TLBlur: The Art of Obscuring SGX Page Accesses Across Space and Time

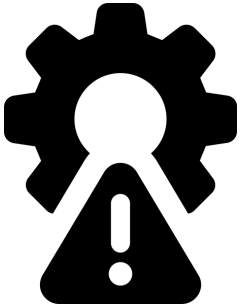
Daan Vanoverloop, Jo Van Bulck

joint work with Andrés Sánchez, Flavio Toffalini, Frank Piessens, Mathias Payer

Intel SPR Tech Talk, March 3, 2025

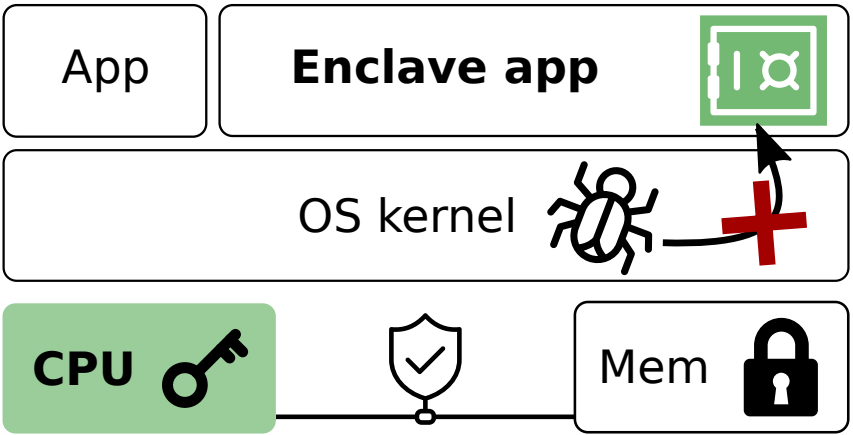
DistriNet, KU Leuven, Belgium

daan.vanoverloop@kuleuven.be jo.vanbulck@kuleuven.be



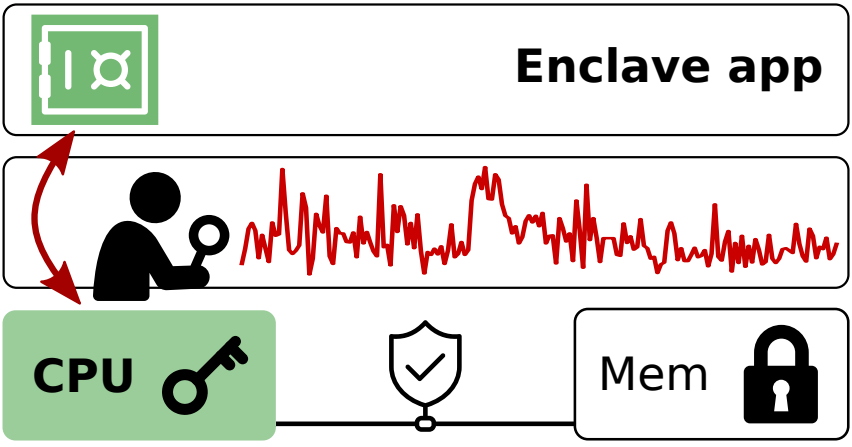
Part I: Problem statement

Enclaved Execution: Reducing Attack Surface



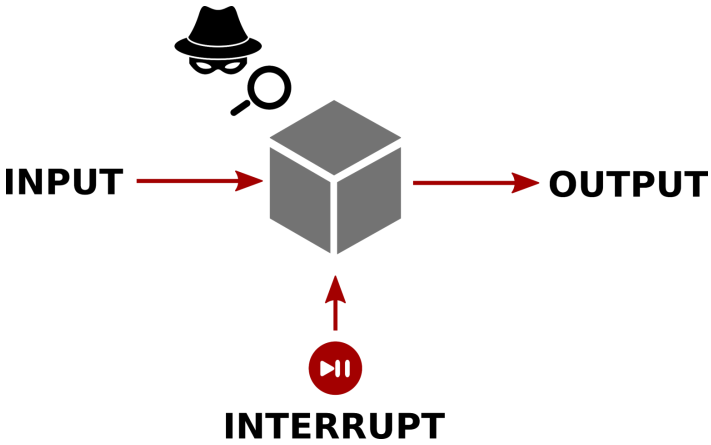
Intel SGX promise: Hardware-level **isolation and attestation**

Enclaved Execution: Privileged Side-Channel Attacks



Game changer: Untrusted OS → new class of powerful **side channels!**

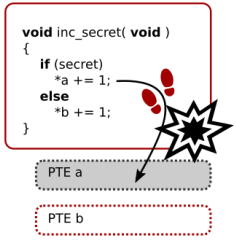
SGX-Step: Executing enclaves one instruction at a time

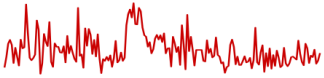


SGX-Step: Executing enclaves one instruction at a time

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

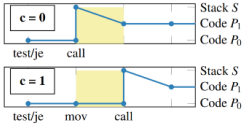
SGX-Step: A Versatile Open-Source Attack Toolkit

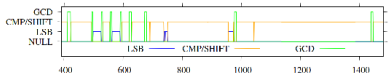




Interrupt latency

[CCS'18, USENIX'21]





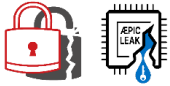
Page-table manipulation

[AsiaCCS'18, USENIX'18-23, CCS20, CHES'20, NDSS'21]

SGX-Step

Interrupt counting

[CCS'19, CHES'20-21, USENIX'20]

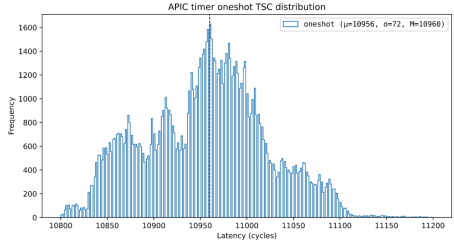


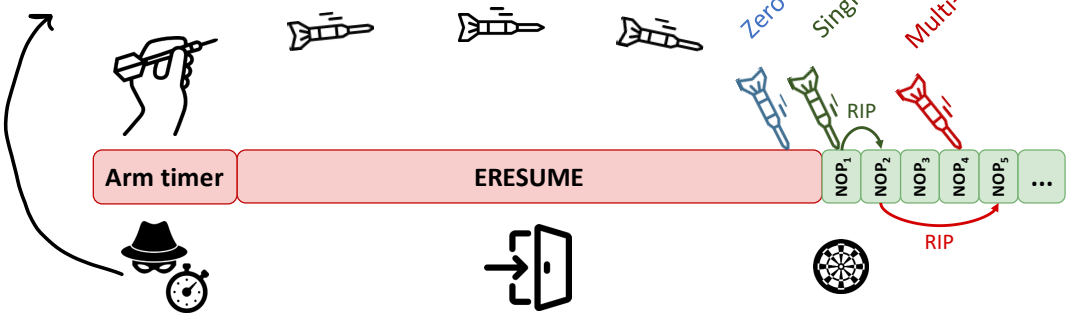
High-resolution probing

[CCS'19/21, CHES'20, S&P'20-21, USENIX'17/18/22]

[USENIX'18, CCS'19, S&P'21] Zero-step replaying

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

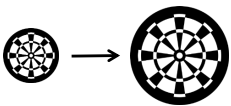
3. Assisted PT walk

1. Clear PTE A-bit

2. TLB flush

page walk ($RIP) | *exec*

Arm timer | **ERESUME** | **NOP$_1$**



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

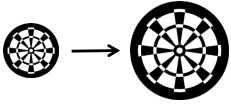
1. Clear PTE A-bit

2. TLB flush

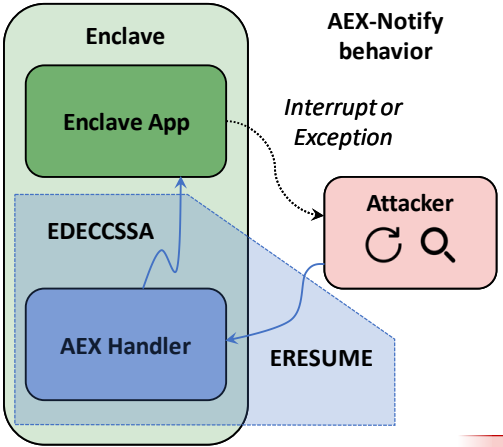
3. Assisted PT walk

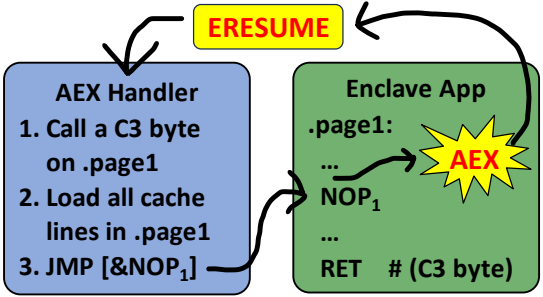
4. Filter zero-step (PTE A-bit)

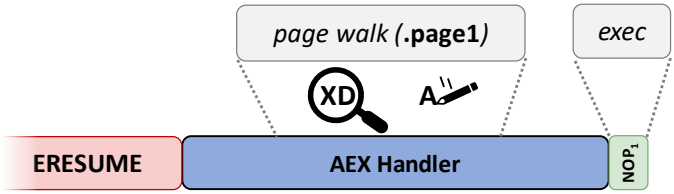
Arm timer	ERESUME	NOP_1



AEX-Notify solution overview

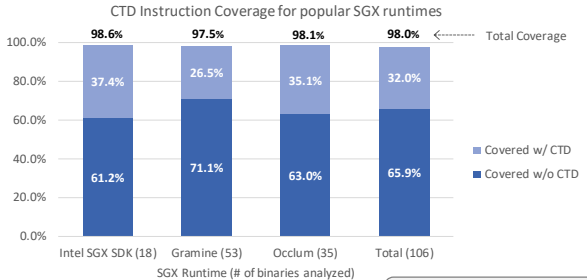


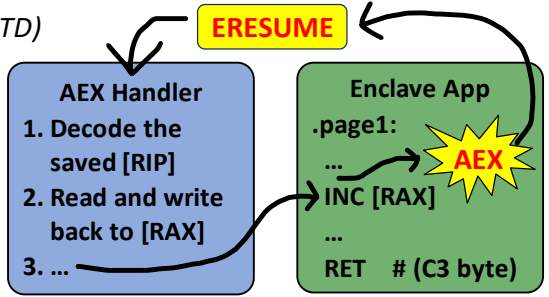


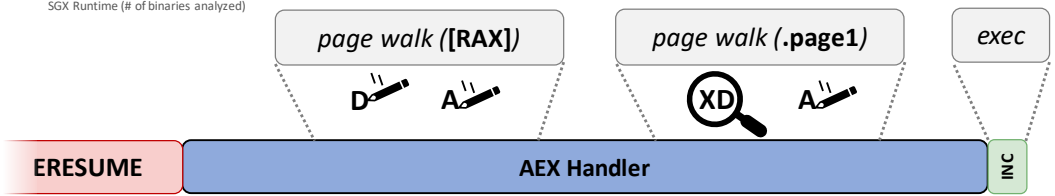


AEX-Notify solution overview

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





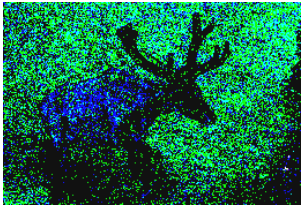


There's a Catch...

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 23.

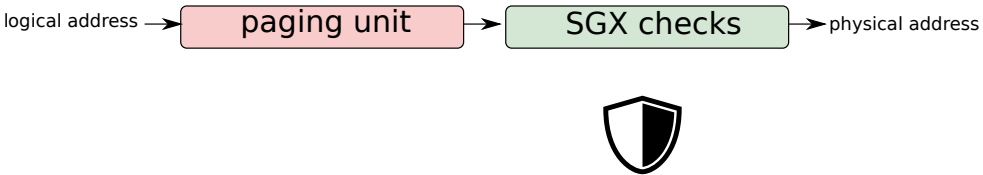
Why Mitigating Single-Stepping is Not Enough



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

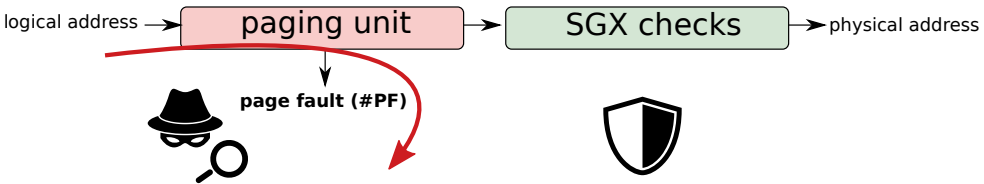
- Xu et al.: "Controlled-channel attacks: Deterministic side channels for untrusted operating systems", Oakland 2015.
- Constable et al.: "AEX-Notify: Thwarting Precise Single-Stepping Attacks through Interrupt Awareness for Intel SGX Enclaves", USENIX Security 2023.

Intel SGX: Page faults as a side channel



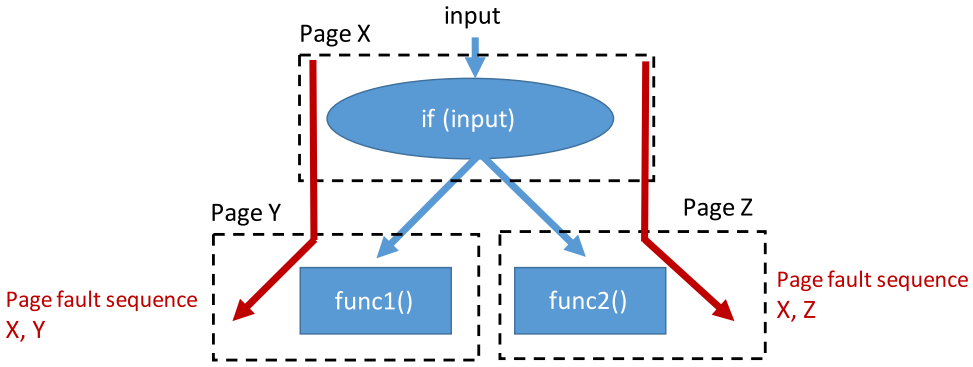
SGX machinery protects against direct address remapping attacks

Intel SGX: Page faults as a side channel



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

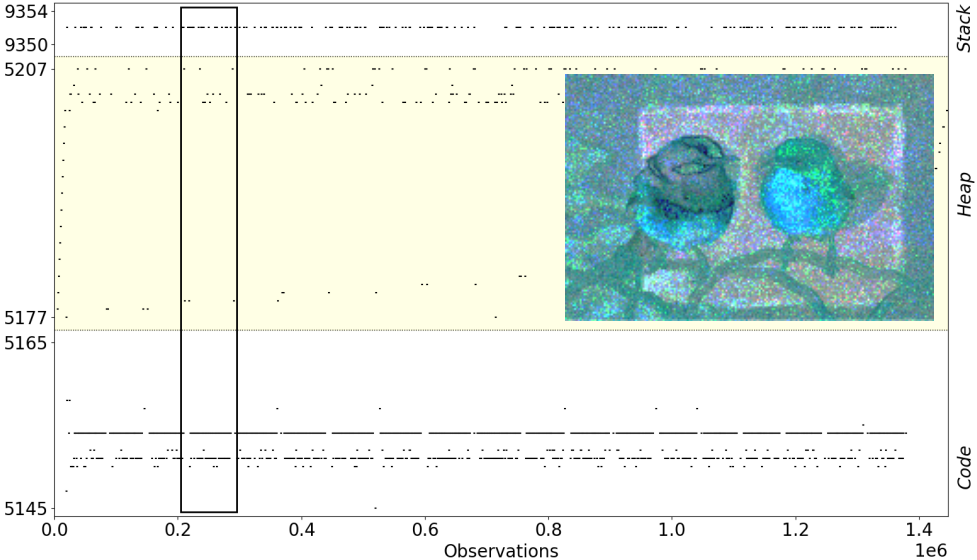
Intel SGX: Page faults as a side channel



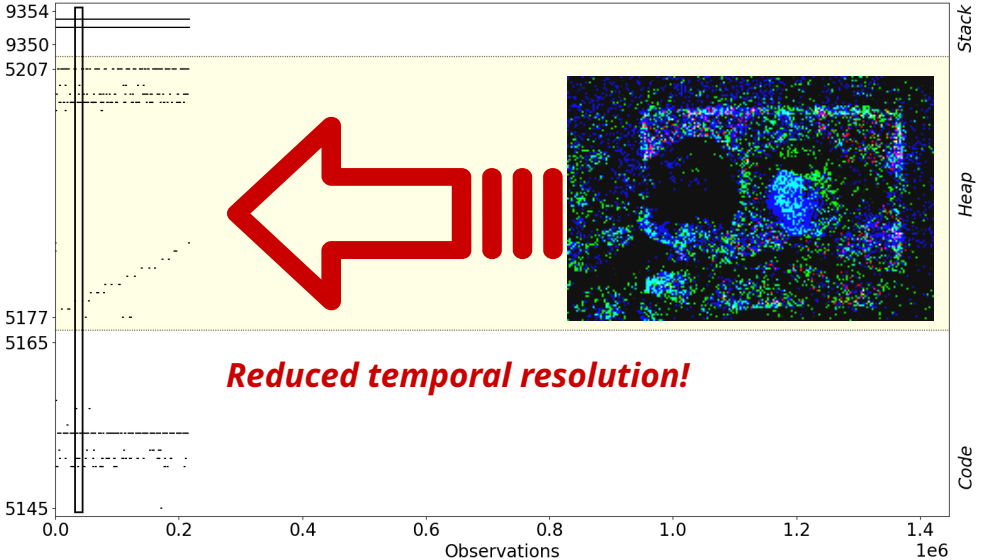
Xu et al.: "Controlled-channel attacks: Deterministic side channels for untrusted operating systems", Oakland 2015.

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

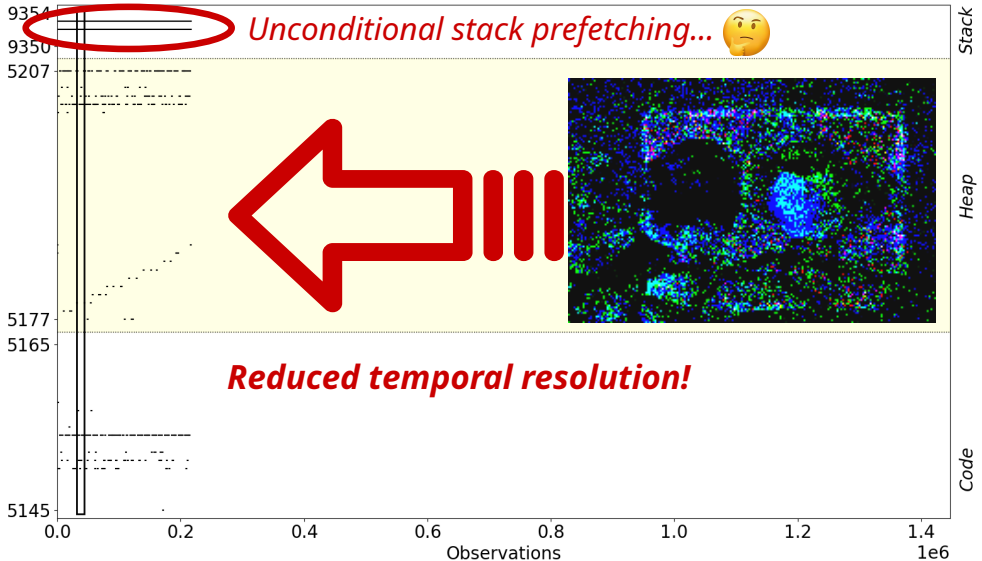
Libjpeg: AEX-Notify's Temporal Reduction in Practice

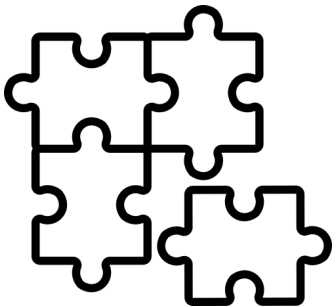


Libjpeg: AEX-Notify's Temporal Reduction in Practice



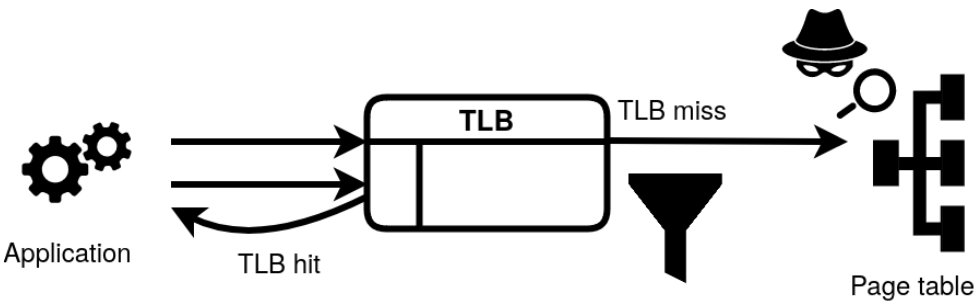
Libjpeg: AEX-Notify's Temporal Reduction in Practice



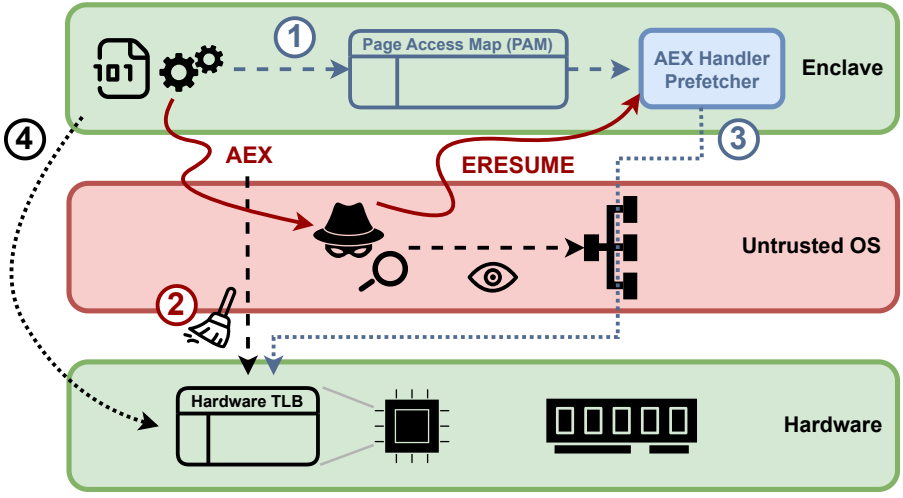


Part II: Solution overview

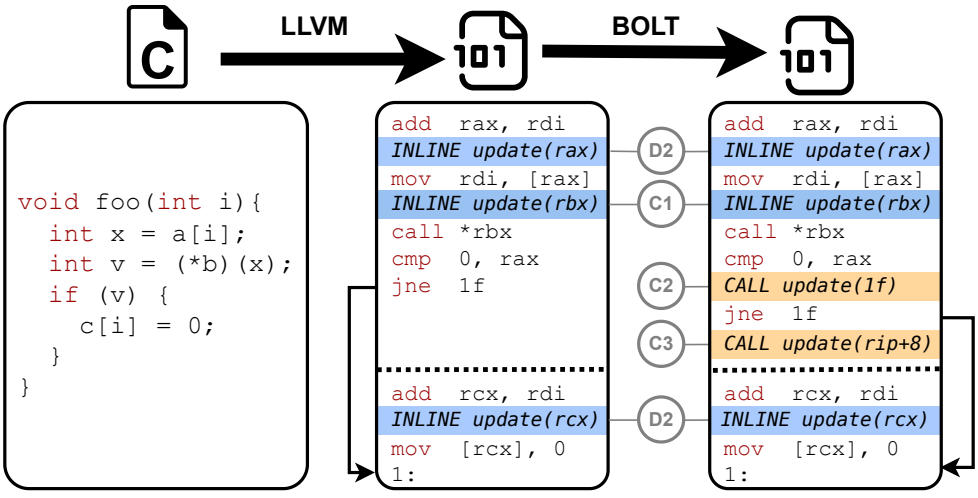
Idea: TLB as a “Filter” to Hide Page Accesses



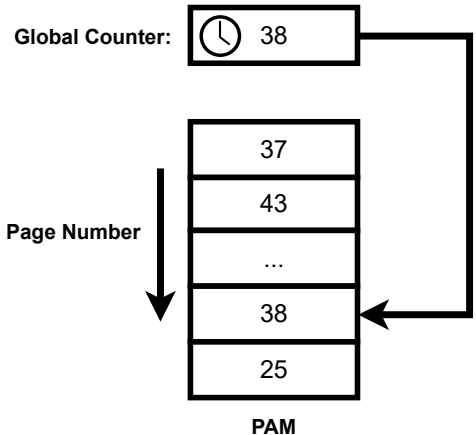
TLBlur: Self-Monitoring and Restoring Enclave Page Accesses



Instrumentation to Self-Monitor Page Accesses at Runtime

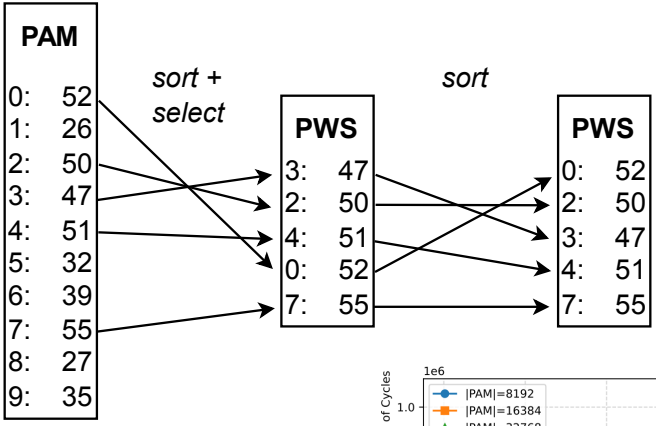


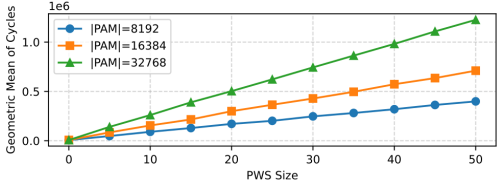
Efficiently Updating the Page-Access Map



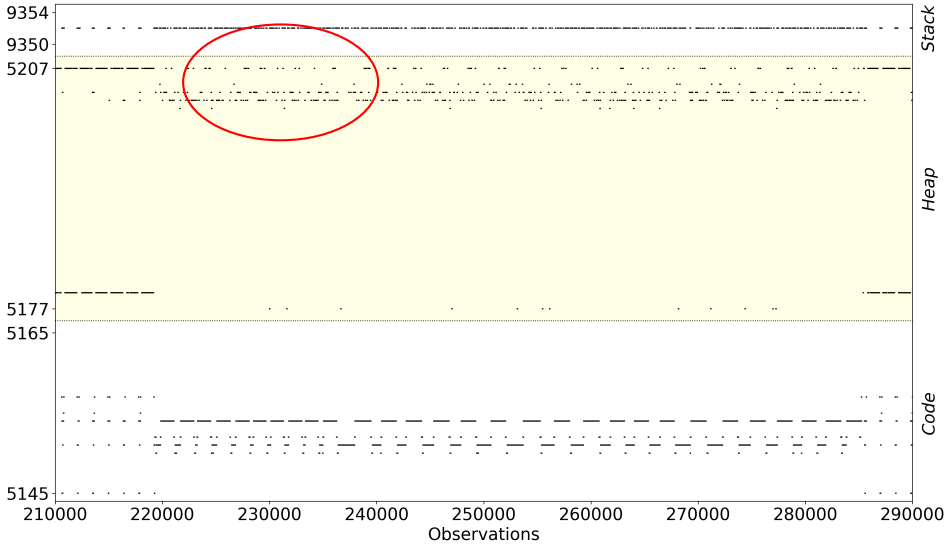

```
tlblur_pam_update:
    page_num = compute_index(addr)

    counter = global_counter
    counter++

    global_counter = counter
    pam[page_num] = counter
```

Constant-Time Page-Access Map Sorting

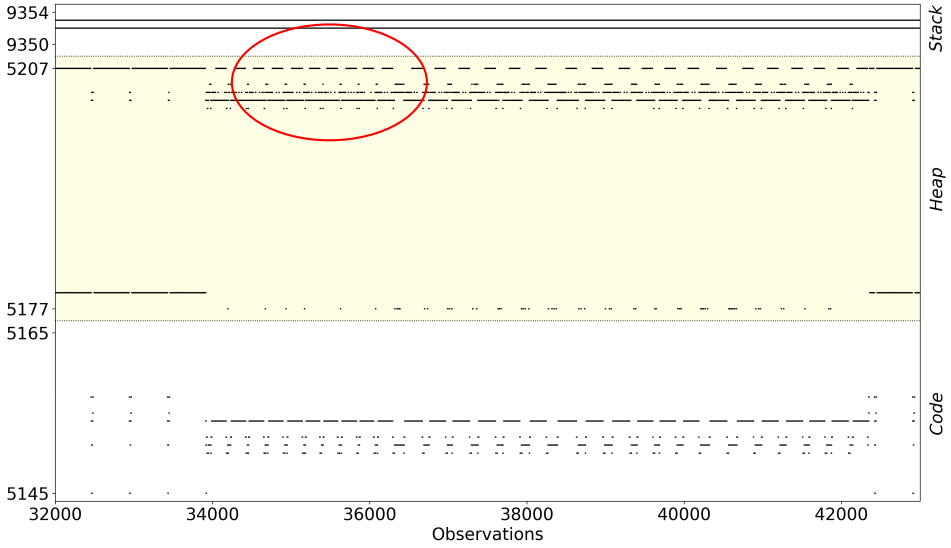




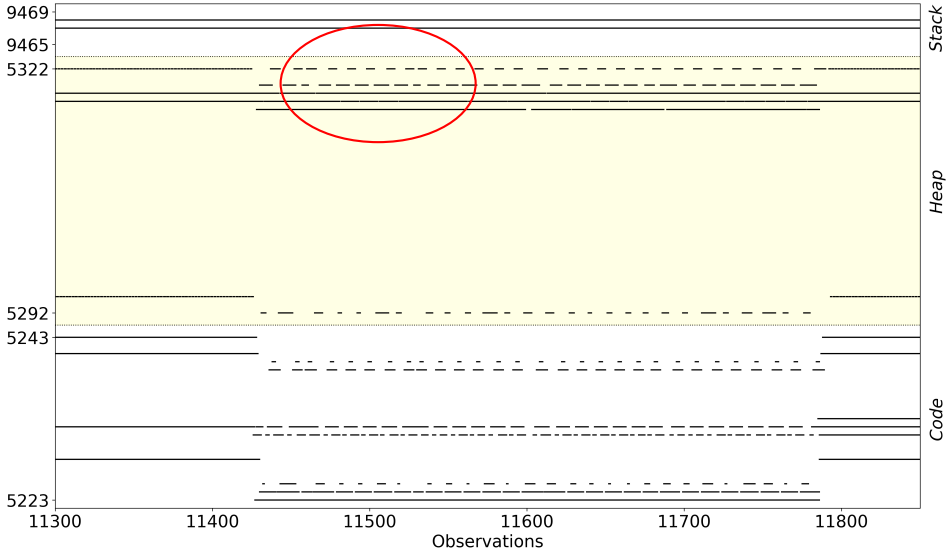
Leakage Reduction in Practice: libjpeg with Single Stepping



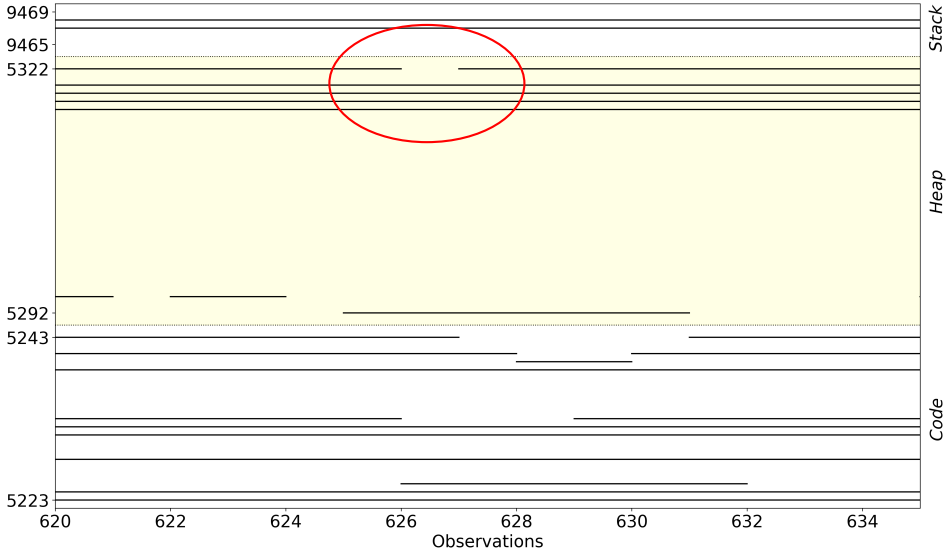
Leakage Reduction in Practice: libjpeg with Page Faults



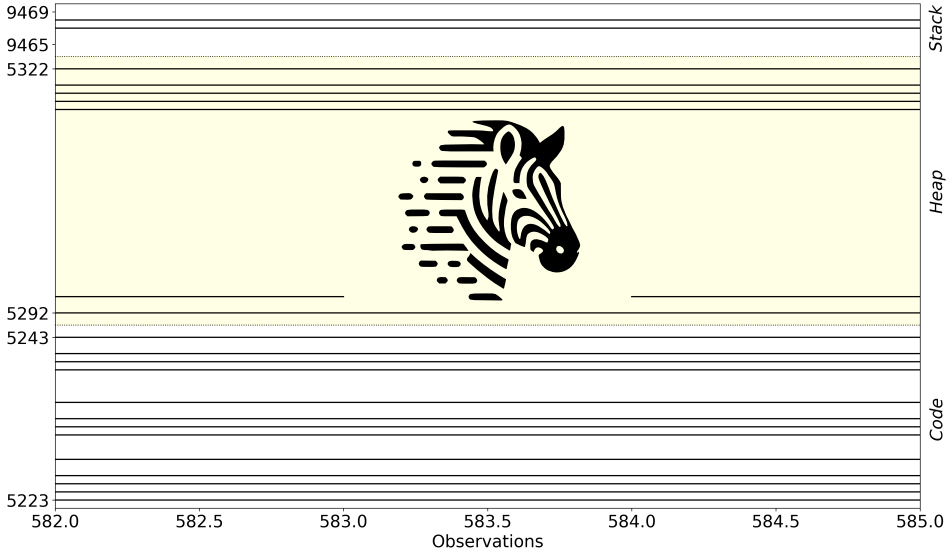
Leakage Reduction in Practice: libjpeg with TLBlur ($N = 10$)



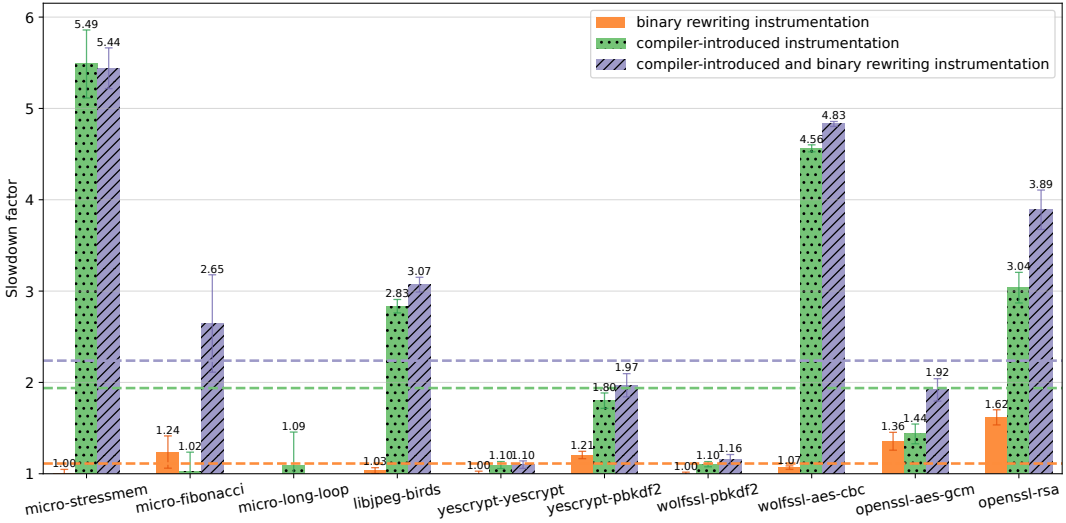
Leakage Reduction in Practice: libjpeg with TLBlur ($N = 20$)



Leakage Reduction in Practice: libjpeg with TLBlur ($N = 30$)



Performance Overhead



Ideas and Future Work

1. More **efficient PAM computation:**
   - Hardware-assisted TLB save on AEX?
   - Binary-only instrumentation with BOLT
   - Compile-time page clustering
2. Generalization to **other TEEs** → TDX-Notify? :-)

Conclusion and Outlook

- **TLBlur**: Compiler instrumentation + AEX-Notify prefetching
  → *Auto "blurring" of enclave page-access traces*
- **Challenge:** Towards principled hardware-software defenses for page-table-based attacks?

`https://vanbulck.net/files/usenix25-tlblur.pdf`
`https://github.com/tlblur-sgx`

Thank you! Questions?