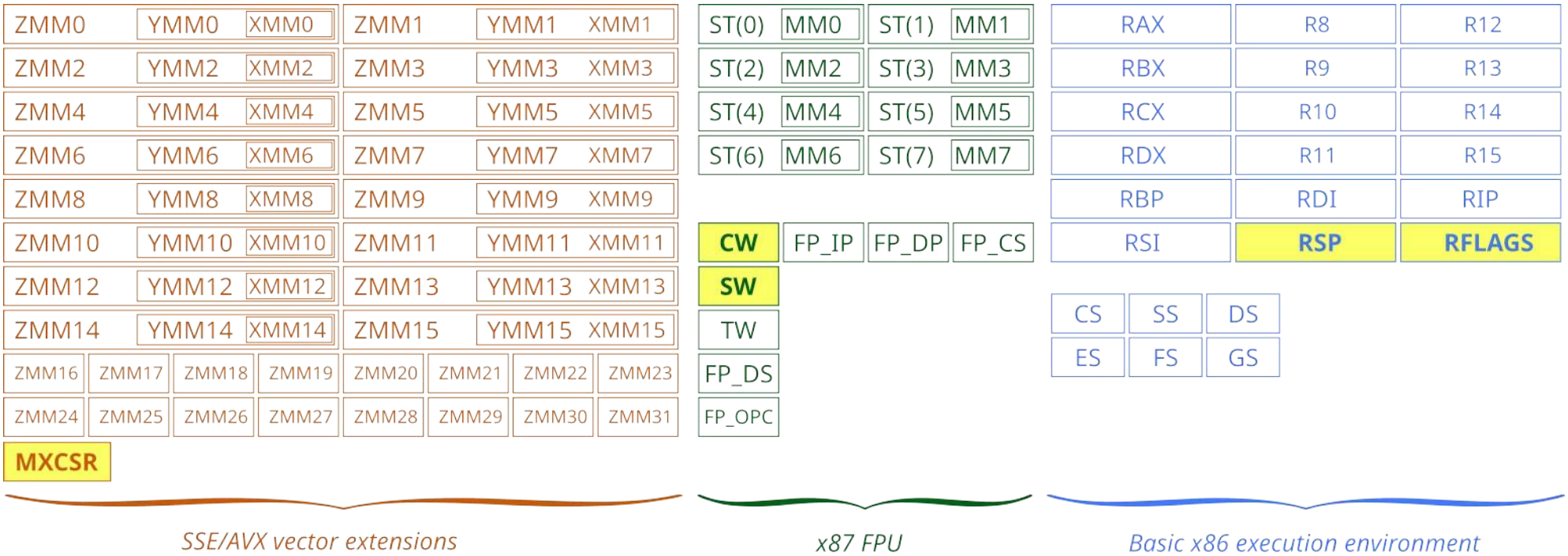
$\Rightarrow$ Need to **initialize CPU registers** on entry and **scrub** before exit!

Intel x86: Complexity Growth…



Baumann, "Hardware is the New Software", HotOS'17.

Intel x86: The Complete Register Set



x86 user-space CPU control registers

Van Bulck et al. “A Tale of Two Worlds: Assessing the Vulnerability of Enclave Shielding Runtimes”, CCS 2019.
Alder et al. “Faulty Point Unit: ABI Poisoning Attacks on Intel SGX”, ACSAC 2020.
Cui et al. “SmashEx: Smashing SGX Enclaves Using Exceptions”, CCS 2021.


SHELDON COOPER
presents
FUN WITH FLAGS
REC

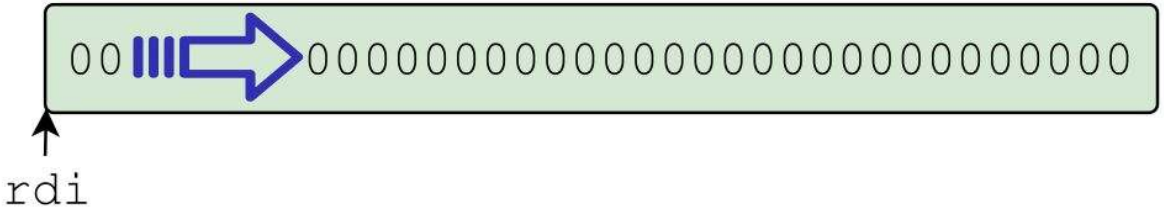
x86 String Instructions: Direction Flag (DF) Operation

- Special x86 rep string instructions to speed up streamed memory operations
- Default operate **left-to-right**

```c
/* memset(buf, 0x0, 100) */
for (int i=0; i < 100; i++)
  buf[i] = 0x0;
```

→

```asm
lea rdi, buf
mov al, 0x0
mov ecx, 100
rep stos [rdi], al
```



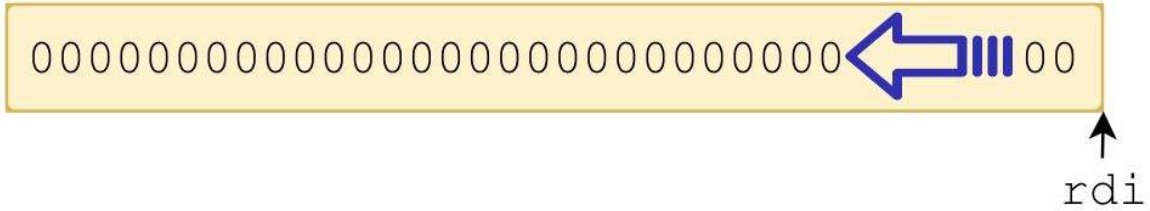
x86 String Instructions: Direction Flag (DF) Operation

- Special x86 rep string instructions to speed up streamed memory operations
- Default operate **left-to-right**, unless software sets *RFLAGS.DF=1*

```
/* memset(buf, 0x0, 100) */
for (int i=0; i < 100; i++)
  buf[i] = 0x0;
```

→

```
lea rdi, buf+100
mov al, 0x0
mov ecx, 100
std ; set direction flag
rep stos [rdi], al
```



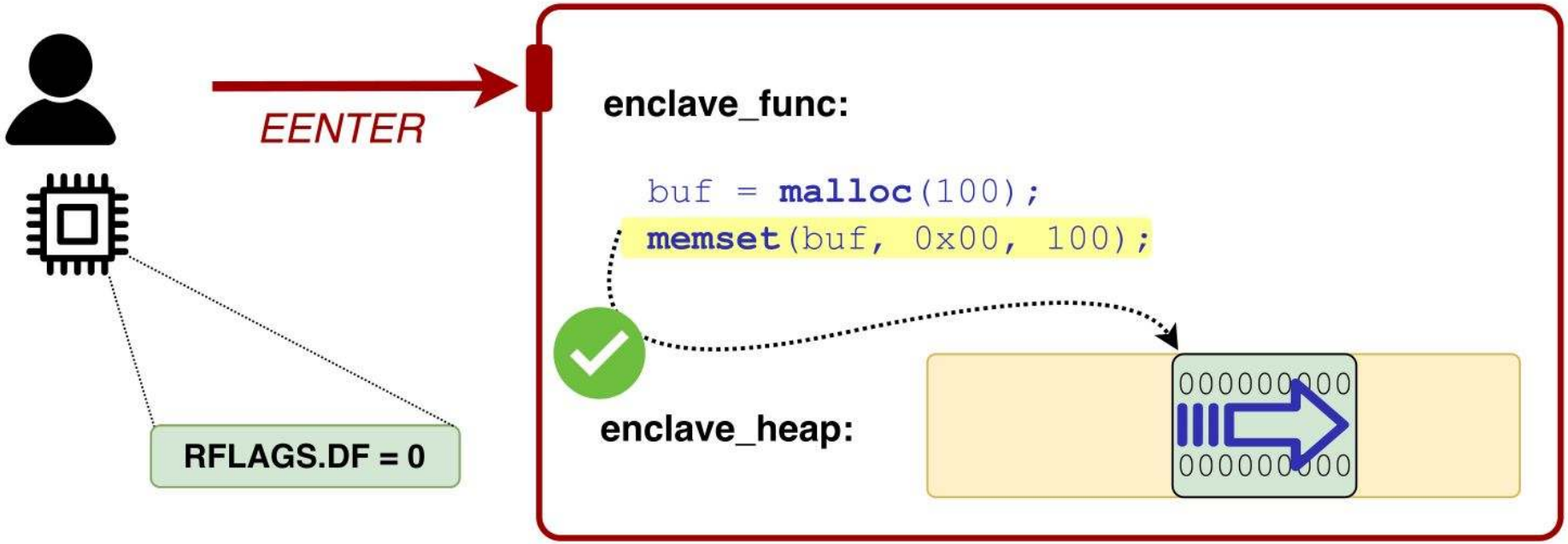
SGX-DF: Inverting Enclaved String Memory Operations

x86 System-V ABI

[8] The direction flag `DF` in the `%rFLAGS` register must be clear (set to "forward" direction) on function entry and return. Other user flags have no specified role in the standard calling sequence and are *not* preserved across calls.

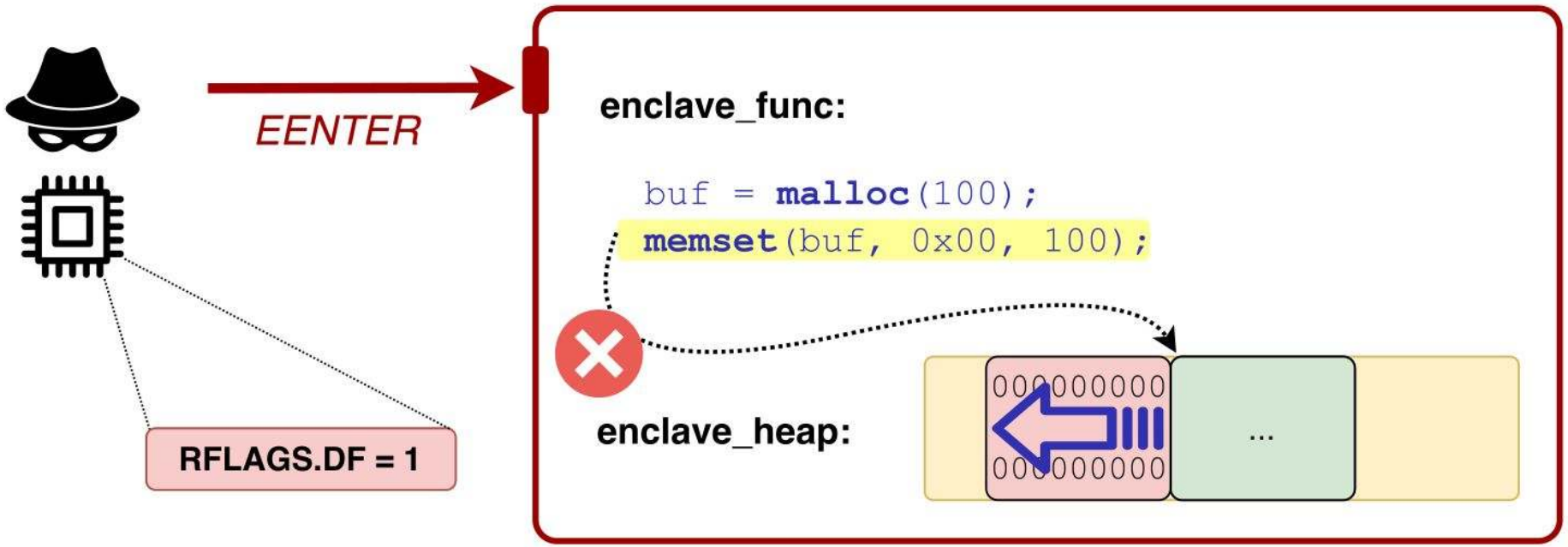
SGX-DF: Inverting Enclaved String Memory Operations

Intended heap memory initialization: left-to-right



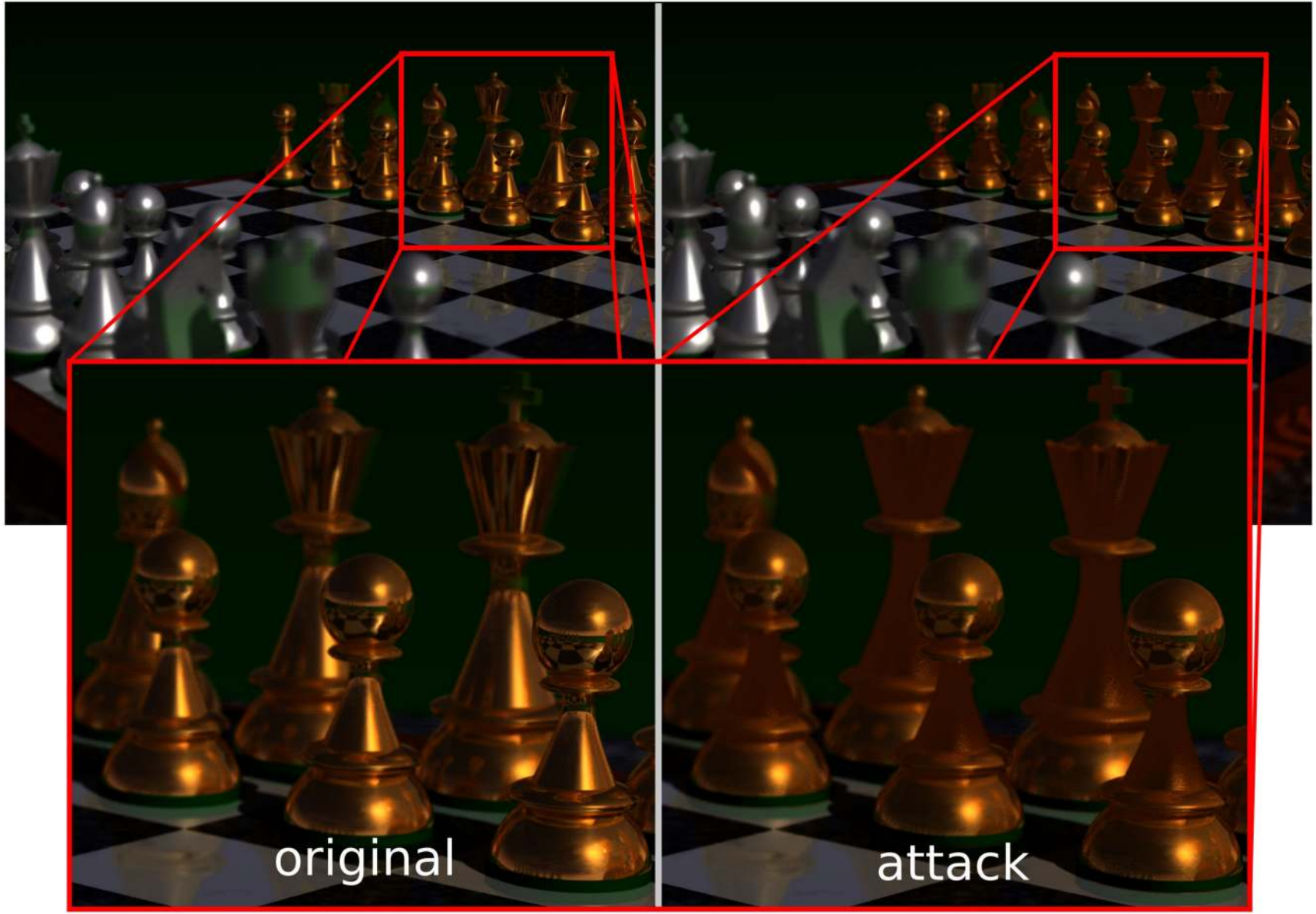
SGX-DF: Inverting Enclaved String Memory Operations

Enclave heap **memory corruption:** right-to-left...

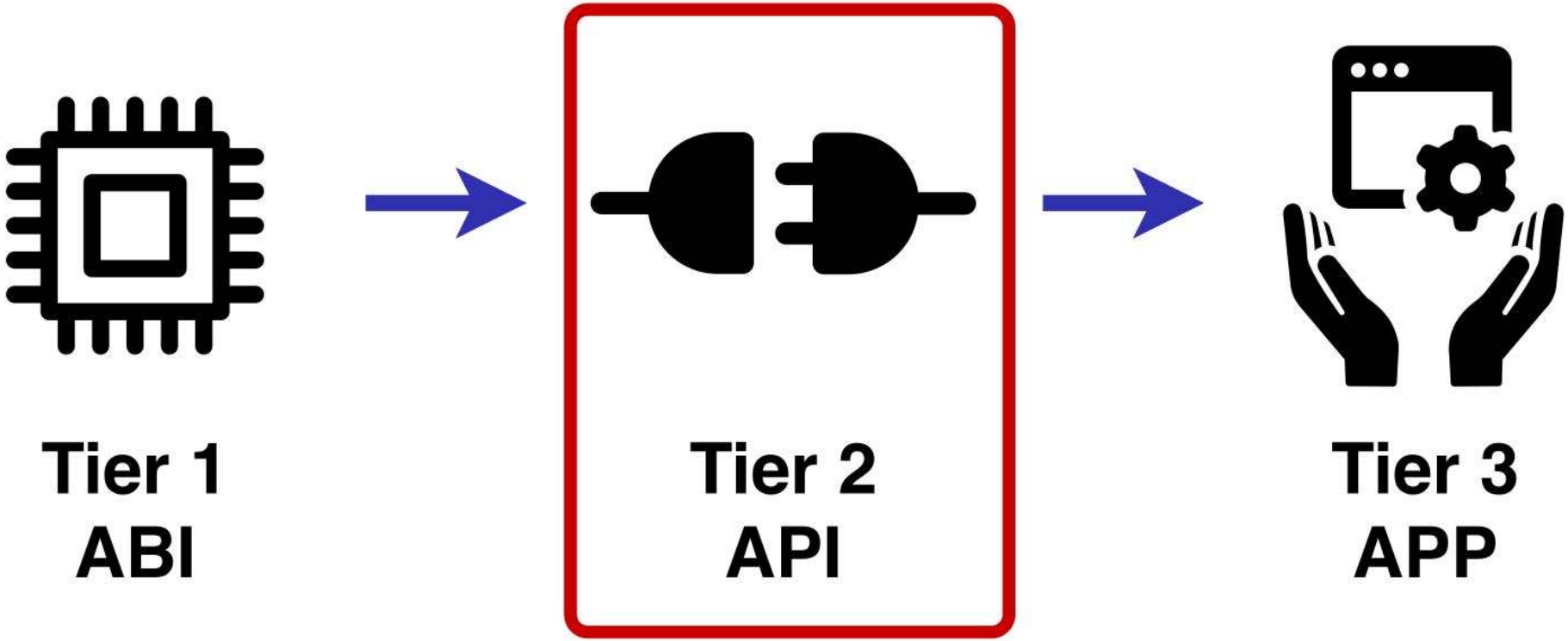


Van Bulck et al. "A Tale of Two Worlds: Assessing the Vulnerability of Enclave Shielding Runtimes", CCS 2019.

Alder et al. “Faulty Point Unit: ABI Poisoning Attacks on Intel SGX”, ACSAC 2020.

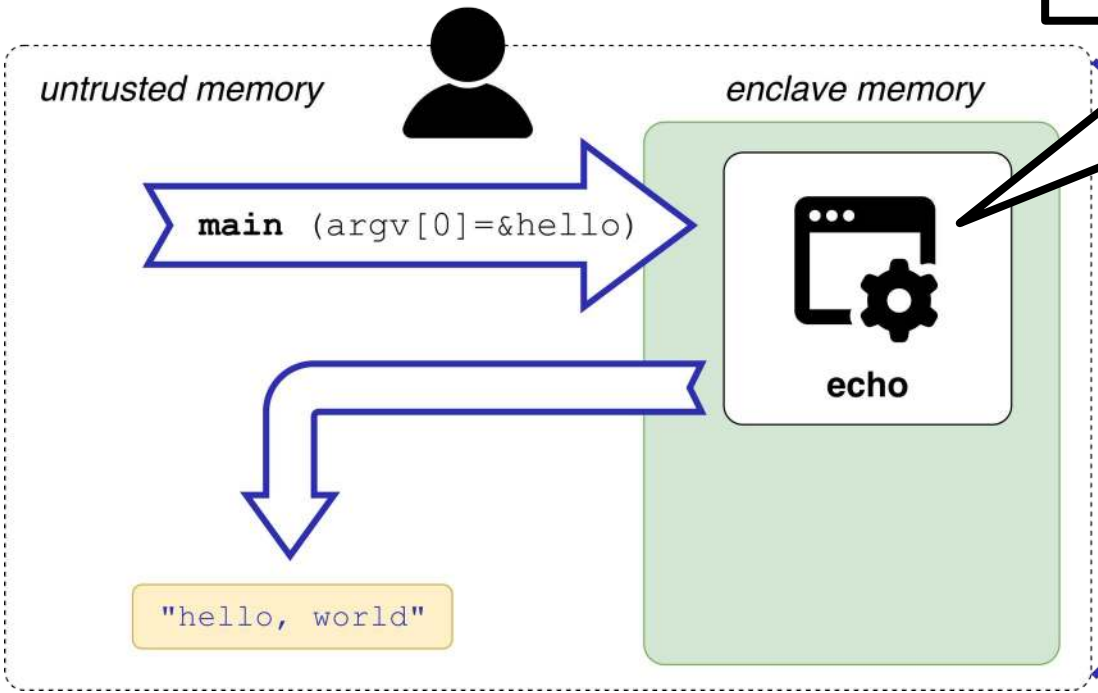

original
attack

Tier 2: Establishing a Trustworthy Enclave API



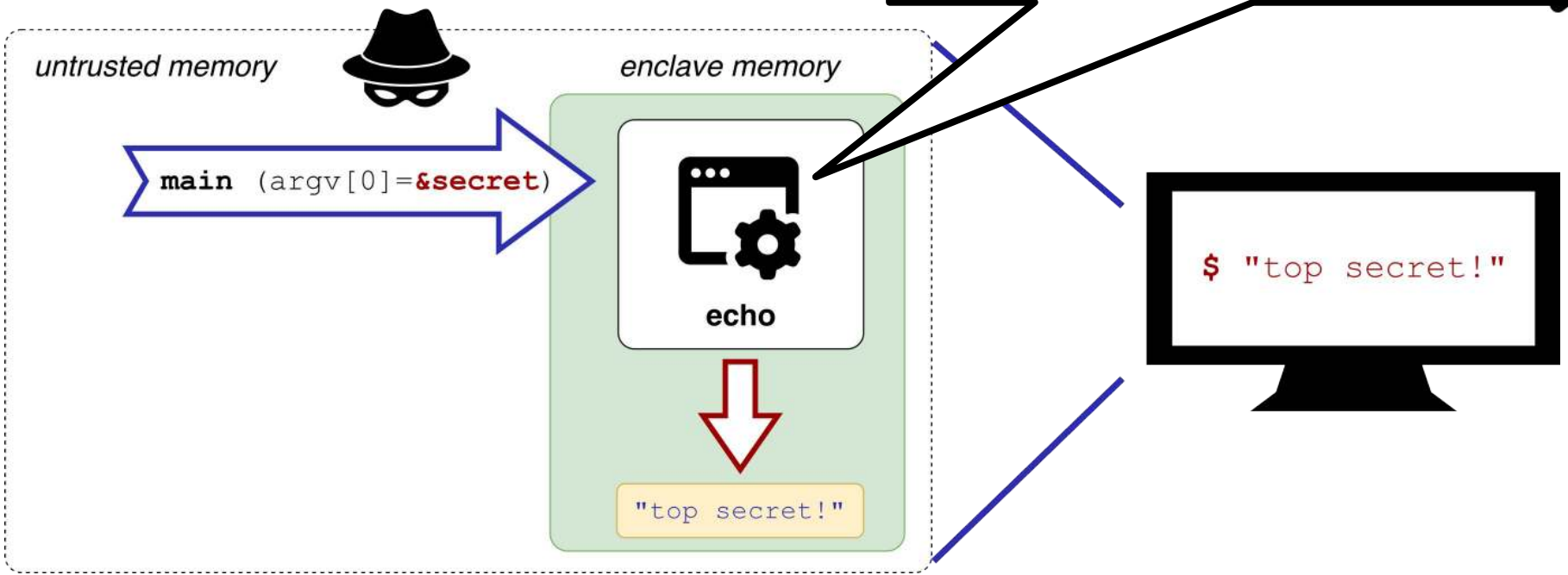
API Vulnerabilities: Confused-Deputy Attacks

```
int secure_func(int *attacker_pt){
    return *attacker_pt;
}
```



API Vulnerabilities: Confused-Deputy Attacks

```
int secure_func(int *attacker_pt){
    return *attacker_pt;
}
```



```
$
```

The Confused Deputy

(or why capabilities might have been invented)

Norm Hardy
Senior Architect

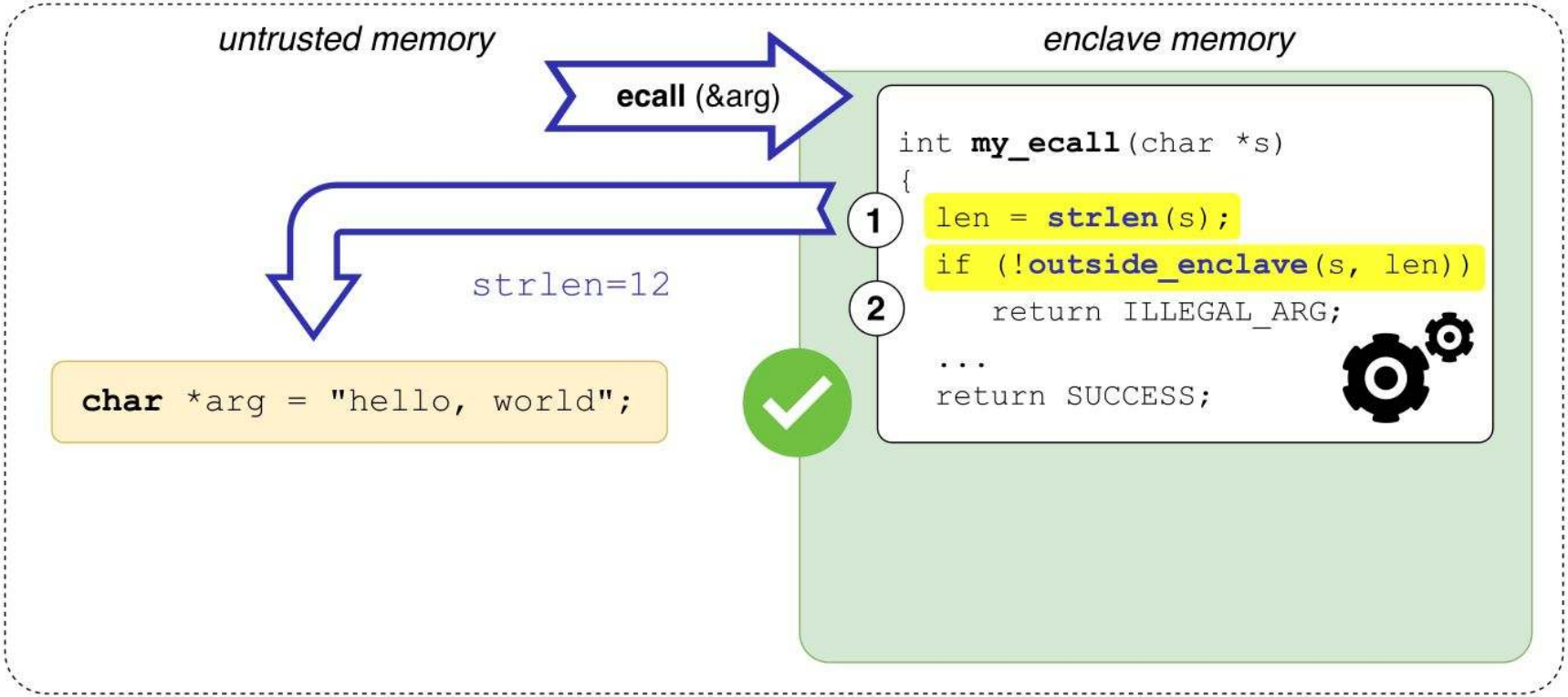
Key Logic
5200 Great America Parkway
Santa Clara, CA 95054-1108

This is a nearly true story (inessential details have been changed). The events happened about eleven years ago at Tymshare, a company which provided commercial timesharing services. Before this happened I had heard of capabilities and thought that they were neat and tidy, but was not yet convinced that they were necessary. This occasion convinced me that they were necessary.

https://dl.acm.org/doi/10.1145/54289.871709

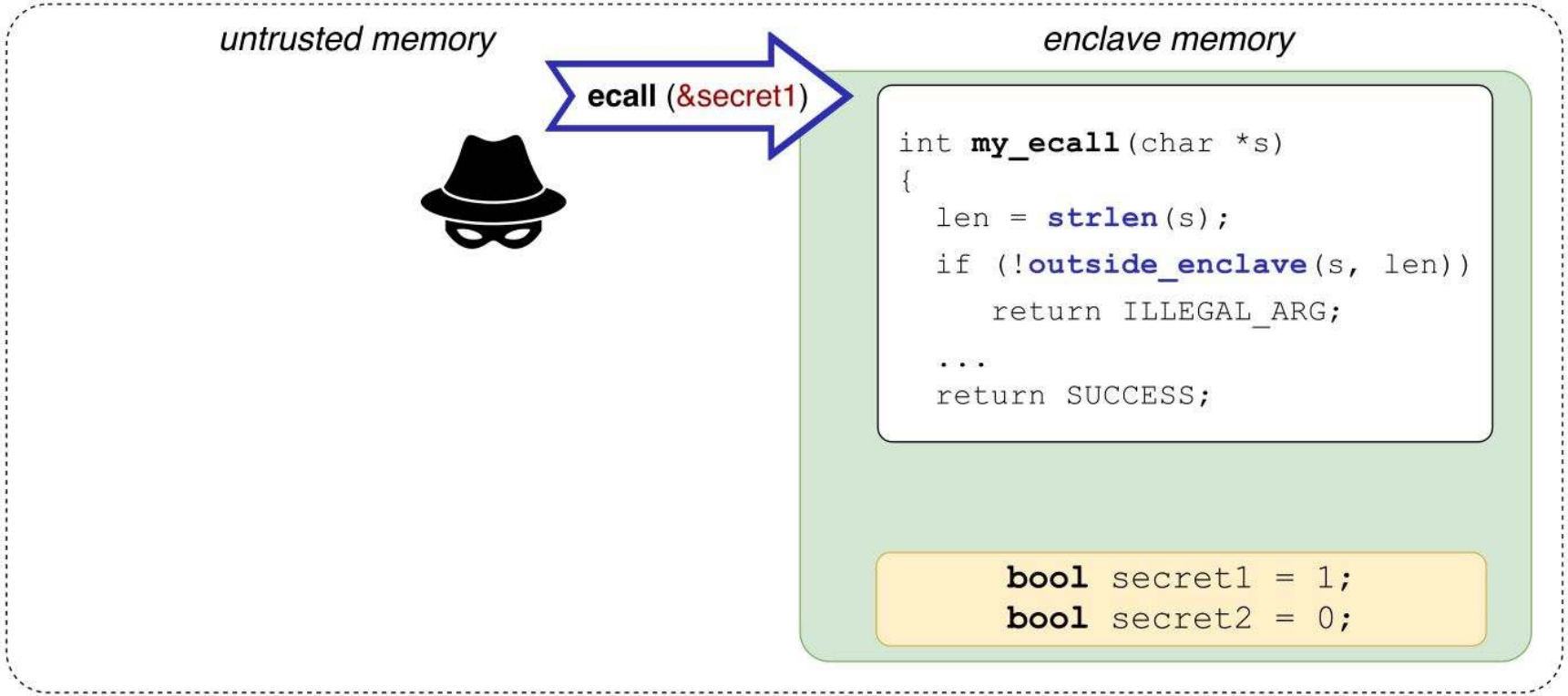
Intel SGX-SDK: Null-terminated strings are hard. . .

Idea: 2-stage approach ensures string arguments fall *entirely* outside enclave



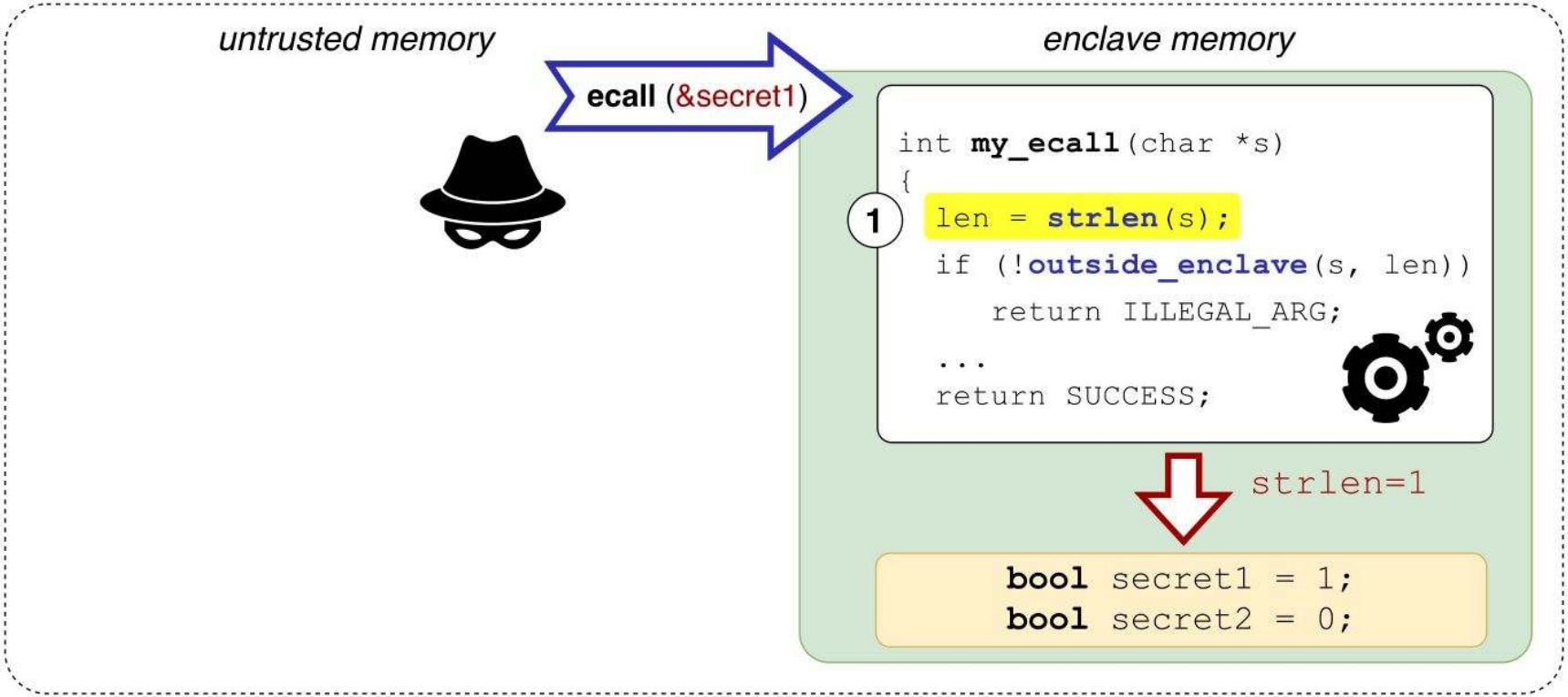
Intel SGX-SDK: Null-terminated strings are hard. . .

. . . **but** what if we try passing an illegal, in-enclave pointer anyway?



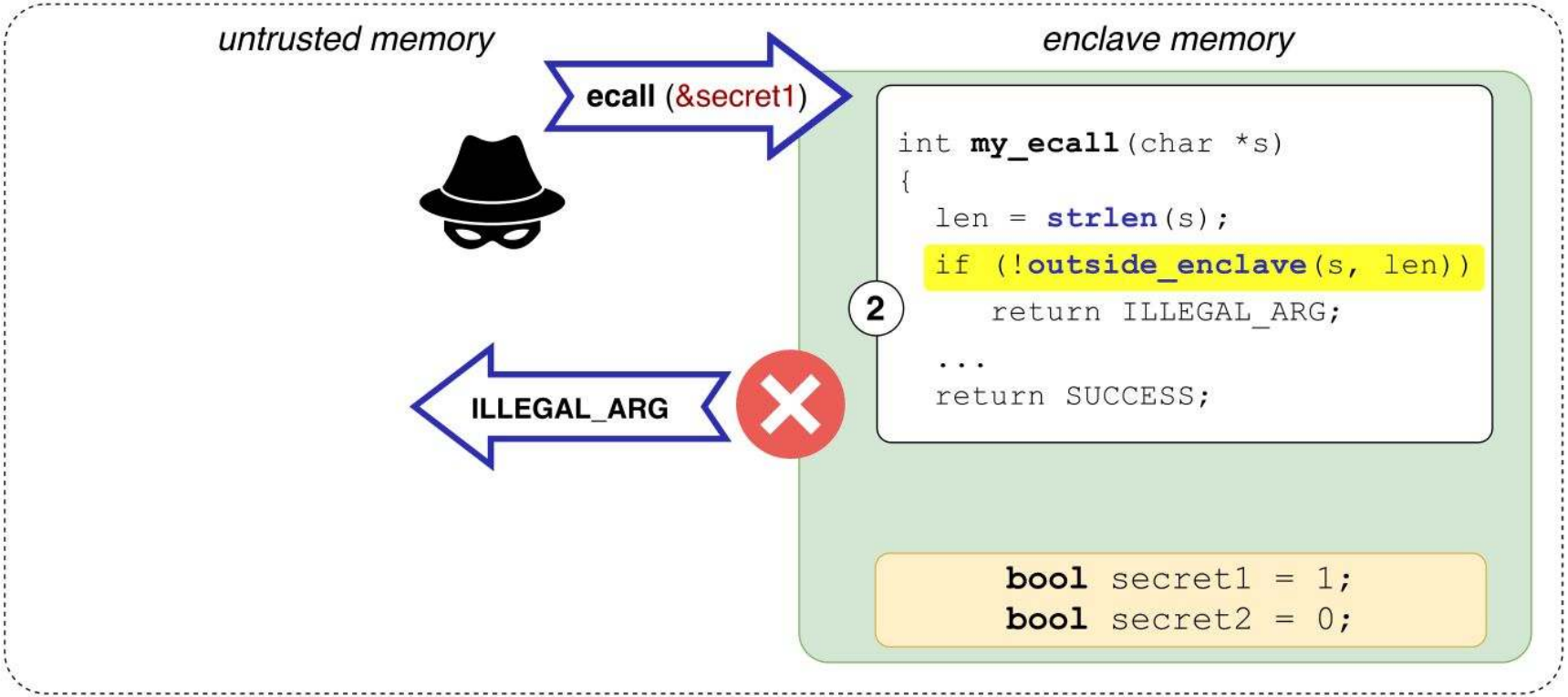
Intel SGX-SDK: Null-terminated strings are hard. . .

Enclave **first** computes length of secret, in-enclave buffer!



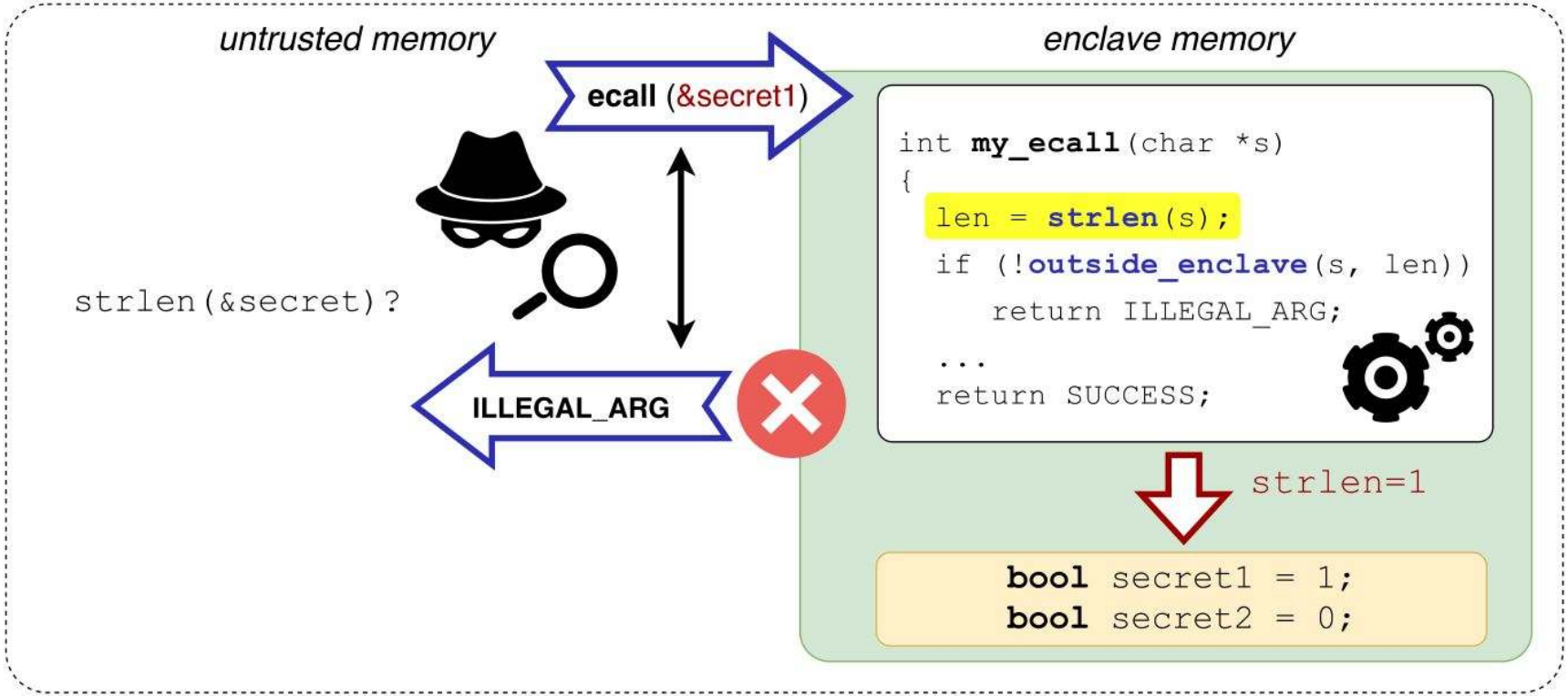
Intel SGX-SDK: Null-terminated strings are hard. . .

. . . and only **afterwards verifies** whether *entire string* falls outside enclave



Intel SGX-SDK: Null-terminated strings are hard. . .

Idea: `strlen()` timing as a side-channel oracle for in-enclave null bytes ☺



Van Bulck et al. "A Tale of Two Worlds: Assessing the Vulnerability of Enclave Shielding Runtimes", CCS 2019.


CVE-2018-3626: ALL YOUR ZERO BYTES
ARE BELONG TO US
imgflip.com

Manual code review finds 35 vulnerabilities in 8 enclave SDKs

All issues have been privately reported and patches are available.

Written by **Catalin Cimpanu,** Contributor

Nov. 12, 2019 at 10:00 a.m. PT

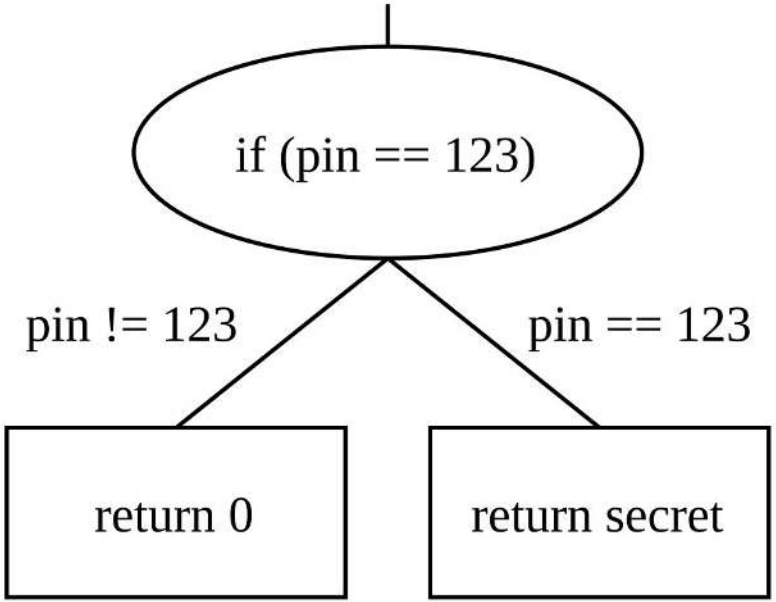
https://www.zdnet.com/article/manual-code-review-finds-35-vulnerabilities-in-8-enclave-sdks/


time to validate

Background: Binary-Level Symbolic Execution

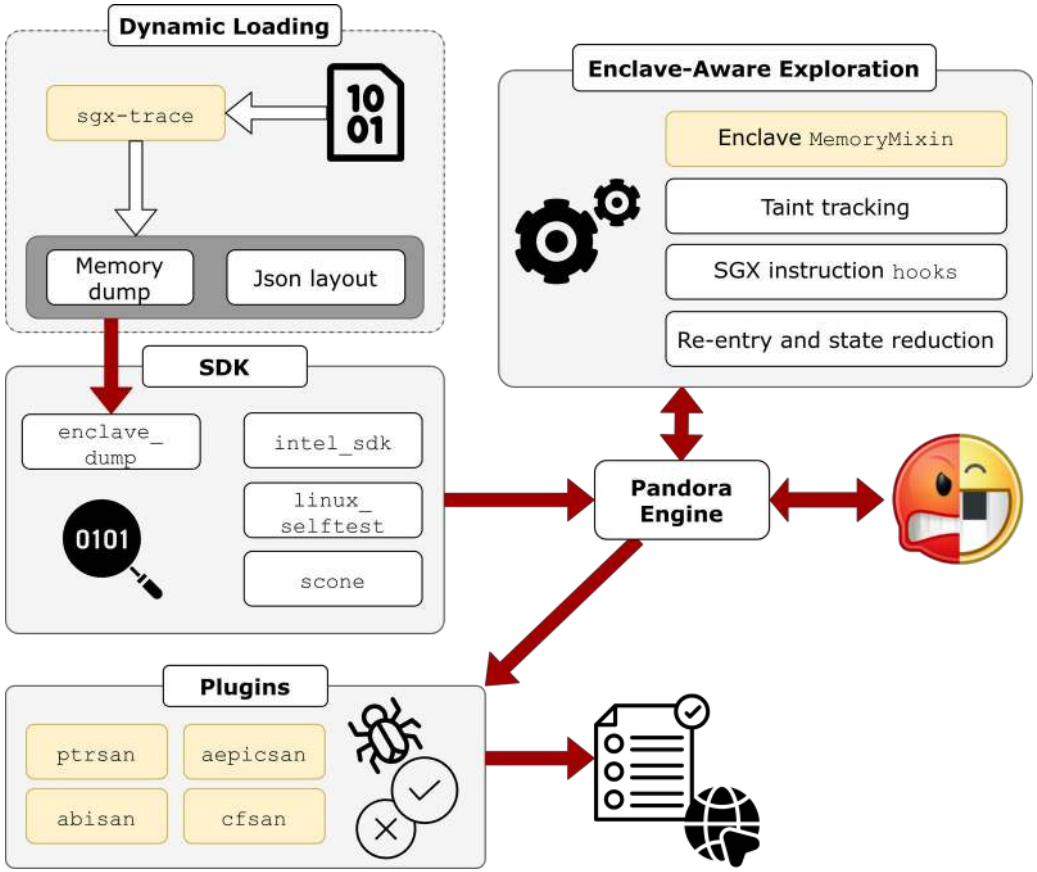
```
1 int ecall(int pin){
2     if(pin == 123){
3         return secret;
4     } else {
5         return 0;
6     }
7 }
```

https://angr.io/



- Symbolic execution uses a constraint solver
- Execution works on instruction-level, i.e., as close to the binary as possible

Pandora: Principled Symbolic Validation of Intel SGX Enclaves



- **Truthful:** SDK-agnostic enclave memory model

  → *Exact attested memory layout (MRENCLAVE)*

- **Extensible:** Validate vulnerability invariants via plugins

  → *ABI + API + …*

Alder et al. "Pandora: Principled Symbolic Validation of Intel SGX Enclave Runtimes", IEEE S&P 2024.

Pandora: Principled Symbolic Validation of Intel SGX Enclaves

- 4 plugins
- 11 runtimes

- > 200 new and 69 reproduced vulnerability instances
- 7 CVEs

Runtime	Version	Prod	Src	Plugin	Instances	CVE
Newly found vulnerabilities in shielding runtimes (total 200 instances)						
EnclaveOS	3.28	✓	✗†	ABISan	1	
EnclaveOS	3.28	✓	✗†	PTRSan	15	CVE-2023-38022
EnclaveOS	3.28	✓	✗†	ÆPICSan	33	CVE-2023-38021
EnclaveOS	3.28	✓	✗†	CFSan	2	
GoTEE	b35f	✗	✓	PTRSan	31	
GoTEE	b35f	✗	✓	ÆPICSan	18	
GoTEE	b35f	✗	✓	CFSan	1	
Gramine	1.4	✓	✓	ABISan	1	
Intel SDK	2.15.1	✓	✓	PTRSan	2	CVE-2022-26509
Intel SDK	2.19	✓	✓	ÆPICSan	22	
↪ Occlum	0.29.4	✓	✓	ÆPICSan	11	
Linux selftest	5.18	✗	✓	ABISan	1	
↪ DCAP	1.16	✓	✓	ABISan	1	
↪ Inclavare	0.6.2	✗	✓	ABISan	1	
Linux selftest	5.18	✗	✓	PTRSan	5	
↪ DCAP	1.16	✓	✓	PTRSan	17	
↪ Inclavare	0.6.2	✗	✓	PTRSan	2	
Linux selftest	5.18	✗	✓	CFSan	1	
↪ Inclavare	0.6.2	✗	✓	CFSan	1	
Open Enclave	0.19.0	✓	✓	ABISan	2	CVE-2023-37479
Rust EDP	1.71	✓	✓	ABISan	1	
SCONE	5.7/5.8	✓	✗	ABISan	2/1	CVE-2022-46487
SCONE	5.7/5.8	✓	✗	PTRSan	10/3	CVE-2022-46486
SCONE	5.7/5.8	✓	✗	ÆPICSan	11/3	CVE-2023-38023
SCONE	5.8	✓	✗	CFSan	1	

Alder et al. "Pandora: Principled Symbolic Validation of Intel SGX Enclave Runtimes", IEEE S&P 2024.

Report PointerSanitizationPlugin

Plugin description: Validates attacker-tainted pointer dereferences.

Analyzed 'pandora_selftest_enclave_sanitization3.elf', with 'Linux selftest enclave' enclave runtime. Ran for 0:00:12.758955 on 2023-08-03_19-16-58.

Enclave info: Address range is [0x0, 0xbfff]

Summary: Found 1 unique WARNING issue; 2 unique CRITICAL issues.

Report summary

Severity	Reported issues
WARNING	• *Attacker tainted read inside enclave* at 0x2476
CRITICAL	• *Unconstrained read* at 0x22c3 • *Unconstrained read* at 0x20be

Issues reported at 0x22c3 1 do_encl_op_get_from_unmeasured CRITICAL Unconstrained read

Unconstrained read CRITICAL RIP=0x22c3

Plugin extra info

Key	Value
Address	<BV64 0x3000 + ((attacker_mem_66_32{UNINITIALIZED} .. 0x1) << 0x3)>
Attacker tainted	True
Length	8
Pointer range	[0x3008, 0xffffffff800003008]
Pointer can wrap address space	False
Pointer can lie in enclave	True
Extra info	Read address may lie inside or outside enclave

Execution state info

Disassembly

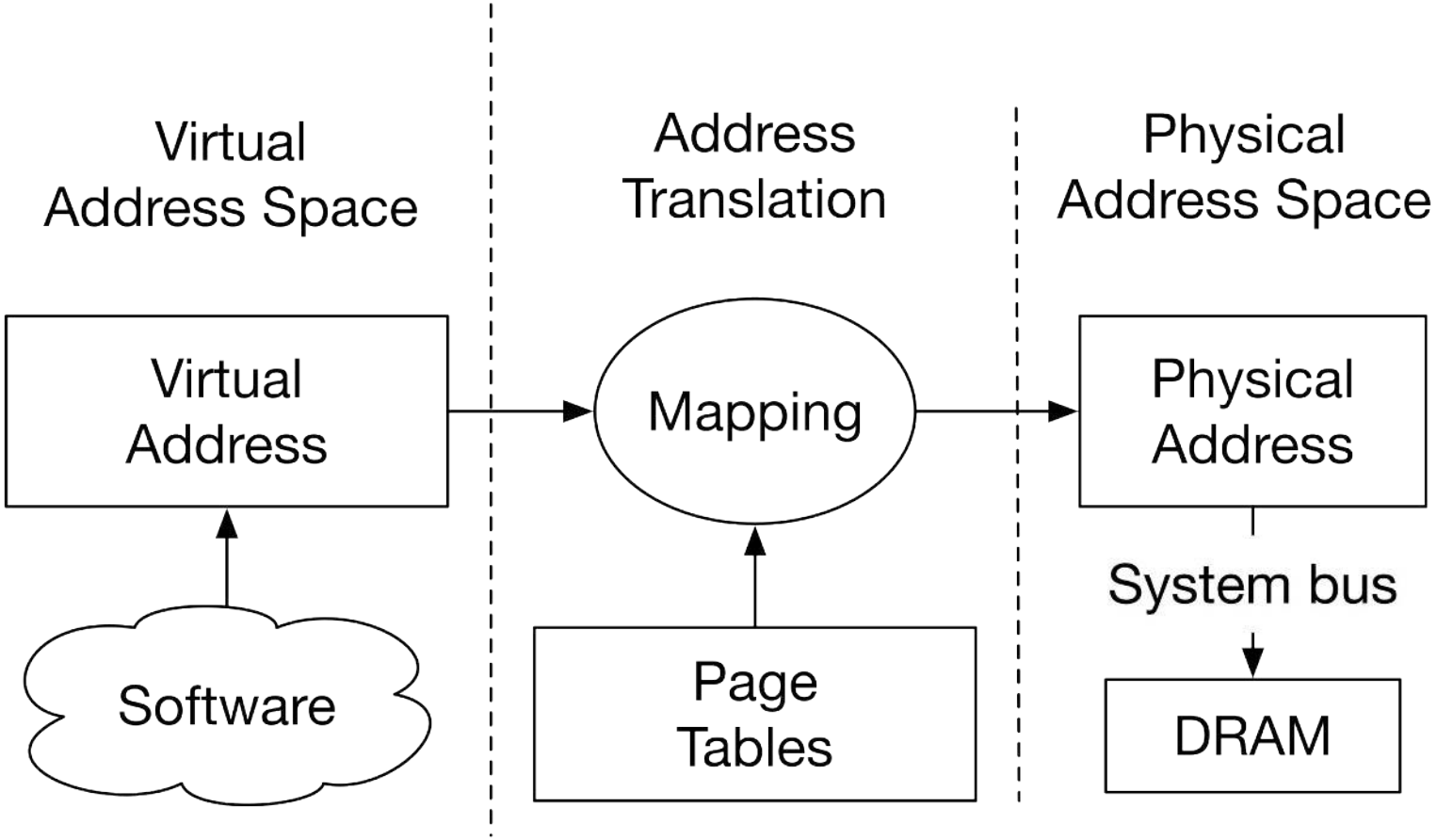
CPU registers

Backtrace

Basic block trace (most recent first)

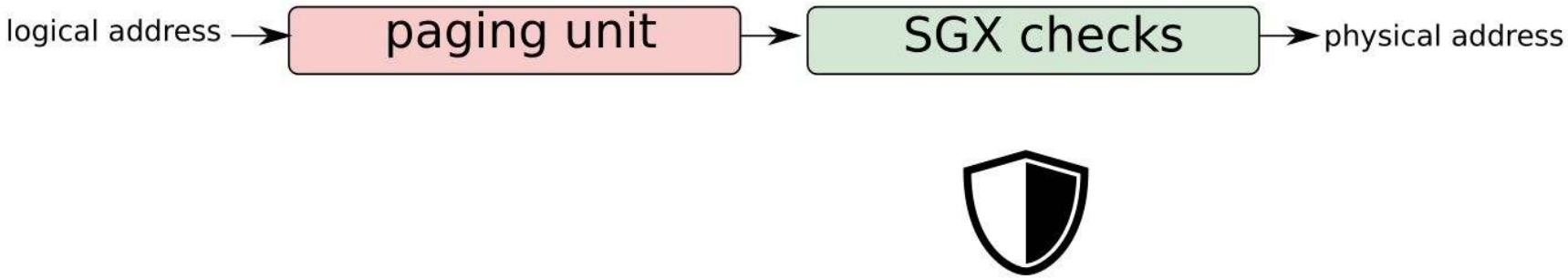
Part #2: Privileged Software Interface

Background: The Virtual Memory Abstraction



Costan et al. "Intel SGX explained", IACR 2016.

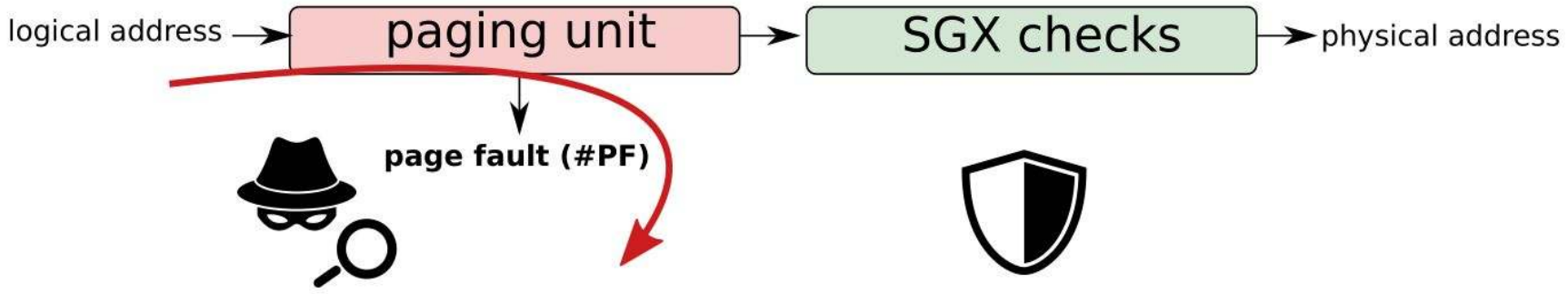
Idea: Page Faults as a Side Channel



SGX machinery protects against direct address remapping attacks

Xu et al. "Controlled-Channel Attacks: Deterministic Side Channels for Untrusted Operating Systems", IEEE S&P 2015.

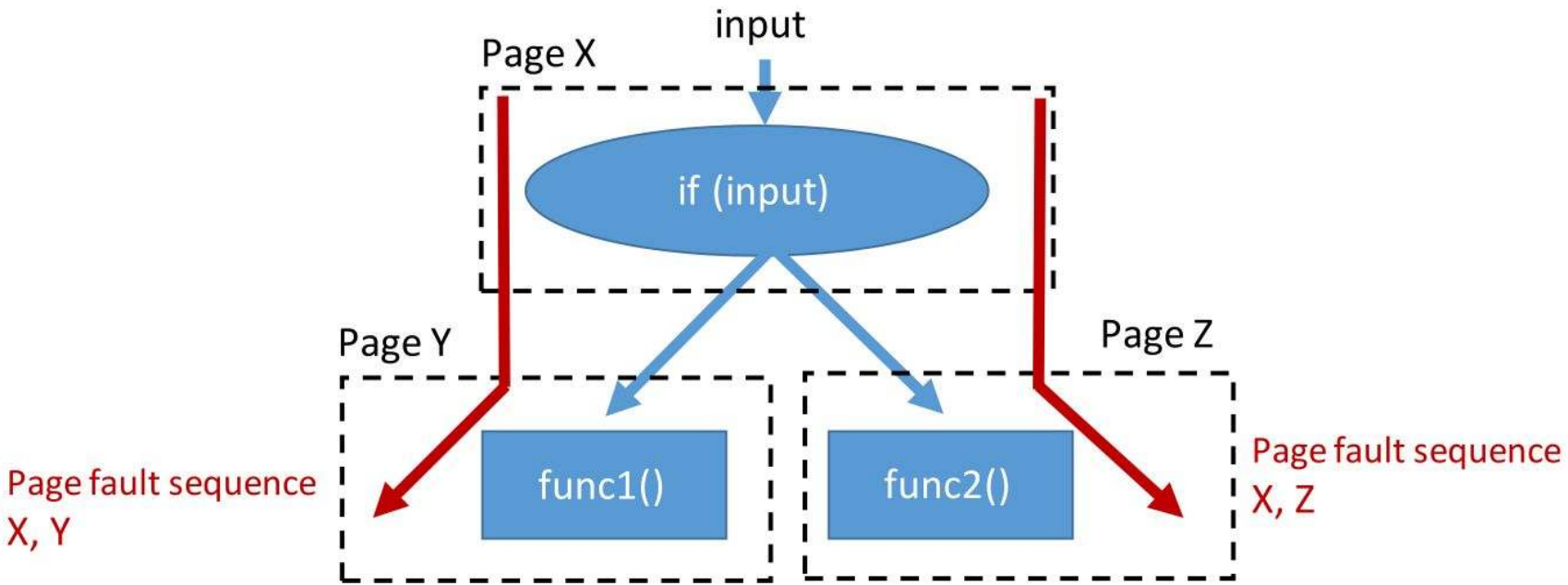
Idea: Page Faults as a Side Channel



. . . but untrusted address translation may **fault(!)**

Xu et al. "Controlled-Channel Attacks: Deterministic Side Channels for Untrusted Operating Systems", IEEE S&P 2015.

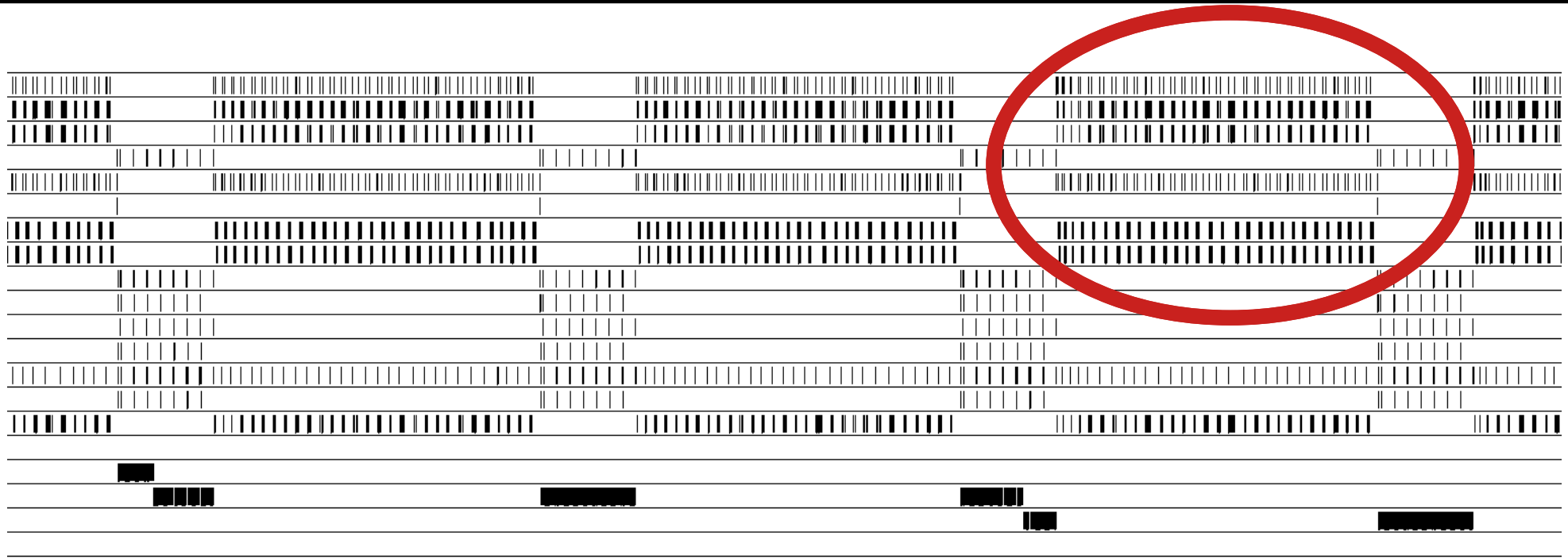
Intel SGX: Page Faults as a Side Channel



Xu et al.: "Controlled-channel attacks: Deterministic side channels for untrusted operating systems", Oakland 2015.

⇒ Page fault traces leak **private control data/flow**

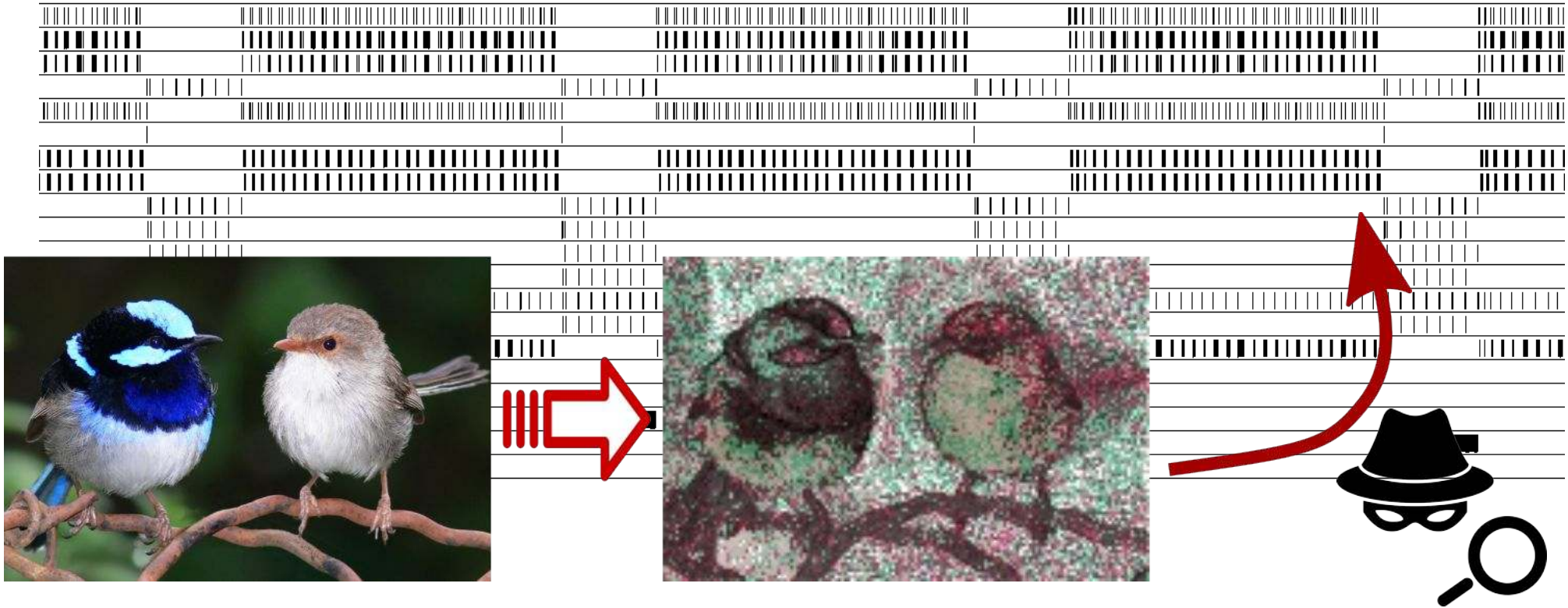
Spatial Resolution: Page-Granular Memory Access Traces



Detailed trace of (coarse-grained) code and data accesses over time…

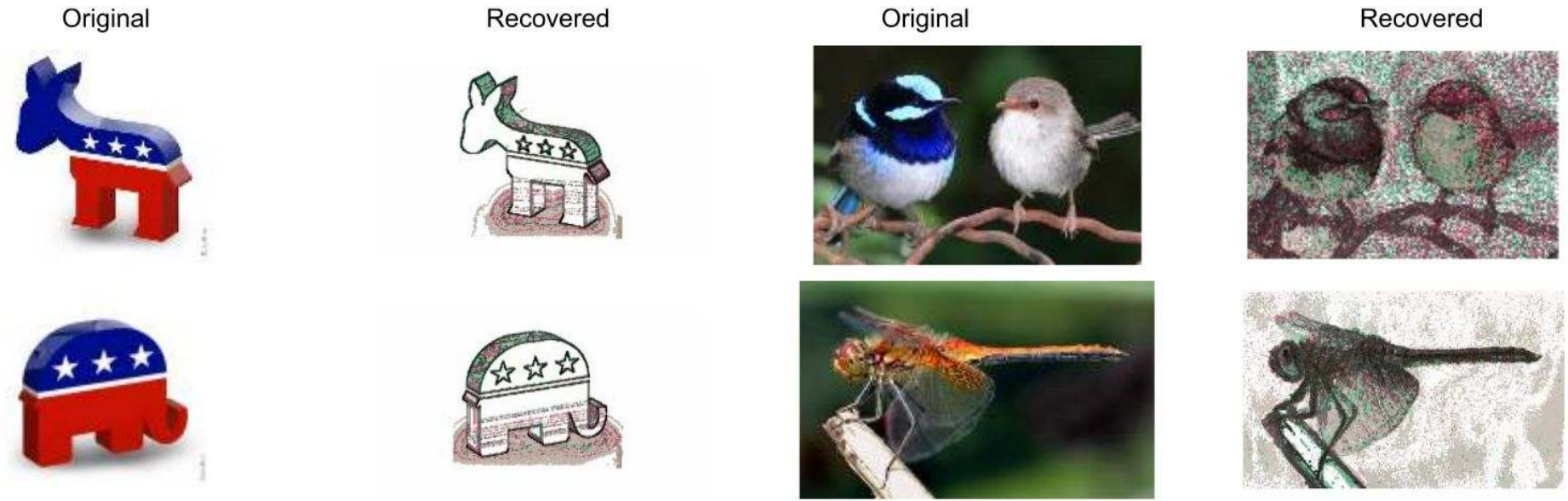
Xu et al. "Controlled-Channel Attacks: Deterministic Side Channels for Untrusted Operating Systems", IEEE S&P 2015.

Spatial Resolution: Page-Granular Memory Access Traces



Xu et al. “Controlled-Channel Attacks: Deterministic Side Channels for Untrusted Operating Systems”, IEEE S&P 2015.

Spatial Resolution: Page-Granular Memory Access Traces



Xu et al.: "Controlled-channel attacks: Deterministic side channels for untrusted operating systems", Oakland 2015.

. . . but **many faults** and a coarse-grained **4 KiB granularity**

Protection from Side-Channel Attacks

Intel® SGX does not provide explicit protection from side-channel attacks. It is the enclave developer's responsibility to address side-channel attack concerns.

In general, enclave operations that require an OCall, such as thread synchronization, I/O, etc., are exposed to the untrusted domain. If using an OCall would allow an attacker to gain insight into enclave secrets, then there would be a security concern. This scenario would be classified as a side-channel attack, and it would be up to the ISV to design the enclave in a way that prevents the leaking of side-channel information.

An attacker with access to the platform can see what pages are being executed or accessed. This side-channel vulnerability can be mitigated by aligning specific code and data blocks to exist entirely within a single page.

More important, the application enclave should use an appropriate crypto implementation that is side channel attack resistant inside the enclave if side-channel attacks are a concern.

Temporal Resolution Limitations for the Page-Fault Oracle

```c
size_t strlen (char *str)
{
  char *s;

  for (s = str; *s; ++s);
  return (s - str);
}
```

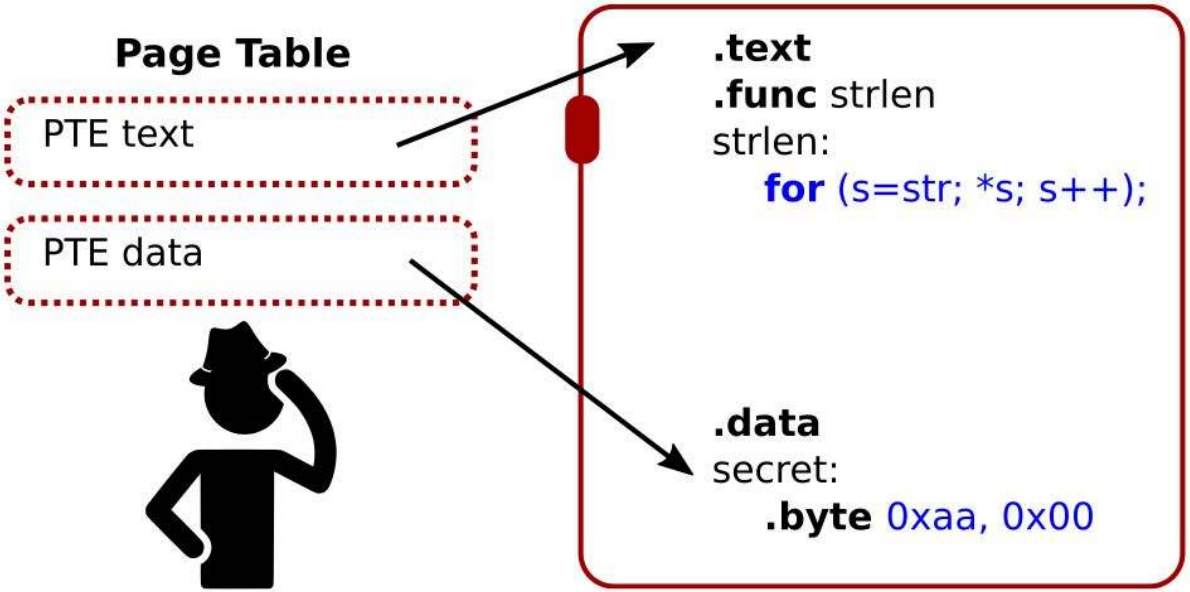
```
    mov  %rdi,%rax
1:  cmpb $0x0,(%rax)
    je   2f
    inc  %rax
    jmp  1b
2:  sub  %rdi,%rax
    retq
```

⇒ tight loop: 4 instructions, single memory operand, single code + data page

Counting `strlen` loop iterations?

Note: Page-fault attacks cannot make progress for 1 code + data page

Temporal Resolution Limitations for the Page-Fault Oracle

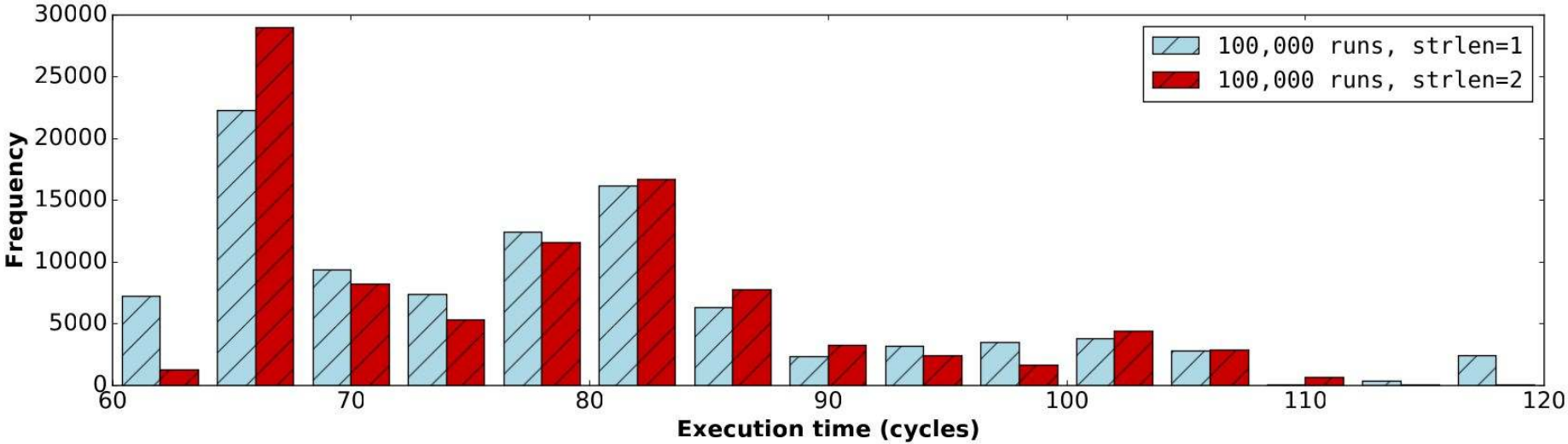


Counting `strlen` loop iterations?

Progress requires both pages present (non-faulting) ↔ page fault oracle

Building the Side-Channel Oracle with Execution Timing?

Too noisy: modern x86 processors are lightning fast. . .



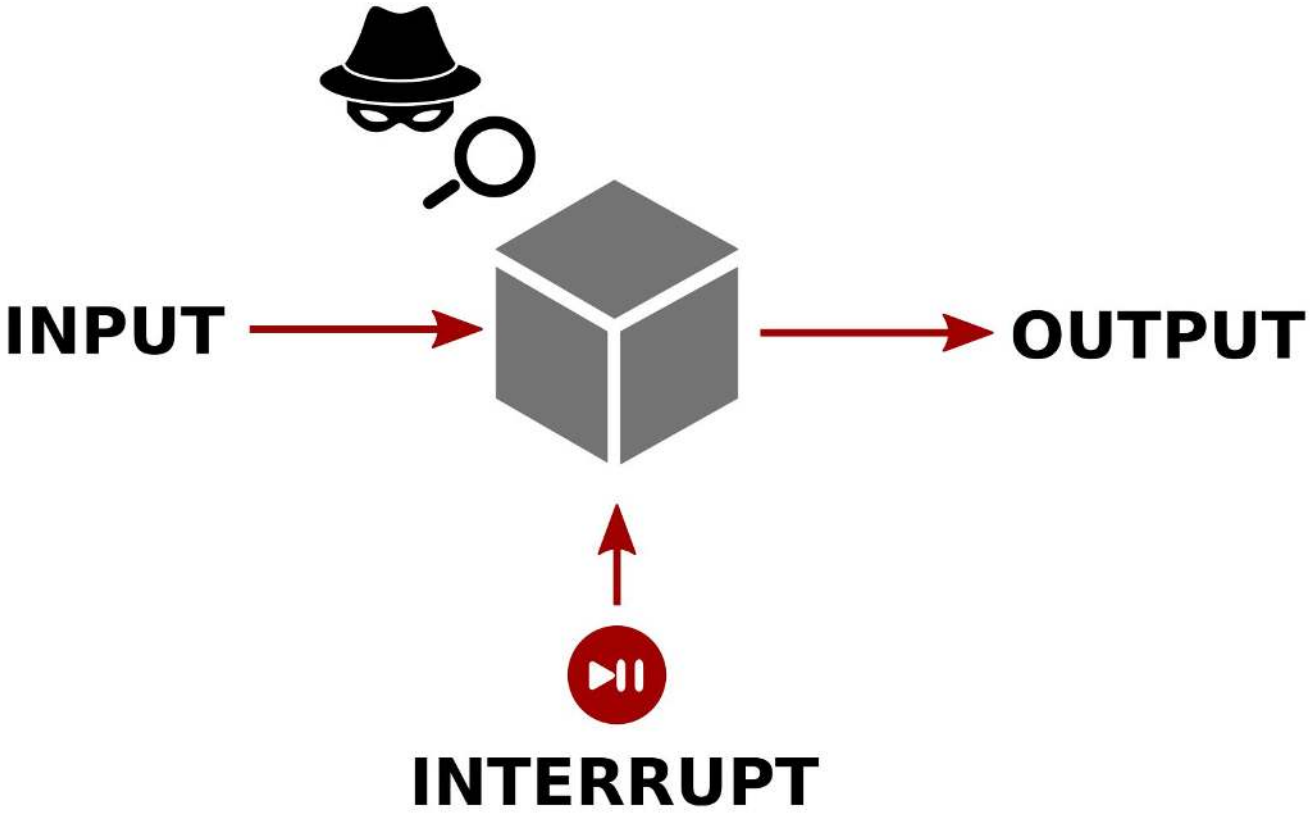
Challenge: Side-Channel Sampling Rate

Slow
shutter speed

Medium
shutter speed

Fast
shutter speed

SGX-Step: Executing Enclaves one Instruction at a Time



SGX-Step: Executing Enclaves one Instruction at a Time

SGX-Step

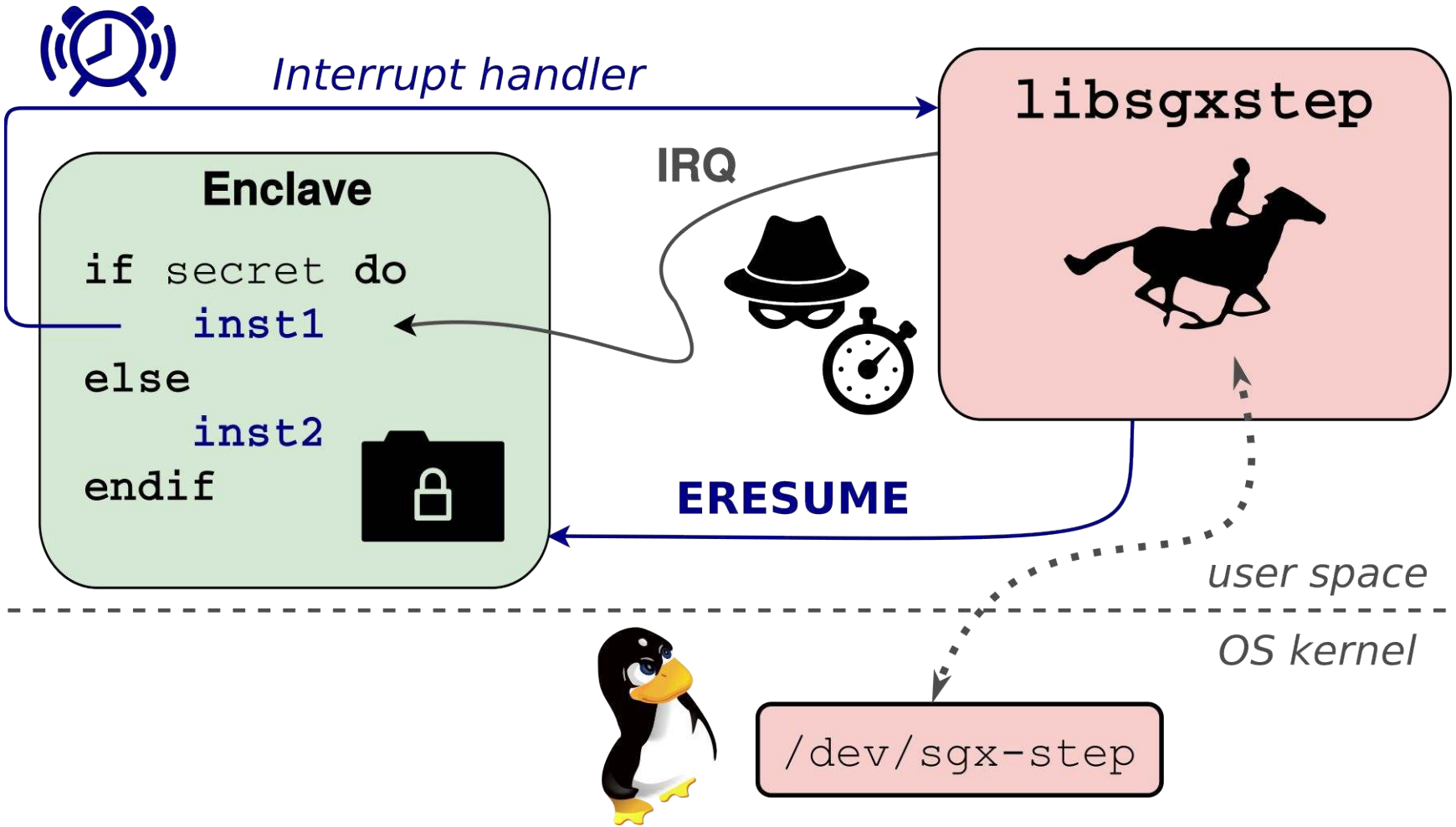
ACSAC 2023
Cybersecurity Artifacts
Impact Award

https://github.com/jovanbulck/sgx-step

Watch 22 | Star 245 | Fork 52

Van Bulck et al., "SGX-Step: A Practical Attack Framework for Precise Enclave Execution Control", SysTEX 2017.

SGX-Step: Executing Enclaves one Instruction at a Time



SGX-Step Demo: Single-Stepping Password Comparison

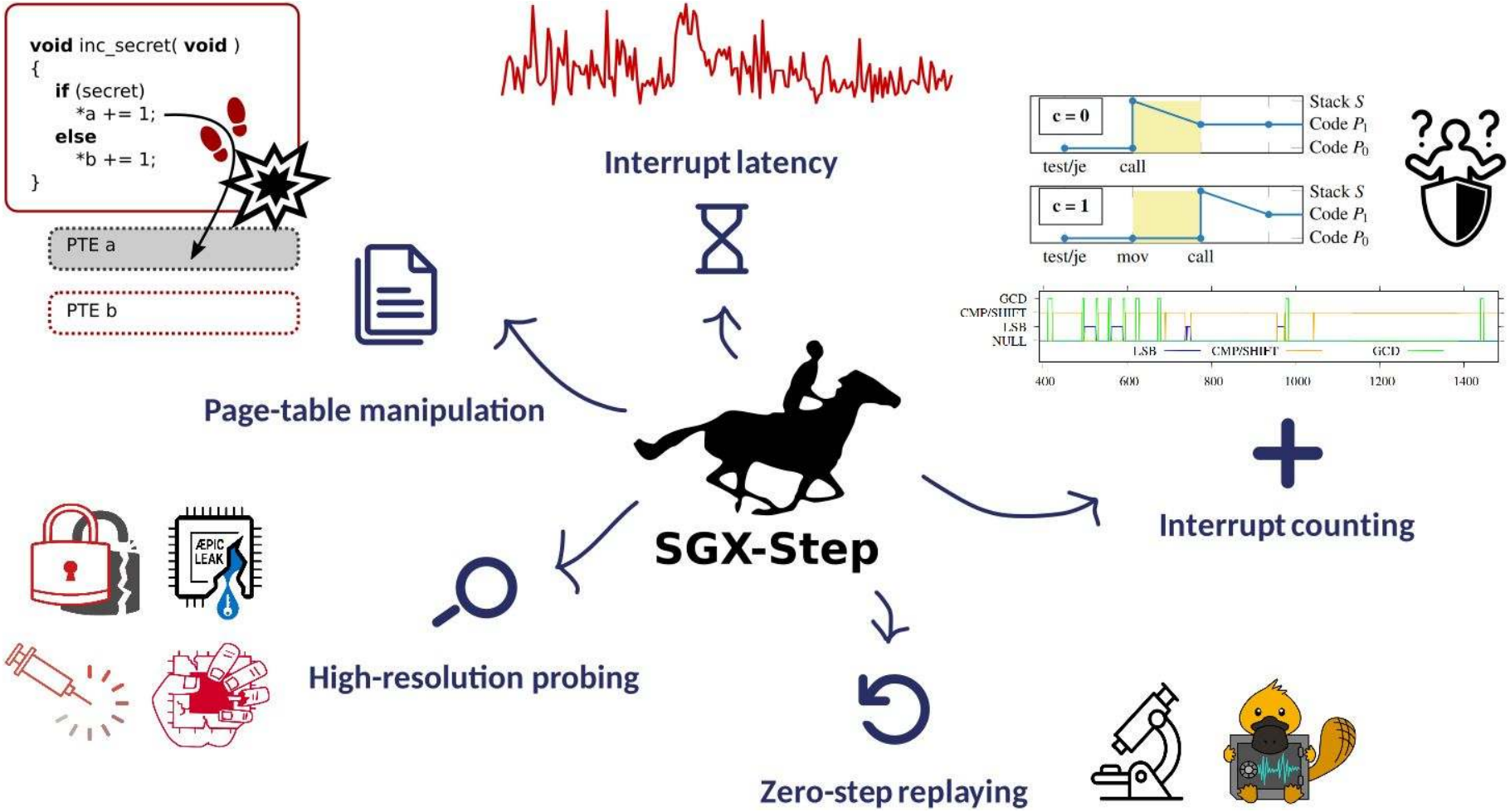
```
jo@breuer:~/sgx-step-demo$ sudo ./app
```

SGX-Step: Enabling a New Line of High-Resolution Attacks

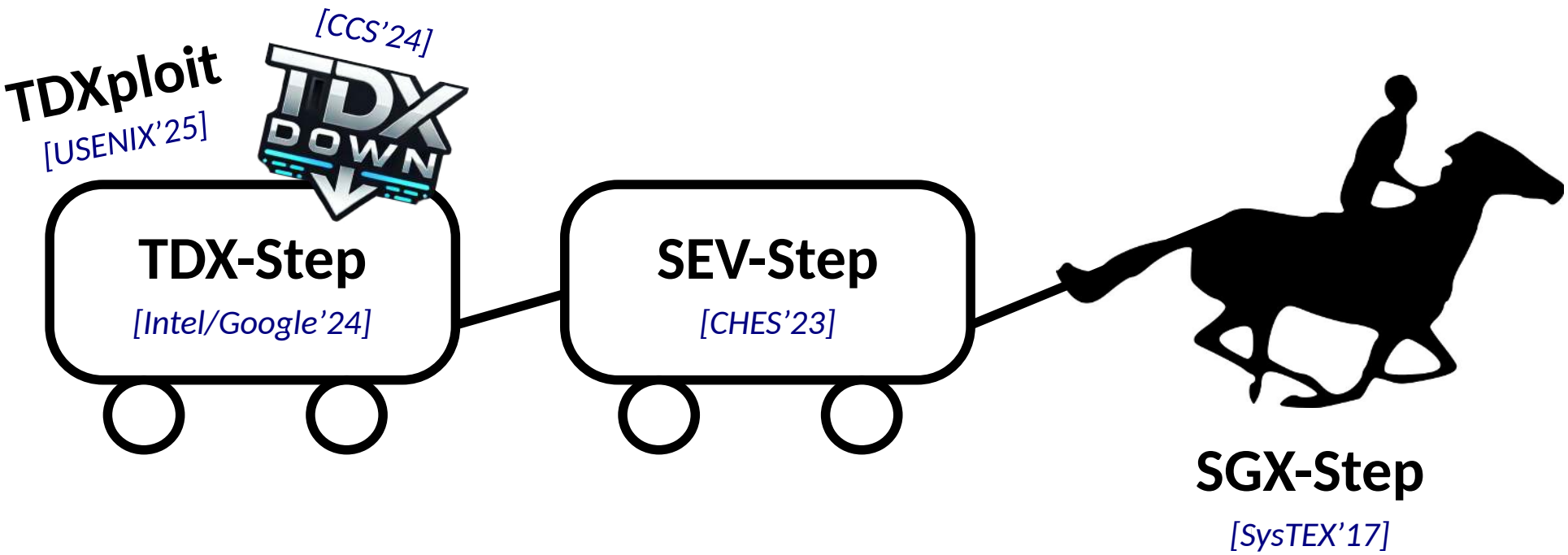
Yr	Venue	Paper	Step	Use Case	Drv
'15	S&P	Ctrl channel	~ Page	*Probe* (page fault)	✓
'16	ESORICS	AsyncShock	~ Page	*Exploit* (mem safety)	–
'17	CHES	CacheZoom	✗ >1	*Probe* (L1 cache)	✓
'17	ATC	Hahnel et al.	✗ 0 - >1	*Probe* (L1 cache)	✓
'17	USENIX	BranchShadow	✗ 5 - 50	*Probe* (BPU)	✗
'17	USENIX	Stealthy PTE	~ Page	*Probe* (page table)	✓
'17	USENIX	DarkROP	~ Page	*Exploit* (mem safety)	✓
'17	SysTEX	SGX-Step	✓ 0 - 1	*Framework*	✓
'18	ESSoS	Off-limits	✓ 0 - 1	*Probe* (segmentation)	✓
'18	AsiaCCS	Single-trace RSA	~ Page	*Probe* (page fault)	✓
'18	USENIX	Foreshadow	✓ 0 - 1	*Probe* (transient exec)	✓
'18	EuroS&P	SgxPectre	~ Page	*Exploit* (transient)	✓
'18	CHES	CacheQuote	✗ >1	*Probe* (L1 cache)	✓
'18	ICCD	SGXlinger	✗ >1	*Probe* (IRQ latency)	✗
'18	CCS	Nemesis	✓ 1	*Probe* (IRQ latency)	✓
'19	USENIX	Spoiler	✓ 1	*Probe* (IRQ latency)	✓
'19	CCS	ZombieLoad	✓ 0 - 1	*Probe* (transient exec)	✓
'19	CCS	Fallout	–	*Probe* (transient exec)	✓
'19	CCS	Tale of 2 worlds	✓ 1	*Exploit* (mem safety)	✓
'19	ISCA	MicroScope	~ 0 - Page	*Framework*	✗
'20	CHES	Bluethunder	✓ 1	*Probe* (BPU)	✓
'20	USENIX	Big troubles	~ Page	*Probe* (page fault)	✓
'20	S&P	Plundervolt	–	*Exploit* (undervolt)	✓
'20	CHES	Viral primitive	✓ 1	*Probe* (IRQ count)	✓
'20	USENIX	CopyCat	✓ 1	*Probe* (IRQ count)	✓
'20	S&P	LVI	✓ 1	*Exploit* (transient)	✓

Yr	Venue	Paper	Step	Use Case	Drv
'20	CHES	A to Z	~ Page	*Probe* (page fault)	✓
'20	CCS	Déjà Vu NSS	~ Page	*Probe* (page fault)	✓
'20	MICRO	PTHammer	–	*Probe* (page walk)	✓
'21	USENIX	Frontal	✓ 1	*Probe* (IRQ latency)	✓
'21	S&P	CrossTalk	✓ 1	*Probe* (transient exec)	✓
'21	CHES	Online template	✓ 1	*Probe* (IRQ count)	✓
'21	NDSS	SpeechMiner	–	*Framework*	✓
'21	S&P	Platypus	✓ 0 - 1	*Probe* (voltage)	✓
'21	DIMVA	Aion	✓ 1	*Probe* (cache)	✓
'21	CCS	SmashEx	✓ 1	*Exploit* (mem safety)	✓
'21	CCS	Util::Lookup	✓ 1	*Probe* (L3 cache)	✓
'22	USENIX	Rapid prototyping	✓ 1	*Framework*	✓
'22	CT-RSA	Kalyna expansion	✓ 1	*Probe* (L3 cache)	✓
'22	SEED	Enclyzer	–	*Framework*	✓
'22	NordSec	Self-monitoring	~ Page	*Defense* (detect)	✓
'22	AutoSec	Robotic vehicles	✓ 1 - >1	*Exploit* (timestamp)	✓
'22	ACSAC	MoLE	✓ 1	*Defense* (randomize)	✓
'22	USENIX	AEPIC	✓ 1	*Probe* (I/O device)	✓
'22	arXiv	Confidential code	✓ 1	*Probe* (IRQ latency)	✓
'23	ComSec	FaultMorse	~ Page	*Probe* (page fault)	✓
'23	CHES	HQC timing	✓ 1	*Probe* (L3 cache)	✓
'23	ISCA	Belong to us	✓ 1	*Probe* (BPU)	✓
'23	USENIX	BunnyHop	✓ 1	*Probe* (BPU)	✓
'23	USENIX	DownFall	✓ 0 - 1	*Probe* (transient exec)	✓
'23	USENIX	AEX-Notify	✓ 1	*Defense* (prefetch)	✓

SGX-Step: A Versatile Open-Source Attack Framework

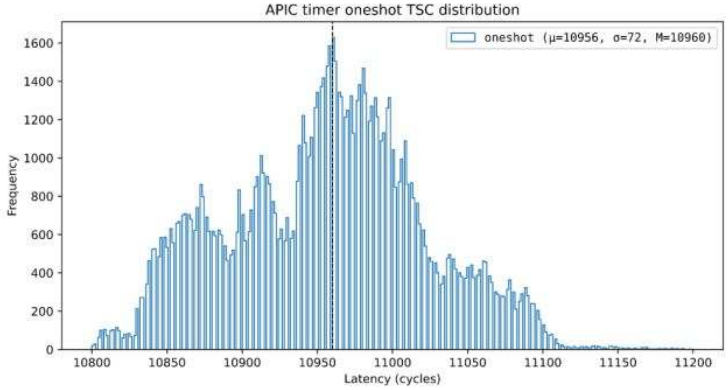


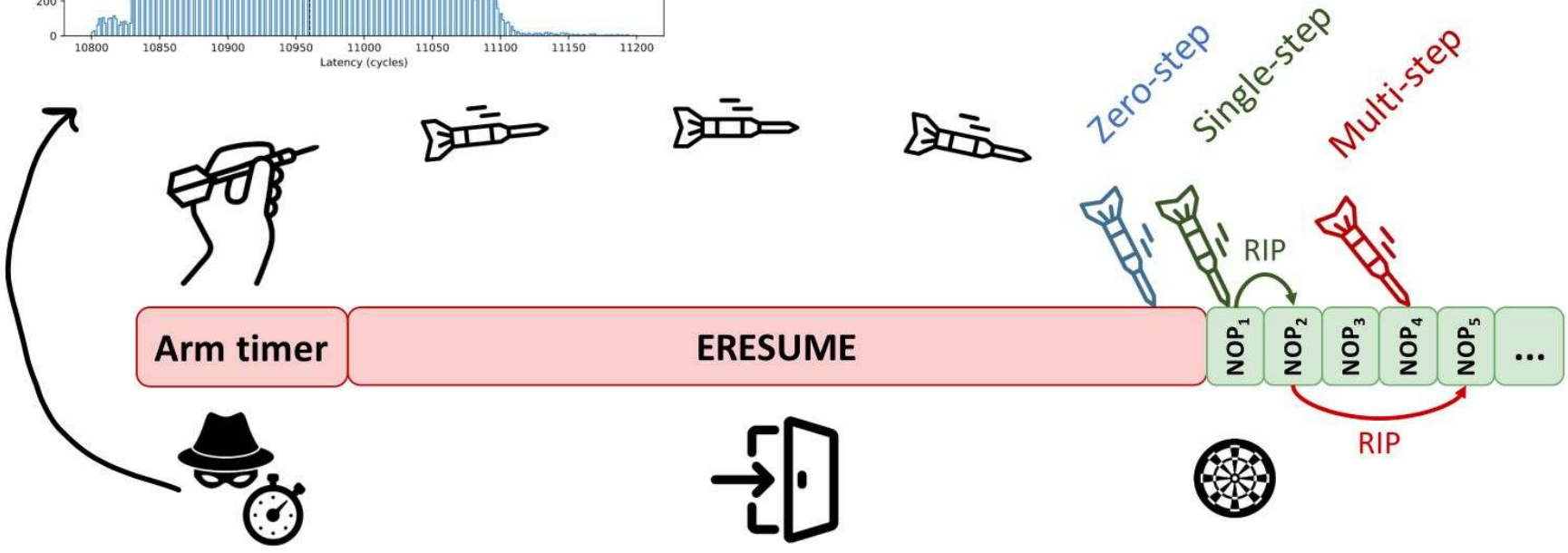
Single-Stepping Beyond Intel SGX



Based on slide from Luka Wilke.

Root-causing SGX-Step: Aiming the timer interrupt

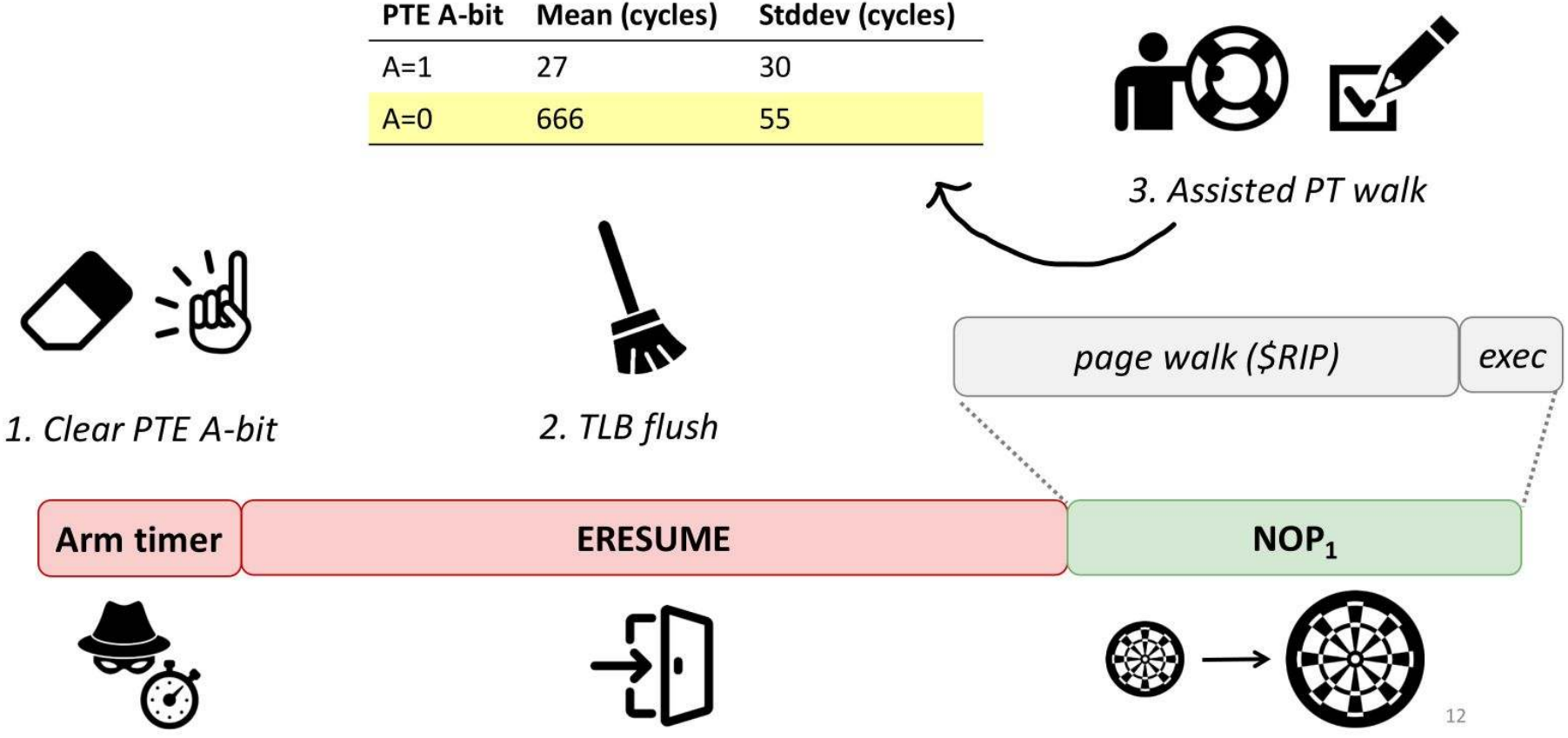




Constable et al. "AEX-Notify: Thwarting Precise Single-Stepping Attacks through Interrupt Awareness for Intel SGX Enclaves", USENIX Security 2023.

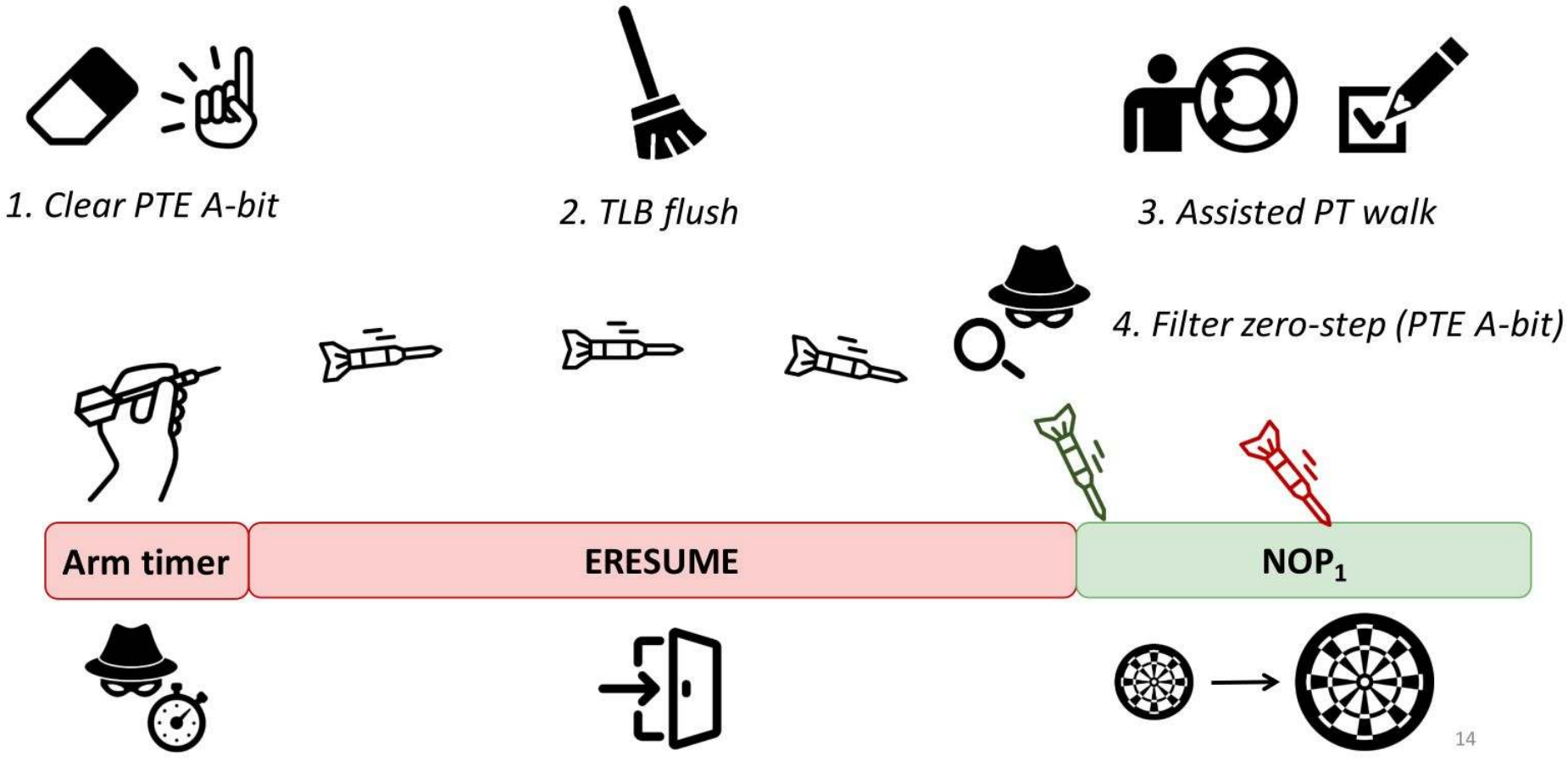
Root-causing SGX-Step: Microcode assists to the rescue!

PTE A-bit	Mean (cycles)	Stddev (cycles)
A=1	27	30
A=0	666	55



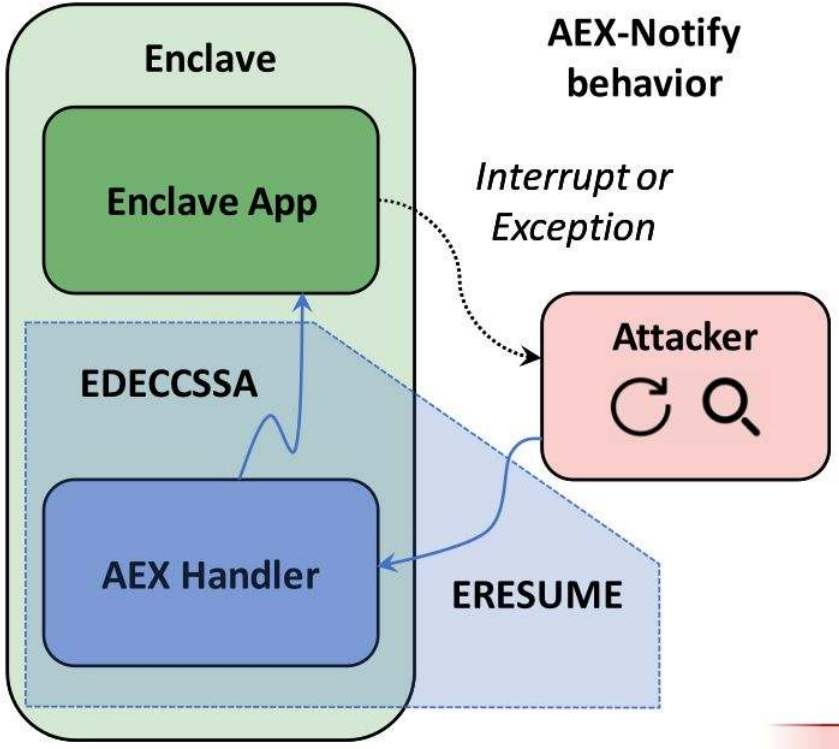
Constable et al. “AEX-Notify: Thwarting Precise Single-Stepping Attacks through Interrupt Awareness for Intel SGX Enclaves”, USENIX Security 2023.

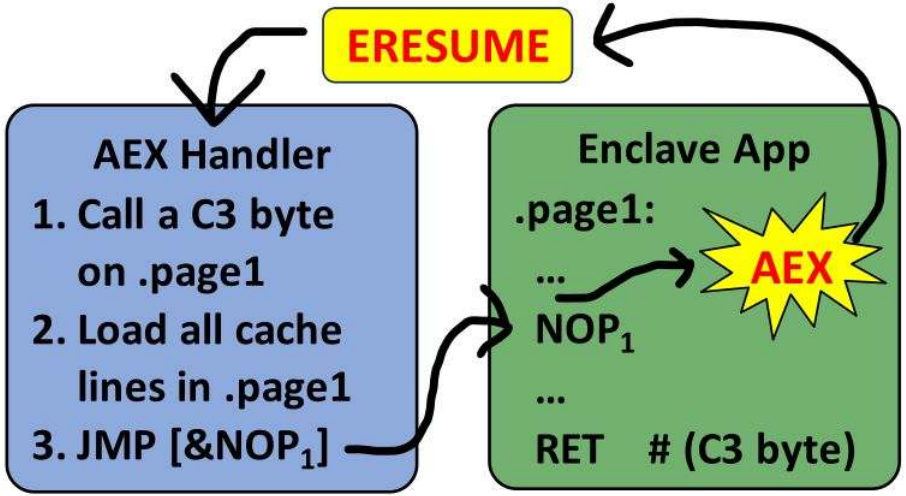
Root-causing SGX-Step: Microcode assists to the rescue!

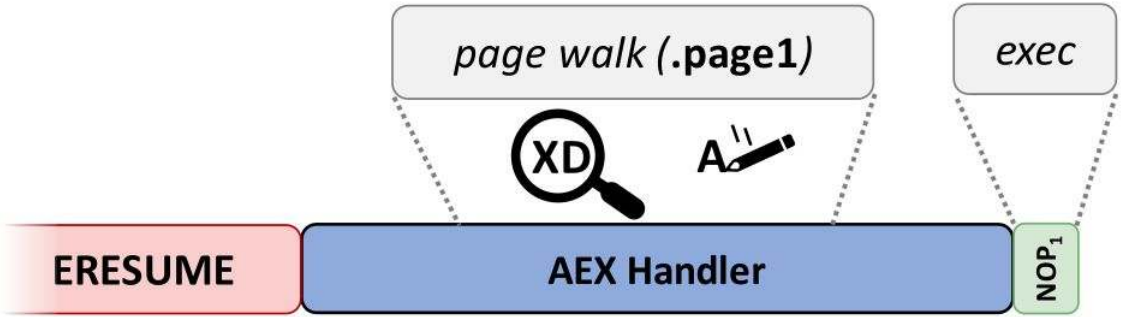


Constable et al. “AEX-Notify: Thwarting Precise Single-Stepping Attacks through Interrupt Awareness for Intel SGX Enclaves”, USENIX Security 2023.

AEX-Notify: Hardware-Software Co-Design Solution



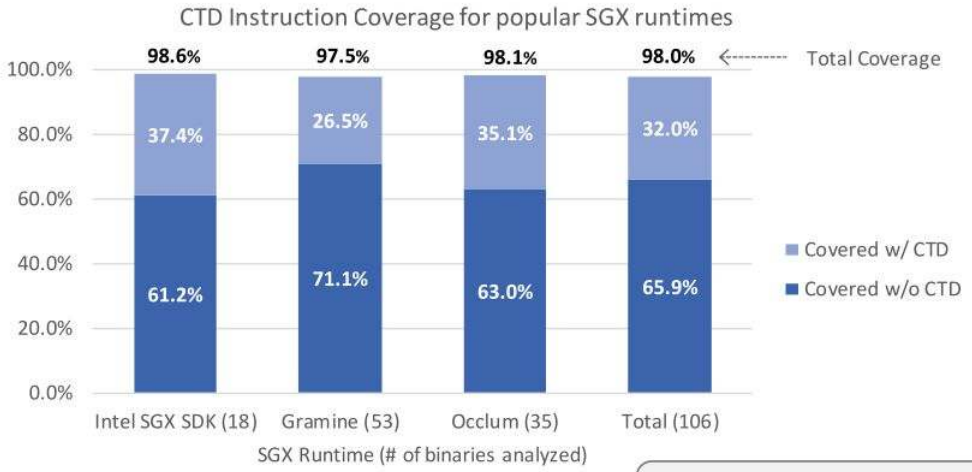


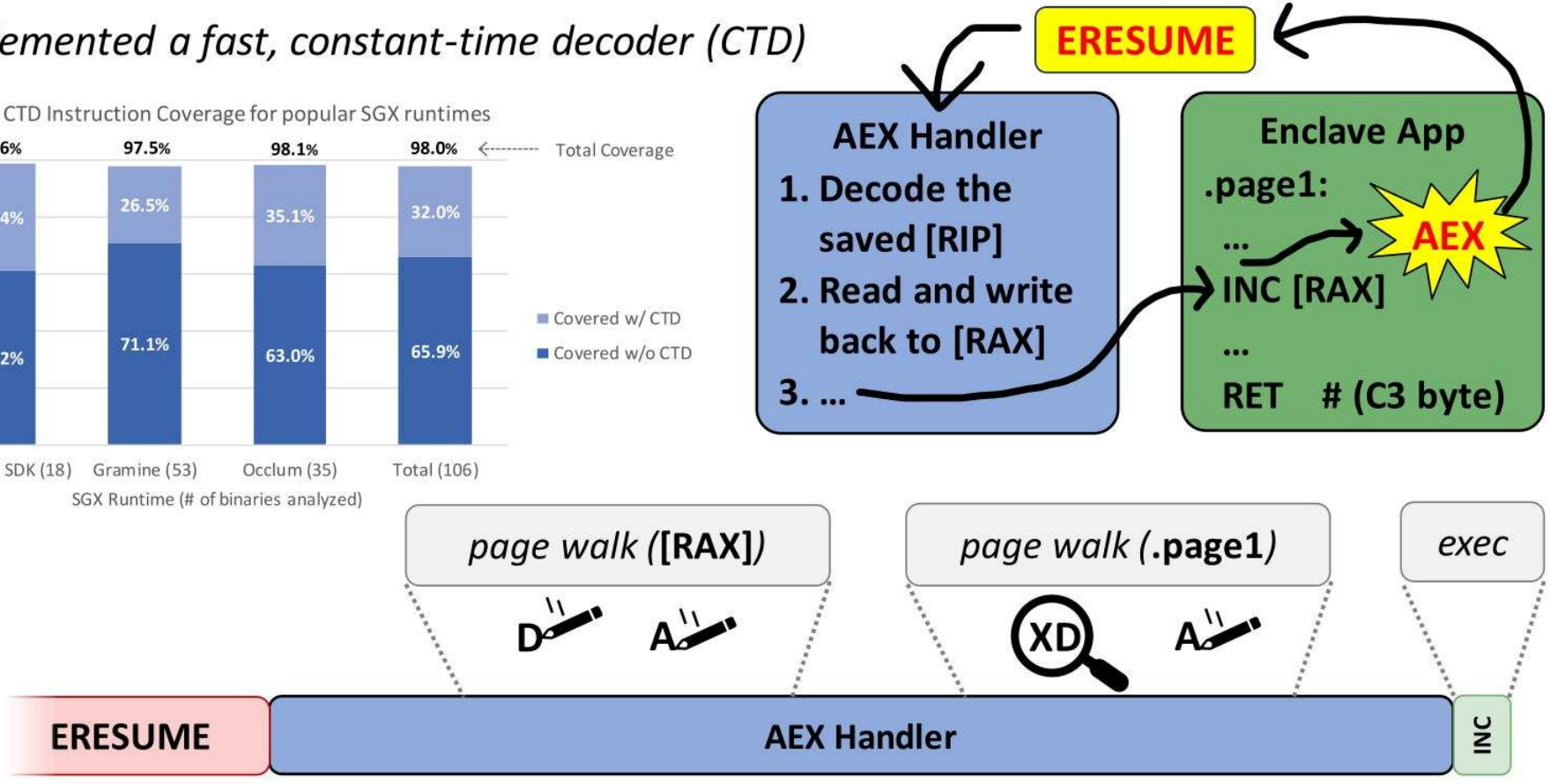


Constable et al. "AEX-Notify: Thwarting Precise Single-Stepping Attacks through Interrupt Awareness for Intel SGX Enclaves", USENIX Security 2023.

AEX-Notify: Hardware-Software Co-Design Solution

We implemented a fast, constant-time decoder (CTD)





Constable et al. "AEX-Notify: Thwarting Precise Single-Stepping Attacks through Interrupt Awareness for Intel SGX Enclaves", USENIX Security 2023.

intel.

CHAPTER 8
ASYNCHRONOUS ENCLAVE EXIT NOTIFY AND THE EDECCSSA USER LEAF FUNCTION

8.1 INTRODUCTION

Asynchronous Enclave Exit Notify (AEX-Notify) is an extension to Intel® SGX that allows Intel SGX enclaves to be notified after an asynchronous enclave exit (AEX) has occurred. EDECCSSA is a new Intel SGX user leaf function (ENCLU[EDECCSSA]) that can facilitate AEX notification handling, as well as software exception handling. This chapter provides information about changes to the Intel SGX architecture that support AEX-Notify and ENCLU[EDECCSSA].

The following list summarizes the a
details are provided in Section 8.3)

- SECS.ATTRIBUTES.AEXNOTIFY
- TCS.FLAGS.AEXNOTIFY: This e
- SSA.GPRSGX.AEXNOTIFY: Enclave-writable byte that allows enclave software to dynamically enable/disable AEX notifications.

An AEX notification is delivered by ENCLU[ERESUME] when the following conditions are met:

SGX-Step led to **new x86 processor instructions!**

→ shipped in millions of devices ≥ 4th Gen Xeon CPU

phoronix

ARTICLES & REVIEWS | NEWS ARCHIVE | FORUMS | PREMIUM | CONTACT | CATEGORIES

Intel AEX Notify Support Prepped For Linux To Help Enhance SGX Enclave Security

Written by Michael Larabel in Intel on 6 November 2022 at 06:01 AM EST. 5 Comments

Future Intel CPUs and some existing processors via a microcode update will support a new feature called the Asynchronous EXit (AEX) notification mechanism to help with Software Guard Extensions (SGX) enclave security. Patches for the Linux kernel are pending for implementing this Intel AEX Notify support with capable processors.

Intel's Asynchronous EXit (AEX) notification mechanism lets SGX enclaves run a handler after an AEX event. Those handlers can be used for things like mitigating SGX-Step as an attack framework for precise enclave execution control.

SGX-Step

Code 1 in intel/linux-sgx Filter

sdk/trts/linux/trts_mitigation.S

```
48  * Description:
49  *     The file provides mitigations for SGX-Step
50  */
71  *  Function: constant_time_apply_sgxstep_mitigation_and_continue_execution
72  *      Mitigate SGX-Step and return to the point at which the most recent
73  *      interrupt/exception occurred.
```

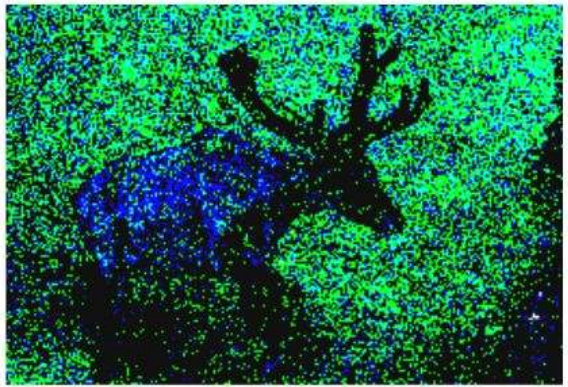
SGX-Step led to **changes in major OSs and enclave SDKs**

There’s a Catch…

Finally note that our proposed mitigation does not protect against interrupting enclaves and observing application code and data page accesses at a coarse-grained 4 KiB spatial resolution. In contrast to the fine-grained, instruction-granular interrupt-driven attacks we consider in this work, such controlled-channel attacks have received ample attention [18, 47, 56, 59] from the research community.

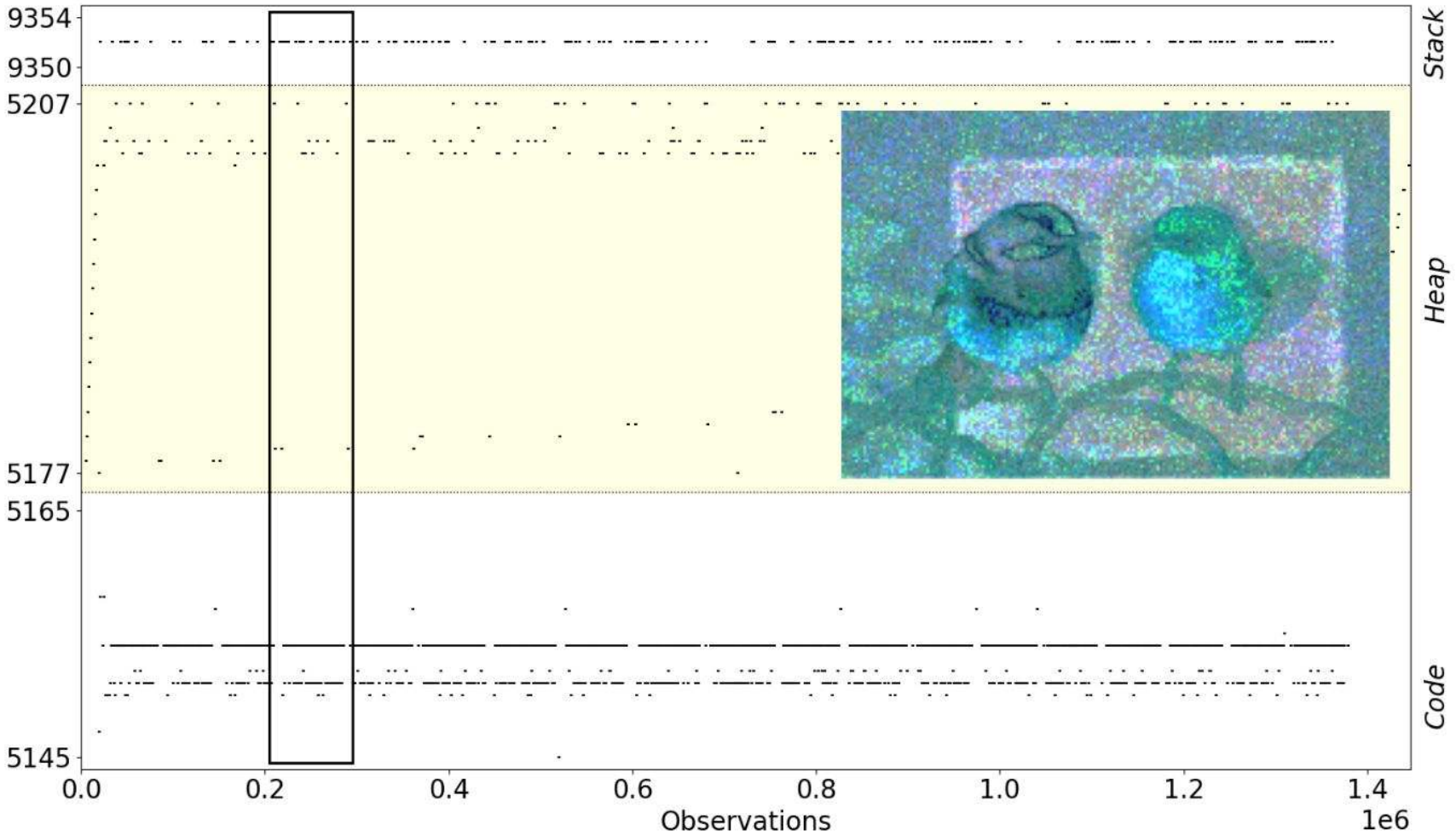
Constable et al. “AEX-Notify: Thwarting Precise Single-Stepping Attacks through Interrupt Awareness for Intel SGX Enclaves”, USENIX Security 2023.

Why Mitigating Single-Stepping is Not Enough

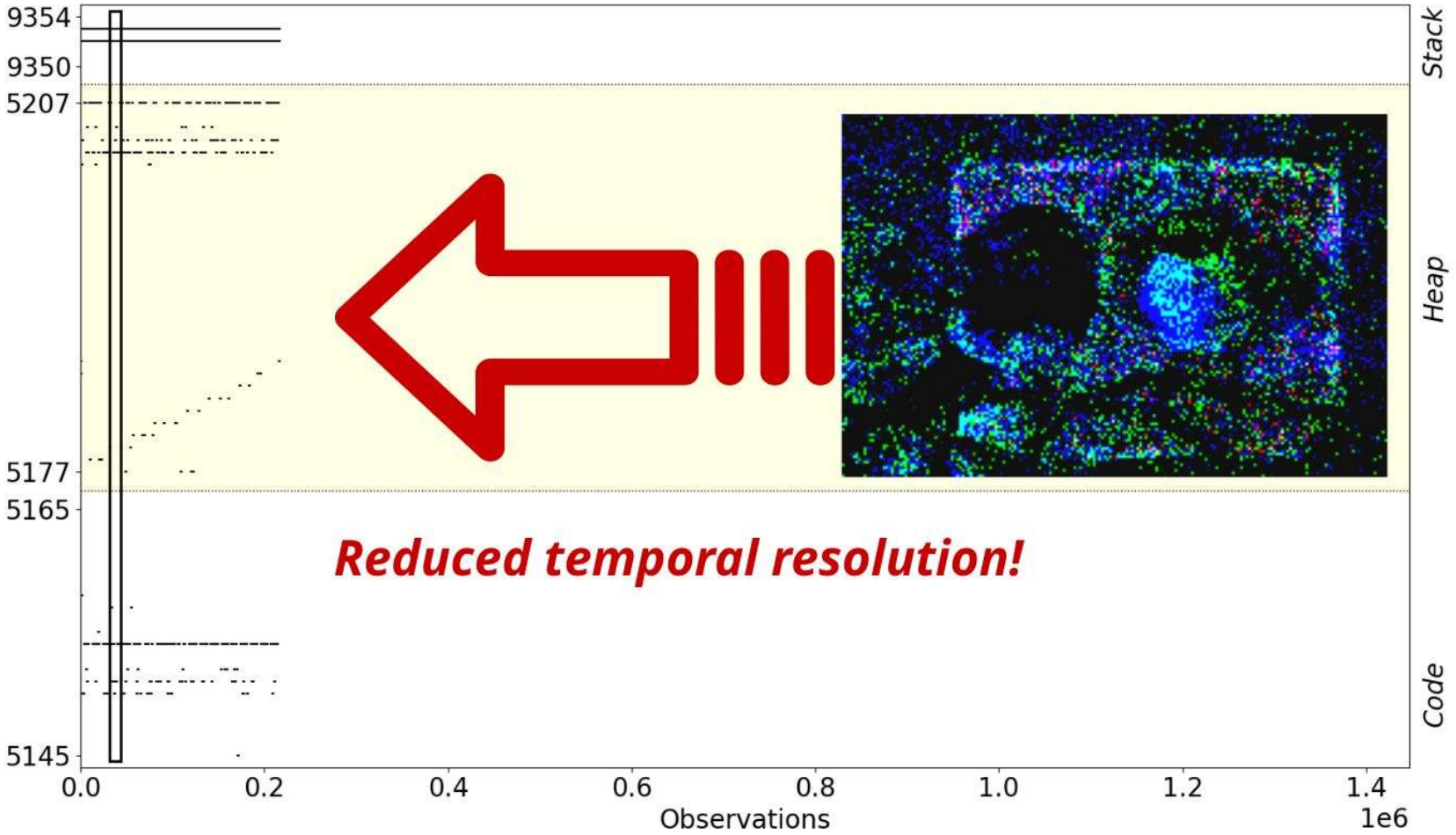


Original (left), Xu et al. (middle), our attack with AEX-Notify single-stepping defense (right)

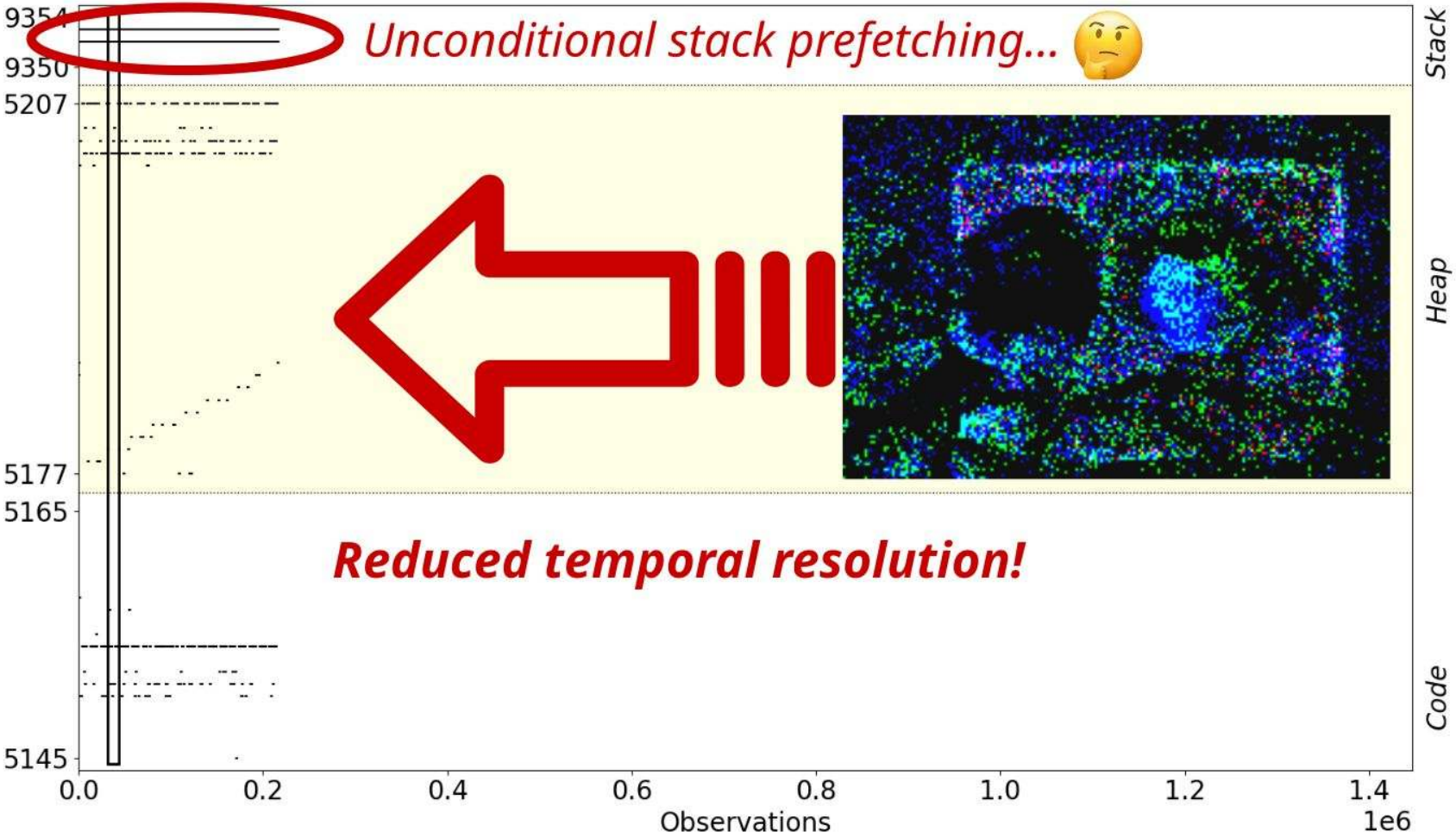
Libjpeg: AEX-Notify's Temporal Reduction in Practice



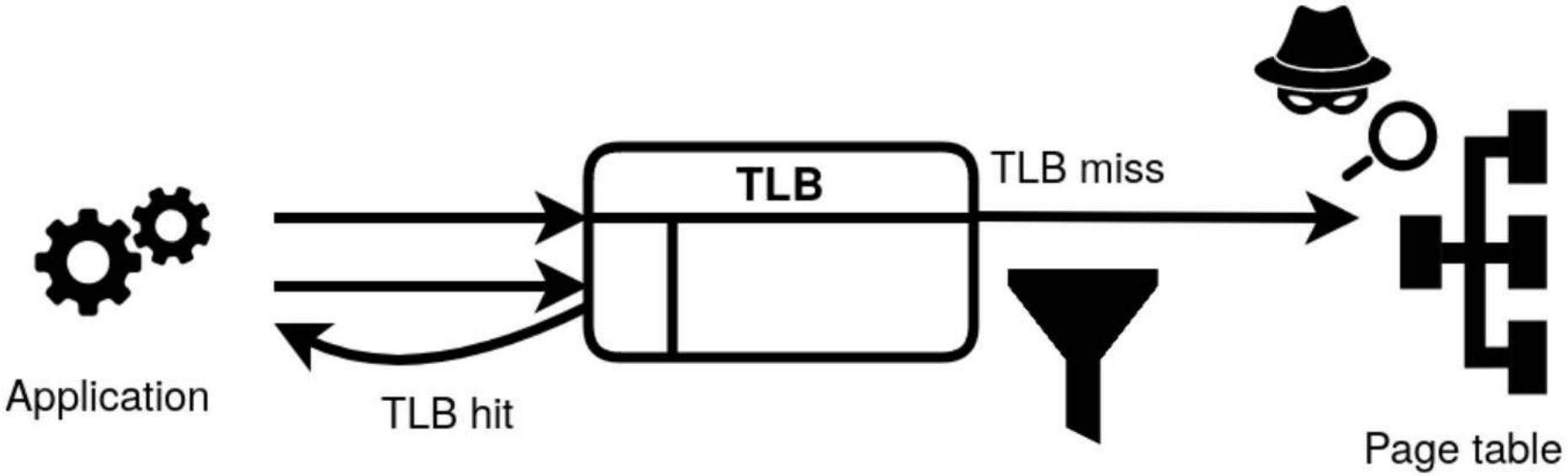
Libjpeg: AEX-Notify's Temporal Reduction in Practice



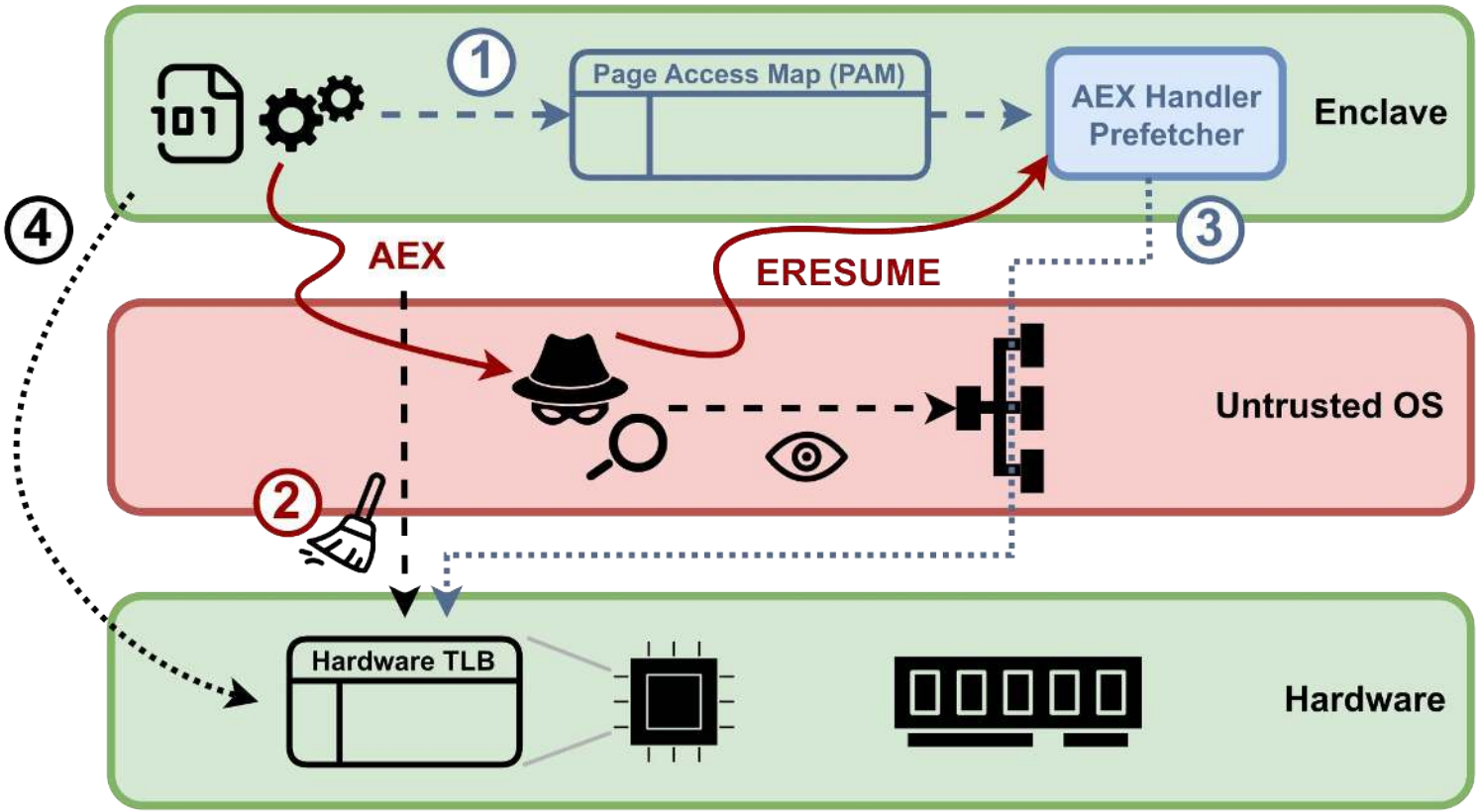
Libjpeg: AEX-Notify's Temporal Reduction in Practice



Idea: TLB as a “Filter” to Hide Page Accesses

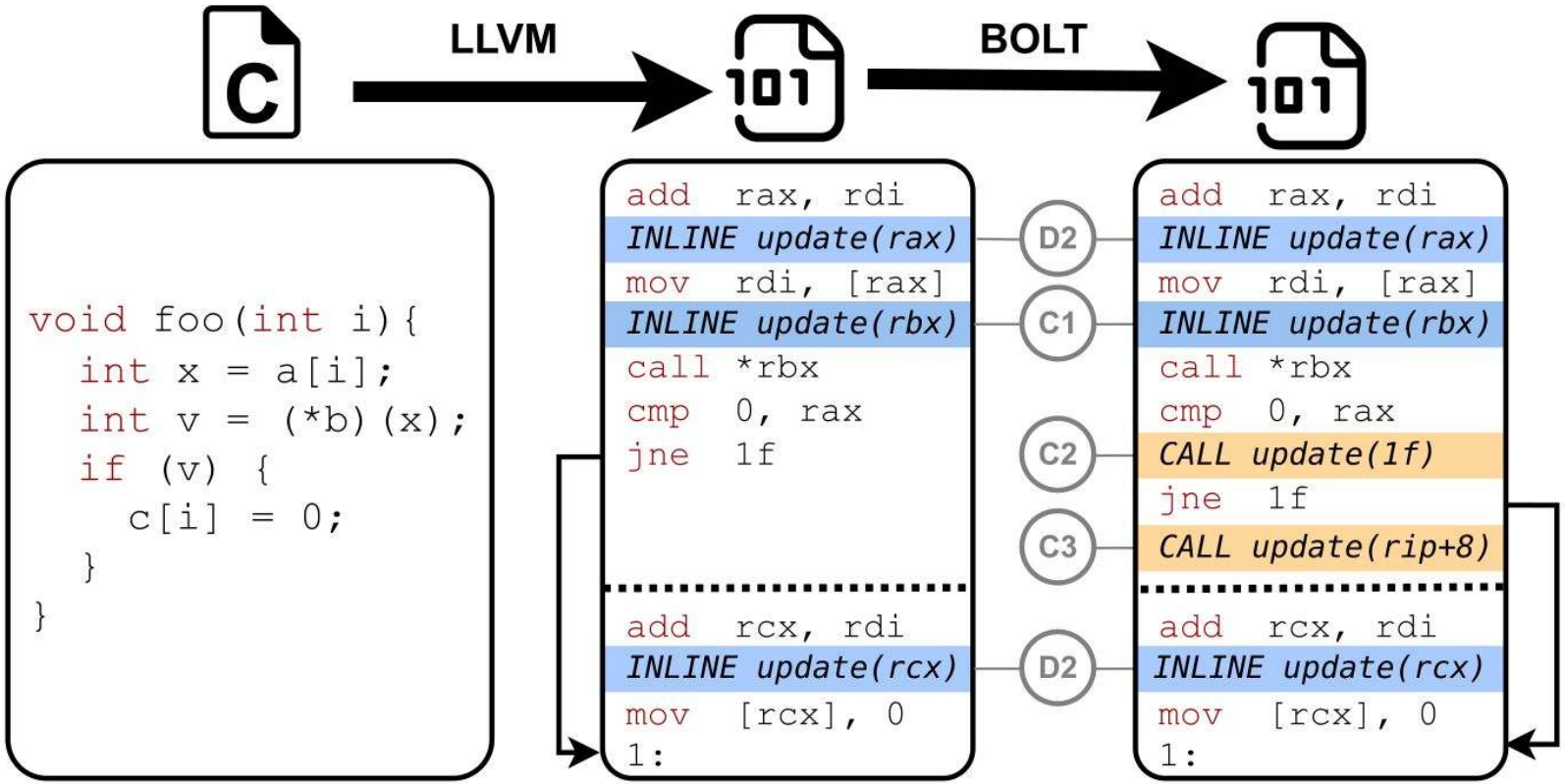


TLBlur: Self-Monitoring and Restoring Enclave Page Accesses



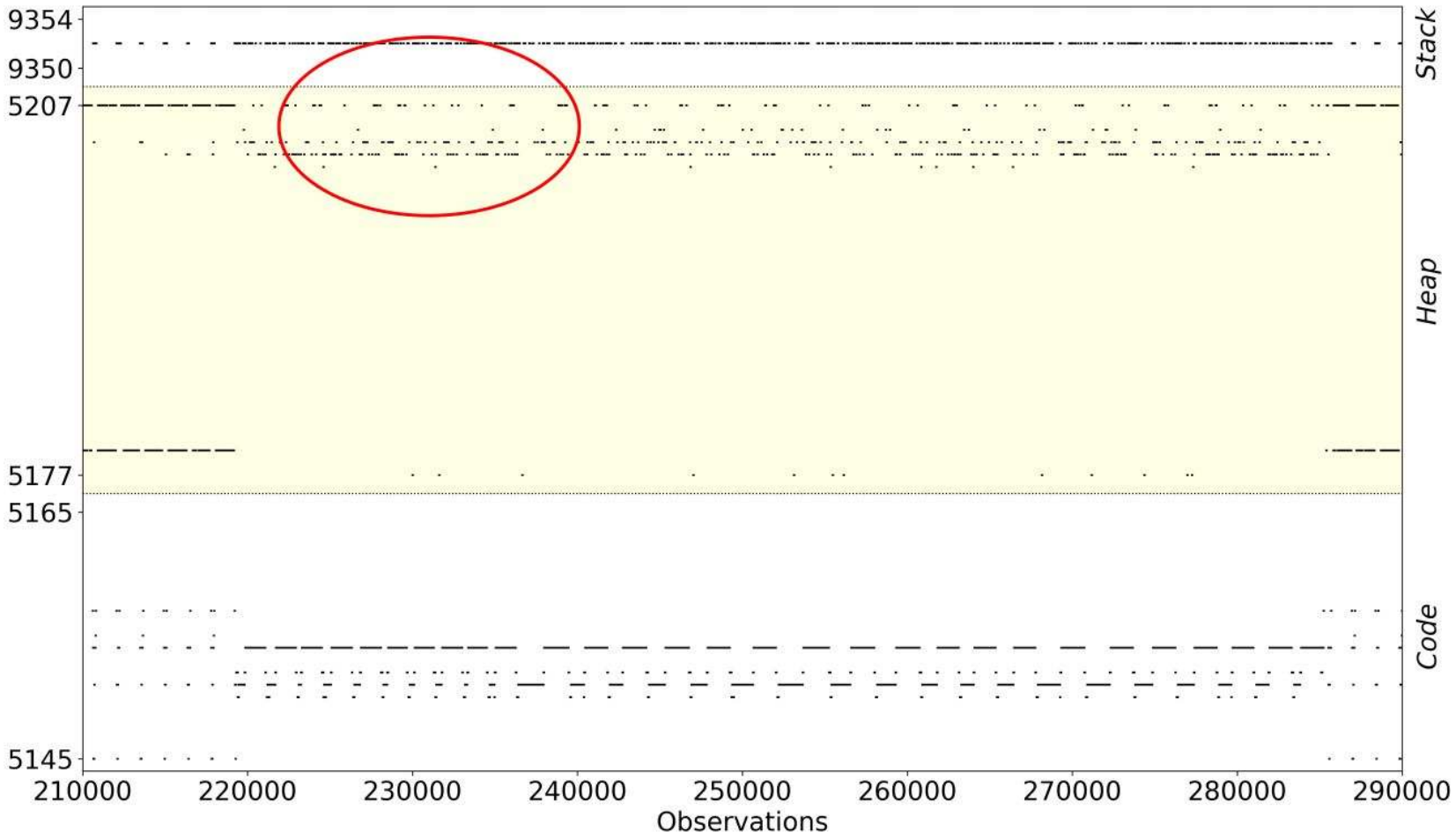
Vanoverloop et. al "TLBlur: Compiler-Assisted Automated Hardening against Controlled Channels on Off-the-Shelf Intel SGX Platforms", USENIX'25.

Instrumentation to Self-Monitor Page Accesses at Runtime

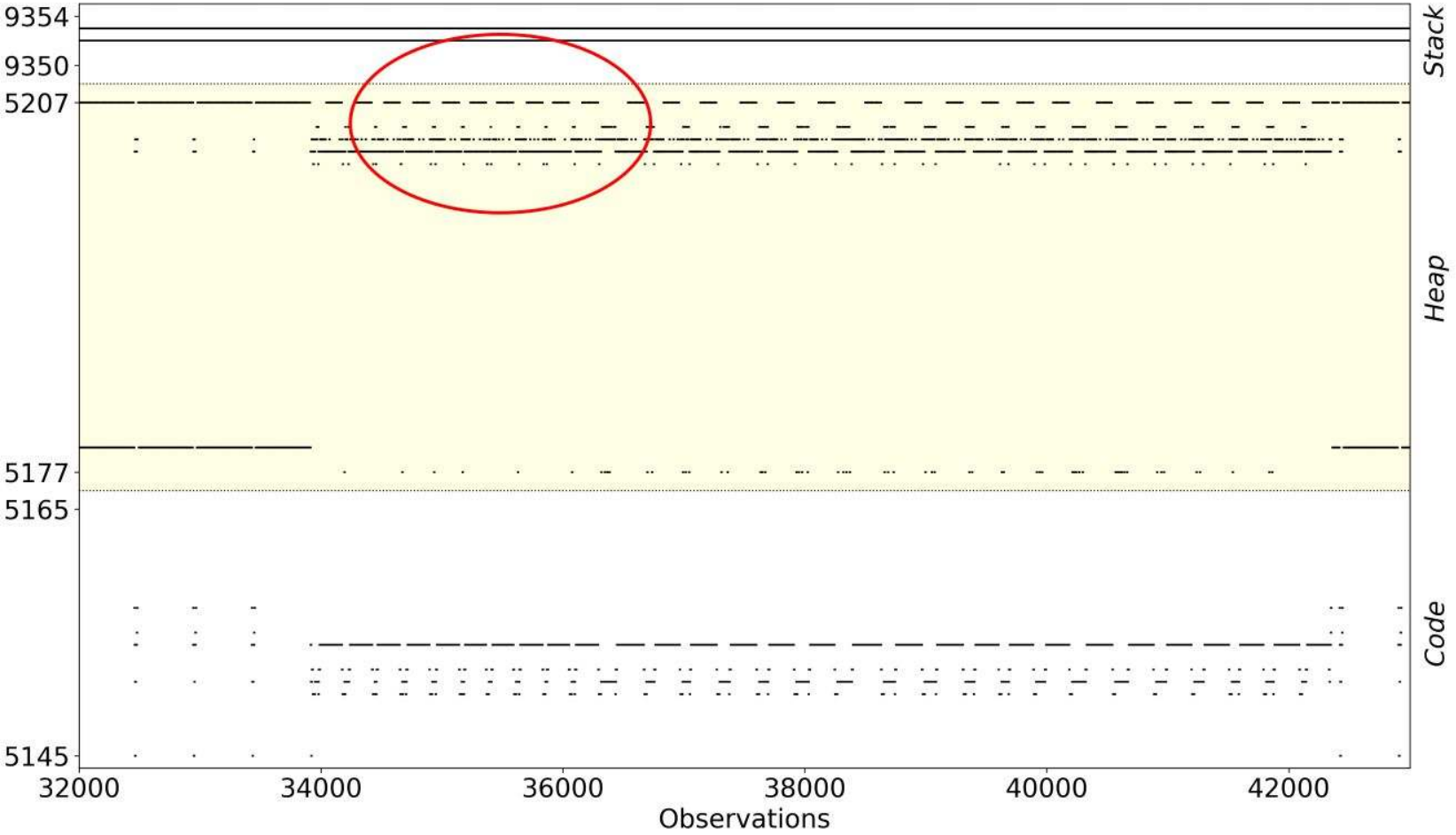


Vanoverloop et. al “TLBlur: Compiler-Assisted Automated Hardening against Controlled Channels on Off-the-Shelf Intel SGX Platforms”, USENIX’25.

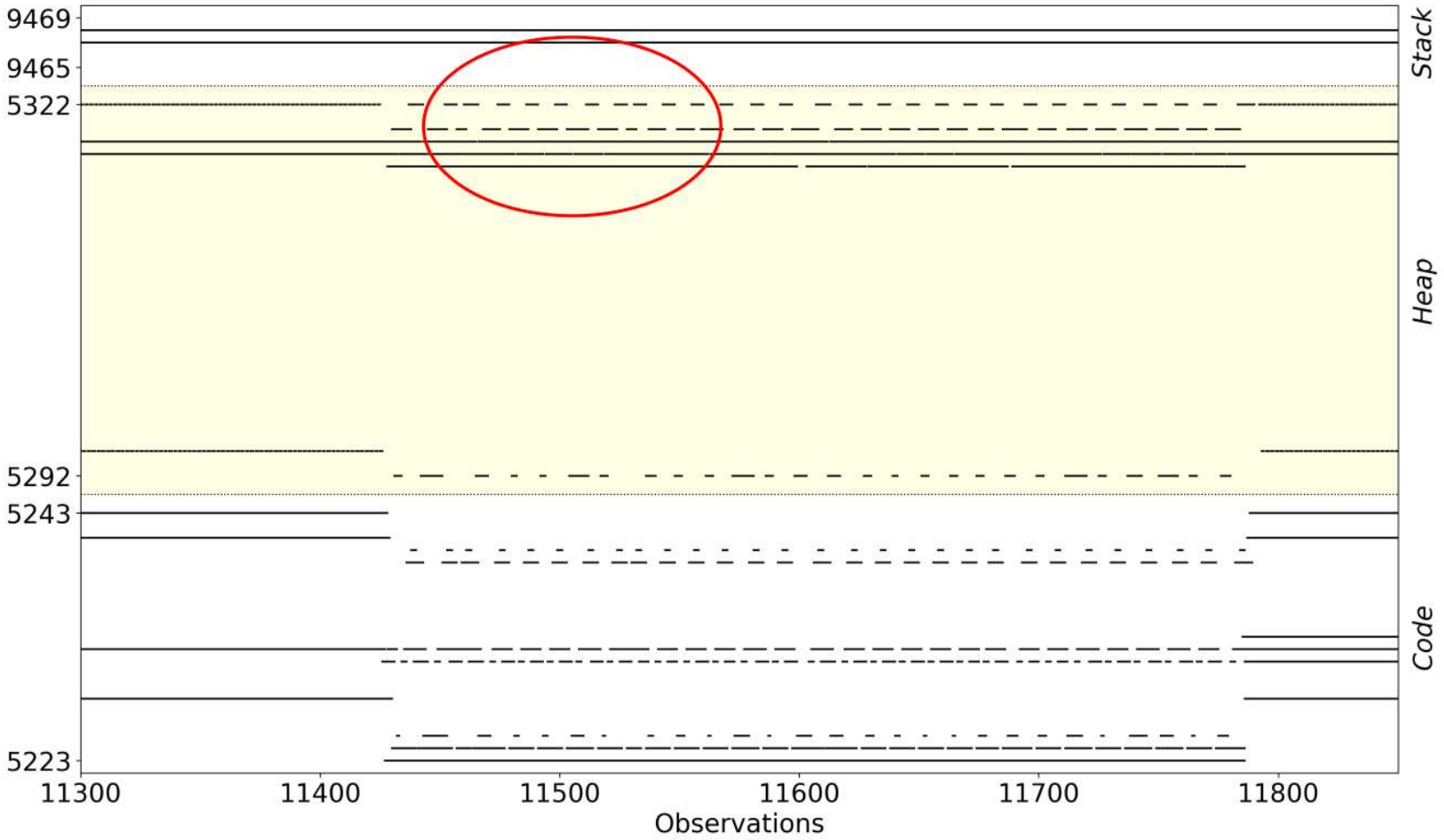
Leakage Reduction in Practice: Libjpeg Single-Stepping



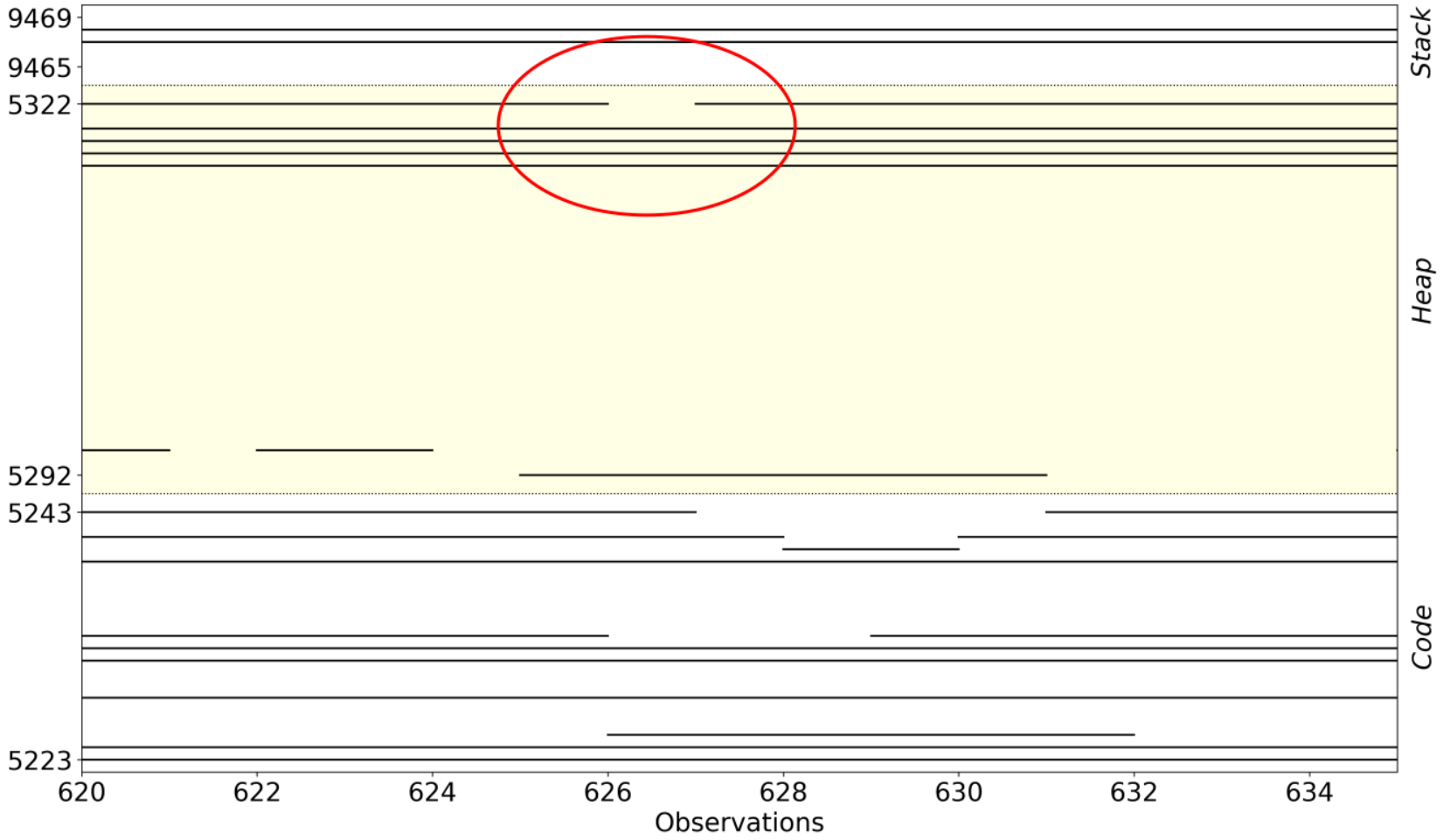
Leakage Reduction in Practice: Libjpeg Page Faults



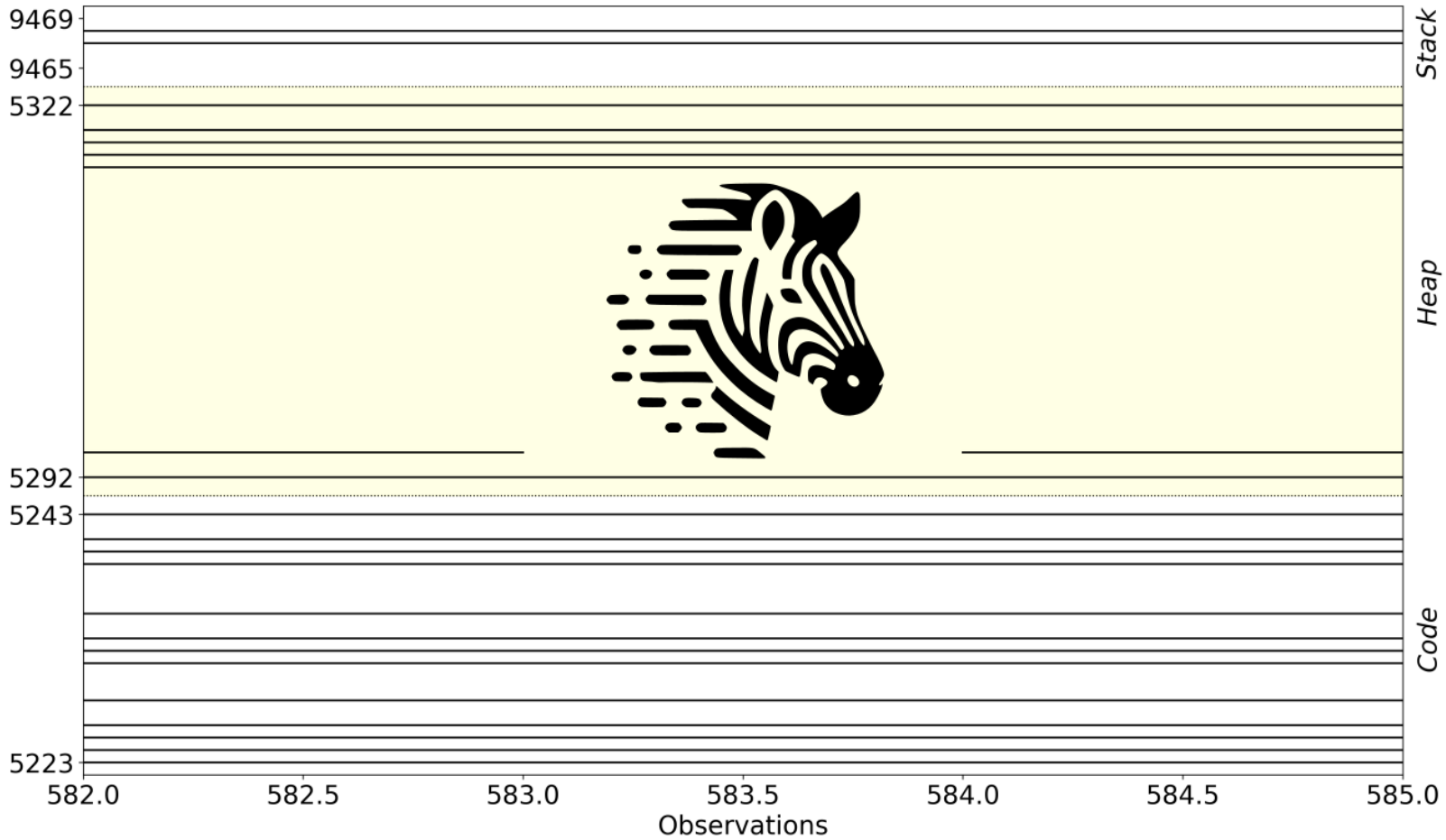
Leakage Reduction in Practice: Libjpeg TLBlur (N=10)



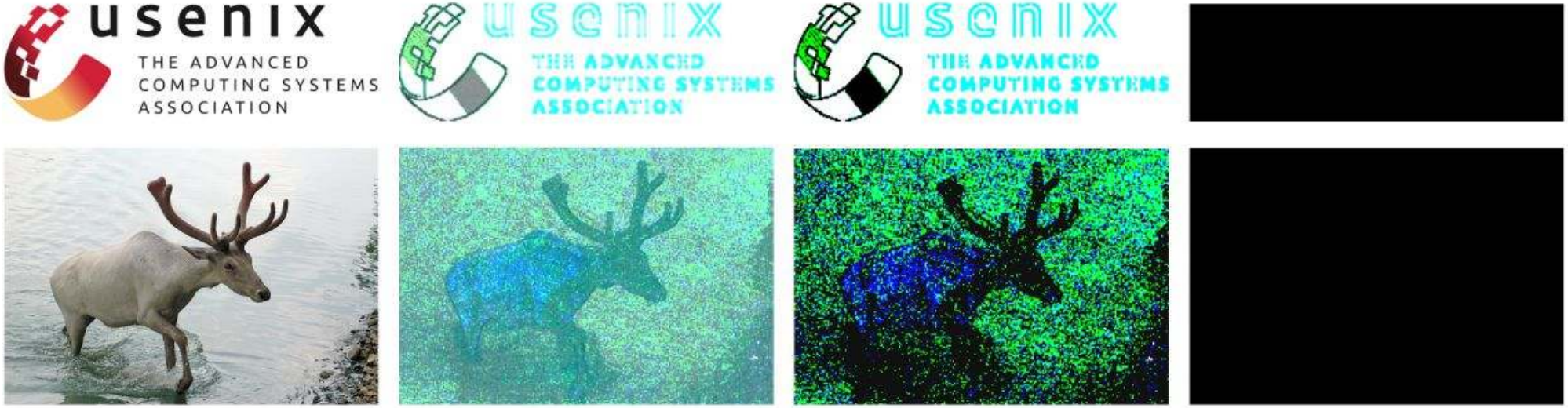
Leakage Reduction in Practice: Libjpeg TLBlur (N=20)



Leakage Reduction in Practice: Libjpeg TLBlur (N=30)



TLBlur: Compiler-Assisted Leakage Reduction in Practice



Automated "blurring" of page-access traces in space and time

Vanoverloop et. al "TLBlur: Compiler-Assisted Automated Hardening against Controlled Channels on Off-the-Shelf Intel SGX Platforms", USENIX'25.

Conclusions and Take-Away

New era of **confidential computing** for the cloud and IoT

… but current architectures are **not perfect!**

Scientific understanding driven by **attacker-defender race**

Thank you!