A case for Unifying Trusted Execution Environments

Thomas Prévost

Pierre-Louis Aublin
Hajime Tazaki

Kuniyasu Suzaki

UNIVERSITÉ Clermont Auvergne

Internet Initiative Japan

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

HAP ‘26 workshop, Charlotte, NC, 22 June 2026

Comparing Trusted Execution Environments

TEE	Client SGX [13]	Scalable SGX [15]	ARM TrustZone [25]	AMD SEV [19]	Intel TDX [5]	Penglai [8]

TABLE I: Non-exhaustive comparison of various TEE implementations. Circles depict full (●), partial (◐) or no support (○).

Comparing Trusted Execution Environments

TEE	Client SGX [13]	Scalable SGX [15]	ARM TrustZone [25]	AMD SEV [19]	Intel TDX [5]	Penglai [8]
Architecture	x86	x86	ARM	x86	x86	RISC-V
Target	Client CPU	Server CPU	Various	Server CPU	Server CPU	Various
Isolation granularity	Process	Process	Process	VM	VM	Process

TABLE I: Non-exhaustive comparison of various TEE implementations. Circles depict full (●), partial (◐) or no support (○).

Comparing Trusted Execution Environments

TEE	Client SGX [13]	Scalable SGX [15]	ARM TrustZone [25]	AMD SEV [19]	Intel TDX [5]	Penglai [8]
Confidentiality	●	●	●	●	●	●
Integrity	●	●	◐	○	●	●
Freshness	●	○	○	○	○	●

TABLE I: Non-exhaustive comparison of various TEE implementations. Circles depict full (●), partial (◐) or no support (○).

Comparing Trusted Execution Environments

TEE	Client SGX [13]	Scalable SGX [15]	ARM TrustZone [25]	AMD SEV [19]	Intel TDX [5]	Penglai [8]
Confidentiality	●	●	●	●	●	●
Integrity	●	●	◐	○	●	●
Freshness	●	○	○	○	○	●
Secure memory size	256MB	512GB	All DRAM	All DRAM	All DRAM	256MB

TABLE I: Non-exhaustive comparison of various TEE implementations. Circles depict full (●), partial (◐) or no support (○).

Comparing Trusted Execution Environments

TEE	Client SGX [13]	Scalable SGX [15]	ARM TrustZone [25]	AMD SEV [19]	Intel TDX [5]	Penglai [8]
Local attestation	●	●	◐	○	●	●
Remote attestation	●	●	◐	●	●	○

TABLE I: Non-exhaustive comparison of various TEE implementations. Circles depict full (●), partial (◐) or no support (○).

Comparing Trusted Execution Environments

TEE	Client SGX [13]	Scalable SGX [15]	ARM TrustZone [25]	AMD SEV [19]	Intel TDX [5]	Penglai [8]
Architecture	x86	x86	ARM	x86	x86	RISC-V
Target					PU	Various
Isolation granularity						Process
Confidentiality						●
Integrity						●
Freshness						●
Secure memory size	256MB	512GB	All DRAM	All DRAM	All DRAM	256MB
Local attestation	●	●	◐	○	●	●
Remote attestation	●	●	◐	●	●	○

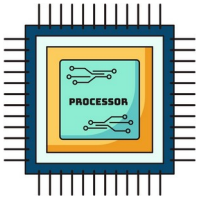
Q1: Which TEE hardware should I use?

TABLE I: Non-exhaustive comparison of various TEE implementations. Circles depict full (●), partial (◐) or no support (○).

Forest of TEE Frameworks

App

TEE Framework



SGX

TrustZone

SEV-SNP

RISC-V

Forest of TEE Frameworks

App

SGX-LKL

SGX-SDK

NestedSGX

HyperEnclave

Penglai

SCONE

OP-TE

Occlum

Q2: What programming model should I use

Q3: Where will my application run?

SGX

TrustZone

SEV-SNP

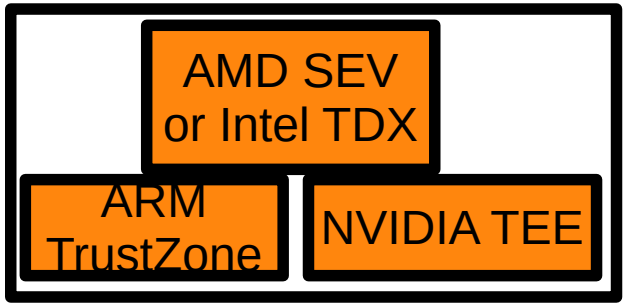
RISC-V

Problem

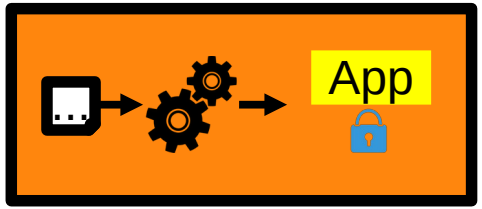
- Current TEE landscape is heterogeneous
  - Implementations offer different security guarantees and architectural features
- TEE frameworks:
  - Constrained to a few specific architectures
  - Miss critical components

Limits TEE applicability and effectiveness in heterogeneous environments

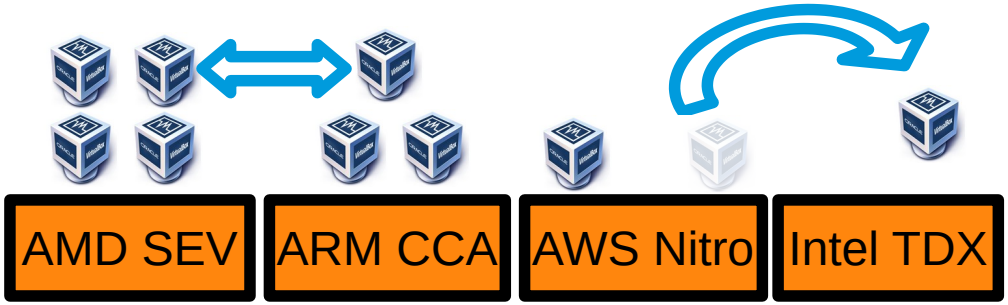
Heterogeneous TEE Platforms are Already Here



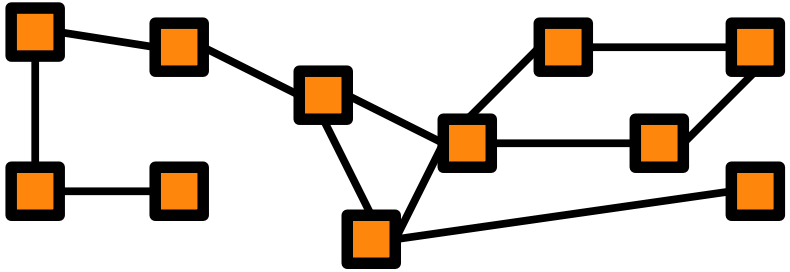
NVIDIA Confidential Computing



Attested Build



Confidential Cloud Computing



P2P Applications (e.g., Blockchain)

Requirements for a Framework to Unify TEEs

- Compatibility with all TEEs
  - Present and Future
- Awareness of each TEE characteristics
  - “Cannot execute this app on this TEE without lowering security”
- Small Trusted Computing Base
  - Smaller TCB => Smaller Attack Surface
- Formal Verification
  - Correctness and security guarantees

FUTEE: Framework for Unifying TEEs

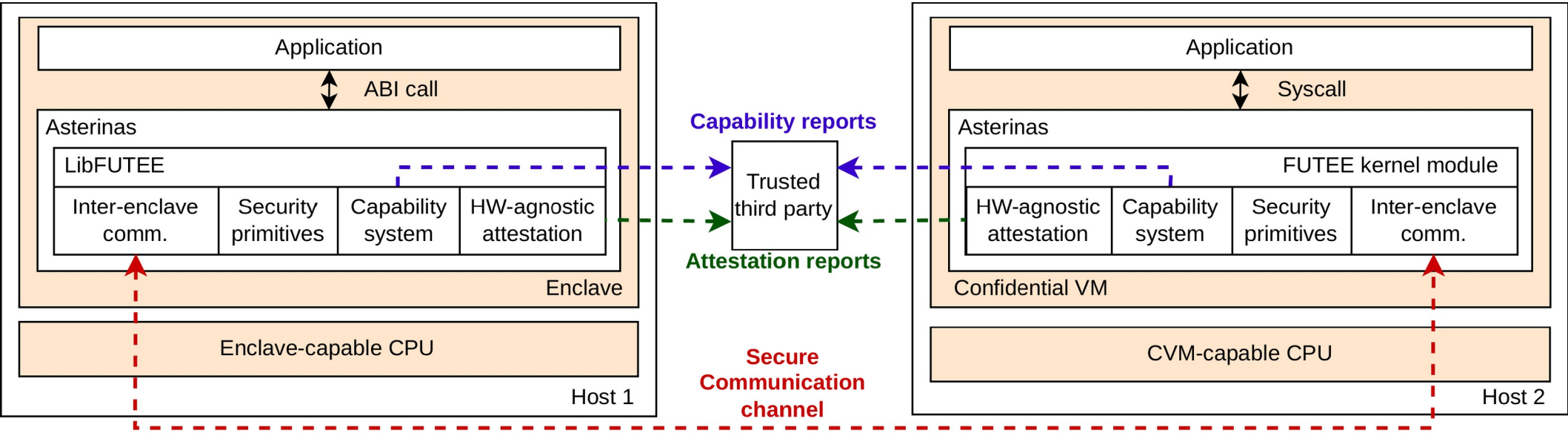


Fig. 1: Two secure applications communicate between each others via FUTEE. Dashed lines represent message exchange.

FUTEE: Framework for Unifying TEEs

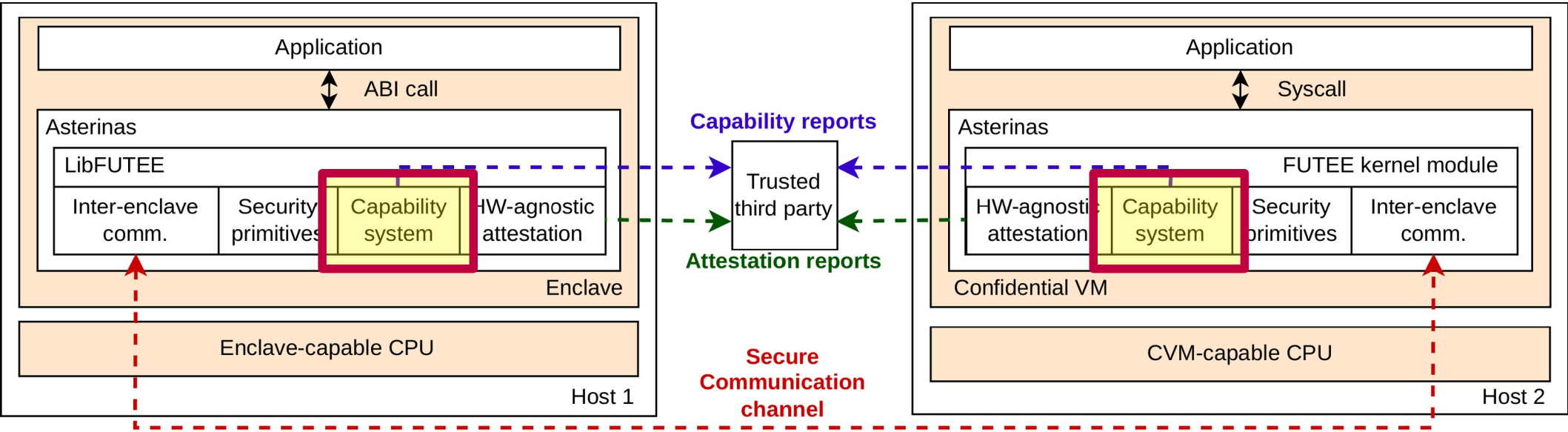


Fig. 1: Two secure applications communicate between each others via FUTEE. Dashed lines represent message exchange.

FUTEE: Framework for Unifying TEEs

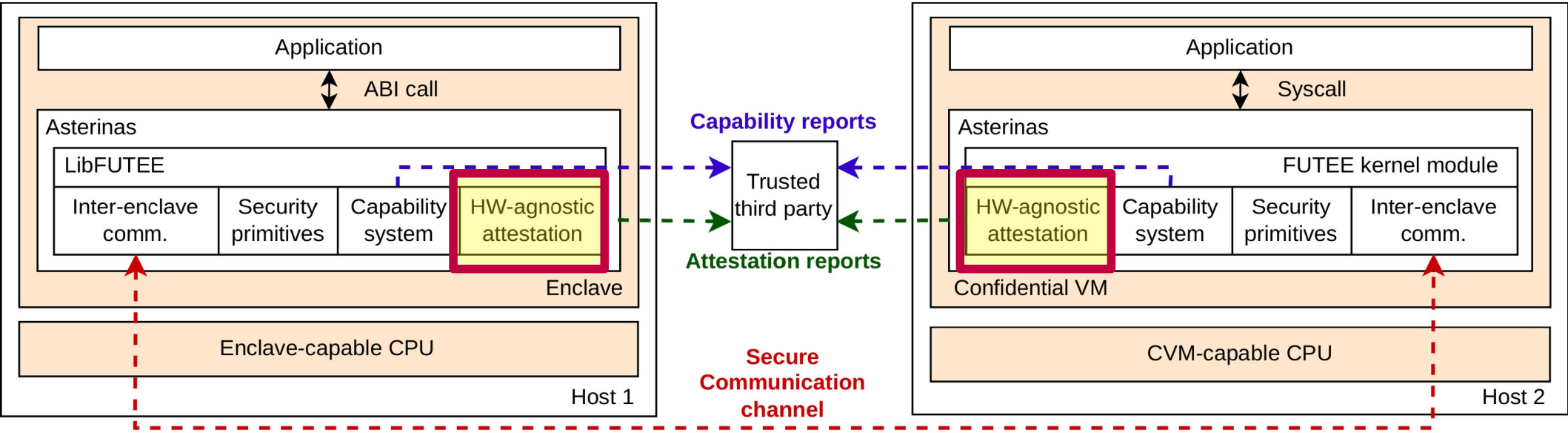


Fig. 1: Two secure applications communicate between each others via FUTEE. Dashed lines represent message exchange.

FUTEE: Framework for Unifying TEEs

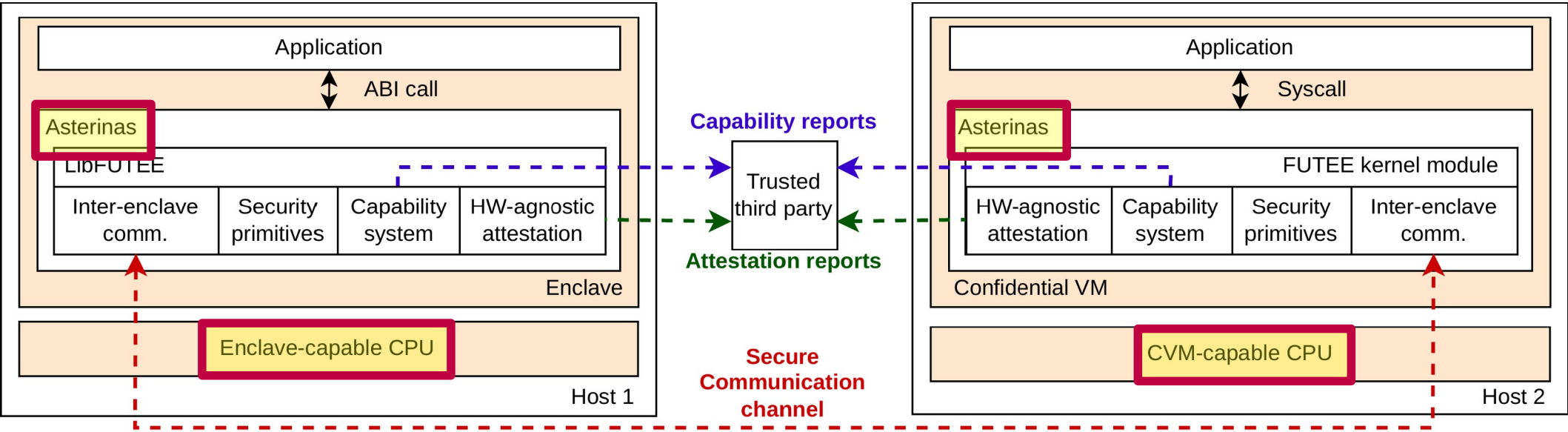


Fig. 1: Two secure applications communicate between each others via FUTEE. Dashed lines represent message exchange.

FUTEE: Framework for Unifying TEEs

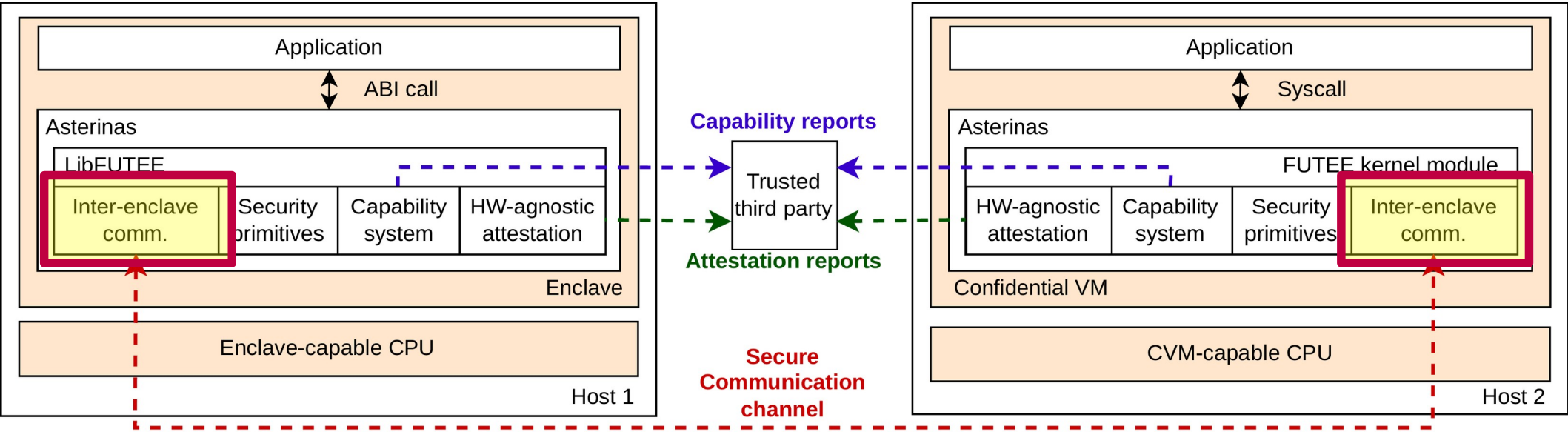


Fig. 1: Two secure applications communicate between each others via FUTEE. Dashed lines represent message exchange.

Capability System

- Augment attestation report with this TEE functionalities
  - “CPUID for TEEs”
  - E.g., secure memory size, monotonic counters
- Is my application running in a TEE?
  - With X and Y guarantees?
- Can I execute my application on a different kind of TEE?

Vendor-Agnostic Attestation

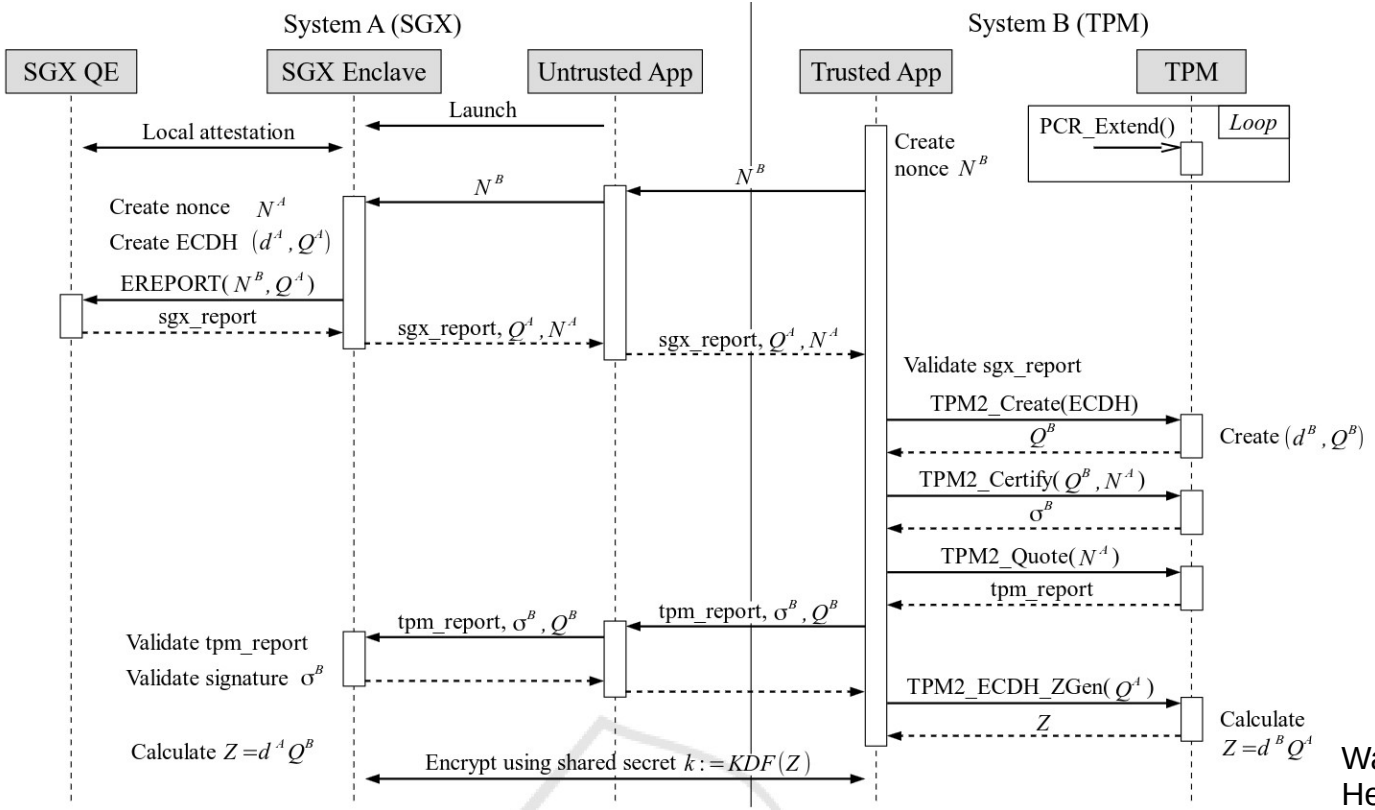


Figure 1: Remote attestation mechanism between SGX enclaves and TPMs.

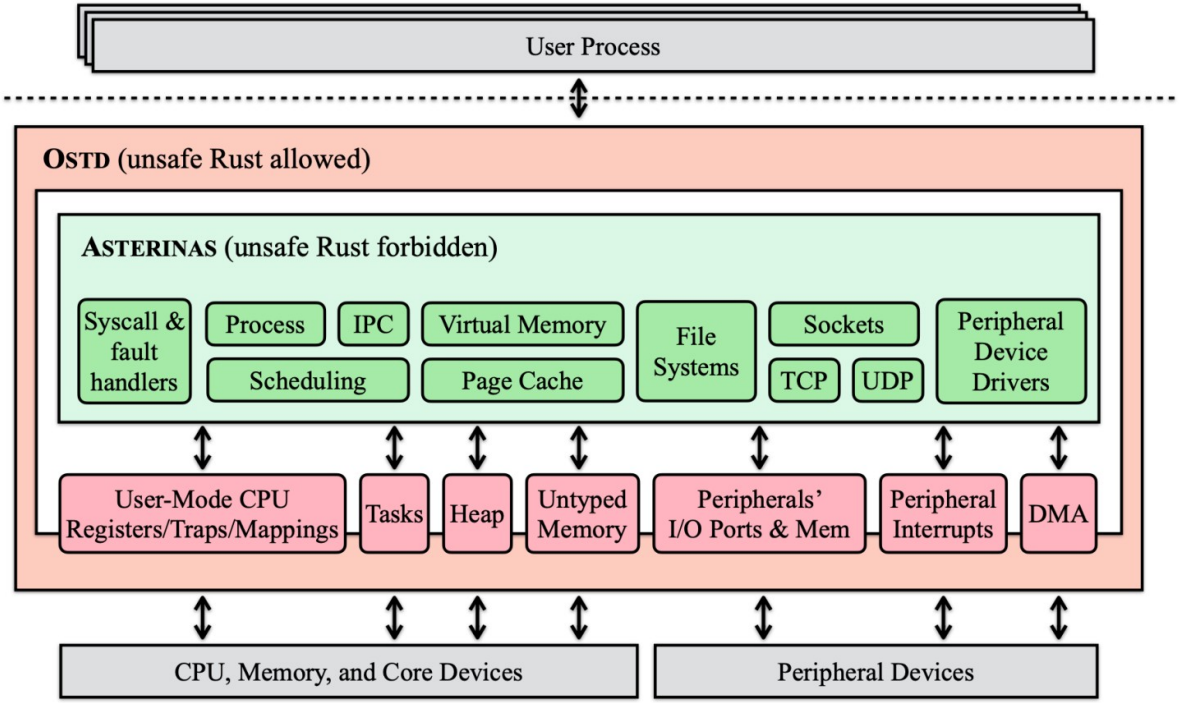
Wagner et al., “Towards Heterogeneous Remote Attestation Protocols”, SECRYPT ‘22

Inter-Enclave Communications

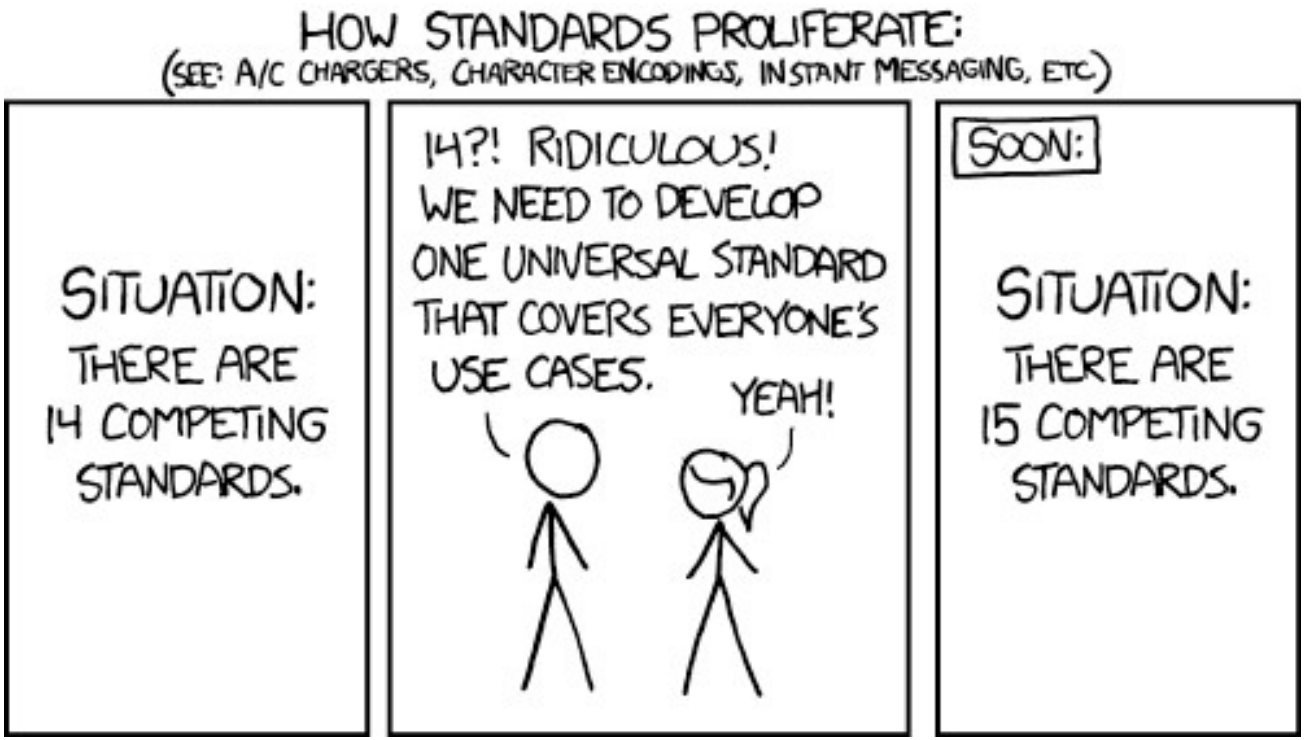
- Generic API
- Different applications need different guarantees
  - Confidentiality, integrity, ...

Framework Implementation Proposal

- Based on Asterinas
  - (Mostly-)Verified OS in Rust
  - TDX support
  - Small TCB



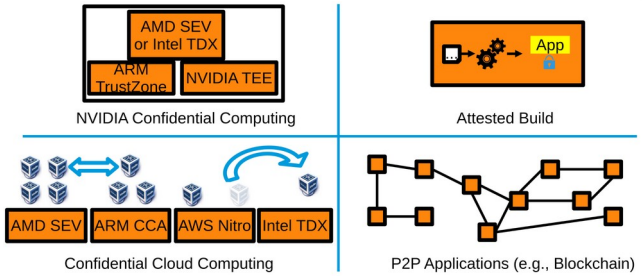
N+1 Framework?



Conclusions

- TEE landscape is heterogeneous
- We need a framework to unify them
  - Without weakening security guarantees
- FUTEE: a Framework to Unify TEEs
  - Capability system
  - TEE-agnostic attestation
  - Inter-TEE communication
  - Leverage Asterinas

pierrelouis@iij.ad.jp



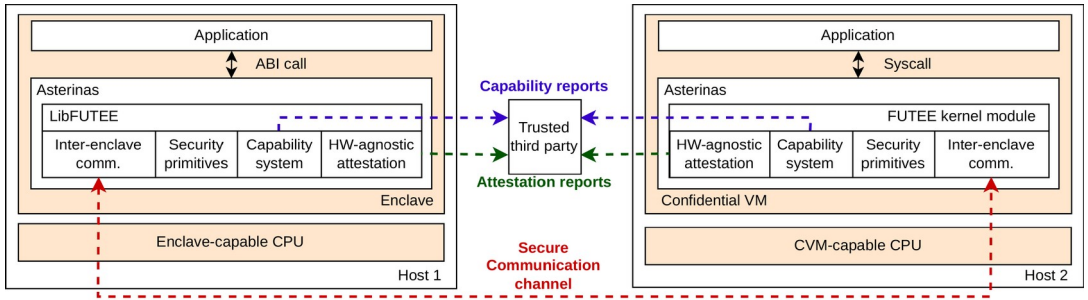


Fig. 1: Two secure applications communicate between each others via FUTEE. Dashed lines represent message exchange.