Reflections on Trusting Trusted Execution

The Story of Microarchitectural Attacks and Defenses

Jo Van Bulck

COSIC Course on Cryptography and Cyber Security, Leuven, July 4, 2024

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

DistriNet
KU LEUVEN

Fast and Efficient Implementation of Homomorphic Encryption?

9:00-9:30	Hardware security: state of the art	Ingrid Verbauwhede, COSIC
9:30-10:30	Homomorphic Encryption (focus point to be added)	Jan-Pieter D'Anvers, COSIC
10:30-11:00	Coffee break (Landbouwinstituut Hoofdgebouw)	
11:00-11:45	Post-quantum cryptography: NIST standardization, Present and Future	Angshuman Karmakar, IIT Kanpur
11:45-12:30	Microarchitectural attacks	Jo Van Bulck, DistriNet
12:30-14:00	Lunch	
14:00-15:00	Side Channel + Lattice Based Systems	Elisabeth Oswald + Matthias Steiner, University of Klagenfurt
15:00-16:00	Homomorphic Encryption: the practical side	Johannes Mono, Ruhr University Bochum
16:00-16:30	Coffee break (Landbouwinstituut Hoofdgebouw)	
16:30-17:30	Fast GPU implementation of the BFV and CKKS homomorphic encryption schemes	Erkay Savas, Sabanci University

Fast and Efficient Implementation of Homomorphic Encryption?

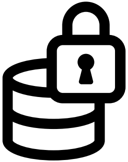
9:00-9:30	Hardware security: state of the art	Ingrid Verbauwhede, COSIC
9:30-10:30	Homomorphic Encryption (focus point to be added)	Jan-Pieter D'Anvers, COSIC
10:30-11:00	Coffee break (Landbouwinstituut Hoofdgebouw)	
11:00-11:45	Post-quantum cryptography: NIST standardization, Present and Future	Angshuman Karmakar, IIT Kanpur
15:00-16:00	Homomorphic Encryption: the practical side	Johannes Mono, Ruhr University Bochum
16:00-16:30	Coffee break (Landbouwinstituut Hoofdgebouw)	
16:30-17:30	Fast GPU implementation of the BFV and CKKS homomorphic encryption schemes	Erkay Savas, Sabanci University

How does today's topic fit in?

The Big Picture: Protecting Private Data

Data in transit

Data in use



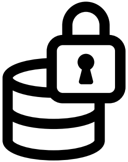
Data at rest

The Big Picture: Protecting Private Data

Data in transit

- ✓ SSL/TLS etc.

Data in use



Data at rest

- ✓ Full disk encryption

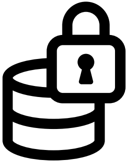
The Big Picture: Protecting Private Data

Data in transit

- SSL/TLS etc.

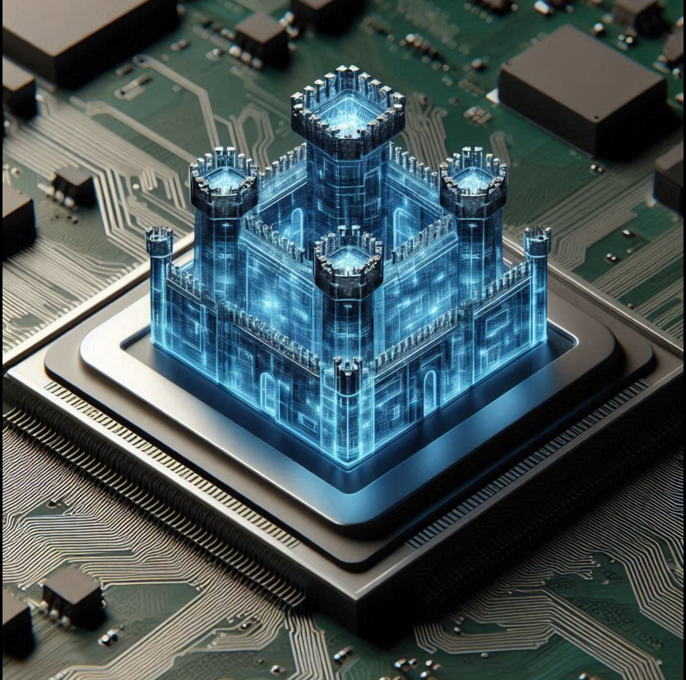
Data in use

- **Homomorphic encryption?**
- **Trusted Execution?**
  - = *Confidential Computing*
  - = *Hardware Enclaves*

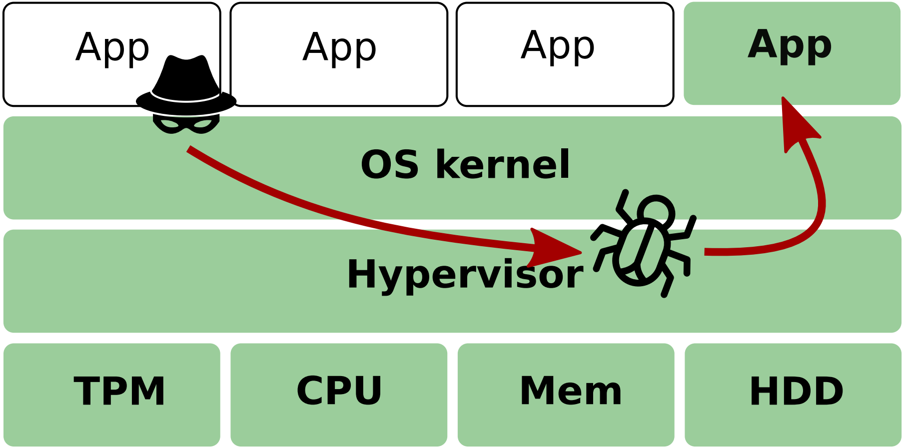


Data at rest

- Full disk encryption

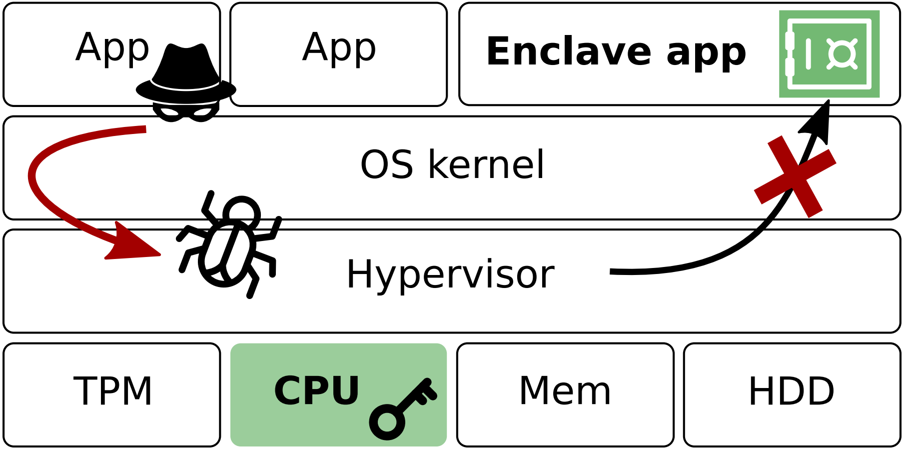

DALL·E 3

The Big Picture: Reducing Attack Surface with Enclaves



Traditional **layered designs:** Large trusted computing base

The Big Picture: Reducing Attack Surface with Enclaves



Intel SGX promise: Hardware-level **isolation and attestation**

The Rise of Trusted Execution Environments

- 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)
- 2024: ARM Confidential Computing Architecture (CCA)

The Rise of Trusted Execution Environments

- 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)
- 2024: ARM Confidential Computing Architecture (CCA)

TEEs are here to stay...

Hardware Enclaves vs. Homomorphic Encryption?

Confidential Computing is available in production today. It provides practical, useful protections for data in use and in a few years, we should see Homomorphic Encryption become available for production

🌐 https://confidentialcomputing.io/2023/03/29/confidential-computing-and-homomorphic-encryption/

Hardware Enclaves vs. Homomorphic Encryption?

	Homomorphic Encryption	Confidential Computing
Data Integrity	X	√
Data Confidentiality	√	√
Code Integrity	X	√
Code Confidentiality	X	√
Authenticated Launch	X	varies
Attestability	X	√
Recoverability	X	√

Confidential Computing is already in active use, while Homomorphic Encryption is still in the experimentation phase

https://www.edgeless.systems/blog/the-differences-between-homomorphic-encryption-he-the-differences-between-homomorphic

Hardware Enclaves vs. Homomorphic Encryption?

	Homomorphic Encryption	Confidential Computing
Data Integrity	X	√
Data Confidentiality	√	√
Code Integrit	[illegible]	[illegible]
Code Confide	[illegible]	[illegible]
Authenticated Launch	X	varies
Attestability	X	√
Recoverability	X	√

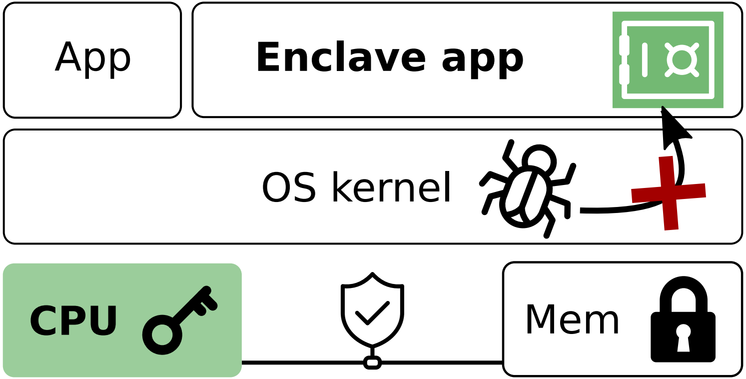
Mathematical guarantees!

Real-world implementation?

Confidential Computing is already in active use, while Homomorphic Encryption is still in the experimentation phase

https://www.edgeless.systems/blog/the-differences-between-homomorphic-encryption-he-the-differences-between-homomorphic

Overview: Architectural Enclave Isolation

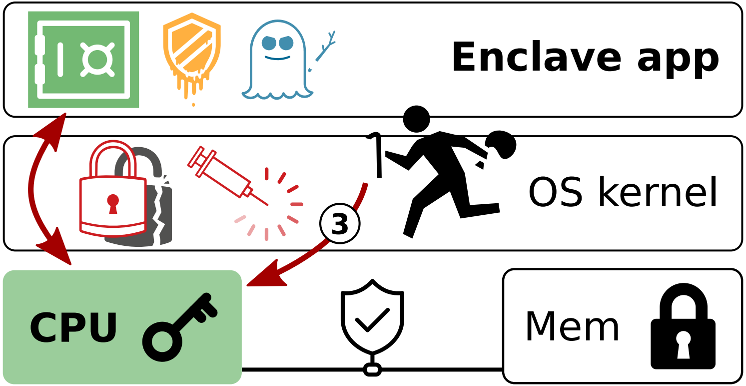


Architectural promise: Transparent data-in-use protection against privileged software adversaries

Overview: Microarchitectural Side-Channel Attacks (today)

Microarchitectural reality: Novel side channels to spy on enclave-CPU interaction metadata

Overview: Transient-Execution Attacks (not today)



Microarchitectural reality: Direct data extraction via transient-execution attacks...

DALL·E 3

A Note on SGX Side-Channel Attacks (Intel)

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.

software.intel.com/en-us/node/703016

Vulnerable Patterns: Secret-Dependent Code/Data Accesses

```c
void secret_vote(char candidate)
{
    if (candidate == 'a')
        vote_candidate_a();
    else
        vote_candidate_b();
}
```

```c
int secret_lookup(int s)
{
    if (s > 0 && s < ARRAY_LEN)
        return array[s];
    return -1;

}
```

Vulnerable Patterns: Secret-Dependent Code/Data Accesses

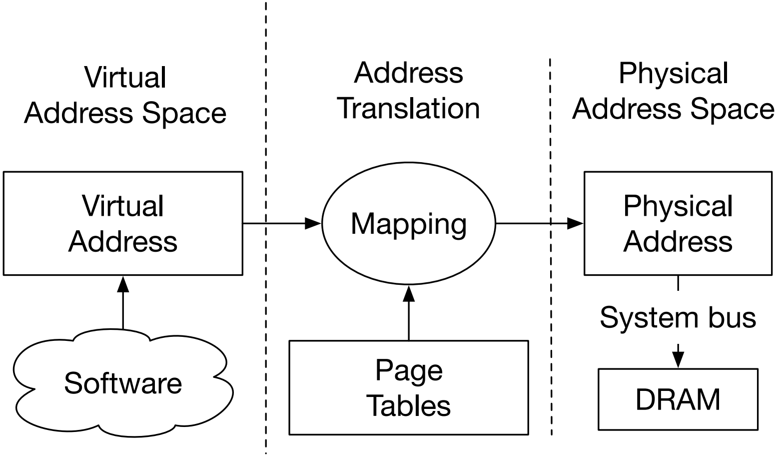
```
void secret_vote(char candidate)
{
    if (candidate == 'a')
        vote_candidate_a();
    else
        vote_candidate_b();
}
```

```
int secret_lookup(int s)
{
    if (s > 0 && s < ARRAY_LEN)
        return array[s];
    return -1;

}
```

What are new ways for privileged adversaries to create an "oracle" for enclave code+data memory accesses?

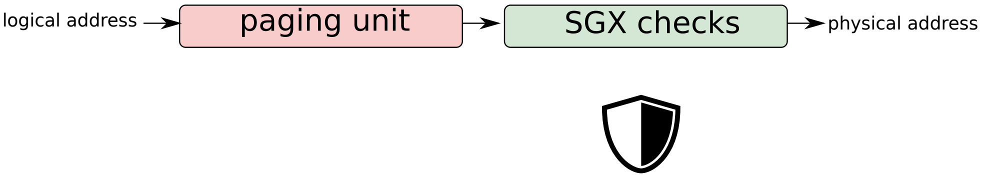
Idea #1: Monitoring Address Translation

The Virtual Memory Abstraction



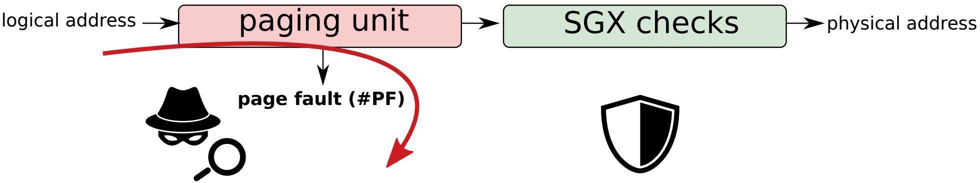
Costan et al. "Intel SGX explained", IACR 2016.

Intel SGX: Page Faults as a Side Channel



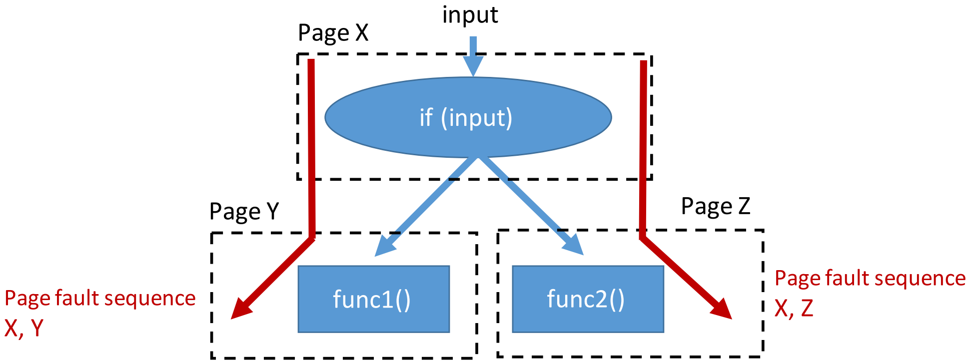
SGX machinery protects against direct address remapping attacks

Intel SGX: Page Faults as a Side Channel



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

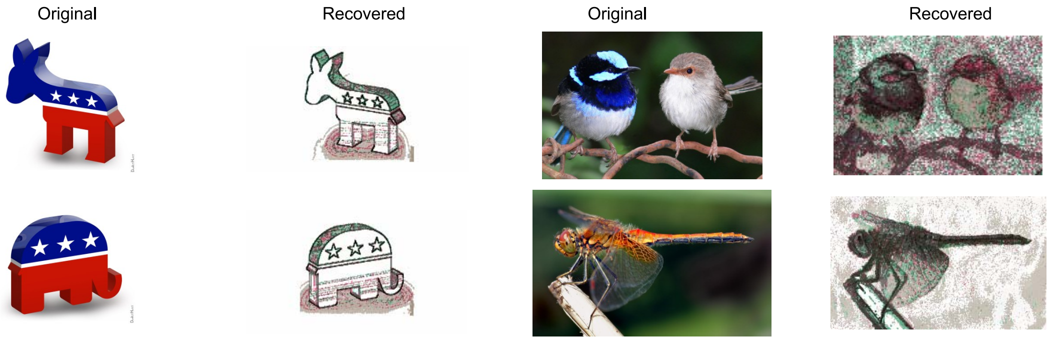
Intel SGX: Page Faults as a Side Channel



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

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

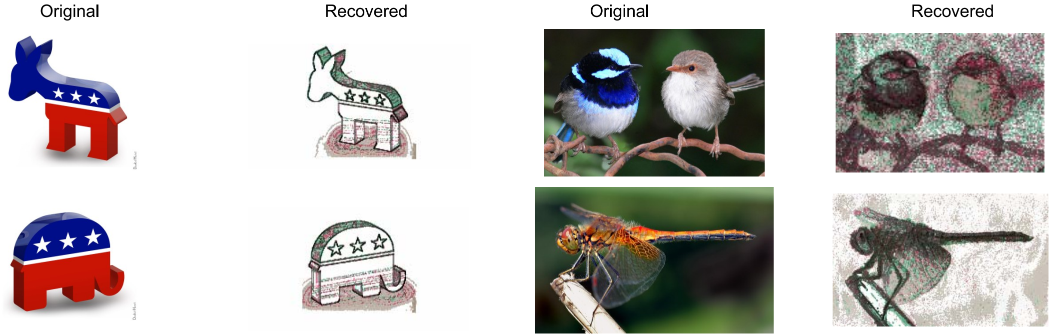
Page Table-Based Attacks in Practice



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

⇒ **Low-noise, single-run** exploitation of legacy applications

Page Table-Based Attacks in Practice



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

. . . but a coarse-grained **4 KiB spatial granularity**

Idea #2: Improving Temporal Resolution

Intel's Note on Side-Channel Attacks (Revisited)

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.

software.intel.com/en-us/node/703016

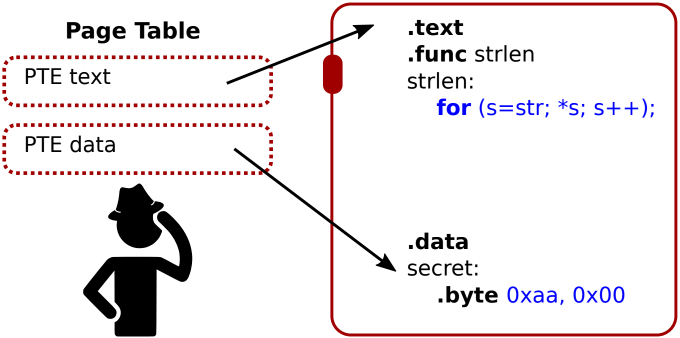
Temporal Resolution Limitations for the Page-Fault Oracle

```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

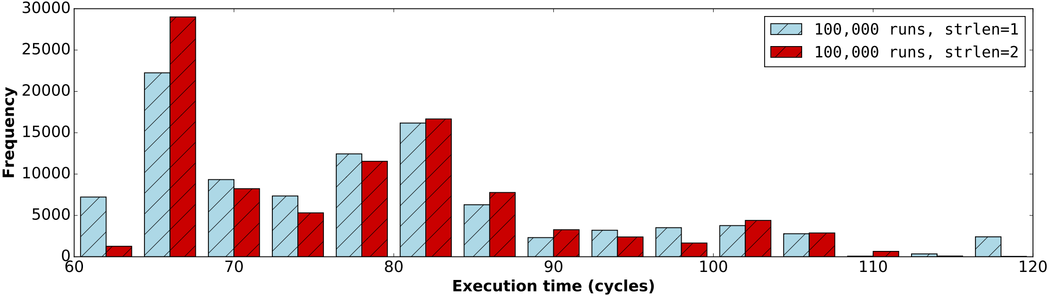
Temporal Resolution Limitations for the Page-Fault Oracle



Counting `strlen` loop iterations?

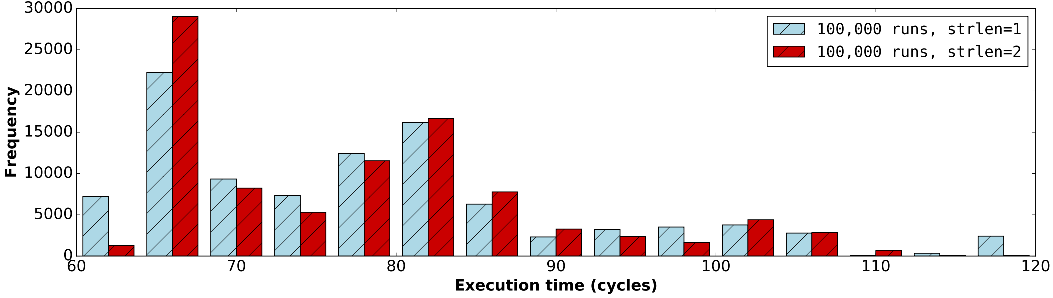
Progress requires both pages present (non-faulting) ↔ page fault oracle

Building the `strlen()` side-channel oracle with execution timing?



Building the `strlen()` side-channel oracle with execution timing?

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



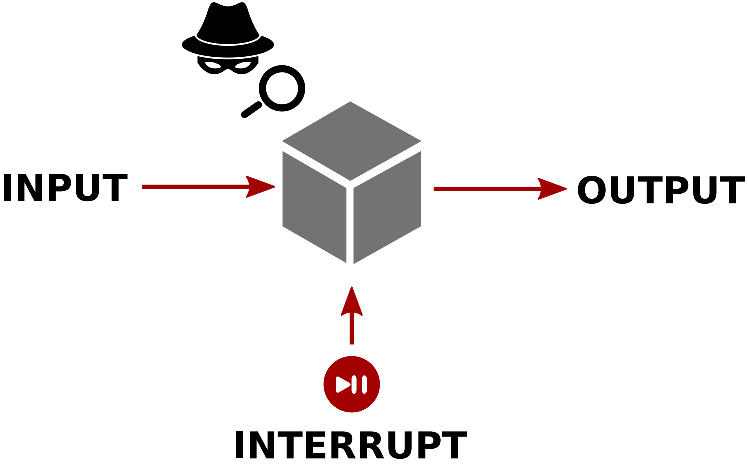
Challenge: Side-channel Sampling Rate

Slow
shutter speed

Medium
shutter speed

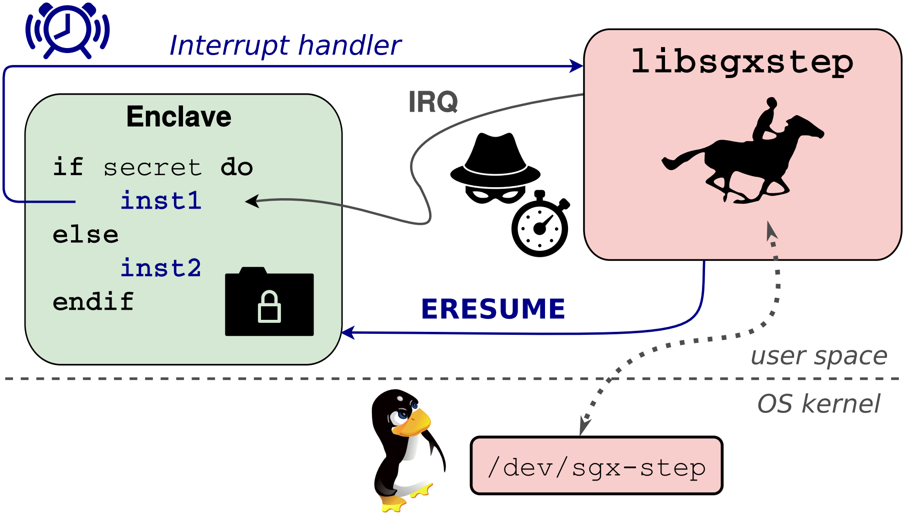
Fast
shutter speed

SGX-Step: Executing Enclaves one Instruction at a Time



🗋 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



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

SGX-Step demo: Building a `memcmp()` Password Oracle

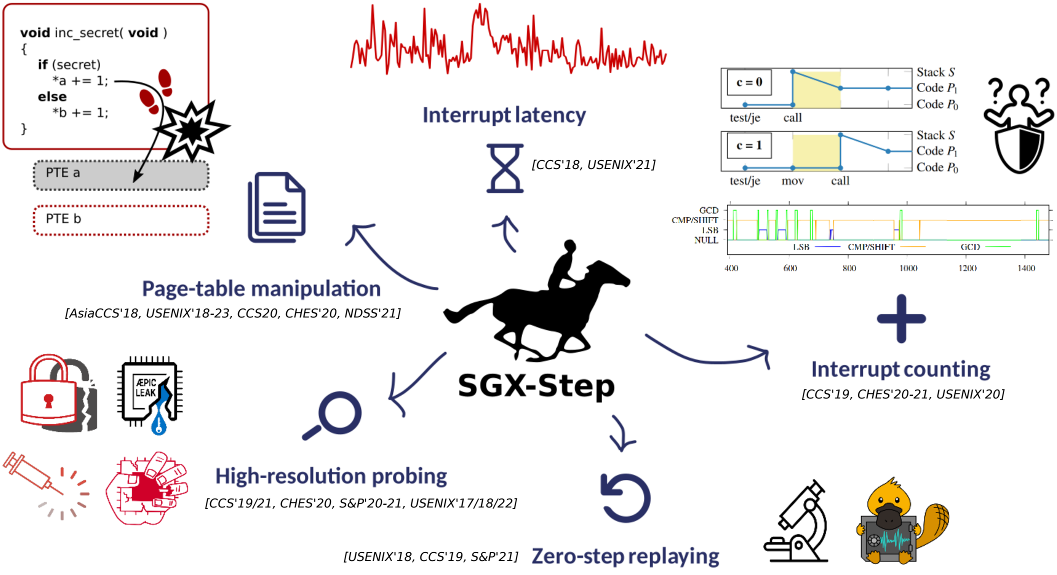
```
[idt.c] DTR.base=0xfffffe0000000000/size=4095 (256 entries)
[idt.c] established user space IDT mapping at 0x7f7ff8e9a000
[idt.c] installed asm IRQ handler at 10:0x56312d19b000
[idt.c] IDT[ 45] @0x7f7ff8e9a2d0 = 0x56312d19b000 (seg sel 0x10); p=1; dpl=3; type=14; ist=0
[file.c] reading buffer from '/dev/cpu/1/msr' (size=8)
[apic.c] established local memory mapping for APIC_BASE=0xfee00000 at 0x7f7ff8e99000
[apic.c] APIC_ID=2000000; LVTT=400ec; TDCR=0
[apic.c] APIC timer one-shot mode with division 2 (lvtt=2d/tdcr=0)

--------------------------------------------------------------------------------
[main.c] recovering password length
--------------------------------------------------------------------------------

[attacker] steps=15; guess='******'
[attacker] found pwd len = 6

--------------------------------------------------------------------------------
[main.c] recovering password bytes
--------------------------------------------------------------------------------

[attacker] steps=35; guess='SECRET' --> SUCCESS

[apic.c] Restored APIC_LVTT=400ec/TDCR=0)
[file.c] writing buffer to '/dev/cpu/1/msr' (size=8)
[main.c] all done; counted 2260/2183 IRQs (AEP/IDT)
jo@breuer:~/sgx-step-demo$
```

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

Yr	Venue	Paper	Step	Use Case	Drv
'15	S&P	Ctrl channel [XCP15]	~ Page	*Probe* (page fault)	✓
'16	ESORICS	AsyncShock [WKPK16]	~ Page	*Exploit* (mem safety)	–
'17	CHES	CacheZoom [MIE17]	✗ >1	*Probe* (L1 cache)	✓
'17	ATC	Hahnel et al. [HCP17]	✗ 0 - >1	*Probe* (L1 cache)	✓
'17	USENIX	BranchShadow [LSG+17]	✗ 5 - 50	*Probe* (BPU)	✗
'17	USENIX	Stealthy PTE [VBWK+17]	~ Page	*Probe* (page table)	✓
'17	USENIX	DarkROP [LJJ+17]	~ Page	*Exploit* (mem safety)	✓
'17	SysTEX	SGX-Step [VBPS17]	✓ 0 - 1	*Framework*	✓
'18	ESSoS	Off-limits [GVBPS18]	✓ 0 - 1	*Probe* (segmentation)	✓
'18	AsiaCCS	Single-trace RSA [WSB18]	~ Page	*Probe* (page fault)	✓
'18	USENIX	Foreshadow [VBMW+18]	✓ 0 - 1	*Probe* (transient exec)	✓
'18	EuroS&P	SgxPectre [CCX+19]	~ Page	*Exploit* (transient)	✓
'18	CHES	CacheQuote [DDME+18]	✗ >1	*Probe* (L1 cache)	✓
'18	ICCD	SGXlinger [HZDL18]	✗ >1	*Probe* (IRQ latency)	✗
'18	CCS	Nemesis [VBPS18]	✓ 1	*Probe* (IRQ latency)	✓
'19	USENIX	Spoiler [IMB+19]	✓ 1	*Probe* (IRQ latency)	✓
'19	CCS	ZombieLoad [SLM+19]	✓ 0 - 1	*Probe* (transient exec)	✓
'19	CCS	Fallout [CGG+19]	–	*Probe* (transient exec)	✓
'19	CCS	Tale of 2 worlds [VBOM+19]	✓ 1	*Exploit* (mem safety)	✓
'19	ISCA	MicroScope [SYG+19]	~ 0 - Page	*Framework*	✗
'20	CHES	Bluethunder [HMW+20]	✓ 1	*Probe* (BPU)	✓
'20	USENIX	Big troubles [WSBS19]	~ Page	*Probe* (page fault)	✓
'20	S&P	Plundervolt [MOG+20]	–	*Exploit* (undervolt)	✓
'20	CHES	Viral primitive [AB20]	✓ 1	*Probe* (IRQ count)	✓
'20	USENIX	CopyCat [MVBH+20]	✓ 1	*Probe* (IRQ count)	✓
'20	S&P	LVI [VBMS+20]	✓ 1	*Exploit* (transient)	✓

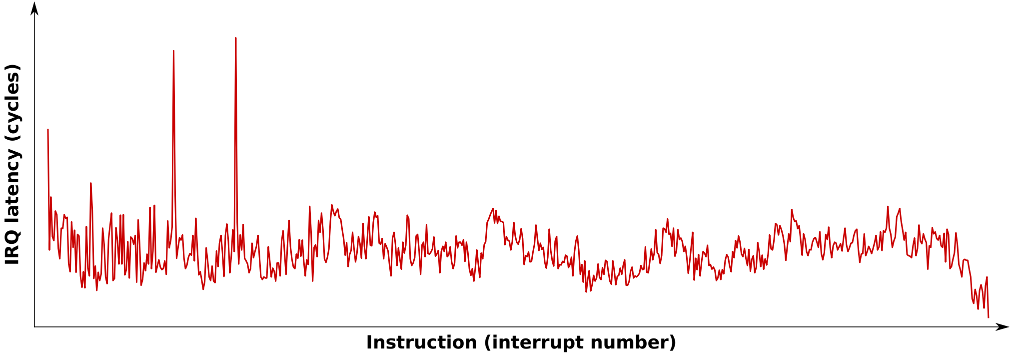
Yr	Venue	Paper	Step	Use Case	Drv
'20	CHES	A to Z [AGB20]	~ Page	*Probe* (page fault)	✓
'20	CCS	Déjà Vu NSS [uHGDL+20]	~ Page	*Probe* (page fault)	✓
'20	MICRO	PTHammer [ZCL+20]	–	*Probe* (page walk)	✓
'21	USENIX	Frontal [PSHC21]	✓ 1	*Probe* (IRQ latency)	✓
'21	S&P	CrossTalk [RMR+21]	✓ 1	*Probe* (transient exec)	✓
'21	CHES	Online template [AB21]	✓ 1	*Probe* (IRQ count)	✓
'21	NDSS	SpeechMiner [XZT20]	–	*Framework*	✓
'21	S&P	Platypus [LKO+21]	✓ 0 - 1	*Probe* (voltage)	✓
'21	DIMVA	Aion [HXCL21]	✓ 1	*Probe* (cache)	✓
'21	CCS	SmashEx [CYS+21]	✓ 1	*Exploit* (mem safety)	✓
'21	CCS	Util::Lookup [SBWE21]	✓ 1	*Probe* (L3 cache)	✓
'22	USENIX	Rapid prototyping [ESSG22]	✓ 1	*Framework*	✓
'22	CT-RSA	Kalyna expansion [CGYZ22]	✓ 1	*Probe* (L3 cache)	✓
'22	SEED	Enclyzer [ZXTZ22]	–	*Framework*	✓
'22	NordSec	Self-monitoring [LBA22]	~ Page	*Defense* (detect)	✓
'22	AutoSec	Robotic vehicles [LS22]	✓ 1 - >1	*Exploit* (timestamp)	✓
'22	ACSAC	MoLE [LWM+22]	✓ 1	*Defense* (randomize)	✓
'22	USENIX	AEPIC [BKS+22]	✓ 1	*Probe* (I/O device)	✓
'22	arXiv	Confidential code [PSL+22]	✓ 1	*Probe* (IRQ latency)	✓
'23	ComSec	FaultMorse [HZL+23]	~ Page	*Probe* (page fault)	✓
'23	CHES	HQC timing [HSC+23]	✓ 1	*Probe* (L3 cache)	✓
'23	ISCA	Belong to us [YJF23]	✓ 1	*Probe* (BPU)	✓
'23	USENIX	BunnyHop [ZTO+23]	✓ 1	*Probe* (BPU)	✓
'23	USENIX	DownFall [Mog23]	✓ 0 - 1	*Probe* (transient exec)	✓
'23	USENIX	AEX-Notify [CVBC+23]	✓ 1	*Defense* (prefetch)	✓

SGX-Step: A Versatile Open-Source Attack Toolkit



Nemesis: Extracting Interrupt Latency Traces with SGX-Step

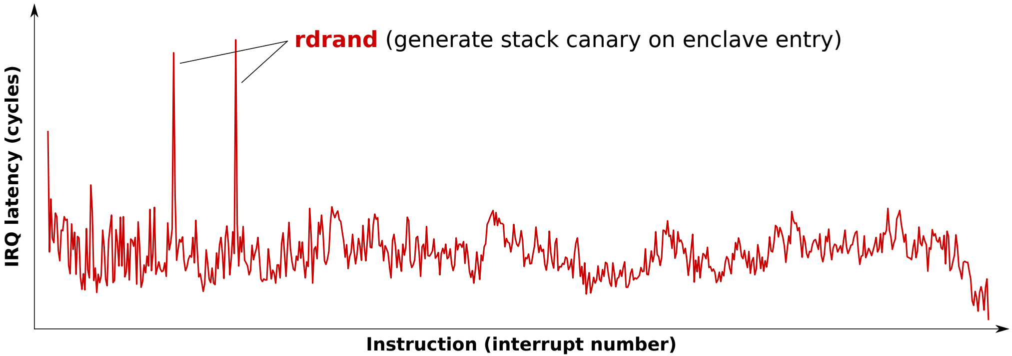
Enclave x-ray: IRQ latency leaks instruction-level μ-arch timing!



Van Bulck et al. "Nemesis: Studying Microarchitectural Timing Leaks in Rudimentary CPU Interrupt Logic", CCS 2018..

Nemesis: Extracting Interrupt Latency Traces with SGX-Step

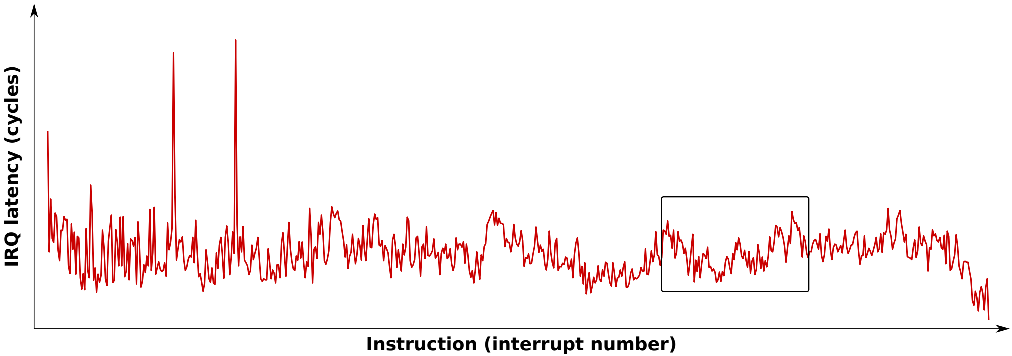
Enclave x-ray: Spotting high-latency instructions



Van Bulck et al. "Nemesis: Studying Microarchitectural Timing Leaks in Rudimentary CPU Interrupt Logic", CCS 2018..

Nemesis: Extracting Interrupt Latency Traces with SGX-Step

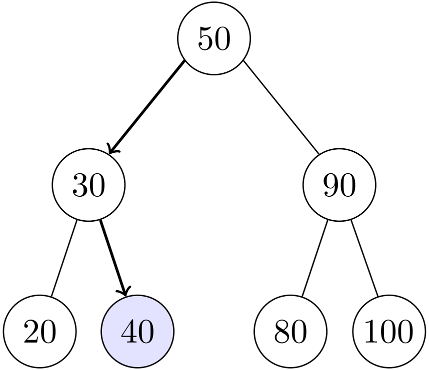
Enclave x-ray: Zooming in on `bsearch` function



Van Bulck et al. "Nemesis: Studying Microarchitectural Timing Leaks in Rudimentary CPU Interrupt Logic", CCS 2018..

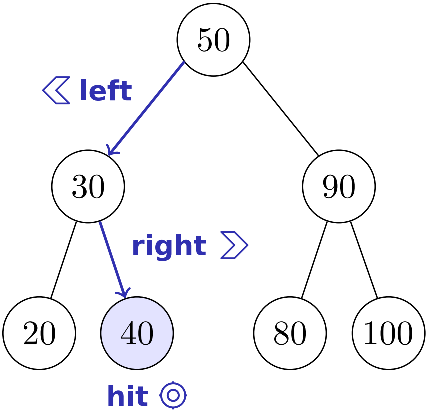
De-Anonymizing Enclave Lookups with Interrupt Latency

Binary search: Find 40 in $\{20, 30, 40, 50, 80, 90, 100\}$



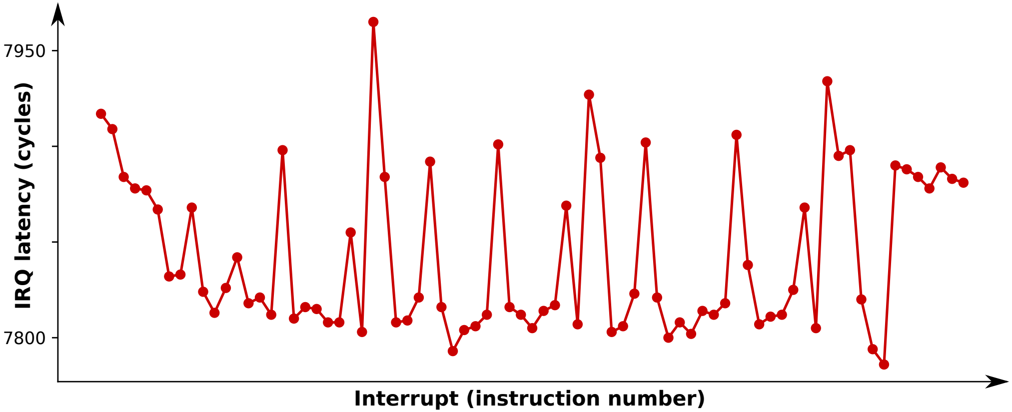
De-Anonymizing Enclave Lookups with Interrupt Latency

Adversary: Infer secret lookup in known array



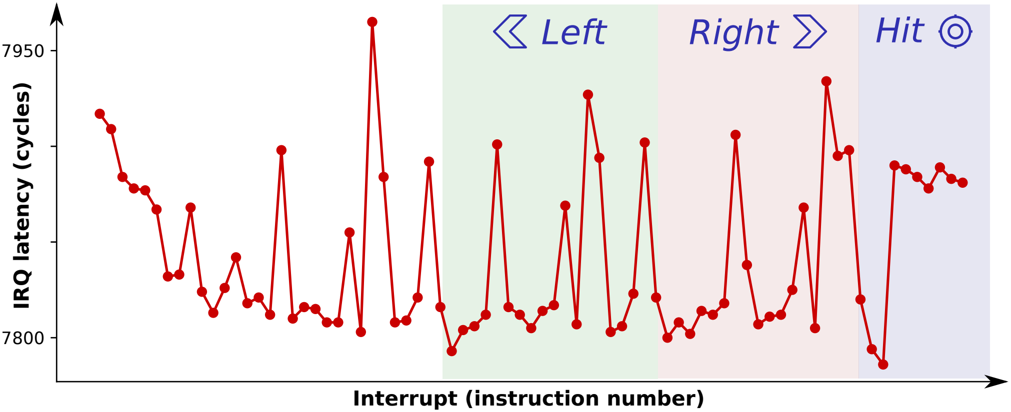
De-Anonymizing Enclave Lookups with Interrupt Latency

Goal: Infer lookup → reconstruct `bsearch` control flow



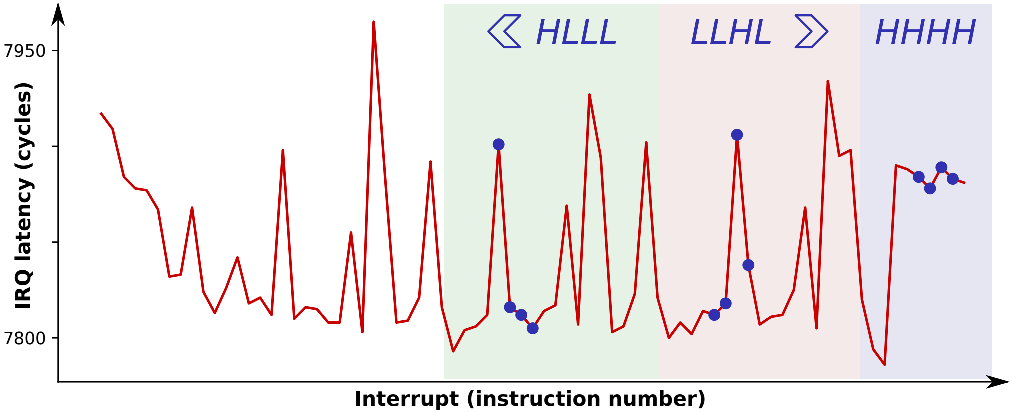
De-Anonymizing Enclave Lookups with Interrupt Latency

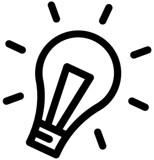
Goal: Infer lookup → reconstruct `bsearch` control flow



De-Anonymizing Enclave Lookups with Interrupt Latency

⇒ Sample **instruction latencies** in secret-dependent path





Idea #3: Interrupt Hardening

Hardening Enclaves against Interrupt-Driven Attacks

SGX-Step sets the **bar for adequate side-channel defenses!**

SGX-Step inspired several dedicated **hardware-software mitigations**

→ Several research prototypes on in-house secure Sancus processor

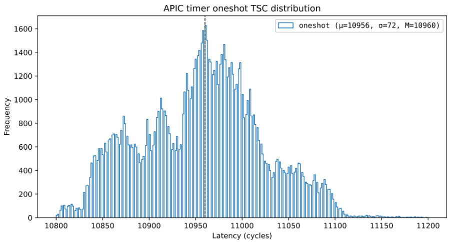
→ Collaboration with Intel on AEX-Notify: Included in recent processors

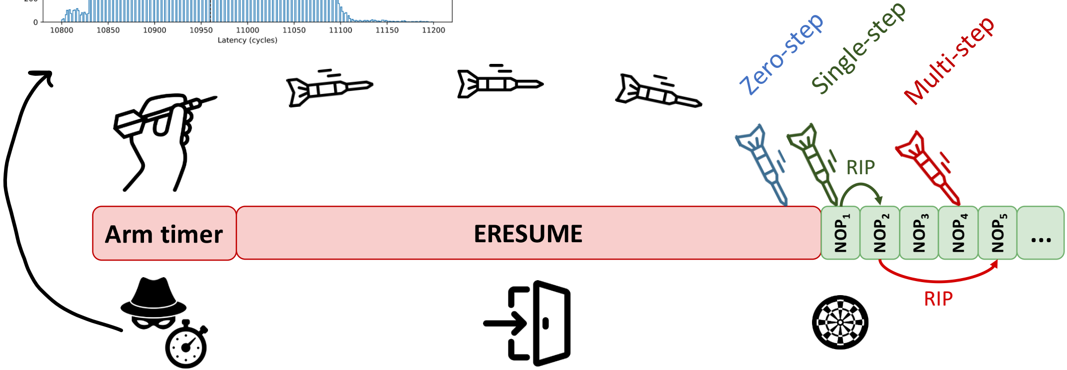
Busi et al., "Provably Secure Isolation for Interruptible Enclaved Execution on Small Microprocessors", CSF 2020..

Bognar et al., "MicroProfiler: Principled Side-Channel Mitigation through Microarchitectural Profiling", EuroS&P 2023..

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

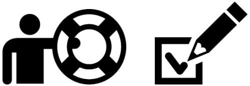
Root-causing SGX-Step: Aiming the timer interrupt

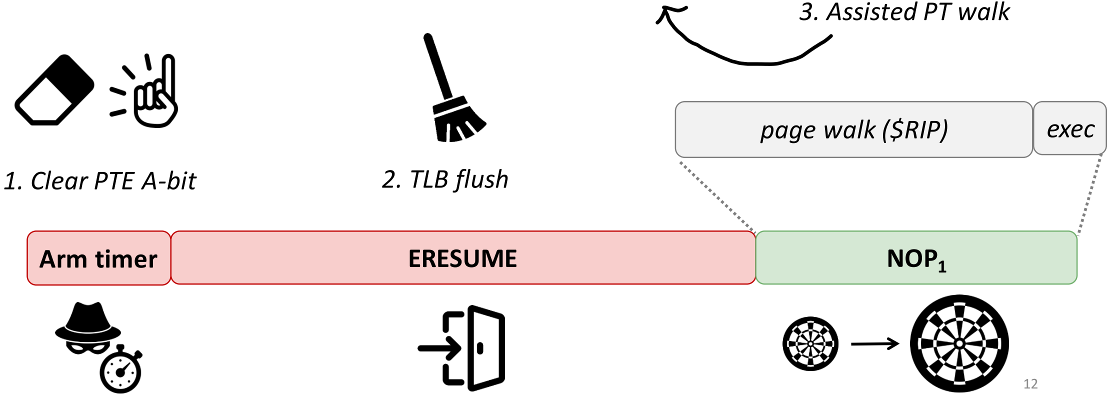




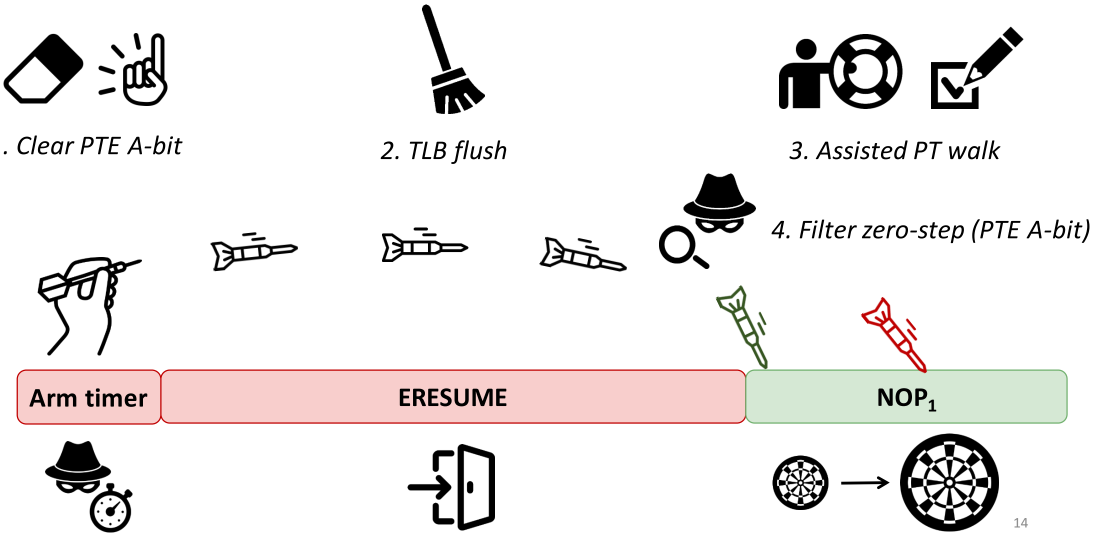
Root-causing SGX-Step: Microcode assists to the rescue!

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

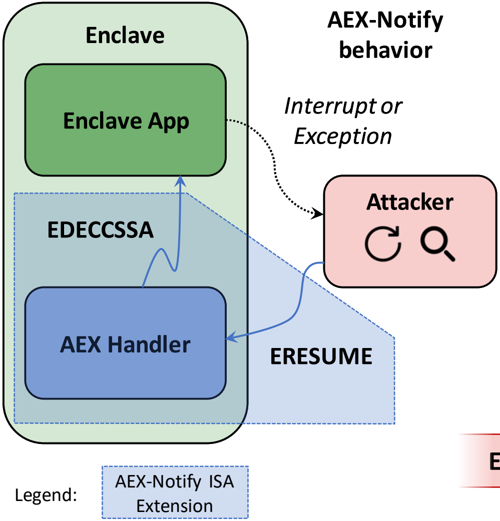


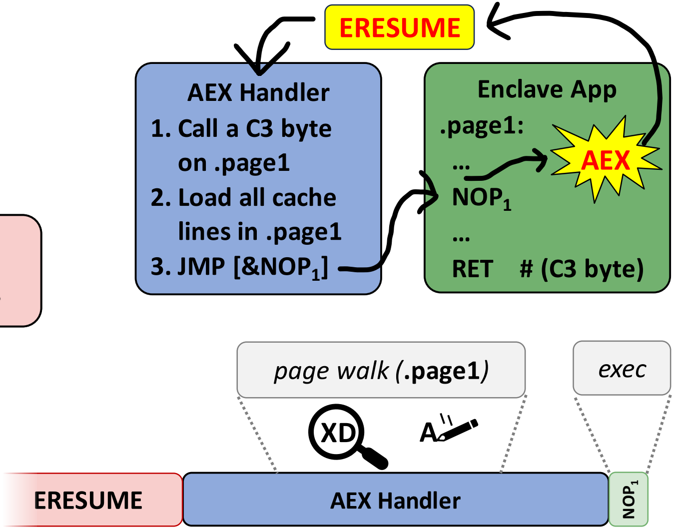


Root-causing SGX-Step: Microcode assists to the rescue!



AEX-Notify solution overview





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!**

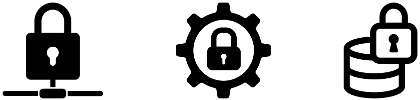
→ shipped in millions of devices ≥ 4th Gen Xeon CPU

Conclusions and Takeaway

⇒ **Trusted execution** environments (Intel SGX) $\neq$ perfect!

⇒ Subtle **side channels** can go a long way…

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



Conclusions and Takeaway

⟹ **Trusted execution** environments (Intel SGX) ≠ perfect!

⟹ Subtle **side channels** can go a long way...

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

Thank you! Questions?

Appendix

References i

A. C. Aldaya and B. B. Brumley.
When one vulnerable primitive turns viral: Novel single-trace attacks on ECDSA and RSA.
IACR Transactions on Cryptographic Hardware and Embedded Systems, pp. 196–221, 2020.

A. C. Aldaya and B. B. Brumley.
Online template attacks: Revisited.
CHES, pp. 28–59, 2021.

A. C. Aldaya, C. P. García, and B. B. Brumley.
From A to Z: Projective coordinates leakage in the wild.
IACR Transactions on Cryptographic Hardware and Embedded Systems, 2020.

P. Borrello, A. Kogler, M. Schwarzl, M. Lipp, D. Gruss, and M. Schwarz.
ÆPIC Leak: Architecturally leaking uninitialized data from the microarchitecture.
In *USENIX Security*, 2022.

G. Chen, S. Chen, Y. Xiao, Y. Zhang, Z. Lin, and T. H. Lai.
SgxPectre attacks: Stealing Intel secrets from SGX enclaves via speculative execution.
In *4th IEEE European Symposium on Security and Privacy (Euro S&P)*, 2019.

C. Canella, D. Genkin, L. Giner, D. Gruss, M. Lipp, M. Minkin, D. Moghimi, F. Piessens, M. Schwarz, B. Sunar, J. Van Bulck, and Y. Yarom.
Fallout: Leaking data on Meltdown-resistant CPUs.
In *26th ACM Conference on Computer and Communications Security (CCS)*, pp. 769–784, November 2019.

C. Chuengsatiansup, D. Genkin, Y. Yarom, and Z. Zhang.
Side-channeling the kalyna key expansion.
In *CT-RSA*, 2022.

S. Constable, J. Van Bulck, X. Cheng, Y. Xiao, C. Xing, I. Alexandrovich, T. Kim, F. Piessens, M. Vij, and M. Silberstein.
Aex-notify: Thwarting precise single-stepping attacks through interrupt awareness for intel sgx enclaves.
In *32nd USENIX Security Symposium*, pp. 4051–4068, August 2023.

J. Cui, J. Z. Yu, S. Shinde, P. Saxena, and Z. Cai.
Smashex: Smashing SGX enclaves using exceptions.
In *CCS*, 2021.

References iii

F. Dall, G. De Micheli, T. Eisenbarth, D. Genkin, N. Heninger, A. Moghimi, and Y. Yarom.
CacheQuote: efficiently recovering long-term secrets of SGX EPID via cache attacks.
IACR Transactions on Cryptographic Hardware and Embedded Systems, (2):171–191, 2018.

C. Easdon, M. Schwarz, M. Schwarzl, and D. Gruss.
Rapid prototyping for microarchitectural attacks.
In *31st USENIX Security Symposium (USENIX Security 22)*, pp. 3861–3877, 2022.

J. Gyselinck, J. Van Bulck, F. Piessens, and R. Strackx.
Off-limits: Abusing legacy x86 memory segmentation to spy on enclaved execution.
In *International Symposium on Engineering Secure Software and Systems (ESSoS)*, pp. 44–60, June 2018.

M. Hähnel, W. Cui, and M. Peinado.
High-resolution side channels for untrusted operating systems.
In *USENIX Annual Technical Conference (ATC)*, 2017.

T. Huo, X. Meng, W. Wang, C. Hao, P. Zhao, J. Zhai, and M. Li.
Bluethunder: A 2-level directional predictor based side-channel attack against SGX.
IACR Transactions on Cryptographic Hardware and Embedded Systems, pp. 321–347, 2020.

References iv

S. Huang, R. Q. Sim, C. Chuengsatiansup, Q. Guo, and T. Johansson.
Cache-timing attack against HQC.
IACR ePrint Archive, 2023.

W. Huang, S. Xu, Y. Cheng, and D. Lie.
Aion attacks: Manipulating software timers in trusted execution environment.
In *DIMVA*, 2021.

W. He, W. Zhang, S. Das, and Y. Liu.
Sgxlinger: A new side-channel attack vector based on interrupt latency against enclave execution.
In *36th IEEE International Conference on Computer Design (ICCD)*, pp. 108–114, 2018.

L. Hu, F. Zhang, Z. Liang, R. Ding, X. Cai, Z. Wang, and W. Jin.
Faultmorse: An automated controlled-channel attack via longest recurring sequence.
Computers & Security, 124:103003, 2023.

S. Islam, A. Moghimi, I. Bruhns, M. Krebbel, B. Gulmezoglu, T. Eisenbarth, and B. Sunar.
SPOILER: Speculative load hazards boost rowhammer and cache attacks.
In *28th USENIX Security Symposium*, 2019.

References v

D. Lantz, F. Boeira, and M. Asplund.
Towards self-monitoring enclaves: Side-channel detection using performance counters.
In *Nordic Conference on Secure IT Systems*, pp. 120–138. Springer, 2022.

J. Lee, J. Jang, Y. Jang, N. Kwak, Y. Choi, C. Choi, T. Kim, M. Peinado, and B. B. Kang.
Hacking in Darkness: Return-Oriented Programming Against Secure Enclaves.
In *26th USENIX Security Symposium*, pp. 523–539, 2017.

M. Lipp, A. Kogler, D. Oswald, M. Schwarz, C. Easdon, C. Canella, and D. Gruss.
Platypus: Software-based power side-channel attacks on x86.
In *S&P*, pp. 355–371, 2021.

M. Luo and G. E. Suh.
Wip: Interrupt attack on tee-protected robotic vehicles.
In *Workshop on Automotive and Autonomous Vehicle Security (AutoSec)*, April 2022.

S. Lee, M.-W. Shih, P. Gera, T. Kim, H. Kim, and M. Peinado.
Inferring fine-grained control flow inside SGX enclaves with branch shadowing.
In *26th USENIX Security Symposium*, pp. 557–574, 2017.

F. Lang, W. Wang, L. Meng, J. Lin, Q. Wang, and L. Lu.
Mole: Mitigation of side-channel attacks against sgx via dynamic data location escape.
In *Proceedings of the 38th Annual Computer Security Applications Conference*, pp. 978–988, 2022.

A. Moghimi, G. Irazoqui, and T. Eisenbarth.
Cachezoom: How SGX amplifies the power of cache attacks.
In *19th International Conference on Cryptographic Hardware and Embedded Systems (CHES)*, 2017.

K. Murdock, D. Oswald, F. D. Garcia, J. Van Bulck, D. Gruss, and F. Piessens.
Plundervolt: Software-based fault injection attacks against Intel SGX.
In *41st IEEE Symposium on Security and Privacy (S&P)*, pp. 1466–1482, May 2020.

D. Moghimi.
Downfall: Exploiting speculative data gathering.
In *32nd USENIX Security Symposium (USENIX Security 23)*, 2023.

D. Moghimi, J. Van Bulck, N. Heninger, F. Piessens, and B. Sunar.
CopyCat: Controlled instruction-level attacks on enclaves.
In *29th USENIX Security Symposium*, pp. 469–486, August 2020.

I. Puddu, M. Schneider, M. Haller, and S. Capkun.
Frontal attack: Leaking control-flow in SGX via the CPU frontend.
In *USENIX Security*, pp. 663–680, 2021.

I. Puddu, M. Schneider, D. Lain, S. Boschetto, and S. Čapkun.
On (the lack of) code confidentiality in trusted execution environments.
arXiv, 2022.

H. Ragab, A. Milburn, K. Razavi, H. Bos, and C. Giuffrida.
CrossTalk: Speculative data leaks across cores are real.
In *42nd IEEE Symposium on Security and Privacy (S&P)*, May 2021.

F. Sieck, S. Berndt, J. Wichelmann, and T. Eisenbarth.
Util::lookup: Exploiting key decoding in cryptographic libraries.
In *CCS*, p. 2456–2473, 2021.

M. Schwarz, M. Lipp, D. Moghimi, J. Van Bulck, J. Stecklina, T. Prescher, and D. Gruss.
ZombieLoad: Cross-privilege-boundary data sampling.
In *26th ACM Conference on Computer and Communications Security (CCS)*, pp. 753–768, November 2019.

D. Skarlatos, M. Yan, B. Gopireddy, R. Sprabery, J. Torrellas, and C. W. Fletcher.
Microscope: Enabling microarchitectural replay attacks.
In *46th International Symposium on Computer Architecture (ISCA)*, pp. 318–331, 2019.

S. ul Hassan, I. Gridin, I. M. Delgado-Lozano, C. P. García, J.-J. Chi-Domínguez, A. C. Aldaya, and B. B. Brumley.
Déjà vu: Side-channel analysis of mozilla's nss.
arXiv preprint arXiv:2008.06004, 2020.

J. Van Bulck, D. Moghimi, M. Schwarz, M. Lipp, M. Minkin, D. Genkin, Y. Yuval, B. Sunar, D. Gruss, and F. Piessens.
LVI: Hijacking transient execution through microarchitectural load value injection.
In *41st IEEE Symposium on Security and Privacy (S&P)*, pp. 54–72, May 2020.

J. Van Bulck, M. Minkin, O. Weisse, D. Genkin, B. Kasikci, F. Piessens, M. Silberstein, T. F. Wenisch, Y. Yarom, and R. Strackx.
Foreshadow: Extracting the keys to the Intel SGX kingdom with transient out-of-order execution.
In *27th USENIX Security Symposium*, pp. 991–1008, August 2018.

J. Van Bulck, D. Oswald, E. Marin, A. Aldoseri, F. D. Garcia, and F. Piessens.
A tale of two worlds: Assessing the vulnerability of enclave shielding runtimes.
In *26th ACM Conference on Computer and Communications Security (CCS)*, pp. 1741–1758, November 2019.

J. Van Bulck, F. Piessens, and R. Strackx.
SGX-Step: A practical attack framework for precise enclave execution control.
In *2nd Workshop on System Software for Trusted Execution (SysTEX)*, pp. 4:1–4:6. ACM, October 2017.

J. Van Bulck, F. Piessens, and R. Strackx.
Nemesis: Studying microarchitectural timing leaks in rudimentary CPU interrupt logic.
In *25th ACM Conference on Computer and Communications Security (CCS)*, pp. 178–195, October 2018.

J. Van Bulck, N. Weichbrodt, R. Kapitza, F. Piessens, and R. Strackx.
Telling your secrets without page faults: Stealthy page table-based attacks on enclaved execution.
In *26th USENIX Security Symposium*, pp. 1041–1056, August 2017.

N. Weichbrodt, A. Kurmus, P. Pietzuch, and R. Kapitza.
Asyncshock: Exploiting synchronisation bugs in Intel SGX enclaves.
In *European Symposium on Research in Computer Security (ESORICS)*, 2016.

References x

S. Weiser, R. Spreitzer, and L. Bodner.
Single trace attack against RSA key generation in Intel SGX SSL.
In *13th ACM Asia Conference on Computer and Communications Security (AsiaCCS)*, pp. 575–586, 2018.

S. Weiser, D. Schrammel, L. Bodner, and R. Spreitzer.
Big numbers–big troubles: Systematically analyzing nonce leakage in (ec) dsa implementations.
In *29th USENIX Security Symposium*, 2019.

Y. Xu, W. Cui, and M. Peinado.
Controlled-channel attacks: Deterministic side channels for untrusted operating systems.
In *36th IEEE Symposium on Security and Privacy (S&P)*, pp. 640–656, 2015.

Y. Xiao, Y. Zhang, and R. Teodorescu.
Speechminer: A framework for investigating and measuring speculative execution vulnerabilities.
In *Network and Distributed System Security Symposium (NDSS)*, 2020.

J. Yu, T. Jaeger, and C. W. Fletcher.
All your pc are belong to us: Exploiting non-control-transfer instruction btb updates for dynamic pc extraction.
In *Proceedings of the 50th Annual International Symposium on Computer Architecture*, pp. 1–14, 2023.

References xi

Z. Zhang, Y. Cheng, D. Liu, S. Nepal, Z. Wang, and Y. Yarom.
Pthammer: Cross-user-kernel-boundary rowhammer through implicit accesses.
arXiv preprint arXiv:2007.08707, 2020.

Z. Zhang, M. Tao, S. O'Connell, C. Chuengsatiansup, D. Genkin, and Y. Yarom.
BunnyHop: Exploiting the instruction prefetcher.
In *32nd USENIX Security Symposium (USENIX Security 23)*, pp. 7321–7337, 2023.

J. Zhou, Y. Xiao, R. Teodorescu, and Y. Zhang.
Enclyzer: Automated analysis of transient data leaks on intel sgx.
In *2022 IEEE International Symposium on Secure and Private Execution Environment Design (SEED)*, pp. 145–156. IEEE, 2022.