UNIVERSITÄT ZU LÜBECK

Single-Stepping Attacks and Defences for Confidential VMs

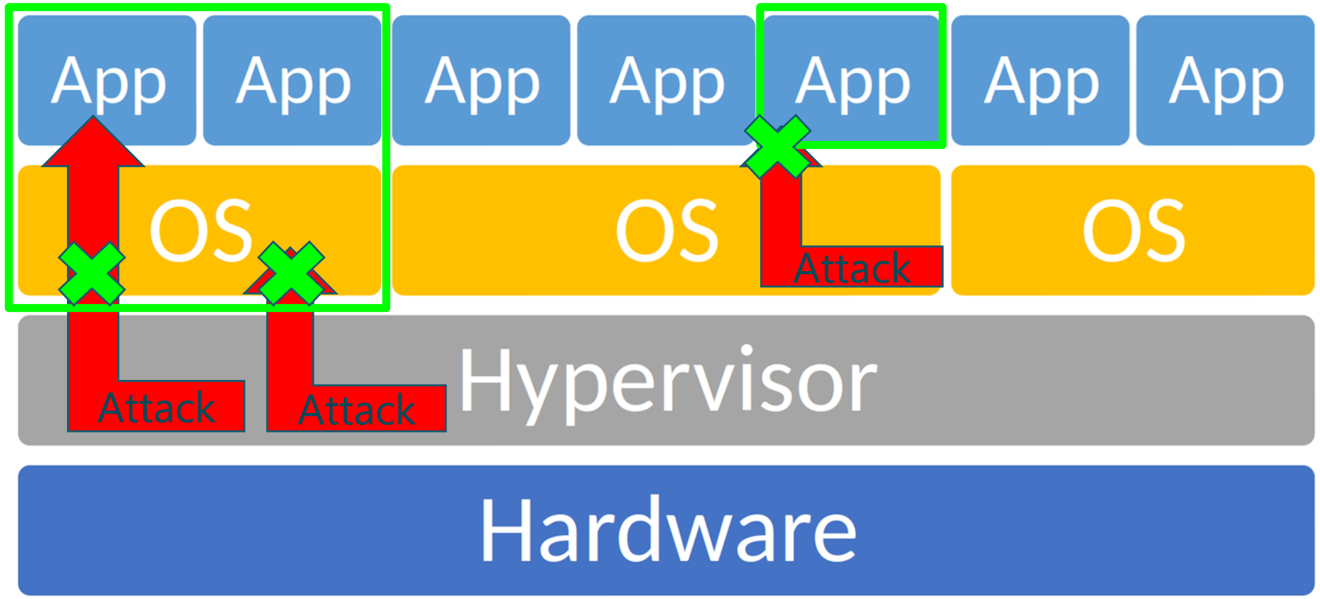
Luca Wilke
Institute for IT-Security
University of Lübeck

UNIVERSITÄT ZU LÜBECK

Trusted Execution Environments

Use Cases

1. Cloud computing
   ➞ remove hypervisor from TCB
2. Complex software stacks
   ➞ isolated compartments



Single-Stepping Attacks

Idea, Attacks on SGX, Attacks on SEV

Single-Stepping: Idea

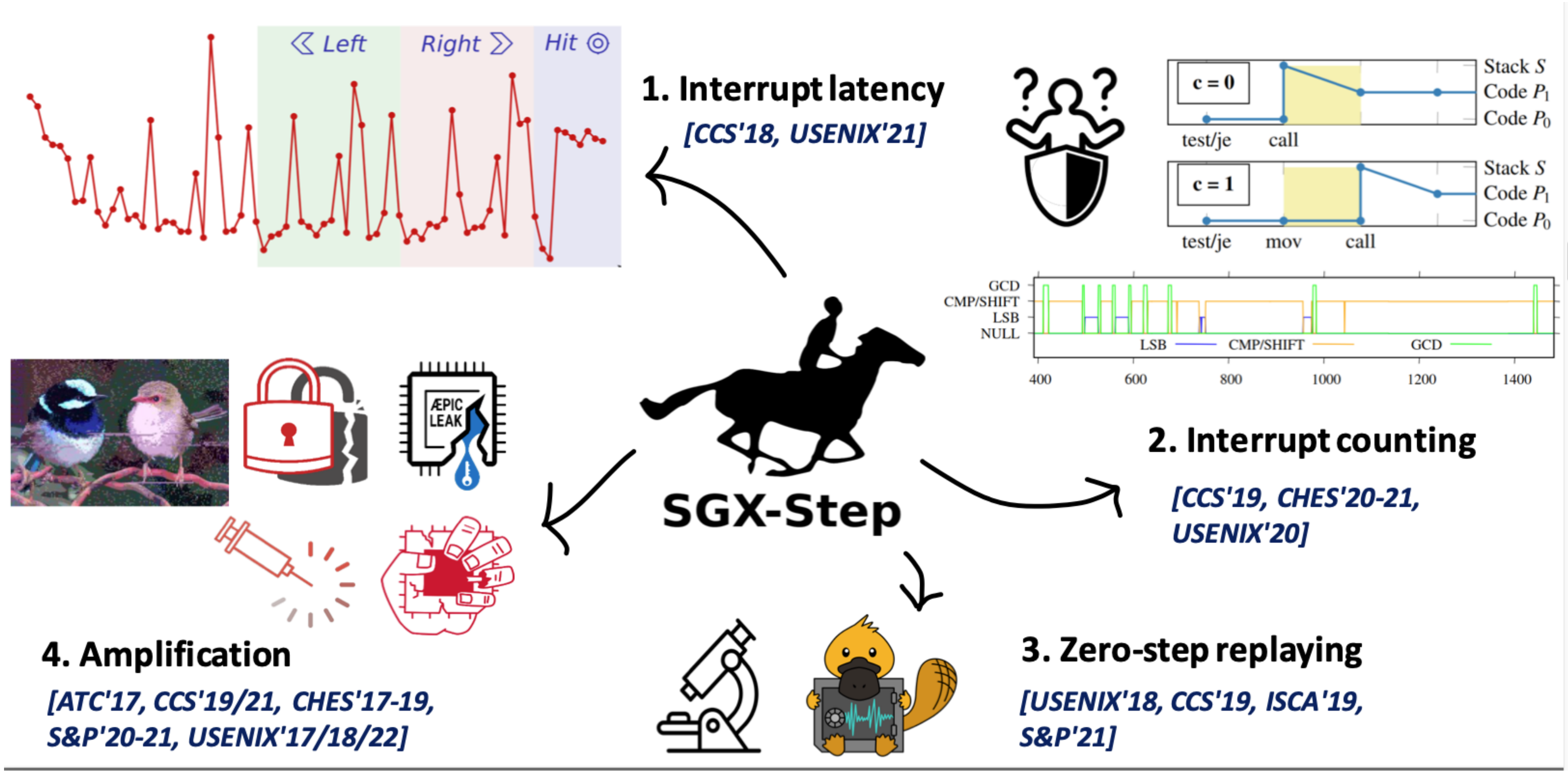
normal

frequently interrupted

single-stepped

UNIVERSITÄT ZU LÜBECK

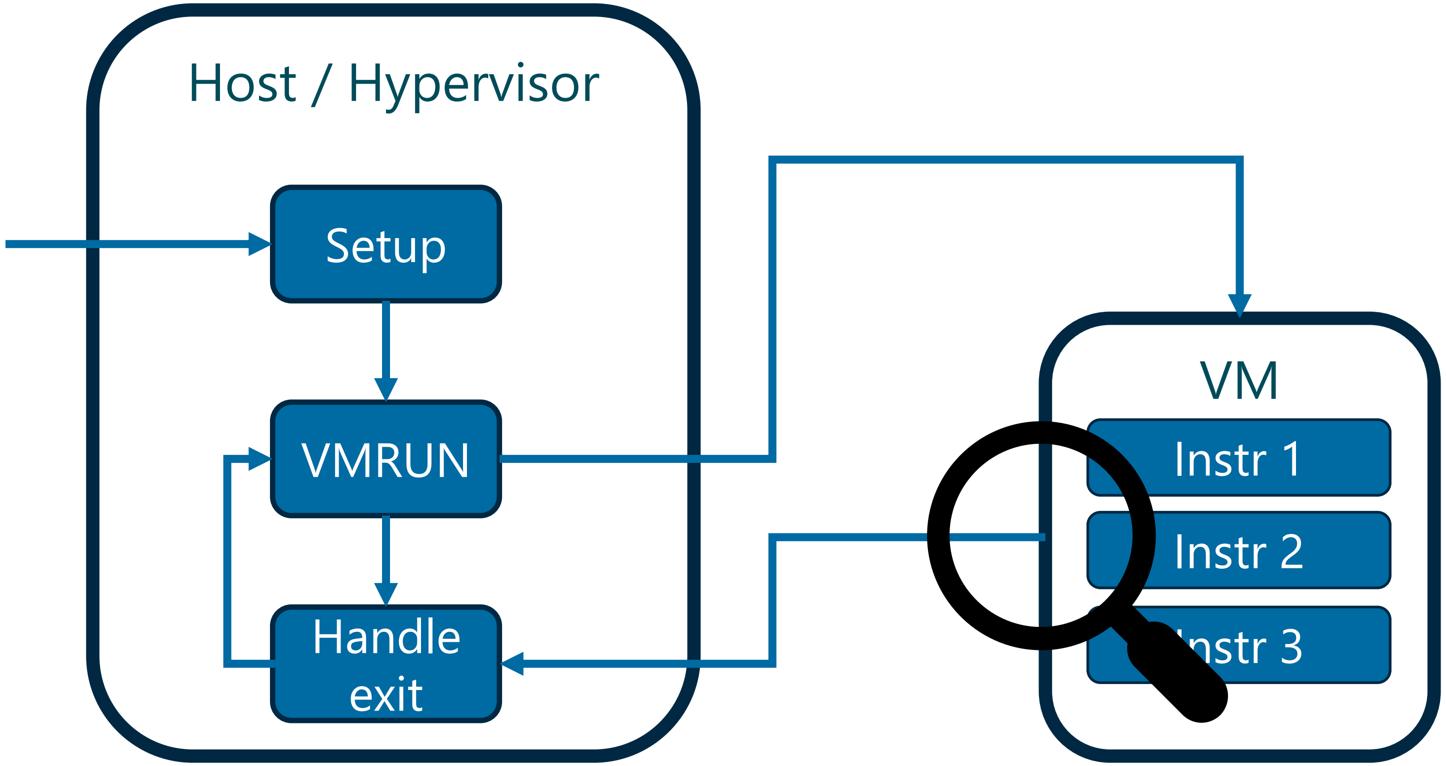
Single-Stepping: Attack Avenues



SGX-Step

Can SEV-SNP be Single-Stepped?

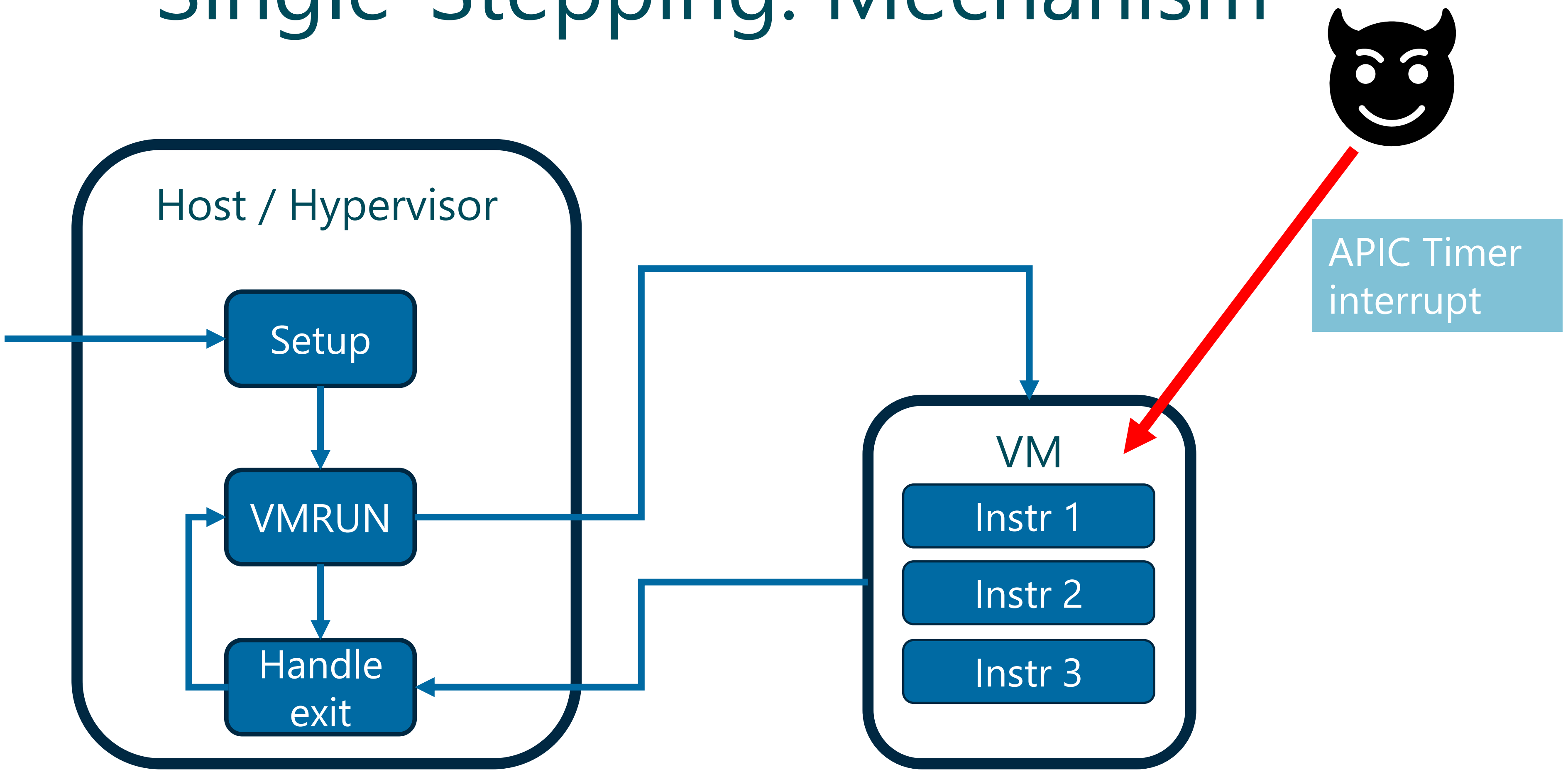
Single-Stepping: Mechanism



When do we exit?

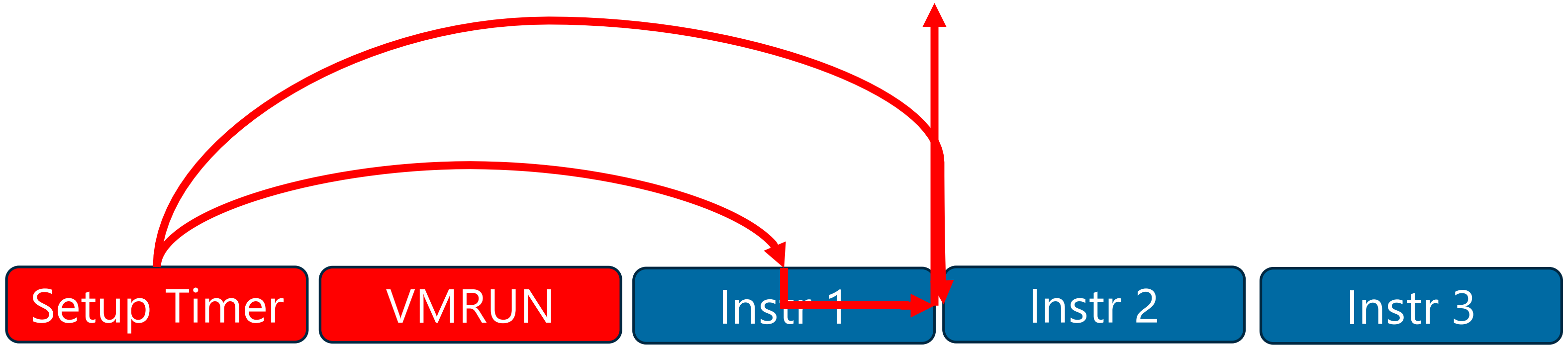
- Intercepted Instruction
- Page Fault
- External Interrupt

Single-Stepping: Mechanism

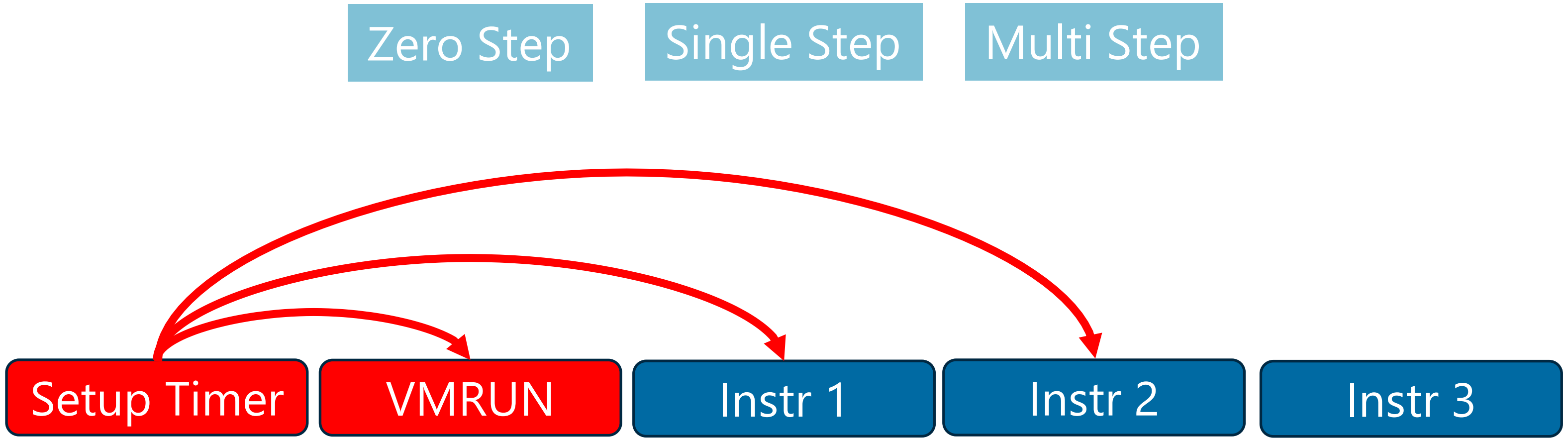


UNIVERSITÄT ZU LÜBECK

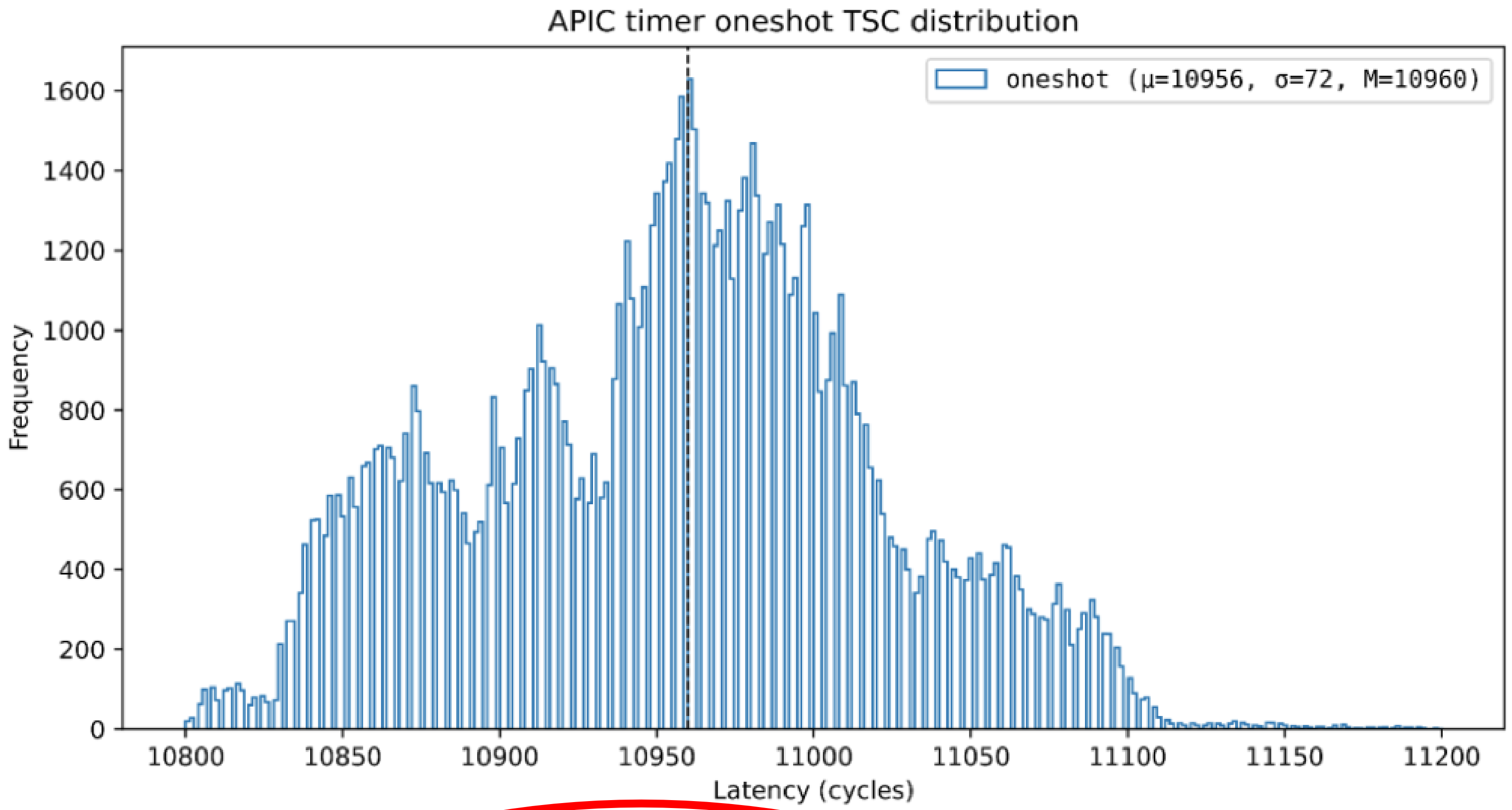
Single-Stepping: Root Cause



Single-Stepping: Naive Implementation



Single-Stepping: APIC Timing on SGX

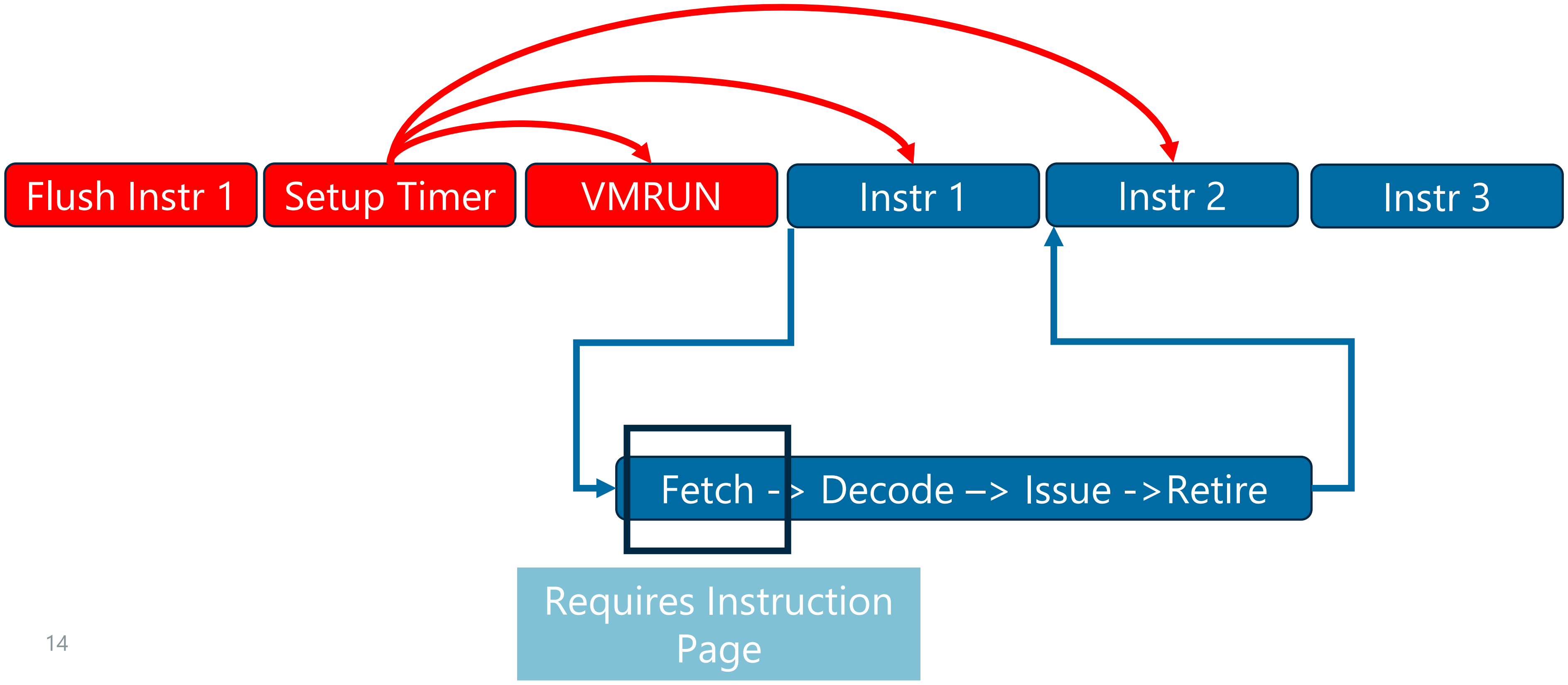


Measurements from AEX-Notify paper

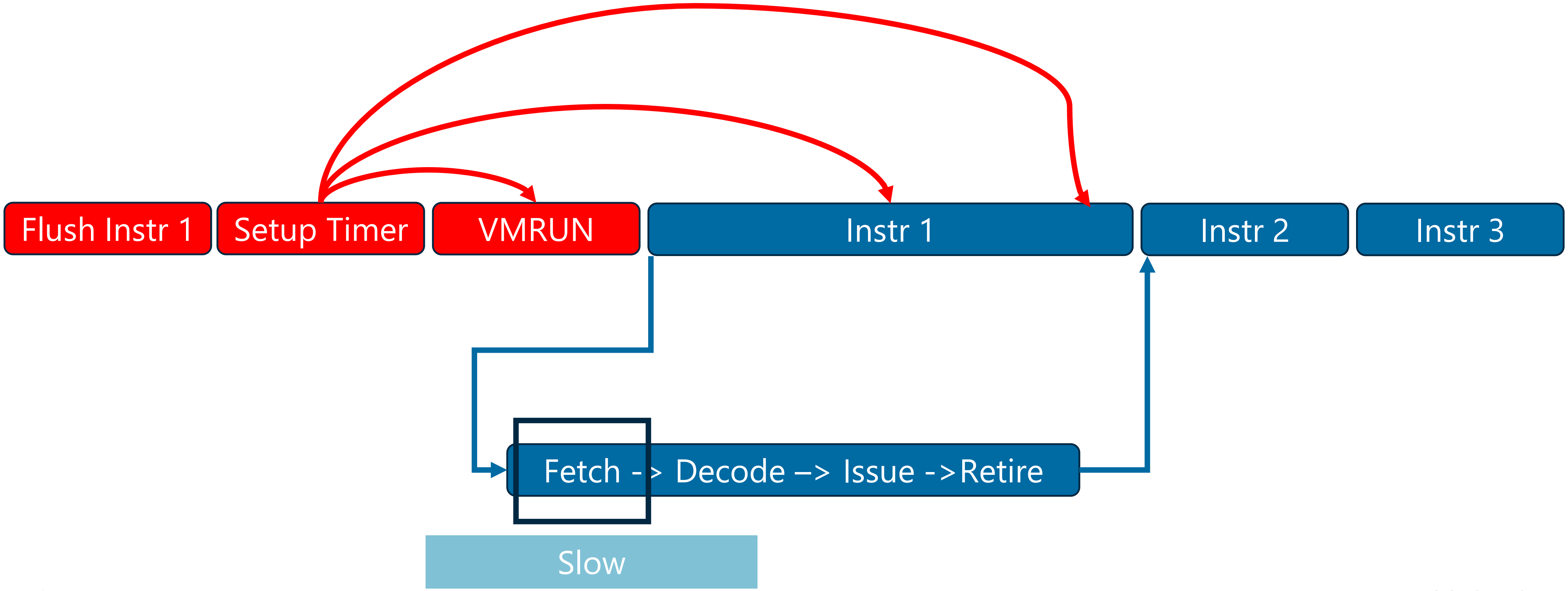
Setup Timer | VMRUN | Instr 1 | Instr 2 | Instr 3

Zero Step | Single Step | Multi Step

Single-Stepping: Enlarge the Window



Single-Stepping: Enlarge the Window



SEV-Step

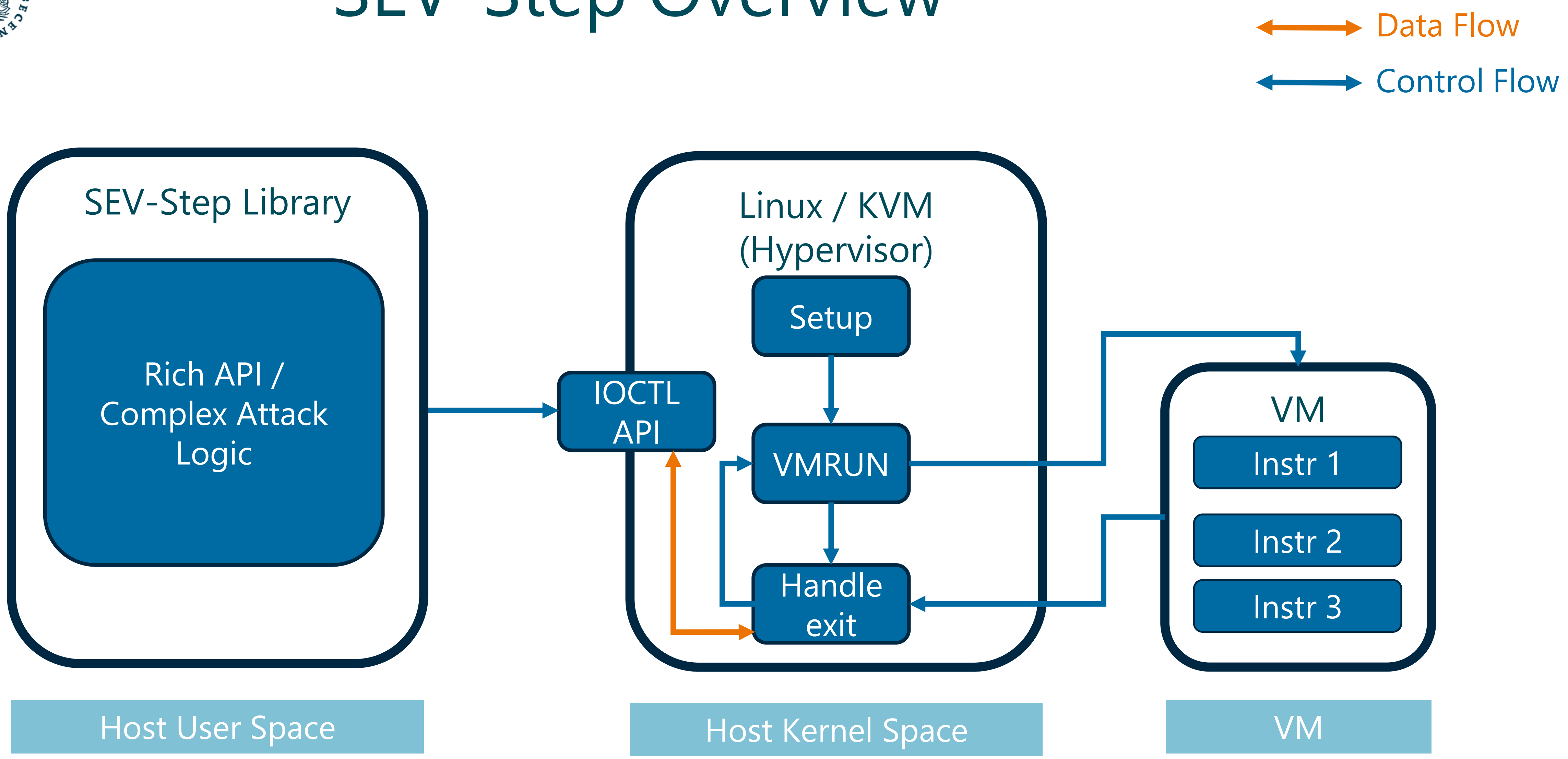
SEV-Step

A Single-Stepping Framework for AMD-SEV, CHES 2024

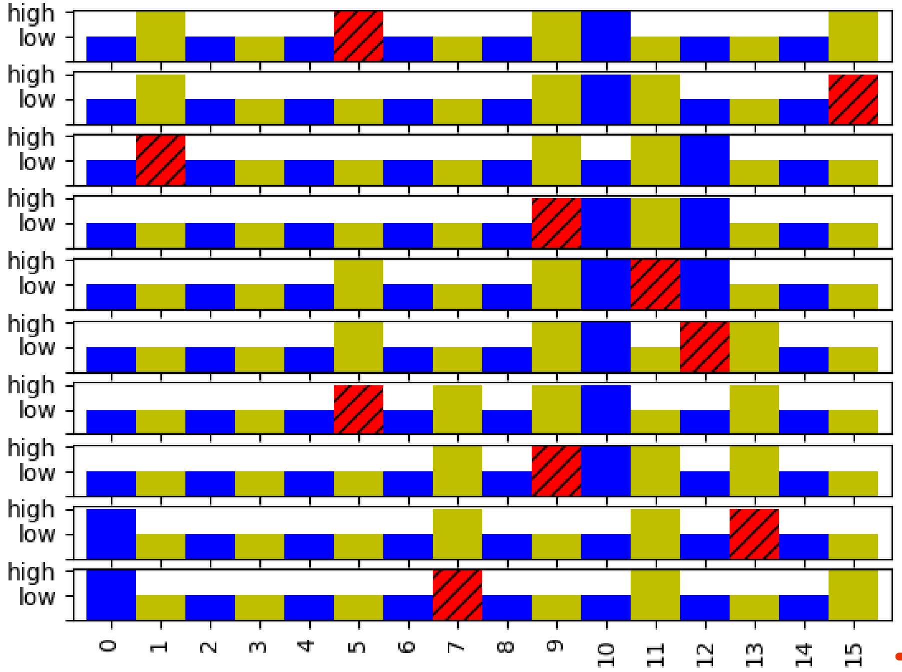
Luca Wilke, Jan Wichelmann, Anja Rabich and Thomas Eisenbarth

UNIVERSITÄT ZU LÜBECK

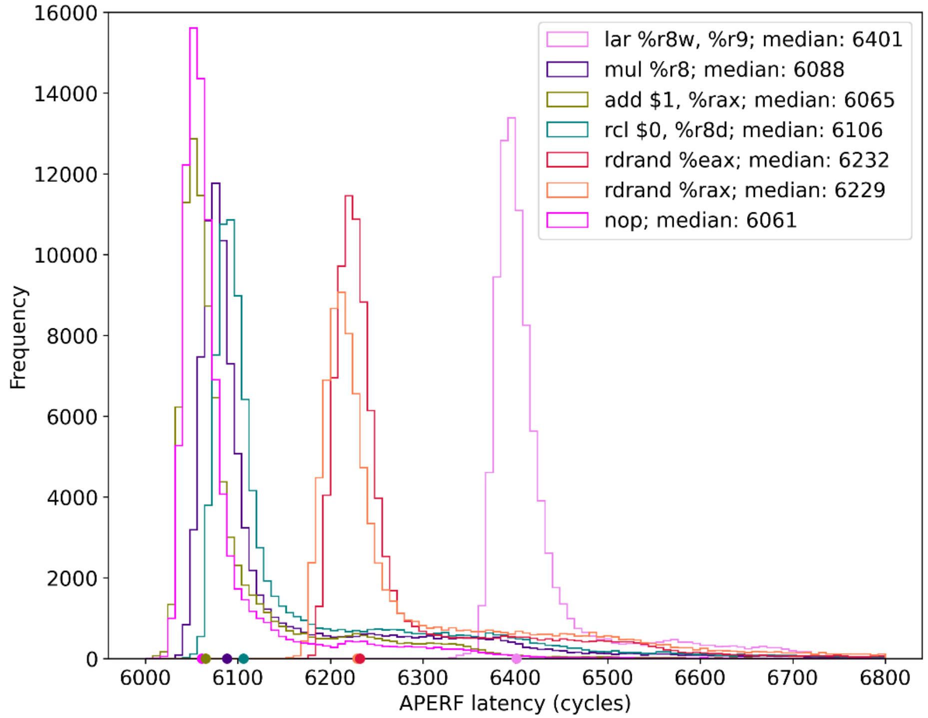
SEV-Step Overview



SEV-Step: Attacks



Cache Attacks



Interrupt Latency

(only used progress detection from SEV-Step)

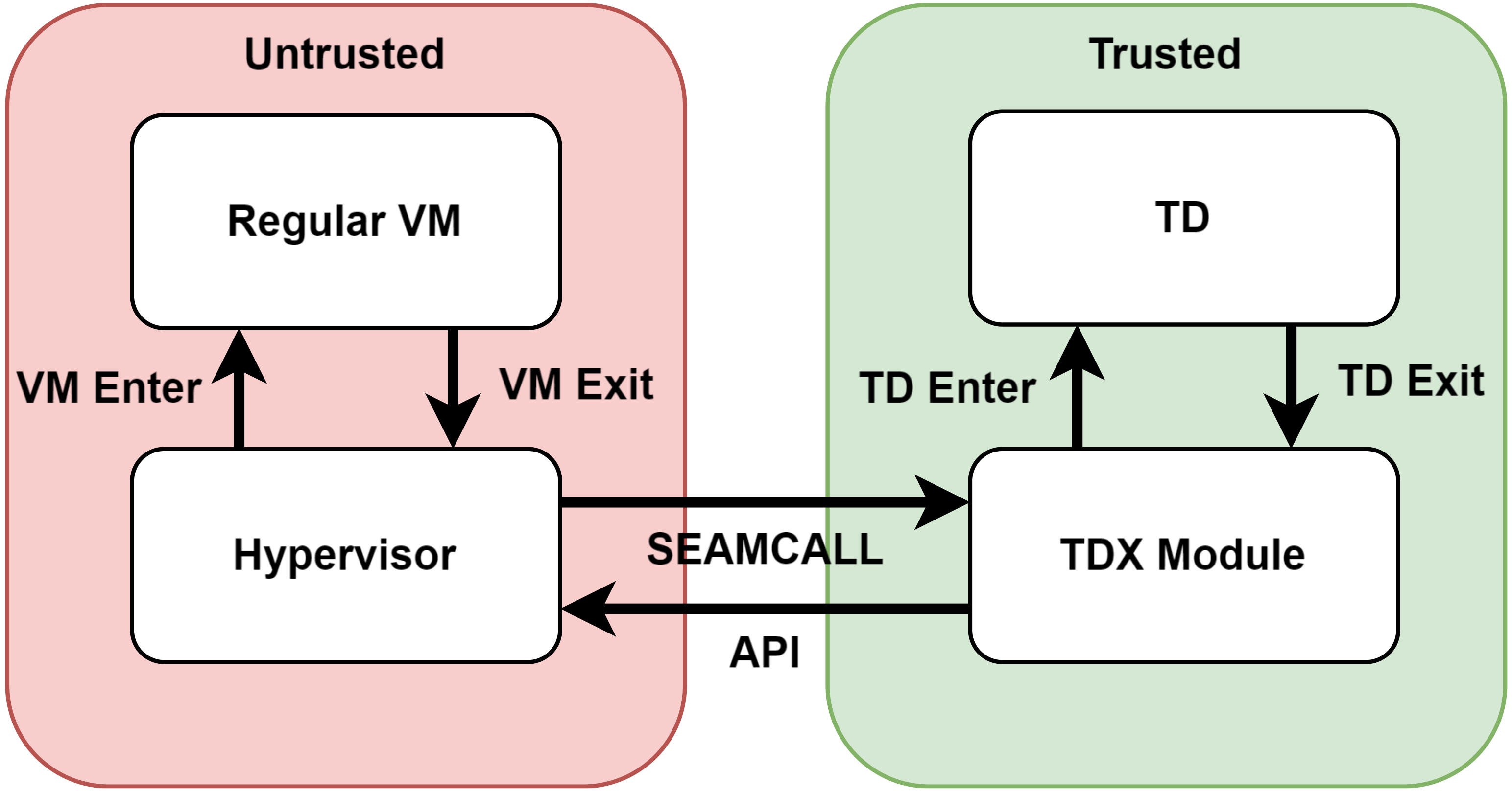
Countermeasures

Detection+Throttling, Prefetching, Application Rewriting

TDX: Detection + Throttling

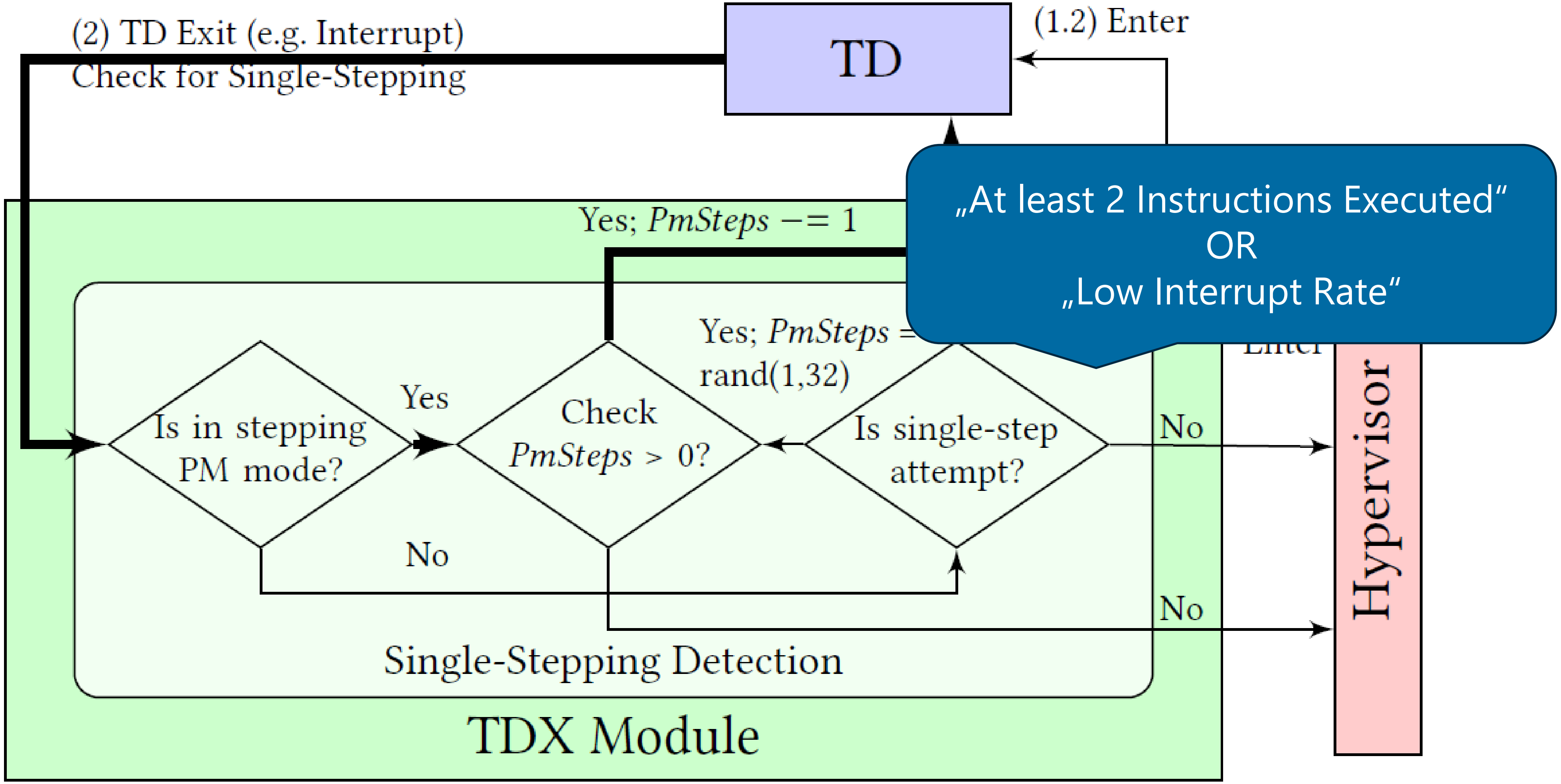
UNIVERSITÄT ZU LÜBECK

TDX: Architecture



UNIVERSITÄT ZU LÜBECK

TDX: Single-Stepping Countermeasure

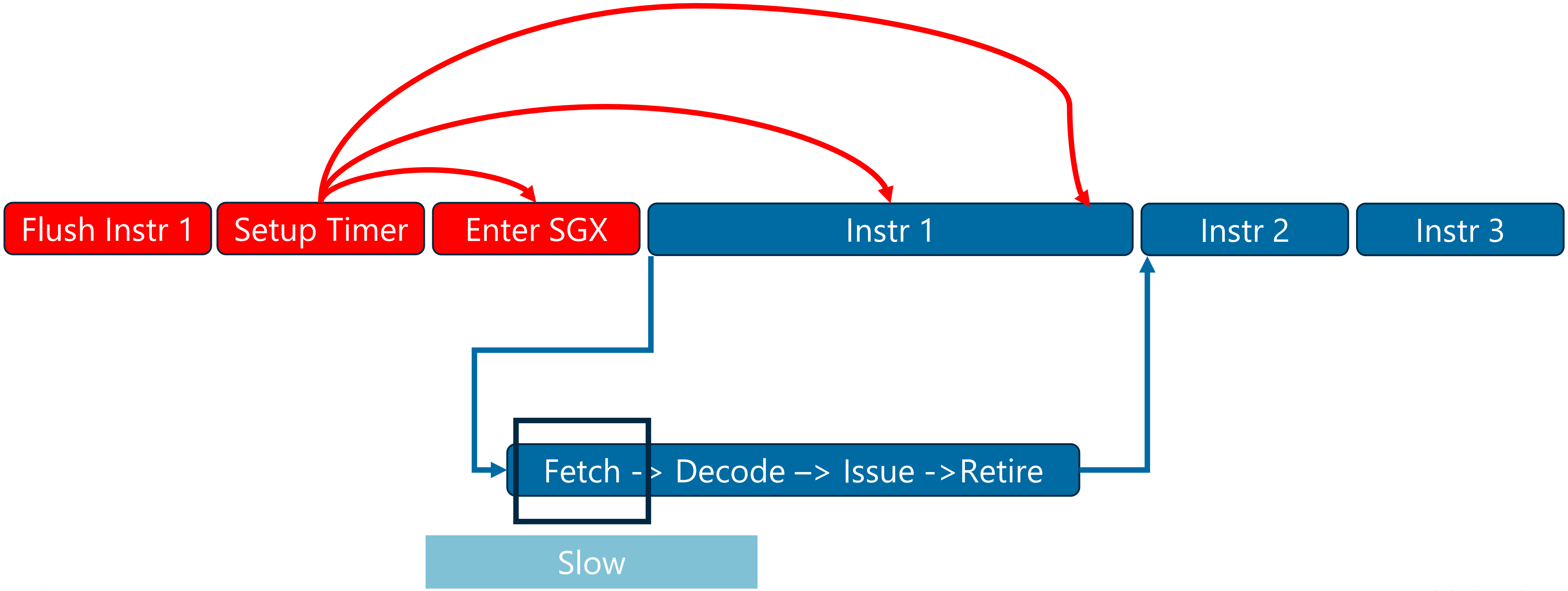


SGX: AEX-Notify

AEX-Notify, 32nd USENIX Security,
S. Constable, J. Van Bulck, X. Cheng, Y. Xiao, C. Xing, I. Alexandrovich, T. Kim, F. Piessens, M. Vij, M. Silberstein.

UNIVERSITÄT ZU LÜBECK

Recap Single-Stepping



AEX-Notify: Undo Slowdown

Flush Instr 1 | Setup Timer | Enter SGX | Countermeasure | Instr 1 | Instr 2 | Instr 3

Undo artificial slowdown by Prefetching Code + Data for Instr. 1

Too fast for reliable single-stepping

SEV-SNP and ARM CCA

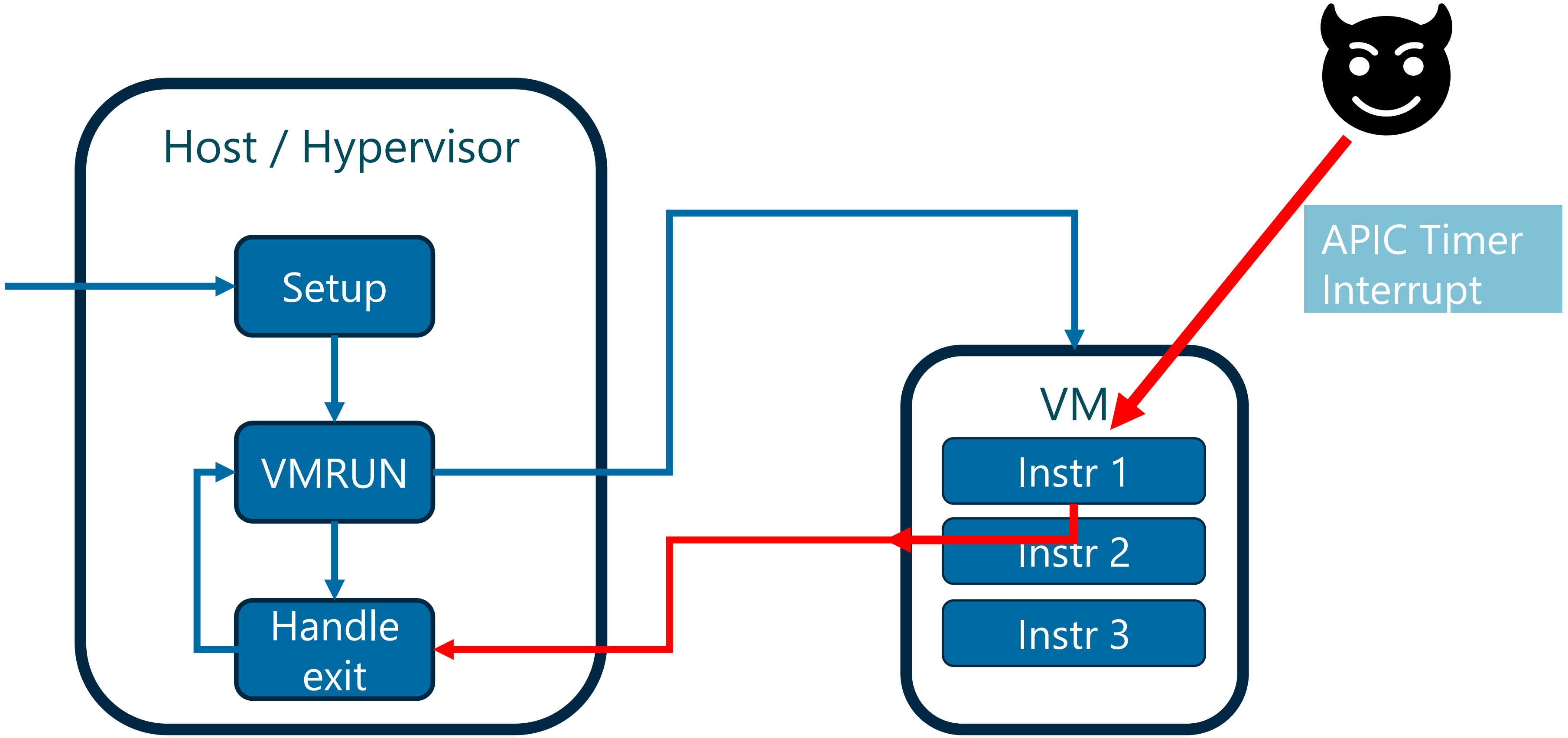
Applying AEX-Notify to CVMs

CVM Requirements for AEX-Notify

Make VM ext. Intr. aware

Deploy Countermeasure in VM

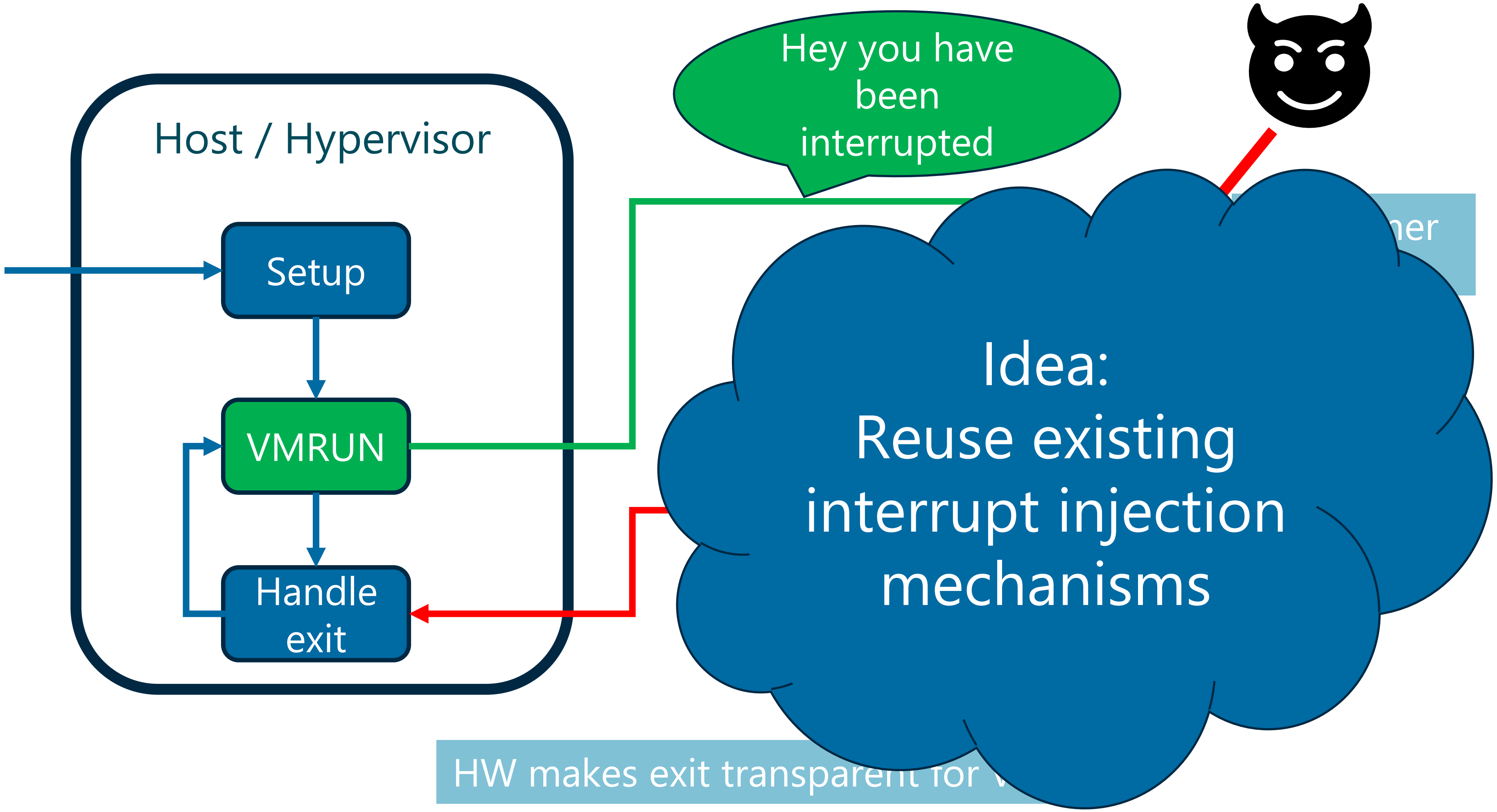
Make VM ext. Interrupt Aware



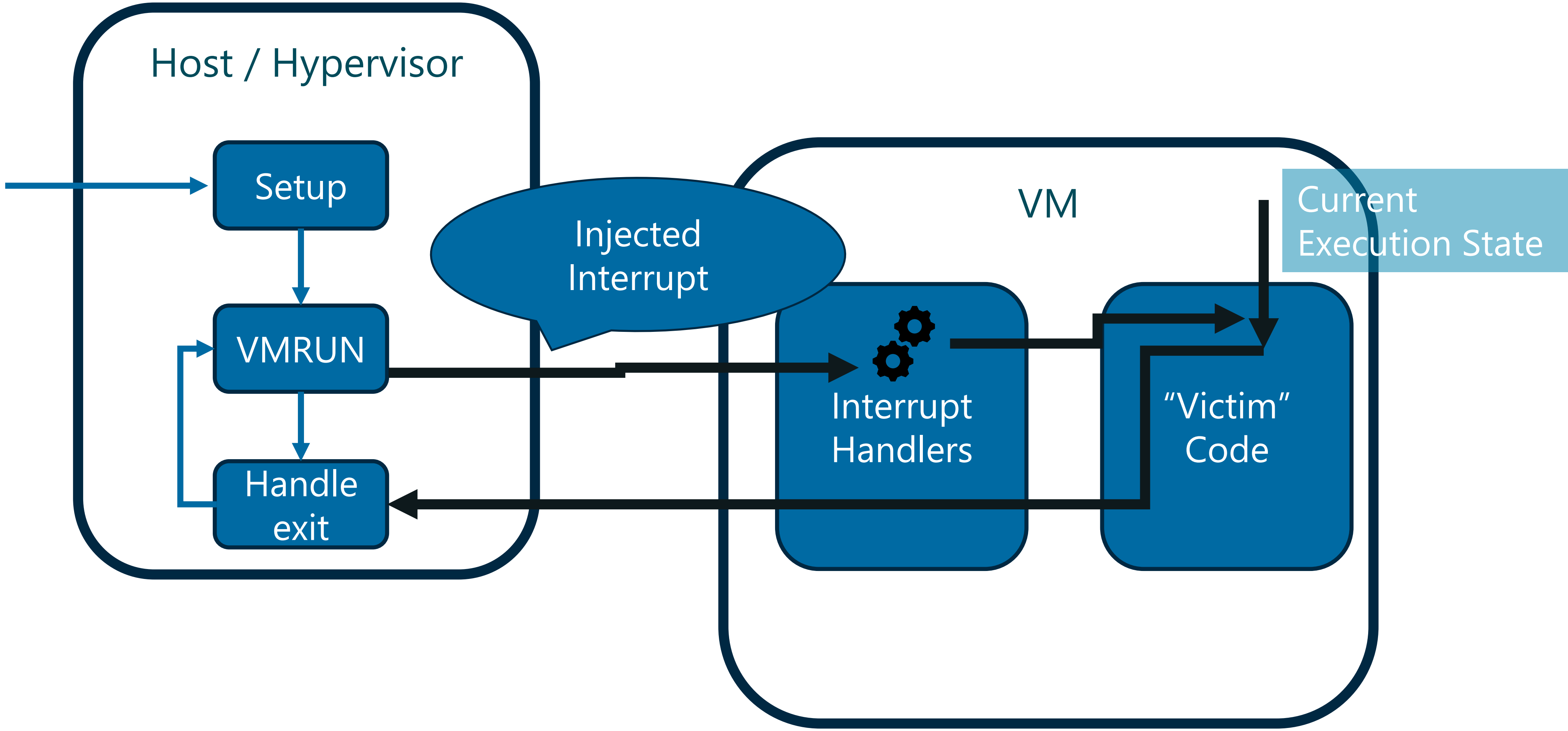
HW makes exit transparent for VM

UNIVERSITÄT ZU LÜBECK

Make VM ext. Interrupt Aware



What About Benign Injections?

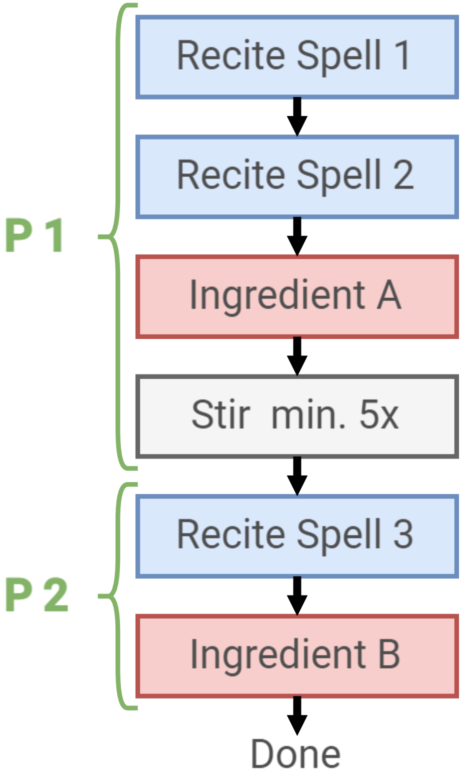


Generic Application Level Countermeaures

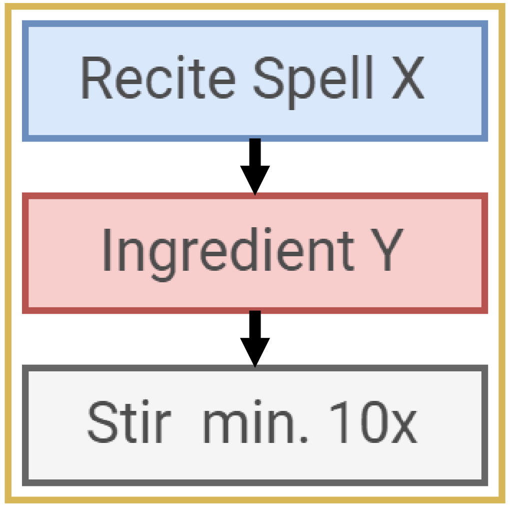
UNIVERSITÄT ZU LÜBECK

OBELIX: Generic App Level Countermeasure

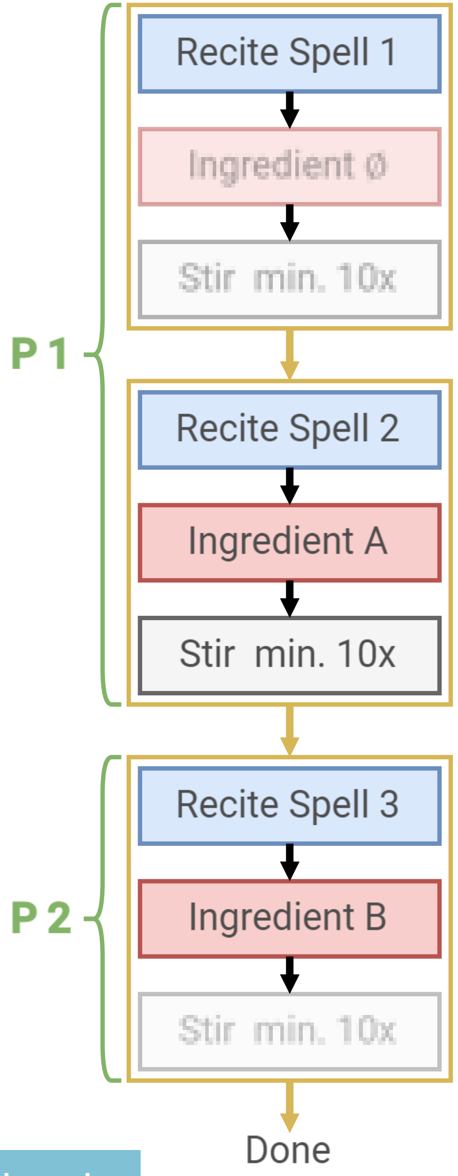
Strength Potion



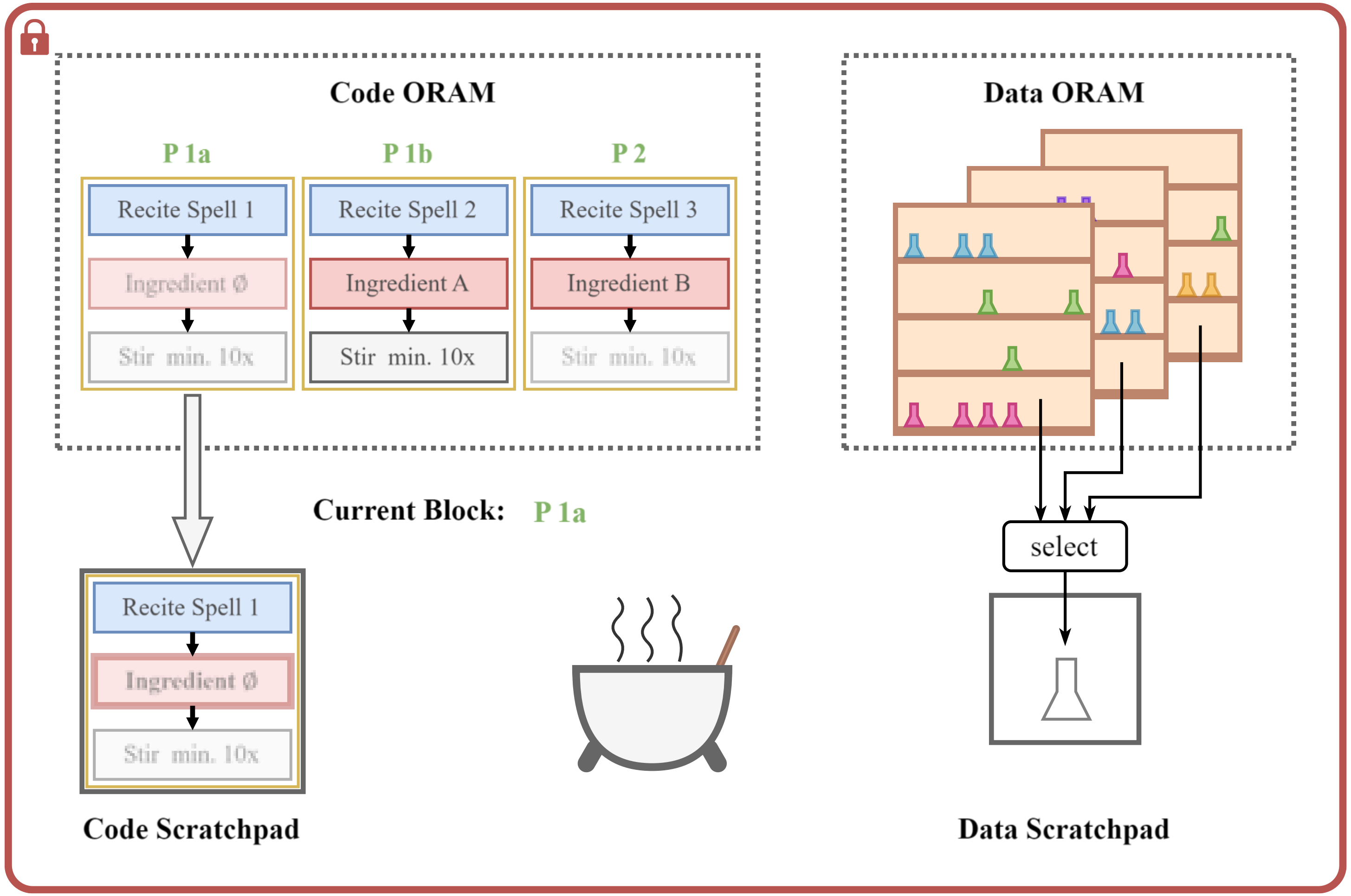
Uniform Code Pattern



Strength Potion



OBELIX: Generic App Level Countermeasure



OBELIX: Generic App Level Countermeasure

Pro

- Generic
- Eliminates
  - Cache attacks
  - Single-Stepping
  - Ciphertext-Side channels
  - Instruction latency attacks
  - …

Con

- High overhead

Summary

Summary

CVMs are vulnerable to single-stepping

SEV-SNP an ARM CCA do not have mitigations

TDX has mitigation, but hard to transfer to other CVMs

AEX-Notify could be applicable to CVMs ➞ generic mitigation?

App level countermeasures: high protection, high overhead