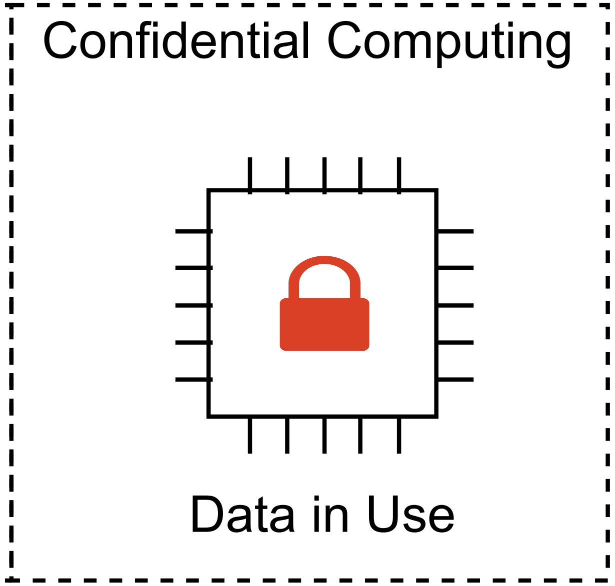
SHELTER: Extending Arm CCA with Isolation in User Space

Yiming Zhang[1,2], Yuxin Hu[1], Zhenyu Ning[3,1], **Fengwei Zhang[1]**✉, Xiapu Luo[2], Haoyang Huang[1], Shoumeng Yan[4], Zhengyu He[4]

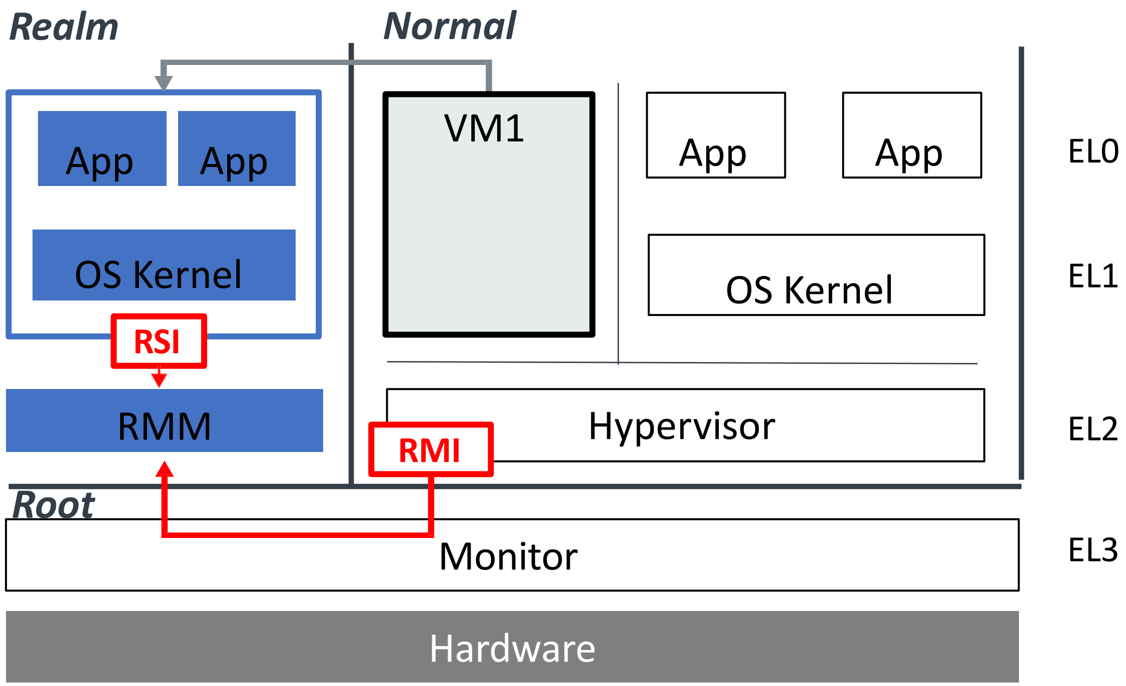
1Southern University of Science and Technology, 2The Hong Kong Polytechnic University, 3Hunan University, 4Ant Group

Confidential Computing

- Hardware-assisted security design
- Cloud and Edge devices
- Intel TDX, AMD SEV, Arm CCA



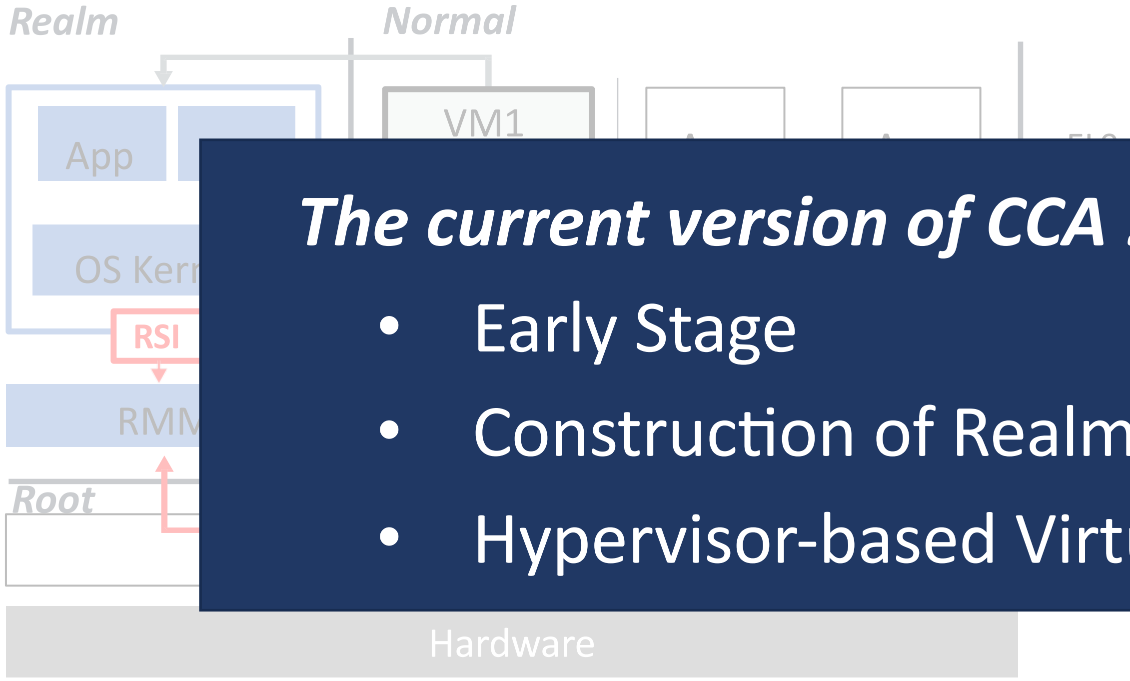
Arm Confidential Compute Architecture (CCA)



RME: Realm Management Extension RMM: Realm Management Monitor RMI: Realm Management Interface RSI: Realm Services Interface

- Introduced as supplement to Armv9.2-A
- Two added additional worlds
  - Secure -> Secure & EL3 Root
  - Normal -> Normal & Realm
- CCA is implemented in hardware and firmware

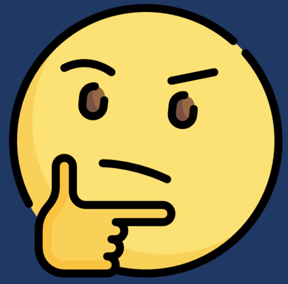
Arm Confidential Compute Architecture (CCA)



- Introduced as supplement to Armv9.2-A
- al VMs
- l parties
- CCA is implemented in hardware and firmware

The current version of CCA :

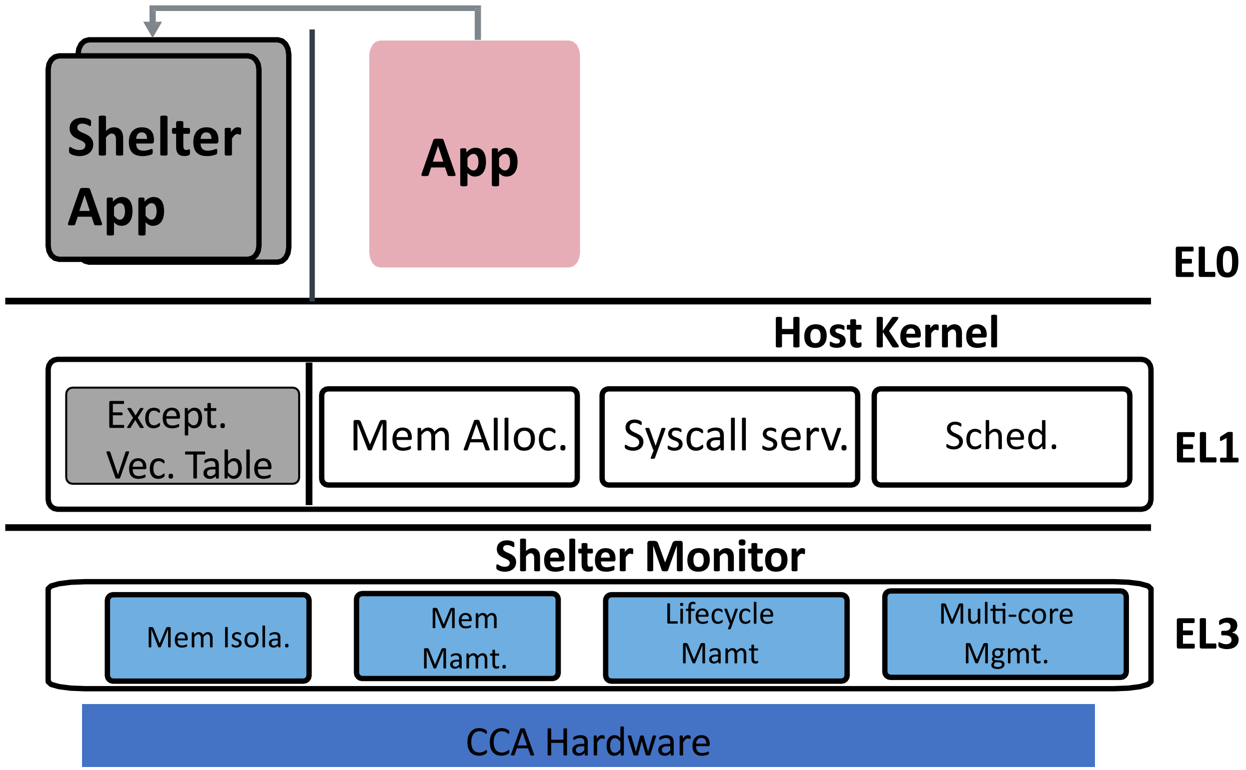
- Early Stage
- Construction of Realm VMs
- Hypervisor-based Virtualization



RME: Realm Management Extension RMM: Realm Management Monitor RMI: Realm Management Interface RSI: Realm Services Interface

Motivation

- Cooperating with CCA hardware to provide user-level isolation
- Complement to CCA's Realm VM architecture

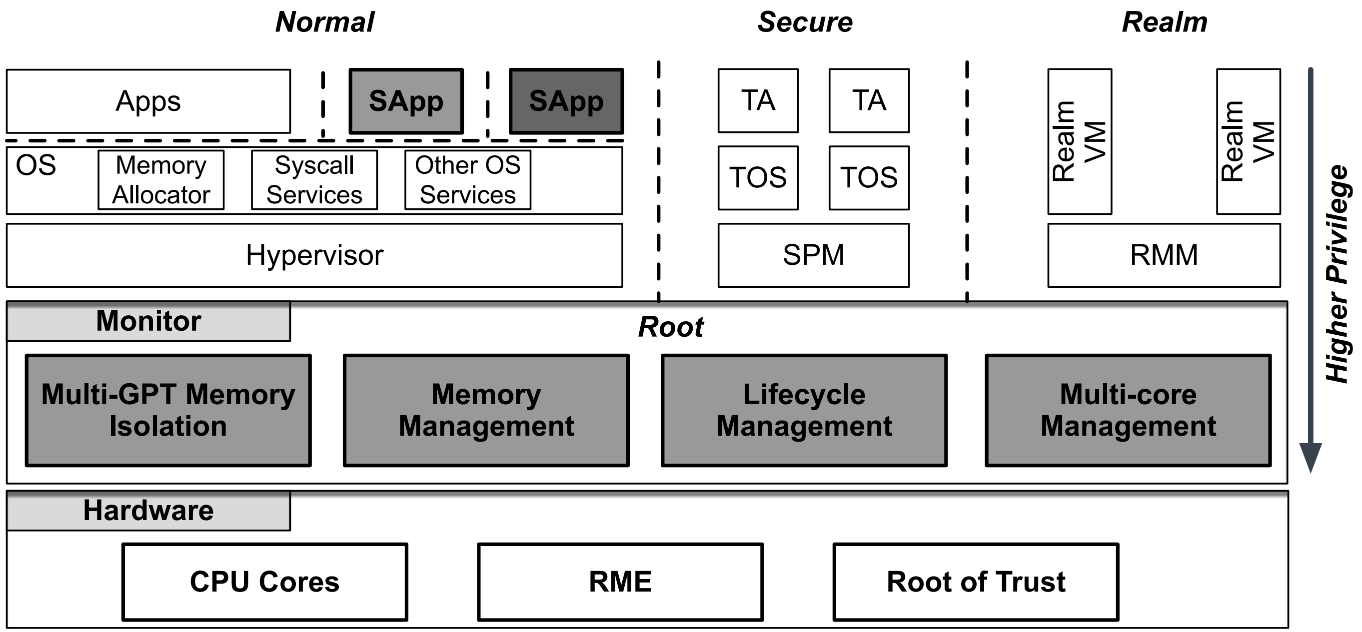


Threat Model & Assumptions

- An attacker **can** compromise Host OS, hypervisor, or **privileged software in Secure, and Realm world (e.g., SPM or RMM)**
- The Monitor is trusted and the hardware is correctly implemented
- Physical/Side-channel/denial-of-service attacks are out of scope
- Assuming remote attestation support and secure boot

Shelter

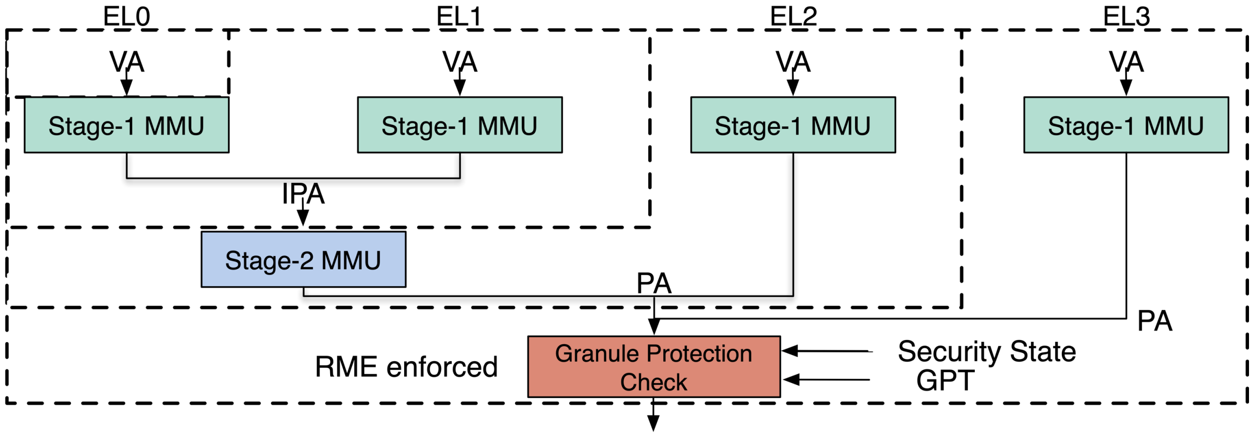
- SHELTER App (SApp)
  - Running on Normal World EL0
- Host OS
  - Non-security responsibilities
- Shelter Monitor
  - In Root world
  - Security responsibilities
- CCA hardware feature
  - Realm Management Extension (RME)



Granule Protection Check (GPC)

- RME enforced isolation is managed through **a new Granule Protection Table (GPT)**
- GPT is controlled by the Monitor in EL3
- GPT specifies what physical address spaces (PAS) a memory page belongs to

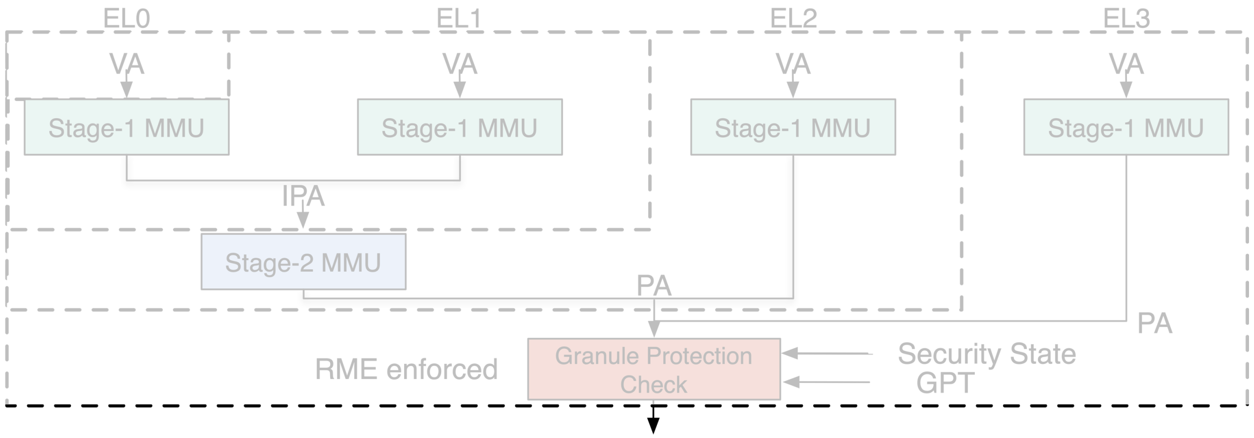
Security state	Normal PAS	Secure PAS	Realm PAS	Root PAS
Normal	✓	×	×	×
Secure	✓	✓	×	×
Realm	✓	×	✓	×
Root	✓	✓	✓	✓



Granule Protection Check (GPC)

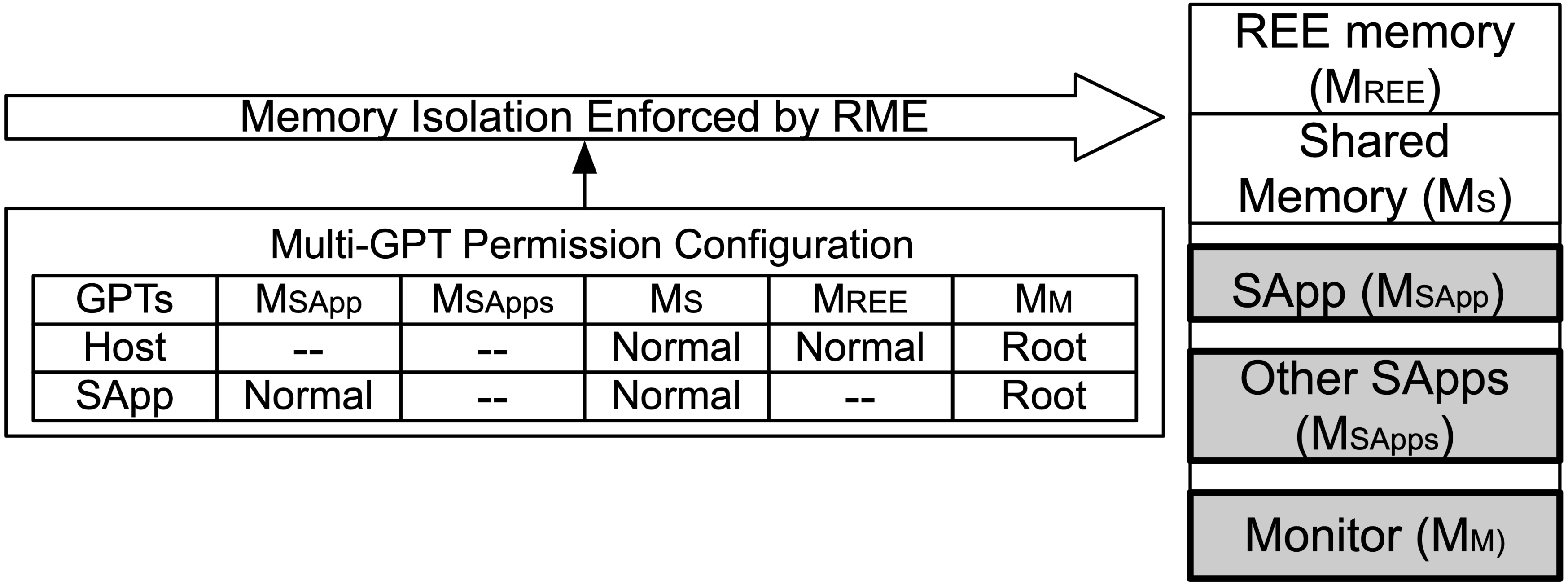
- RME enforced isolation is managed through **a new Granule Protection Table (GPT)**
- GPT is controlled by the Monitor in EL3
- GPT specifies what physical address spaces (PAS) a memory page belongs to

It is not satisfied with the goal of isolating memory between SApps and other privileged software in Normal, Secure, and Realm world.



Multi-GPT Memory Isolation

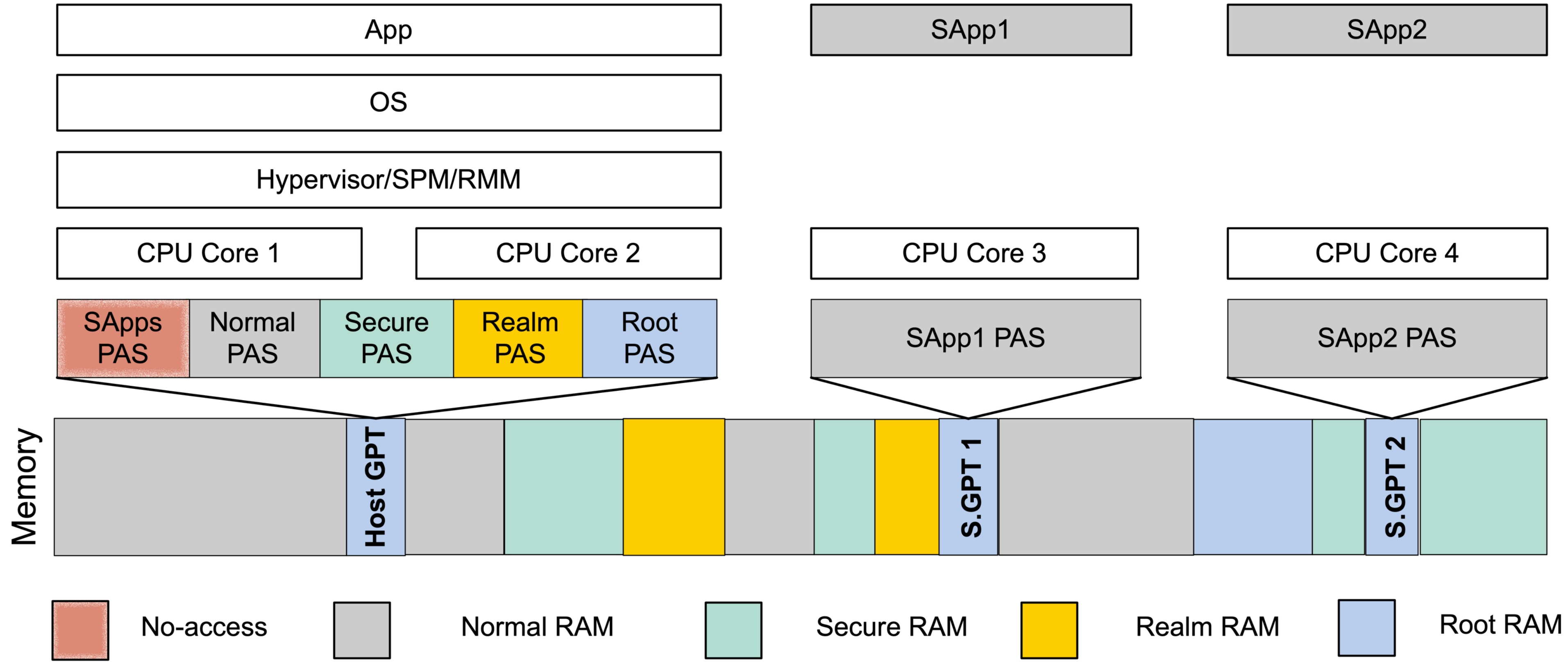
- Maintain multiple GPTs in EL3 Monitor
- Divide the physical address space (PAS) for different programs



GPTs	M_{SApp}	M_{SApps}	M_S	M_{REE}	M_M
Host	--	--	Normal	Normal	Root
SApp	Normal	--	Normal	--	Root

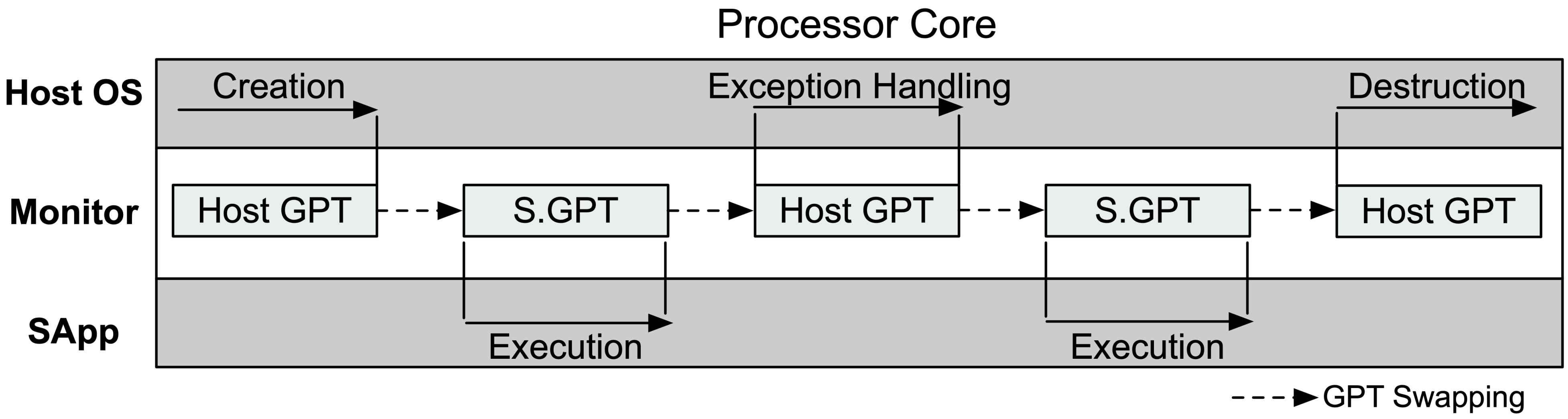
Multi-GPT Memory Isolation

- Establishing address-space-per-core for each SApp and other code region



Multi-GPT Memory Isolation

- The Monitor dynamically controls the access permissions of different programs

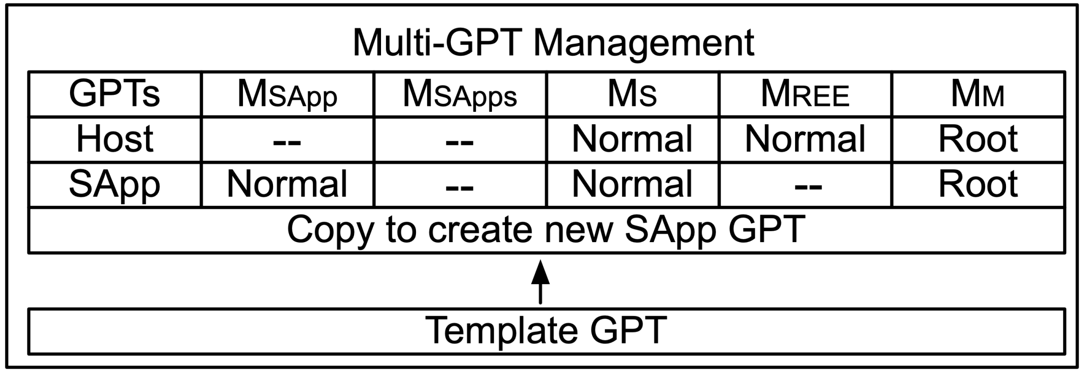


Performance Optimization

- New GPT construction causes long startup latency for SApps
  - **Root cause:** Shelter needs to add granule information containing a layout of the entire main memory for the new GPT and measure each GPT entry

Performance Optimization

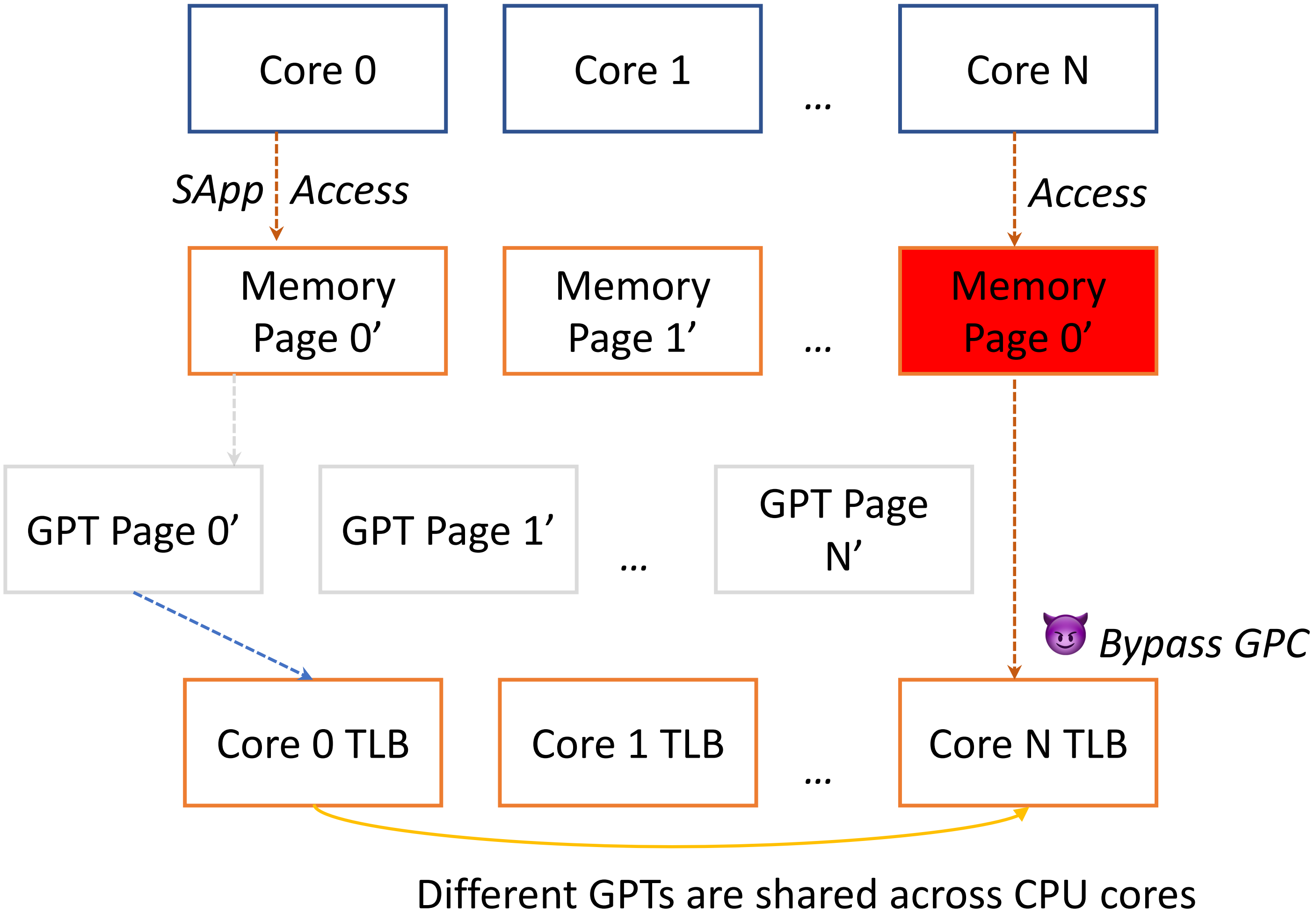
- New GPT construction causes long startup latency for SApps
  - **Root cause:** Shelter needs to add granule information containing a layout of the entire main memory for the new GPT and measure each GPT entry



**Using shadow GPT, a template with copy and update to speed up SApp creation*

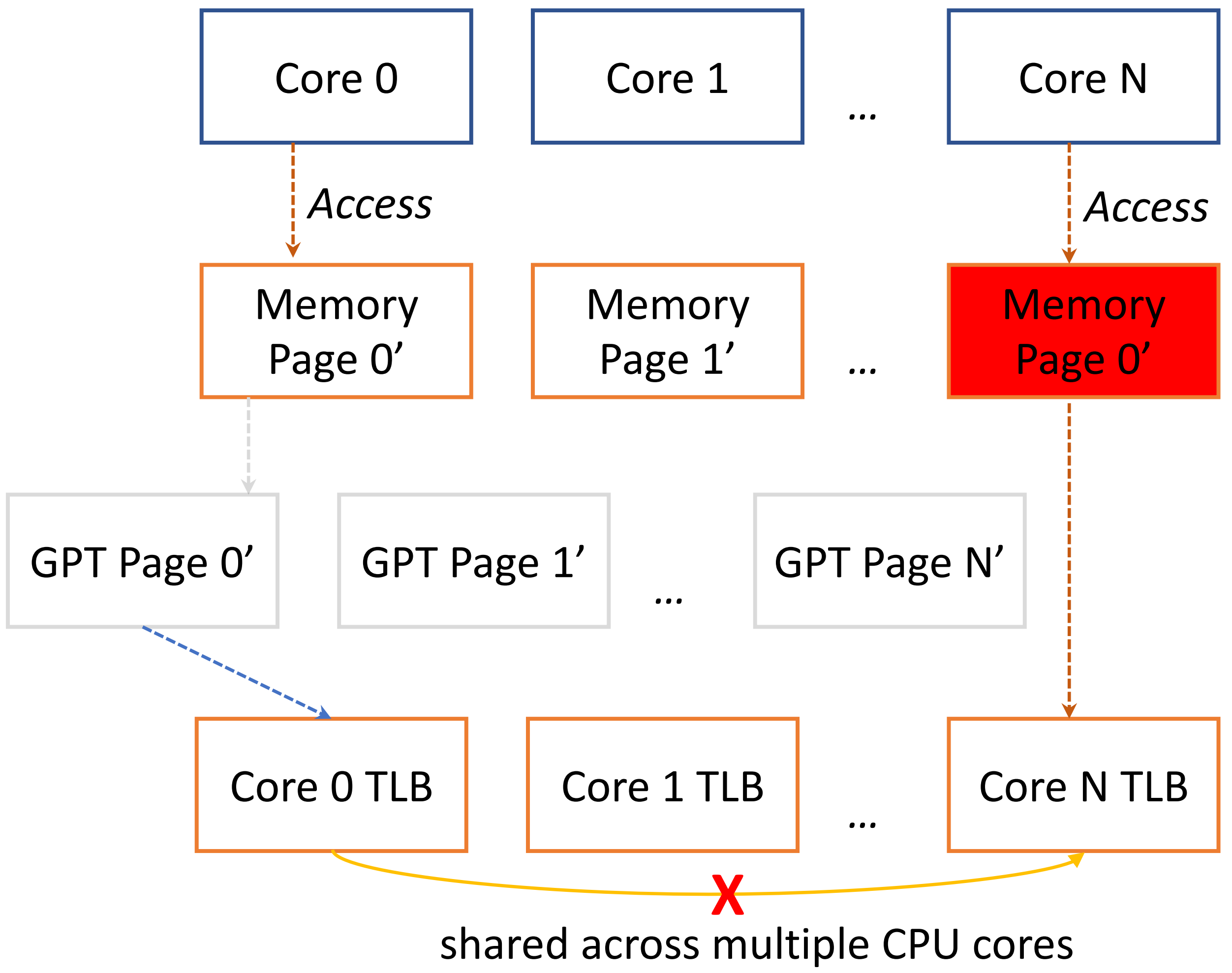
TLB-based GPT attack

- GPT entries are permitted to be cached in TLB as part of TLB entry
- GPT information in a TLB is permitted to be shared across multiple CPU cores



Defend against TLB-based GPT attacks

- TLB invalidation during switches and GPT modifications
- Disable the shareable property of TLB



Some Execution Features

- Memory management
  - Contiguous physical memory pool
  - Ensure multiple SApps do not have memory overlap
  - SApp Page table is isolated

- Syscall & Iago attack checks
- Interrupt & Signal
- Multi-threaded synchronization primitive

Security Analysis

Adversary Subject	Main Attacks	Defense
OS/Hypervisor	Unauthorized memory access	❶
	Invalid mapping or return value	❷
	Illegal GPT modification	❸❺
	GPC circumvention	❺
SApp	SApp abuse	❶❷
TLB/Cache	Untended GPT sharing in TLB	❹
	Unauthorized cache access	❶❹
	EL3 code cache injection	❺
Peripherals	Malicious DMA	❶

❶ Multi-GPT isolation enforced by GPC; ❷ Monitor checks (e.g., ensuring no memory overlap between SApps, checking syscall return value, verifying validity of the runtime); ❸ Multi-core synchronization; ❹ Microarchitectural Maintenance; ❺ Monitor Maintaining in the highest privilege.

Shelter Implementation

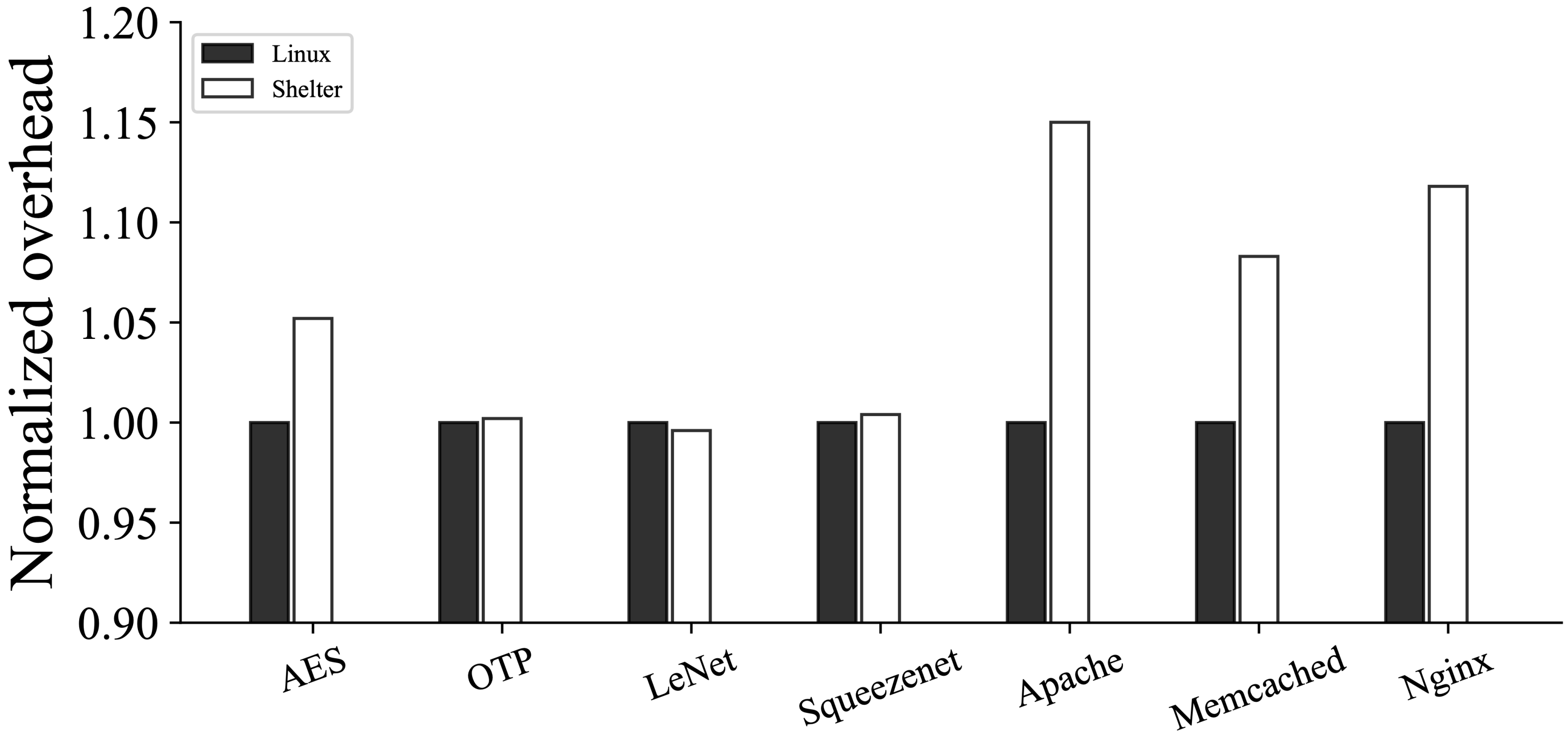
- Functional prototype implementation
    - FVP Base RevC-2xAEMvA with RME-enabled features
    - TCB: ATF with **2k SLoCs additions**
- Official CCA software stacks
    - TCB: ATF + TF-RMM (released date 2022/11/09)
    - TF-RMM(v0.2.0) is around 8.2k SLoCs
- TCB comparison with CCA
    - ***2k vs 8k SLoCs***

Performance Evaluation

- No commercial hardware supporting CCA is available on the market
  - FVP Simulator is **not cycle accurate**
- GPT-analogue in Armv8-A Juno Board
  - Mimic all **GPT in-memory** operations
  - Replace the **GPT-related registers** with **idle EL3 registers**
  - **Invalidate all TLBs** instead of TLB GPT invalidation instructions (e.g., **TLBI PAALLOS**)
  - The other functionality are the same as those on the FVP

Application Benchmarks

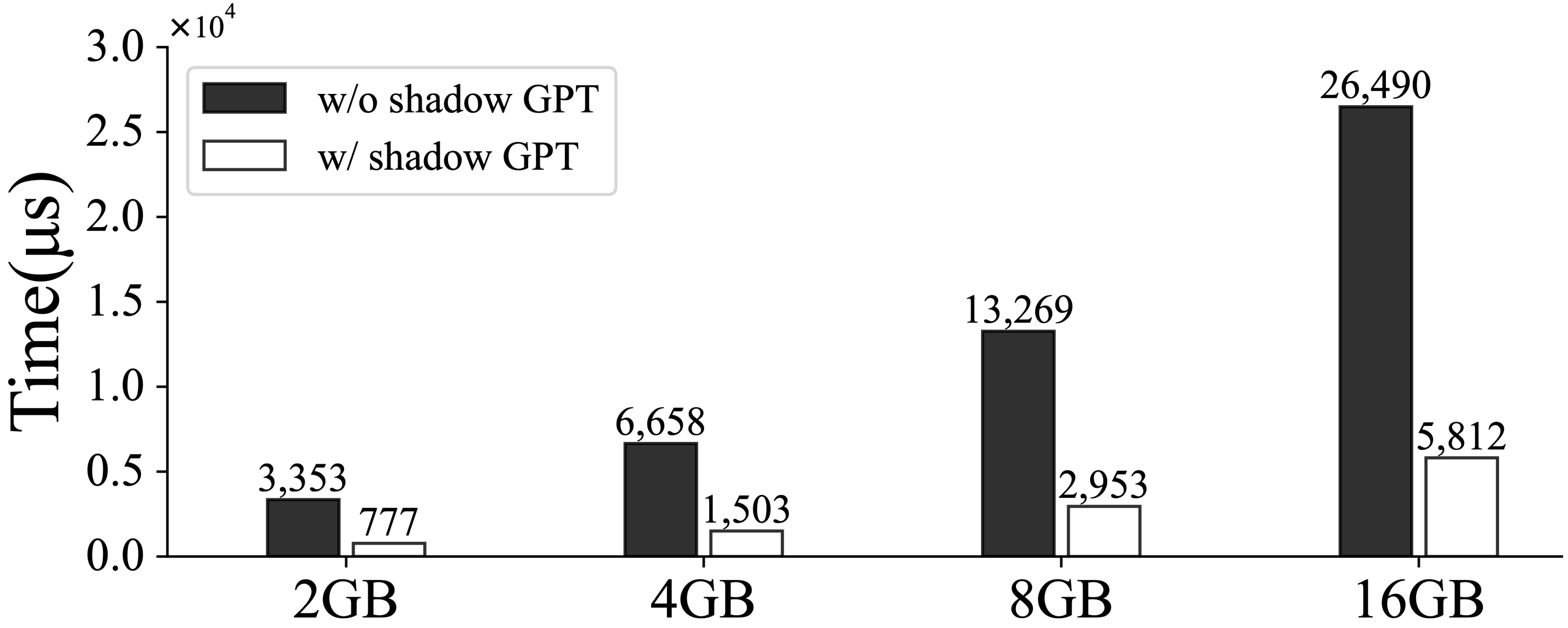
*Runtime Overhead on **real-world programs***



SHELTER incurs <15% runtime-overhead on real-world workloads compared with Linux

Performance Optimization

*Overhead in **GPT construction***

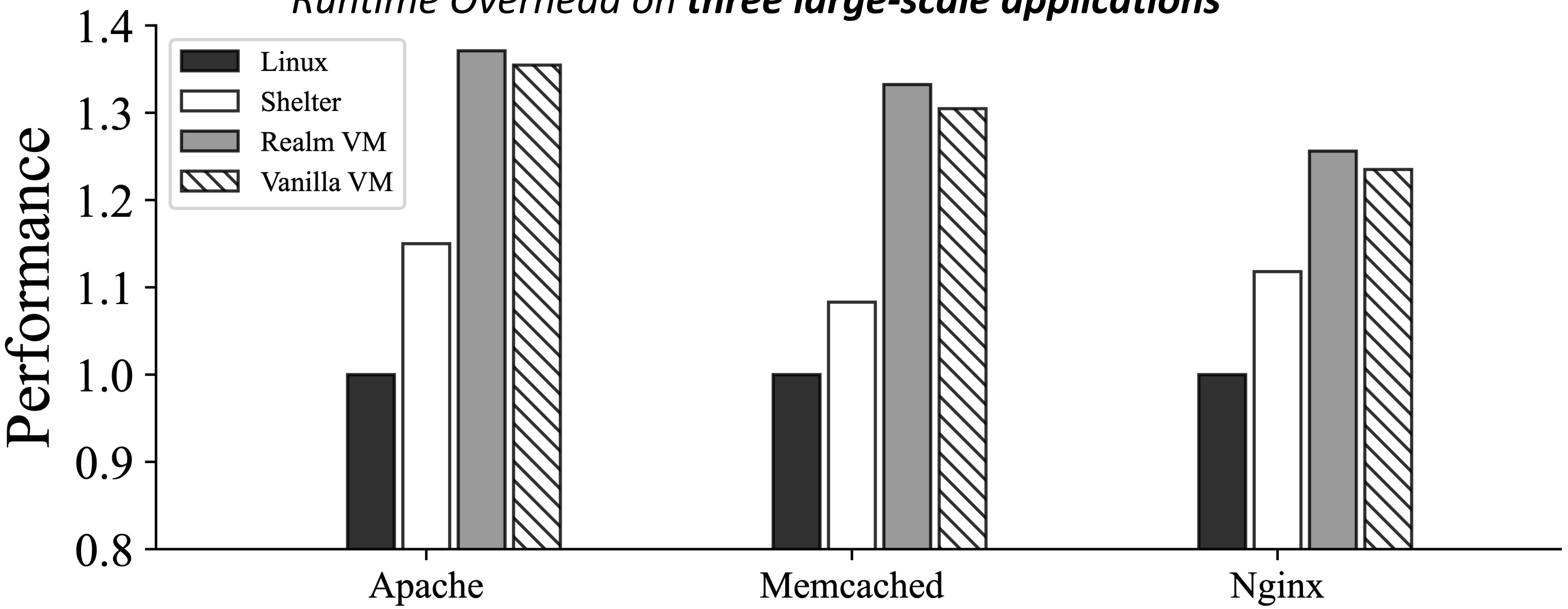


✓ With shadow GPT, reducing overhead on average of 77.5% in SApp Creation

Comparison with CCA's VM-based approach

- A basic CCA VM-based performance prototype with same GPT-analogue methodology and a Realm-context simulation

Runtime Overhead on ***three large-scale applications***



✓ Avg. SHELTER 11.7% vs CCA Realm VM 32.0%

Conclusion

- **Shelter** leverages CCA hardware for a new creation of **user-level** isolated environment
    - complementary to CCA's primary Realm VM-style architecture
    - A smaller TCB
    - Lower performance overhead
    - No hardware modification for compatible platforms, including mobile and server
- Open Source
    - https://github.com/Compass-All/Shelter

Thanks for listening!

Q & A

zhangfw@sustech.edu.cn