The Road to Trust:
Building Enclaves within Confidential VMs

Wenhao Wang, **Linke Song**, Benshan Mei, Shuang Liu,
Shijun Zhao, Shoumeng Yan, XiaoFeng Wang, Dan Meng, Rui Hou

中国科学院信息工程研究所
INSTITUTE OF INFORMATION ENGINEERING,CAS

蚂蚁集团
ANT GROUP

INDIANA UNIVERSITY
BLOOMINGTON

SYMPOSIUM/2025

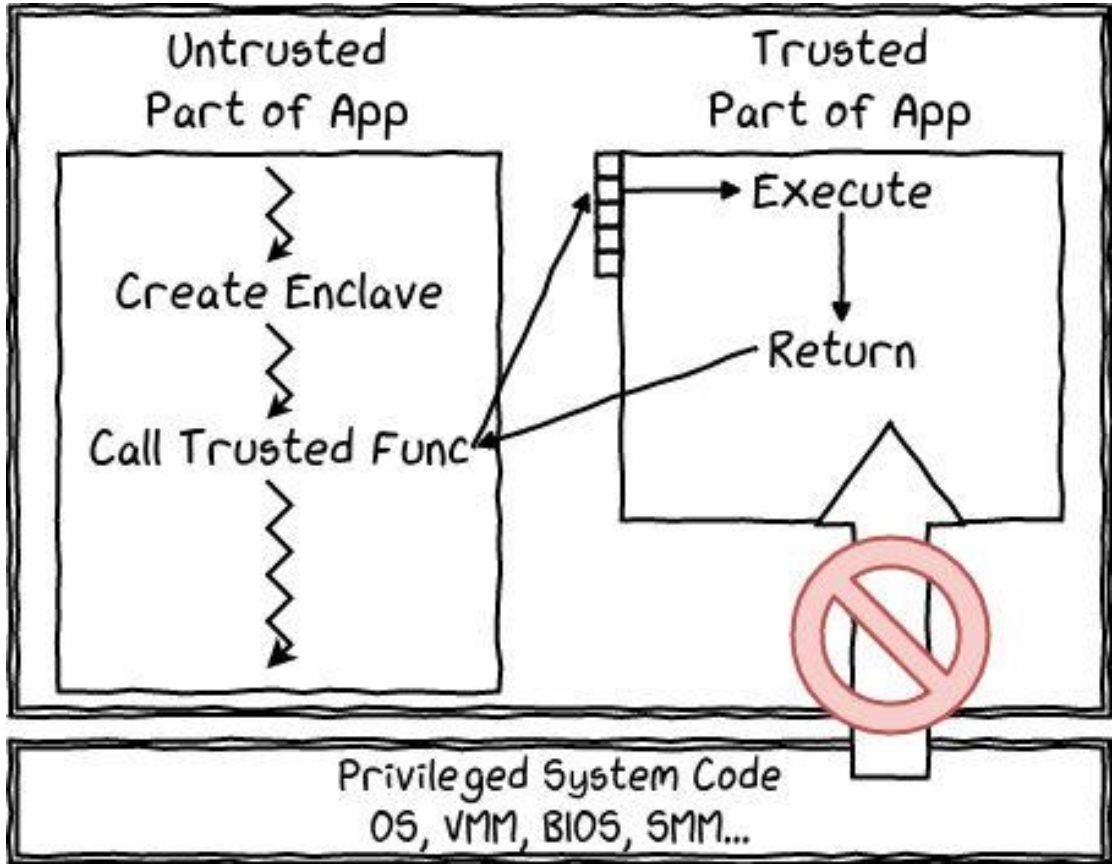
Trusted Execution Environment

- Isolate the code and data of a confidential workload from other code

Trusted Execution Environment

- Isolate the code and data of a confidential workload from other code

- **INTEL SGX**
- Intel SGX workflow:
  - Create enclave and add pages
  - Initialize
  - Enter enclaves
  - Execute
  - Exit enclaves
  - Remove enclaves and recycle



Trusted Execution Environment

- Isolate the code and data of a confidential workload from other code

- **AMD SEV:**
  - Encrypts the entire VM
  - Explicitly shares data

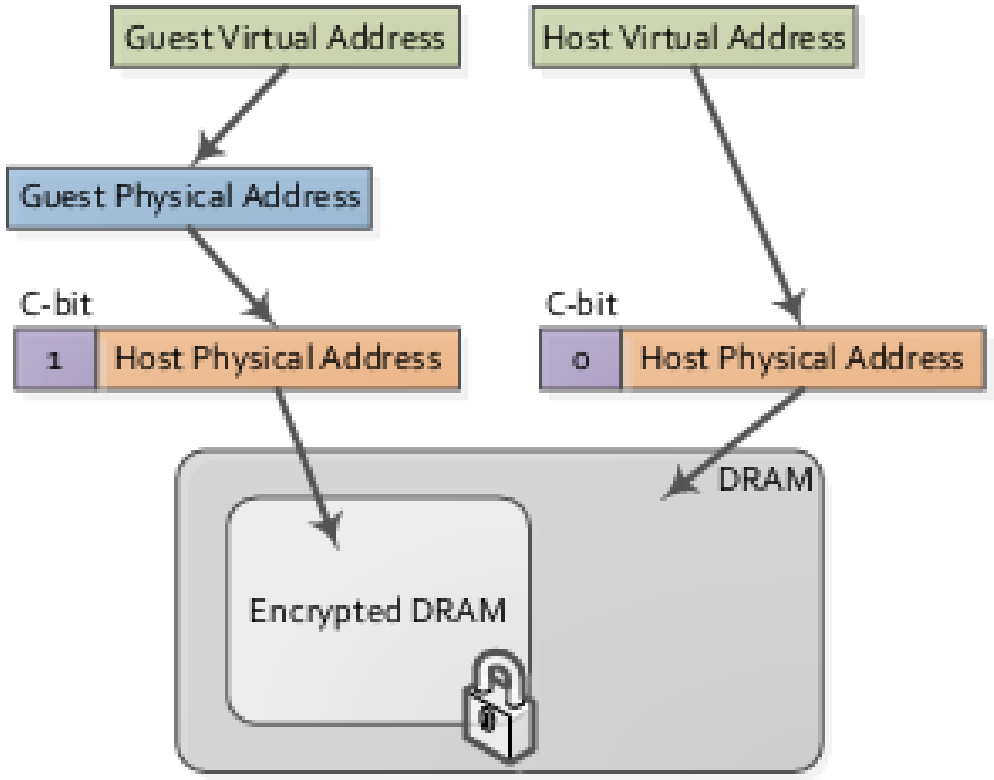
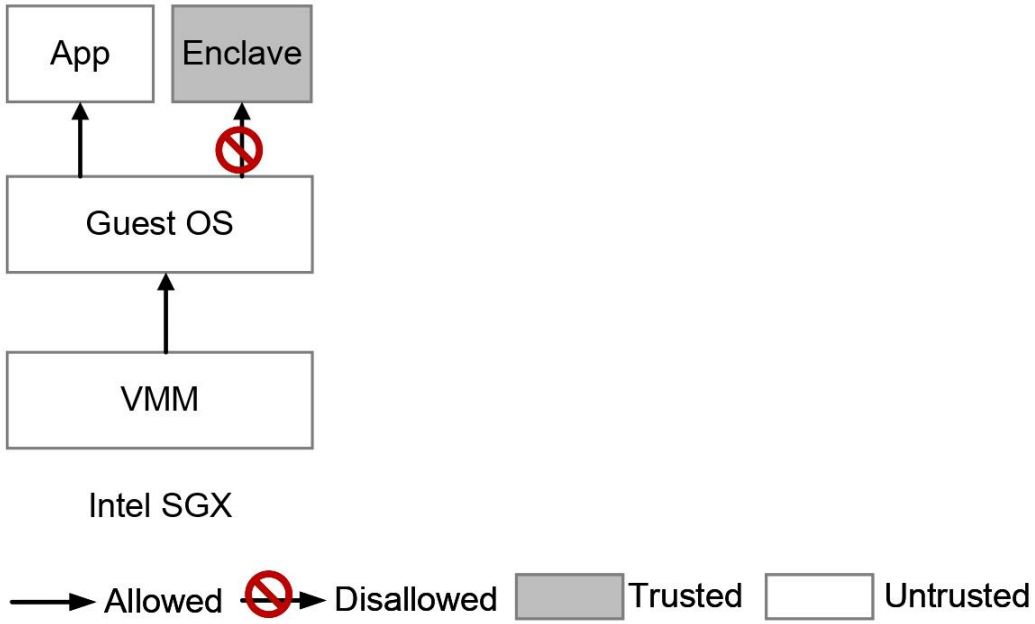


Figure 3: Encrypted VMs

** https://www.amd.com/content/dam/amd/en/documents/epyc-business-docs/white-papers/memory-encryption-white-paper.pdf*

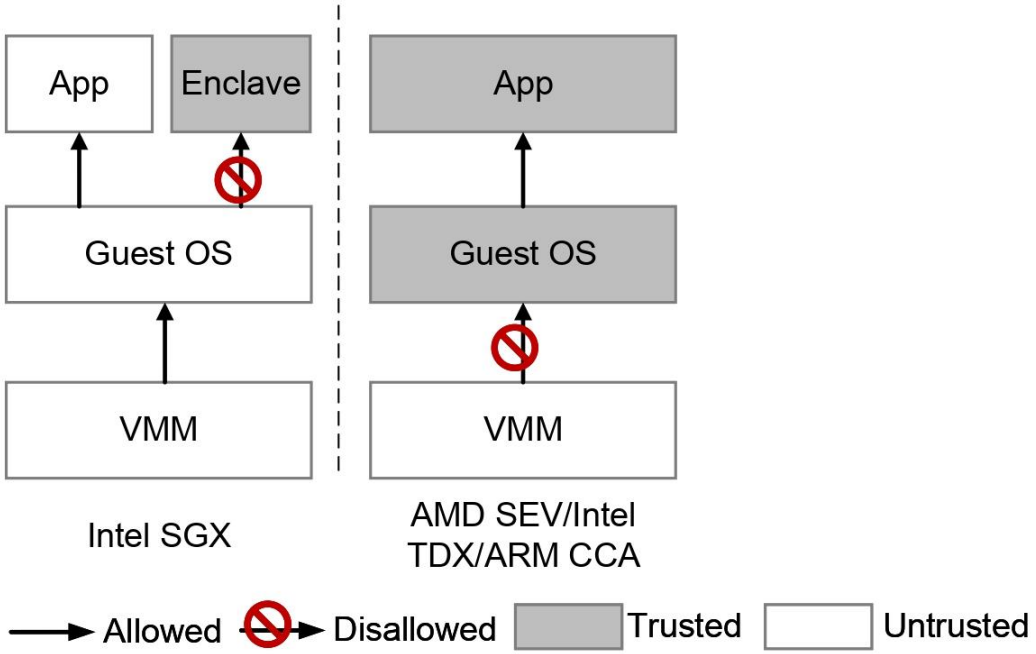
Trusted Execution Environment

- Isolate the code and data of a confidential workload from other code
- Types:
  - Process-based: Intel SGX



Trusted Execution Environment

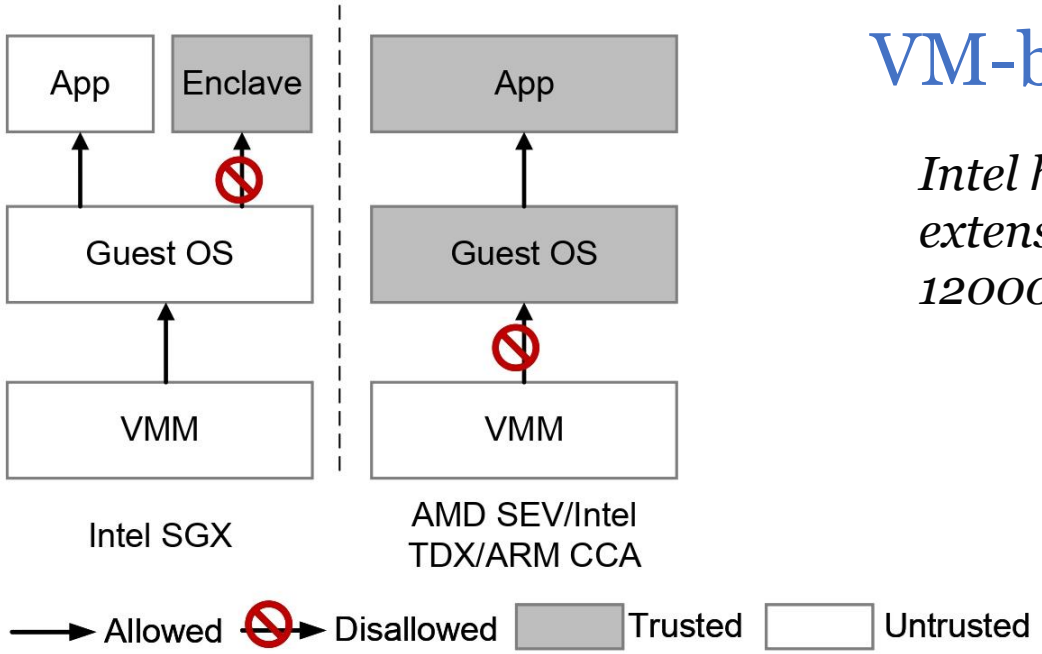
- Isolate the code and data of a confidential workload from other code
- Types:
  - Process-based: Intel SGX
  - VM-based: AMD SEV/Intel TDX/ARM CCA



** https://www.bleepingcomputer.com/news/security/new-intel-chips-wont-play-blu-ray-disks-due-to-sgx-deprecation/*

Trusted Execution Environment

- Isolate the code and data of a confidential workload from other code
- Types:
  - Process-based: Intel SGX
  - VM-based: AMD SEV/Intel TDX/ARM CCA



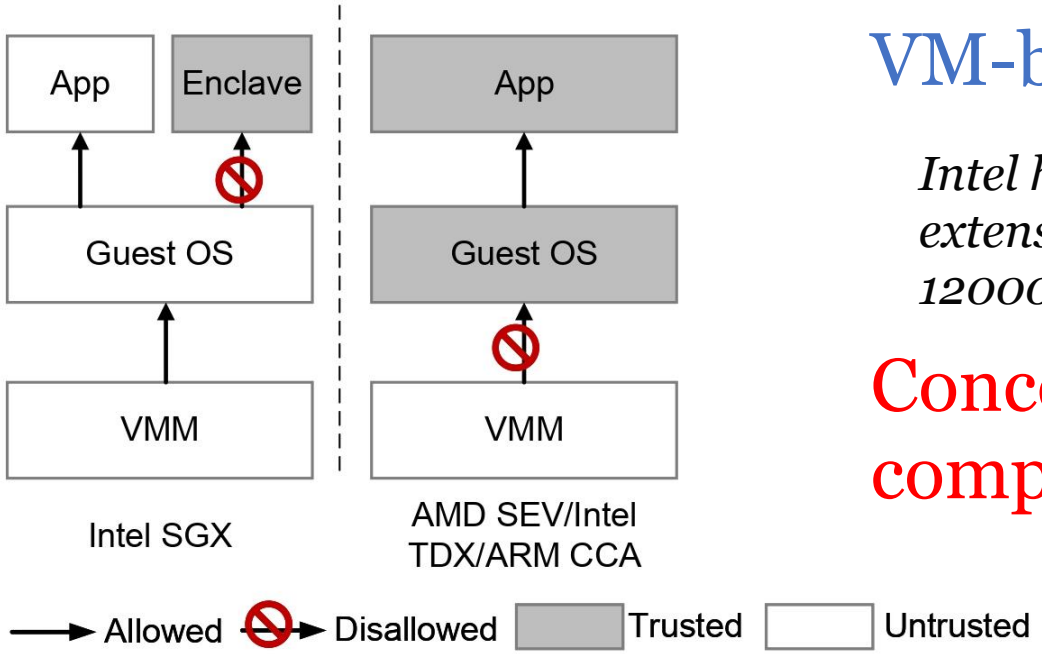
VM-based TEEs are more prevalent now

*Intel has removed support for SGX (software guard extension) in 12th Generation Intel Core 11000 and 12000 processors …**

** https://www.bleepingcomputer.com/news/security/new-intel-chips-wont-play-blu-ray-disks-due-to-sgx-deprecation/*

Trusted Execution Environment

- Isolate the code and data of a confidential workload from other code
- Types:
  - Process-based: Intel SGX
  - VM-based: AMD SEV/Intel TDX/ARM CCA



VM-based TEEs are more prevalent now

*Intel has removed support for SGX (software guard extension) in 12th Generation Intel Core 11000 and 12000 processors …**

Concern: What if the guest OS might be compromised in VM-Based TEEs?

** https://www.bleepingcomputer.com/news/security/new-intel-chips-wont-play-blu-ray-disks-due-to-sgx-deprecation/*

Research Questions of VM-Based TEE

- **Research Gap:** In VM-based TEE, how can users establish trust in applications within confidential VM, if the guest OS is compromised?

- The whole lifecycle of enclaves should be trusted
  - build, load, modification, execution, etc.

Research Questions of VM-Based TEE

- **Research Gap:** In VM-based TEE, how can users establish trust in applications within confidential VM, if the guest OS is compromised?

- The whole lifecycle of enclaves should be trusted
  - build, load, modification, execution, etc.

- Seminal work: vSGX [S&P'22]

Research Questions of VM-Based TEE

- **Research Gap:** In VM-based TEE, how can users establish trust in applications within confidential VM, if the guest OS is compromised?

- The whole lifecycle of enclaves should be trusted
  - build, load, modification, execution, etc.

- Seminal work: vSGX [S&P'22]
  - Using two-VMs, one for untrusted apps, the other for enclaves.

Research Questions of VM-Based TEE

- **Research Gap:** In VM-based TEE, how can users establish trust in applications within confidential VM, if the guest OS is compromised?

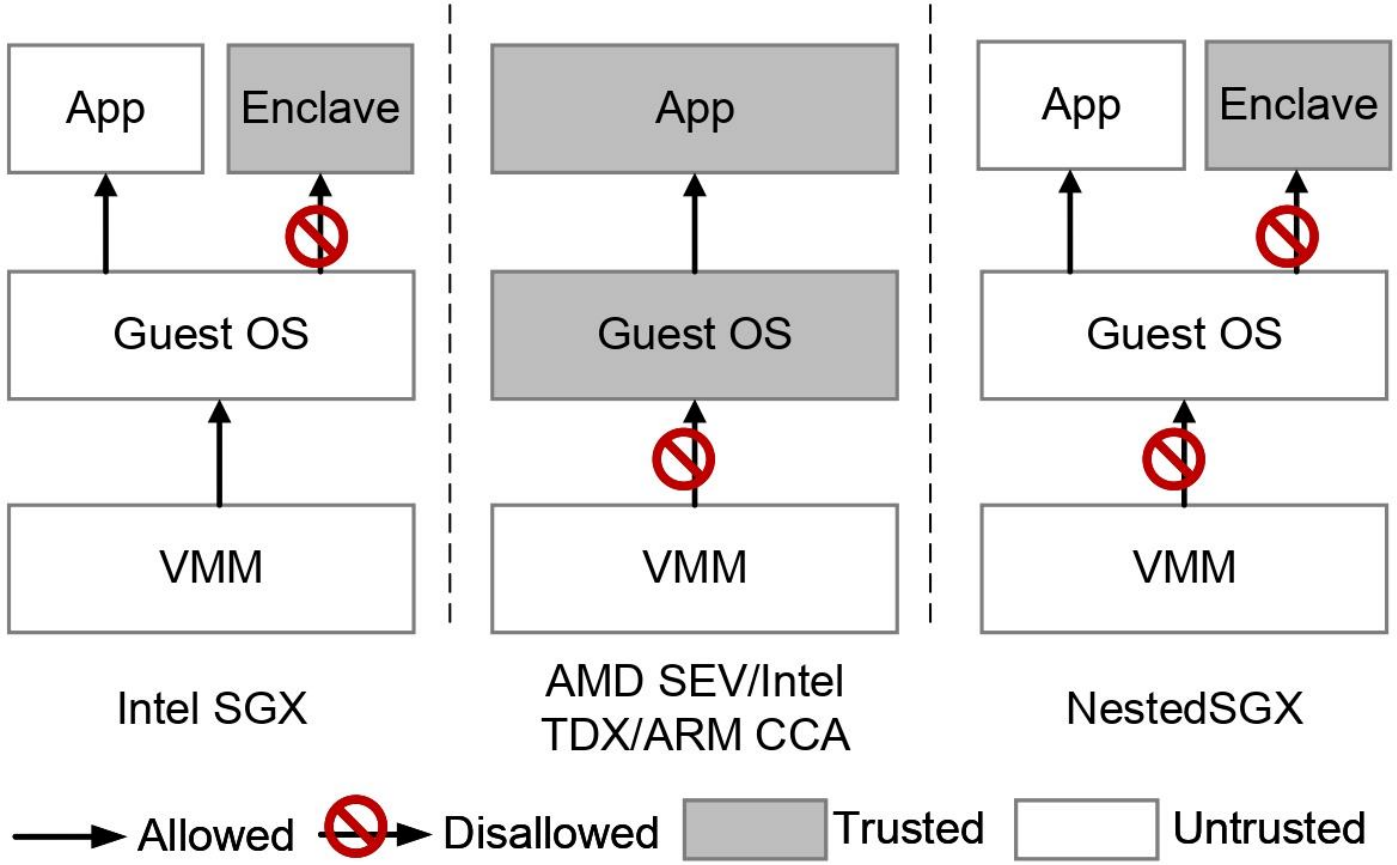
- The whole lifecycle of enclaves should be trusted
  - build, load, modification, execution, etc.

- Seminal work: vSGX [S&P'22]
  - Using two-VMs, one for untrusted apps, the other for enclaves.
  - Great TCBs
  - Significant overhead for 'world-switch'
    - Empty ECall: 1.5 ms, ~160x slower than Native Intel SGX
    - Benchmarks:
      - 6x slower cURL, 5 mins for launching 256 MB Enclave

NestedSGX

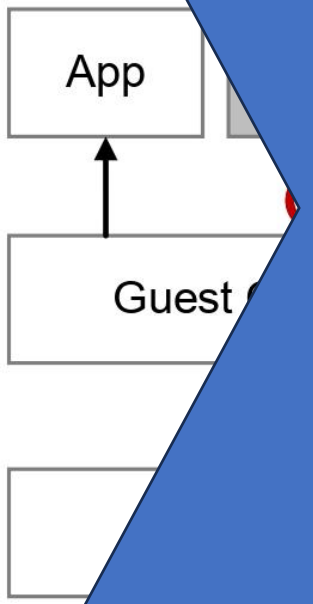
Overview of NestedSGX

- **NestedSGX:**
  - The only trusted component is the **enclave**



Overview o

- **NestedSGX:**
  - The only trust



Not Enough

Trivial Questions for NestedSGX

A.K.A **Goals**

Trivial Questions for NestedSGX

A.K.A **Goals**

- How to mitigate the significant overhead of ‘world-switching’ without compromising security requirements?

Trivial Questions for NestedSGX

A.K.A **Goals**

- How to mitigate the significant overhead of ‘world-switching’ without compromising security requirements?

- Is NestedSGX built on CVM sufficiently compatible? Can it support a certain level of application ecosystem?

Our contributions: NestedSGX

- **Design:** Using the VMPL feature within **AMD SEV-SNP**, NestedSGX establishes trust for applications while minimizing reliance on the guest OS.

Our contributions: NestedSGX

- **Design:** Using the VMPL feature within **AMD SEV-SNP**, NestedSGX establishes trust for applications while minimizing reliance on the guest OS.

- **Compatibility with Intel SGX:** NestedSGX has similar management of enclaves to Intel SGX, and it significantly reduces the effort required to port existing applications of Intel SGX onto the NestedSGX framework.

Our contributions: NestedSGX

- **Design:** Using the VMPL feature within **AMD SEV-SNP**, NestedSGX establishes trust for applications while minimizing reliance on the guest OS.

- **Compatibility with Intel SGX:** NestedSGX has similar management of enclaves to Intel SGX, and it significantly reduces the effort required to port existing applications of Intel SGX onto the NestedSGX framework.

- **Implementations and Evaluations:** We Implemented NestedSGX on commercial hardware, the evaluation shows the overhead is comparable to that of Intel SGX, affirming its efficiency and viability.

Architecture

- **Design:** Using the VMPL feature of AMD SEV-SNP

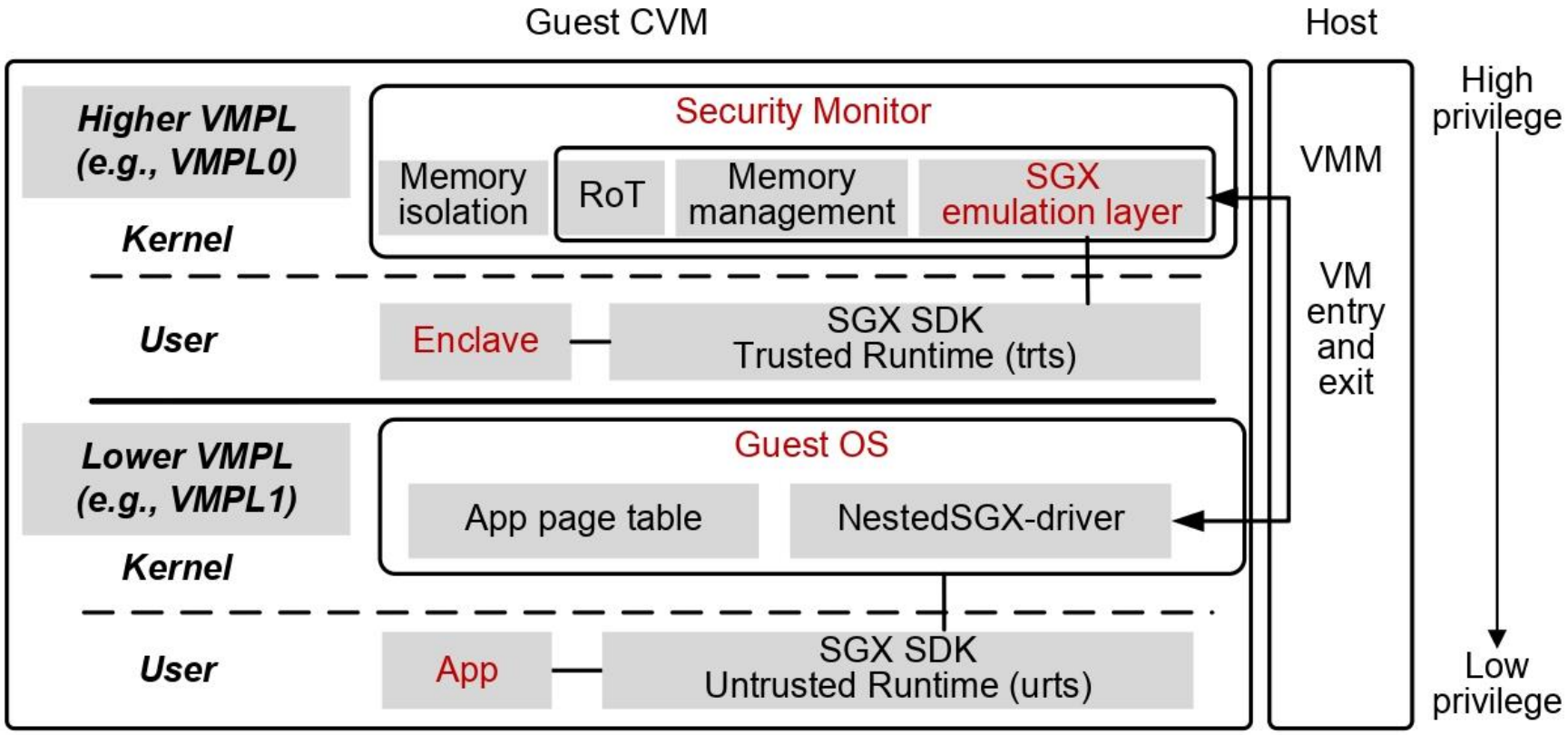
Architecture

- **Design:** Using the VMPL feature of AMD SEV-SNP

- **VMPL**: Virtual Machine Privilege Levels (VMPL0 – VMPL3)
  - Every VMPL shares the same address space, with VMPL0 representing the highest privilege level, and VMPL3 representing the lowest privilege level.

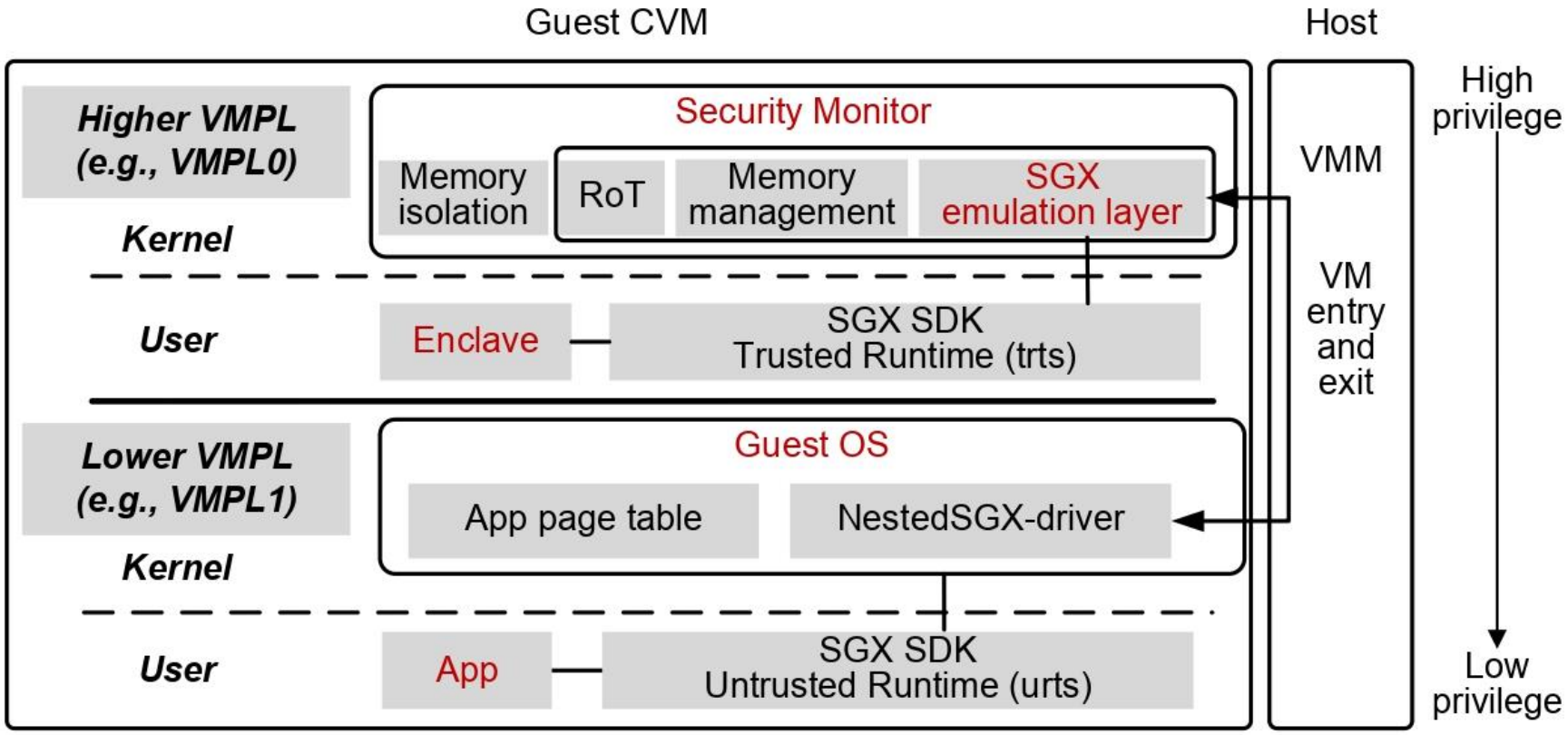
Architecture

- **Privilege Separation: VMPL0 (Trusted), VMPL1 (Untrusted)**



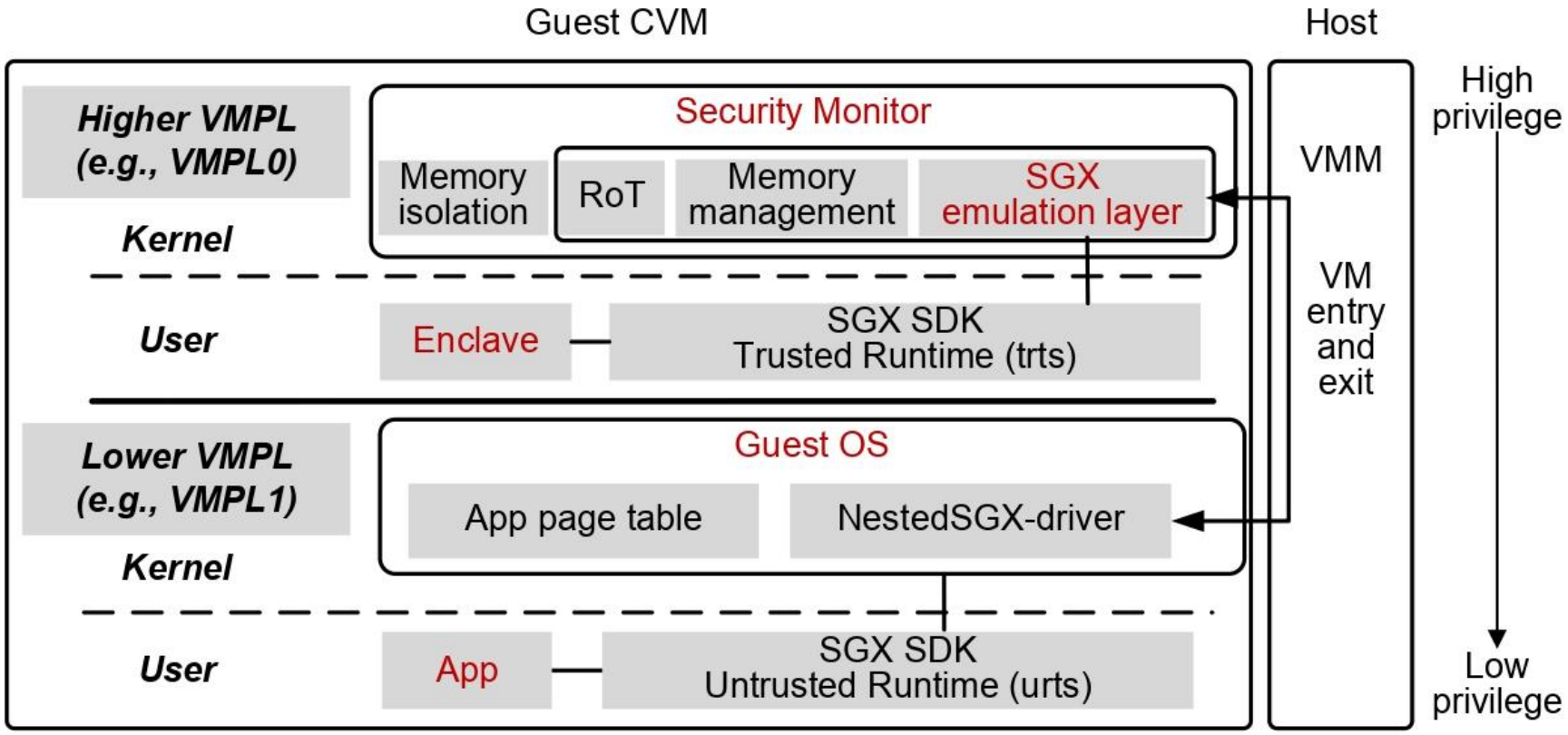
Architecture

- **Privilege Separation: VMPL0 (Trusted), VMPL1 (Untrusted)**
  - **VMPL0, Kernel:** Security Monitor
  - **VMPL0, User:** Enclave



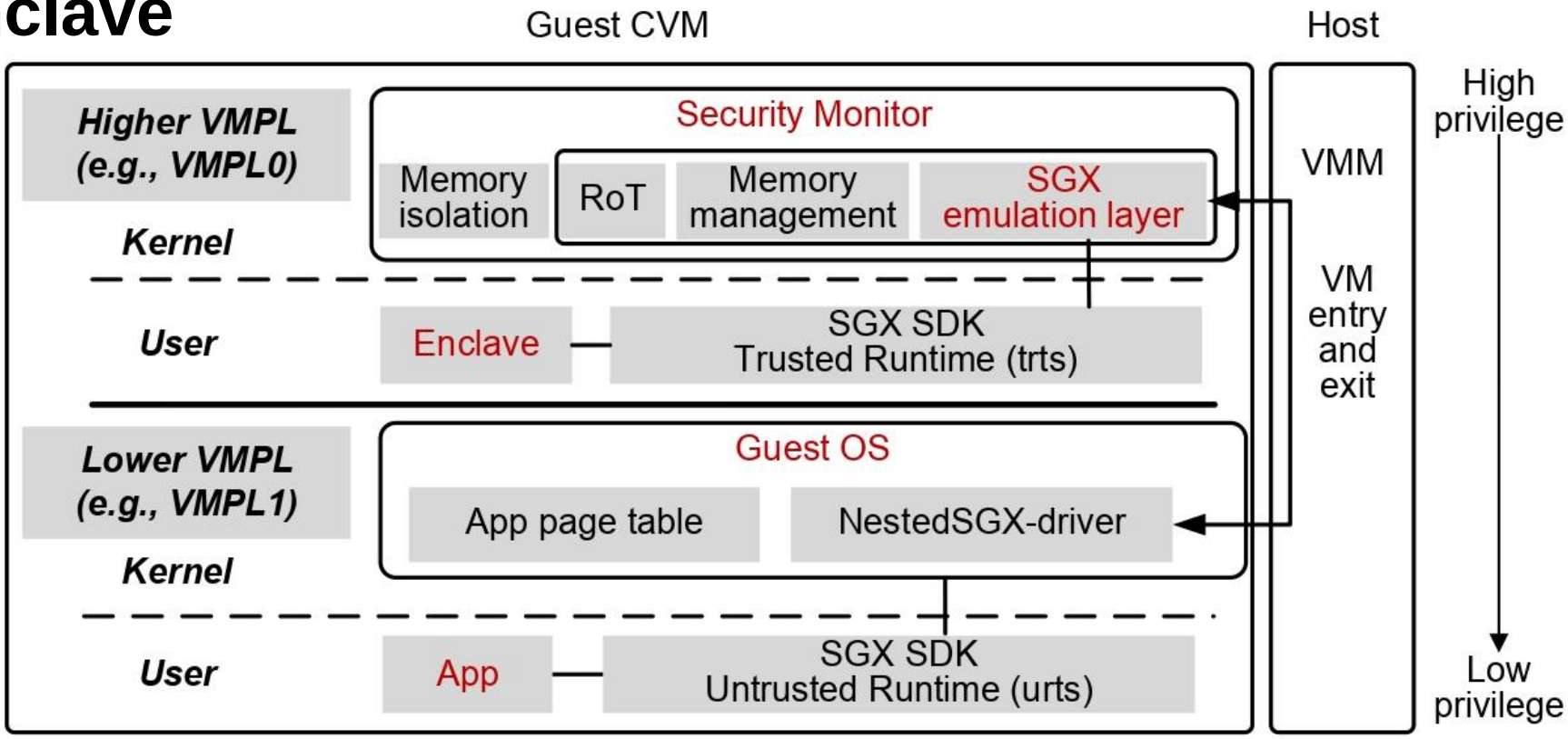
Architecture

- **Privilege Separation: VMPL0 (Trusted), VMPL1 (Untrusted)**
  - **VMPL0, Kernel:** Security Monitor
  - **VMPL0, User:** Enclave
  - **VMPL1:** Guest OS, Application



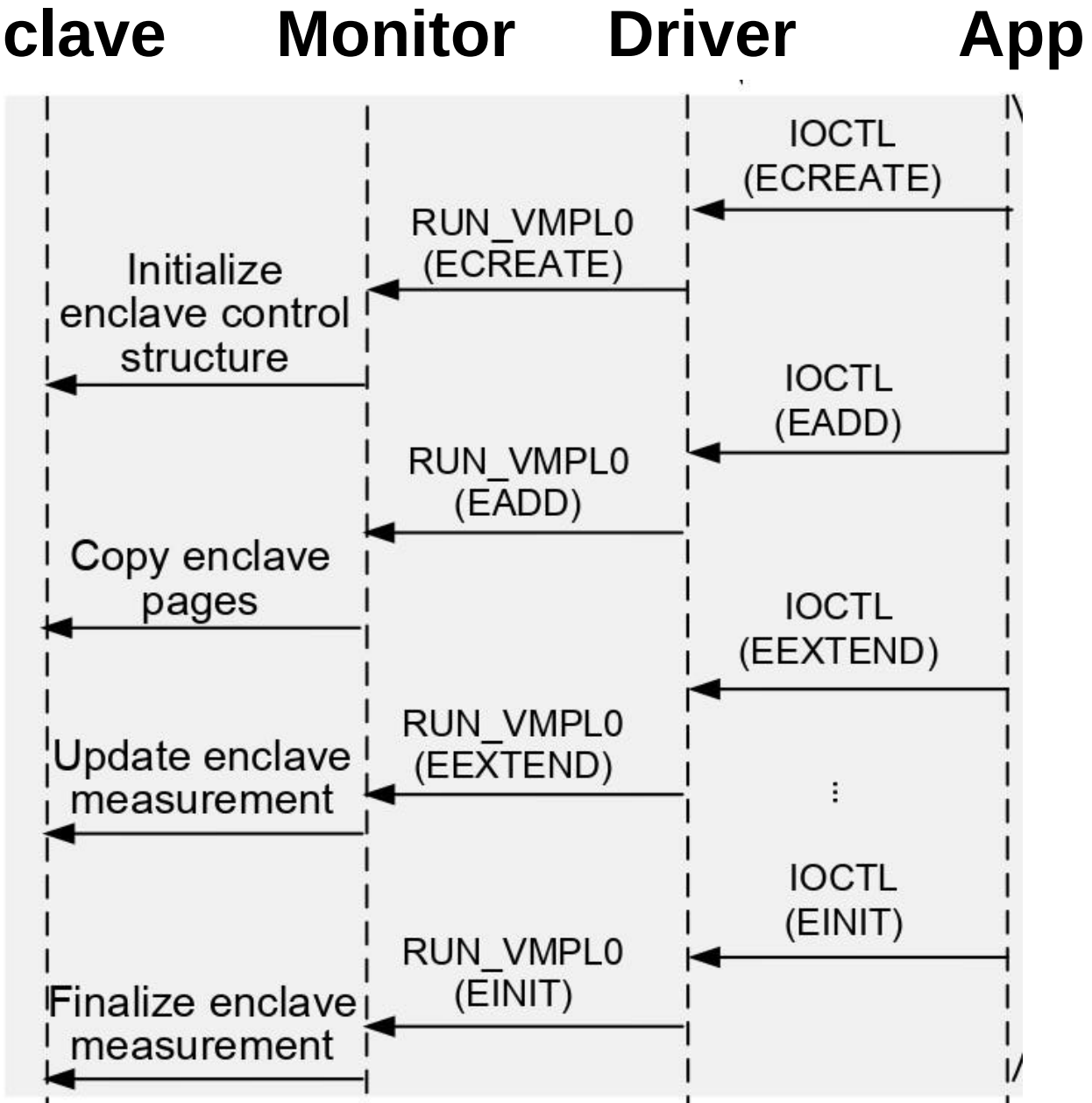
Architecture

- **Key Features:**
  - **Security Monitor**
  - **Memory Isolation**
  - **Integrity of Enclave**



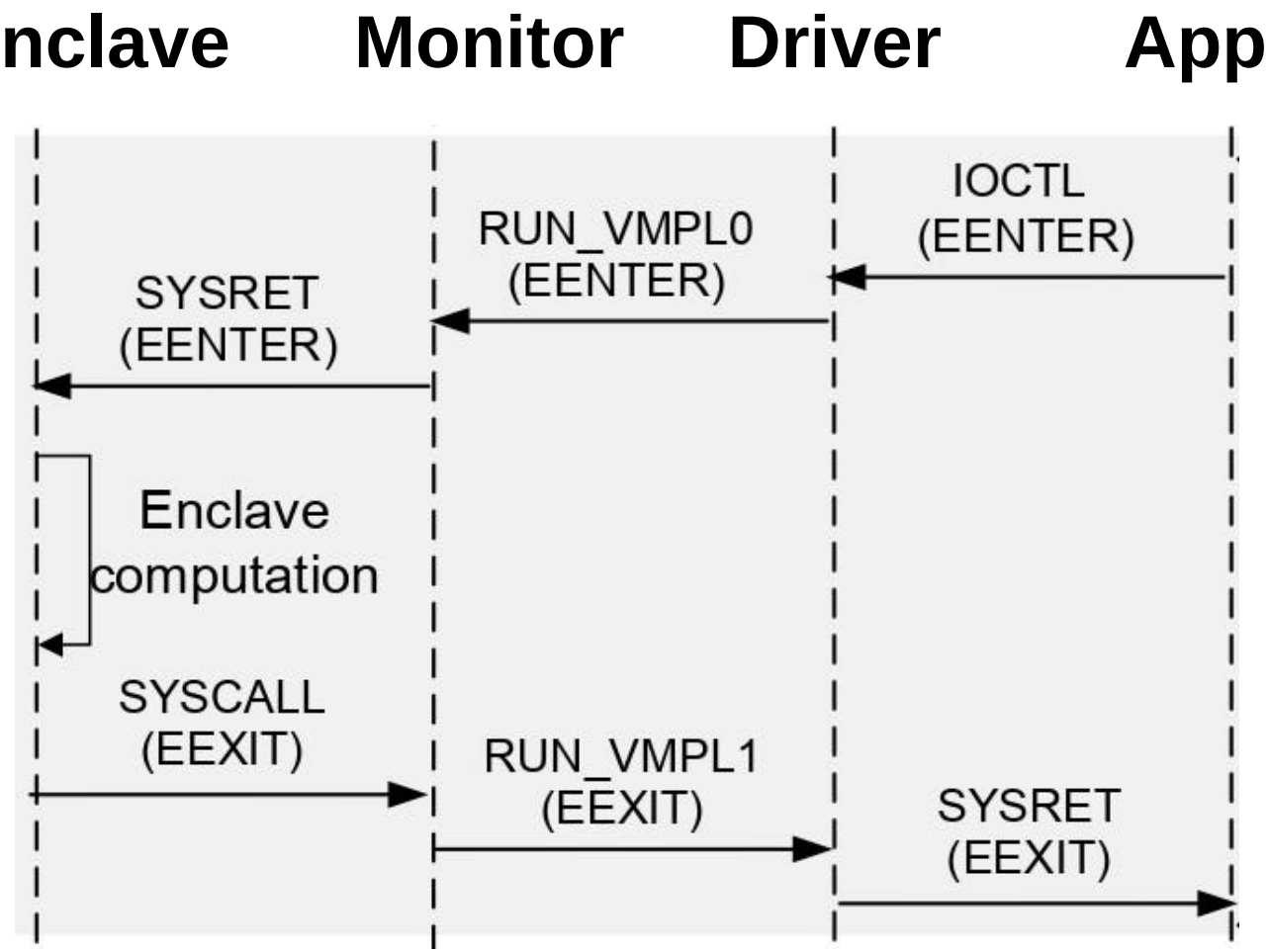
Security Monitor: enclave life cycle management

- **Instruction-level emulation**
  - ECREATE/EADD/EINIT
  - Enclave/Monitor: **VMPL0**
  - Driver/App: **VMPL1**



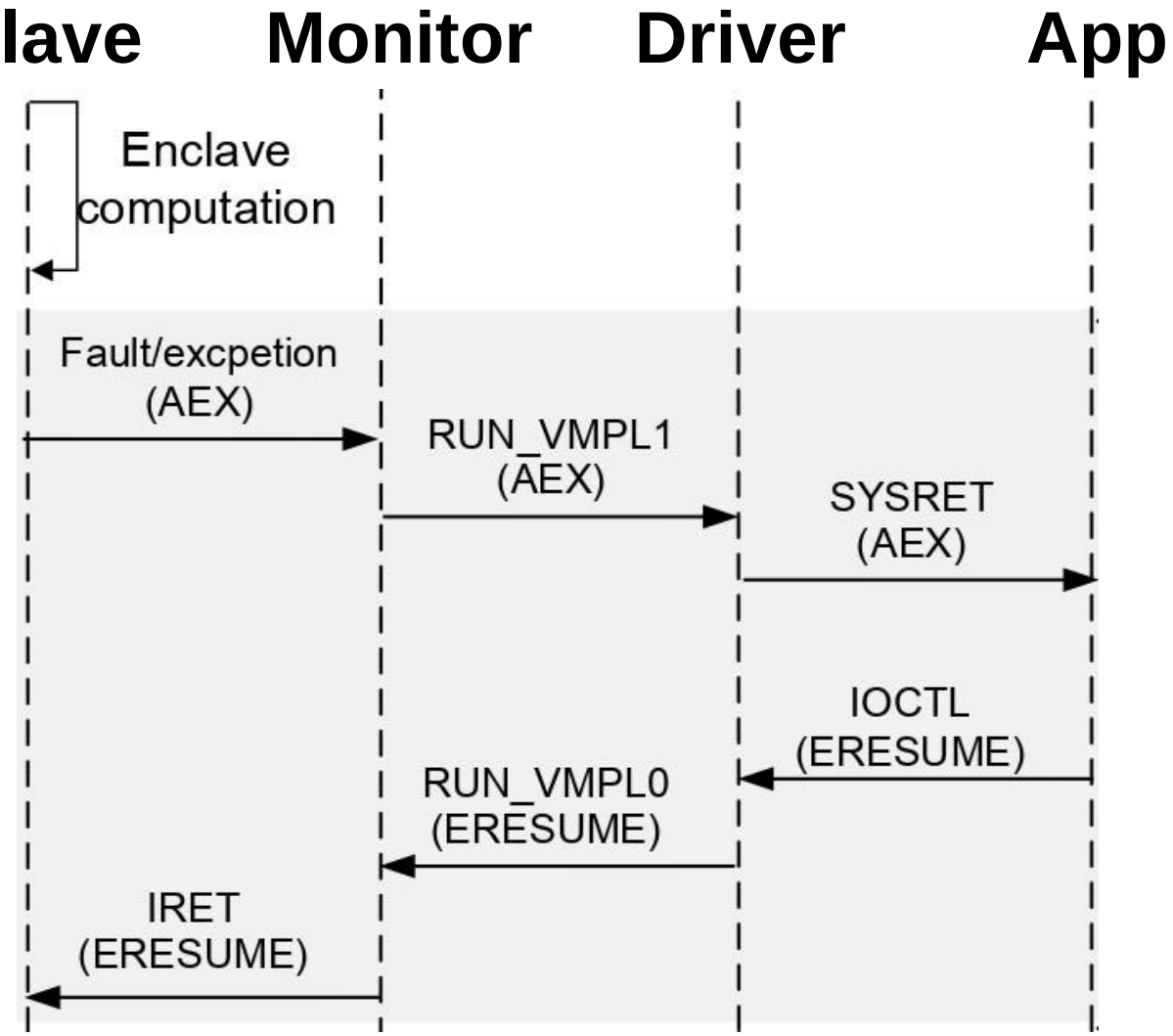
Security Monitor: enclave life cycle management

- **Instruction-level emulation**
  - Sync Entry/Exit
  - Enclave/Monitor: **VMPL0**
  - Driver/App: **VMPL1**



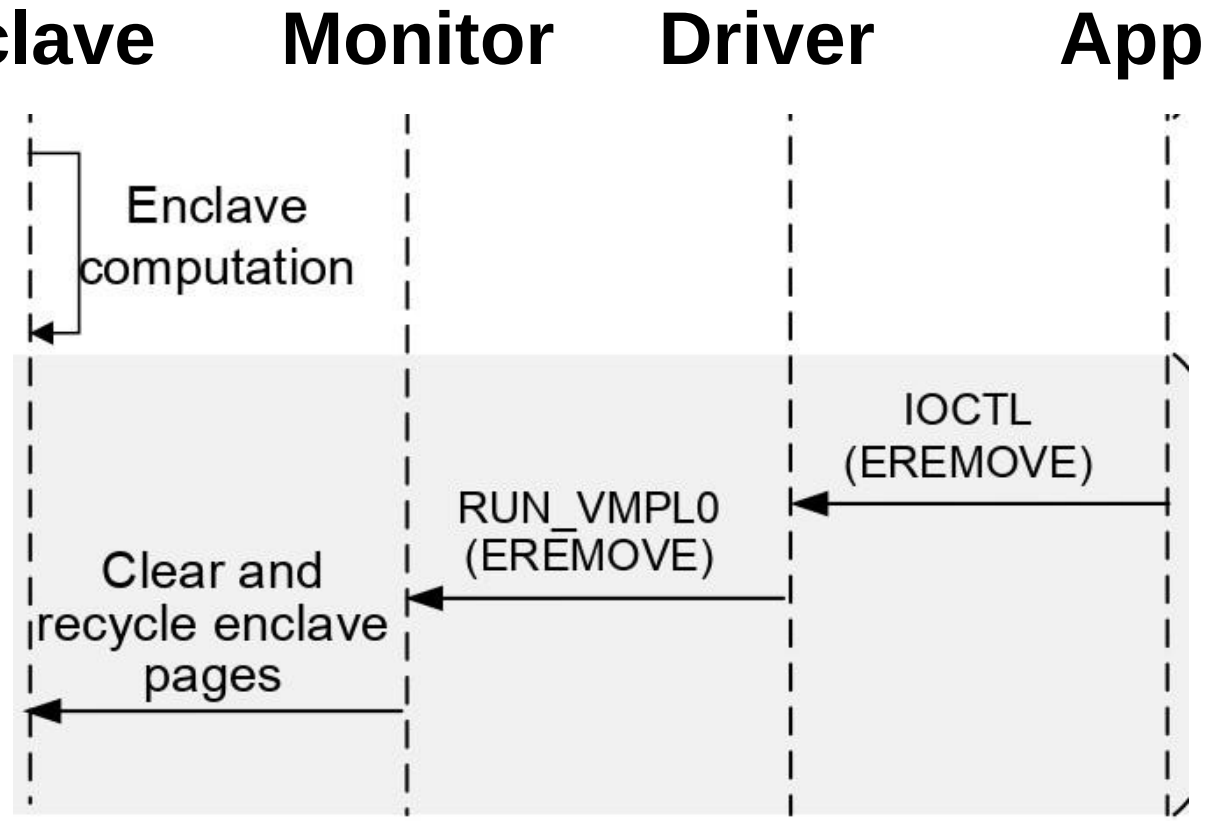
Security Monitor: enclave life cycle management

- **Instruction-level emulation**
  - Async Entry/Exit
  - Enclave/Monitor: **VMPL0**
  - Driver/App: **VMPL1**



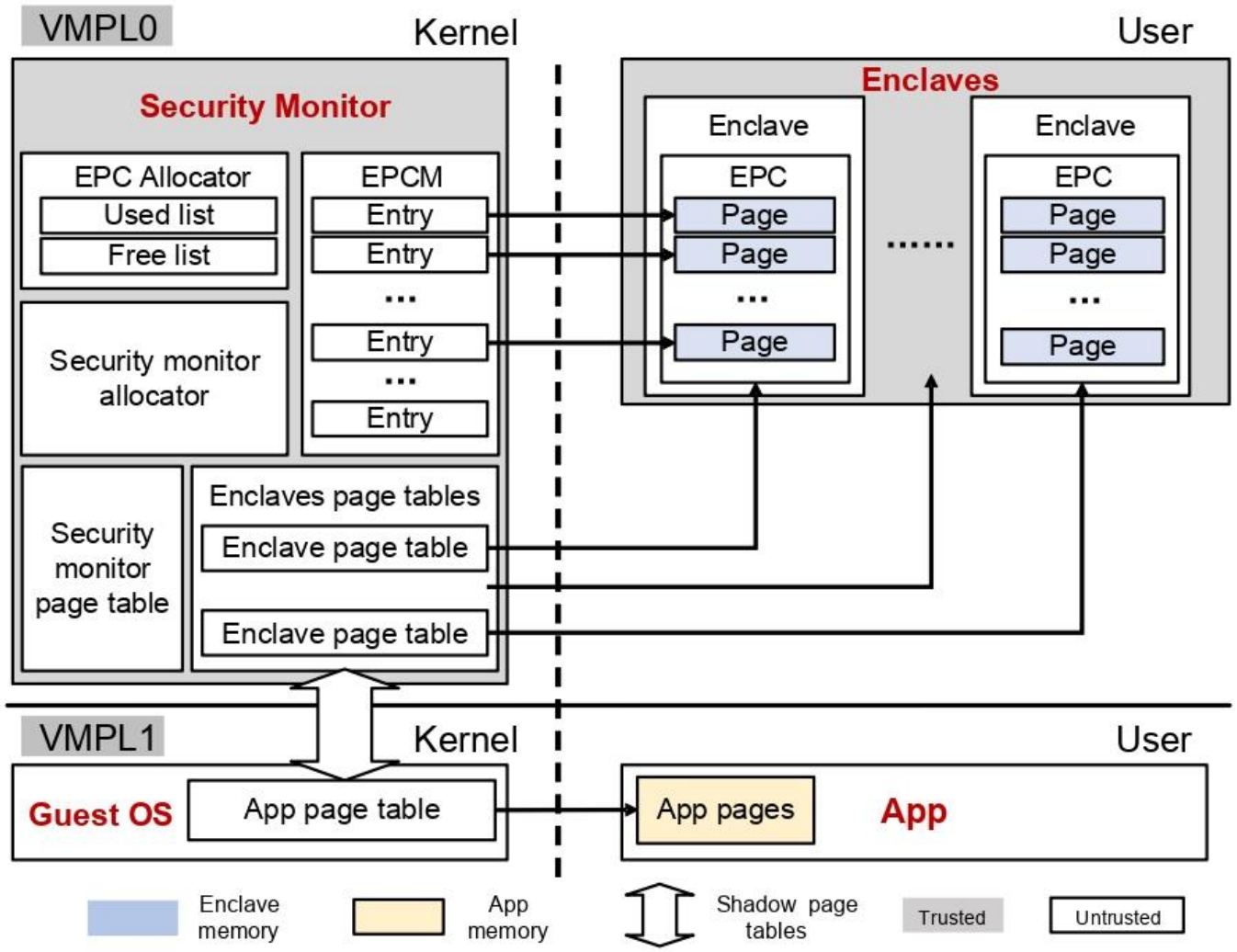
Security Monitor: enclave life cycle management

- **Instruction-level emulation**
  - Enclave Destroy
  - Enclave/Monitor: **VMPL0**
  - Driver/App: **VMPL1**



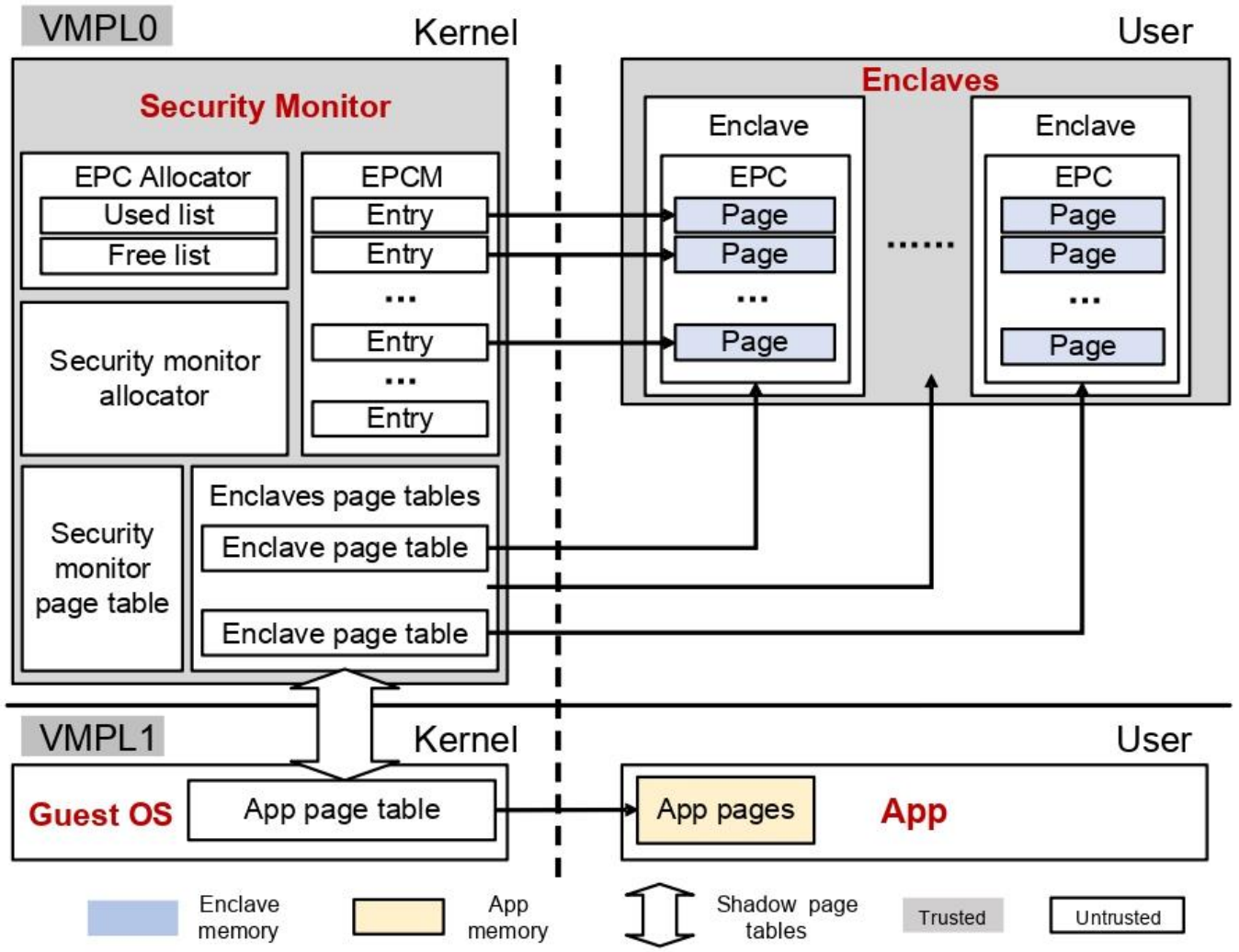
Memory Isolation

- **Booting:**
  - Security monitor initializes the Guest OS, and set up its page table
  - Then the security monitor hands over control to BIOS of Guest OS



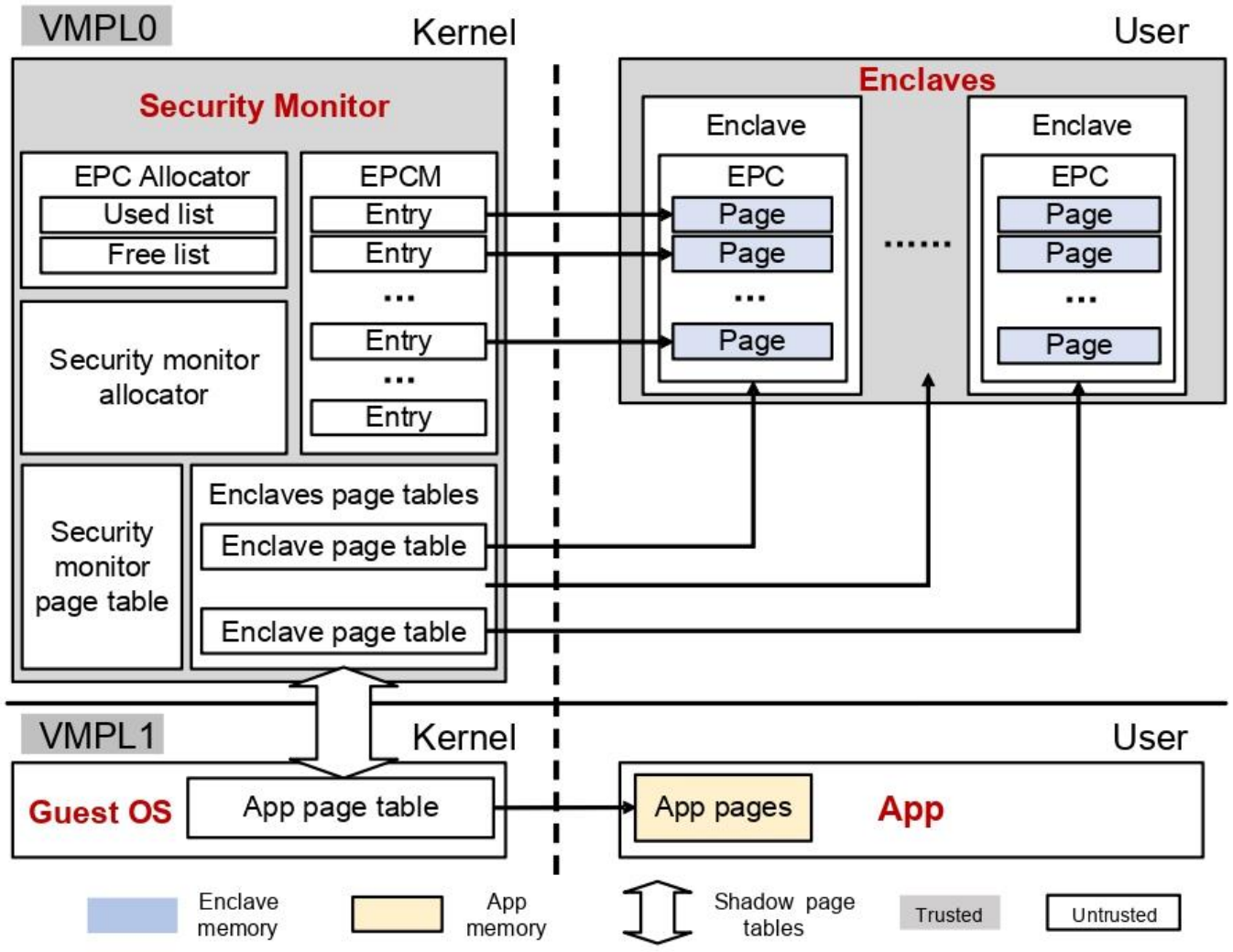
Memory Isolation

- **EPC Management:**
  - EPC pages are allocated and recorded with EPCM by security monitor
  - Security monitor handles page fault of Enclaves
  - Shadow Page Tables: same gVA-to-gPA mappings for Apps and Enclaves



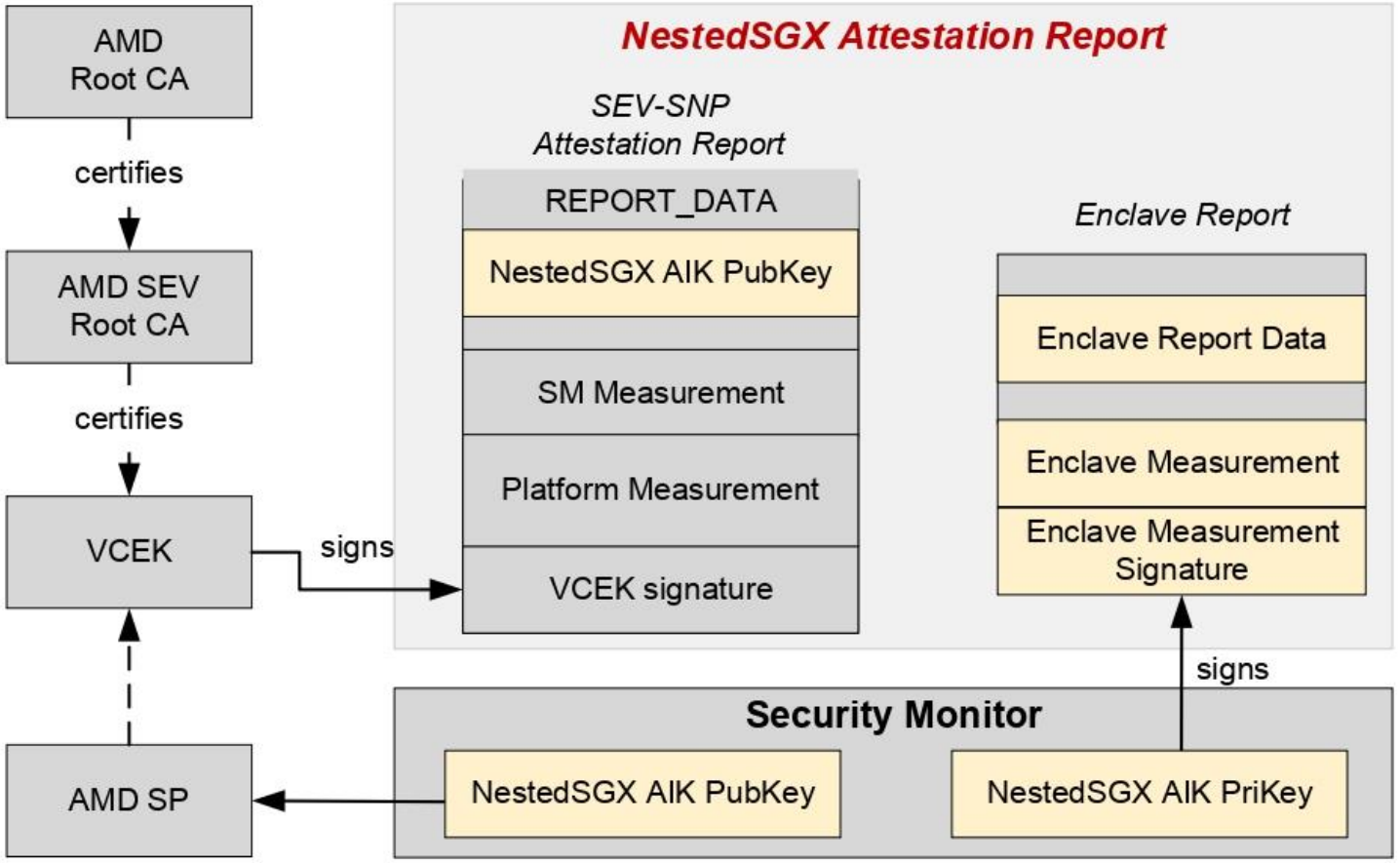
Memory Isolation

- **Isolation of Enclaves**
  - Guest OS can't access Secure memory in VMPL0
  - Security monitor ensures the page table of Enclaves does not map to any gPA of security monitor and guest OS



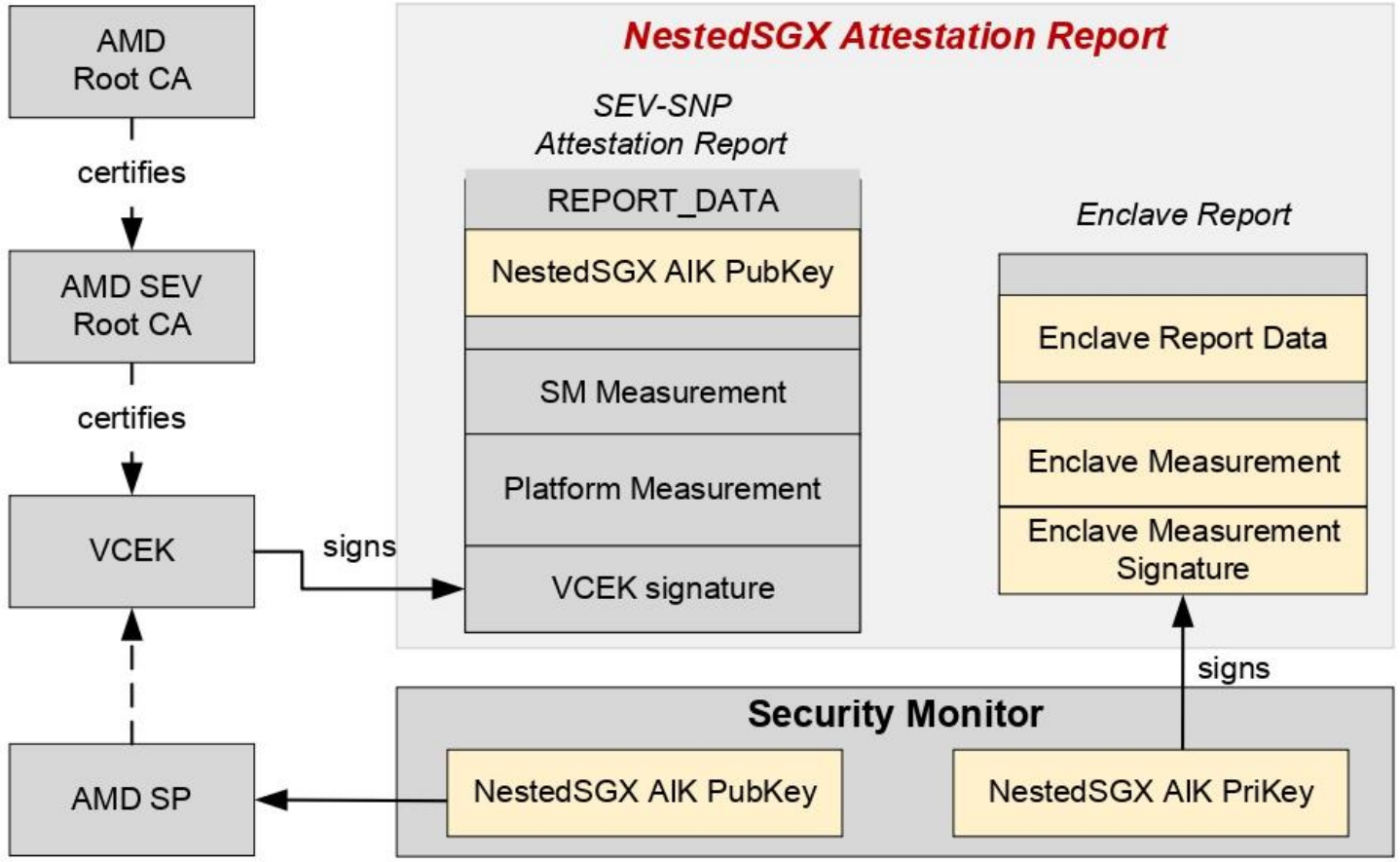
Measurement, attestation, sealing

- **Enclaves:**
  - EGETKEY/EREPORT



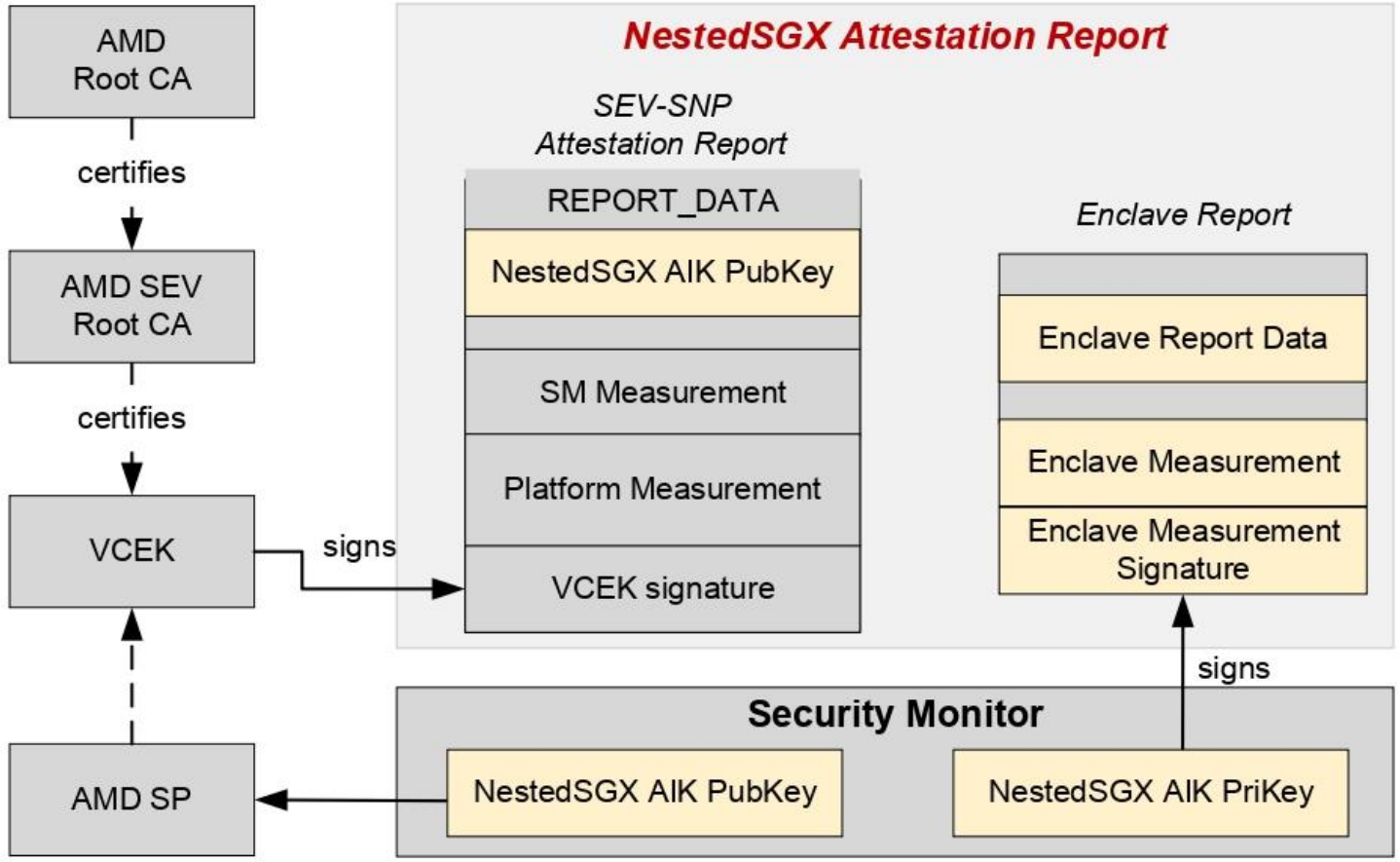
Measurement, attestation, sealing

- **Enclaves:**
  - EGETKEY/EREPORT
- **Chain of Trust:**
  - Generate **AIK**
    - Pub key for CVM
    - Pri key for Quoting Enclave
  - AMD SP decrypts CA and generate attestation report

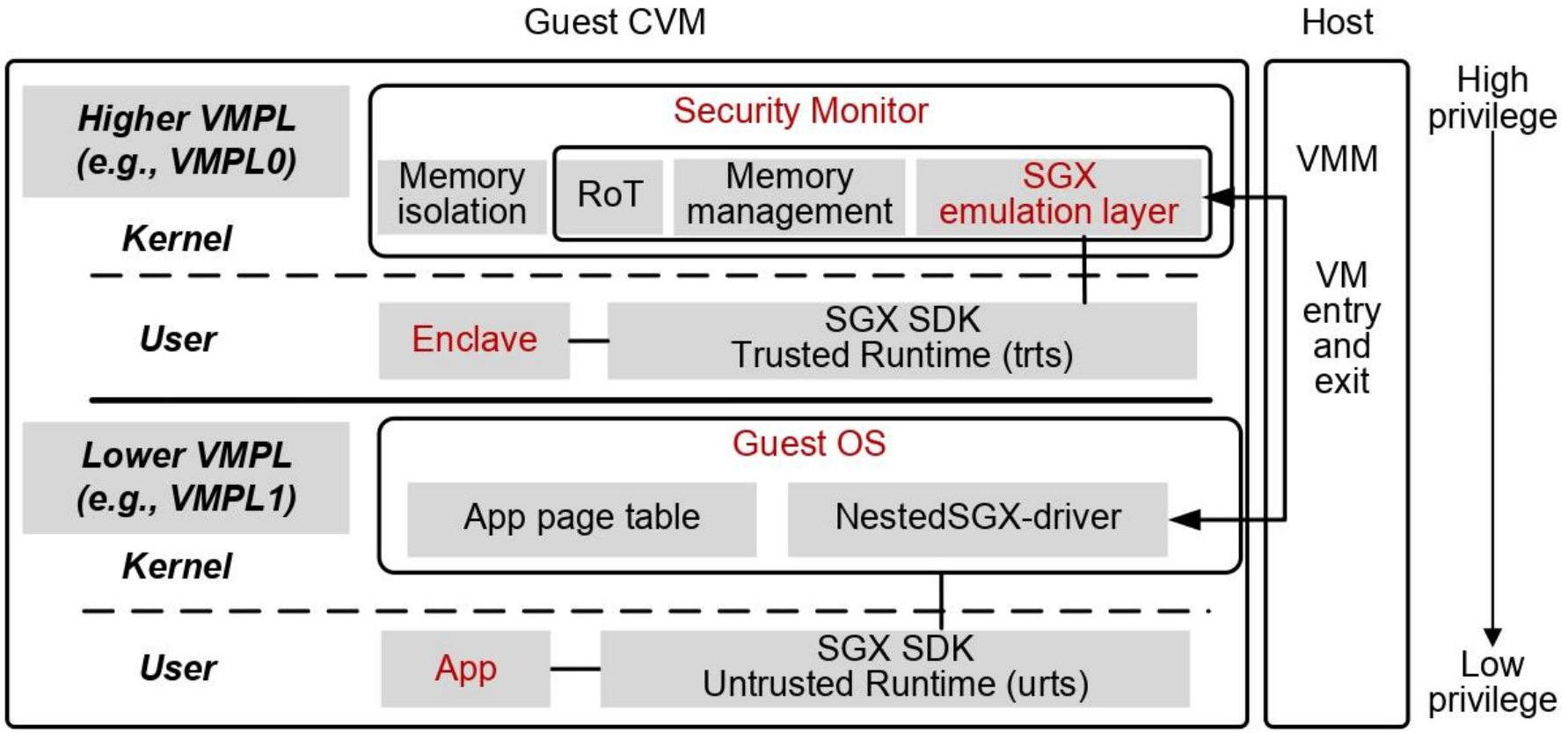


Measurement, attestation, sealing

- **Enclaves:**
  - EGETKEY/EREPORT
- **Chain of Trust:**
  - Generate **AIK**
    - Pub key for CVM
    - Pri key for Quoting Enclave
  - AMD SP decrypts CA and generate attestation report
- **Sealing:** MSG_KEY_REQ

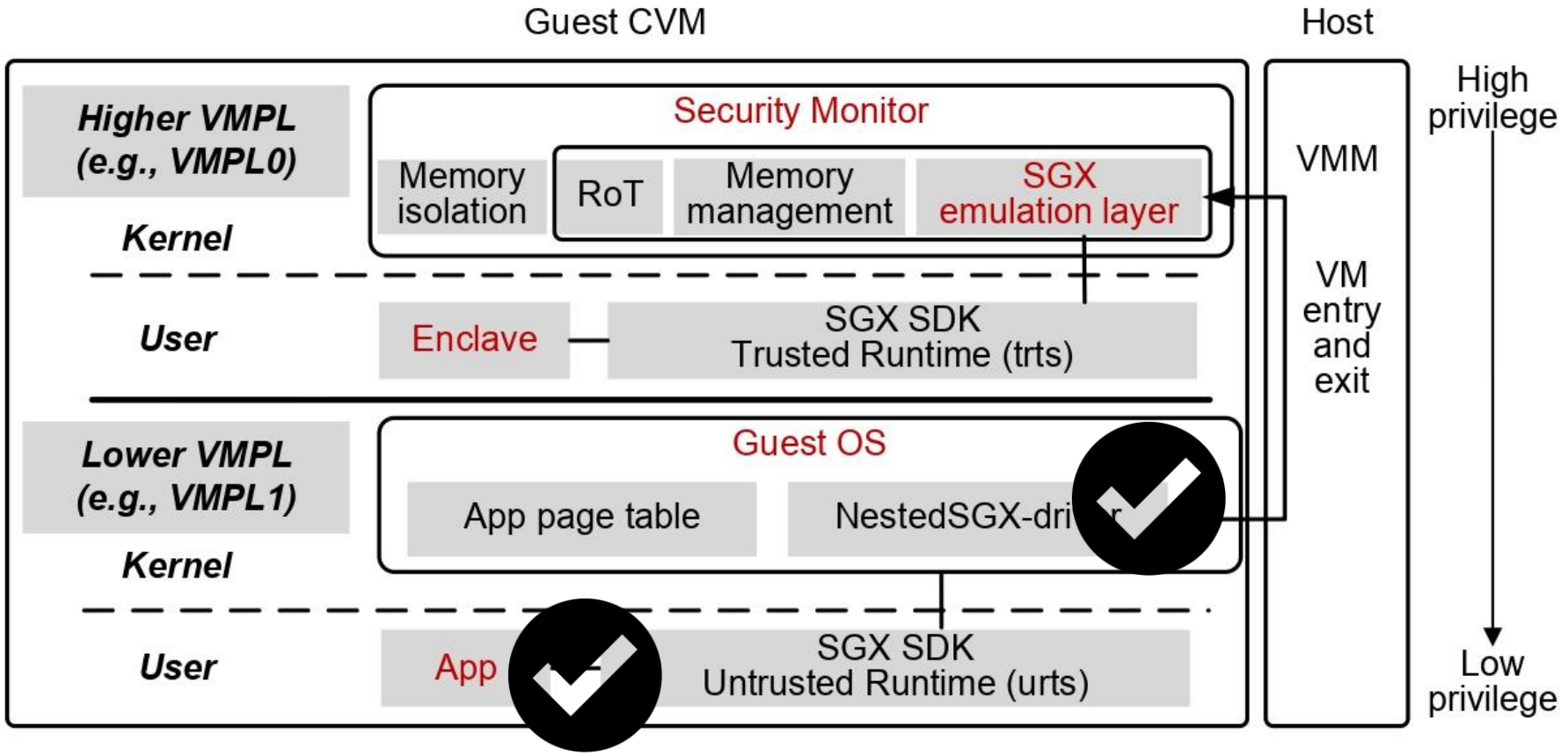


Security Analysis



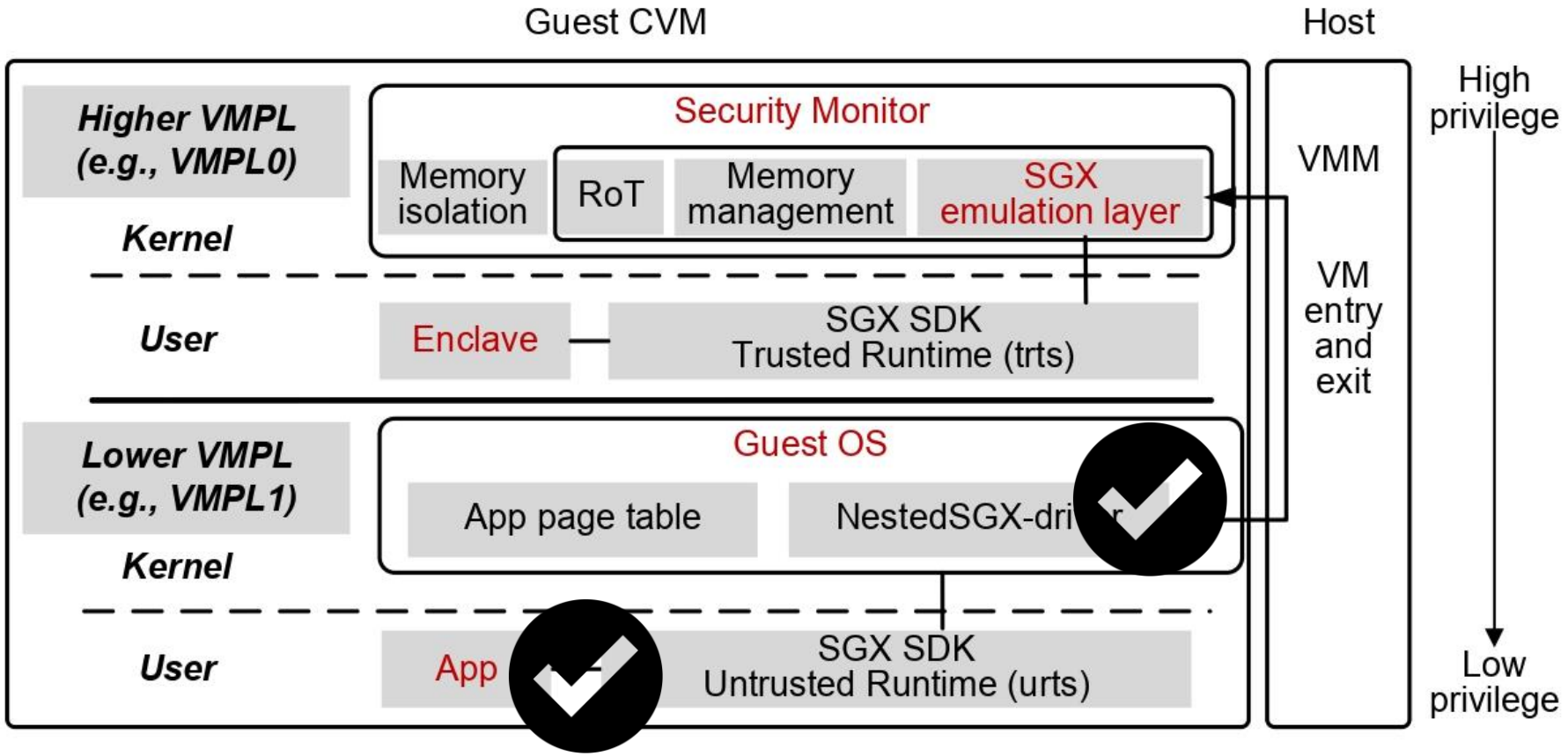
- **Untrusted App, Guest OS**
  - Separate page tables
- **NestedSGX Driver**
  - Solely tasked with switching VMPL
  - not introduce additional attack vendors

Security Analysis



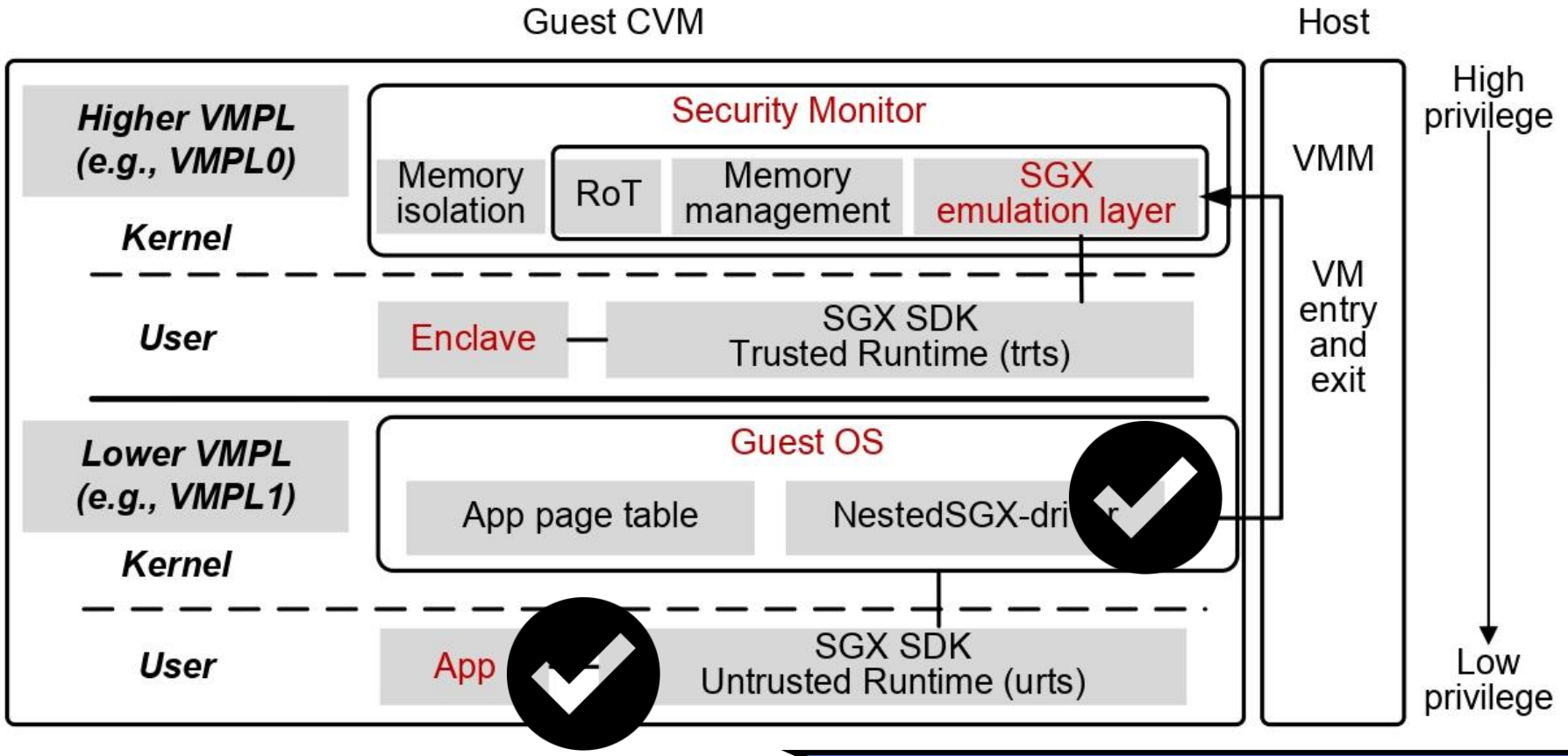
- **Untrusted App, Guest OS**
  - Separate page tables
- **NestedSGX Driver**
  - Solely tasked with switching VMPL
  - not introduce additional attack vendors

Security Analysis



- **Untrusted host VMM**
  - Refuse to switch VMPL (DoS)
  - Observe Ecall, Ocall, AEX patterns (Also exists in SGX)

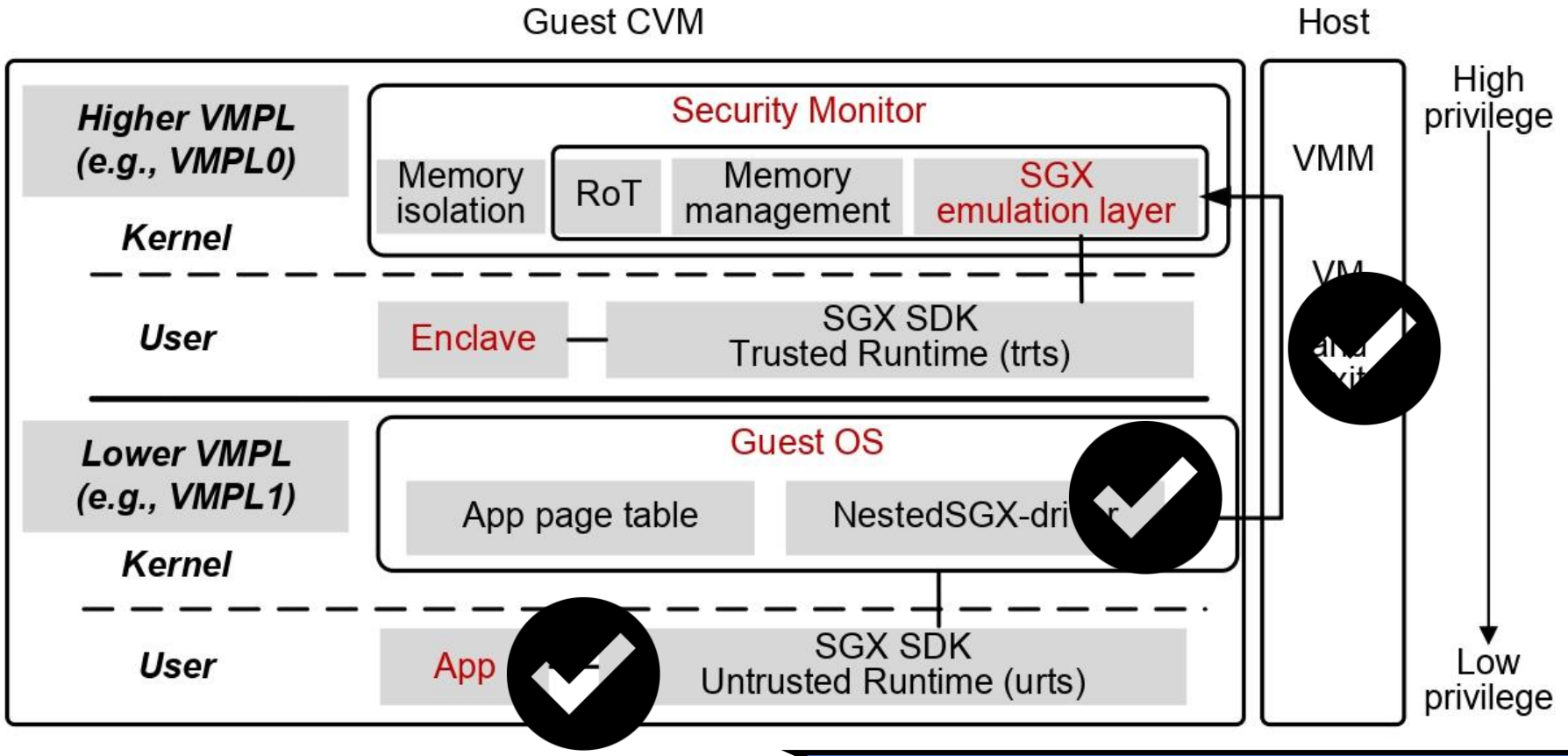
Security Analysis



- **Untrusted host VMM**
  - Refuse to switch VMPL (DoS)
  - Observe Ecall, Ocall, AEX patterns (Also exists in SGX)

Out of Our Scope

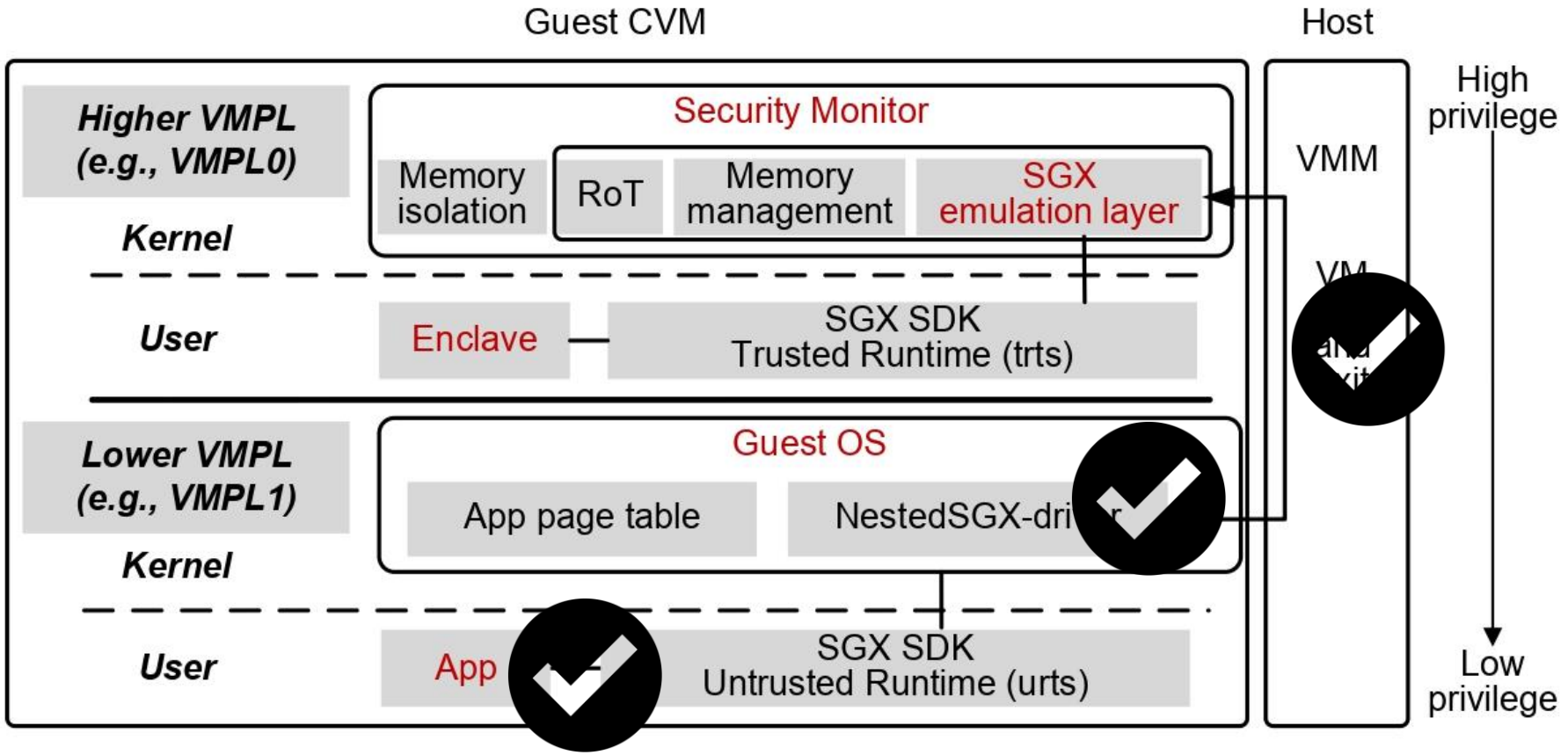
Security Analysis



- **Untrusted host VMM**
  - Refuse to switch VMPL (DoS)
  - Observe Ecall, Ocall, AEX patterns (Also exists in SGX)

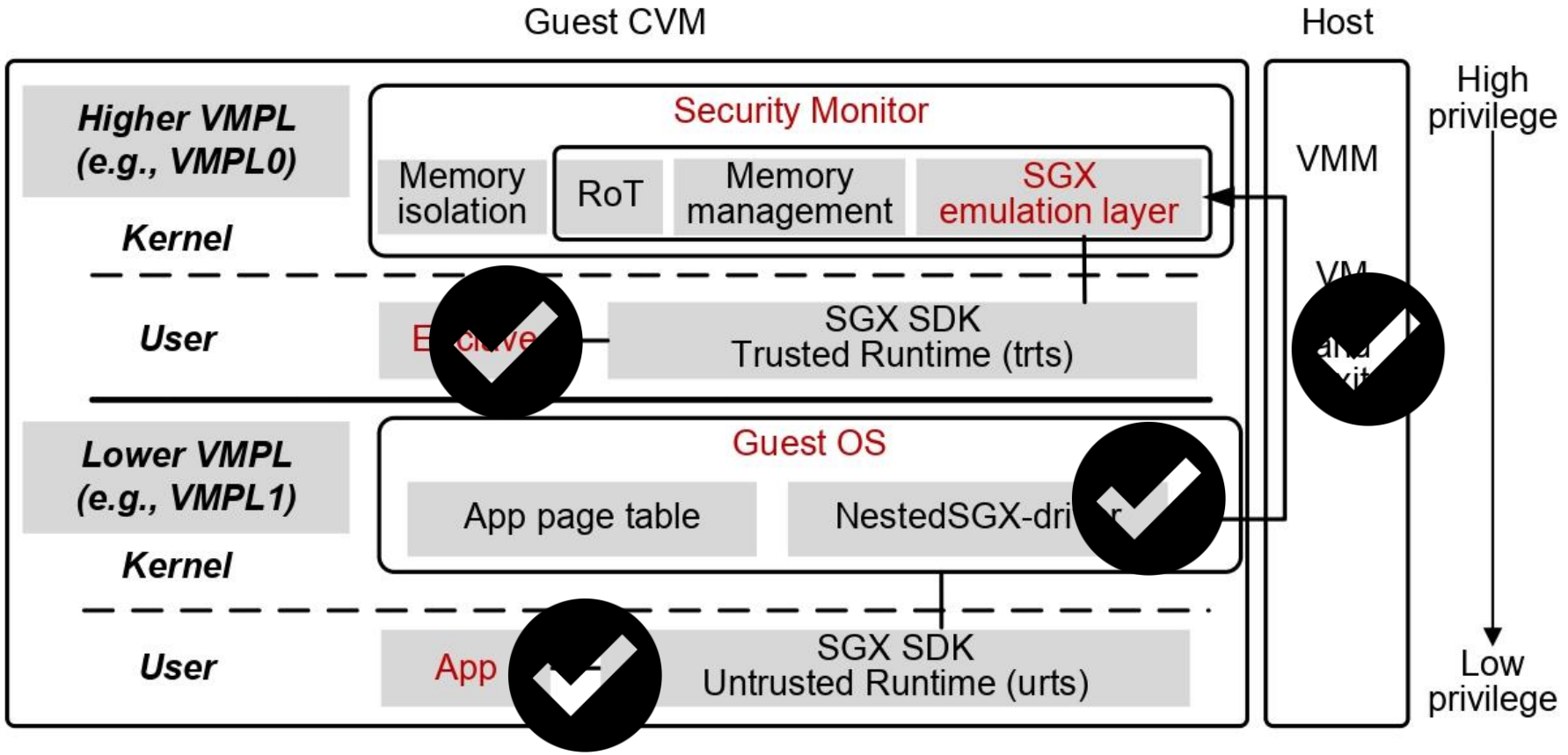
Out of Our Scope

Security Analysis



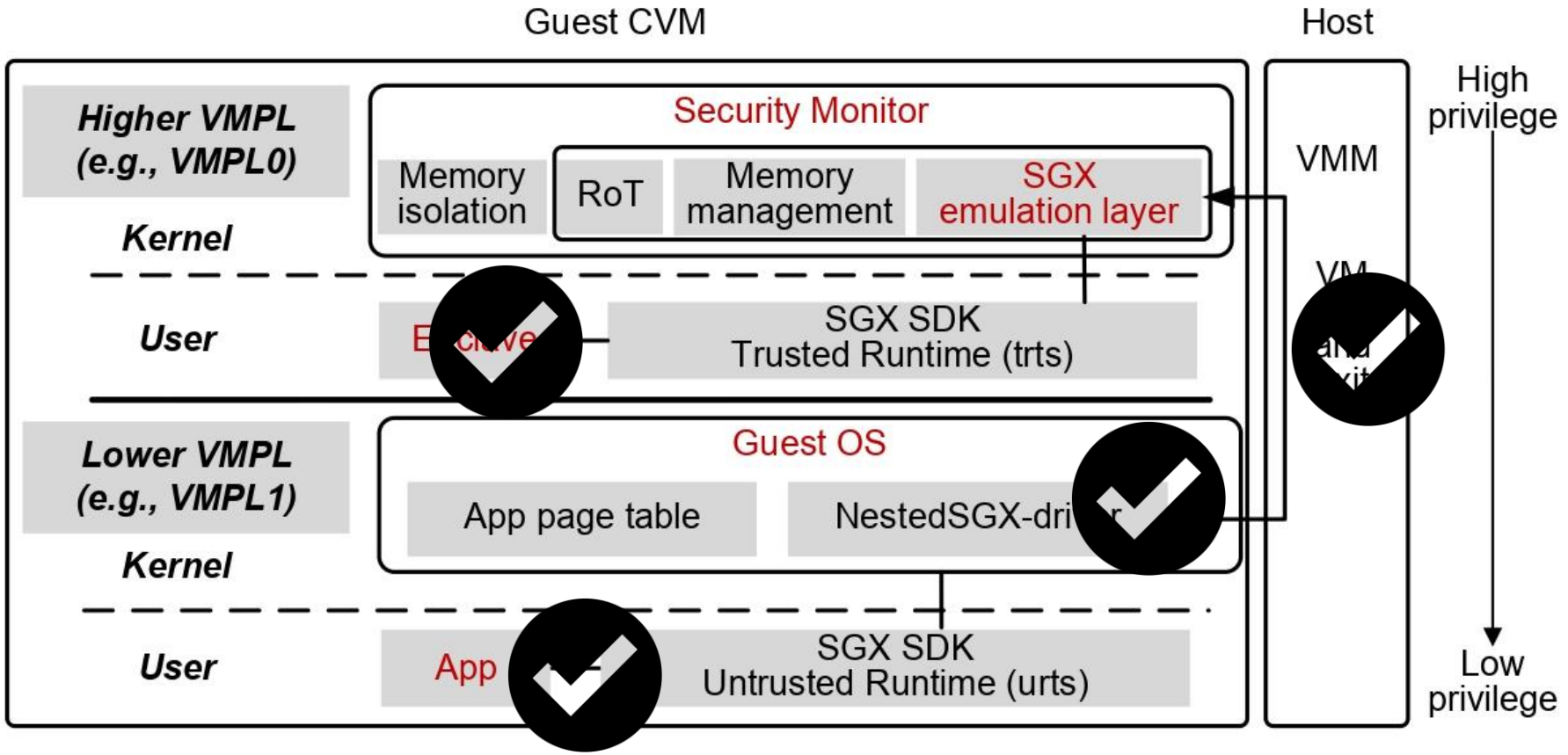
- **Untrusted Enclave**
  - User mode: Security Monitor controls its page table
  - Can’t bypass the Security Monitor

Security Analysis



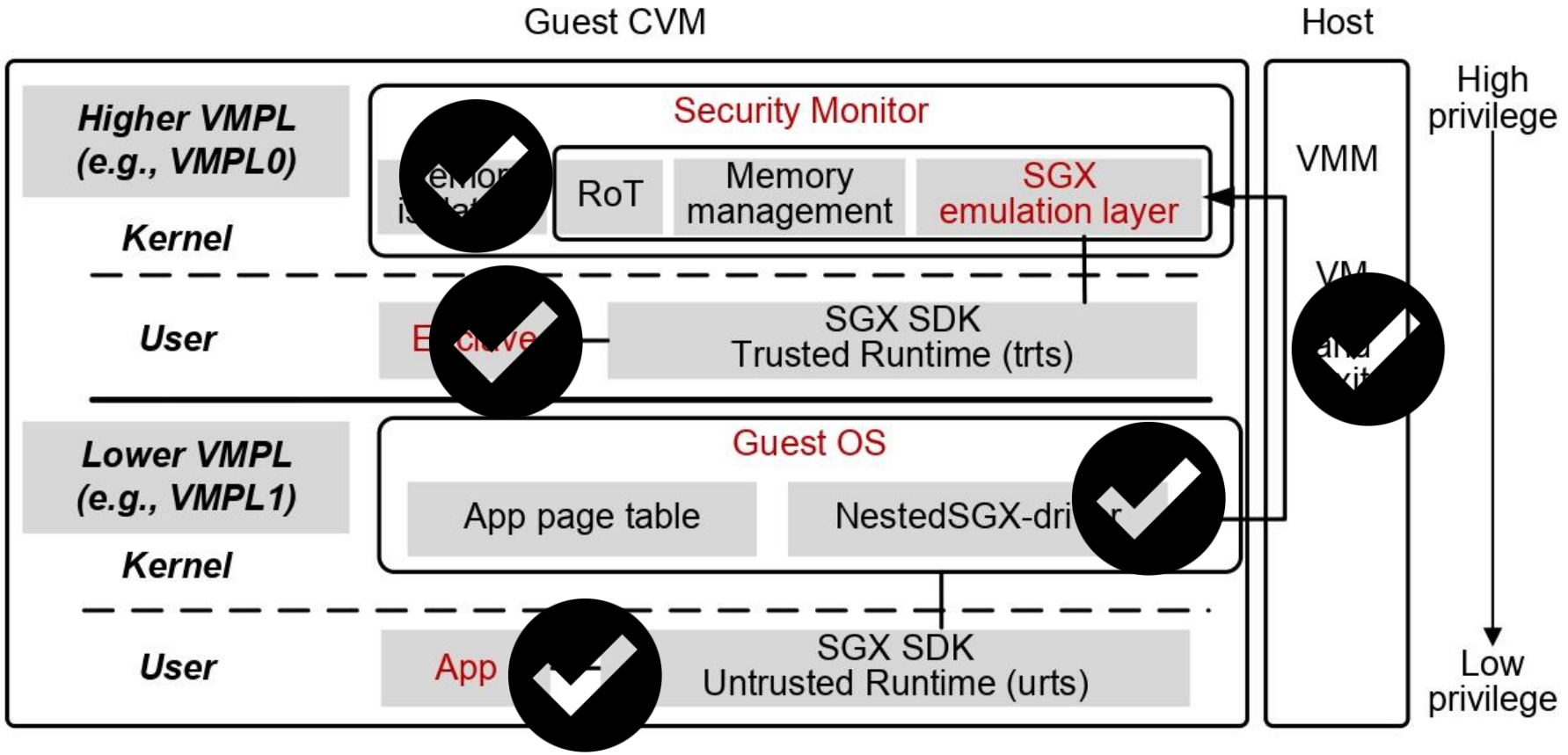
- **Untrusted Enclave**
  - User mode: Security Monitor controls its page table
  - Can’t bypass the Security Monitor, no data can be shared

Security Analysis



- **Chain of Trust Analysis:**
  - NestedSGX AIK private key: inaccessible to other components
  - SNP_REPORT_REQ: can only initiated from the kernel mode
  - Attestation Generation: lower or equal privilege than current

Security Analysis



- **Chain of Trust Analysis:**
  - NestedSGX AIK private key: inaccessible to other components
  - SNP_REPORT_REQ: can only initiated from the kernel mode
  - Attestation Generation: lower or equal privilege than current

Implementation

- ~7500 LoC
  - Occlum, Monitor: Rust
  - Others: C

Component	Description	Line of Code
Security monitor	Emulation of SGX data structures, instructions and AEX	5,500
NestedSGX-driver	Handling switches between App and security monitor	800
SGX SDK & Occlum	Replacing SGX instructions with IOCTL and system calls, HotCalls	1,200
Total		**7,500**

- Based on open-source Linux Secure VM Service Module (**SVSM**) framework

- Deployed on a server with two AMD EPYC 7543 CPU
  - Ubuntu 22.04
  - Kernel: svsm-preview-guest-v3(**CVM**), svsm-preview-host-v3(**Host**)
  - Linux SGX SDK: v2.20
  - Occlum: v0.29.7

Evaluation:

- **Baseline for comparison:**
  - Intel SGX Hardware
  - Simulation mode SGX

- **Benchmarks:**
  - Micro Benchmarks
  - Real-world evaluations

Evaluation: Micro Benchmarks

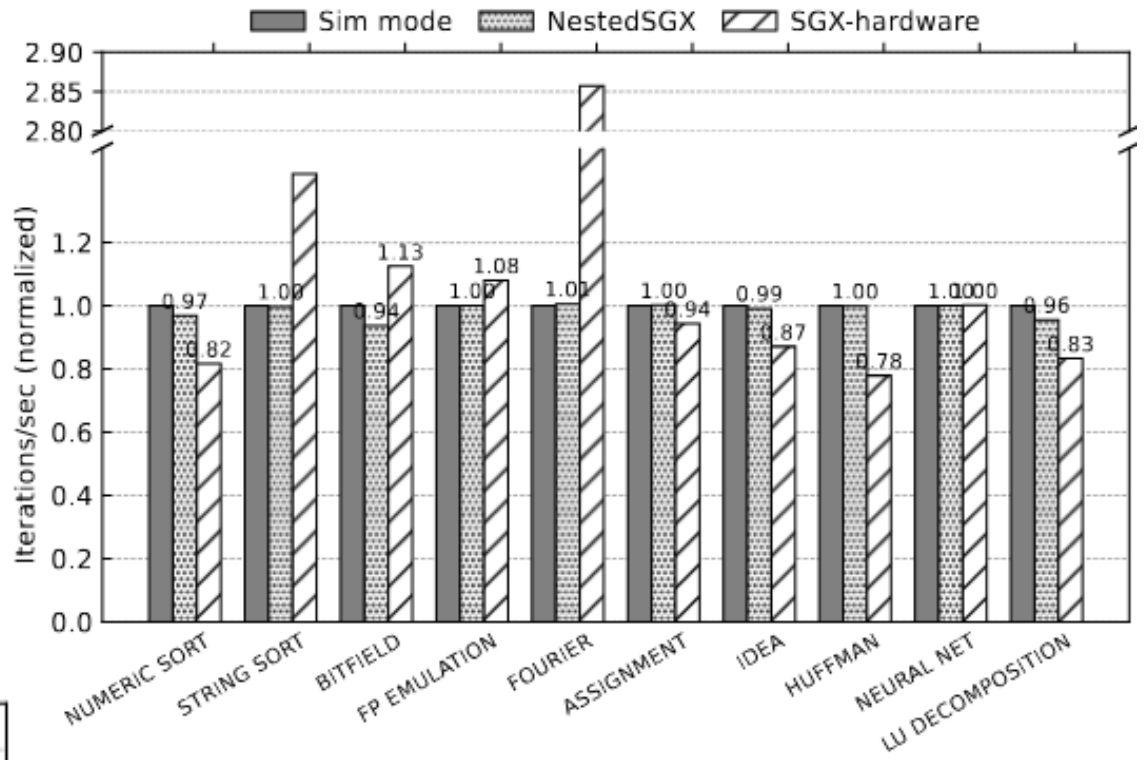
- **SGX-Leaf Instruction:**
  - Rust Implementations of cryptographic crates is a little slower (**EGETKEY/EREPORT**)
- **Context switches:**
  - VMPL switch: 19400 cycles

		vSGX	NestedSGX
ENCLS	ECREATE	3,719 us	8.4 us
	EADD	1,421 us	8.0 us
	EEXTEND	987 us	33 us
	EINIT	811 us	46 us
	EREMOVE	1,014 us	7.8 us
	EAUG	990 us	7.9 us
	EBLOCK	841 us	9.2 us
	ELDB/ELDU	1,958 us	9.3 us
	EMODPR	1,071 us	9.9 us
	EWB	1,819 us	7.9 us
ENCLU	EGETKEY	5.0 us	17 us
	EREPORT	19 us	30 us

	Intel SGX	vSGX	NestedSGX
ECALL	10,988 cycles (4.1 us)	$\approx 1,500$ us	33,584 cycles (12 us)
OCALL	9,337 cycles (3.5 us)	-	32,014 cycles (11 us)

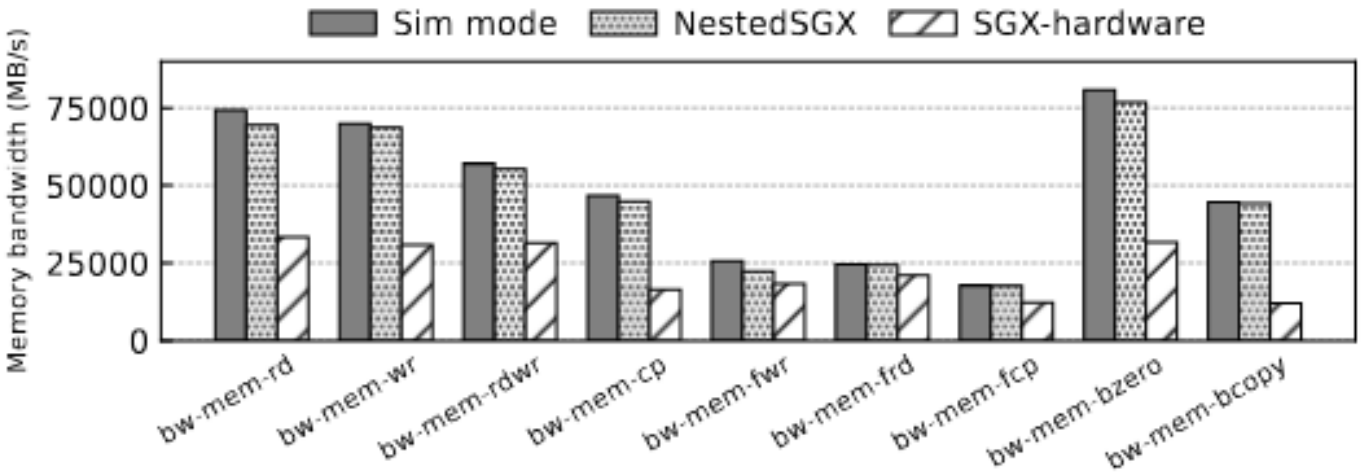
Evaluation: Micro Benchmarks

- **Linux/Unix nbench**



(a) Linux/Unix nbench

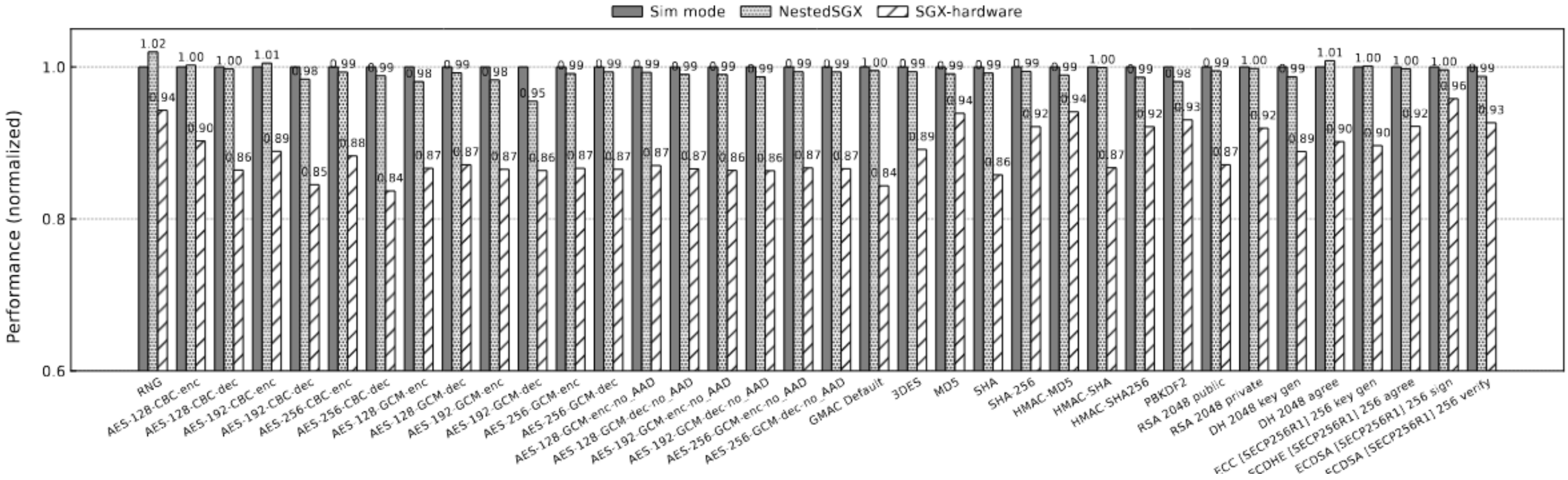
- **Lmbench**



(b) Lmbench

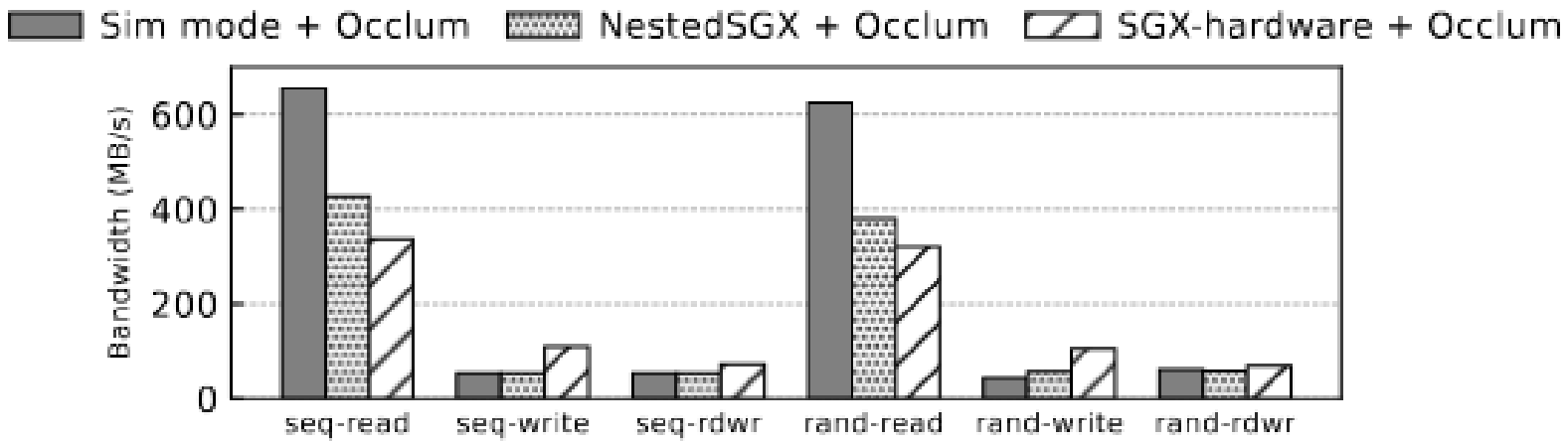
Evaluation: Micro Benchmarks

- **WolfSSL**



Evaluation: Micro Benchmarks

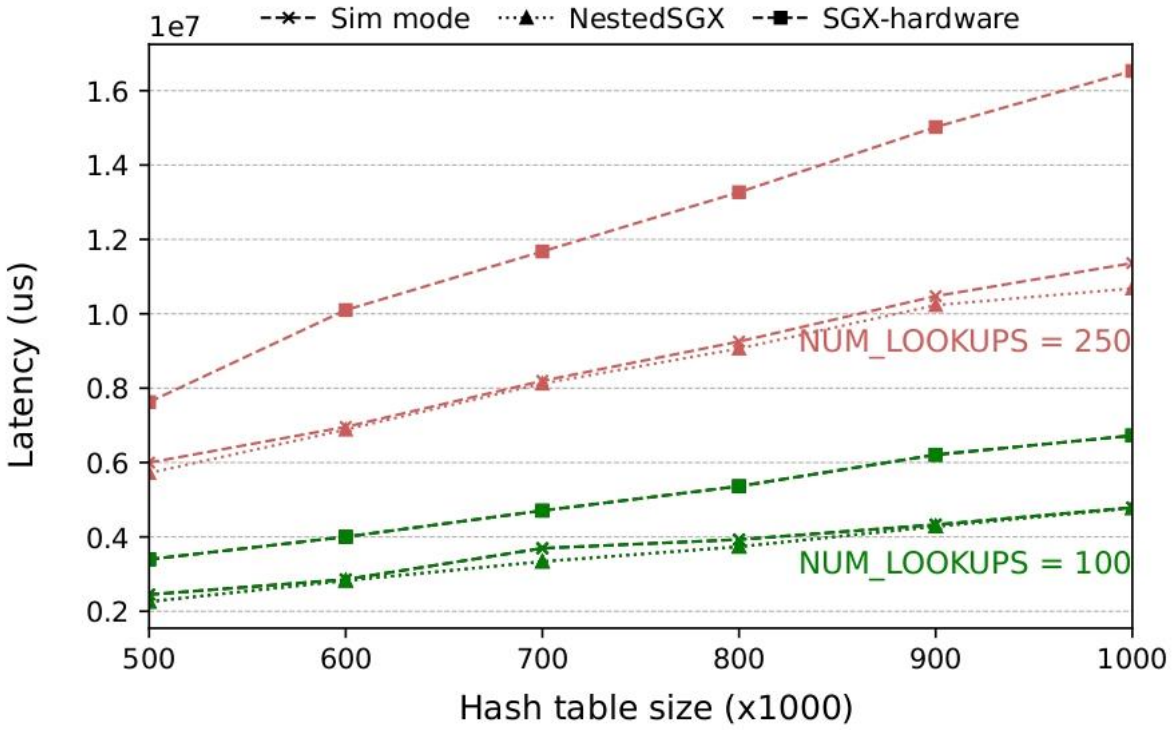
- **Flexible I/O Tester (With Occlum)**

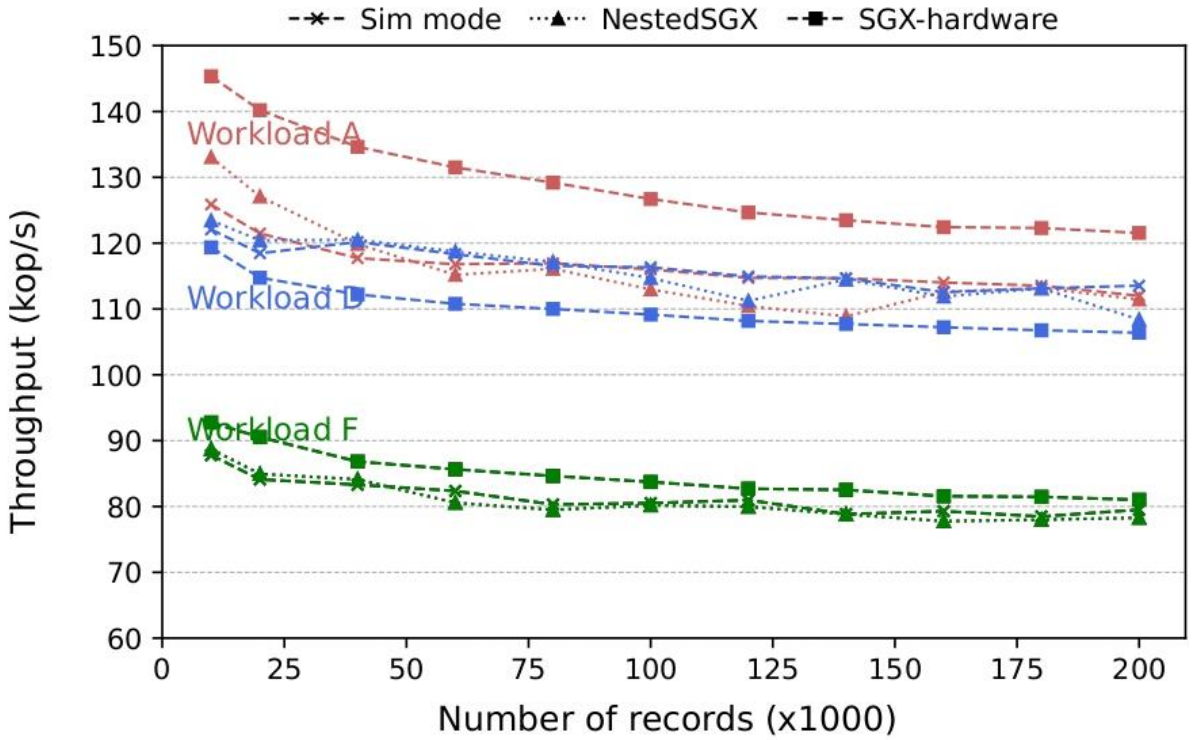


(c) FIO

Evaluation: Real World Applications

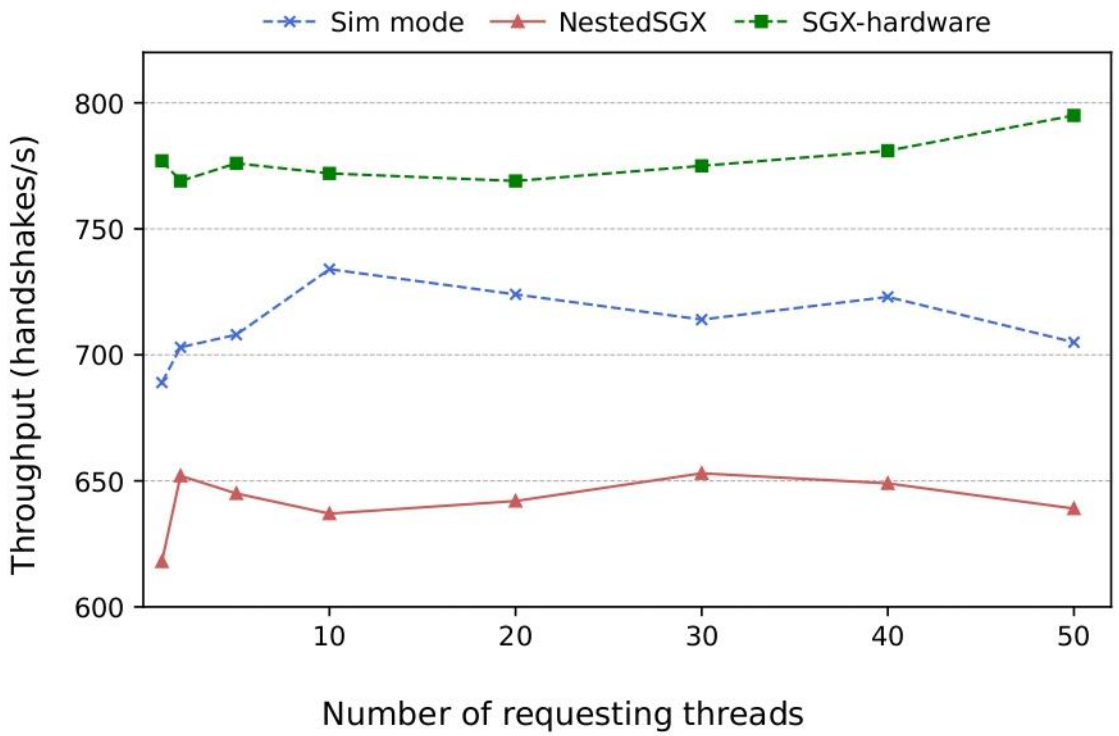
- **Hash Join**
- **SQLite**

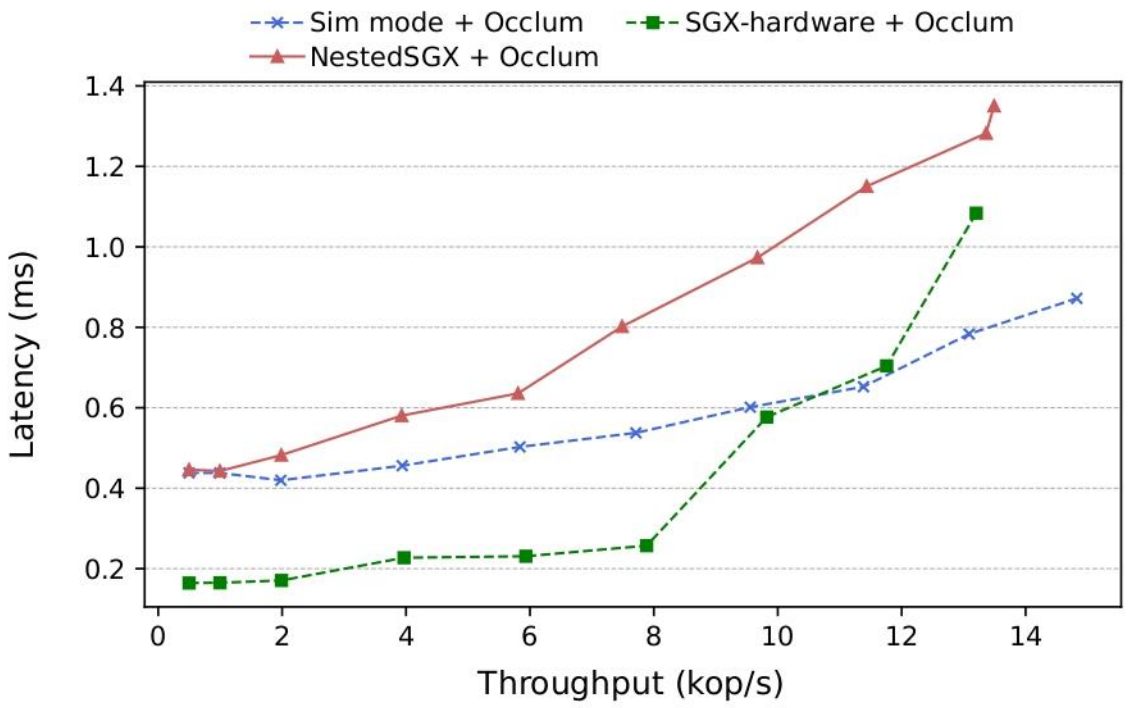




Evaluation: Real World Applications

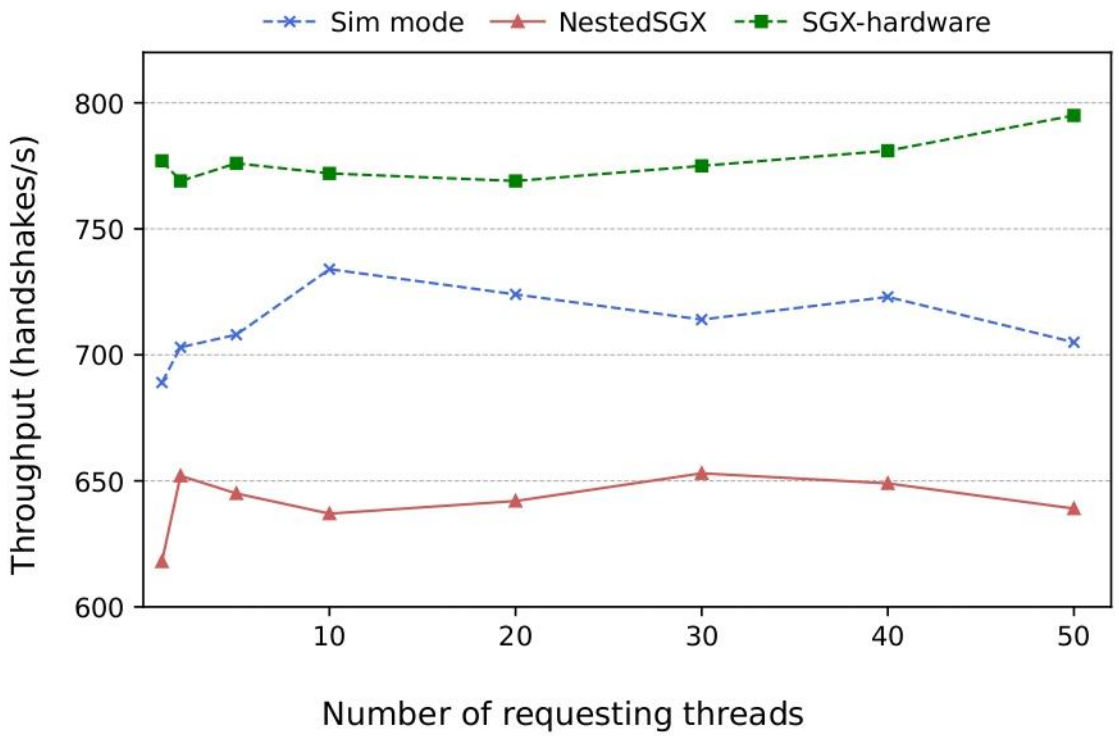
- TLS Server
- Redis (With Occlum)

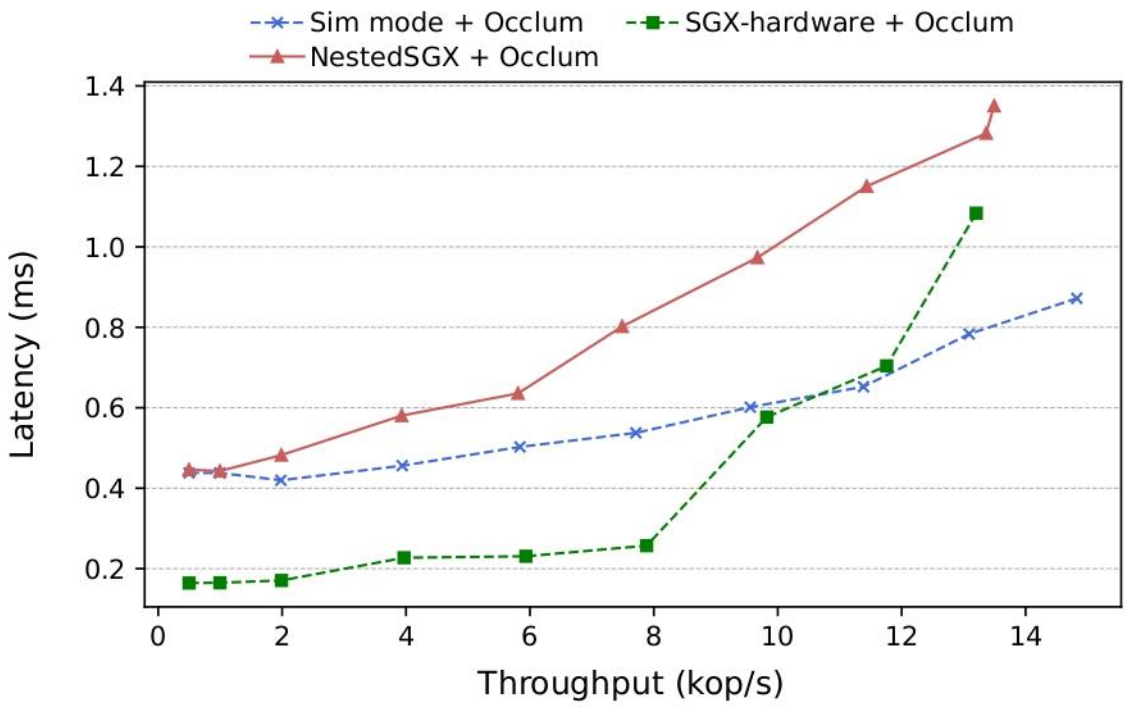




Evaluation: Real World Applications

- TLS Server
- Redis (With Occlum)





Limitation and Future Work

- **Limitations exist!!**
  - Not all SGX model features are supported
    - *user_check, memory sharing*
  - Scheduling of enclave threads is lacking
  - Applications written without SDK and Occlum are not supported

Limitation and Future Work

- **Limitations exist!!**
  - Not all SGX model features are supported
    - *user_check, memory sharing*
  - Scheduling of enclave threads is lacking
  - Applications written without SDK and Occlum are not supported

- **Future?**
  - Extend NestedSGX to other CVM platforms
  - Other TEE abstractions

Conclusion

- **NestedSGX**
  - Hybrid VM-Based and Process-Based TEE models
  - Build trust enclaves within **Confidential VMs**

- **A more compatible, secure, and efficient CVM framework**
  - Enable compatibility to current Intel SGX ecosystem (With Occlum)
  - Build trust within CVM
  - Achieve significant performance improvements

Thanks

Q&A

中国科学院大学

University of Chinese Academy of Sciences

songlinke@iie.ac.cn