HyperEnclave: An Open and Cross-platform Trusted Execution Environment

Yuekai Jia[1], Shuang Liu[2], Wenhao Wang[3,4], Yu Chen[1], Zhengde Zhai[2], Shoumeng Yan[2], Zhengyu He[2]

[1]*Tsinghua University*
[2]*Ant Group*
[3]*SKLOIS, Institute of Information Engineering, CAS*
[4]*School of Cyber Security, University of Chinese Academy of Sciences*

清華大學
Tsinghua University

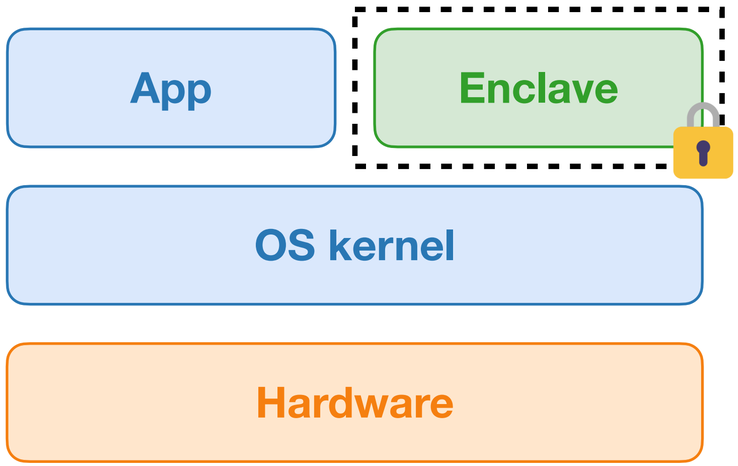
蚂蚁集团
ANT GROUP

中國科學院
CHINESE ACADEMY OF SCIENCES

USENIX ATC '22, July 11-13, 2022

Background

- **Trusted execution environments** (TEEs):
  - Protected sensitive data in hardware-enforced isolated environments (enclaves)
  - Thwart OS-level and physical attackers



Process-based

VM-based

Separate worlds

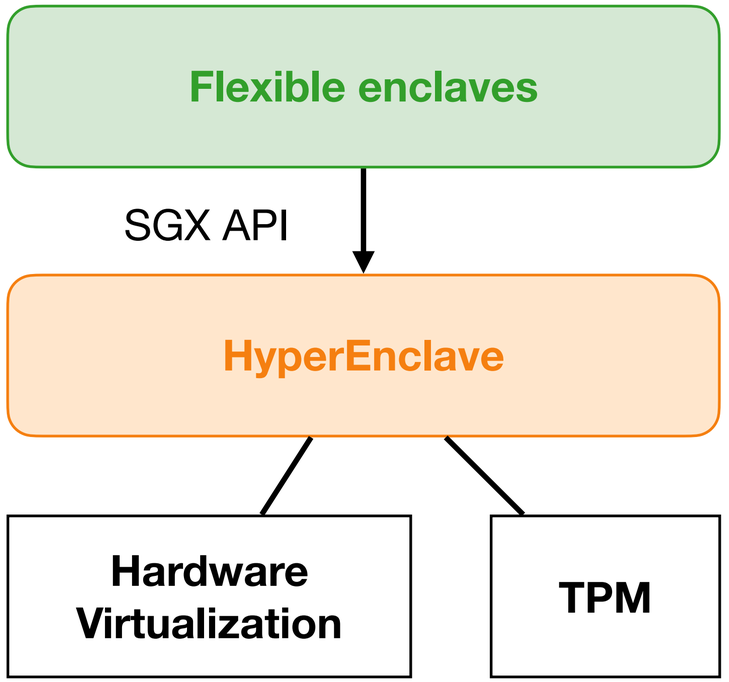
Motivation

- Most of today's TEEs:
  - Require **specific hardware** changes: slow to evolve
  - **Close-sourced**: difficult to audit
  - Enclaves run in **fixed mode**: inflexible to adapt to diverse workloads

    (e.g., applications that want to access privileged resources)

Design Goals

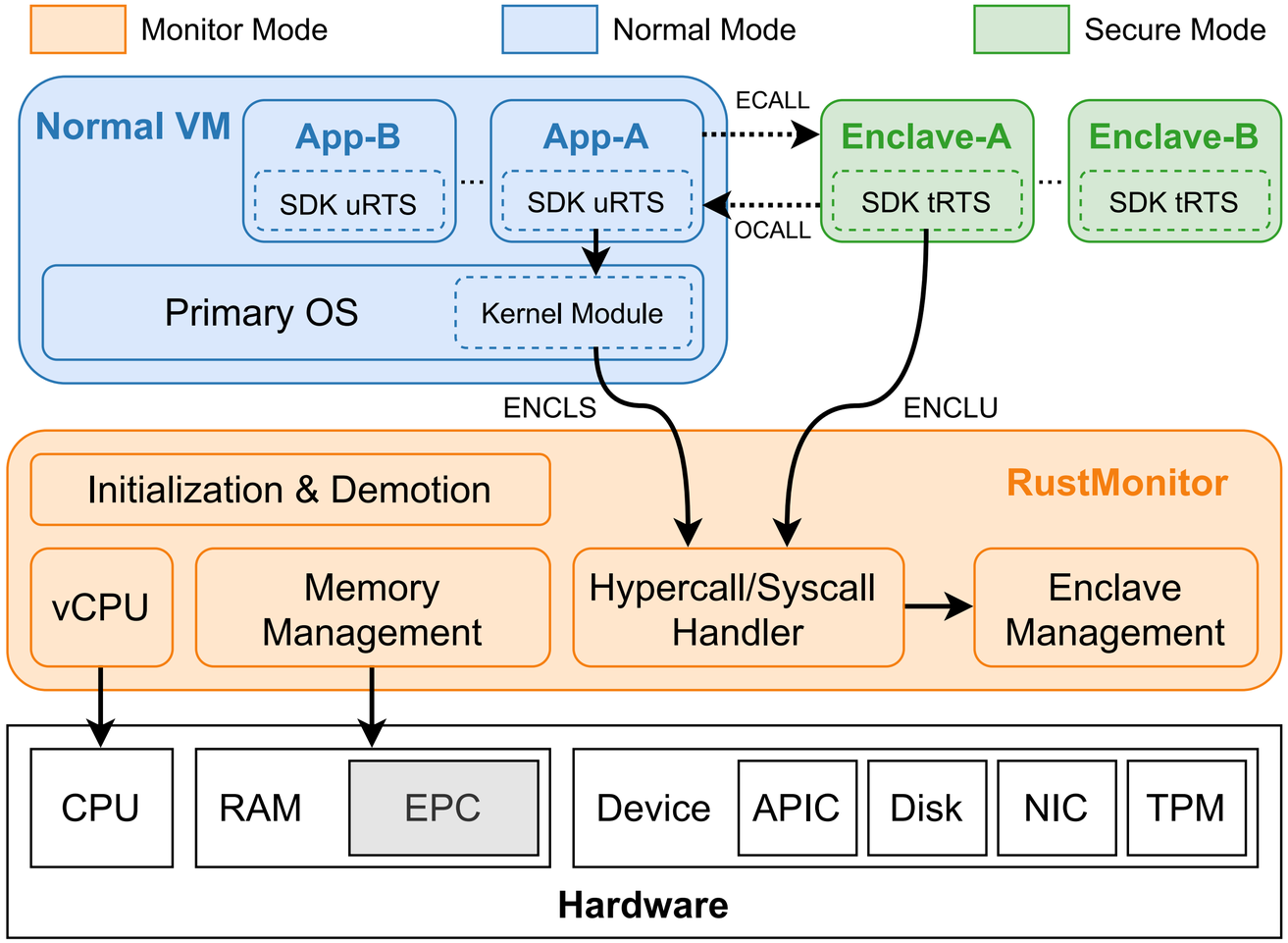
- **G1: Minimum hardware requirements**
  - Hardware virtualization + TPM
  - Cross-platform
- **G2: Easy to develop**
  - SGX compatible
  - Port existing SGX programs with little or no code changes
- **G3: Flexible enclave modes**
  - Better fulfill the needs for specific enclave workloads



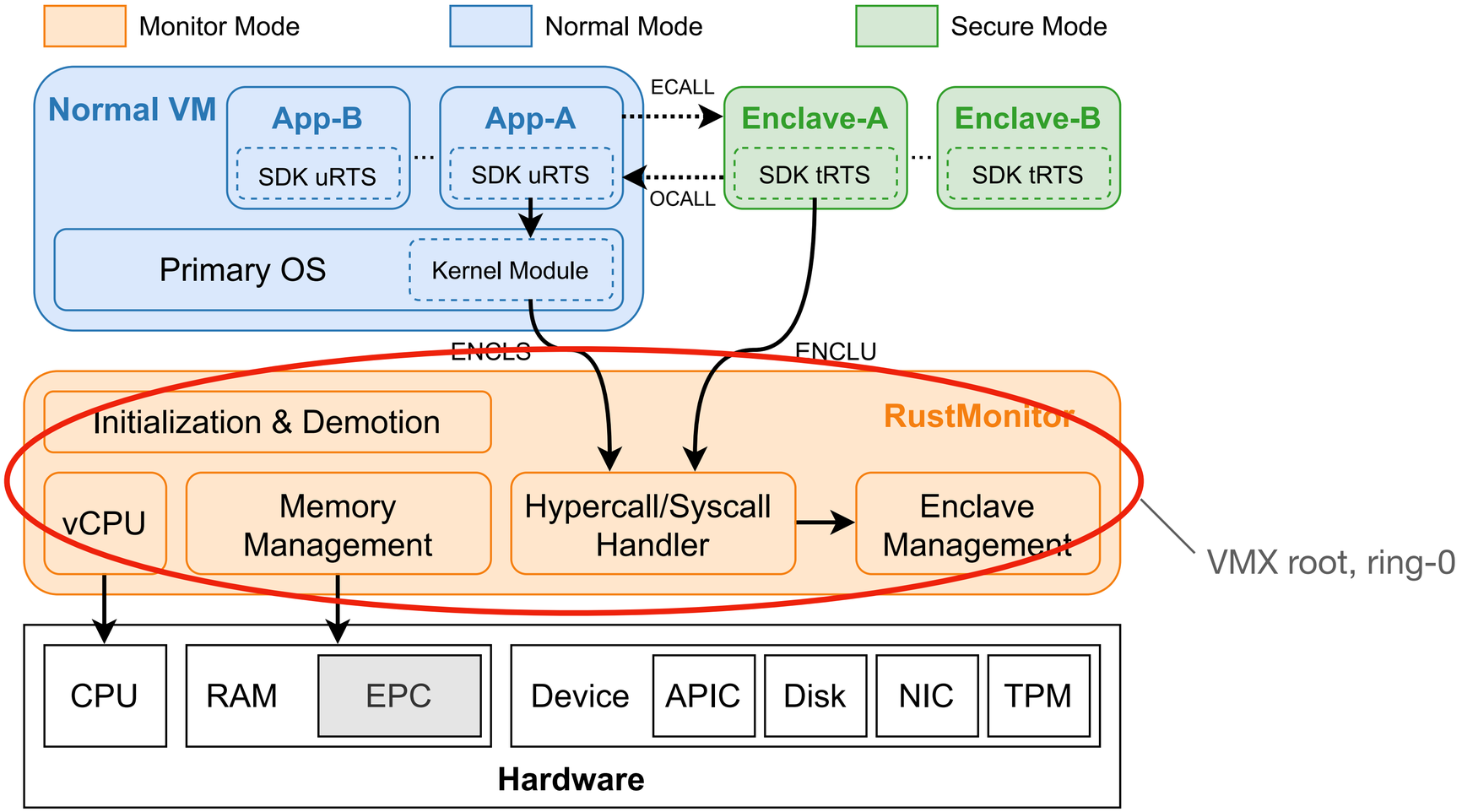
Threat Model

- System is initially benign
  - ✔ Can be compromised after system boot (cloud scenario)
- Malicious software
  - ✔ Application, OS kernel, enclave malware
- Certain physical memory attacks
  - ✔ cold boot attacks, bus snooping attacks
- Certain side channel attacks
  - ✔ page-table-based attacks
- Out of scope
  - ✘ DoS, other side channel attacks (cache timing, speculative execution)

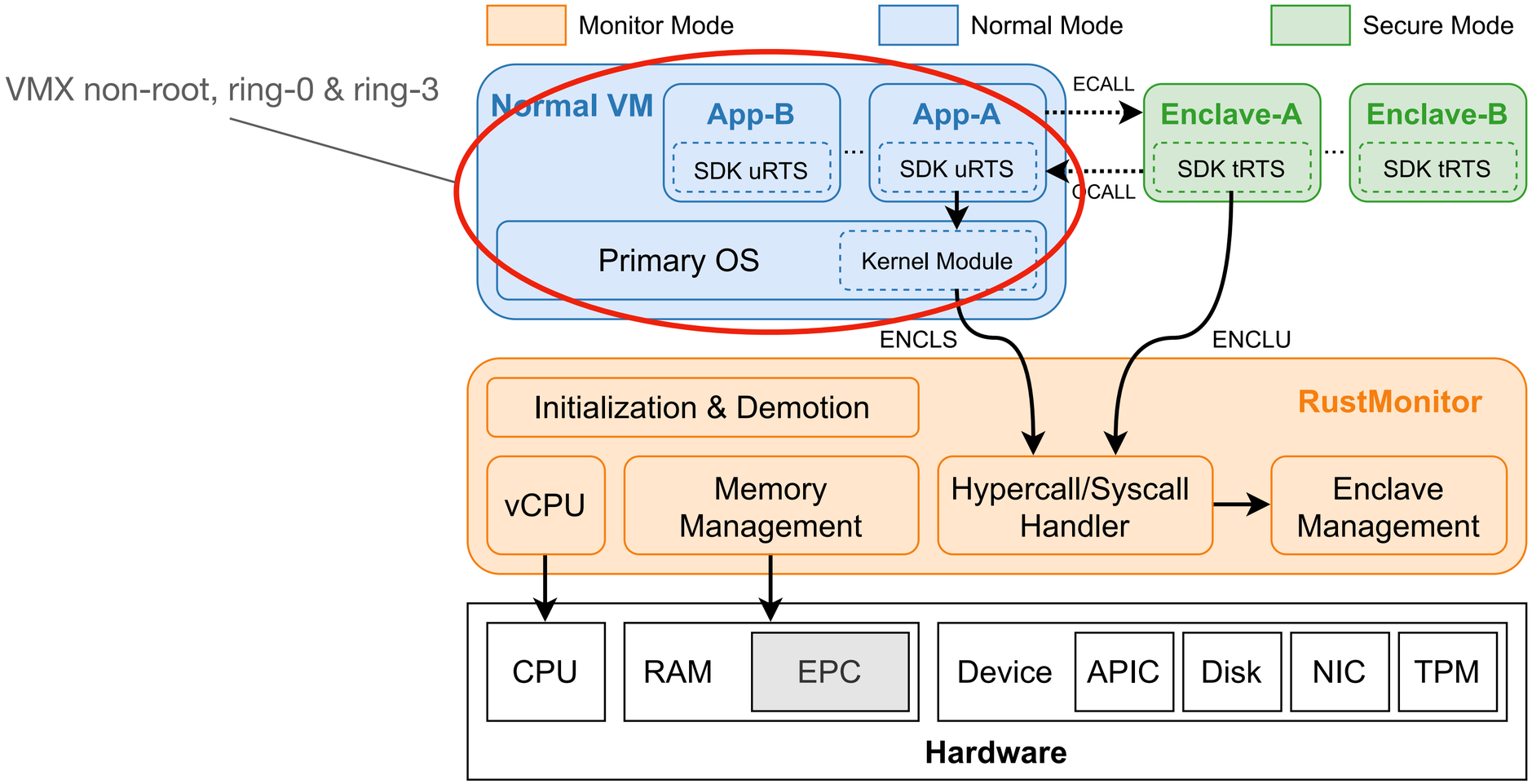
HyperEnclave Overview



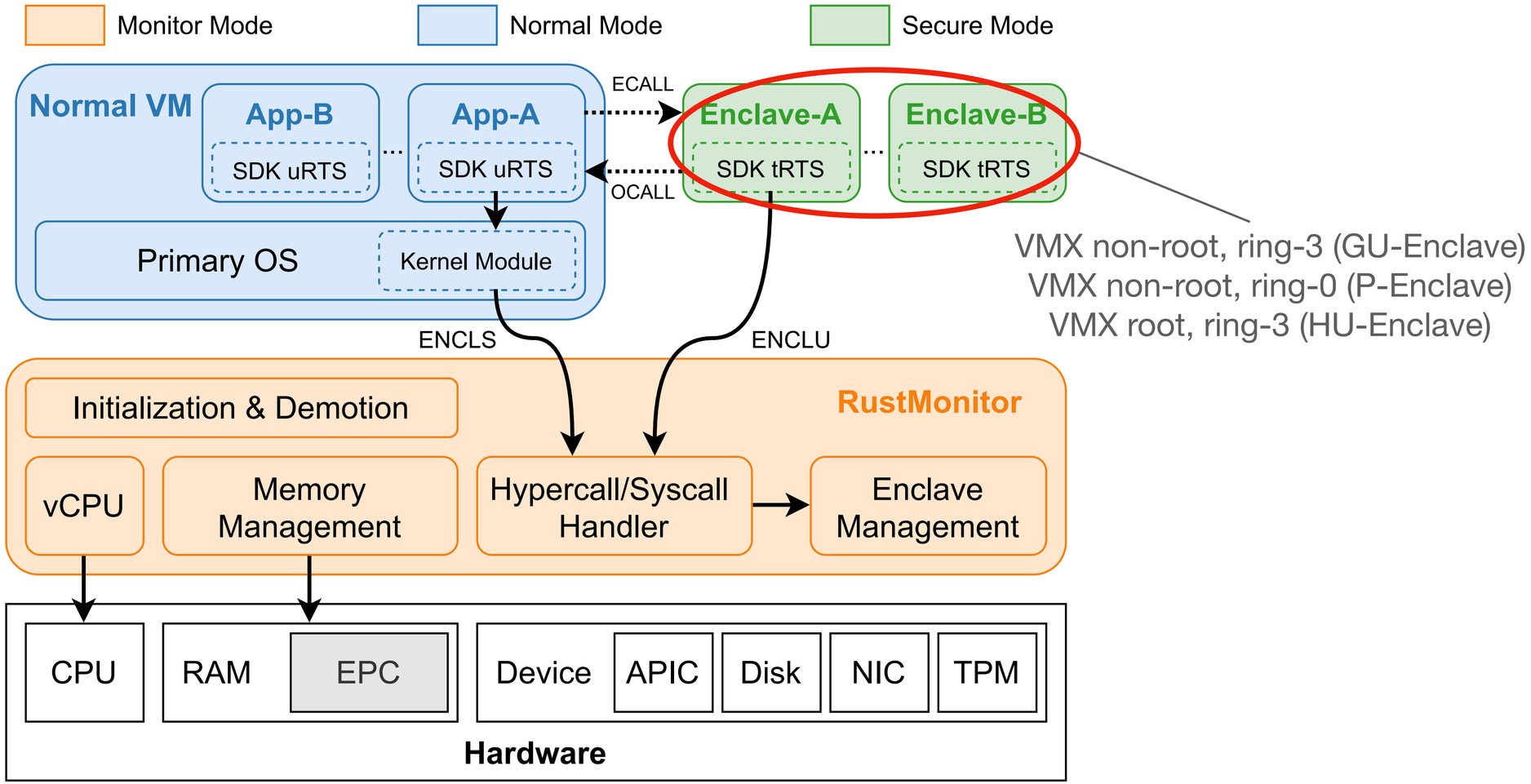
HyperEnclave Overview



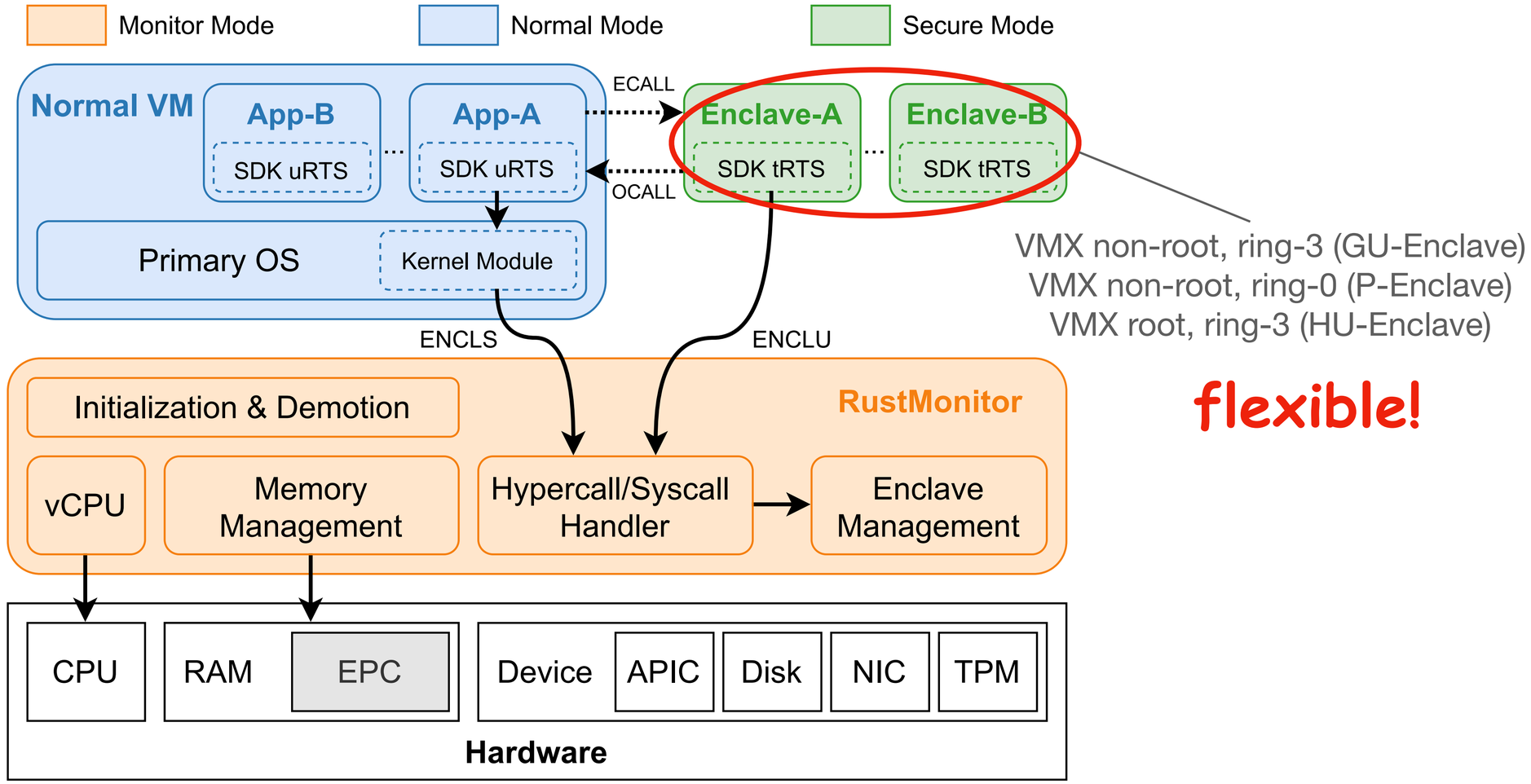
HyperEnclave Overview



HyperEnclave Overview



HyperEnclave Overview

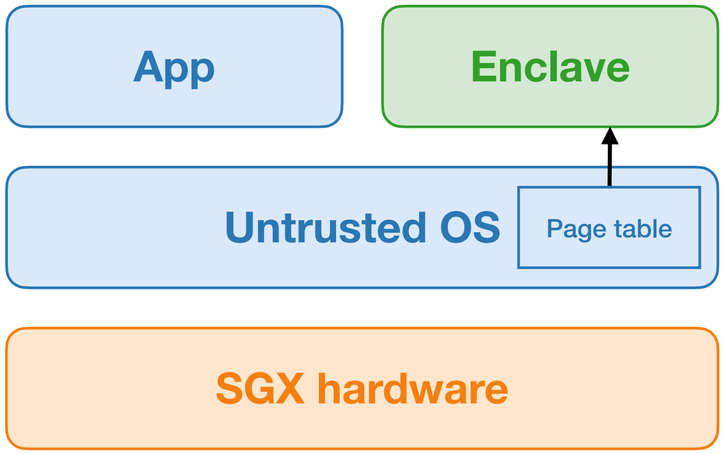


Design Details

1. Memory management and protection
2. Flexible enclave operation mode
3. Trusted Boot
4. The enclave SDK

1. Memory Management and Protection

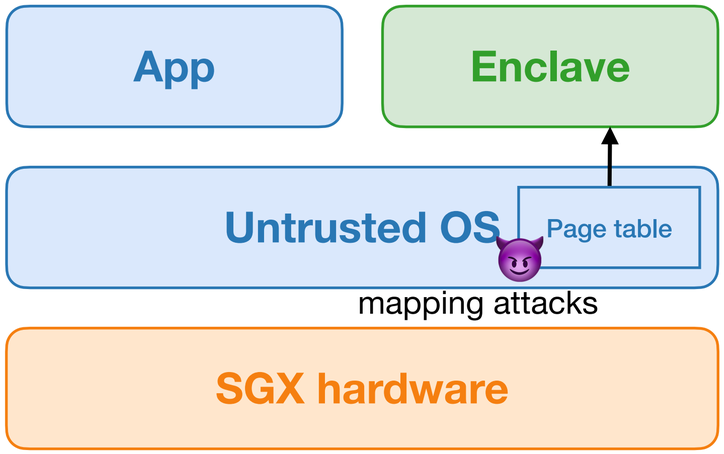
- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



Intel SGX

1. Memory Management and Protection

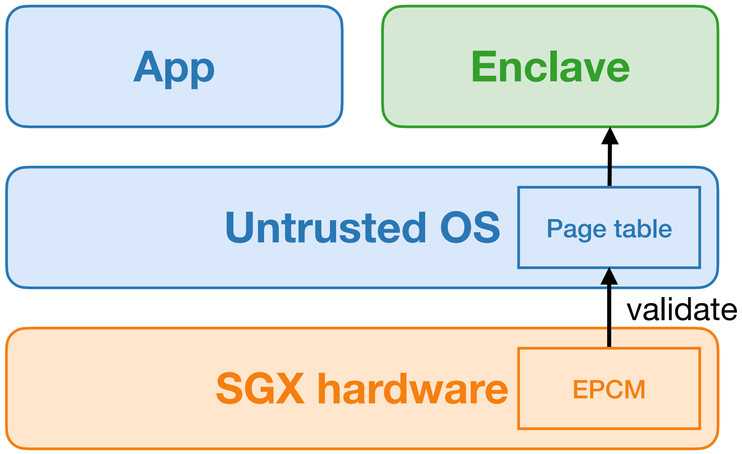
- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



Intel SGX

1. Memory Management and Protection

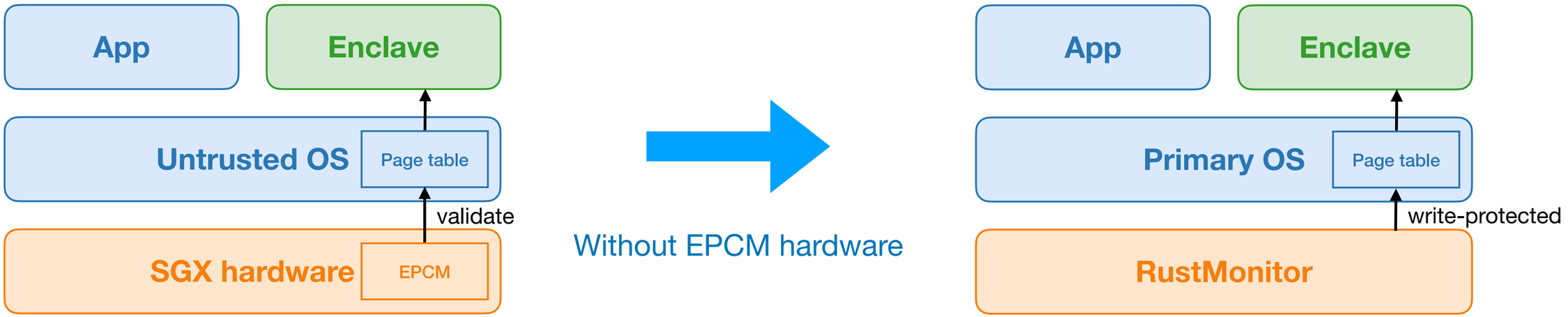
- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



Intel SGX

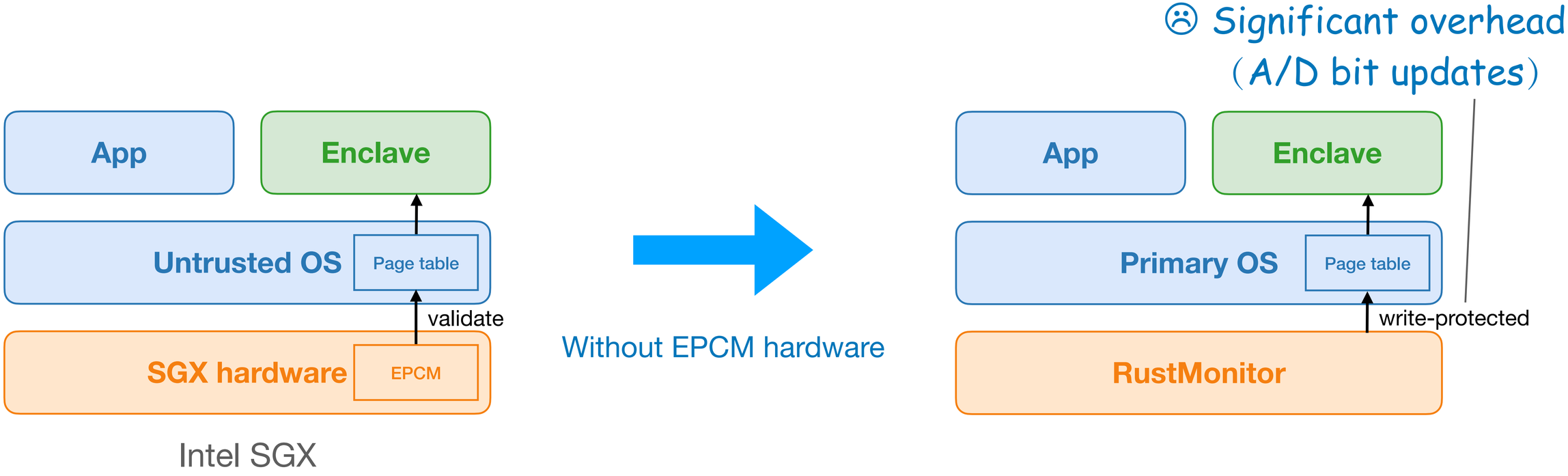
1. Memory Management and Protection

- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



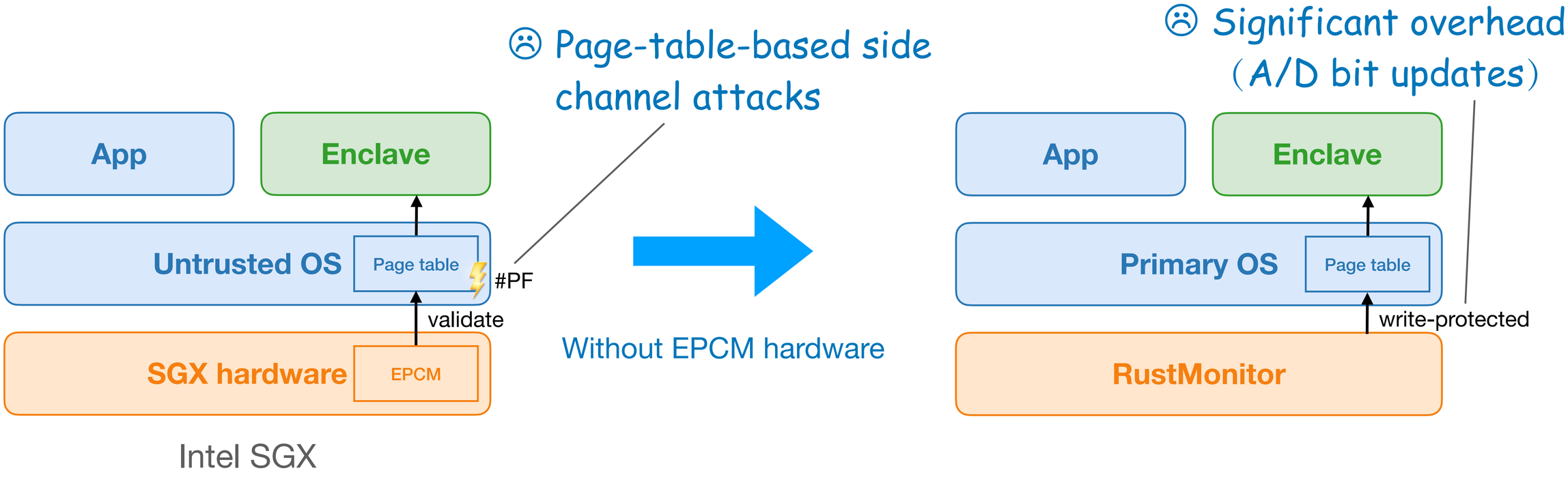
1. Memory Management and Protection

- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



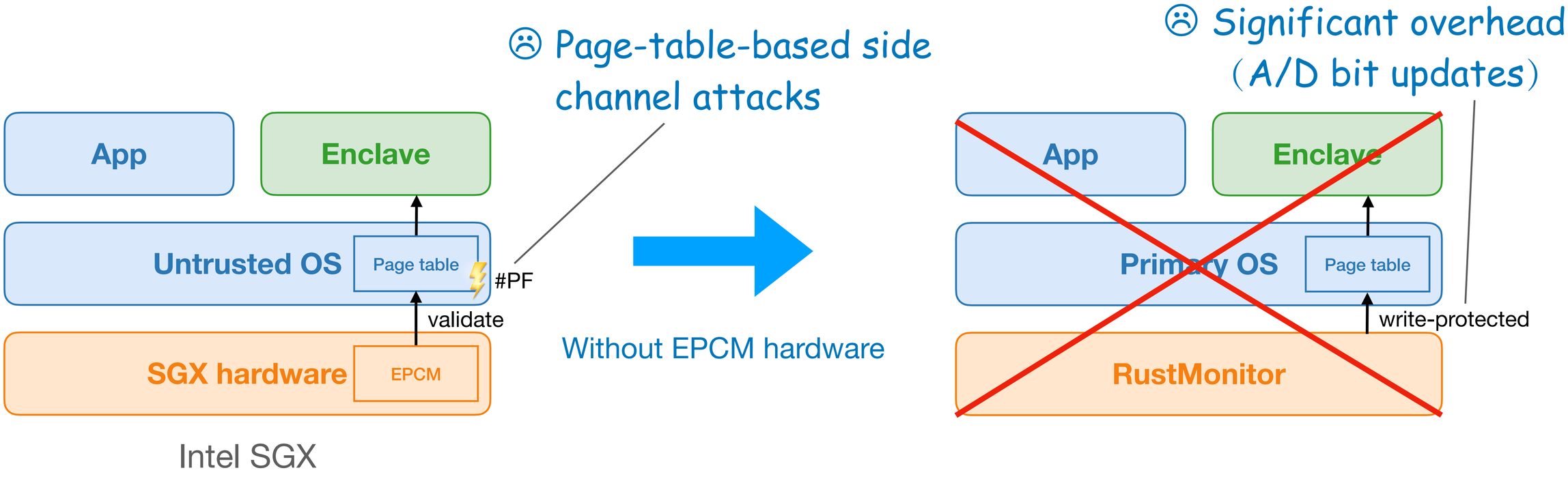
1. Memory Management and Protection

- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



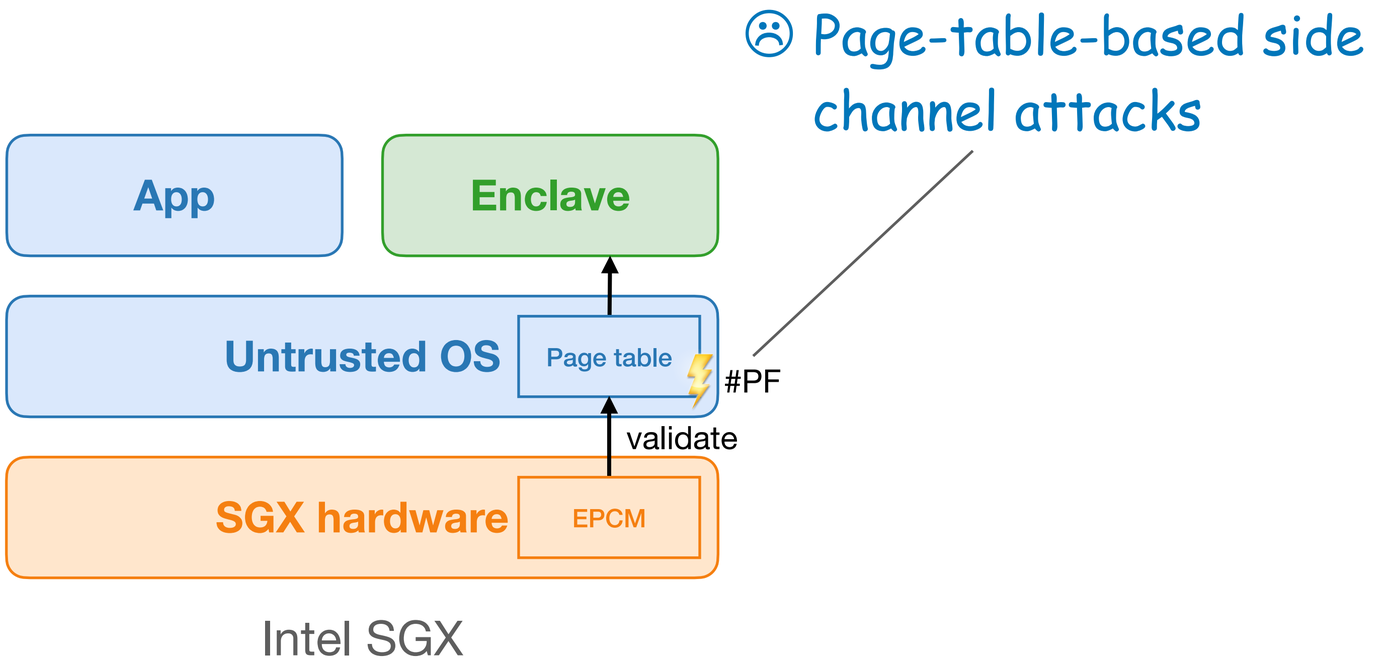
1. Memory Management and Protection

- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



1. Memory Management and Protection

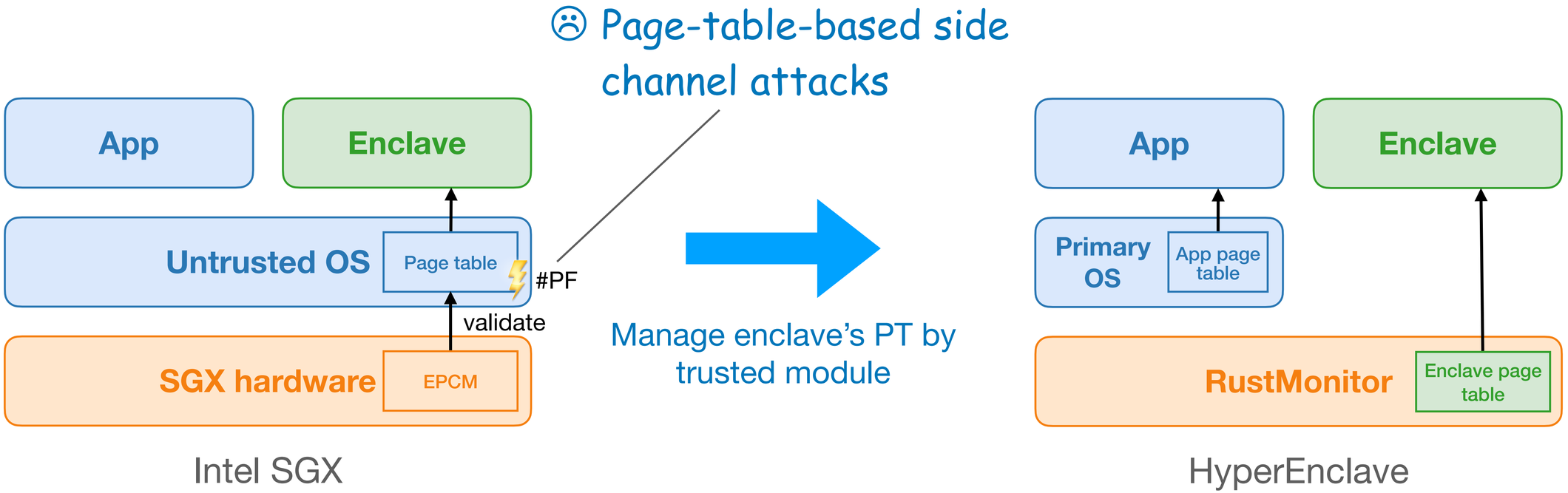
- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



Intel SGX

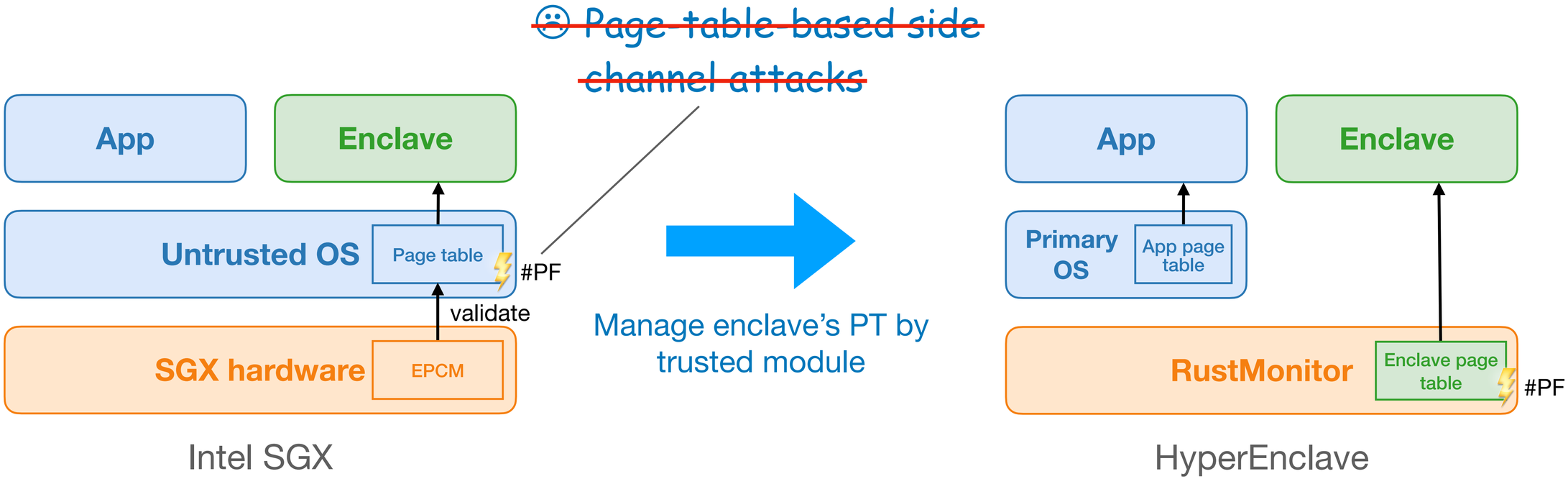
1. Memory Management and Protection

- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



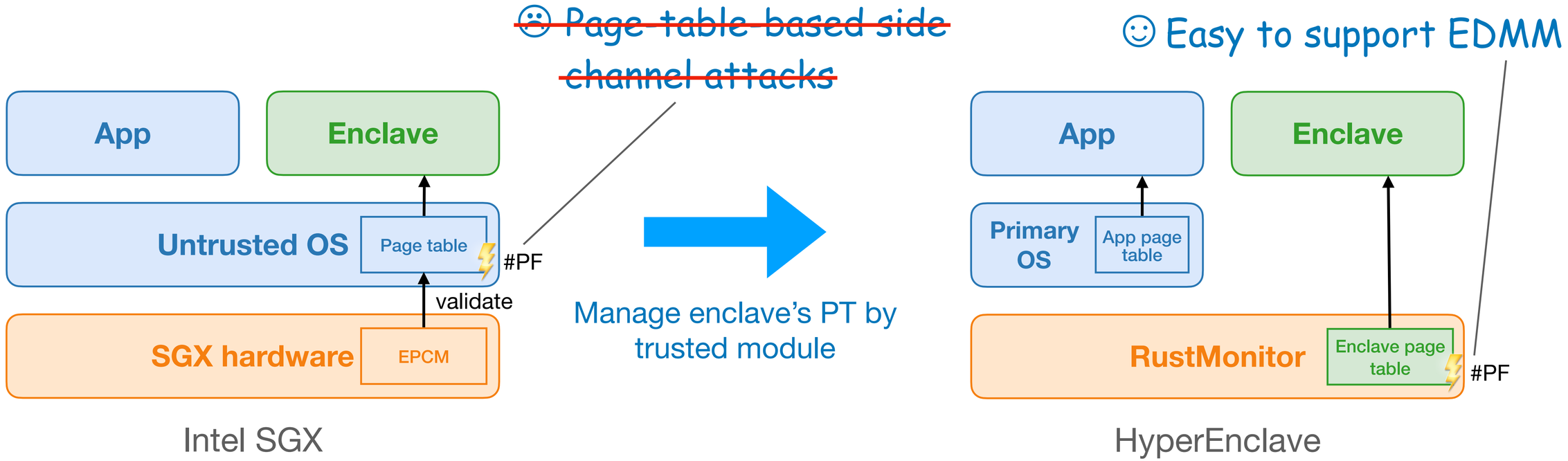
1. Memory Management and Protection

- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



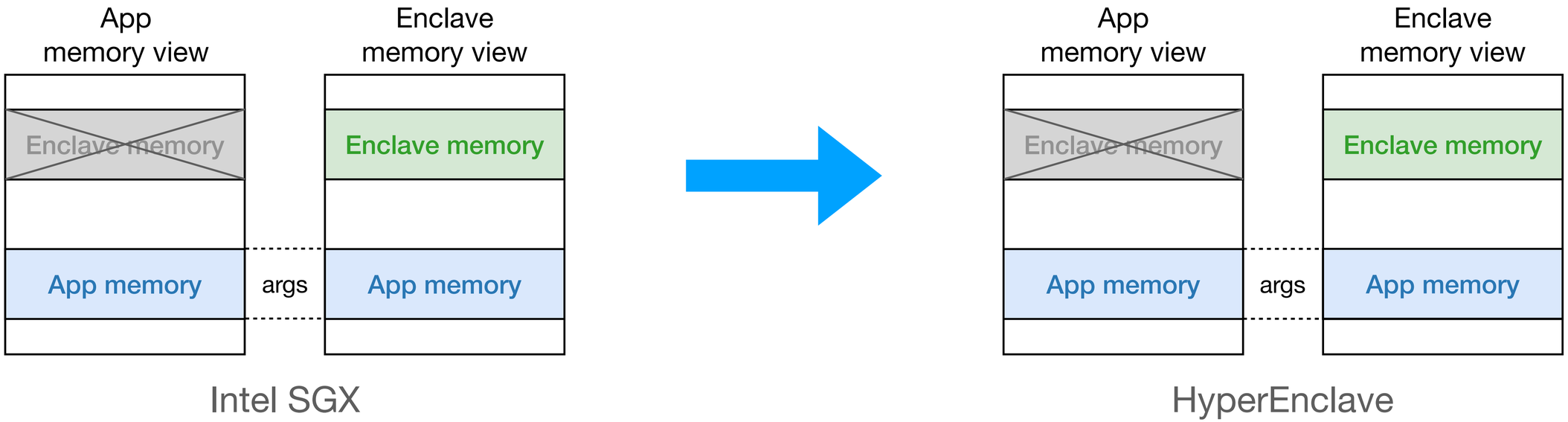
1. Memory Management and Protection

- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



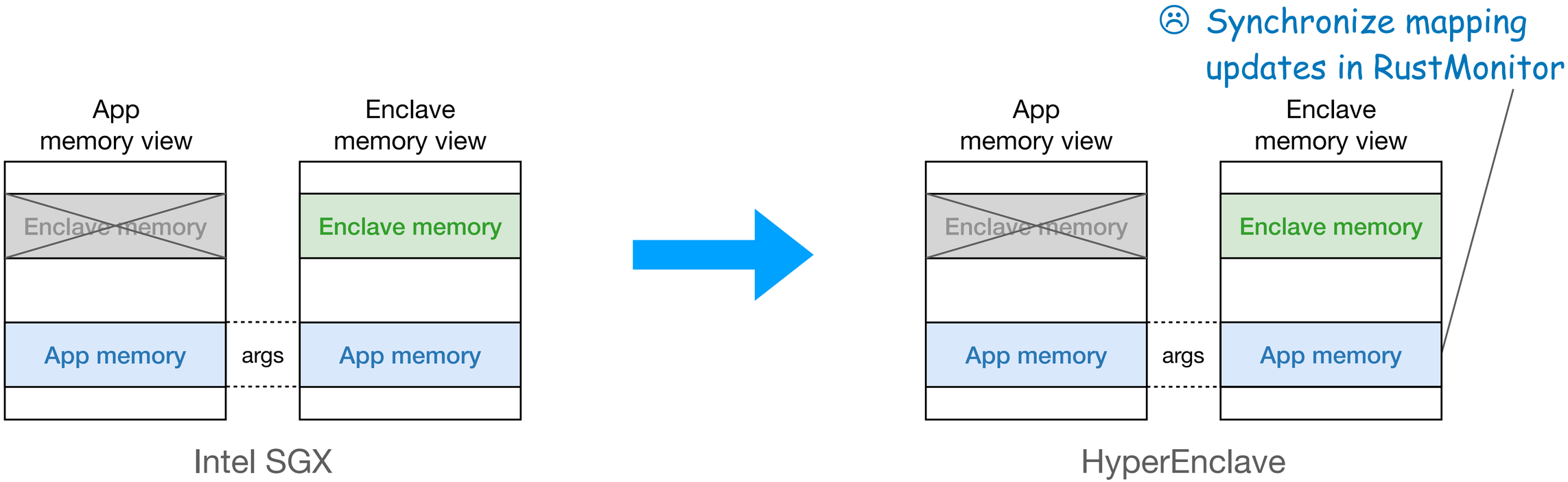
1. Memory Management and Protection

- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



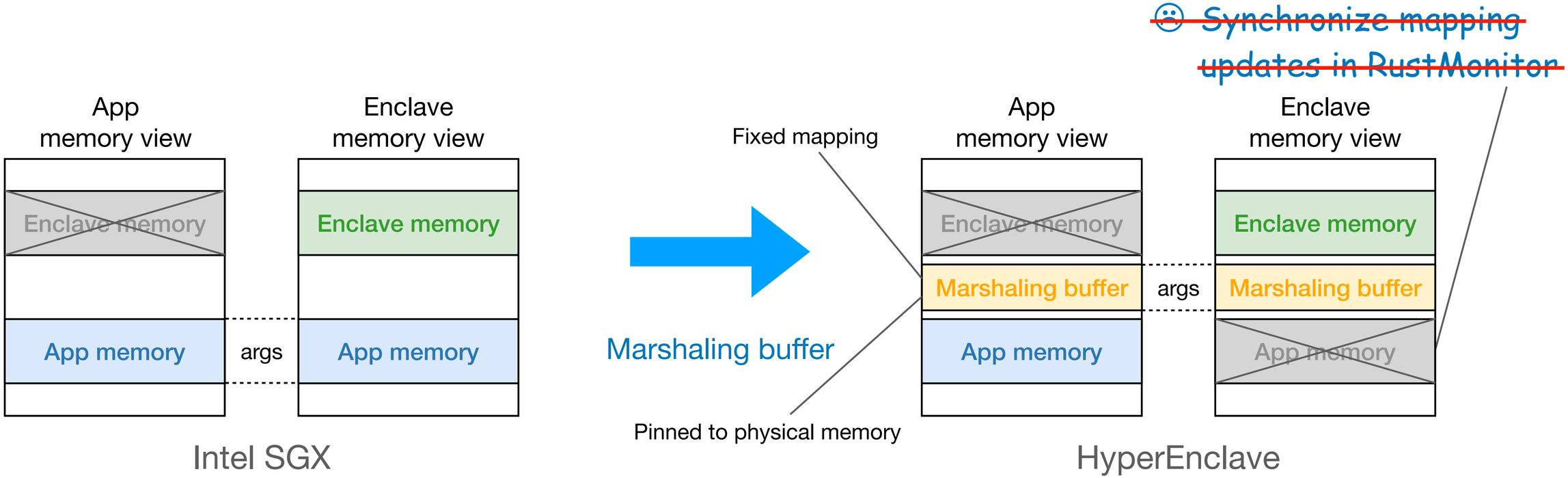
1. Memory Management and Protection

- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



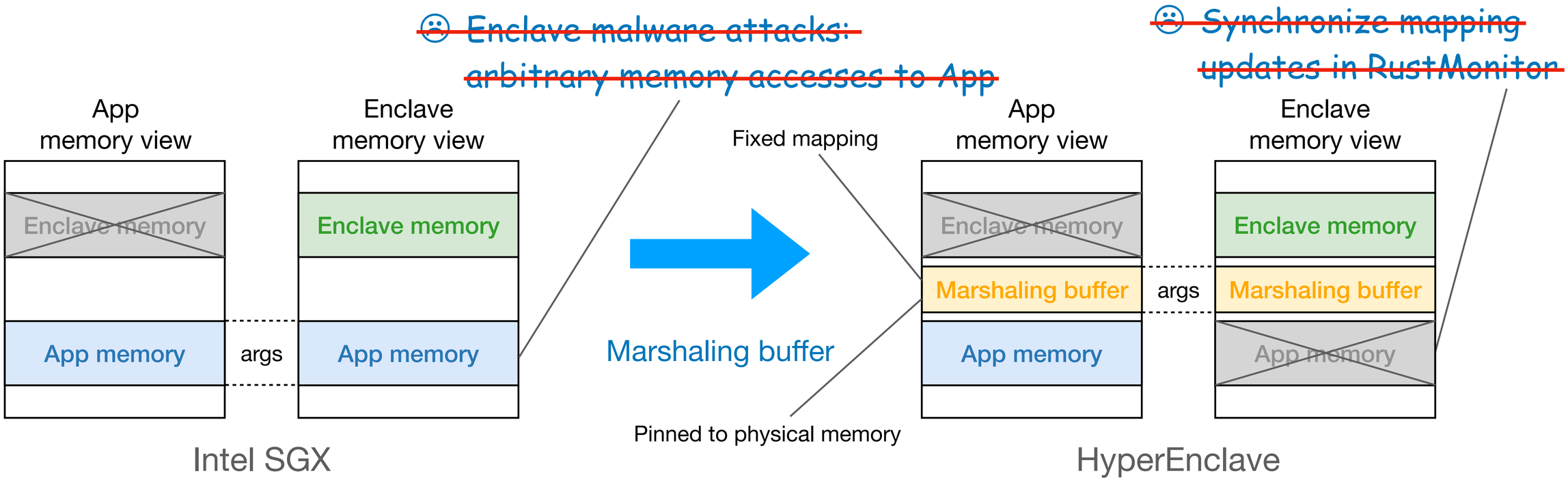
1. Memory Management and Protection

- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



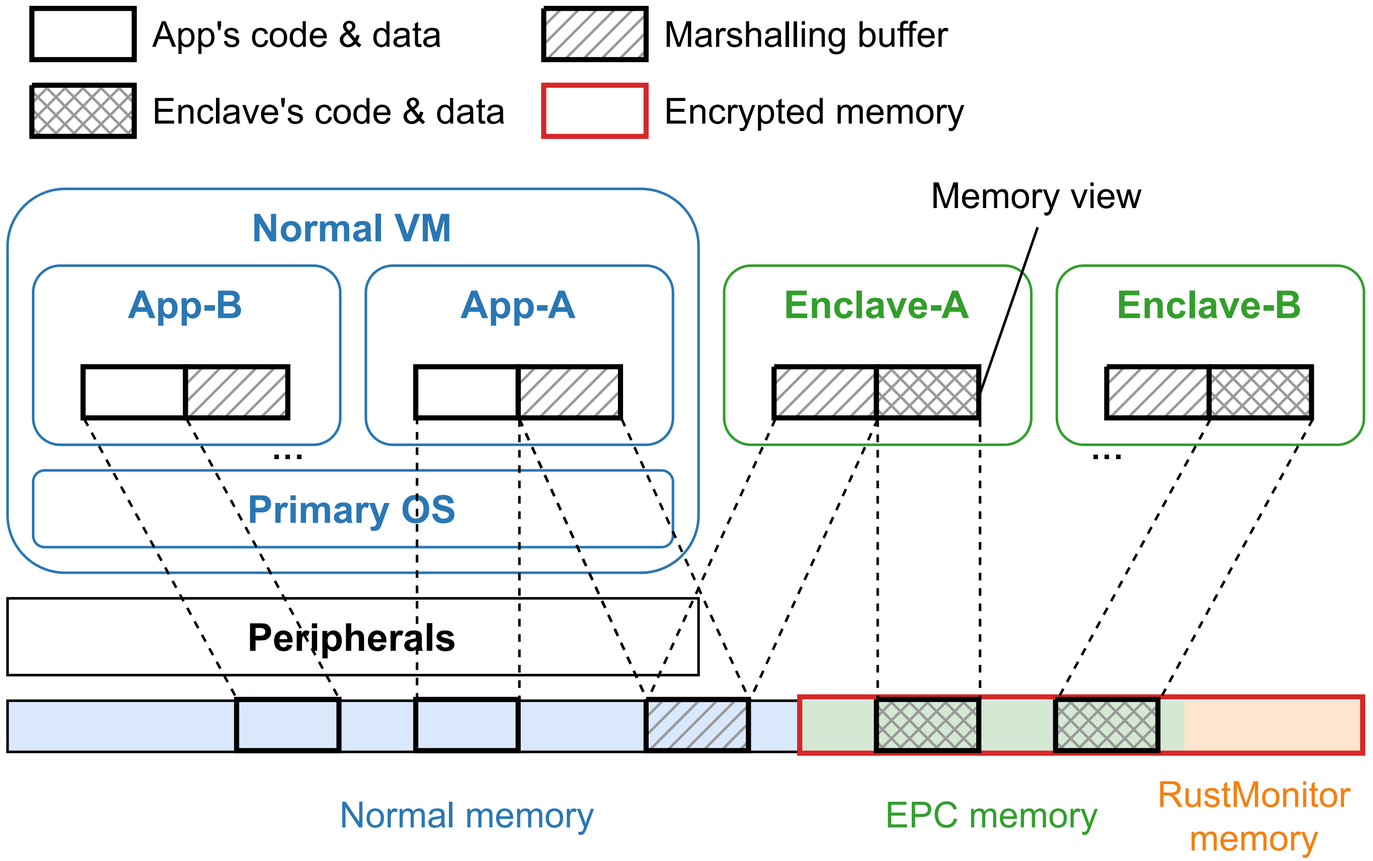
1. Memory Management and Protection

- Existing designs:
  1. Enclave's page table is managed by untrusted OS
  2. Enclave can access the application's entire address space



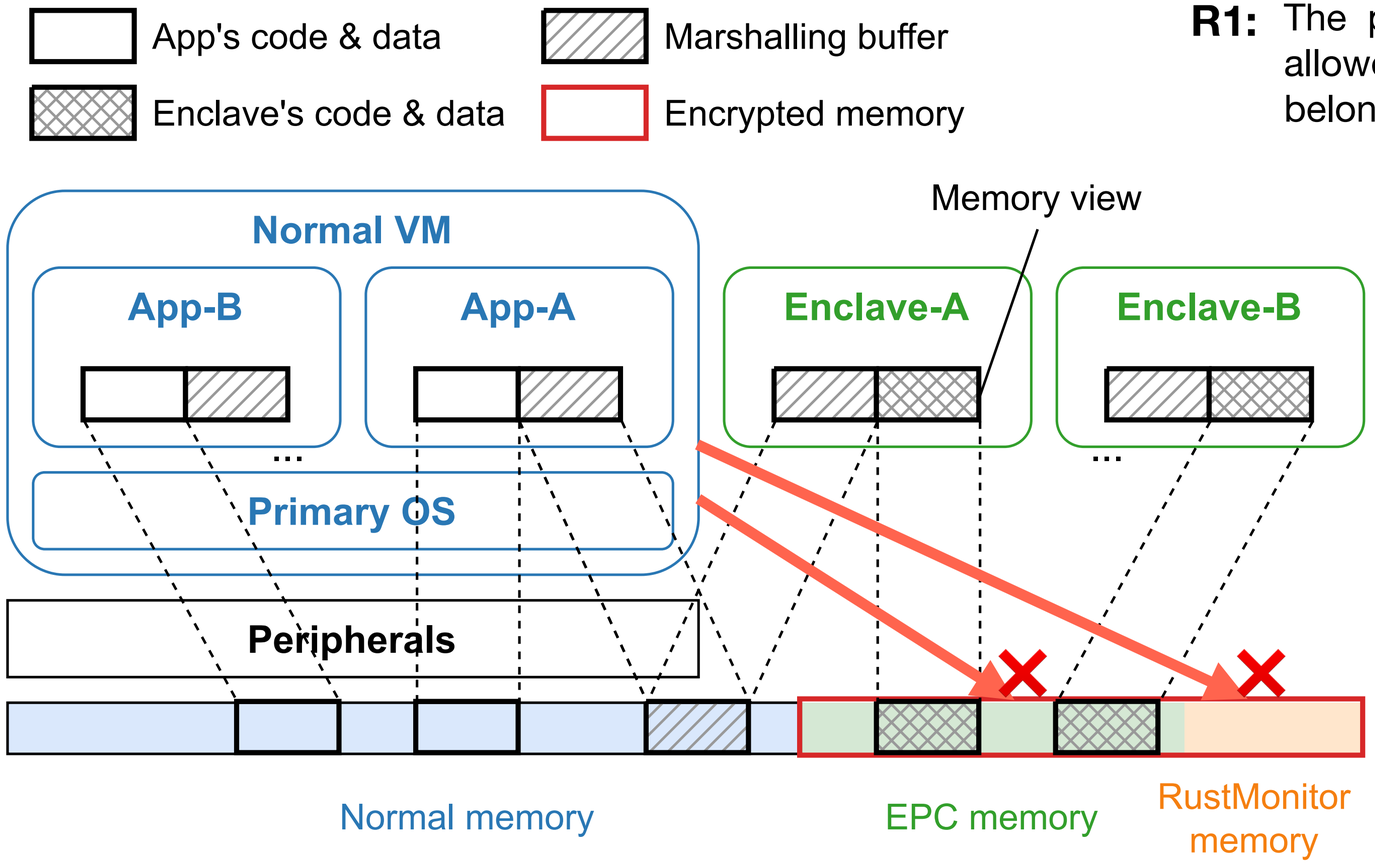
1. Memory Management and Protection

HyperEnclave memory isolation



1. Memory Management and Protection

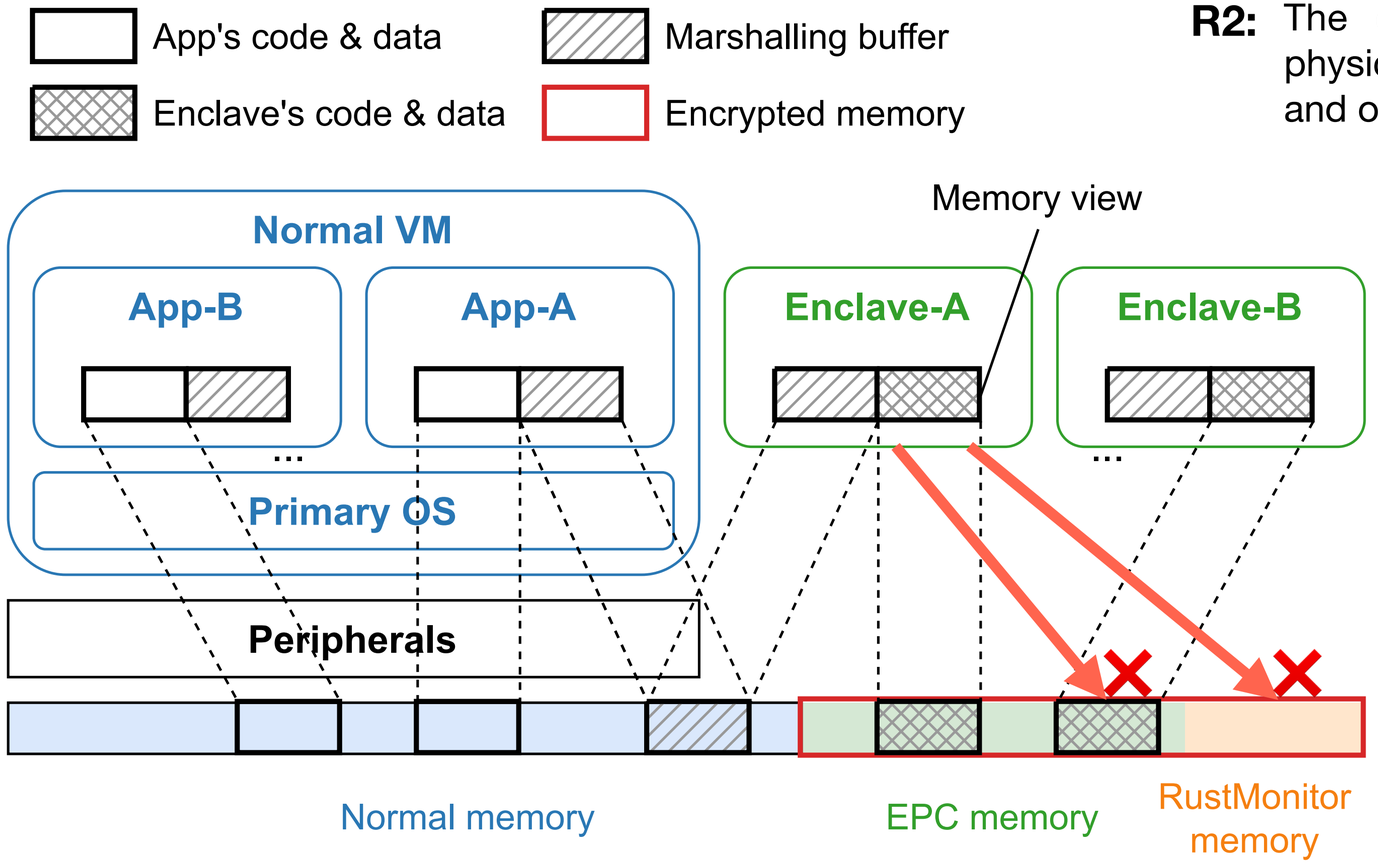
HyperEnclave memory isolation



R1: The primary OS and applications are not allowed to access the physical memory belonging to RustMonitor and the enclaves.

1. Memory Management and Protection

HyperEnclave memory isolation

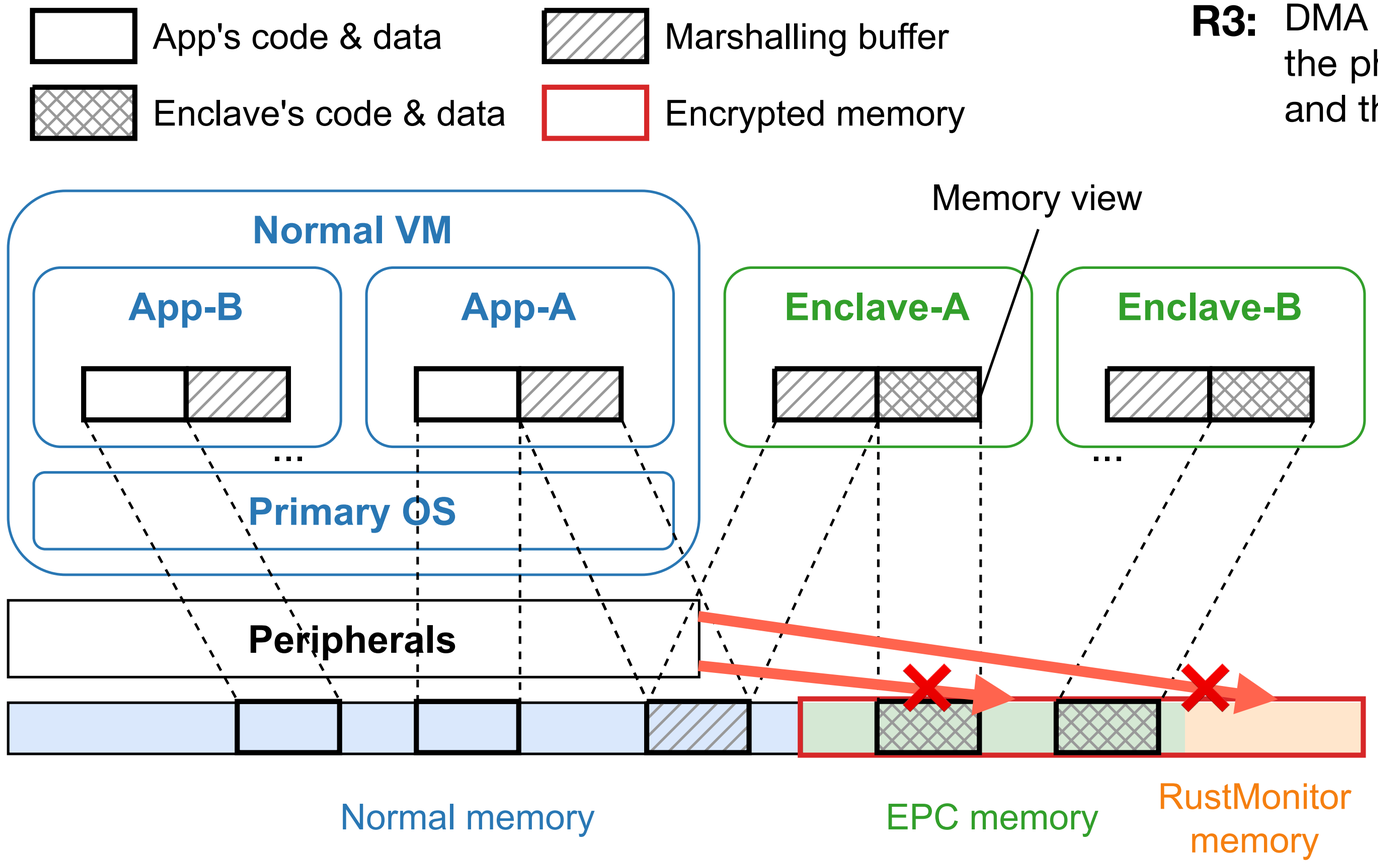


R2: The enclave is not allowed to access physical memory belonging to RustMonitor and other enclaves.

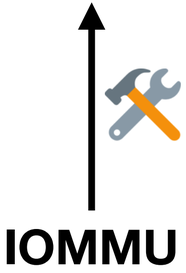
Nested paging, normal 1-level paging

1. Memory Management and Protection

HyperEnclave memory isolation

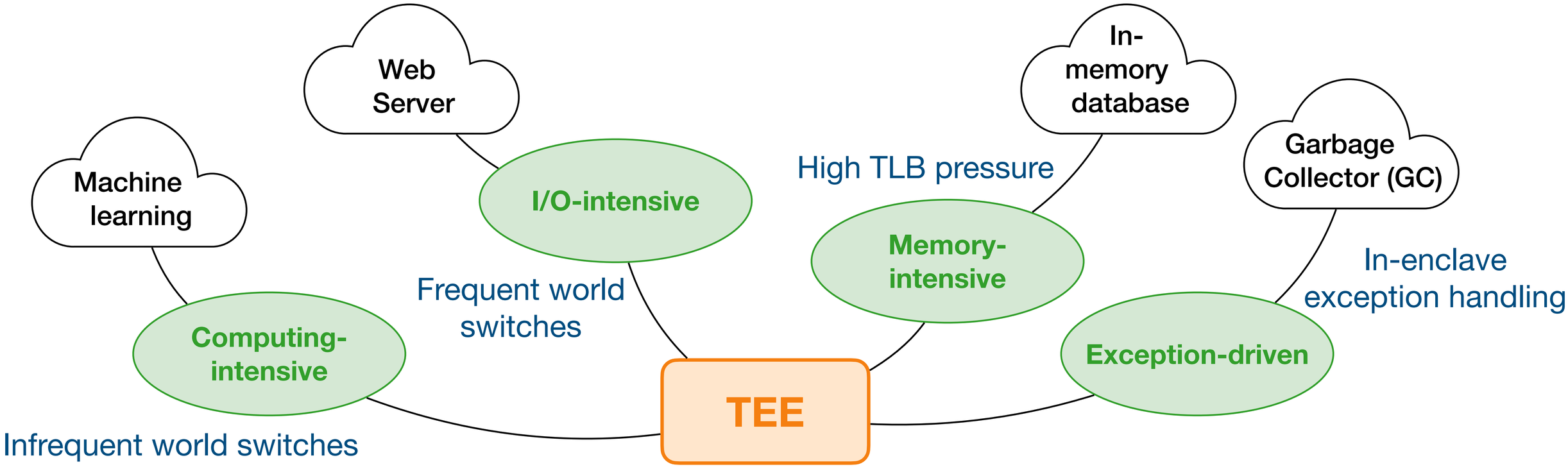


R3: DMA accesses from malicious peripherals to the physical memory belonging to RustMonitor and the enclaves are not allowed.



2. Flexible Enclave Operation Mode

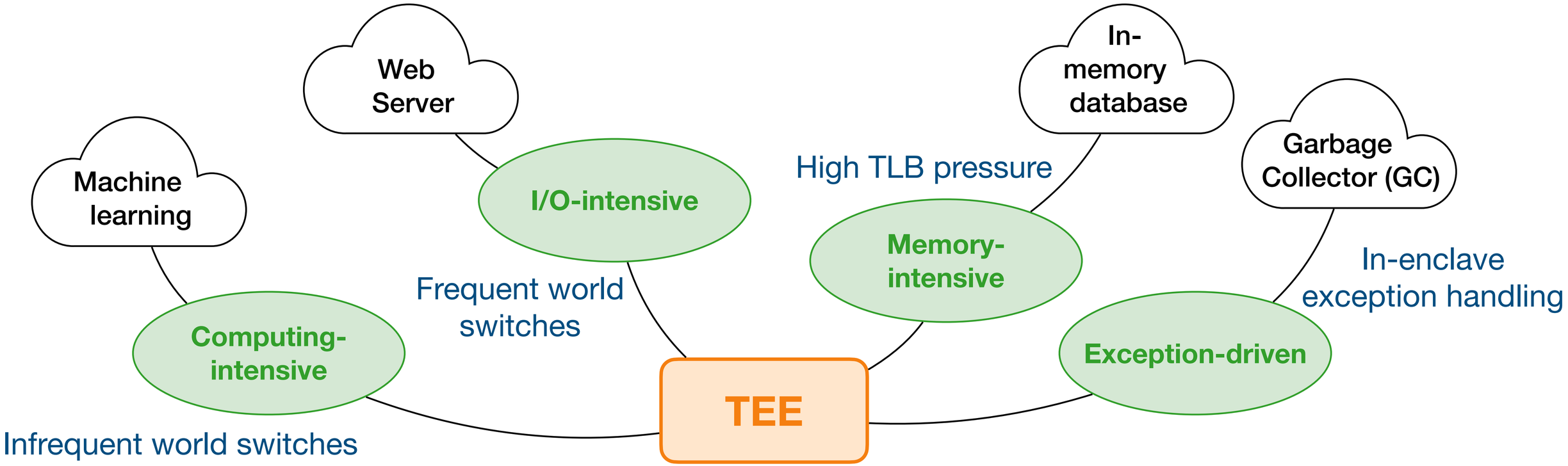
- A wide range of existing applications can be offloaded to TEEs:



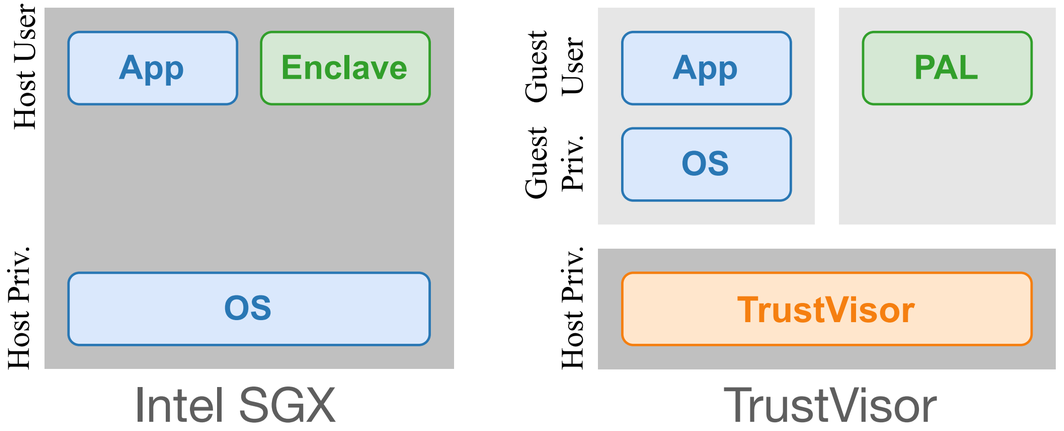
2. Flexible Enclave Operation Mode

- A wide range of existing applications can be offloaded to TEEs:

TEEs need to better fulfill the requirements for specific enclave workloads



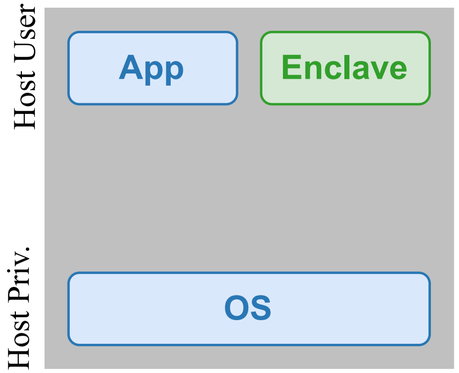
2. Flexible Enclave Operation Mode



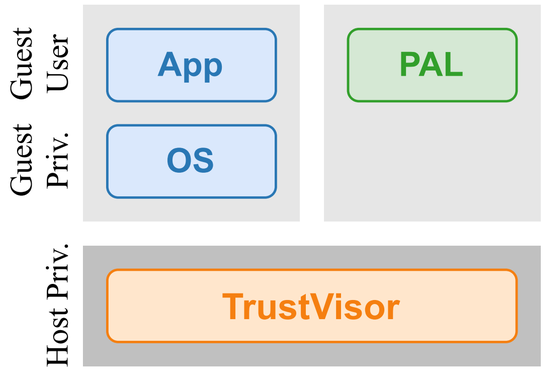
2. Flexible Enclave Operation Mode

☹ Slow to handle enclave exception

☹ Not adapt well to various enclave workloads



Intel SGX

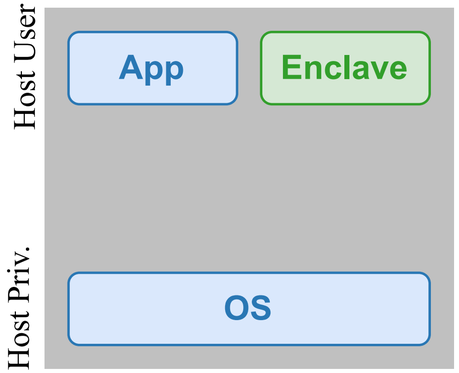


TrustVisor

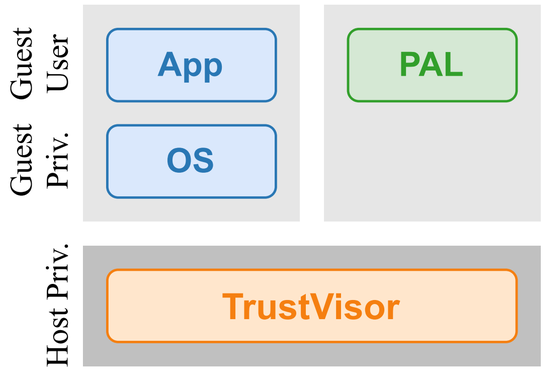
2. Flexible Enclave Operation Mode

☹ Slow to handle enclave exception

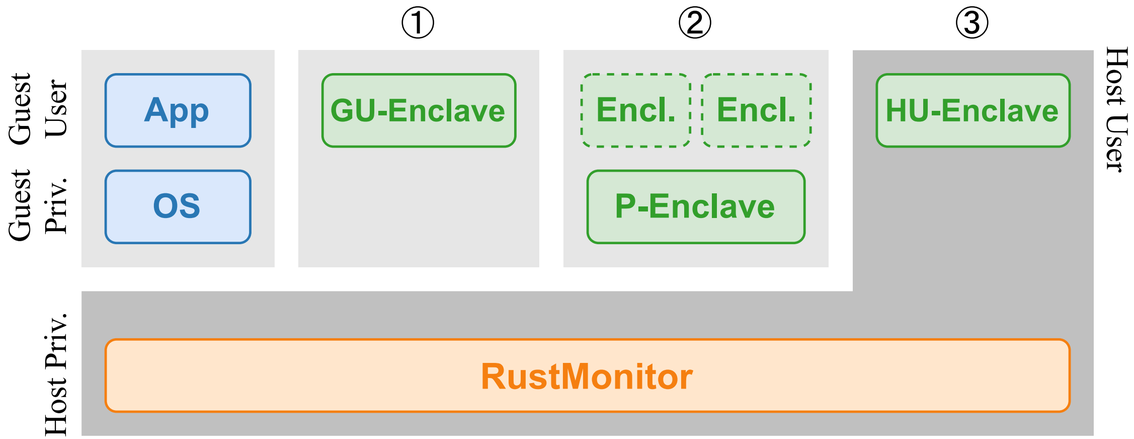
☹ Not adapt well to various enclave workloads



Intel SGX



TrustVisor



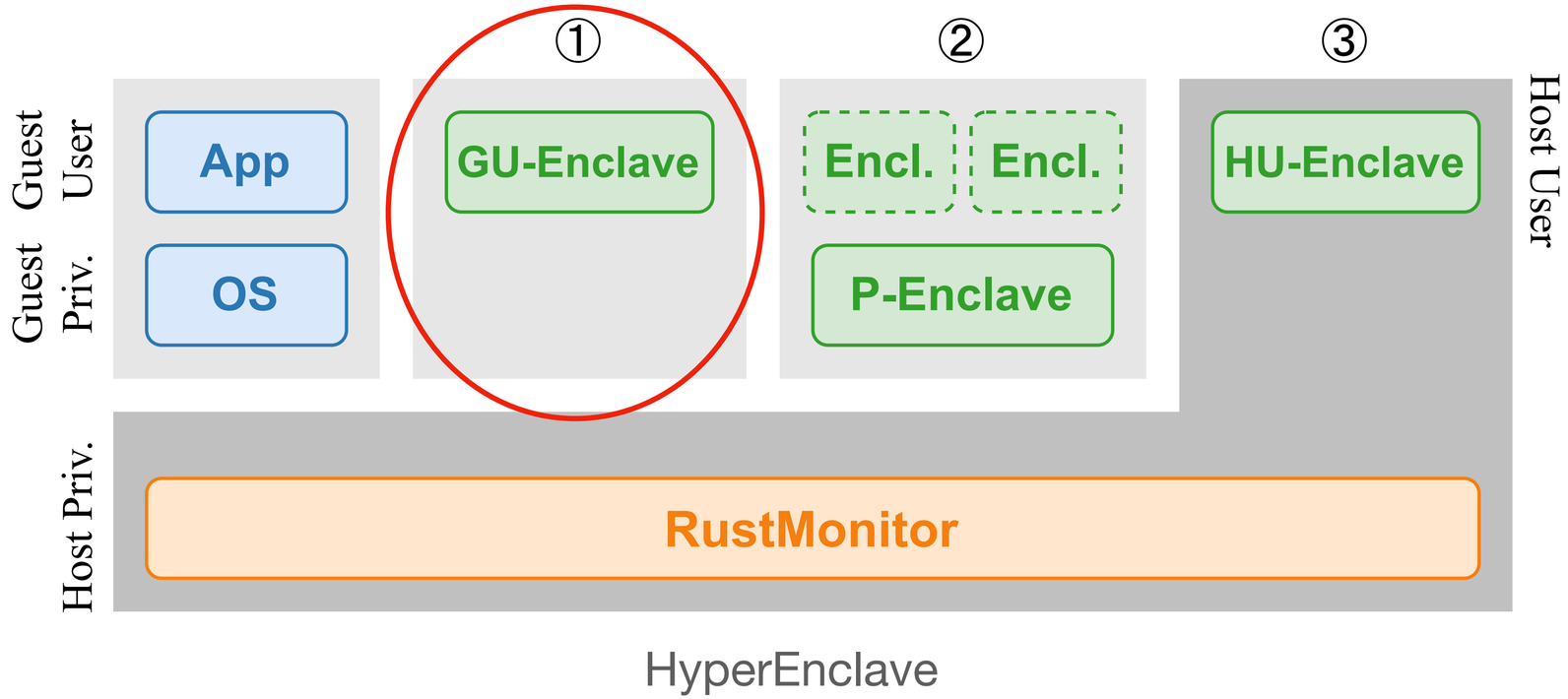
HyperEnclave

Flexible modes!

2. Flexible Enclave Operation Mode

GU-Enclave

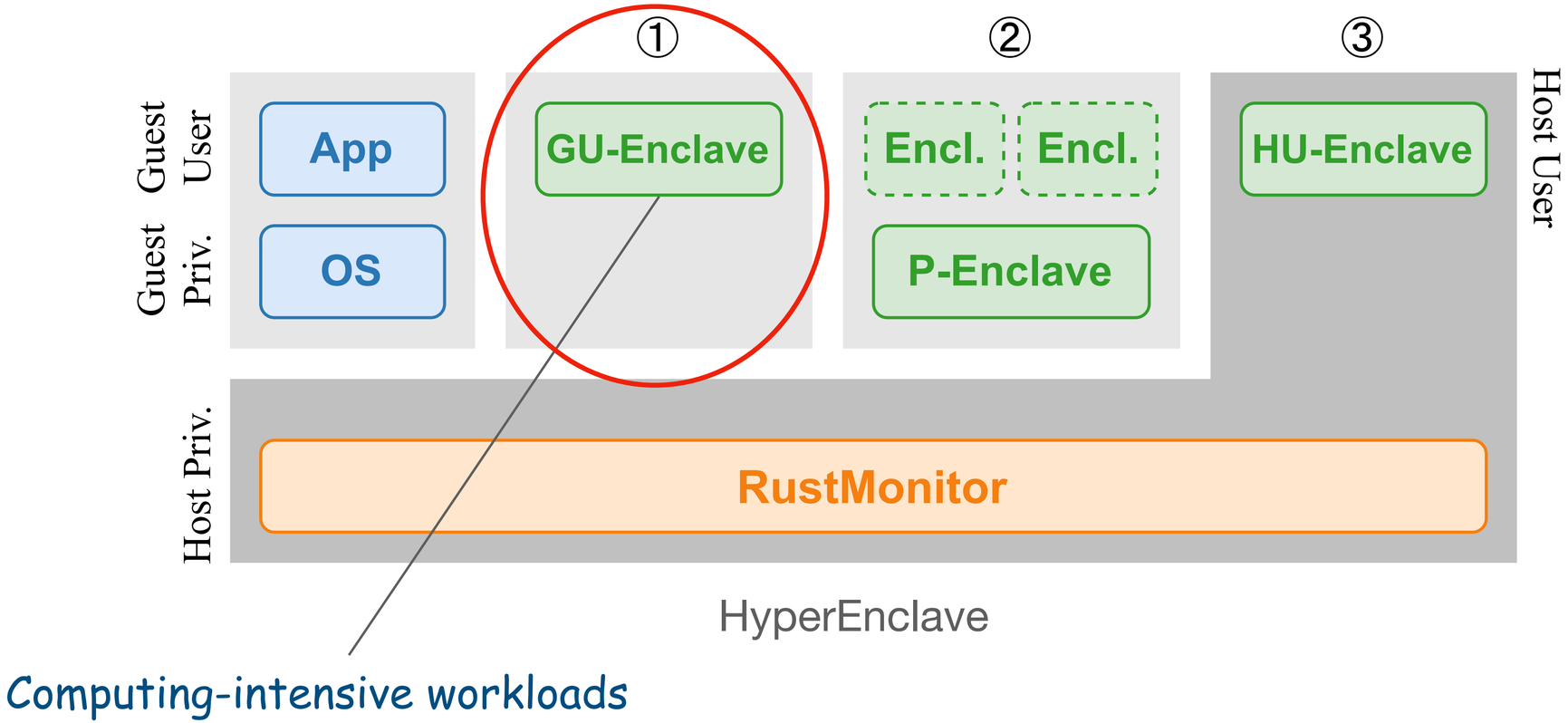
- Basic form



2. Flexible Enclave Operation Mode

GU-Enclave

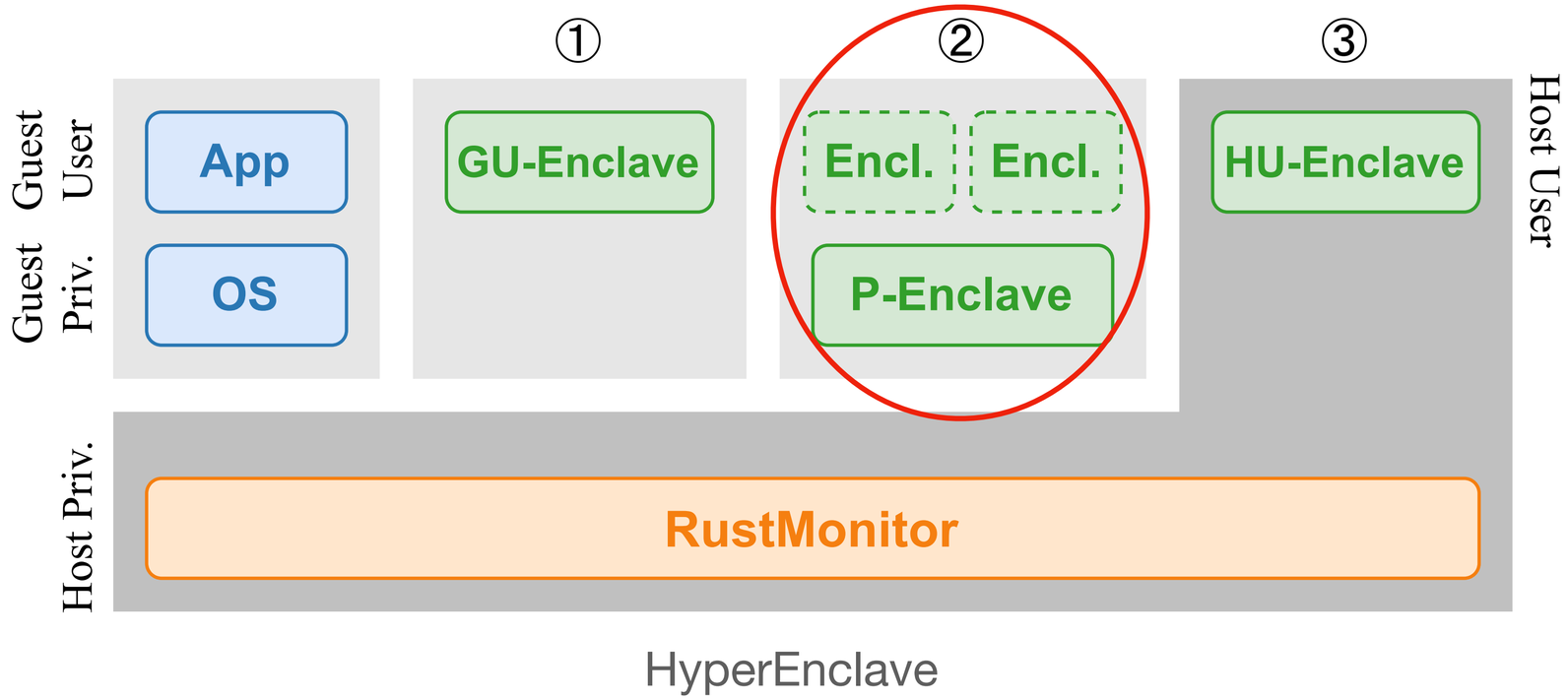
- Basic form



2. Flexible Enclave Operation Mode

P-Enclave

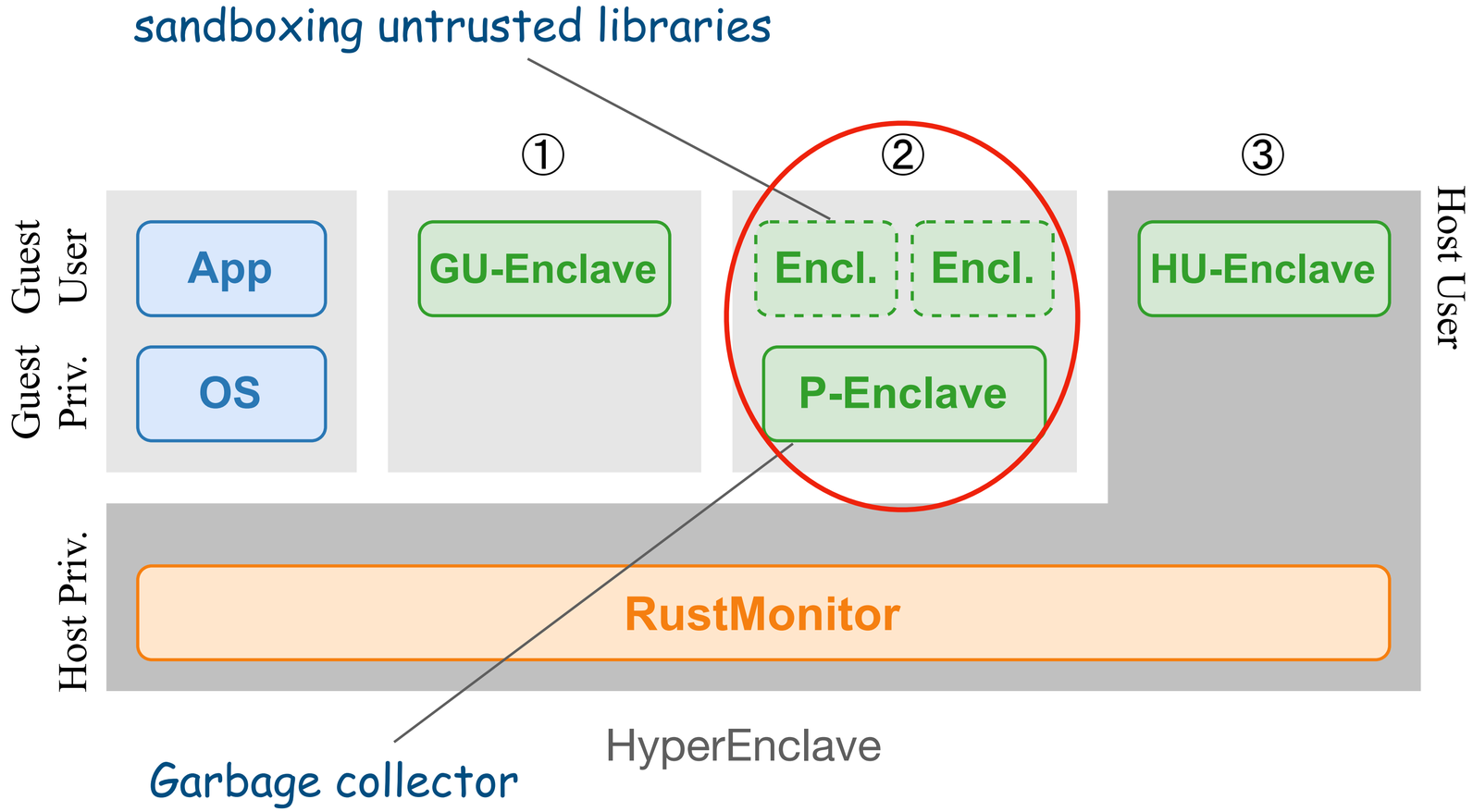
- Access privileged resources:
  - IDT
  - Page tables
- Process privileged events:
  - Interrupts
  - Exceptions



2. Flexible Enclave Operation Mode

P-Enclave

- Access privileged resources:
  - IDT
  - Page tables
- Process privileged events:
  - Interrupts
  - Exceptions



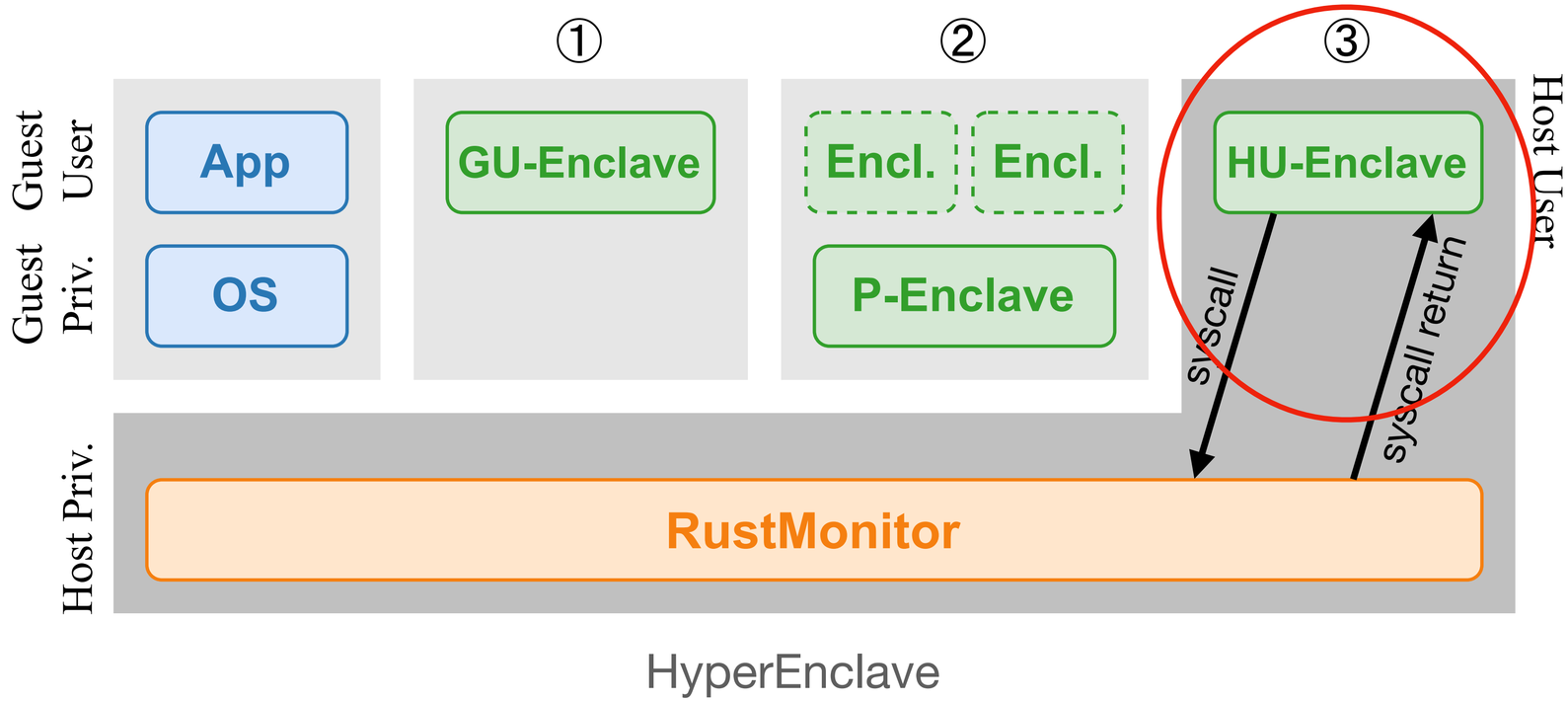
2. Flexible Enclave Operation Mode

HU-Enclave

- Fast world switches

  hypercall (~880 cycles)
  ↓
  syscall (~120 cycles)

- No virtualization overhead



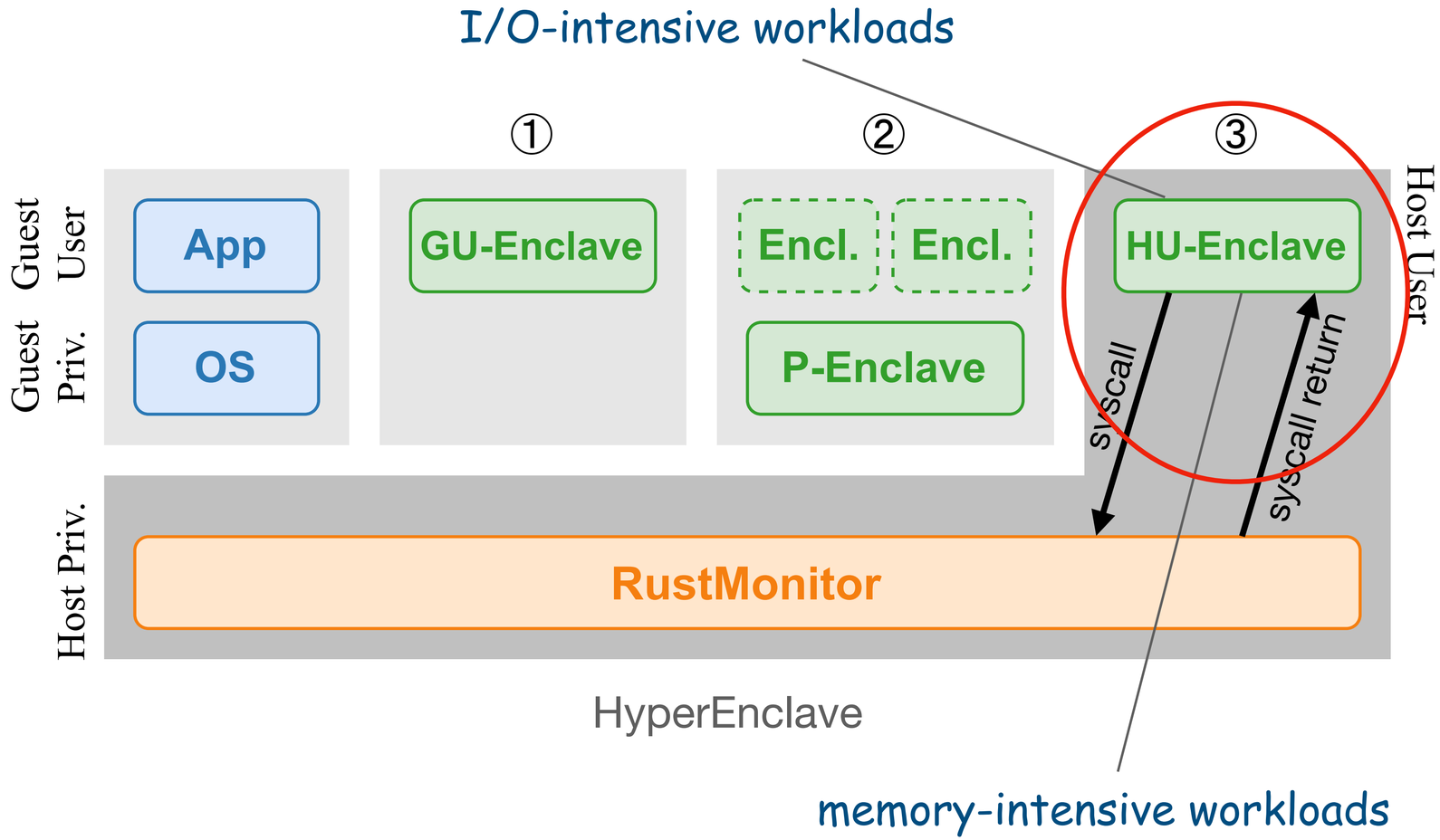
2. Flexible Enclave Operation Mode

HU-Enclave

- Fast world switches

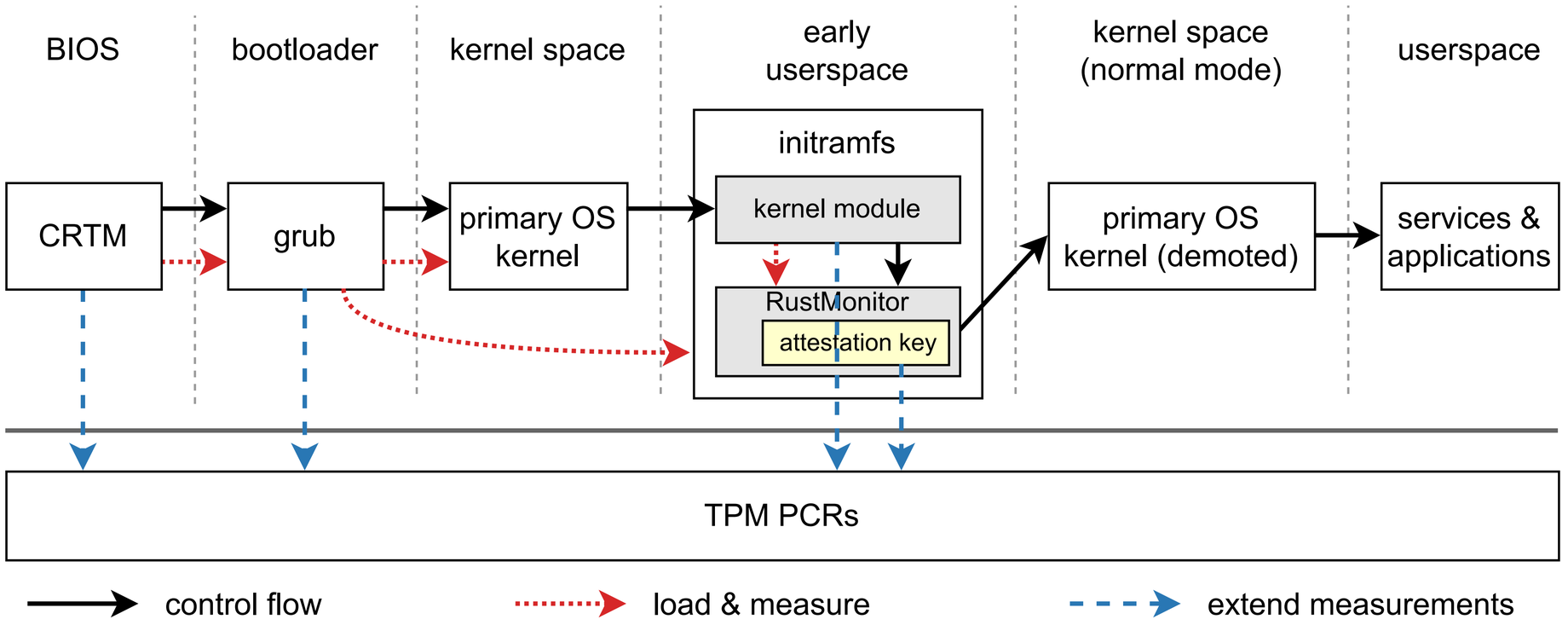
  hypercall (~880 cycles)
  ↓
  syscall (~120 cycles)

- No virtualization overhead



3. Trusted Boot

Measured Late Launch



More details about trusted boot and attestation are in the paper.

4. The Enclave SDK

- Retrofit the official SGX SDK[1]
- Supported SGX applications:
  - Rust SGX SDK[2]
  - Occulm library OS[3]
- Run **existing SGX programs** with little or no code changes

Occlum

Instruction	Leaf	Descrption	Solution
ENCLU (user)	EENTER	Enter the enclave	hypercall/ syscall
	EEXIT	Exit the enclave	
	EREPORT	Create a cryptographic report	
	…		
ENCLS (privileged)	ECREATE	Create an enclave	`ioctl()` ↓ kernel module ↓ hypercall
	EADD	Add an enclave page	
	EEXTEND	Extend enclave measurement	
	EINIT	Initialize an enclave	
	…		

API compatible!

[1] https://github.com/intel/linux-sgx
[2] https://github.com/apache/incubator-teaclave-sgx-sdk
[3] https://github.com/occlum/occlum

Implementation

- **AMD** EYPC 7601
  - Hardware virtualization (SVM)
  - Memory encryption (SME)
- Future: ARM, RISC-V, …
- **Rust**Monitor

Component	Language	LoC
RustMonitor	Rust	+7,500
Kernel module	C	+3,500
Enclave SDK	C++	+2,000

Evaluation

Methodology

- Platform A: **AMD** EPYC 7601, 512 GB RAM, with **HyperEnclave** and **SME**
- Platform B: **Intel** Xeon E3-1270 v6, 64 GB RAM, with **SGX**

Evaluation

Methodology

- Platform A: **AMD** EPYC 7601, 512 GB RAM, with **HyperEnclave** and **SME**
- Platform B: **Intel** Xeon E3-1270 v6, 64 GB RAM, with **SGX**

How to compare performance on the two different platforms?

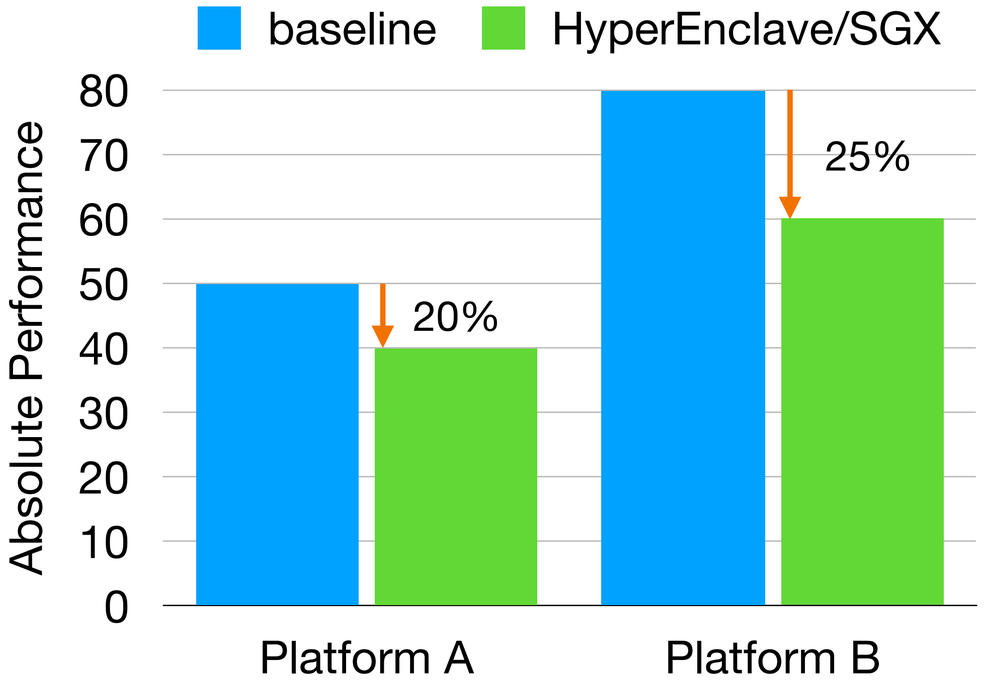
Evaluation

Methodology

- Platform A: **AMD** EPYC 7601, 512 GB RAM, with **HyperEnclave** and **SME**
- Platform B: **Intel** Xeon E3-1270 v6, 64 GB RAM, with **SGX**

How to compare performance on the two different platforms?

Relative slowdown ↓



Baseline: SDK simulation mode (no security protections)

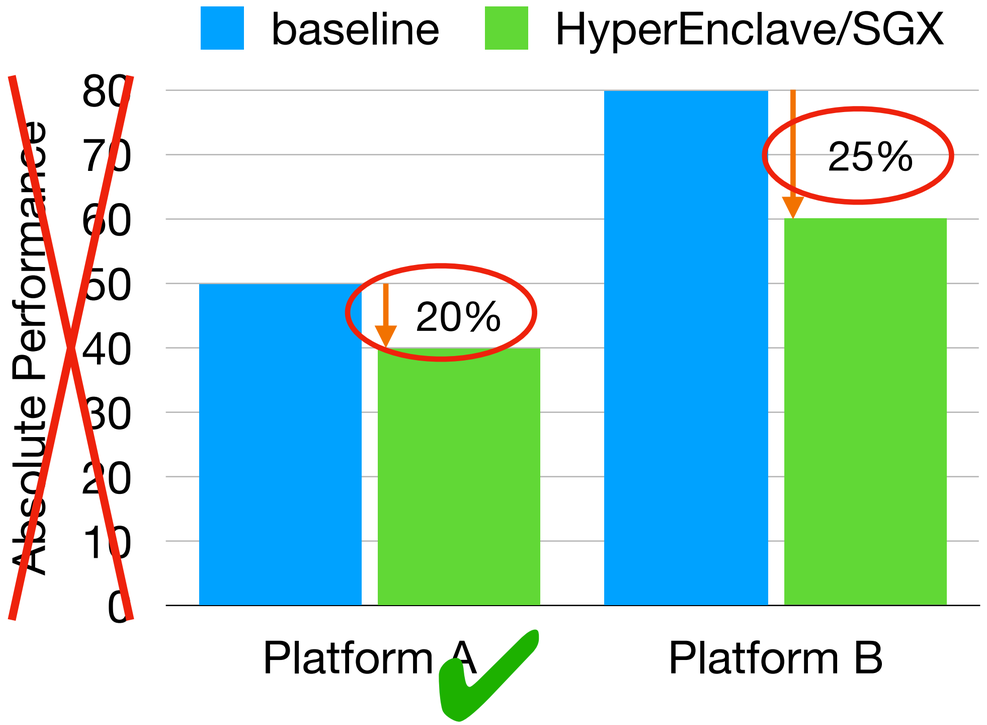
Evaluation

Methodology

- Platform A: **AMD** EPYC 7601, 512 GB RAM, with **HyperEnclave** and **SME**
- Platform B: **Intel** Xeon E3-1270 v6, 64 GB RAM, with **SGX**

How to compare performance on the two different platforms?

Relative slowdown ↓



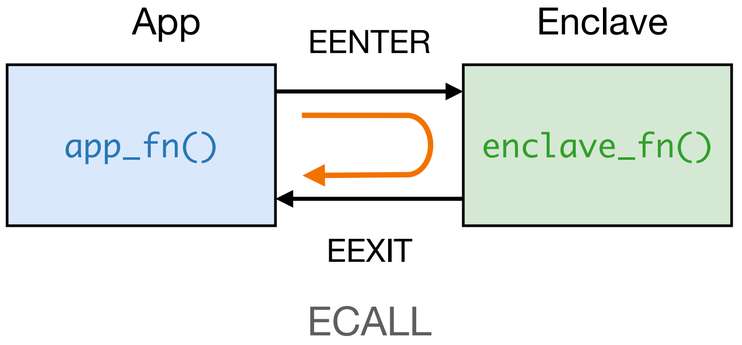
Baseline: SDK simulation mode (no security protections)

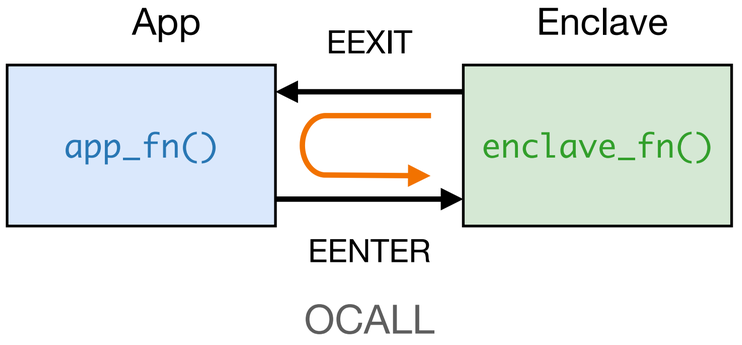
Evaluation

World switches performance

	EENTER	EEXIT	ECALL	OCALL
Intel SGX	—[1]	—	14,432	12,432
GU-Enclave	1,704	1,319	9,480	4,920
HU-Enclave	1,163	1,144	8,440	4,120
P-Enclave	1,649	1,401	9,700	5,260

Latency of SGX primitives on HyperEnclave and Intel (in CPU cycles)

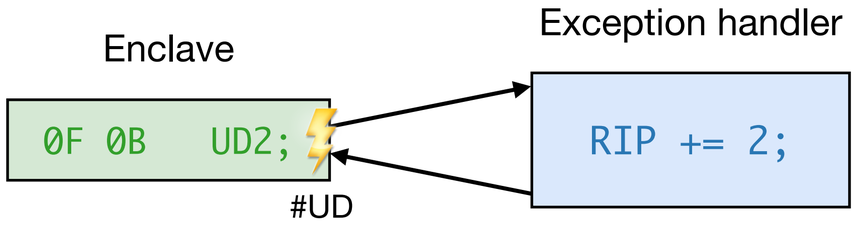




[1]We are unable to measure the instruction latencies on the SGX hardware since the RDTSCP instruction is not supported in the SGX enclaves.

Evaluation

P-Enclave use cases: exception handling

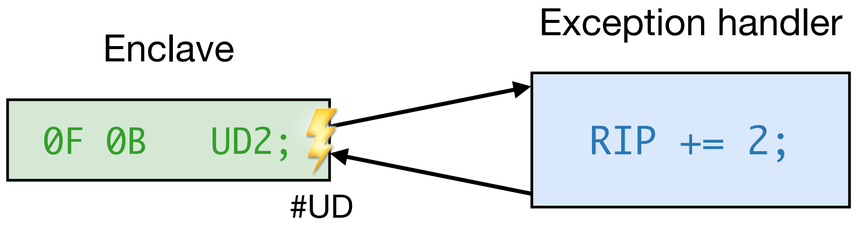


	Intel SGX	GU-Enclave	P-Enclave
#UD	28,561	17,490	258

Average CPU cycles of handling an #UD exception inside the enclaves

Evaluation

P-Enclave use cases: exception handling



Two-phase exception handling[1]

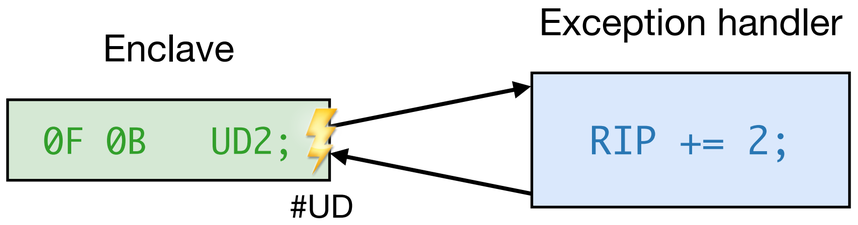
	Intel SGX	GU-Enclave	P-Enclave
#UD	28,561	17,490	258

Average CPU cycles of handling an #UD exception inside the enclaves

[1] https://github.com/MWShan/linux-sgx/blob/master/docs/DesignDocs/IntelSGXExceptionHandling-Linux.md

Evaluation

P-Enclave use cases: exception handling



Handled in the enclave!

Two-phase exception handling[1]

	Intel SGX	GU-Enclave	P-Enclave
#UD	28,561	17,490	258

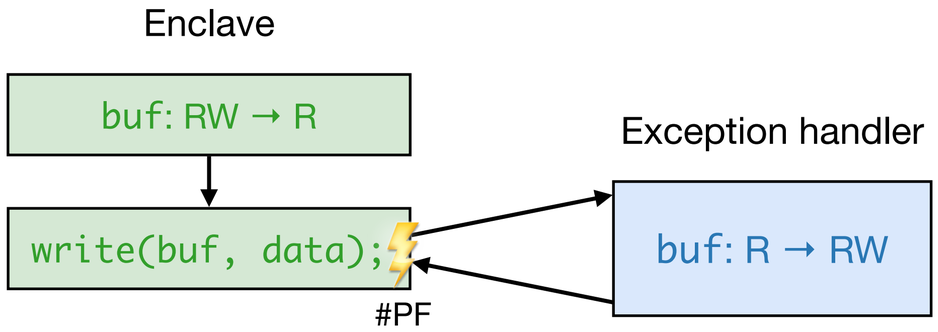
68x

Average CPU cycles of handling an #UD exception inside the enclaves

[1] https://github.com/MWShan/linux-sgx/blob/master/docs/DesignDocs/IntelSGXExceptionHandling-Linux.md

Evaluation

P-Enclave use cases: exception handling



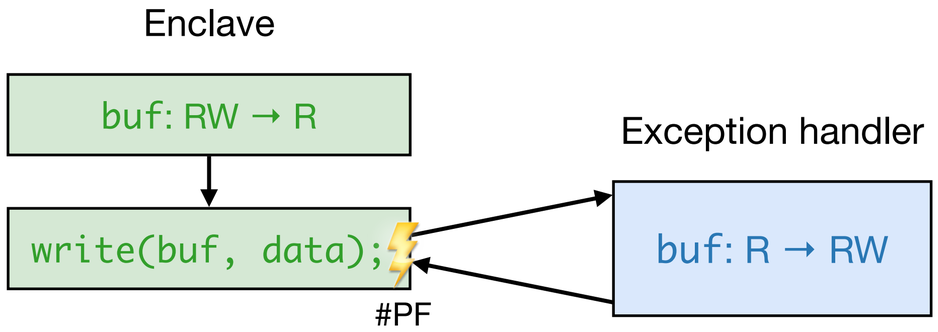
	Intel SGX	GU-Enclave	P-Enclave
#PF	—[1]	2,660	1,132

Average CPU cycles of handling a #PF exception inside the enclaves

[1]Our SGX1 platform does not support page permission modifications.

Evaluation

P-Enclave use cases: exception handling



Trap into RustMonitor

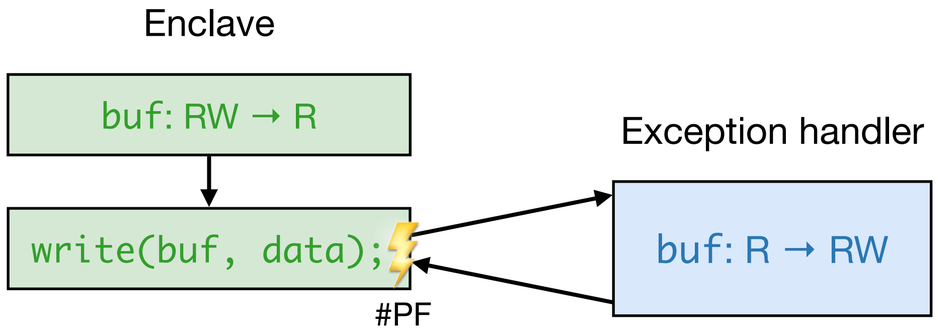
	Intel SGX	GU-Enclave	P-Enclave
#PF	—[1]	2,660	1,132

Average CPU cycles of handling a #PF exception inside the enclaves

[1]Our SGX1 platform does not support page permission modifications.

Evaluation

P-Enclave use cases: exception handling



Handled in the enclave!

Trap into RustMonitor

	Intel SGX	GU-Enclave	P-Enclave
#PF	—[1]	2,660	1,132

2.3x

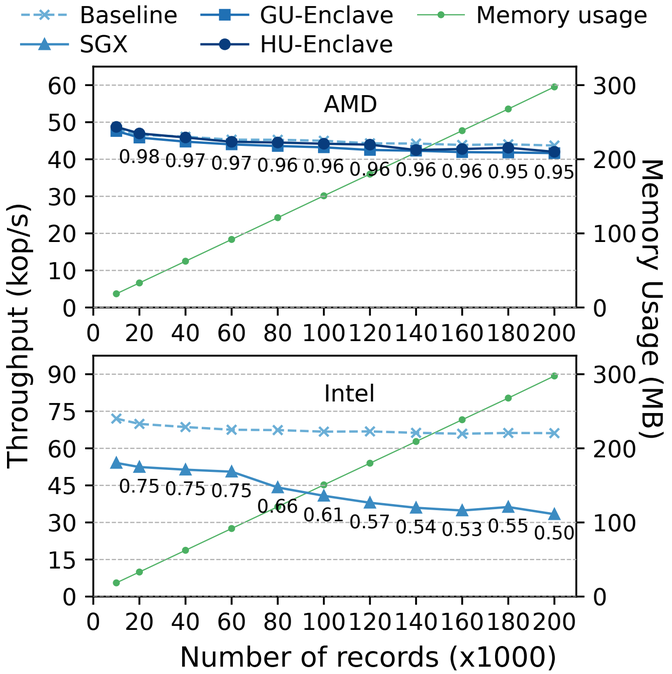
Average CPU cycles of handling a #PF exception inside the enclaves

[1]Our SGX1 platform does not support page permission modifications.

Evaluation

Real-world workloads

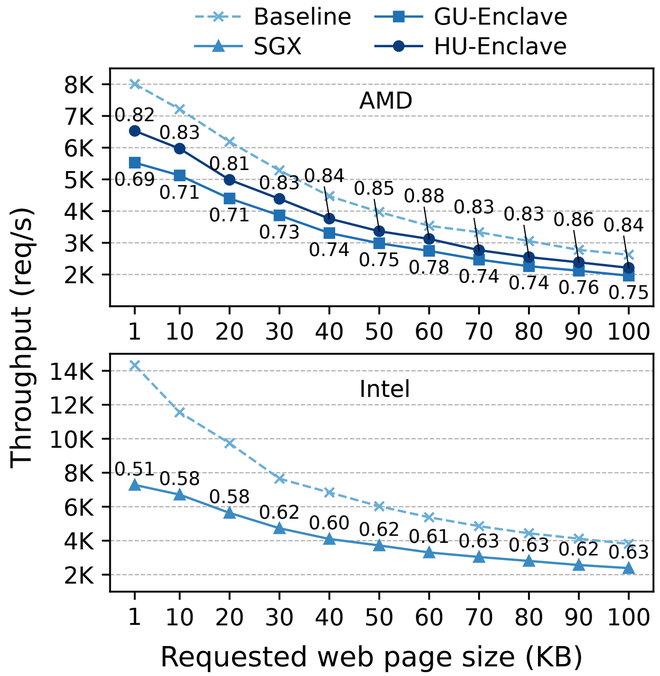
- SQLite (memory-intensive)
  - < **5%↓**
  - Support large EPC size to avoid page swapping

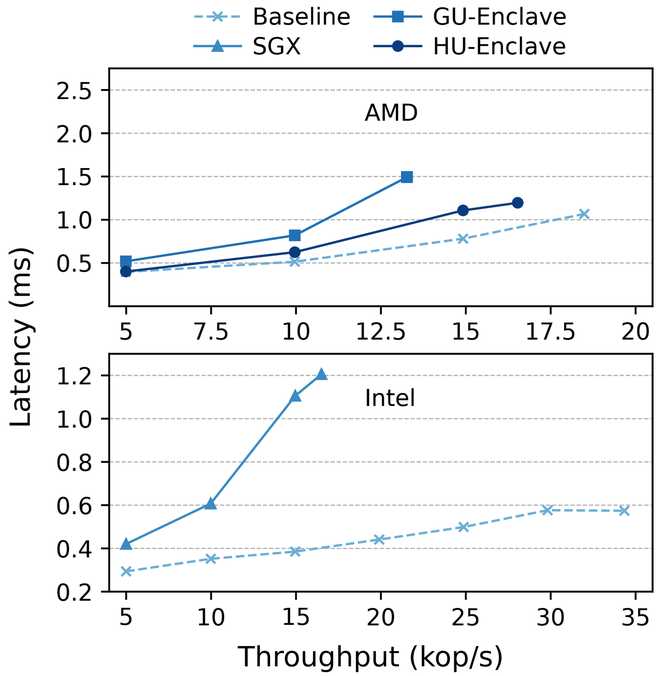


Evaluation

Real-world workloads

- Lighttpd (I/O-intensive)
  - GU-Enclave: 22%↓
  - HU-Enclave: **12%**↓
- Redis (I/O and memory intensive)
  - GU-Enclave: 28%↓
  - HU-Enclave: **11%**↓



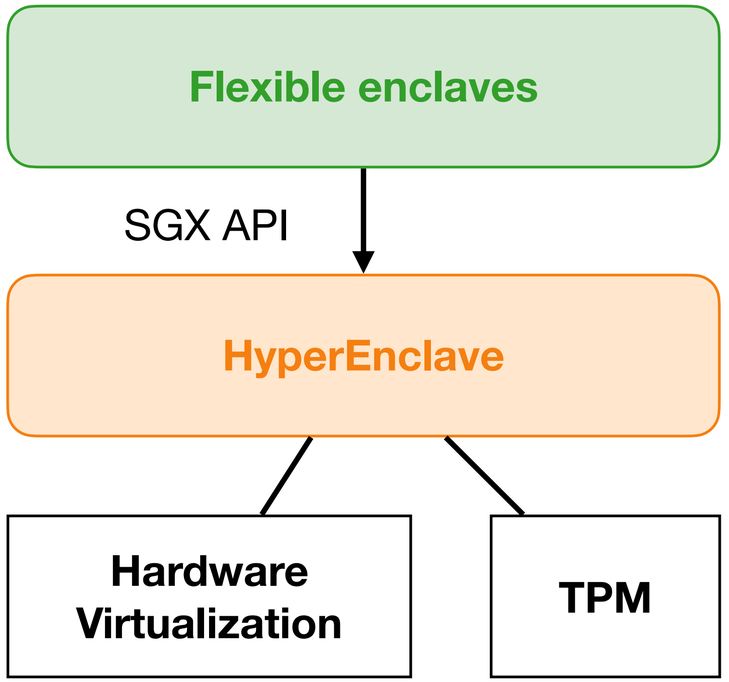


LIGHTTPD fly light. + Occlum

redis + Occlum

Conclusion

- **An open process-based TEE**
  - Minimum hardware requirements
- **SGX compatible**
  - Run existing SGX programs with little or no code changes
- **Flexible enclave modes**
  - Better to fulfill various enclave workloads



Thanks!

- Source code will be available at (still in progress):

https://github.com/HyperEnclave

- Contact us:
  - **Yuekai Jia**, equation618@gmail.com
  - Shuang Liu, ls123674@antgroup.com
  - Wenhao Wang✉, wangwenhao@iie.ac.cn