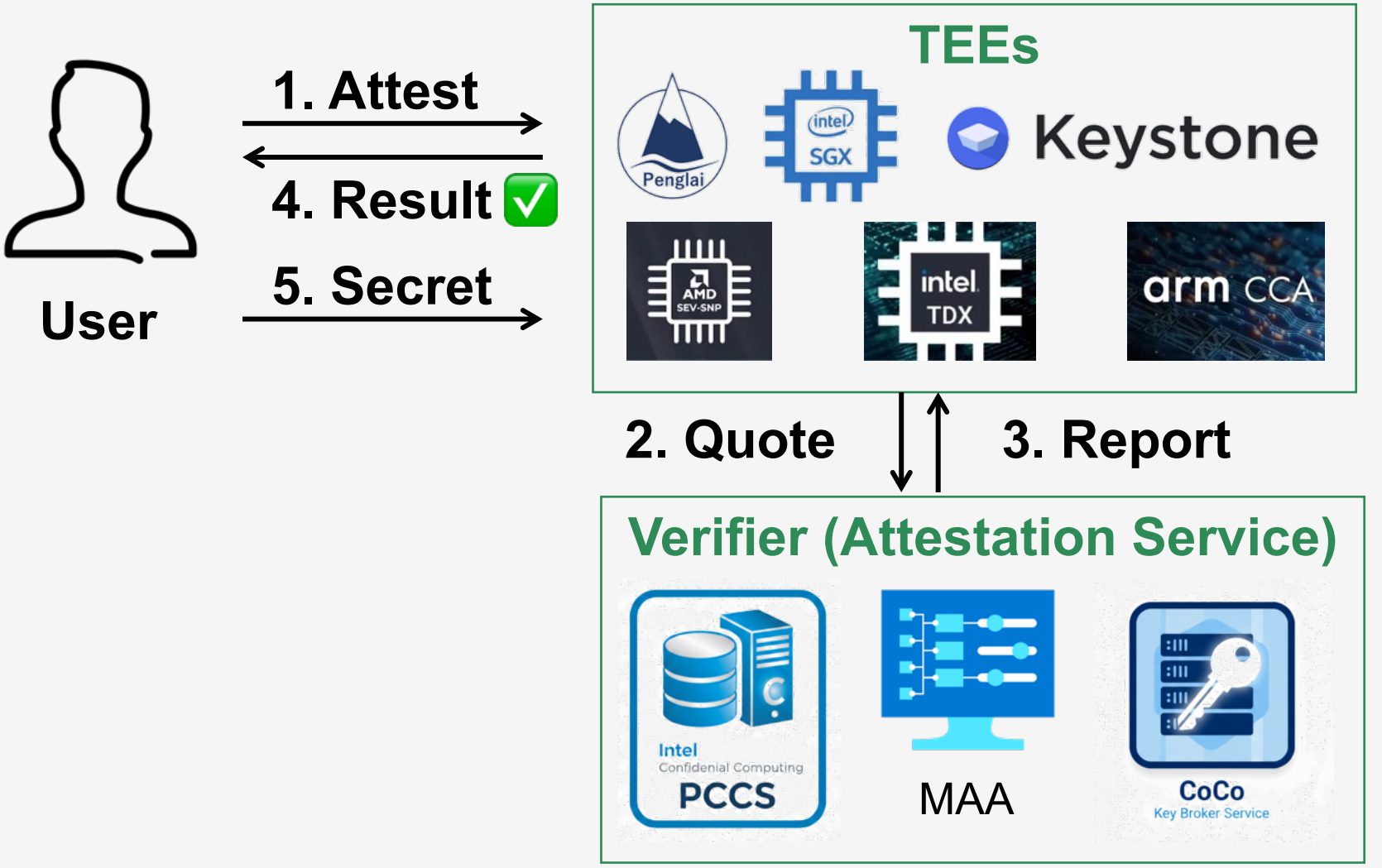
EXIA: Trusted Transitions for Enclaves via External-Input Attestation

Zhen Huang[1], Yidi Kao[2], Sanchuan Chen[2], Guoxing Chen[1], Yan Meng[1], Haojin Zhu[1]

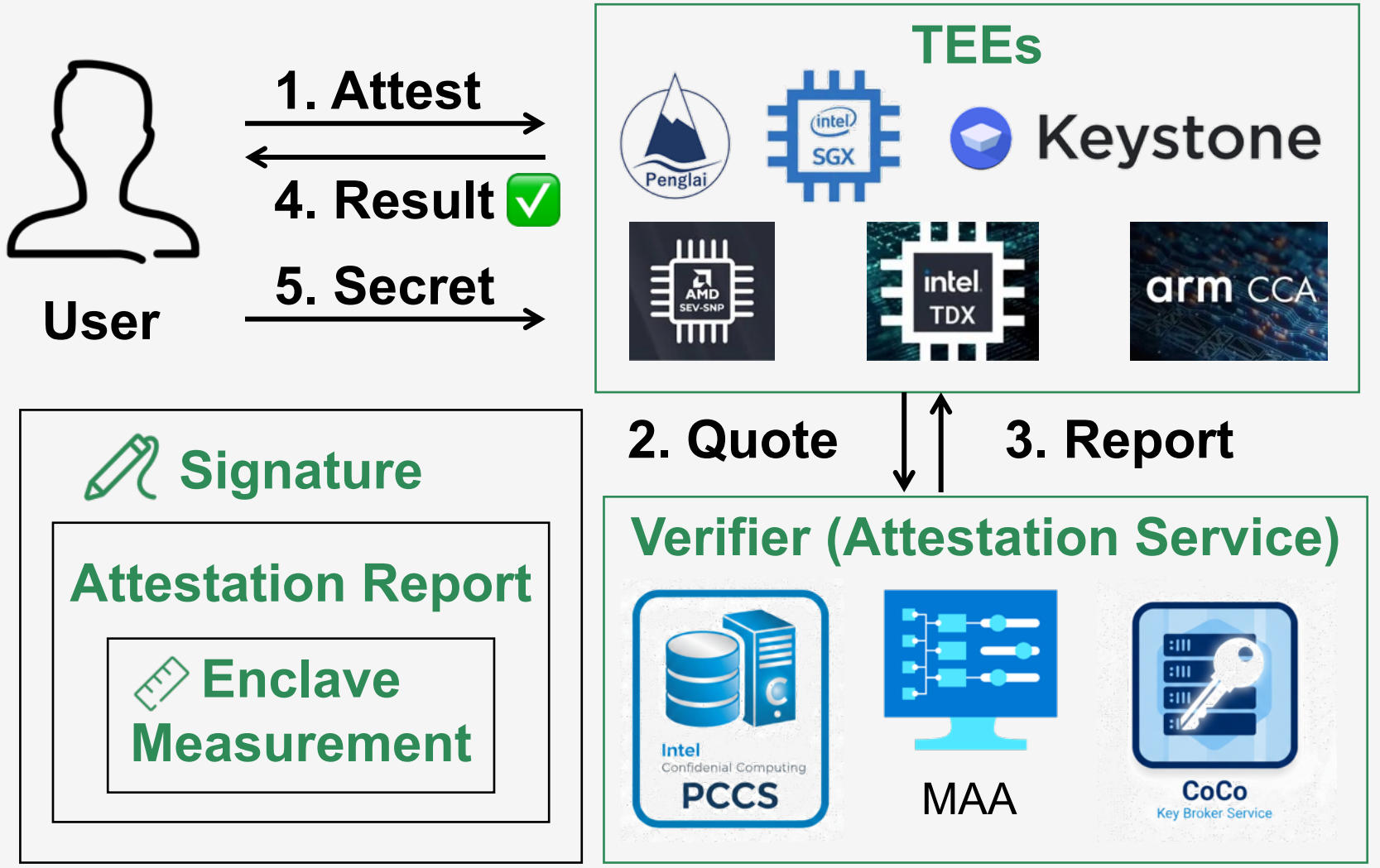
1. Shanghai Jiao Tong University
2. Auburn University

Trusted Execution Environment (TEE)

Remote Attestation



Remote Attestation



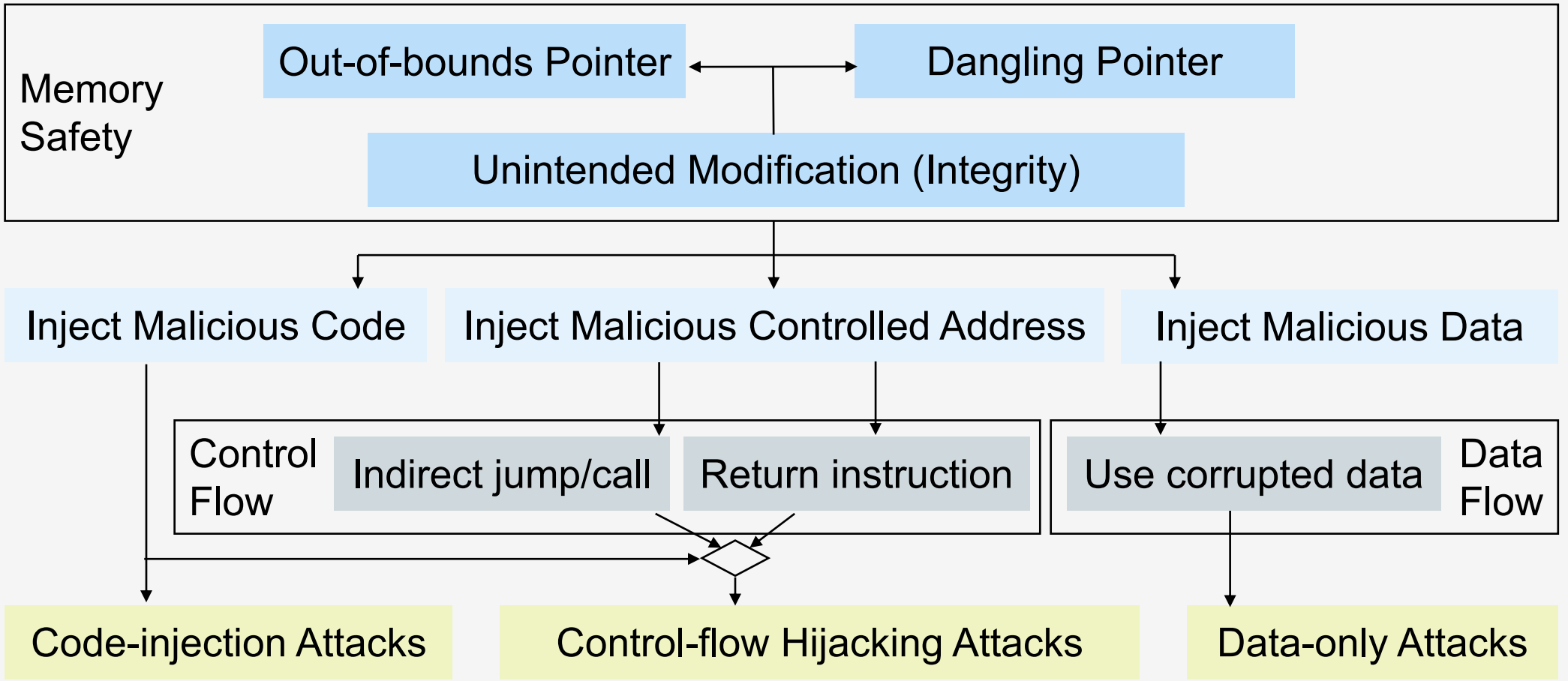
Launch-time Attestation is not Enough

Confidentiality	Side-Channel	Spectre, Cache Attacks, …	SgxPectre, Heracles, SCASE, CounterSEVeillance, …
Integrity	Memory Corruption	Buffer Overflow, ROP, NULL Pointer Dereference, …	TeeRex, EnclaveFuzz, …
	Hardware Fault	Voltage, Rowhammer, …	Plundervolt, VoltPillager, TEE.Fail, Battering RAM, …
	Iago	Ahoi, …	WeSee, Heckler, Sigy, …
Availability	DoS	Resource Exhaustion	

Launch-time Attestation is not Enough

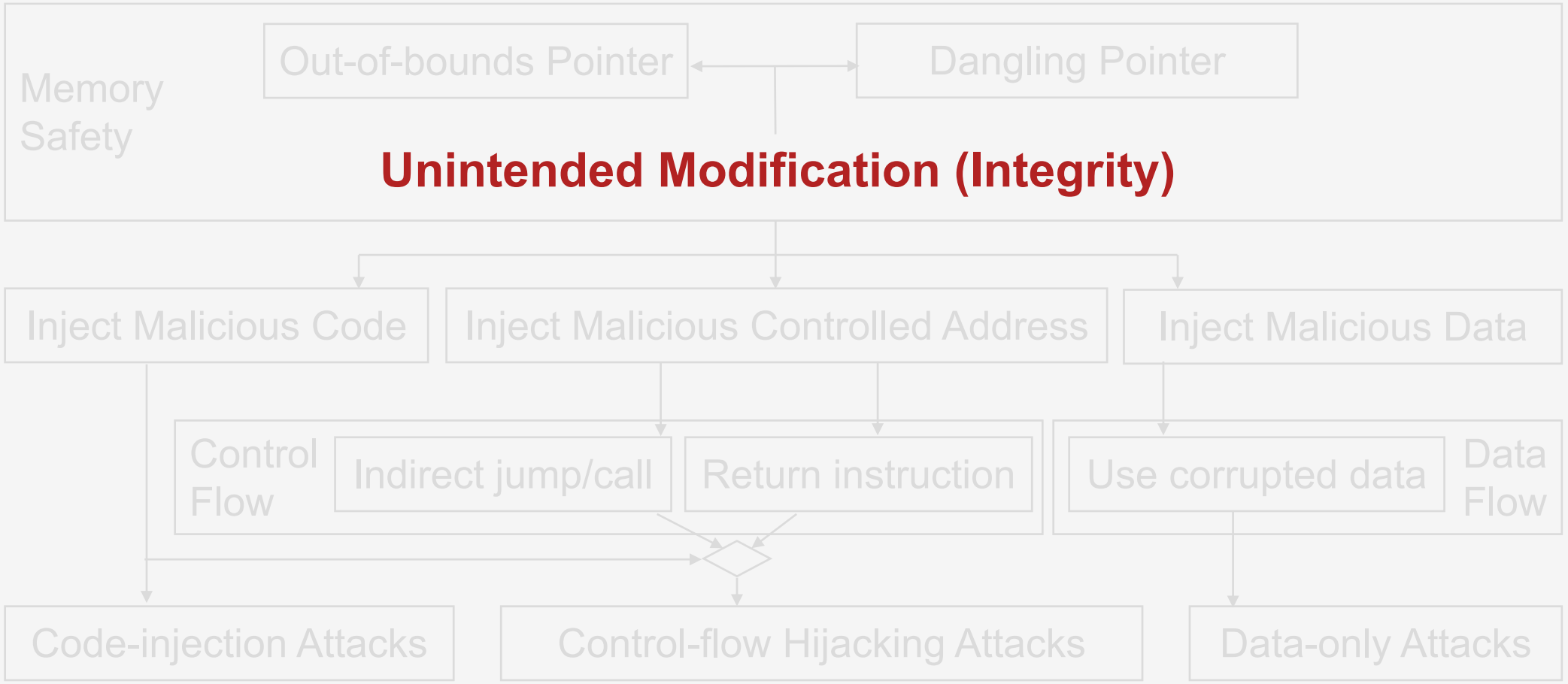
Confidentiality	Side-Channel	Spectre, Cache Attacks, …	SgxPectre, Heracles, SCASE, CounterSEVeillance, …
	Memory Corruption	**Buffer Overflow, ROP, NULL Pointer Dereference, …**	**TeeRex, EnclaveFuzz, …**
Integrity	Hardware Fault	Voltage, Rowhammer, …	Plundervolt, VoltPillager, TEE.Fail, Battering RAM, …
	Iago	Ahoi, …	WeSee, Heckler, Sigy, …
Availability	DoS	Resource Exhaustion	

Memory Corruption Attacks



Ammar, Mahmoud, et al. "SoK: Integrity, attestation, and auditing of program execution." *2025 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2025.

Memory Corruption Attacks

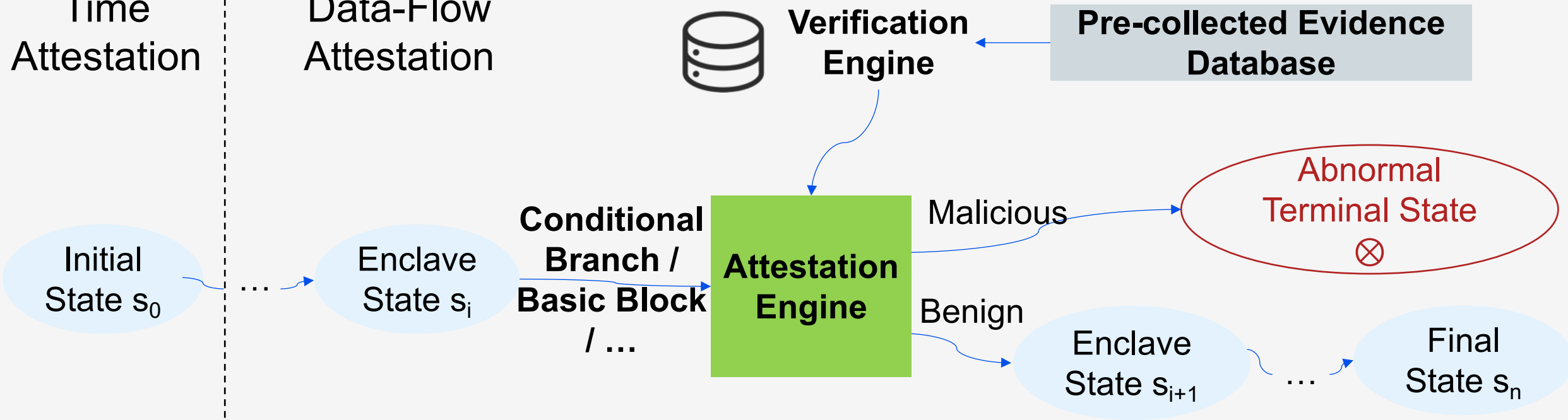


Ammar, Mahmoud, et al. "SoK: Integrity, attestation, and auditing of program execution." *2025 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2025.

Related Work

Launch-Time Attestation

Control-Flow / Data-Flow Attestation



EXIA: External Input Attestation

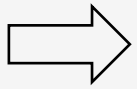
C1: **Evidence Integrity**

C2: **Verification Efficiency**

EXIA: External Input Attestation

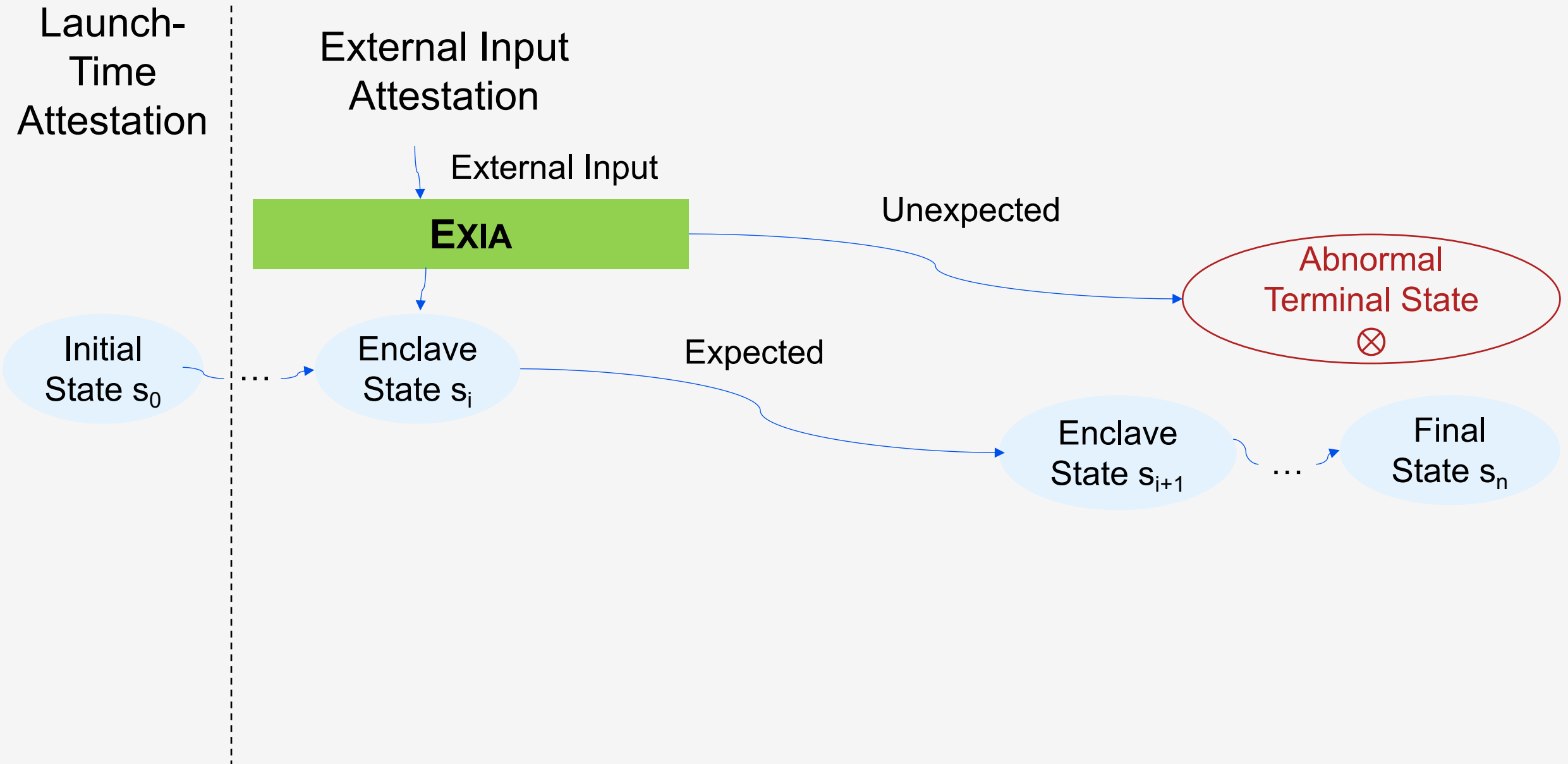
C1: **Evidence Integrity**

C2: **Verification Efficiency**

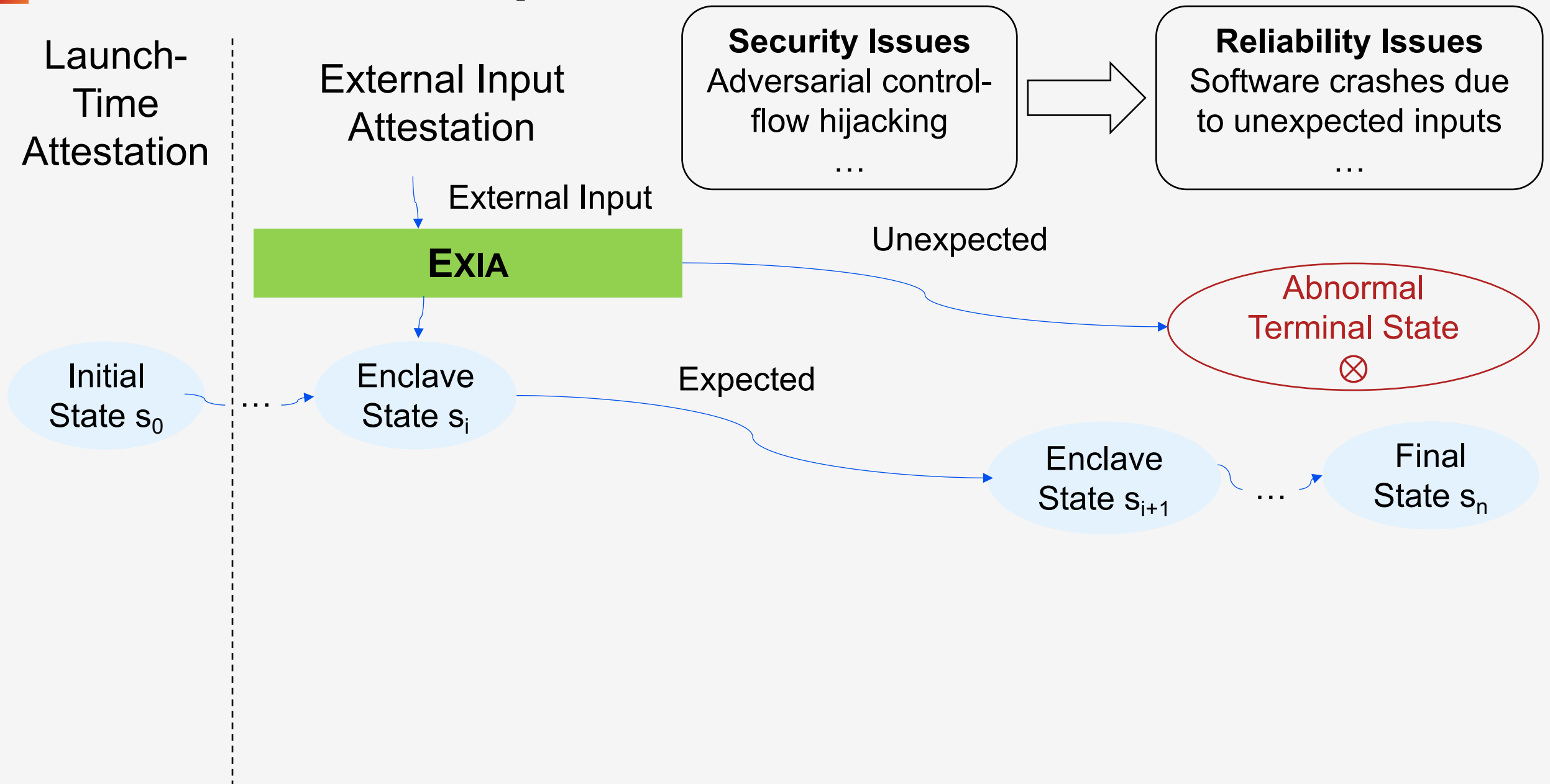


Can we design a framework for attesting the integrity of execution traces with only user-side knowledge?

EXIA: External Input Attestation



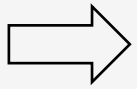
EXIA: External Input Attestation



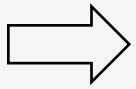
EXIA: External Input Attestation

C1: **Evidence Integrity**

C2: **Verification Efficiency**



Can we design a framework for attesting the integrity of execution traces with only user-side knowledge?



Privileged Attesting Environment

Trusted Input Gateway

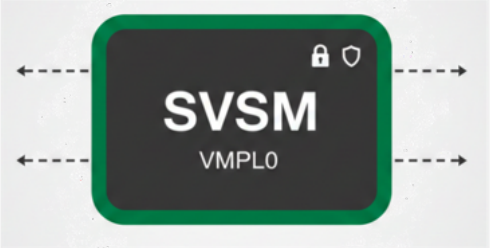
Trusted Interrupt Handling

Privileged Attesting Environment

Inv. 1-1: be **isolated and protected** from the enclaved application.

Inv. 1-2: **measure inputs first.**

- Off-the-shelf Privilege Leveling



- Software-based Isolation
  - PALANT´IR (NDSS 25), …
- Architectural Extension RISC-V

Trusted Input Gateway

Inv. 2-1: the enclaved application **does not access addresses directly** shared with the host program.

- Dual-Page Transfer

- Shared-Page Permission Flipping

Inv. 2-2: defend **replay attack**.

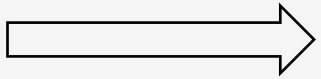
- Freshness Verification

Trusted Interrupt Handling

Transparent Exit Events (TEx)

Non-Transparent Exit Events (NTEx)

w/o modify enclave content



❌ measure

Trusted Interrupt Handling

Transparent Exit Events (TEx)

Non-Transparent Exit Events (NTEx)

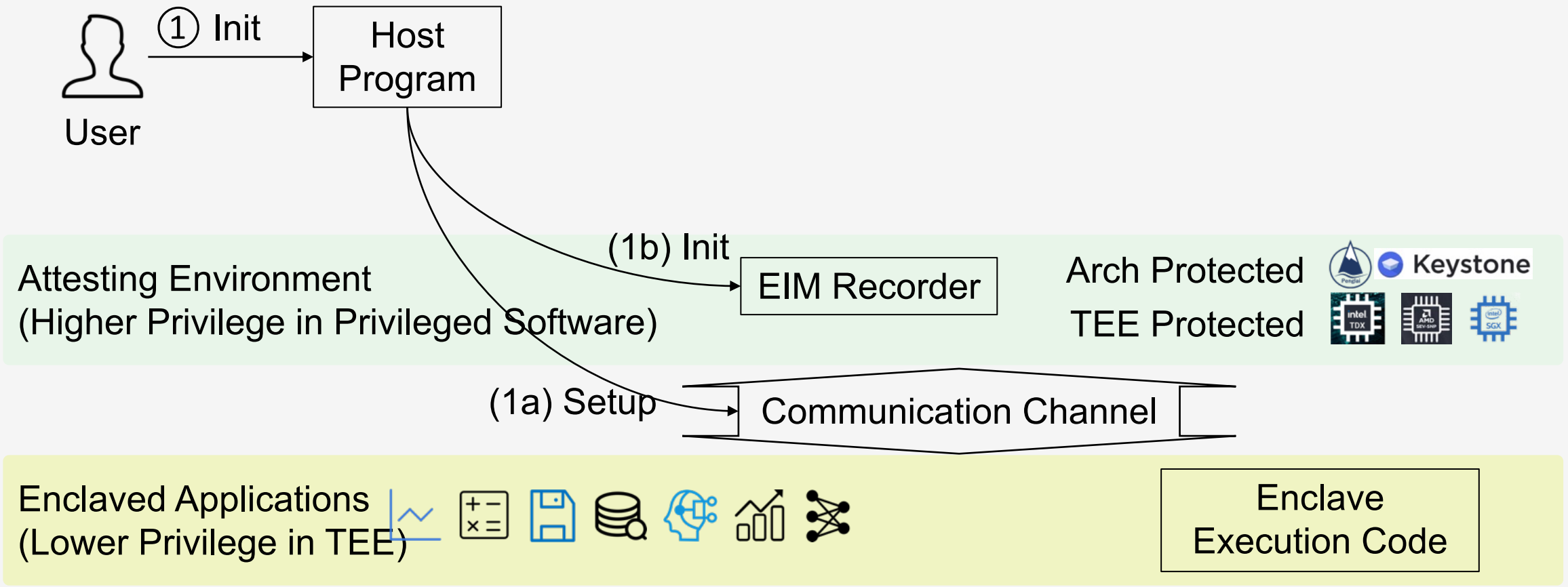
w/o host inputs

Inv. 3-1: the **authenticity** of the interrupts needs to be checked in the customized interrupt handler. (e.g., division by zero)

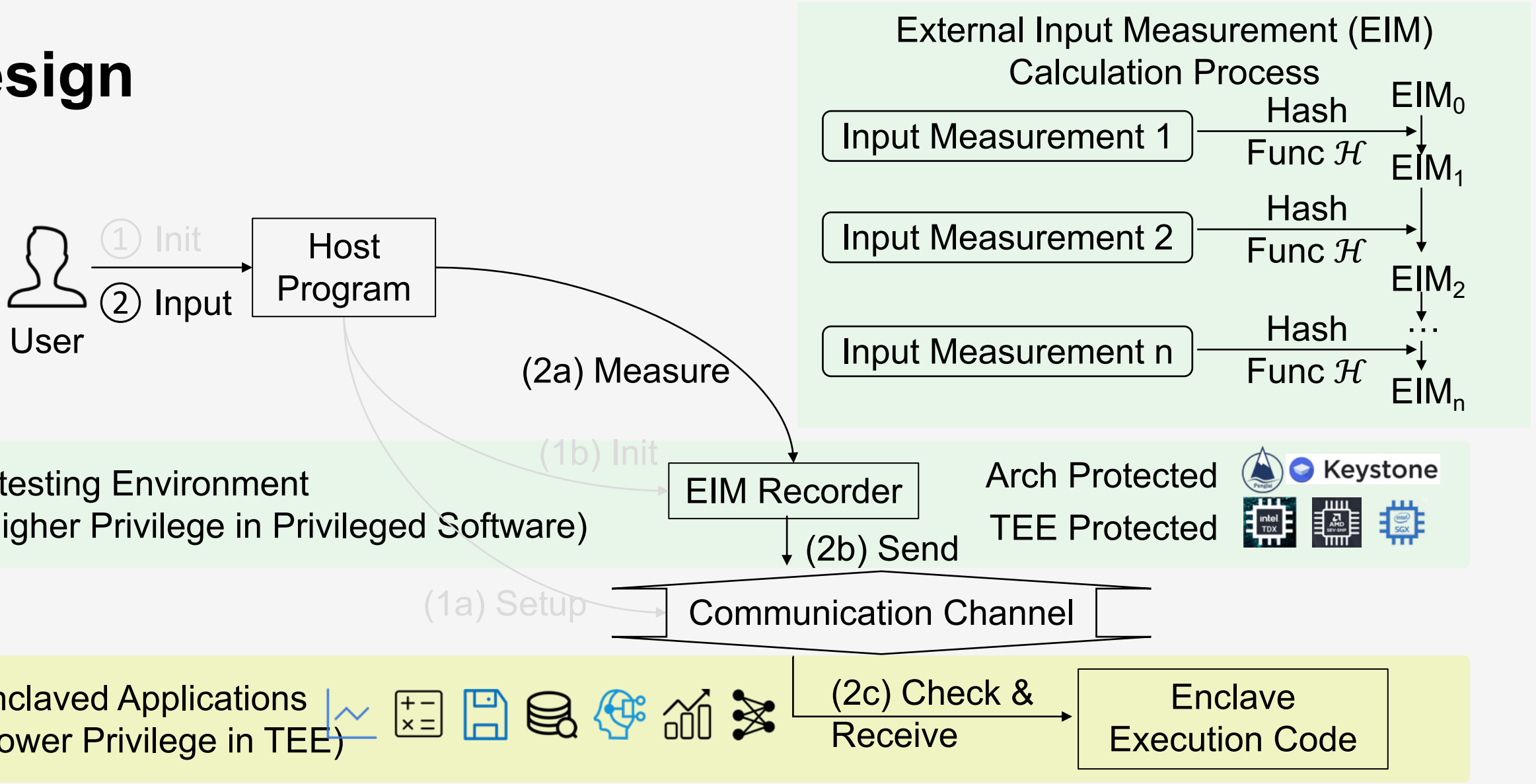
w/ host inputs

Inv. 3-2: the host program inputs need to be **synchronized** with the user and **integrated** into the request-response protocols. (e.g., syscall)

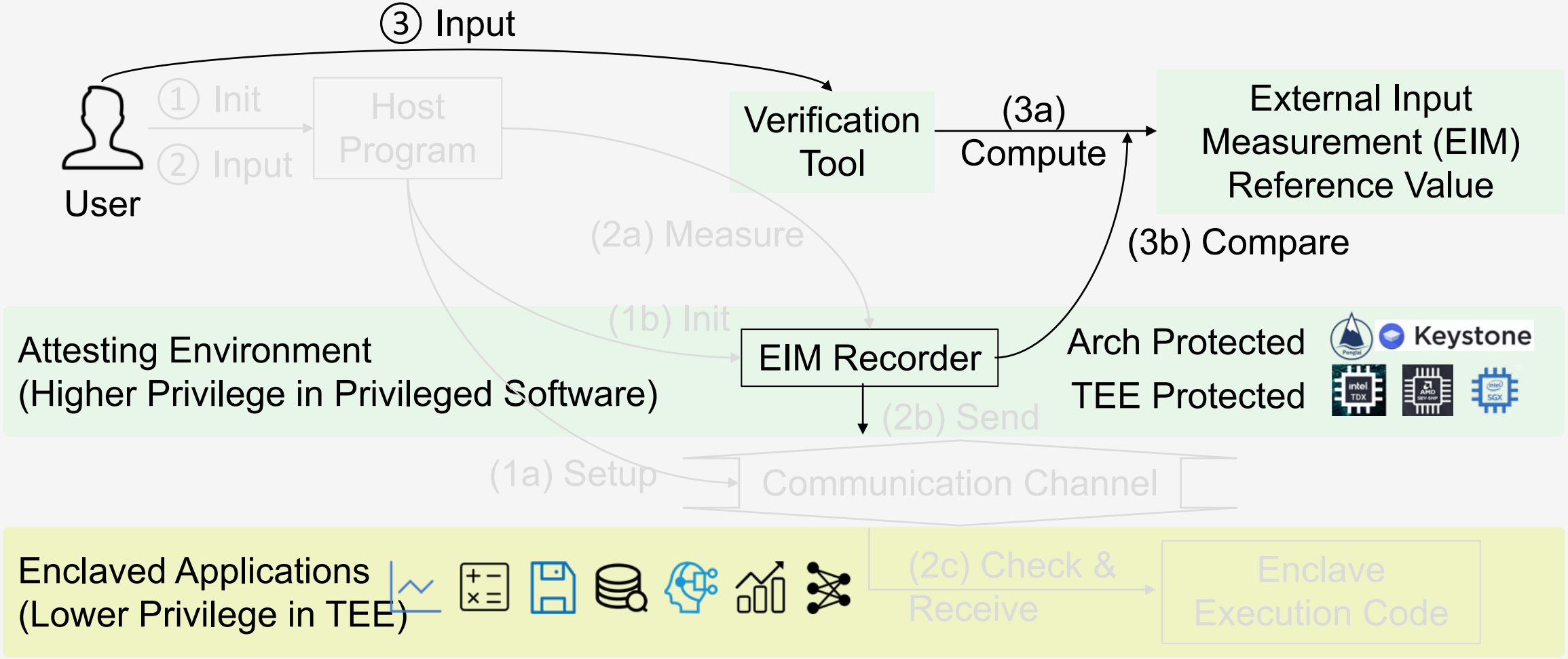
Design


① Init
Host Program
User
(1b) Init
EIM Recorder
Attesting Environment
(Higher Privilege in Privileged Software)
Arch Protected
TEE Protected
Keystone
(1a) Setup
Communication Channel
Enclaved Applications
(Lower Privilege in TEE)
Enclave Execution Code

Design

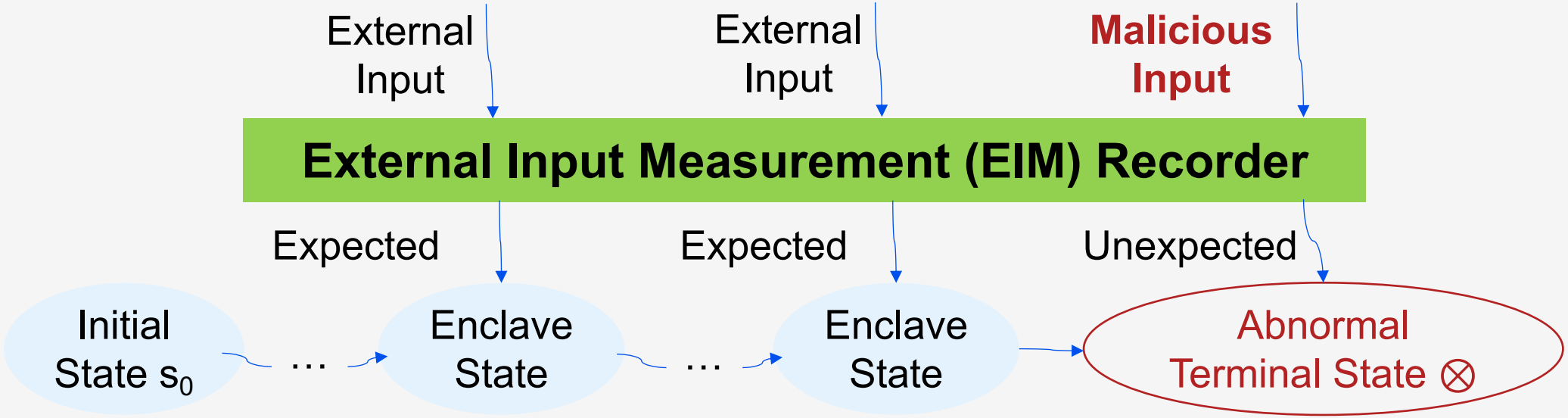

External Input Measurement (EIM) Calculation Process
Input Measurement 1
Hash Func $\mathcal{H}$
EIM_0
EIM_1
Input Measurement 2
Hash Func $\mathcal{H}$
EIM_2
Input Measurement n
Hash Func $\mathcal{H}$
EIM_n
User
① Init
② Input
Host Program
(2a) Measure
(1b) Init
EIM Recorder
(2b) Send
Attesting Environment
(Higher Privilege in Privileged Software)
Arch Protected
TEE Protected
Keystone
(1a) Setup
Communication Channel
Enclaved Applications
(Lower Privilege in TEE)
(2c) Check & Receive
Enclave Execution Code

Design


③ Input
User
① Init
② Input
Host Program
Verification Tool
(3a) Compute
External Input Measurement (EIM) Reference Value
(3b) Compare
(2a) Measure
(1b) Init
Attesting Environment
(Higher Privilege in Privileged Software)
EIM Recorder
Arch Protected
TEE Protected
Keystone
(2b) Send
(1a) Setup
Communication Channel
Enclaved Applications
(Lower Privilege in TEE)
(2c) Check & Receive
Enclave Execution Code

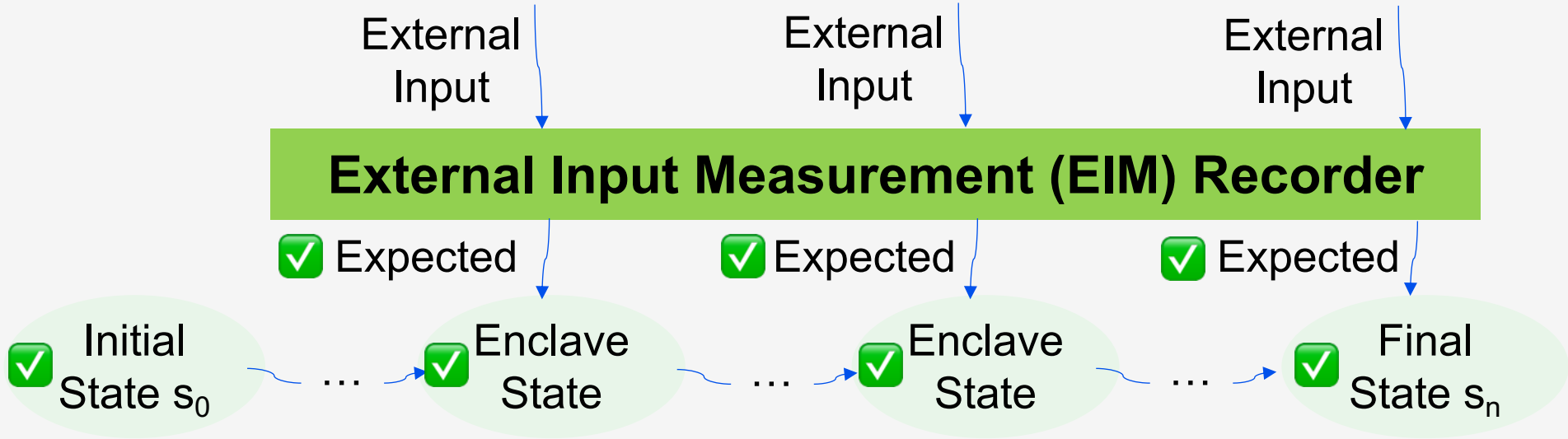
Security Analysis

- Lemma. The first malicious input causing violations of EXIA's design invariants will be captured by EIM.

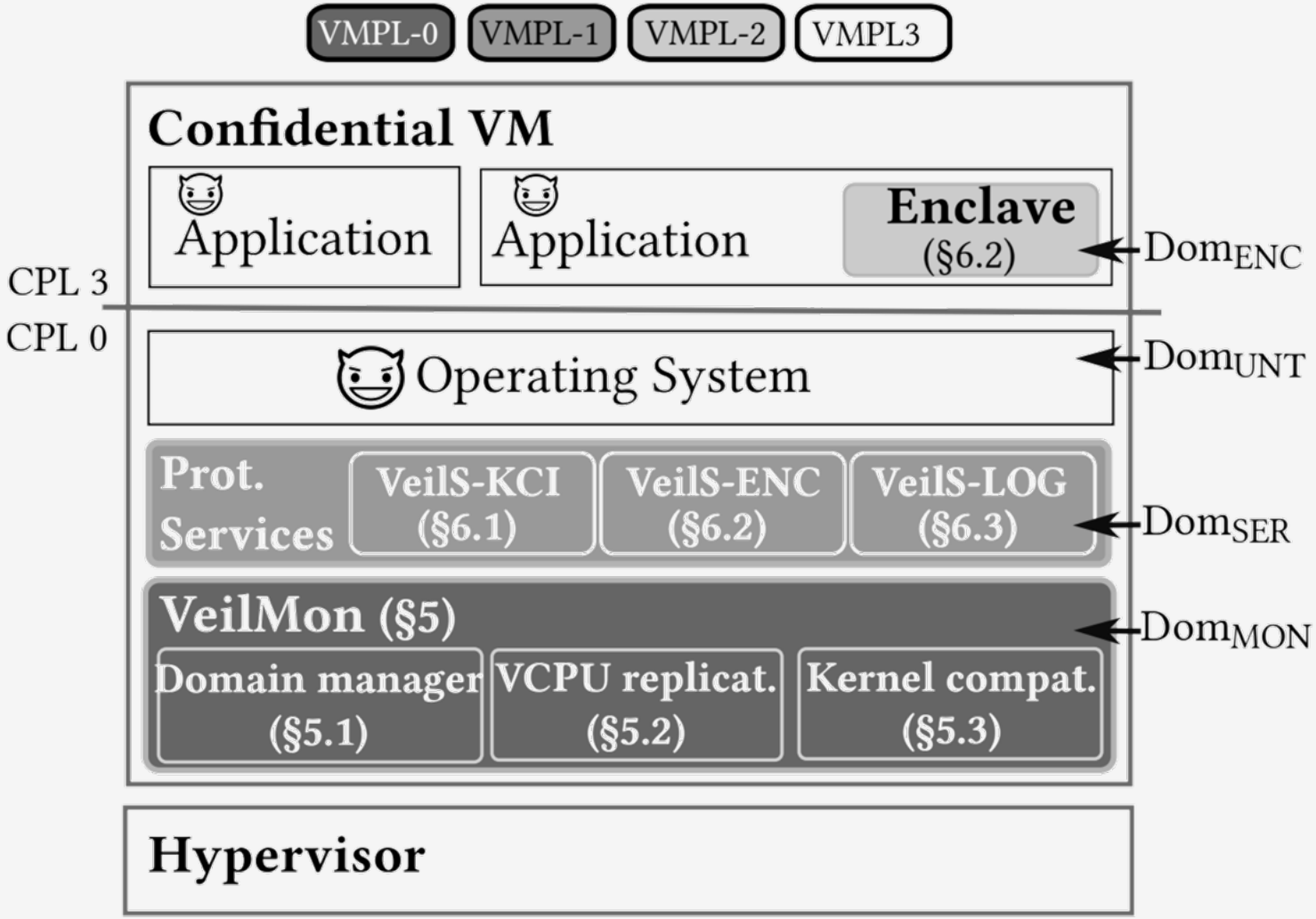


Security Analysis

- Theorem. If the received initial state s_0 and EIM match the reference values the user derives locally from its benign inputs, the user can ensure that the resulting state of the enclaved application is trusted.

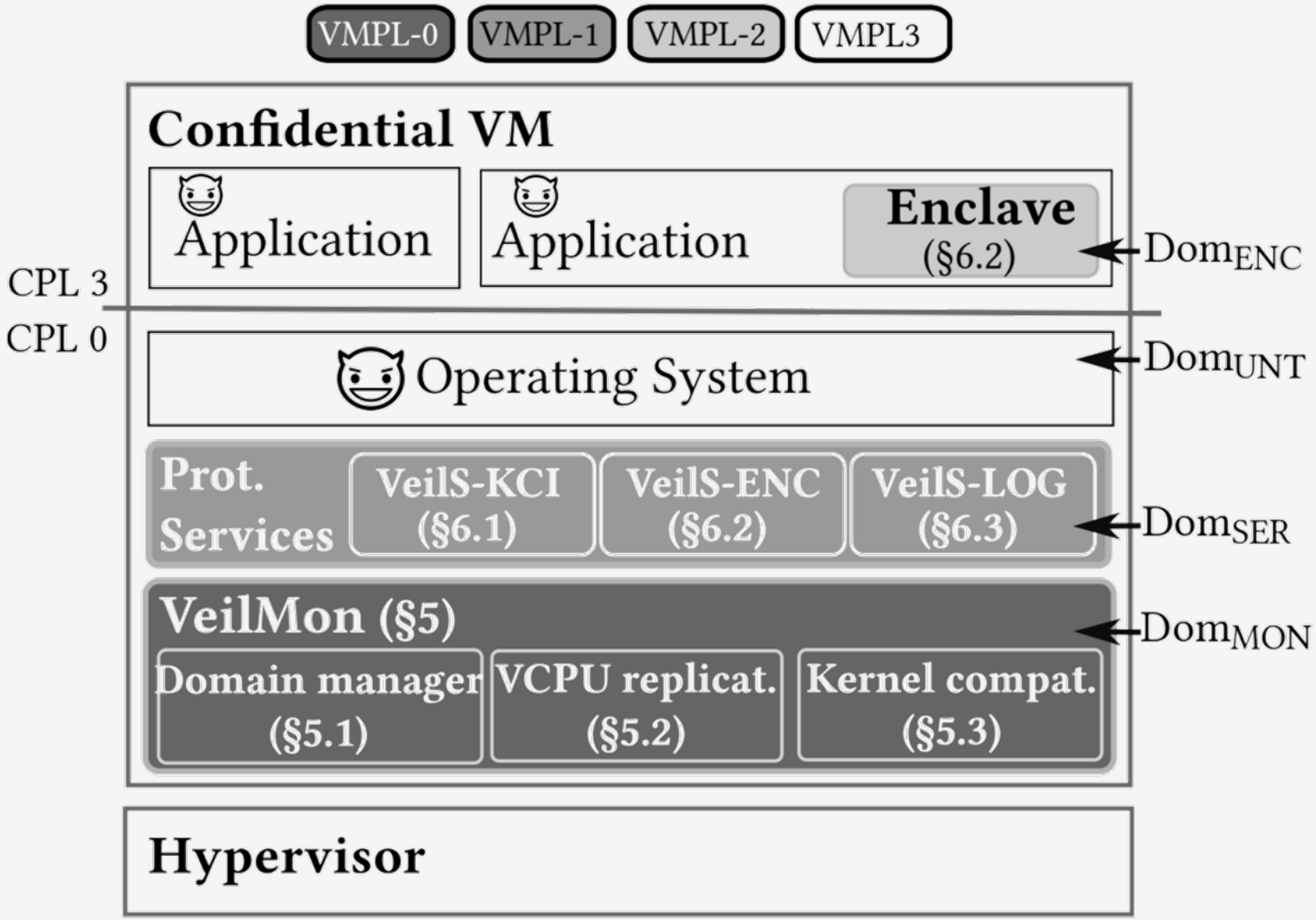


EXIA-SEV



Ahmad, Adil, et al. "Veil: A protected services framework for confidential virtual machines." *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Volume 4. 2023.

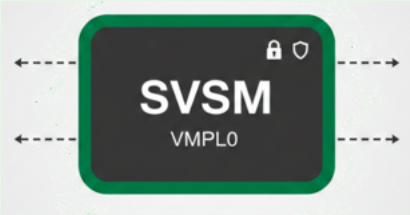
EXIA-SEV



Off-the-shelf

Privilege

Leveling



Dual-Page Transfer

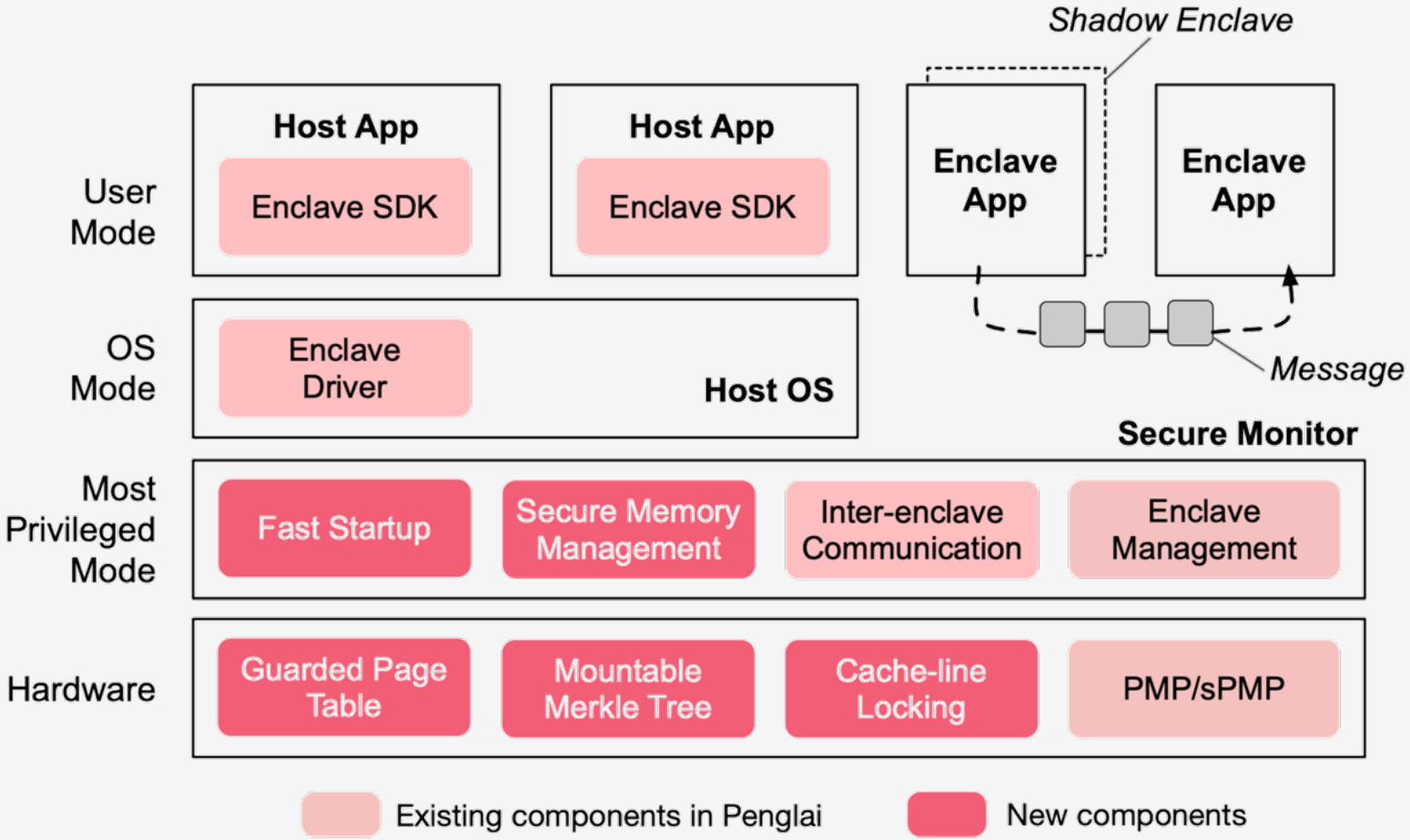
Freshness Verification

VC Handler / …

Syscalls (VEIL supported)

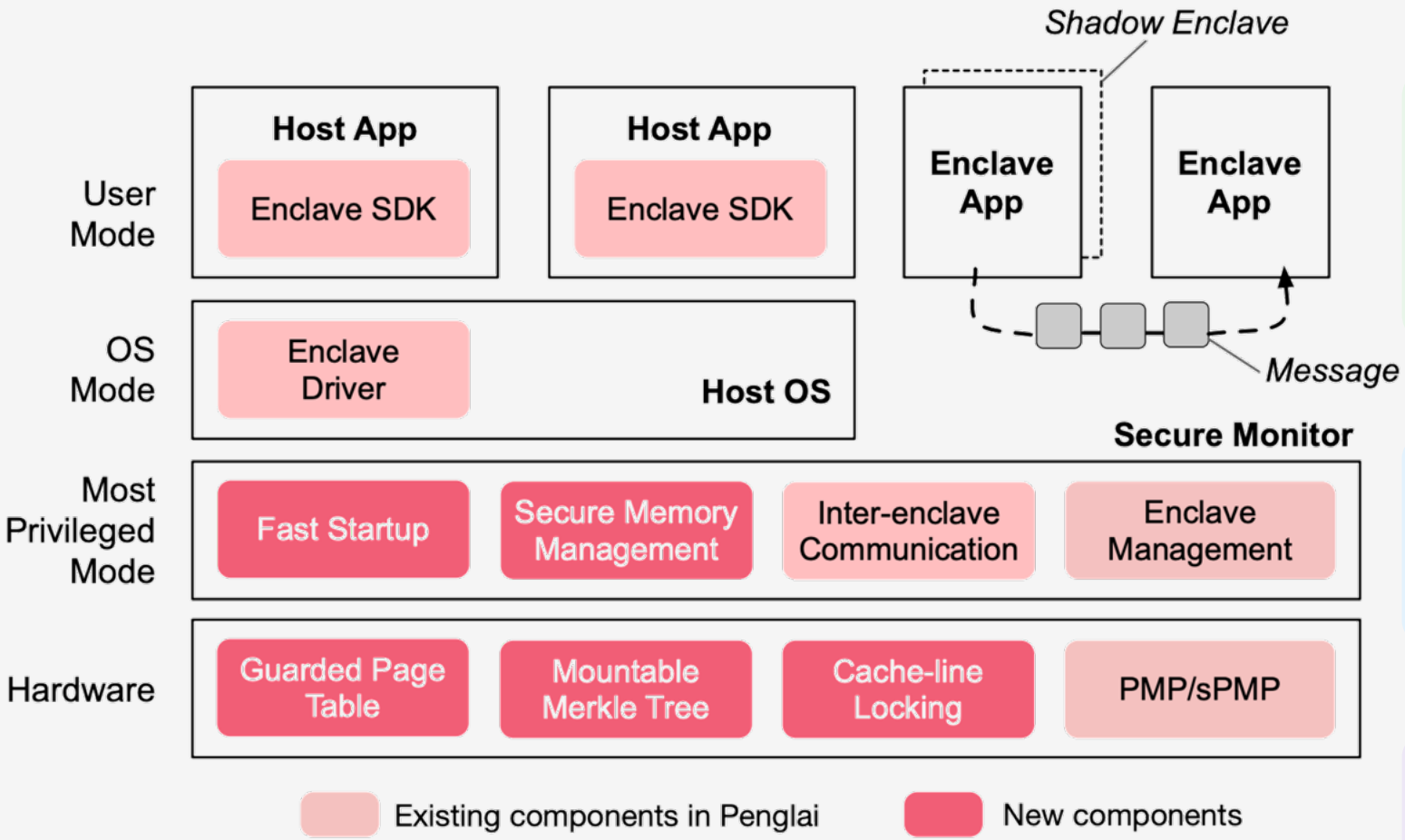
Ahmad, Adil, et al. "Veil: A protected services framework for confidential virtual machines." *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Volume 4. 2023.

EXIA-Penglai



Feng, Erhu, et al. "Scalable memory protection in the PENGLAI enclave." *15th USENIX Symposium on Operating Systems Design and Implementation OSDI 21*. 2021.

EXIA-Penglai



Architectural Extension

RISC-V

Shared-Page Permission Flipping

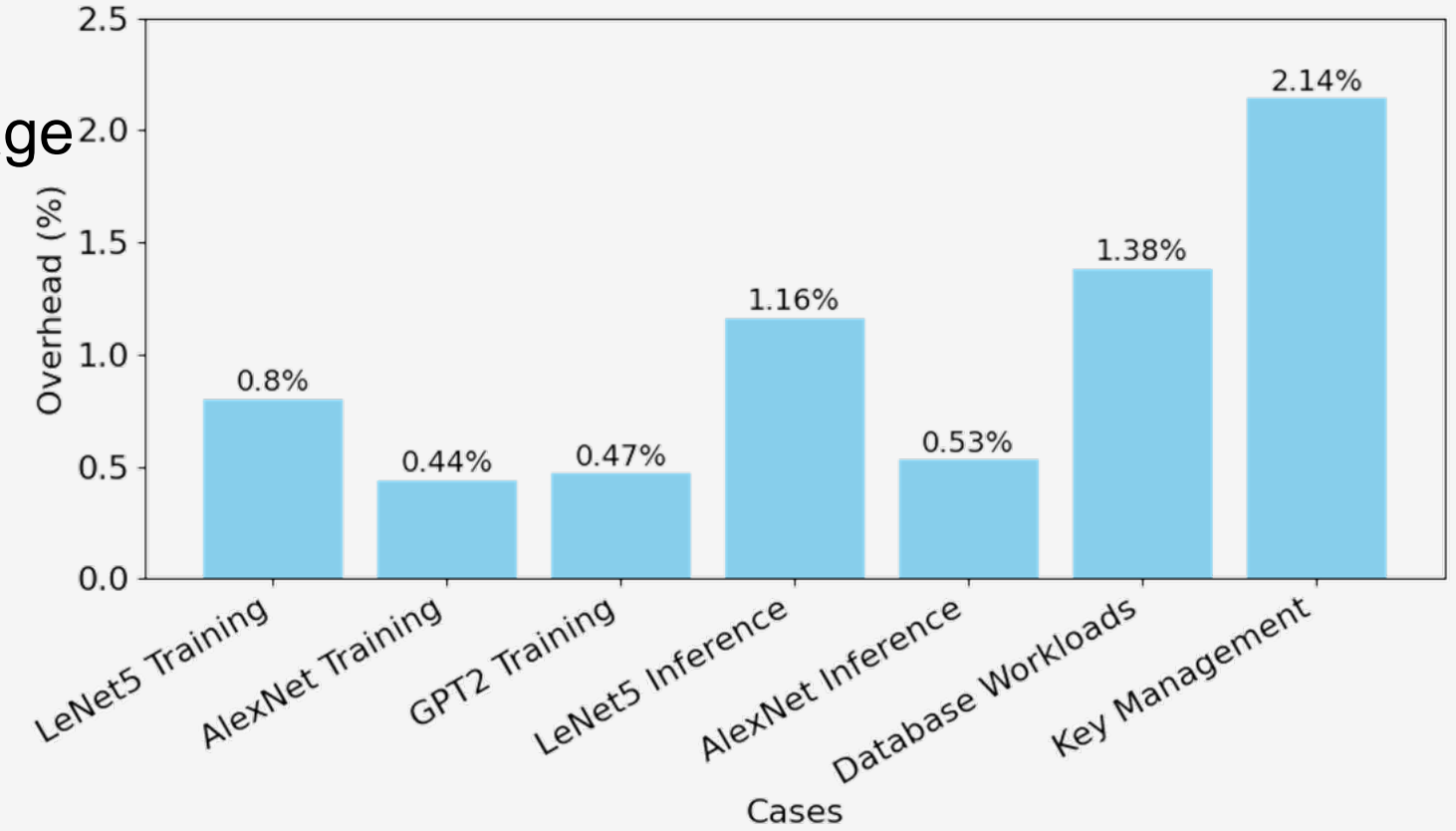
Freshness Verification

Does not support interrupts

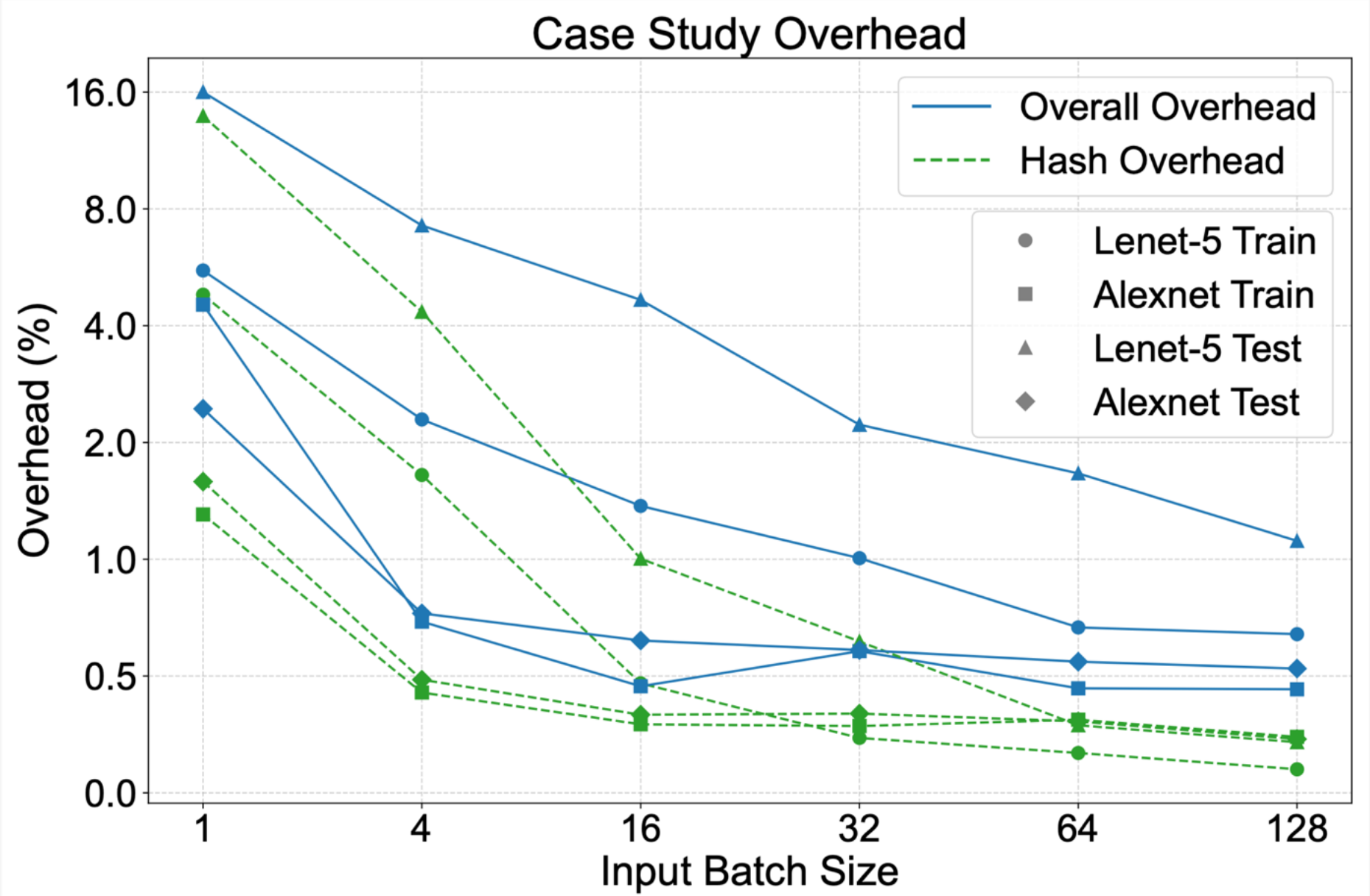
Feng, Erhu, et al. "Scalable memory protection in the PENGLAI enclave." *15th USENIX Symposium on Operating Systems Design and Implementation OSDI 21*. 2021.

Evaluation and Case Studies

- Micro Performance Overhead
  - EXIA-SEV: 1.51ms / 4K page
  - EXIA-Penglai: 0.17ms / 4K page
- Security Evaluation
  - ✅ Overflow
  - ✅ Dangling pointer misuse



Case Studies



Thank You!
Q&A

GitHub: https://github.com/xmhuangzhen/Exia

Zhen Huang: xmhuangzhen@sjtu.edu.cn