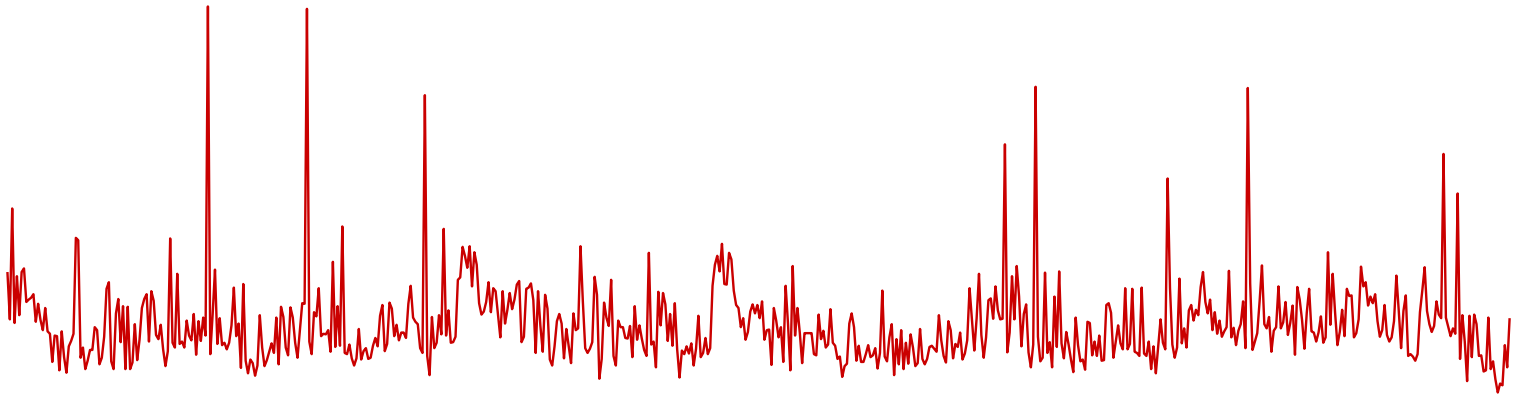
Nemesis: Studying Microarchitectural Timing Leaks in Rudimentary CPU Interrupt Logic

Jo Van Bulck Frank Piessens Raoul Strackx

imec-DistriNet, KU Leuven

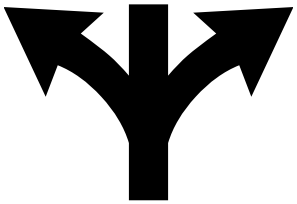
ACM CCS, October 2018



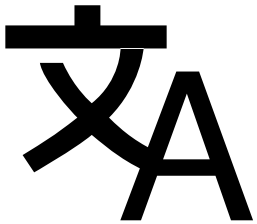

SHARING IS NOT CARING
SHARING IS LOSING
YOUR STUFF TO OTHERS
imgflip.com

Microarchitectural side-channels and where to find them

CPU cache



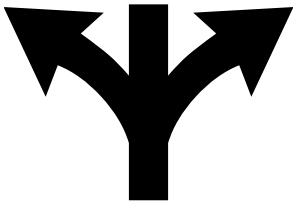
Branch prediction



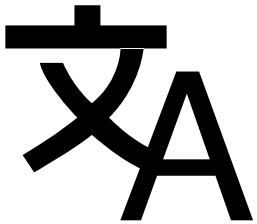
Address translation

Microarchitectural side-channels and where to find them

CPU cache

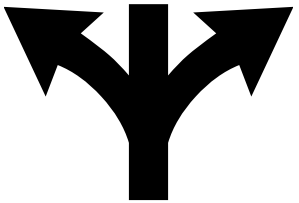
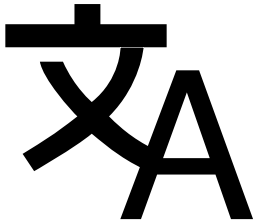


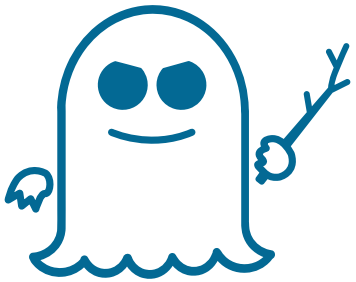
Branch prediction



Address translation

Microarchitectural side-channels and where to find them

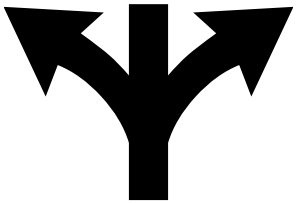
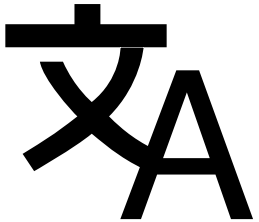




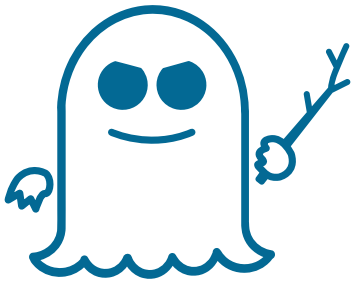



Intel response [Int18]

This is not a bug or a flaw . . . [side-channels] can't be eliminated

Microarchitectural side-channels and where to find them

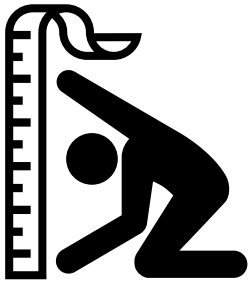



Intel response [Int18]

This is not a bug or a flaw . . . [side-channels] can't be eliminated

⇒ **Systematically study microarchitectural leakage**

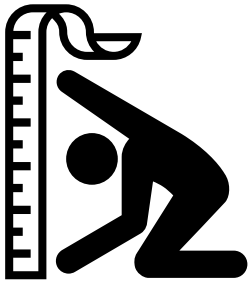
Nemesis: Studying rudimentary CPU interrupt logic



Overview

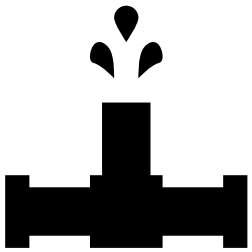
- ⇒ Interrupts leak **instruction execution times**
- ⇒ Determine control flow in **enclave** programs

Nemesis: Studying rudimentary CPU interrupt logic



Overview

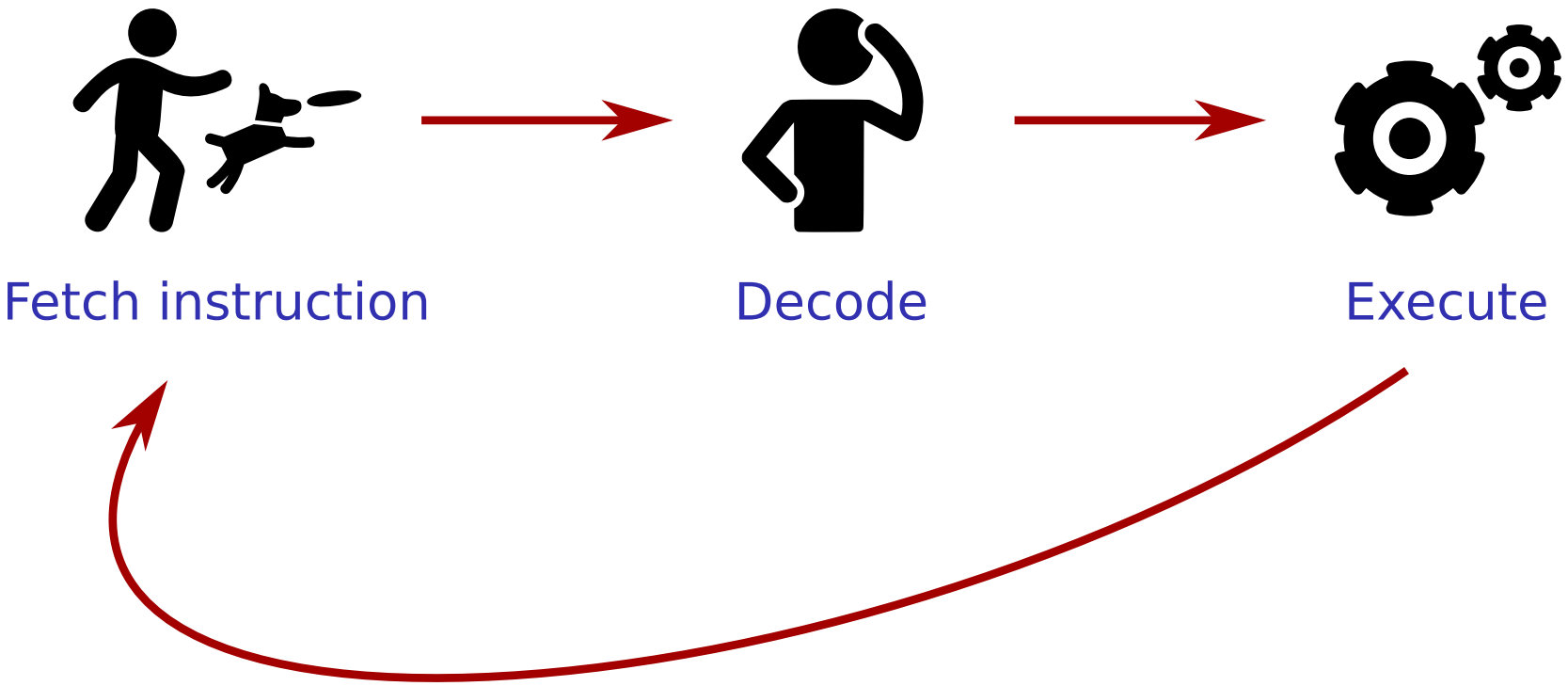
- ⇒ Interrupts leak **instruction execution times**
- ⇒ Determine control flow in **enclave** programs



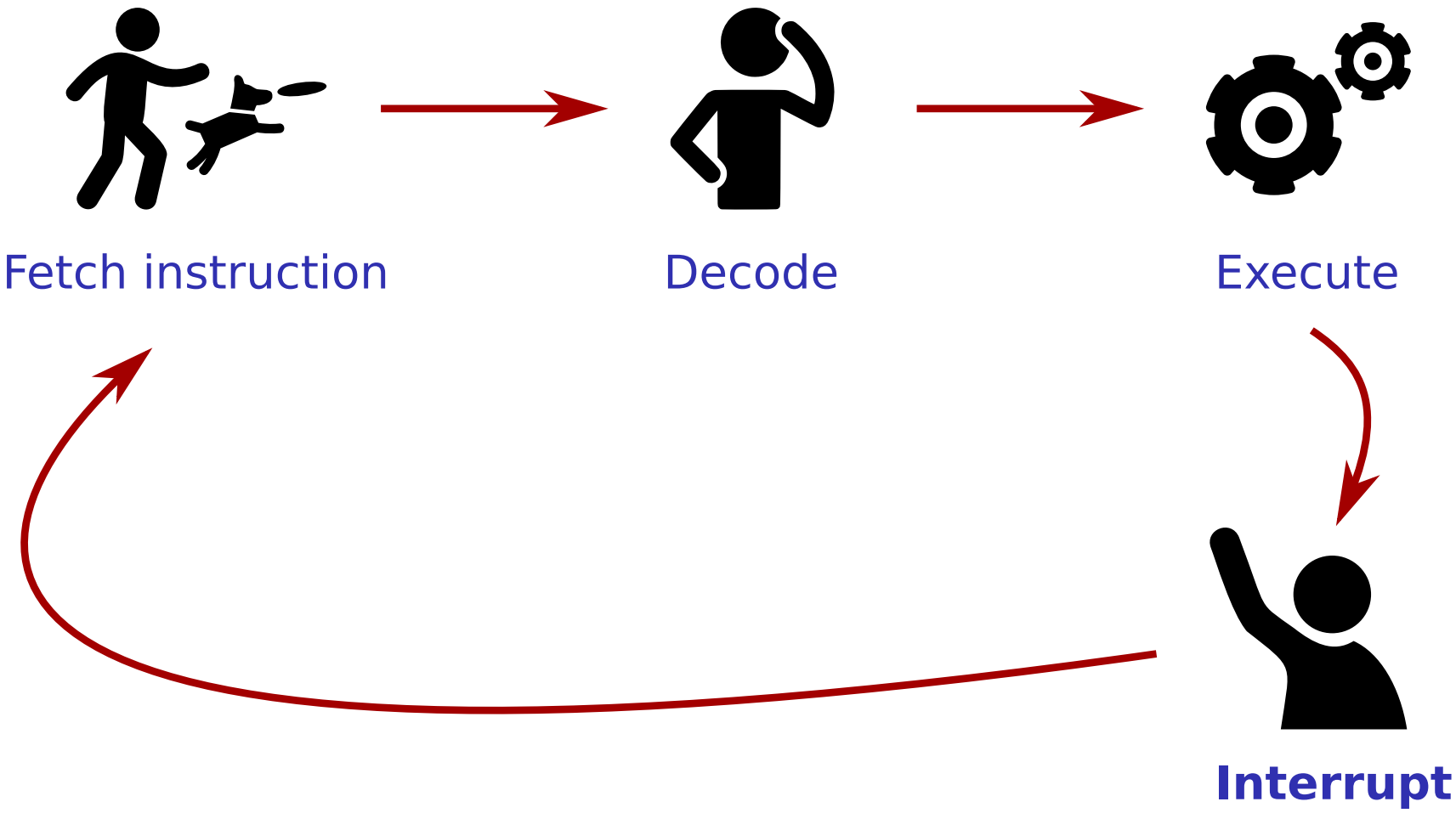
Research contributions

- ⇒ (First) remote μ-arch attack on **embedded** CPUs
- ⇒ Understanding **CPU pipeline** leakage (˜Meltdown)

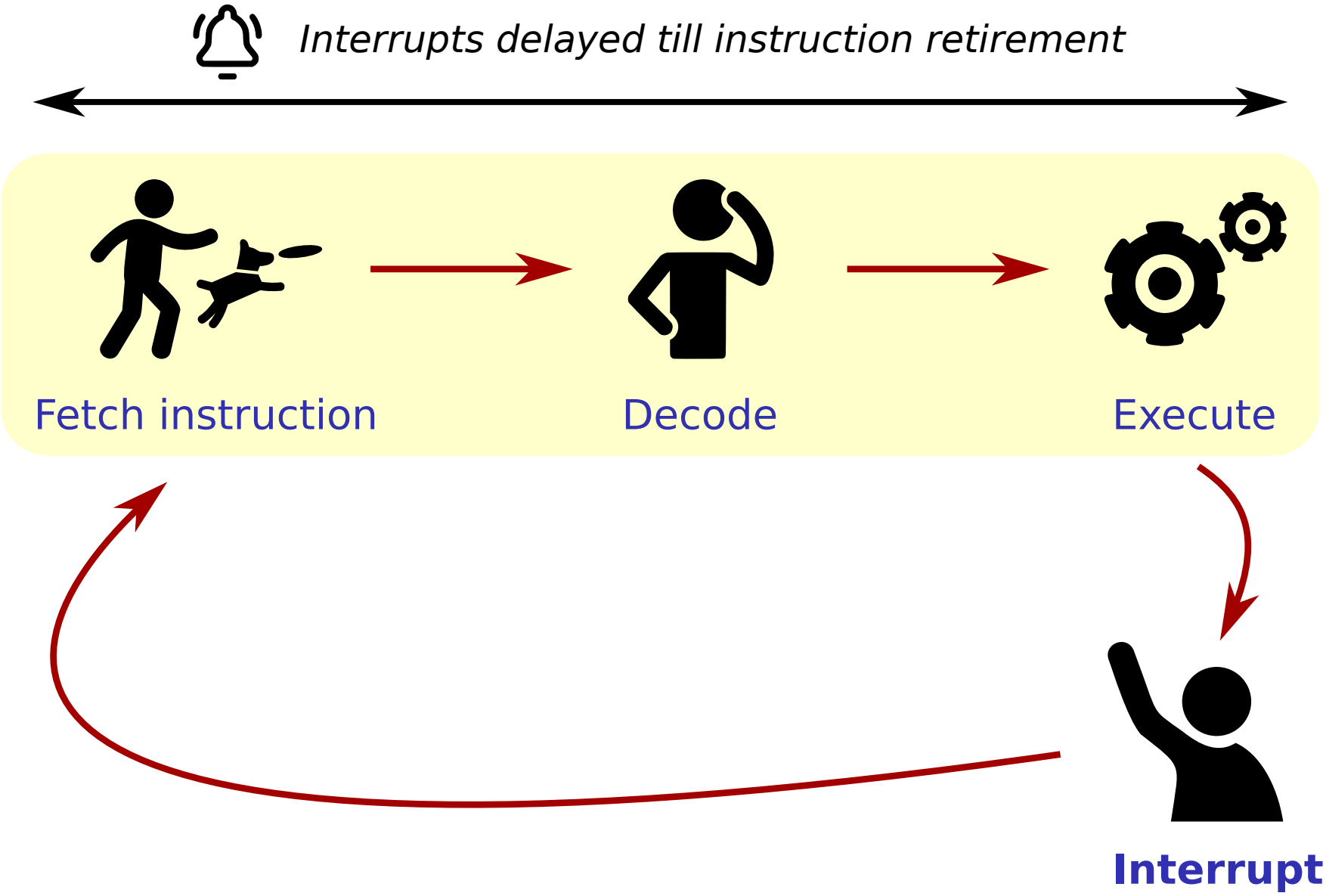
Back to basics: Fetch decode execute



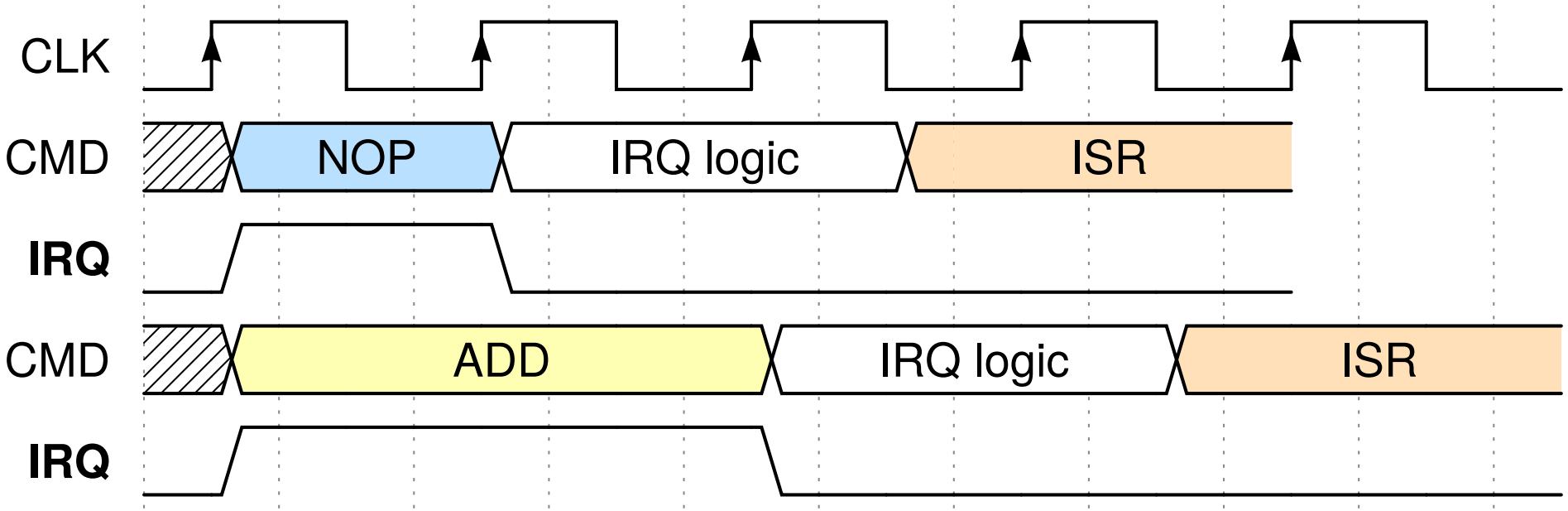
Back to basics: Fetch decode execute



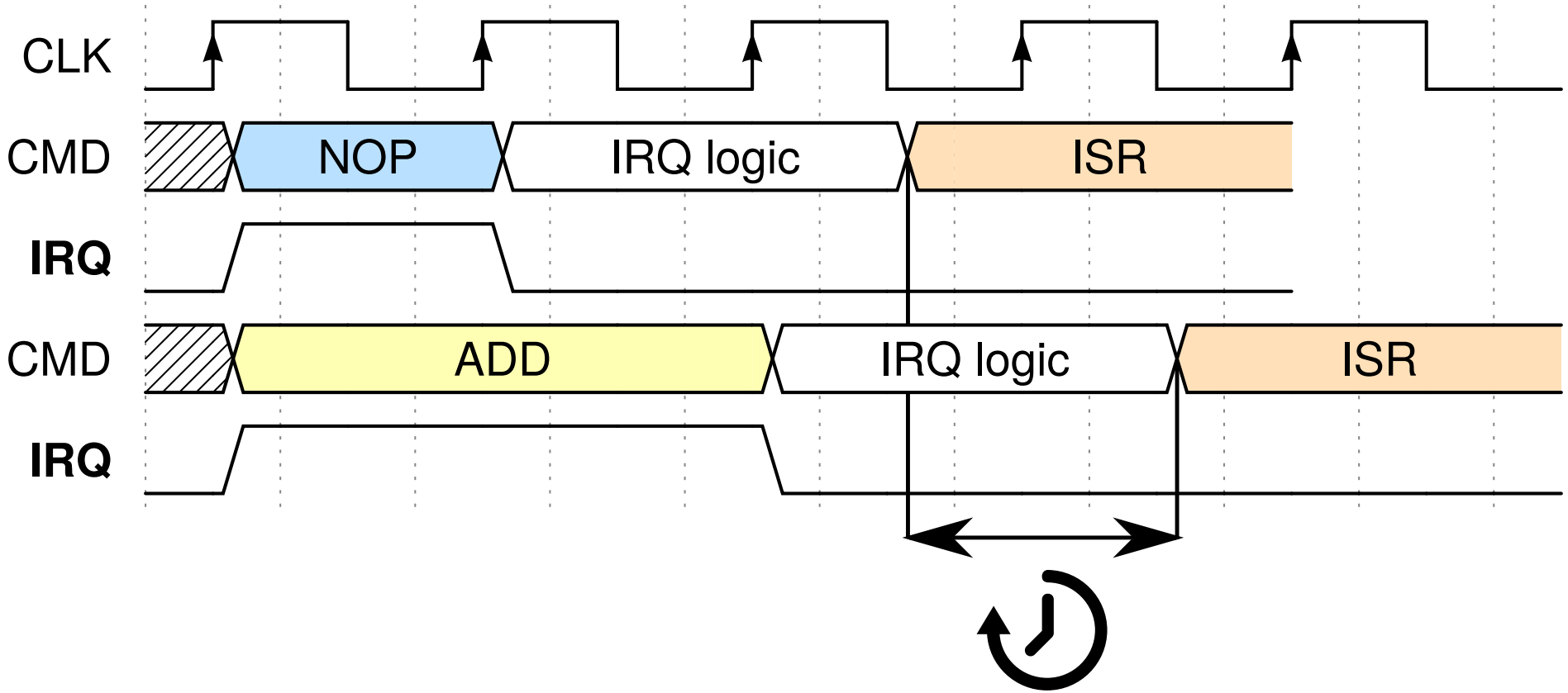
Back to basics: Fetch decode execute



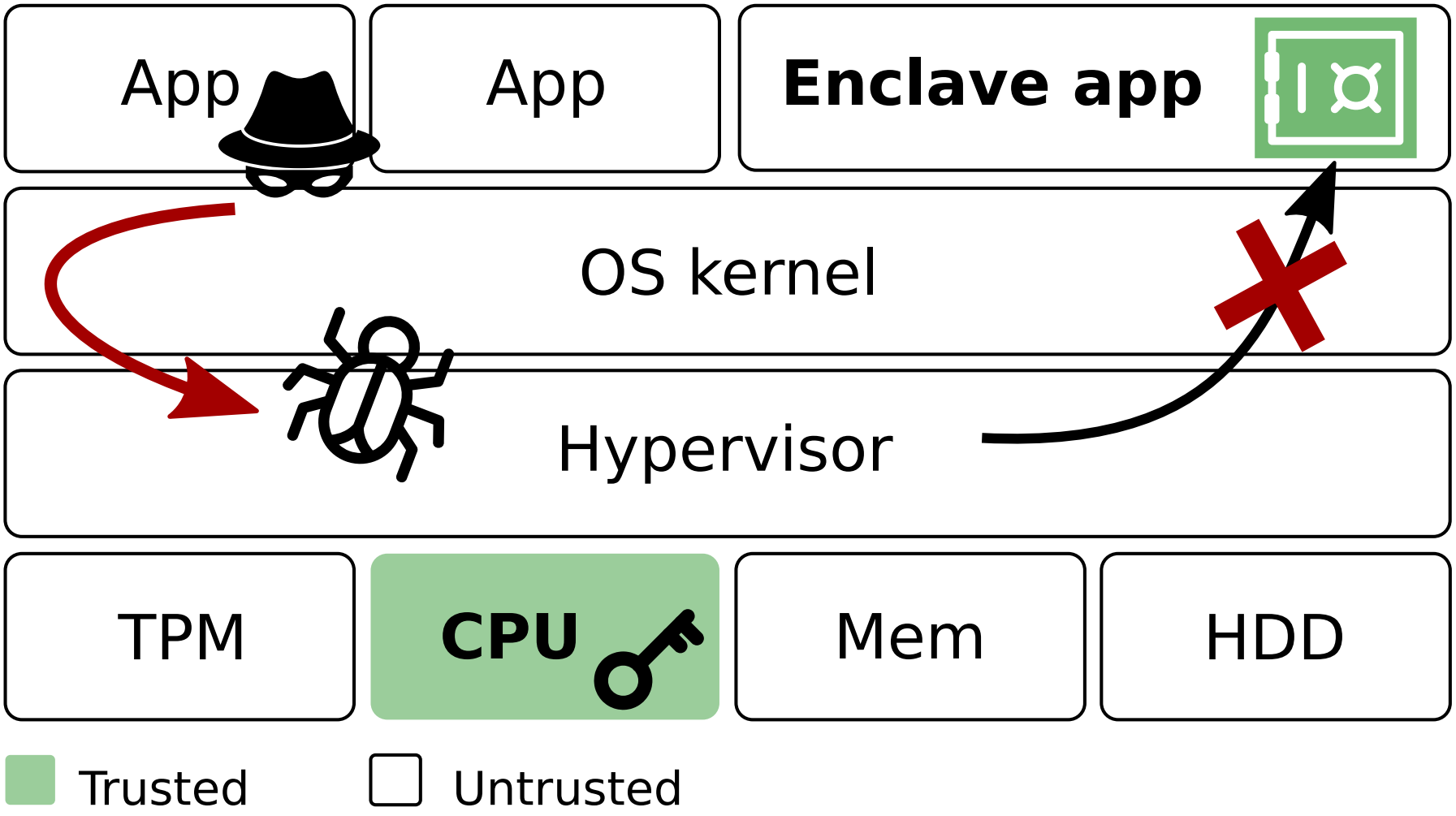
Wait a cycle: Interrupt latency as a side-channel



Wait a cycle: Interrupt latency as a side-channel

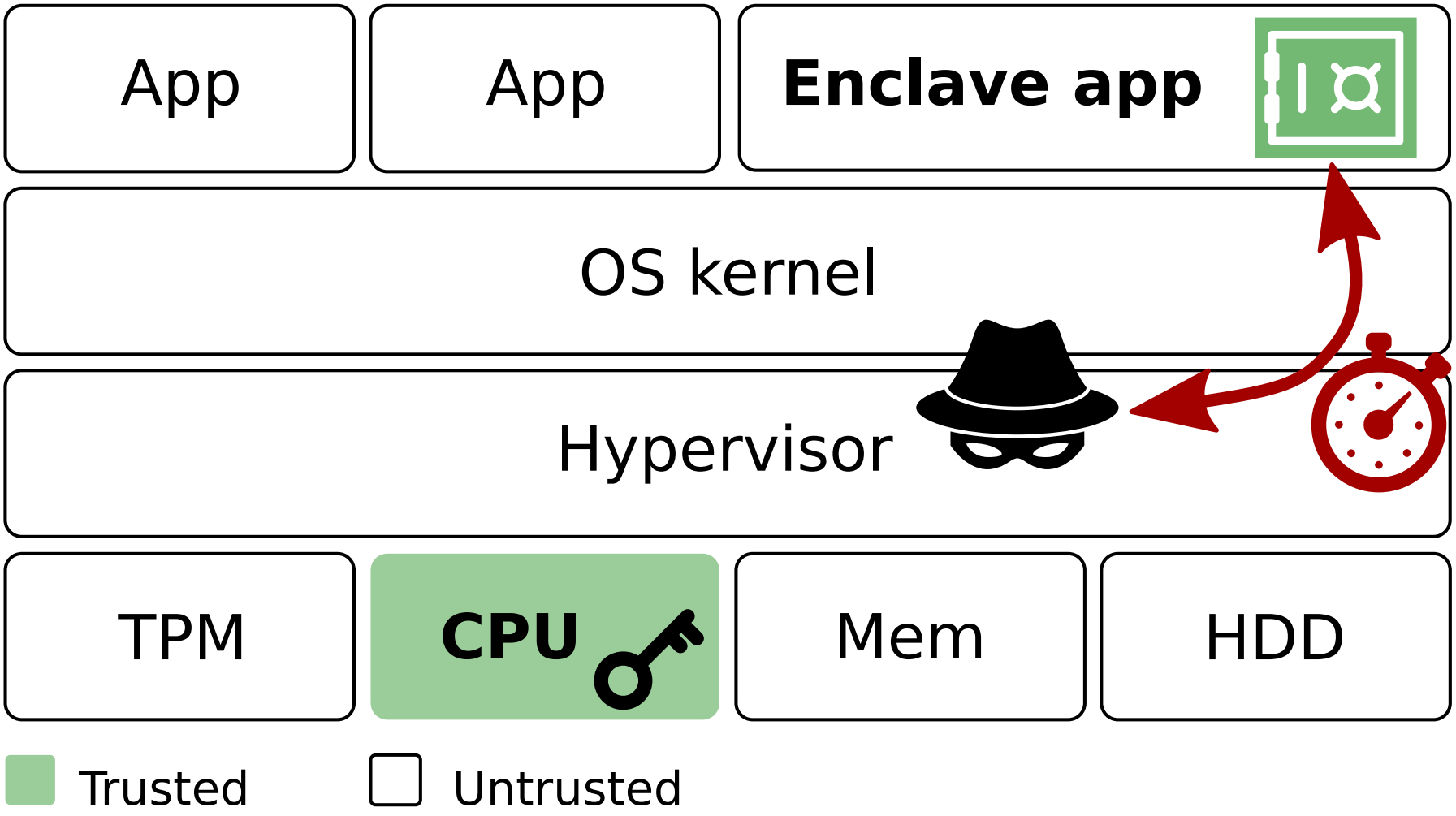


Enclaved execution adversary model



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

Enclaved execution adversary model

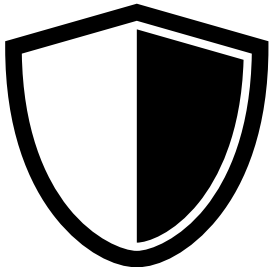


Untrusted OS → new class of powerful **side-channels**

Sancus: Open source trusted computing for the IoT

Embedded **enclaved execution:**

- ISA extensions for isolation & attestation
- Save + clear CPU state on enclave interrupt



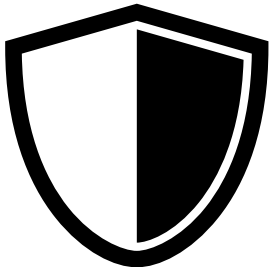
Noorman et al. "Sancus 2.0: A Low-Cost Security Architecture for IoT devices", TOPS 2017 [NVBM+17]

https://github.com/sancus-pma and https://distrinet.cs.kuleuven.be/software/sancus/

Sancus: Open source trusted computing for the IoT

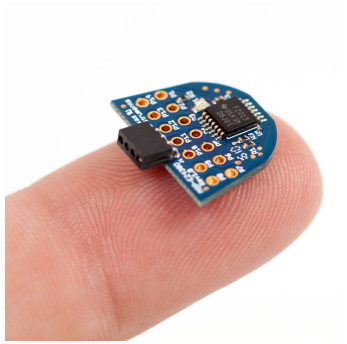
Embedded **enclaved execution:**

- ISA extensions for isolation & attestation
- Save + clear CPU state on enclave interrupt



Extremely **low-end processor** (openMSP430):

- Area: $\leq$ 2 kLUTs
- Deterministic execution: *no pipeline/cache/MMU/...*
- No known microarchitectural side-channels (!)



Noorman et al. "Sancus 2.0: A Low-Cost Security Architecture for IoT devices", TOPS 2017 [NVBM+17]

`https://github.com/sancus-pma` and `https://distrinet.cs.kuleuven.be/software/sancus/`

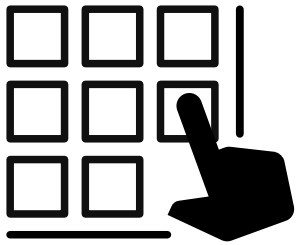
Secure input-output with Sancus enclaves

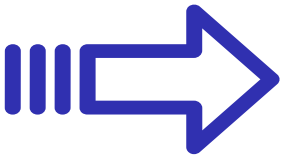
Driver enclave: Exclusive access to memory-mapped I/O device

Van Bulck et al. "VulCAN: Vehicular component authentication and software isolation", ACSAC 2017 [VBMP17]

Secure input-output with Sancus enclaves

Driver enclave: *16-bit vector* indicates which keys are down





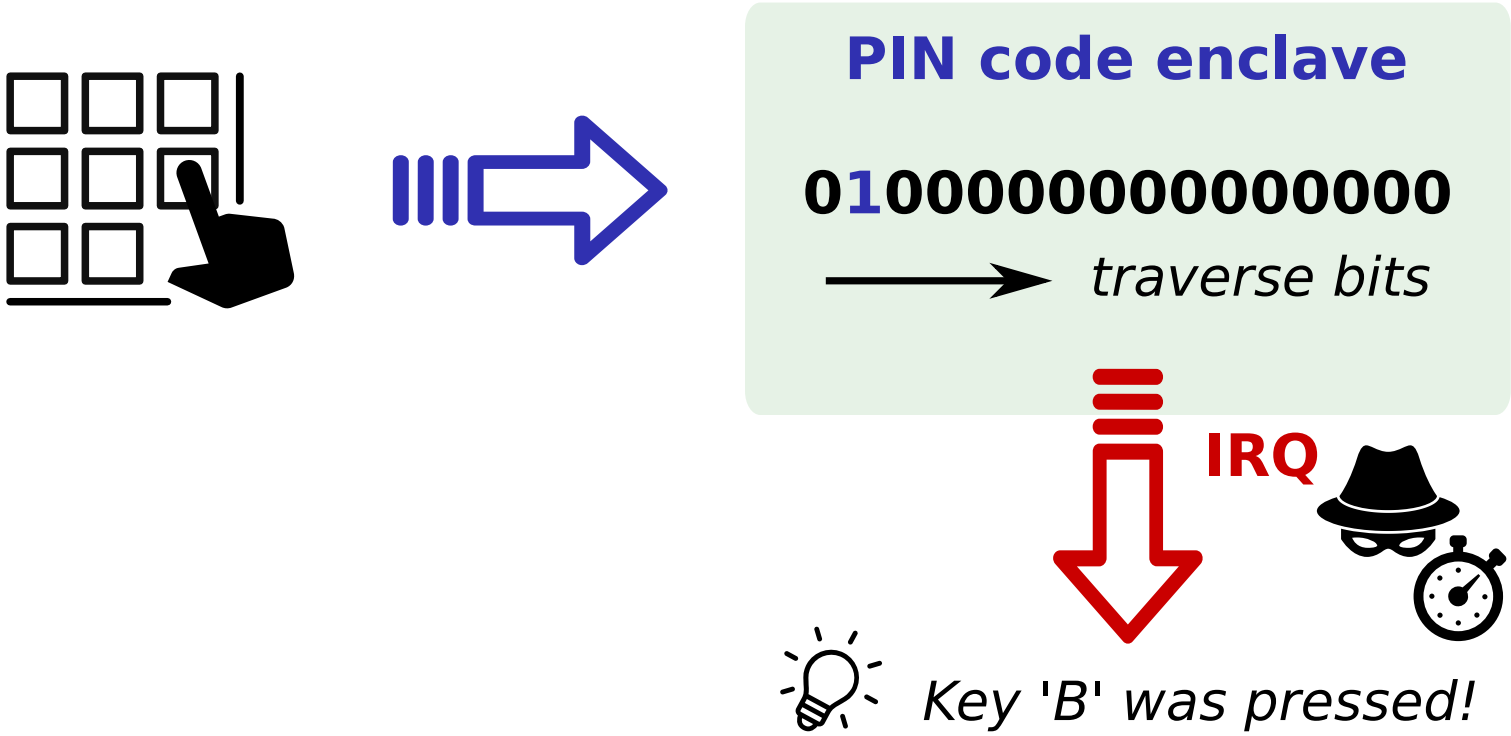
PIN code enclave

0100000000000000

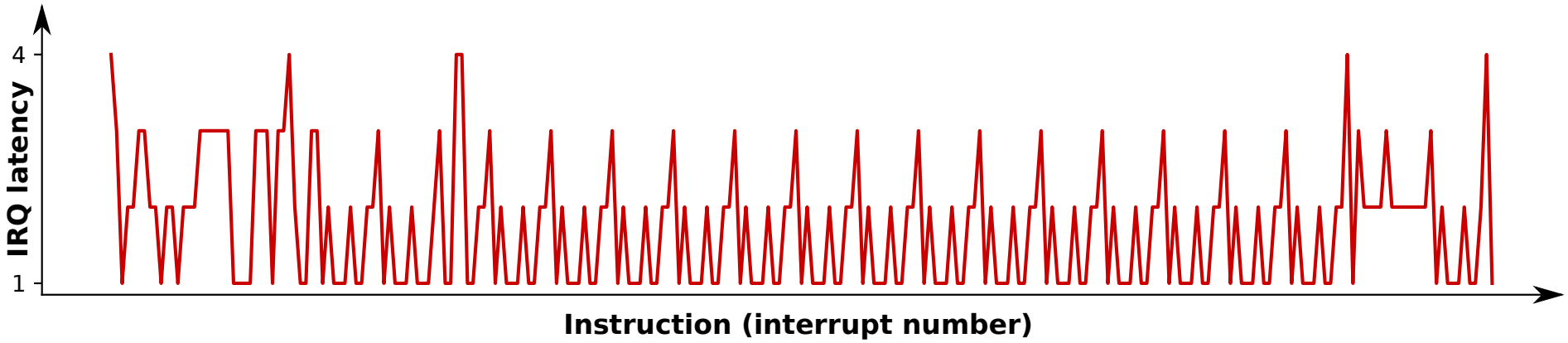
→ *traverse bits*

Secure input-output with Sancus enclaves

Attacker: Interrupt *conditional control flow* to infer secret PIN

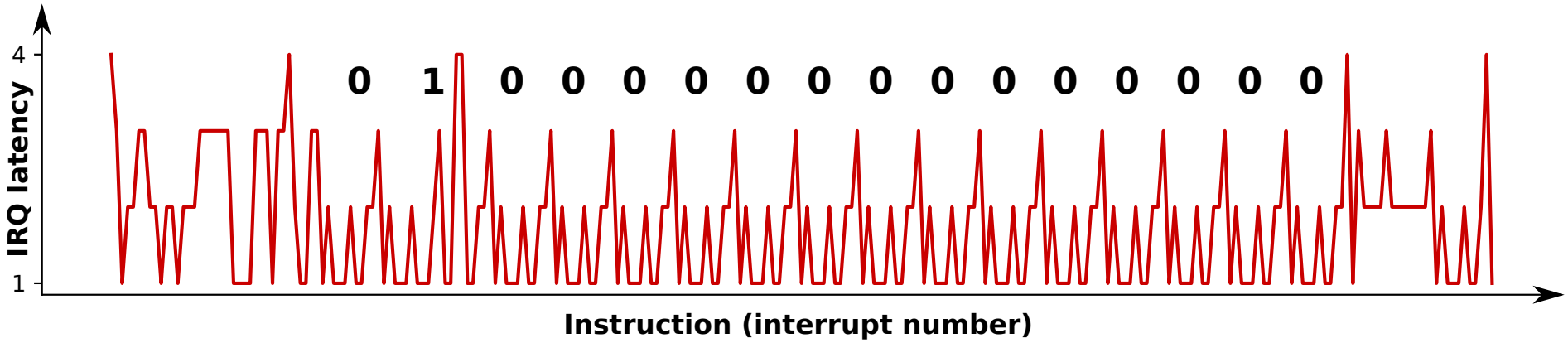


Sancus IRQ timing attack: Inferring key strokes



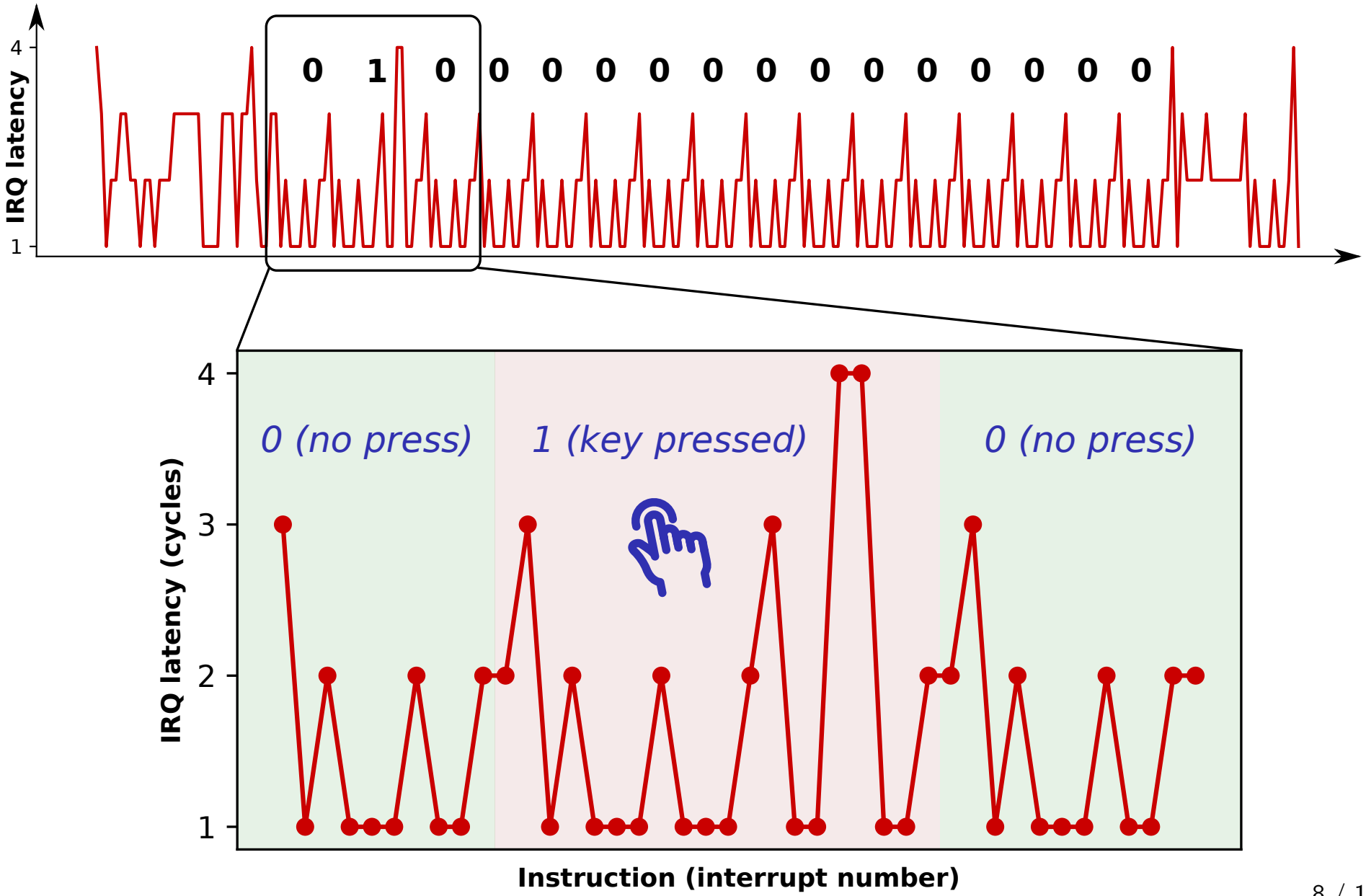
Enclave x-ray: Start-to-end trace enclaved execution

Sancus IRQ timing attack: Inferring key strokes



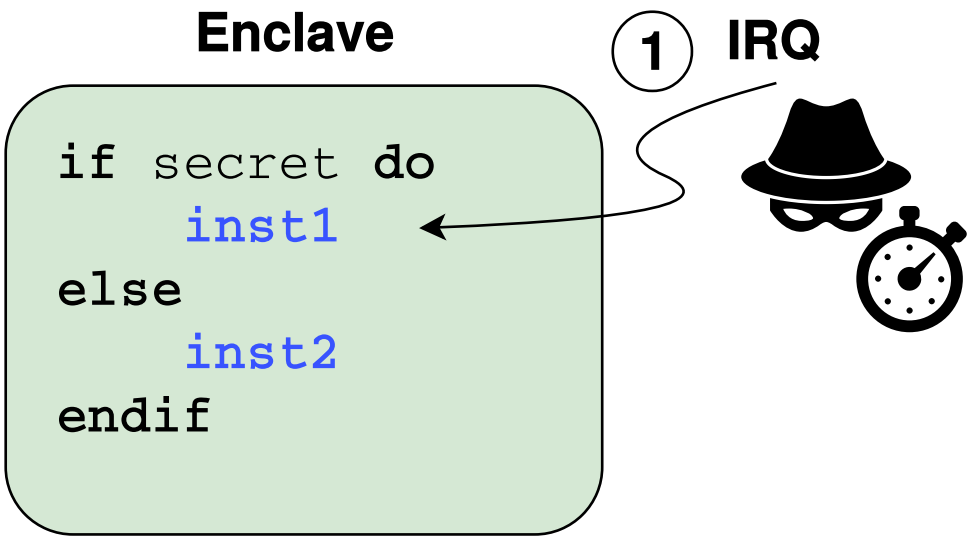
Enclave x-ray: Keymap bit traversal (ground truth)

Sancus IRQ timing attack: Inferring key strokes



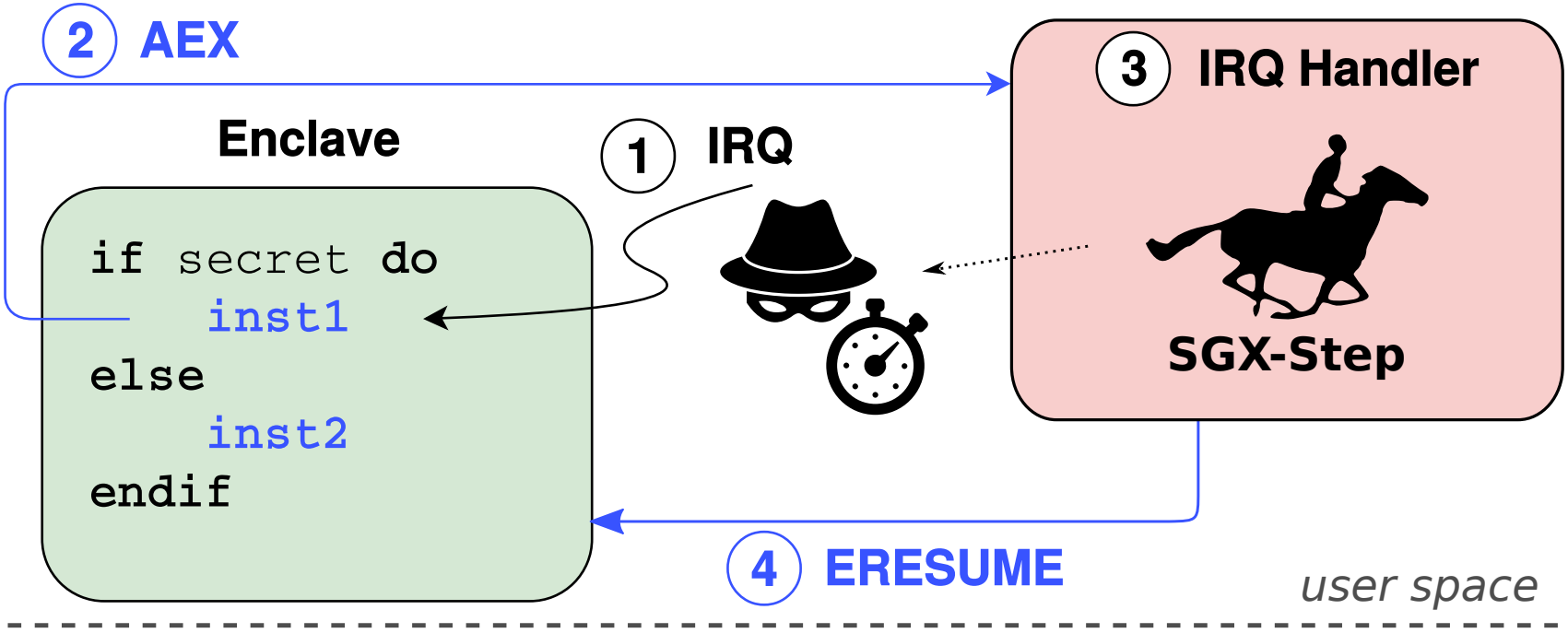
Interrupting and resuming Intel SGX enclaves

Challenge: x86 execution time prediction (timer) ☹



Interrupting and resuming Intel SGX enclaves

SGX-Step: user space APIC timer + IRQ handling ☺

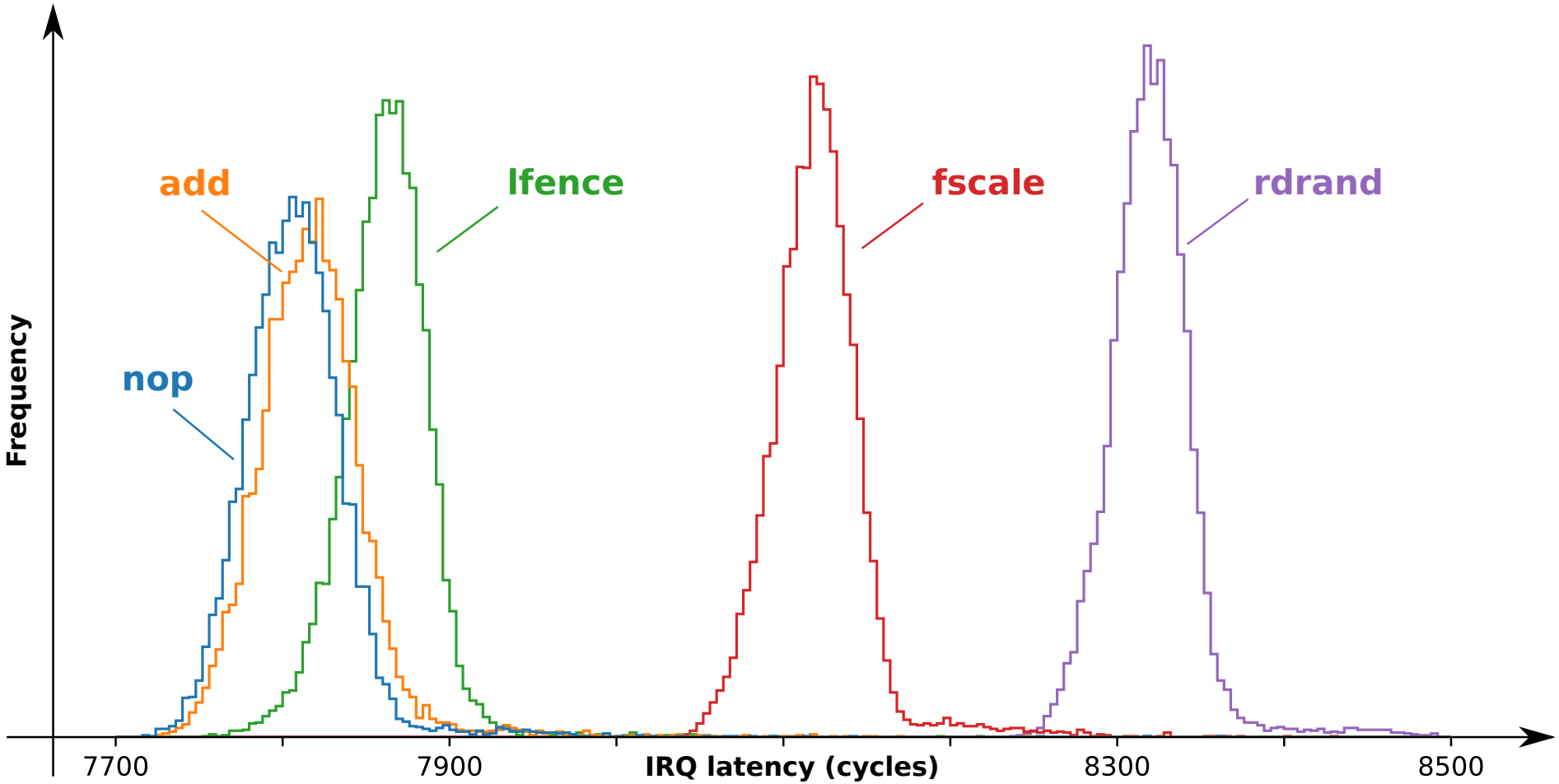


Van Bulck et al. "SGX-Step: A practical attack framework for precise enclave execution control", SysTEX 2017 [VBPS17]

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

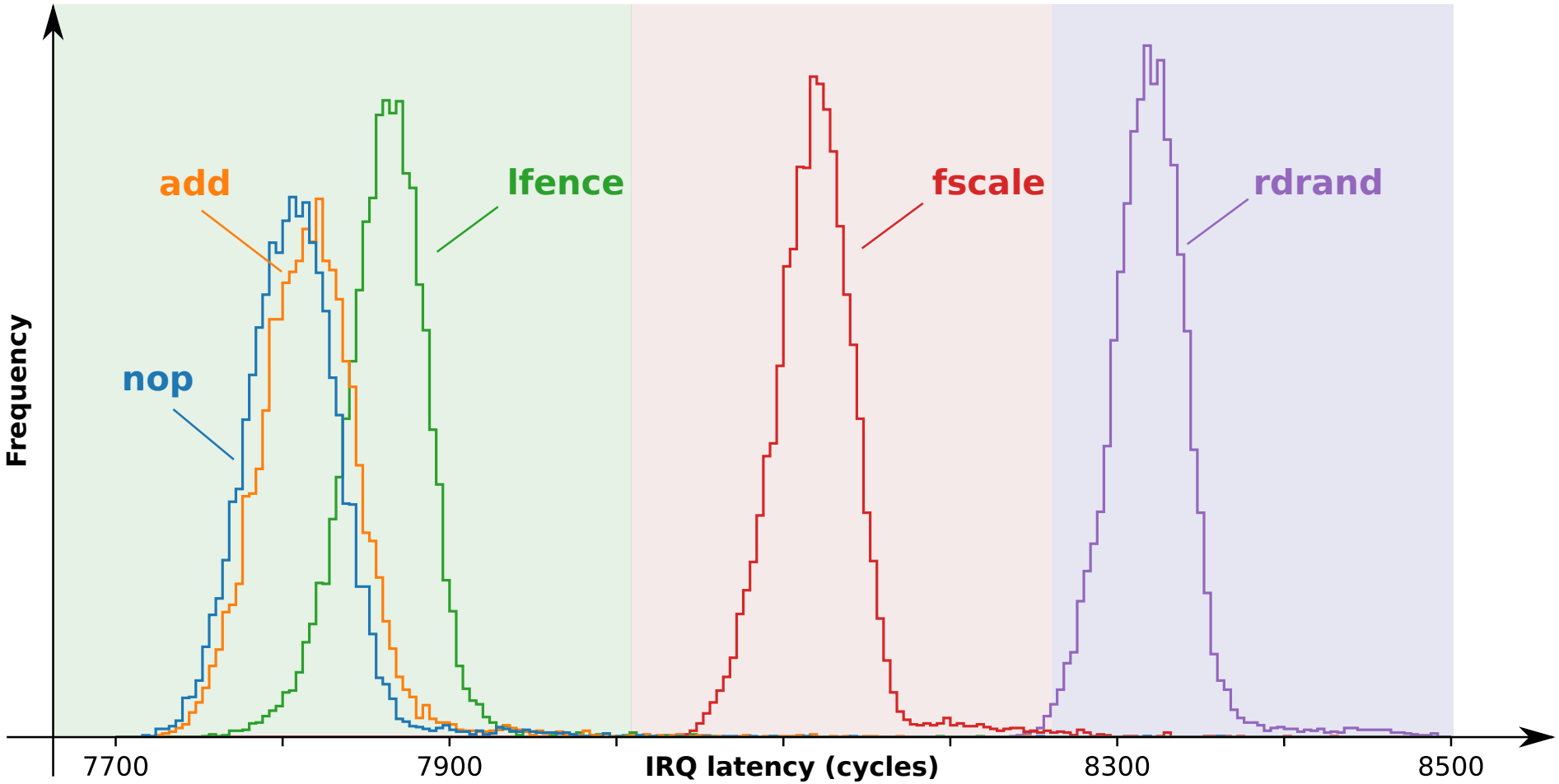
Microbenchmarks: Measuring x86 instruction latencies

Latency distribution: 10,000 samples from benchmark enclave



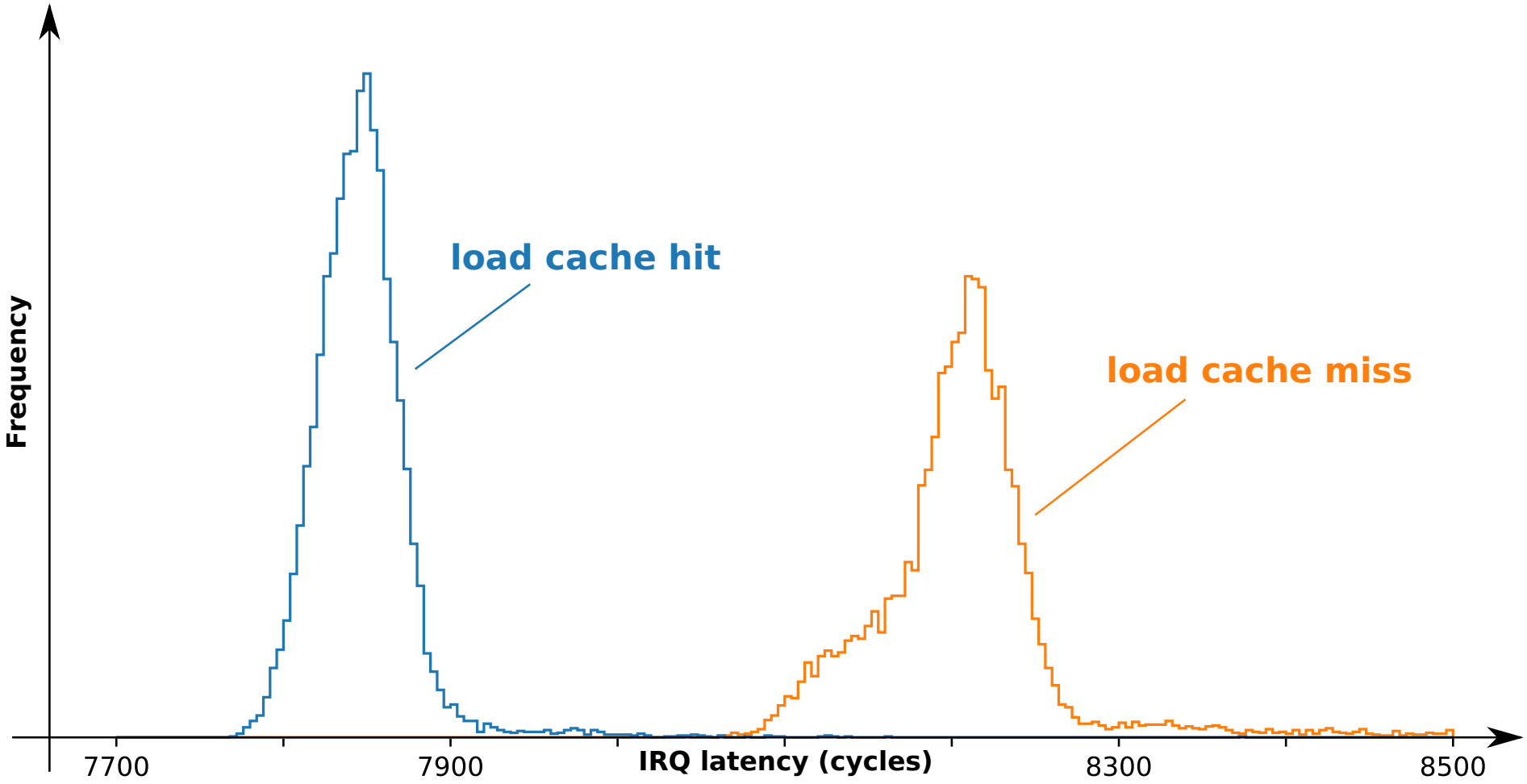
Microbenchmarks: Measuring x86 instruction latencies

Timing leak: reconstruct *instruction latency class*



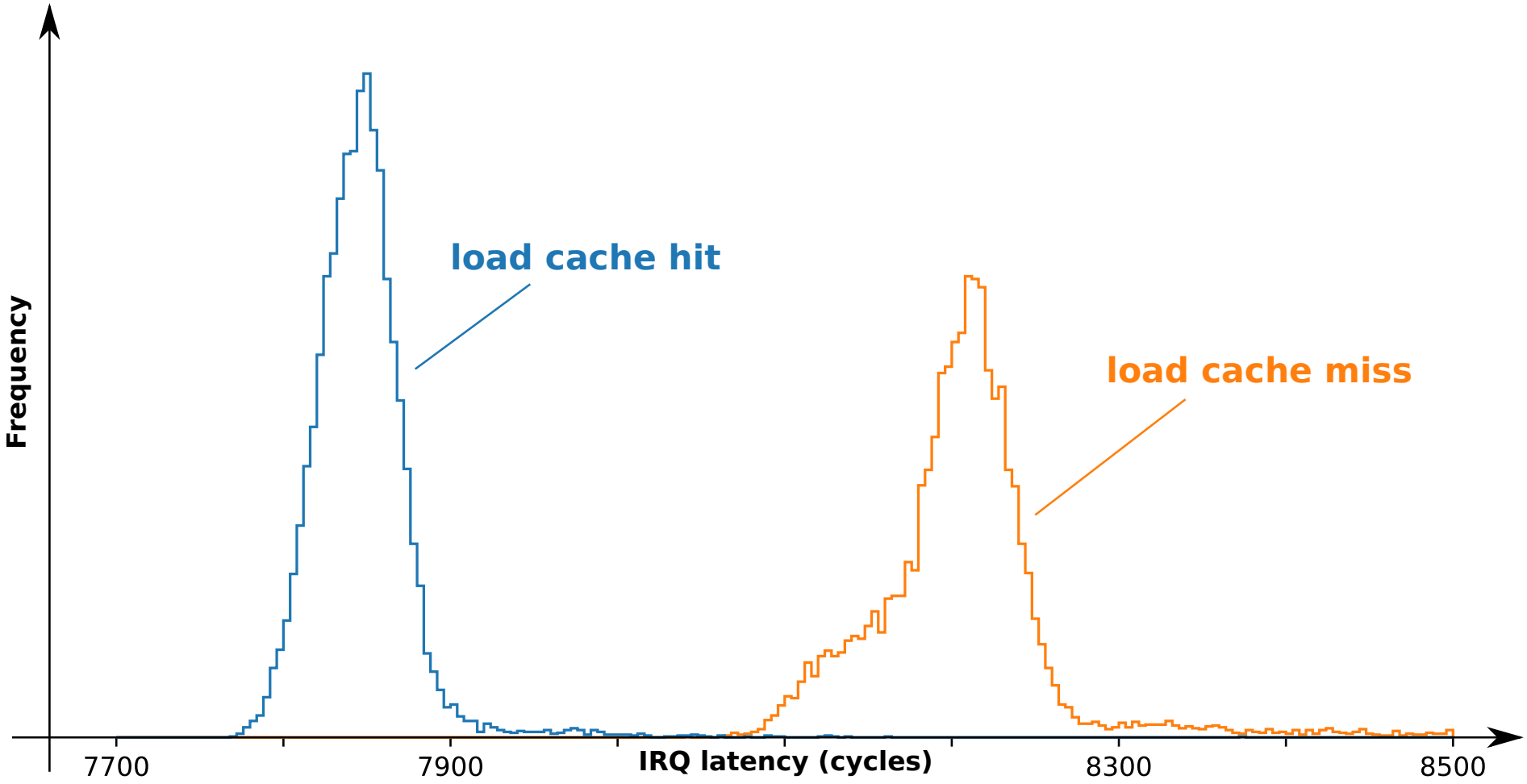
Microbenchmarks: Measuring x86 cache misses

Timing leak: reconstruct *micro-architectural cache state*



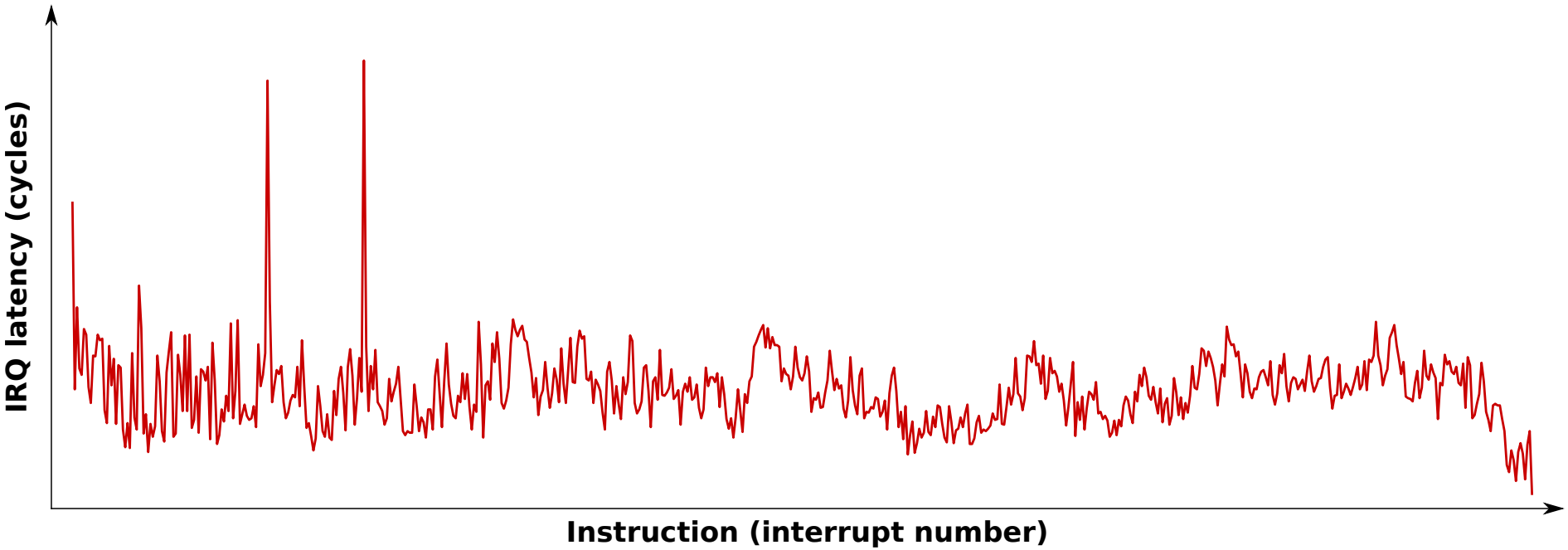
Microbenchmarks: Measuring x86 cache misses

Timing leak: many more → see paper!



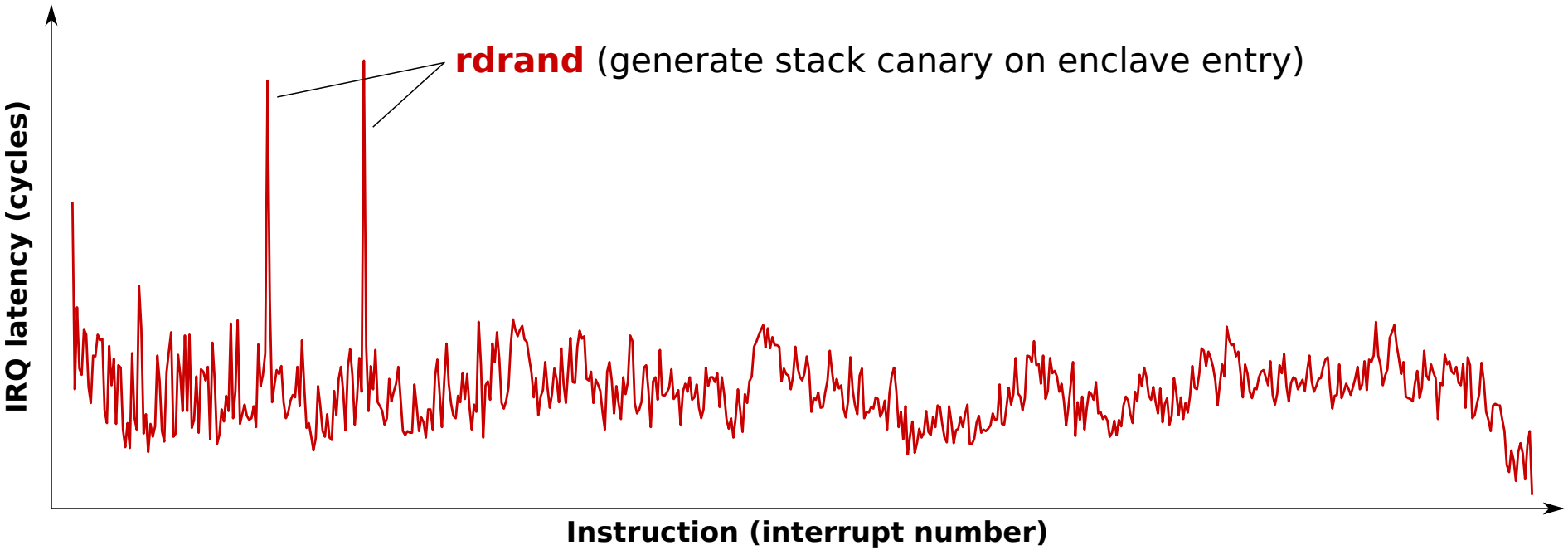
Single-stepping SGX enclaves in practice

Enclave x-ray: Start-to-end trace enclaved execution



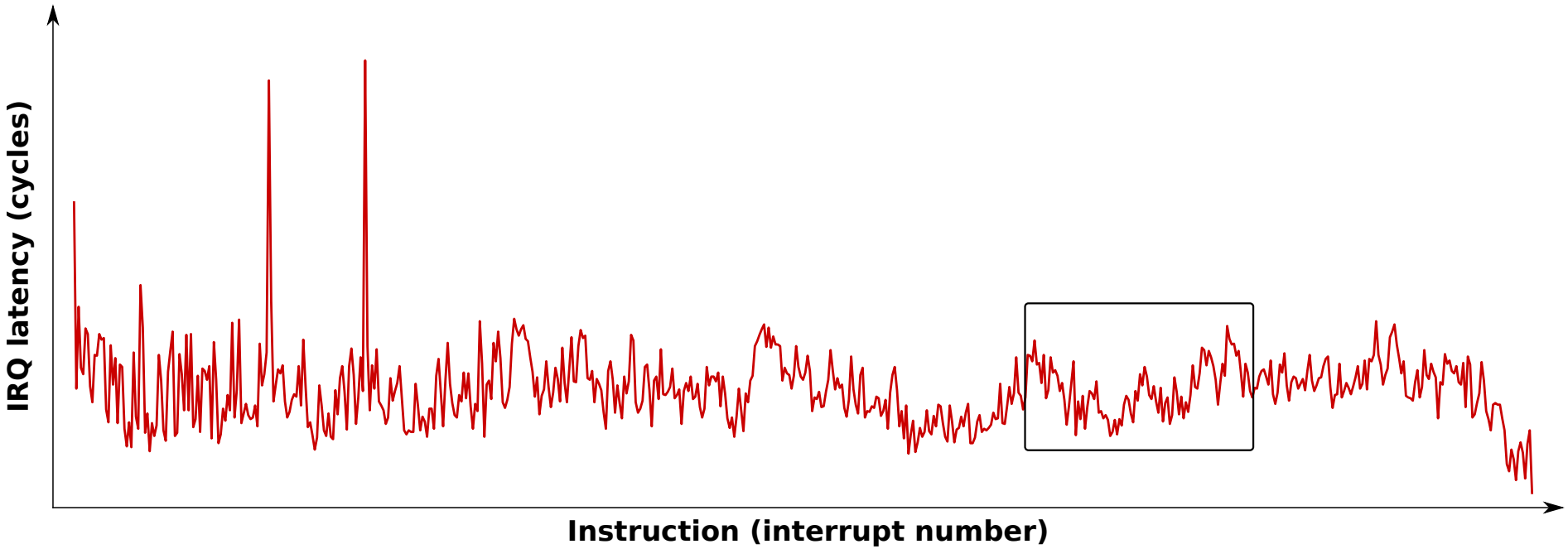
Single-stepping SGX enclaves in practice

Enclave x-ray: Spotting high-latency instructions



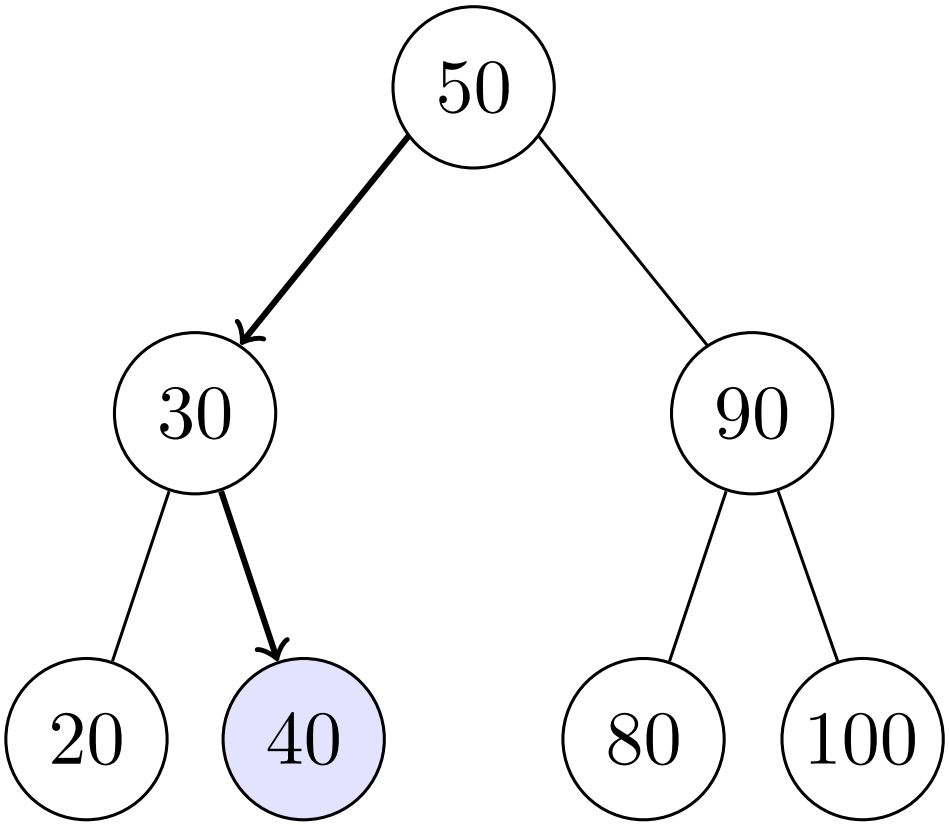
Single-stepping SGX enclaves in practice

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



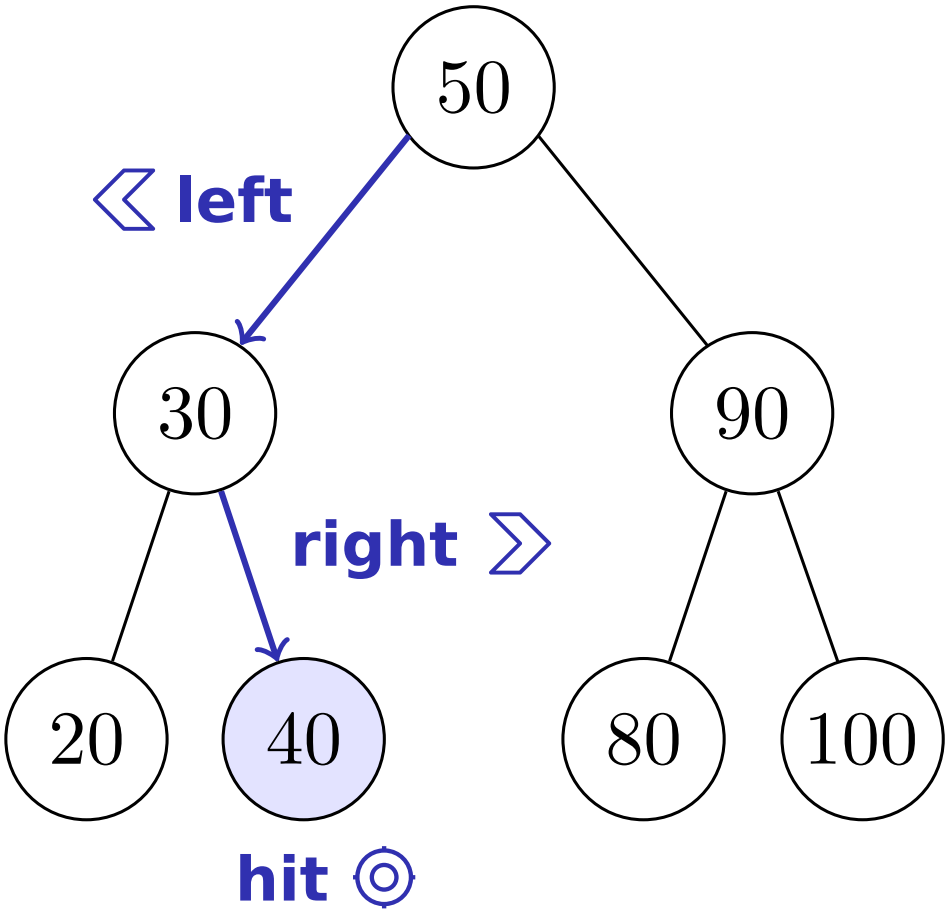
De-anonymizing enclave lookups

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



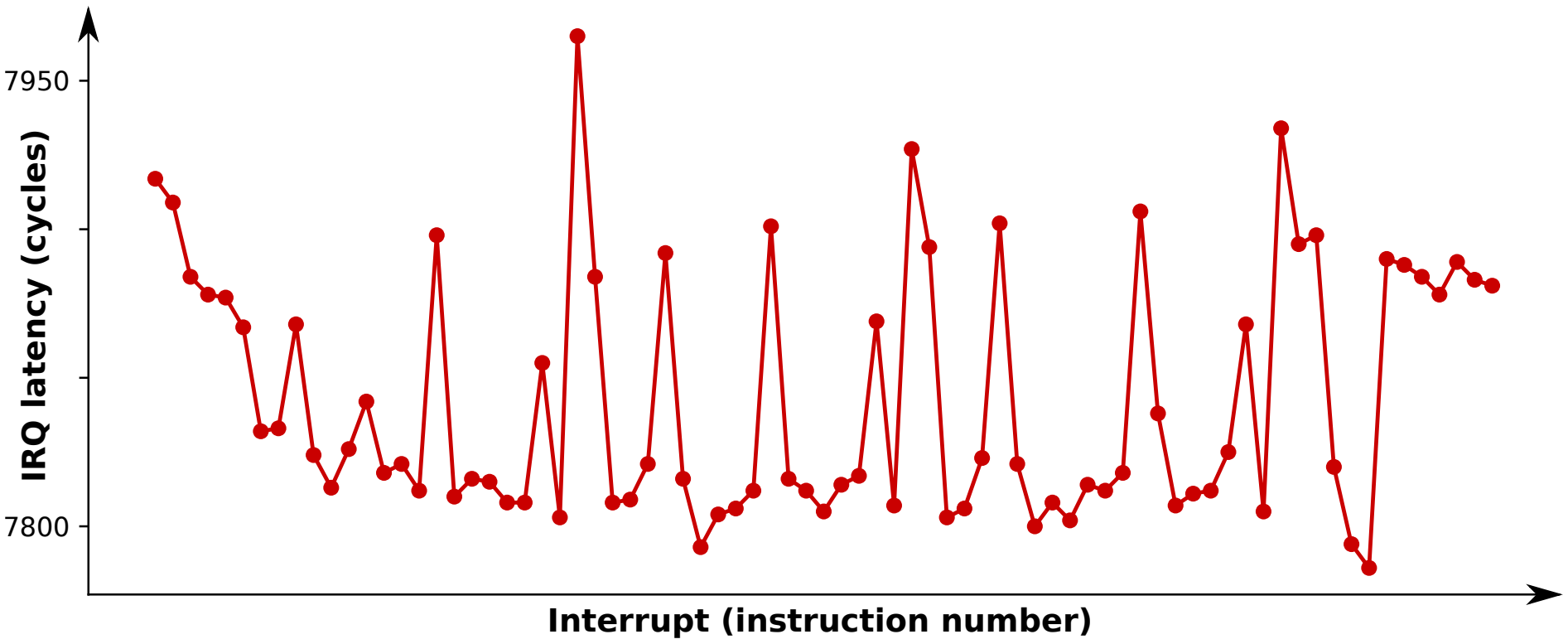
De-anonymizing enclave lookups

Adversary: Infer secret lookup in known array



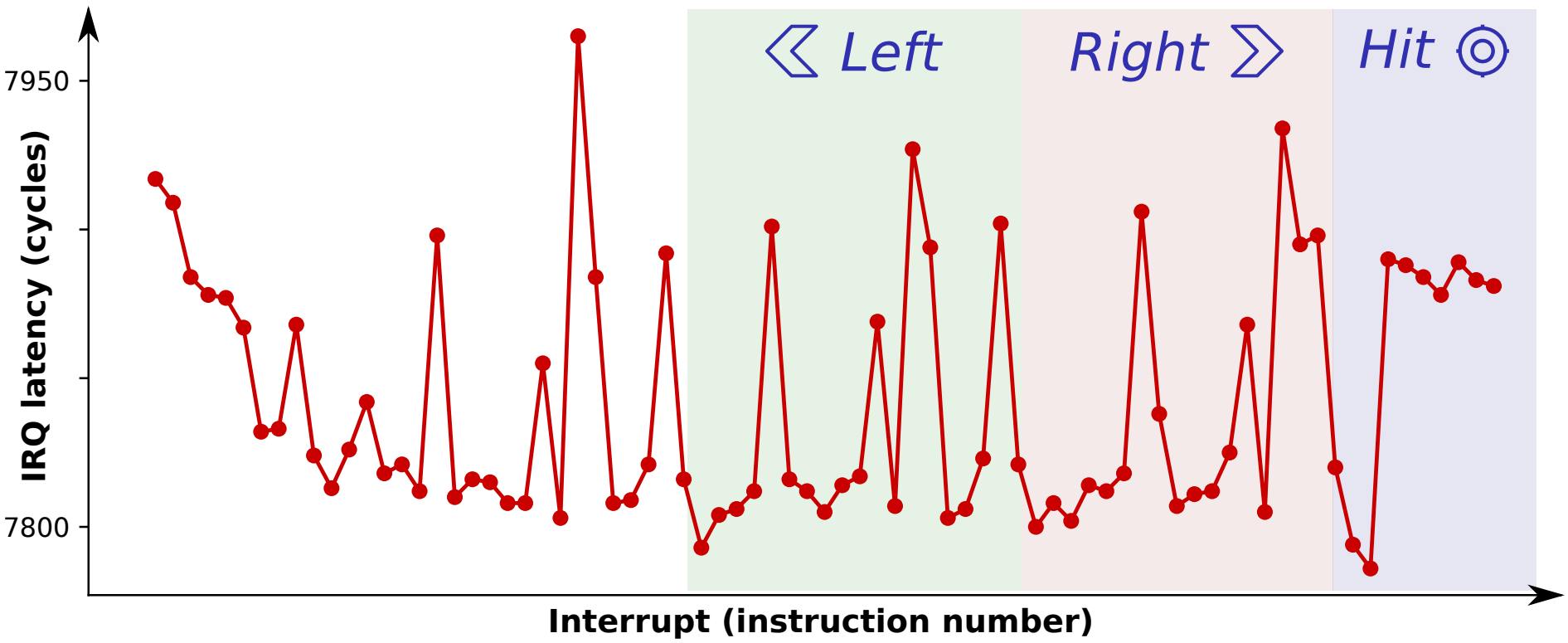
De-anonymizing enclave lookups

Goal: Infer lookup → reconstruct `bsearch` control flow



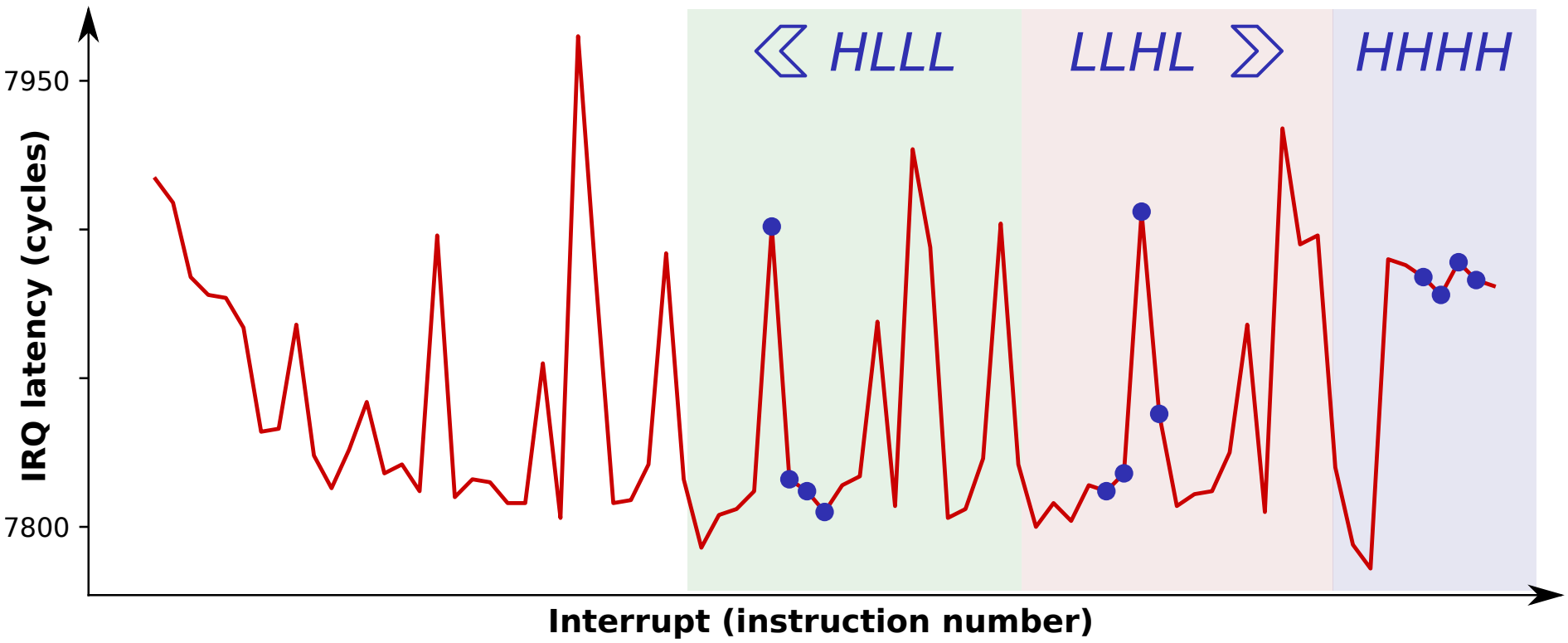
De-anonymizing enclave lookups

Goal: Infer lookup → reconstruct `bsearch` control flow

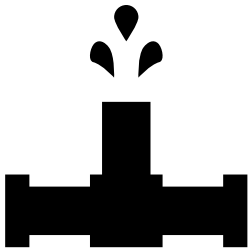


De-anonymizing enclave lookups

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



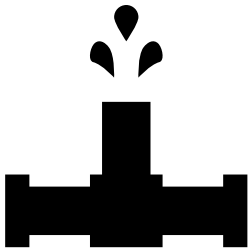
Conclusions



Nemesis contributions

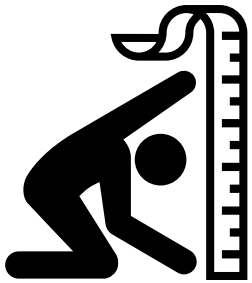
- ⇒ Understanding **CPU interrupt** leakage
- ⇒ (First) **embedded** + high-end μ-arch channel

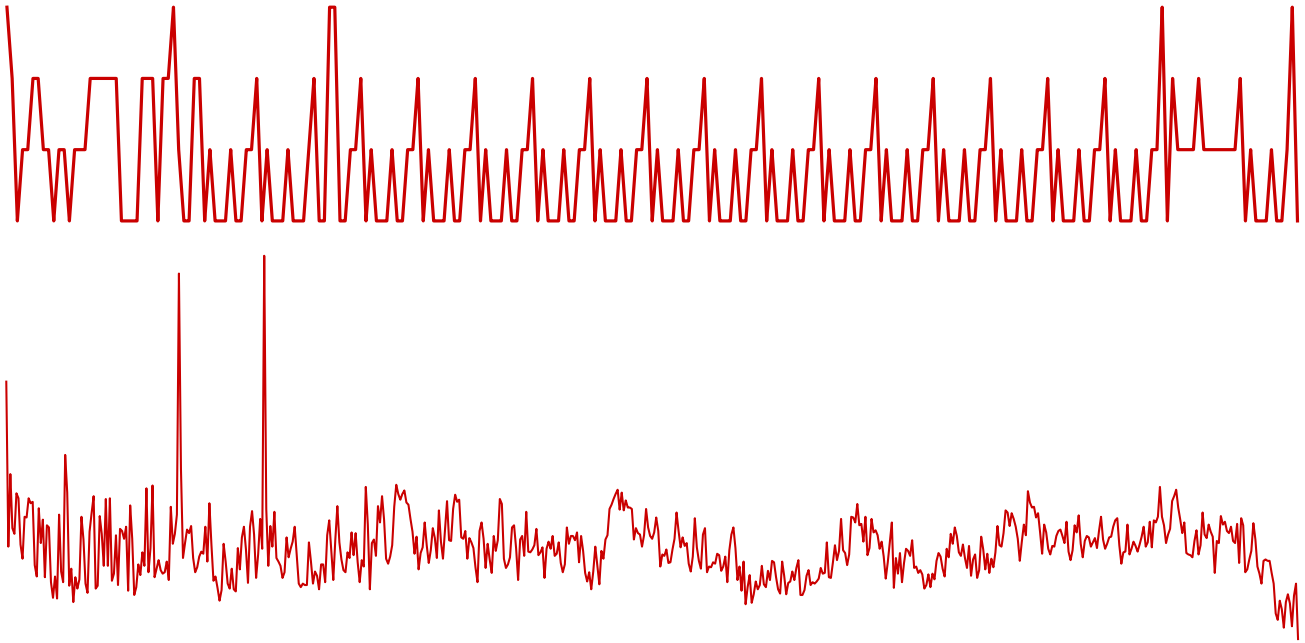
Conclusions



Nemesis contributions

- ⇒ Understanding **CPU interrupt** leakage
- ⇒ (First) **embedded** + high-end μ-arch channel





https://github.com/jovanbulck/nemesis

References I

Intel Corporation.
Resources and response to side channel variants 1, 2, 3.
intel.com/content/www/us/en/architecture-and-technology/side-channel-variants-1-2-3.html, 2018.

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 *Proceedings of the 26th USENIX Security Symposium*. USENIX Association, 2017.

J. Noorman, J. T. Mühlberg, and F. Piessens.
Authentic execution of distributed event-driven applications with a small TCB.
In *13th International Workshop on Security and Trust Management (STM'17)*, vol. 10547 of *LNCS*, pp. 55–71, Heidelberg, 2017. Springer.

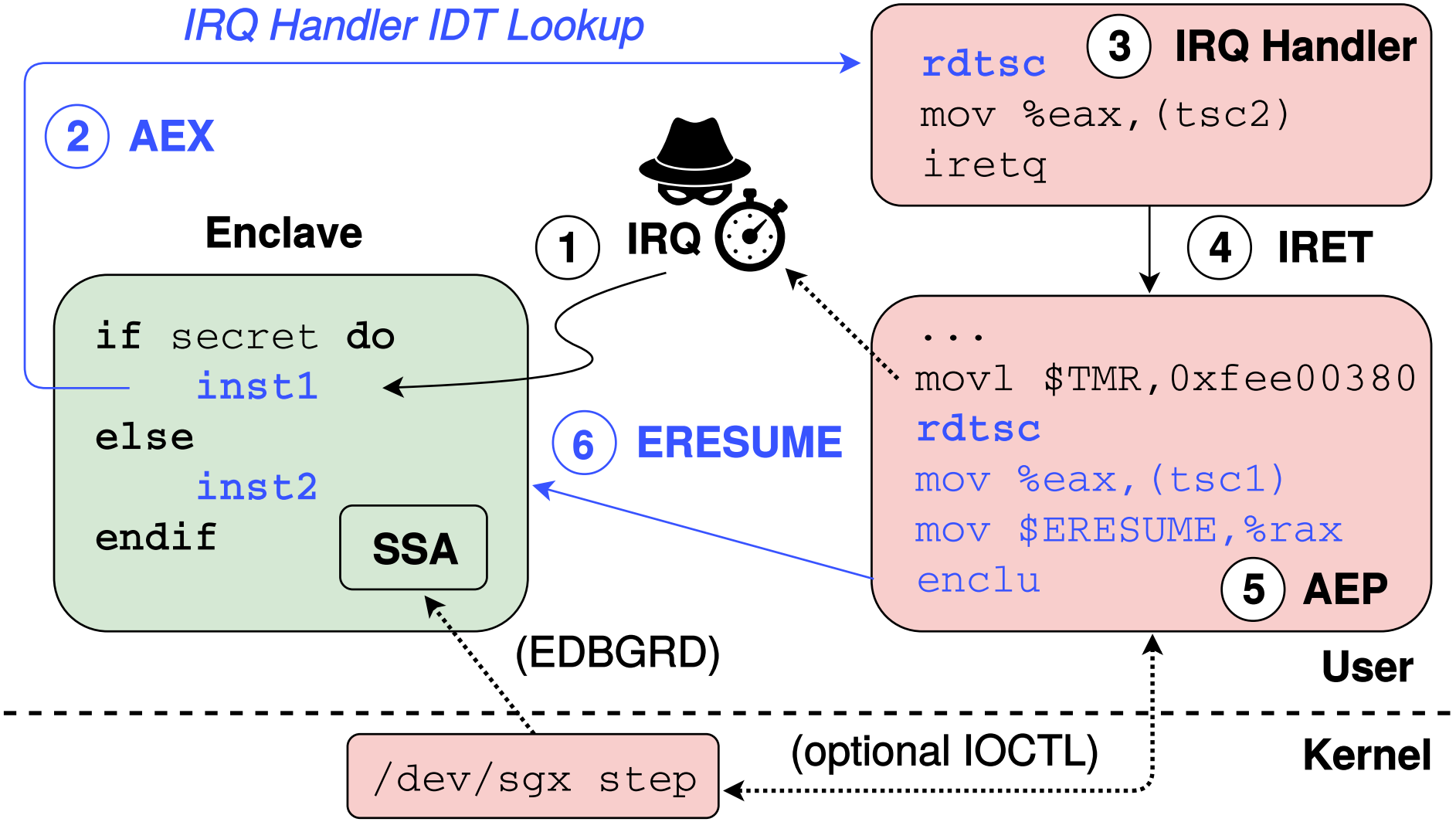
J. Noorman, J. Van Bulck, J. T. Mühlberg, F. Piessens, P. Maene, B. Preneel, I. Verbauwhede, J. Götzfried, T. Müller, and F. Freiling.
Sancus 2.0: A low-cost security architecture for IoT devices.
ACM Transactions on Privacy and Security (TOPS), 2017.

J. Van Bulck, J. T. Mühlberg, and F. Piessens.
VulCAN: Efficient component authentication and software isolation for automotive control networks.
In *Proceedings of the 33th Annual Computer Security Applications Conference (ACSAC'17)*. ACM, 2017.

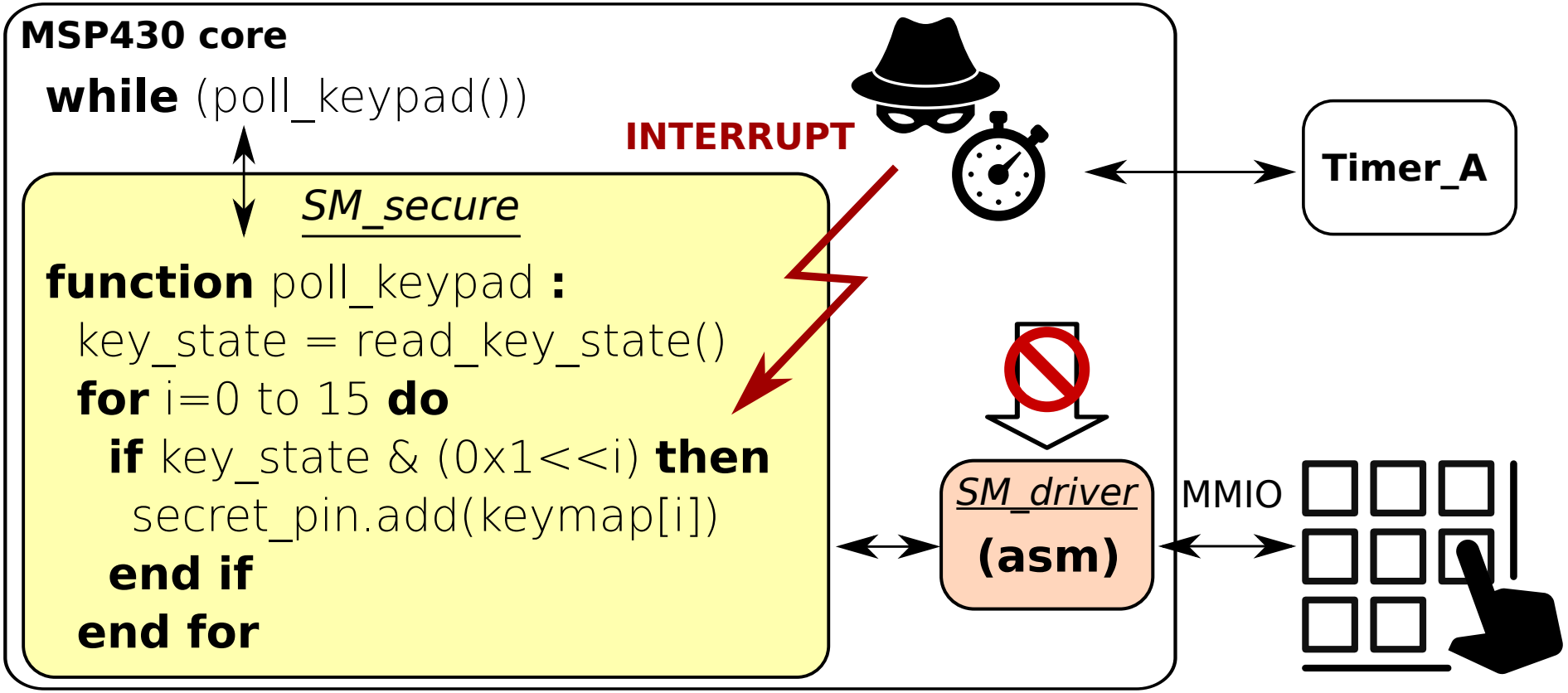
J. Van Bulck, J. Noorman, J. T. Mühlberg, and F. Piessens.
Towards availability and real-time guarantees for protected module architectures.
In *Companion Proceedings of the 15th International Conference on Modularity (MASS'16)*, pp. 146–151. ACM, 2016.

J. Van Bulck, F. Piessens, and R. Strackx.
SGX-Step: A practical attack framework for precise enclave execution control.
In *Proceedings of the 2nd Workshop on System Software for Trusted Execution*, SysTEX'17, pp. 4:1–4:6. ACM, 2017.

Appendix: Interrupting and resuming SGX enclaves

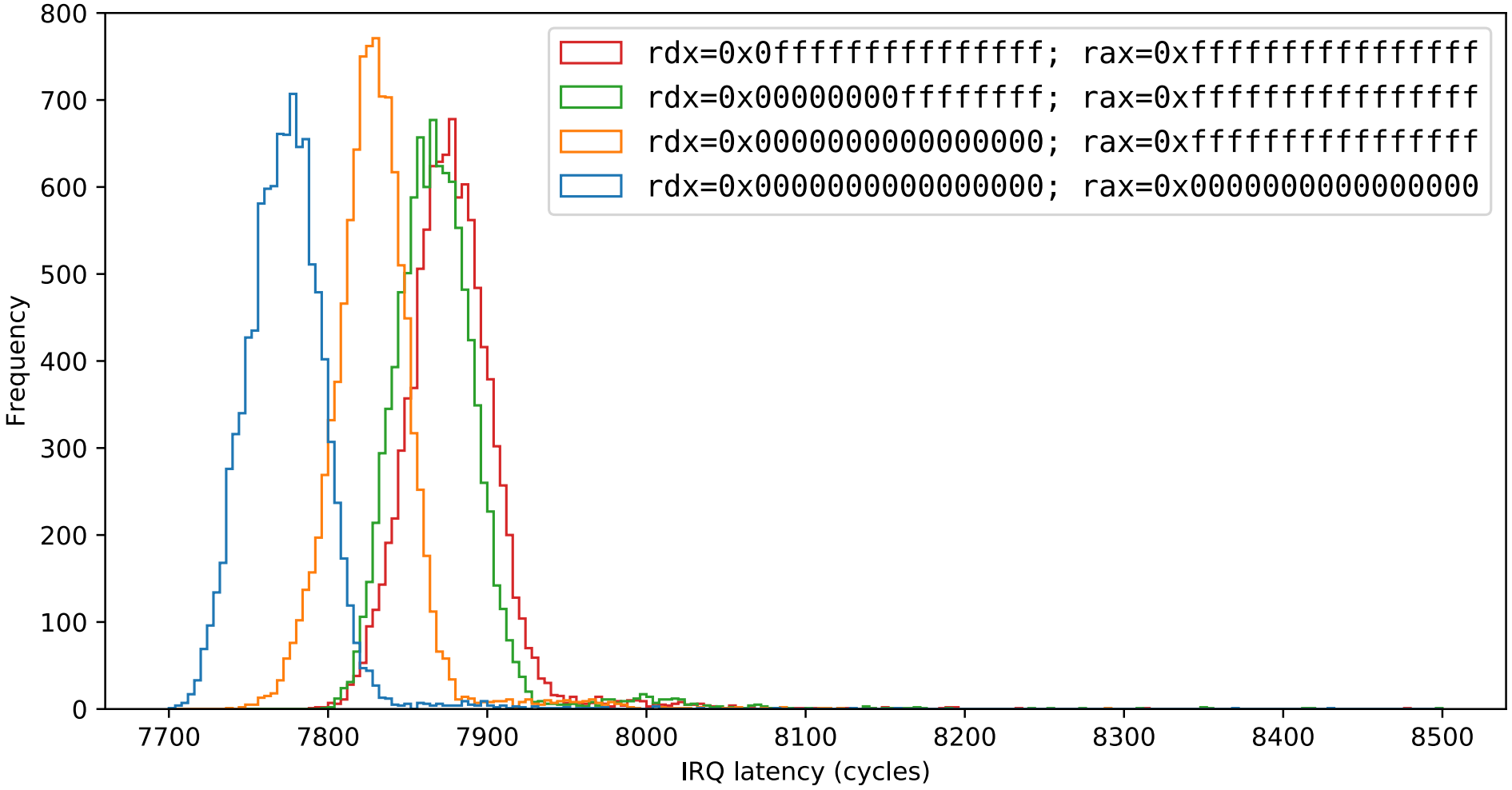


Appendix: Sancus keypad application scenario



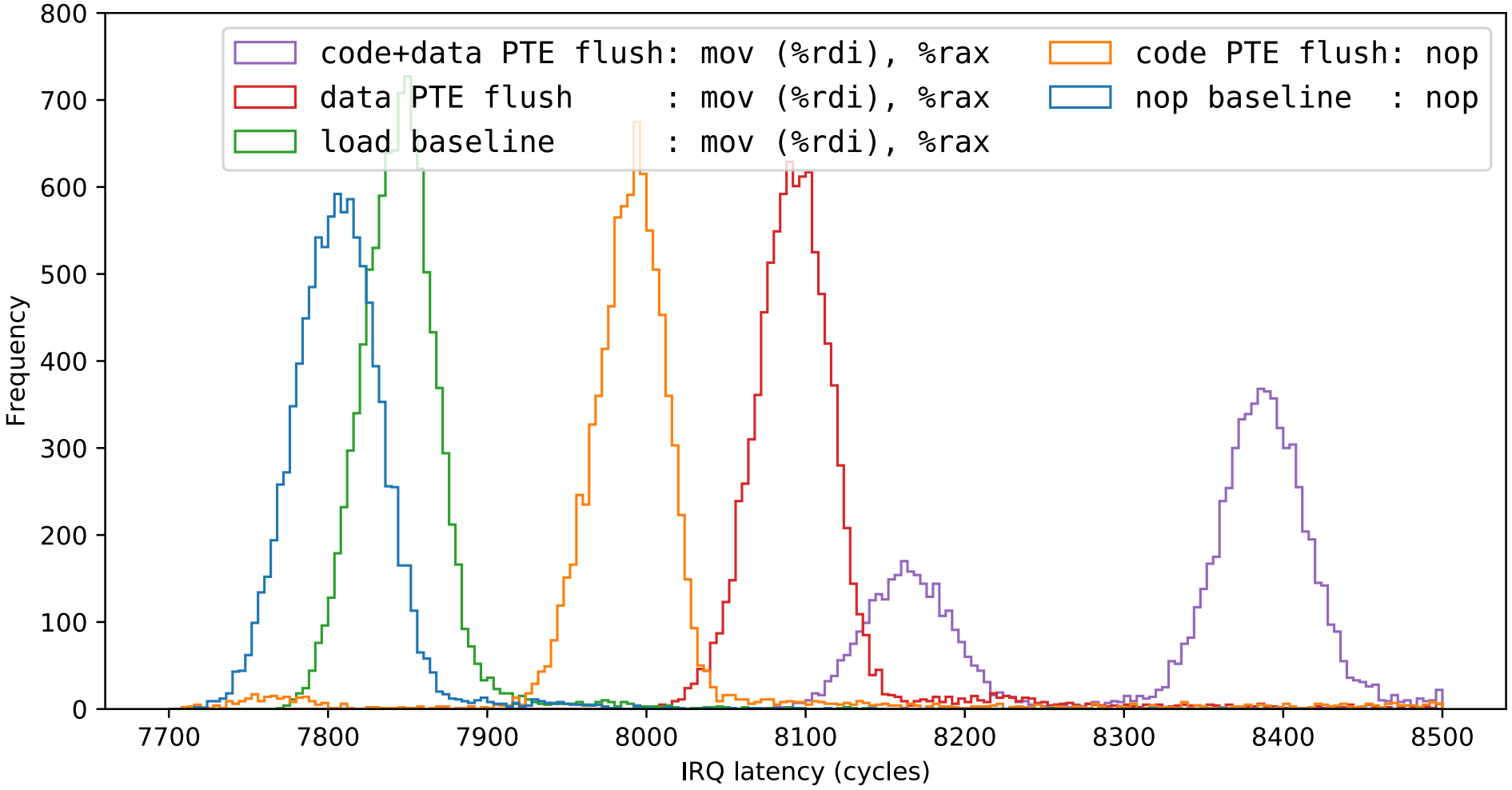
Appendix: Measuring x86 data dependencies

Division: execution time ≈ dividend significant bits

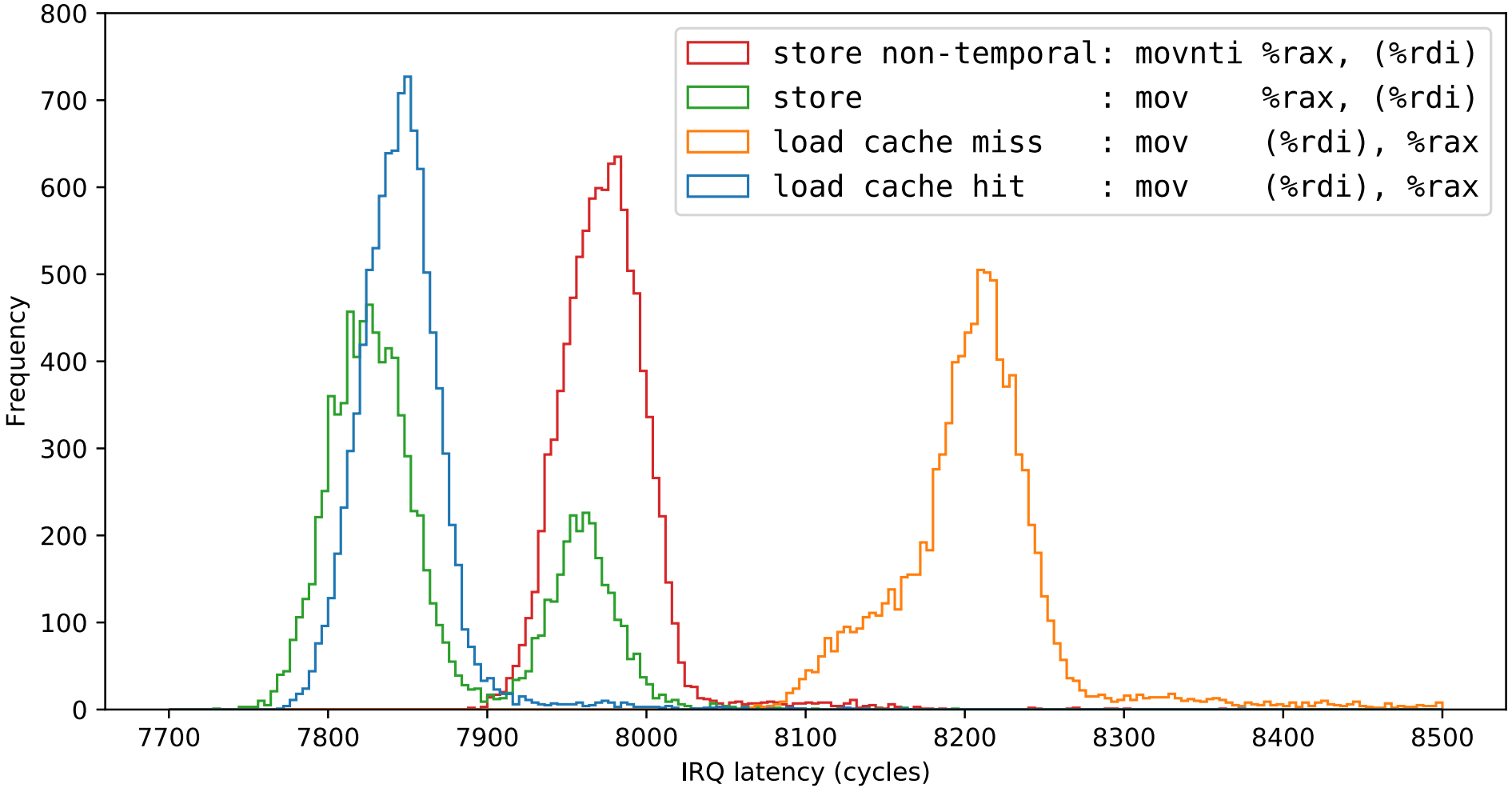


Appendix: Measuring x86 page table walks

TLB miss: flush *unprotected* page table entries

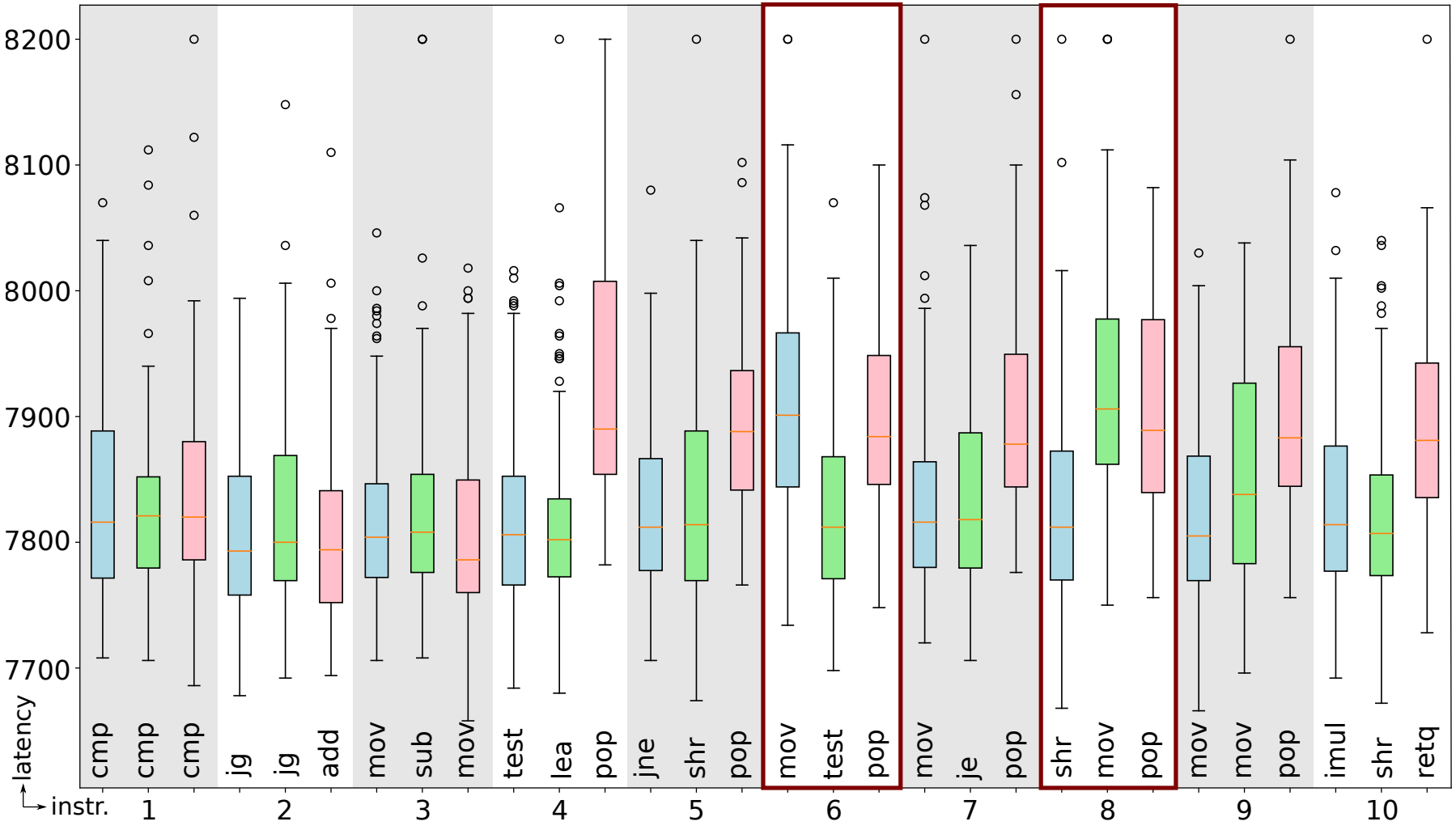


Appendix: Measuring x86 cache misses



Appendix: Boxplot binary search distribution

⇒ 100 `bsearch` runs: left (blue), right (green), hit (red)



Appendix: Boxplot Zigzagger distribution

⇒ 100 `zigzag` runs: branch taken (blue), not-taken (red)