Introduction to Keystone

Confidential Computing Consortium Webinar

Dayeol Lee dayeol@berkeley.edu

University of California, Berkeley

Keystone

CONFIDENTIAL COMPUTING CONSORTIUM

Confidential Computing

*"Confidential Computing is the protection of **data in use** by performing computation in a hardware-based **Trusted Execution Environment**."*

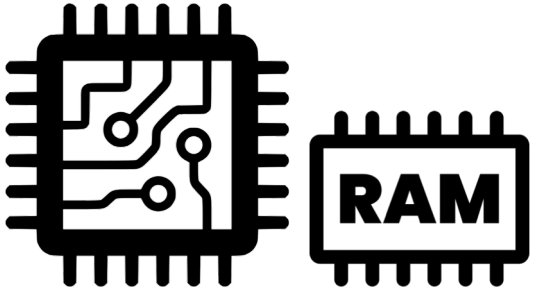
– Confidential Computing Consortium

Data at Rest

Encryption

Data in Transit

✅ Encryption



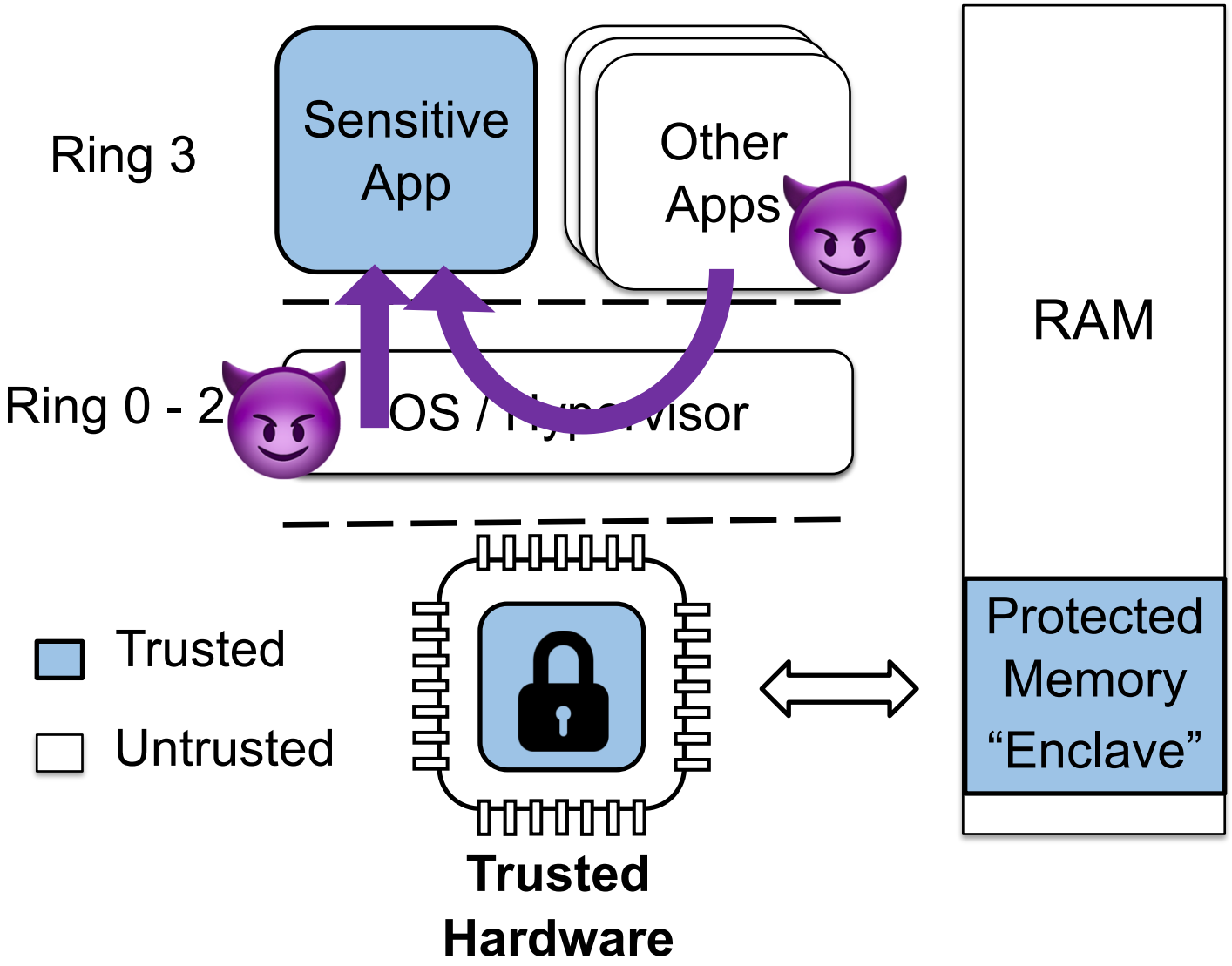
Data in Use

TEE

Keystone

Sources: https://findicons.com/icon/158405/hdd_2 https://www.visualpharm.com/free-icons/in%20transit-595b40b75ba036ed117d8d2f

Trusted Execution Environments (TEEs)

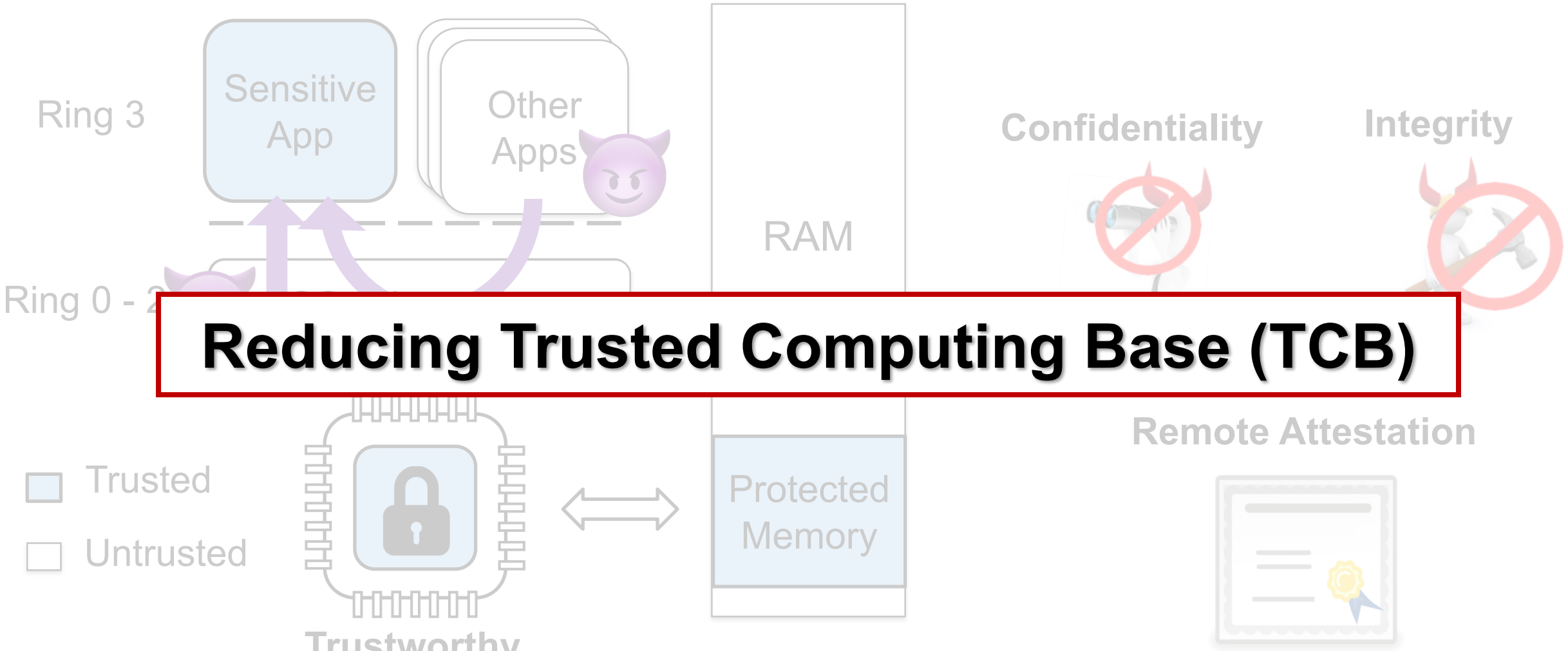


Confidentiality

Integrity

Remote Attestation

Trusted Execution Environments (TEEs)



Reducing Trusted Computing Base (TCB)

TEE Recipe

Application Interface
(e.g., API, Shield)

Development Tools
(e.g., SDK, Libraries)

Asylo

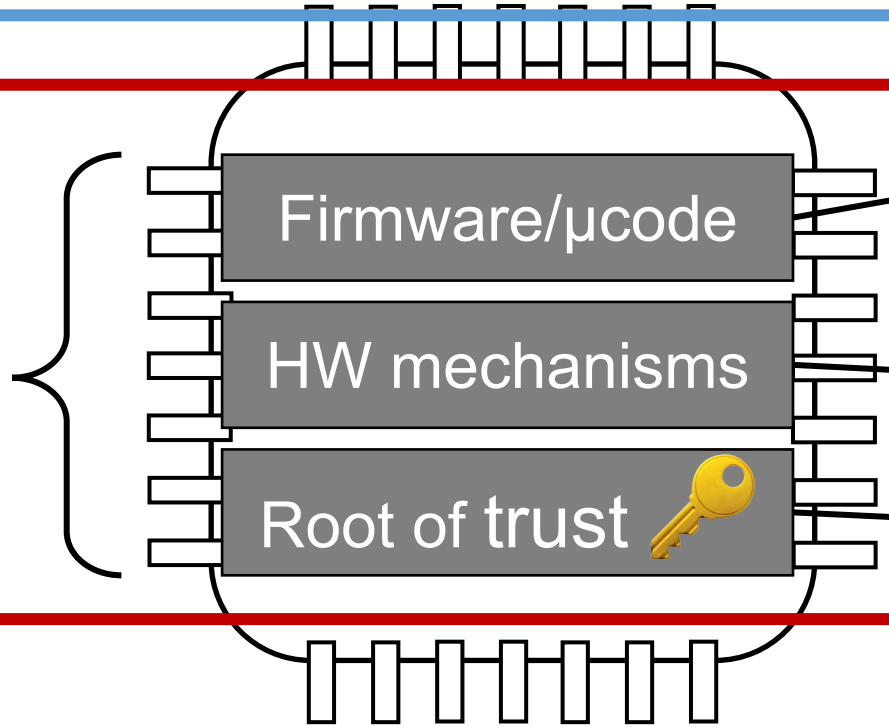
OP-TEE

GRAPHENE

SCONE

SGX SDK

Open Enclave SDK



(Formal)
Verification

SGX insts, Monitor, …

MMU, IOTLB, Bus, …

Crypto IP, Key Store, …

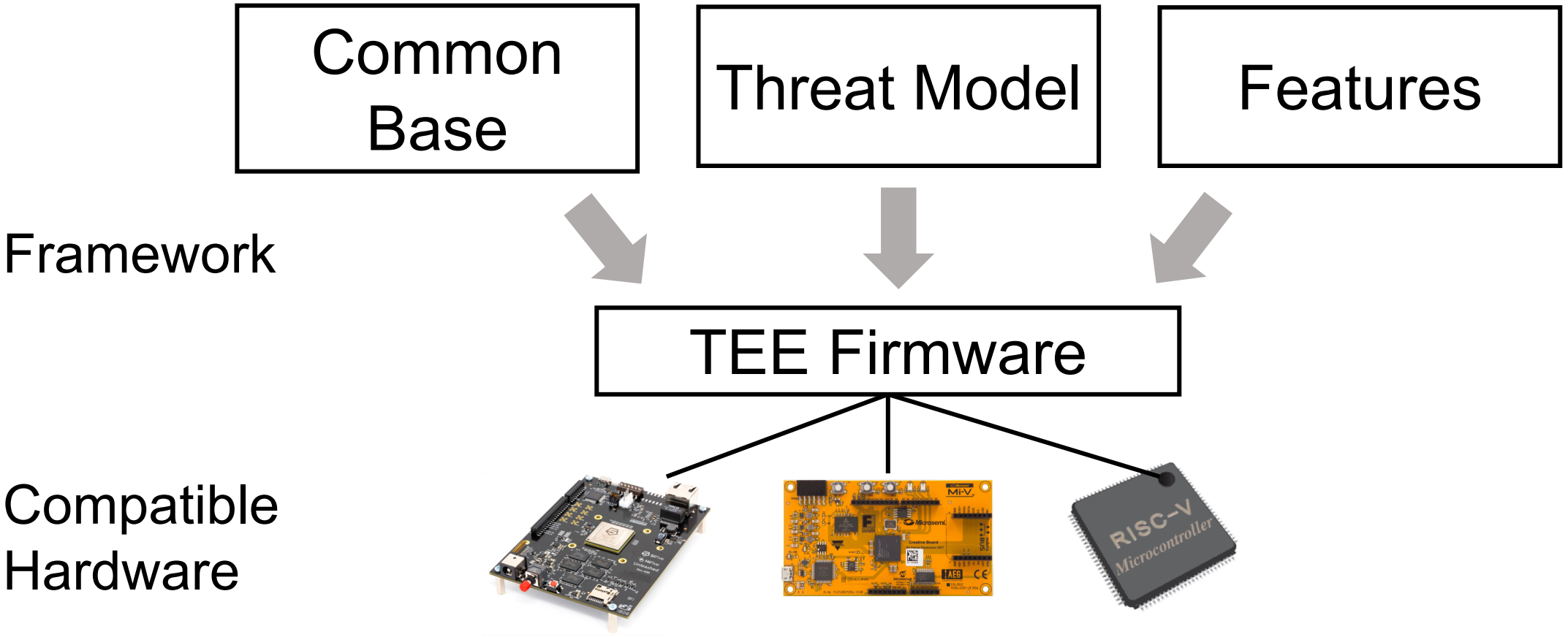
Building Custom TEE?

- Vendor TEE: Implemented on closed-source hardware/microcode
  - Slow iteration dictated by a company; researchers can't step forward
  - Any additional features/defenses need significant workaround
- Fixed design and threat model
  - Intel SGX – partitioned server/desktop apps (e.g., DRM, cryptography, etc)
  - ARM TZ – vendor-provisioned mobile apps (e.g., fingerprint, ledger)
  - AMD SEV – full VM isolation

Building Custom TEE is Expensive

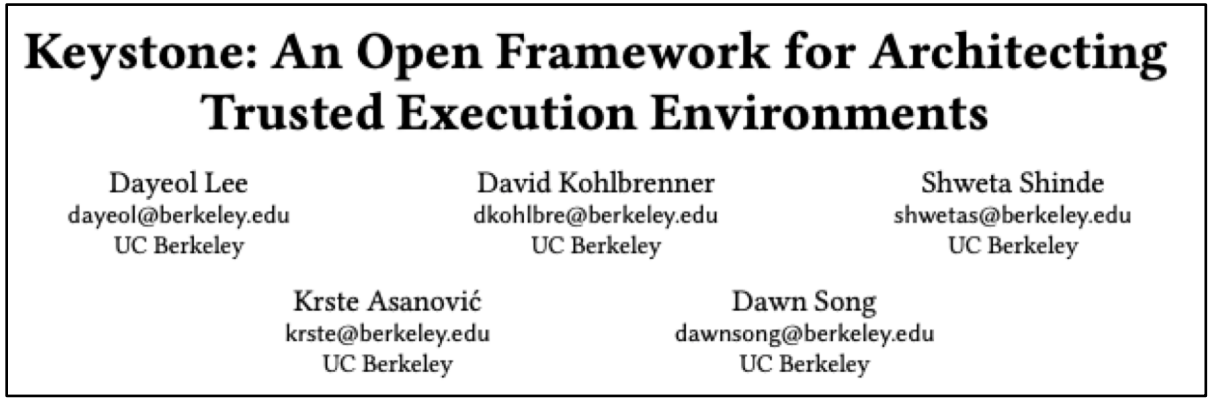
Customizable TEE

- ❑ A framework provides software-defined building blocks of TEEs
- ❑ You can “customize” the TEE using the framework



Keystone Enclave Project

- ❑ A framework for trusted execution environments
- ❑ Open source
- ❑ Started in 2018
- ❑ RISC-V
- ❑ Academically started in the UC Berkeley
- ❑ Published in EuroSys'20



Keystone: An Open Framework for Architecting Trusted Execution Environments

Dayeol Lee
dayeol@berkeley.edu
UC Berkeley

David Kohlbrenner
dkohlbre@berkeley.edu
UC Berkeley

Shweta Shinde
shwetas@berkeley.edu
UC Berkeley

Krste Asanović
krste@berkeley.edu
UC Berkeley

Dawn Song
dawnsong@berkeley.edu
UC Berkeley

Goals of the Project

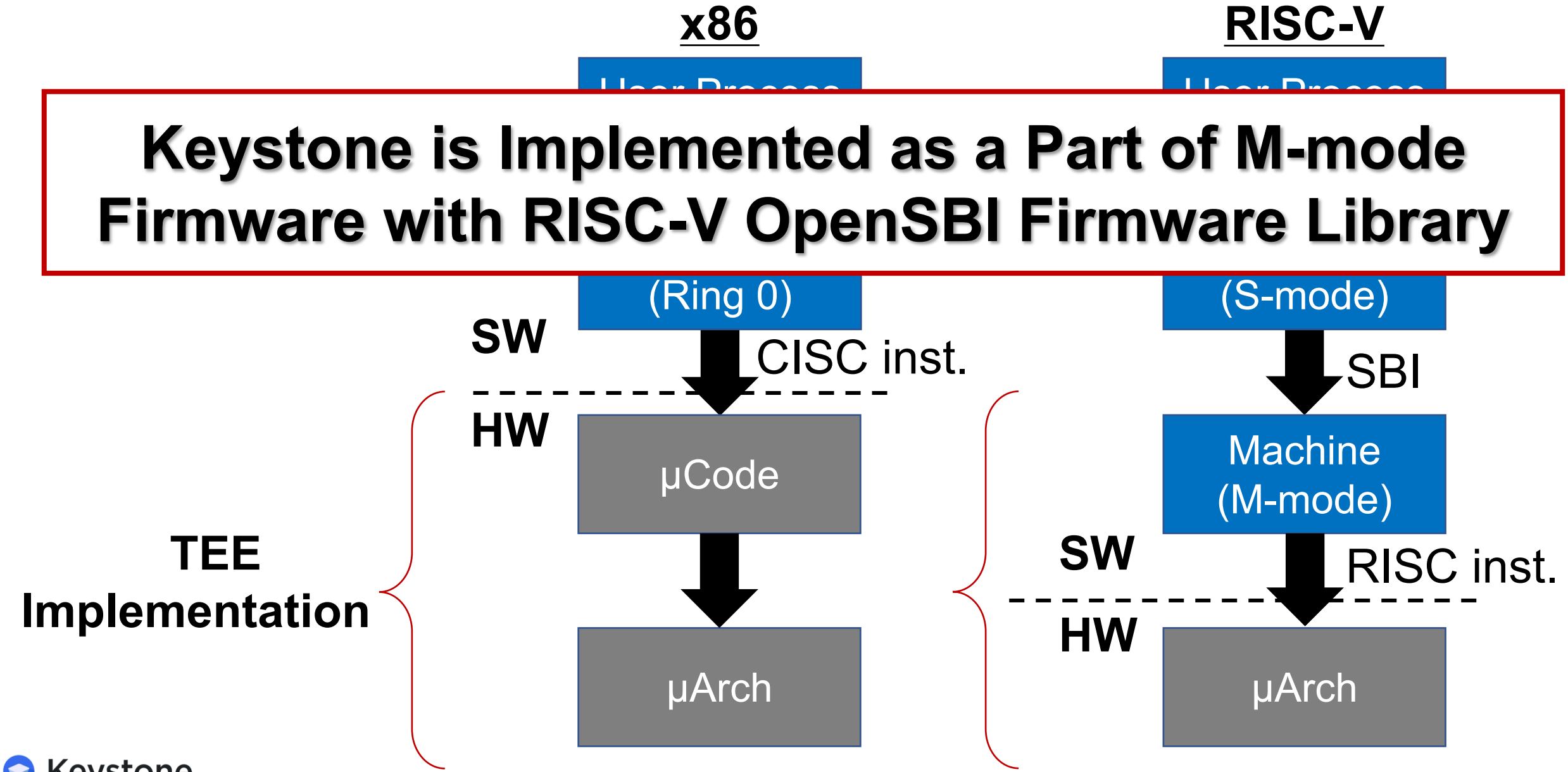
- Enable TEE on (almost) **all RISC-V processors**
  - Follow RISC-V standard ISA
  - Standard TEE specification for various RISC-V sub-ISA
- Make TEE **easy to customize** depending on needs
  - Base implementation vs. platform-specific implementation
  - Reuse the implementation across multiple platforms
- **Reduce the cost** of building TEE
  - Reduce hardware integration cost
  - Reduce verification cost
  - Integrate with existing software tools

RISC-V

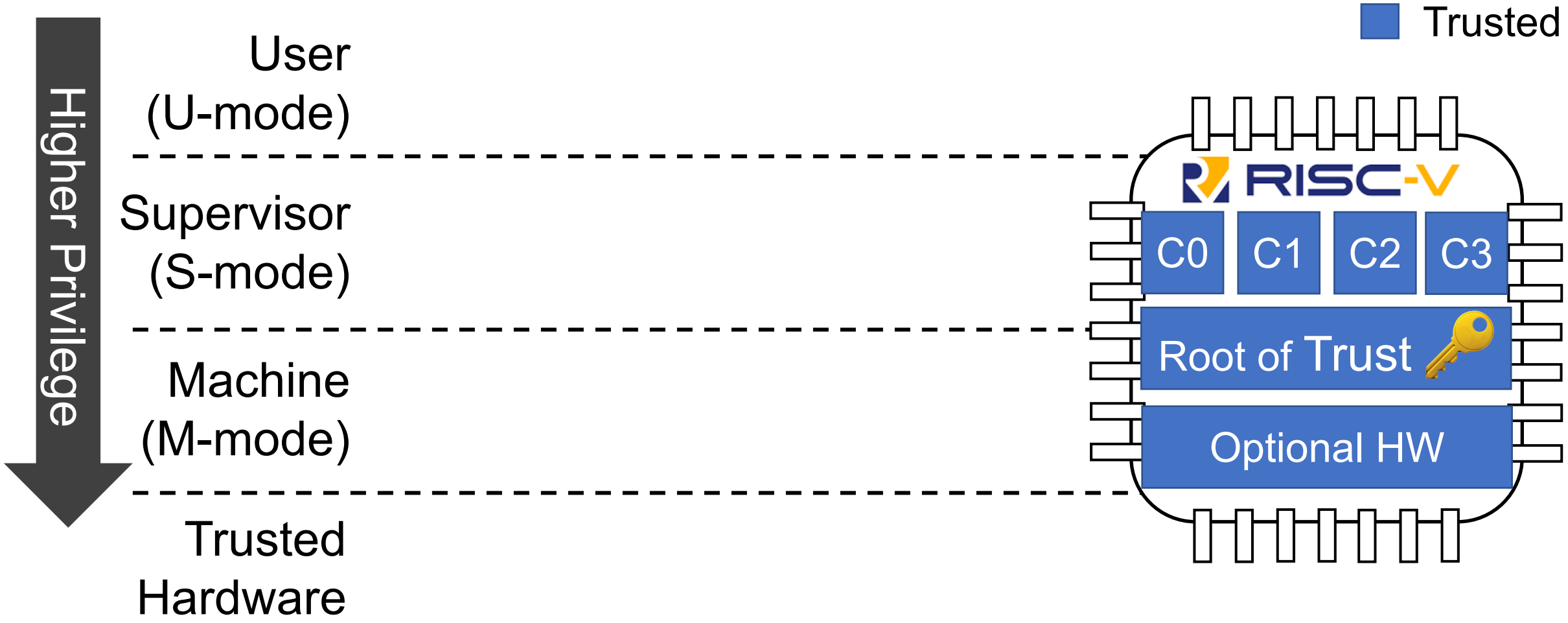
- Open Instruction Set Architecture (ISA)
- Simple, efficient, and extensible
- Becoming the industry-standard ISA for all computing devices
- Growing ecosystem
  - Both software and hardware
  - Both open-source and commercial
- More companies building RISC-V chips

TEE in x86 vs. RISC-V

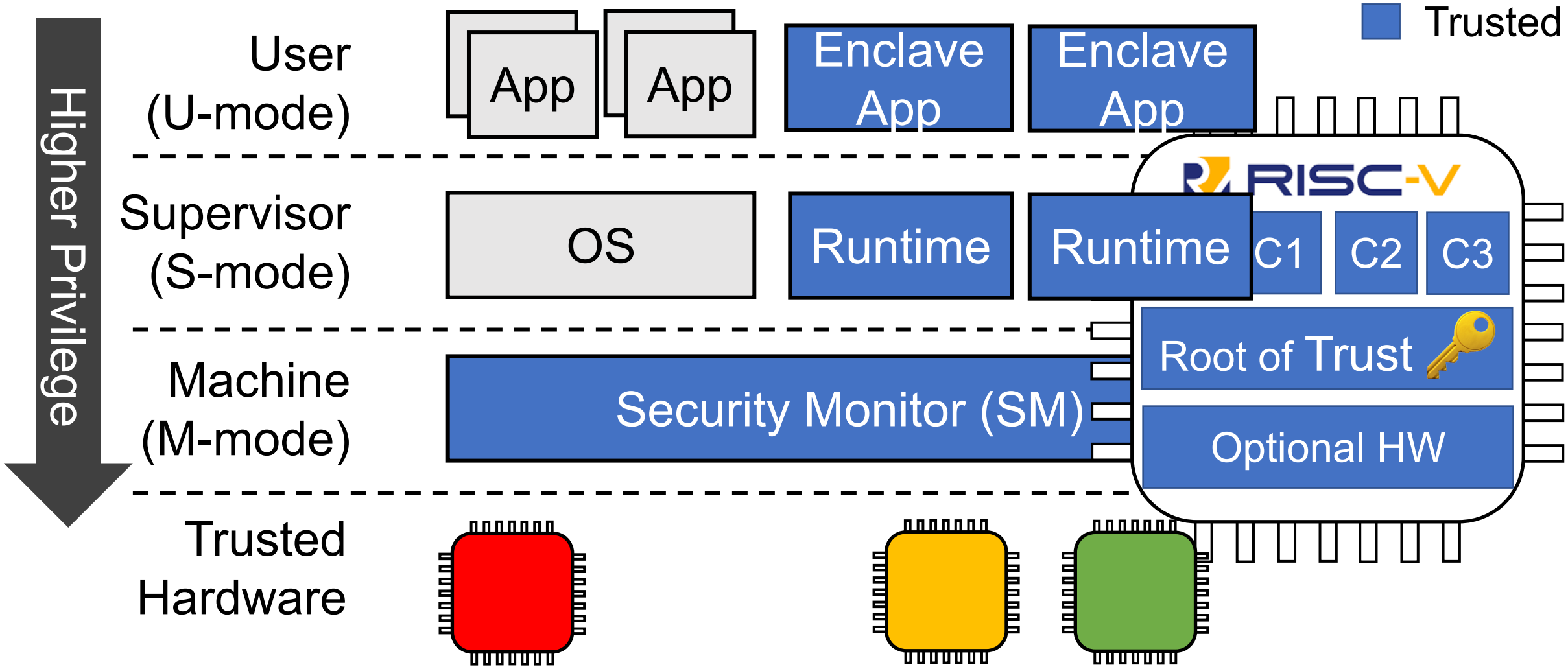
Keystone is Implemented as a Part of M-mode Firmware with RISC-V OpenSBI Firmware Library



Keystone Architecture and Trust Model

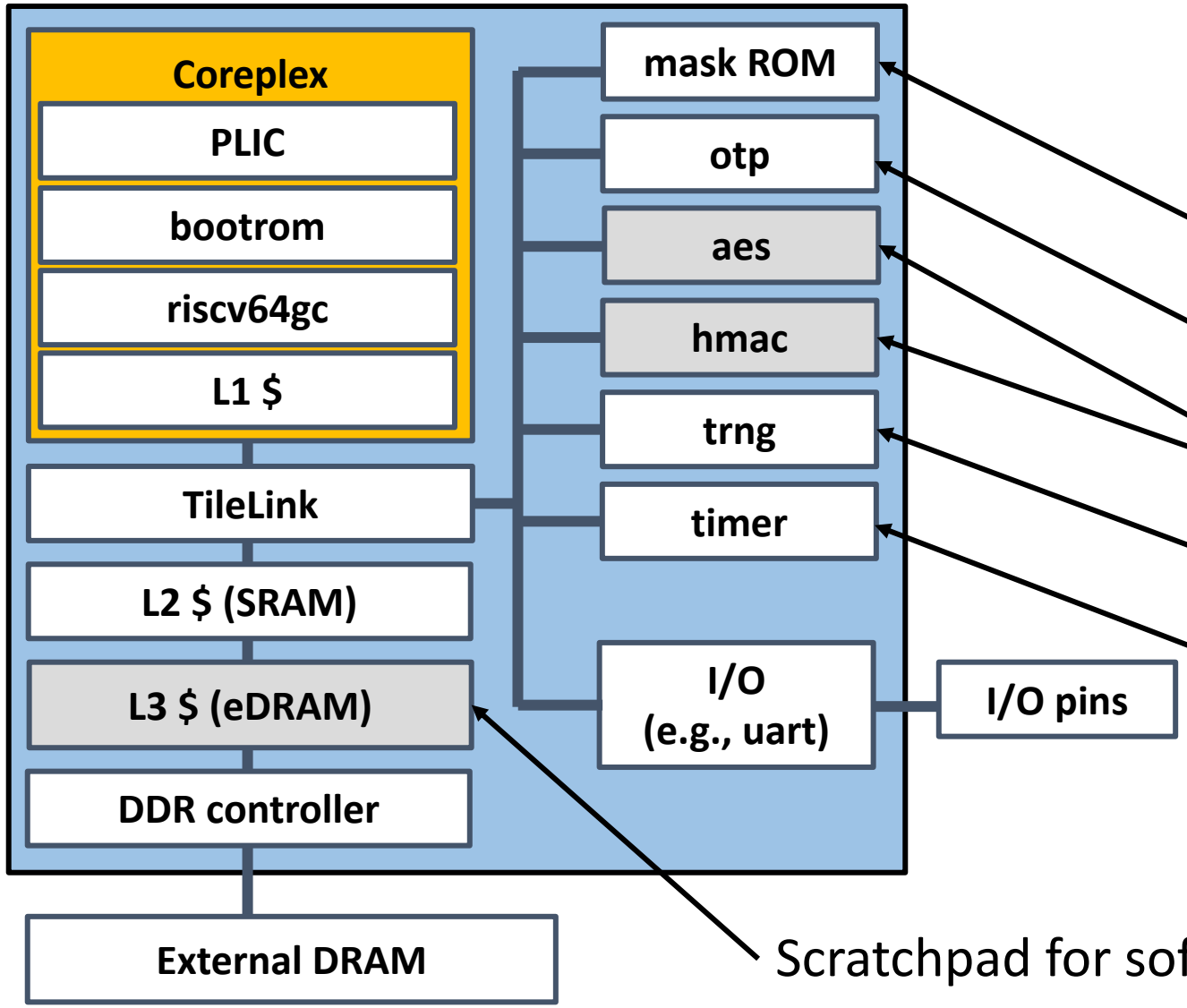


Keystone Architecture and Trust Model



Example Custom Hardware Deployment

e.g., OpenTitan Security Specification
https://docs.opentitan.org/doc/security/specs/



Zeroth-stage bootloader (ZSBL)

Device root key & signature

Crypto accelerator(s)

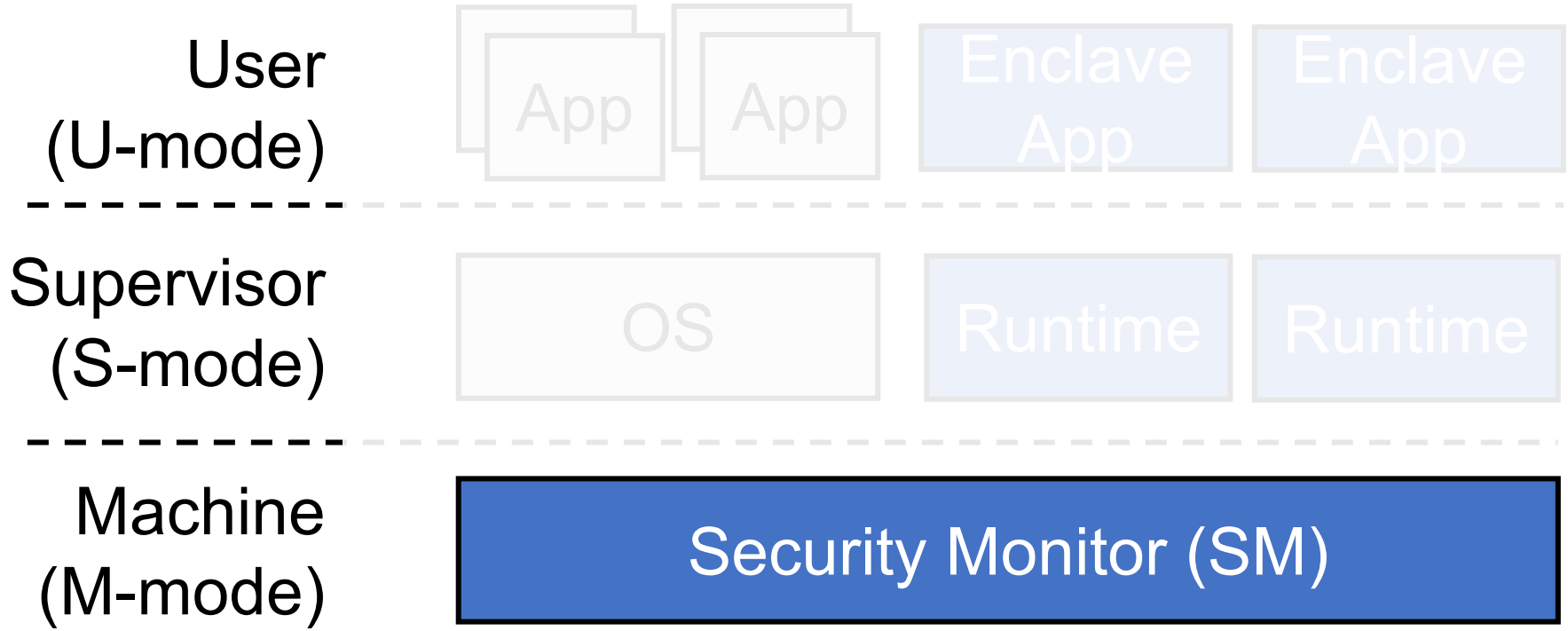
Random number generator

Trusted timer

Optional

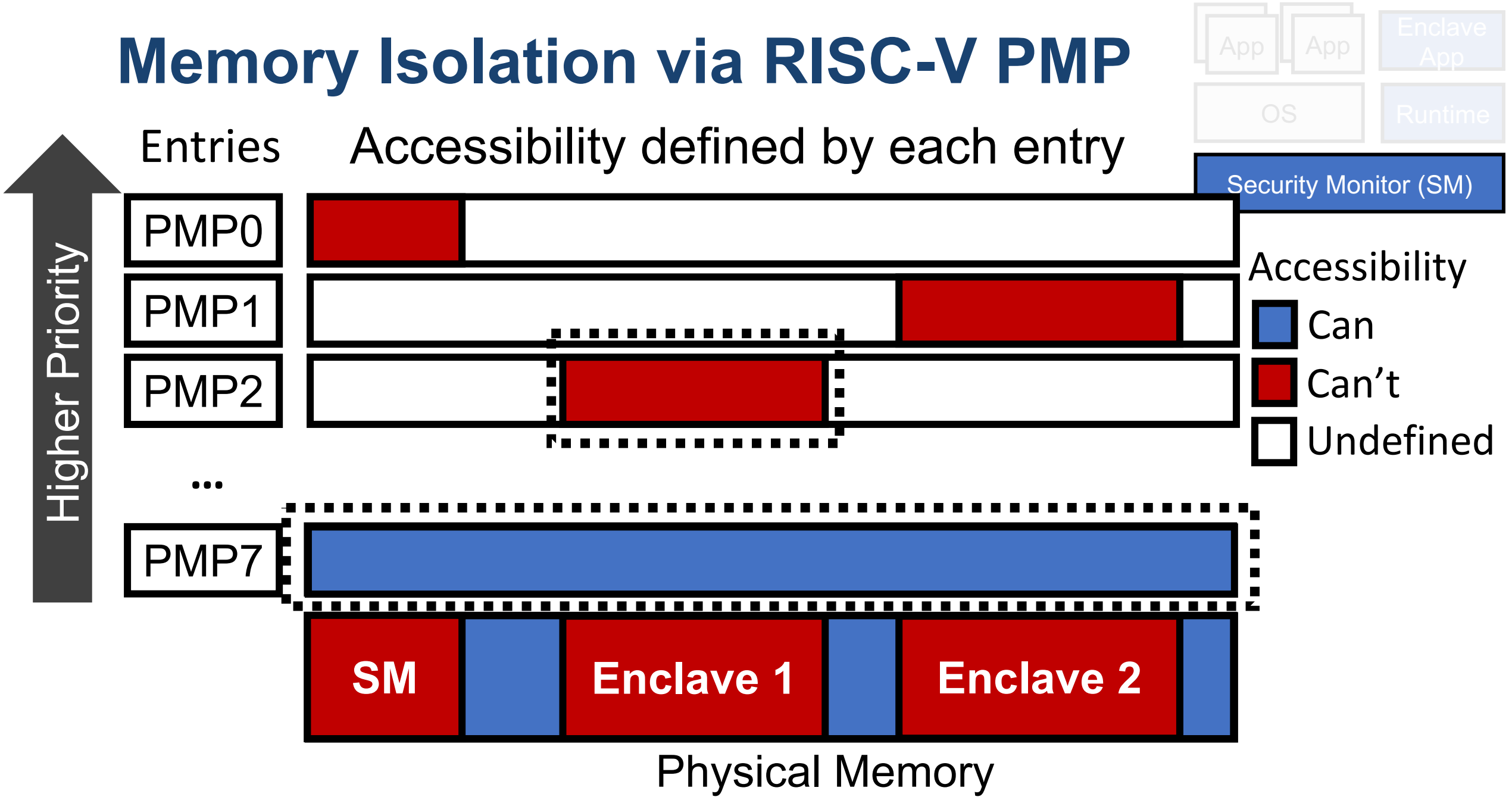
Scratchpad for software memory encryption

Keystone Architecture and Trust Model

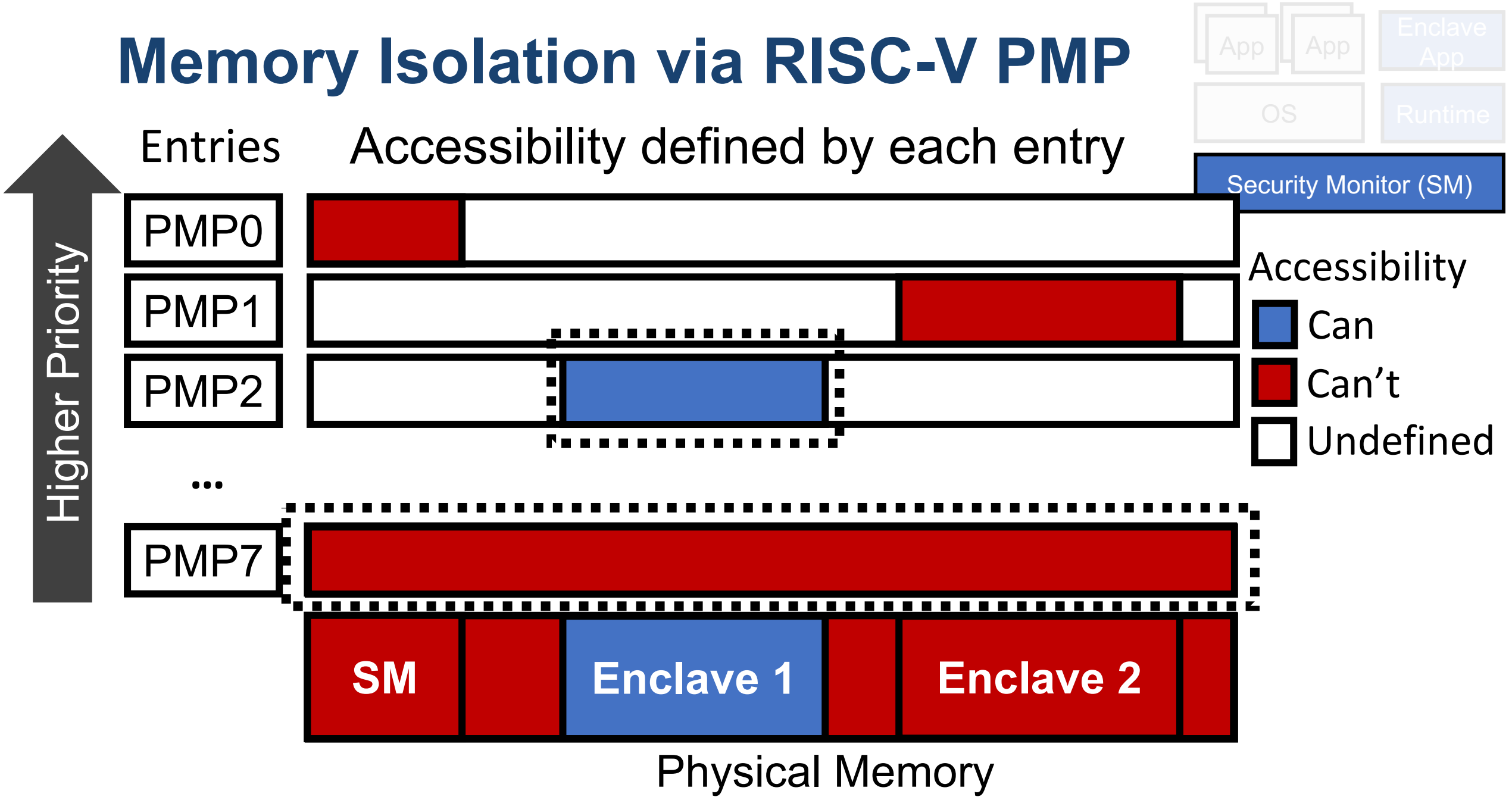


Hardware-Enforced and Software-Defined Isolation

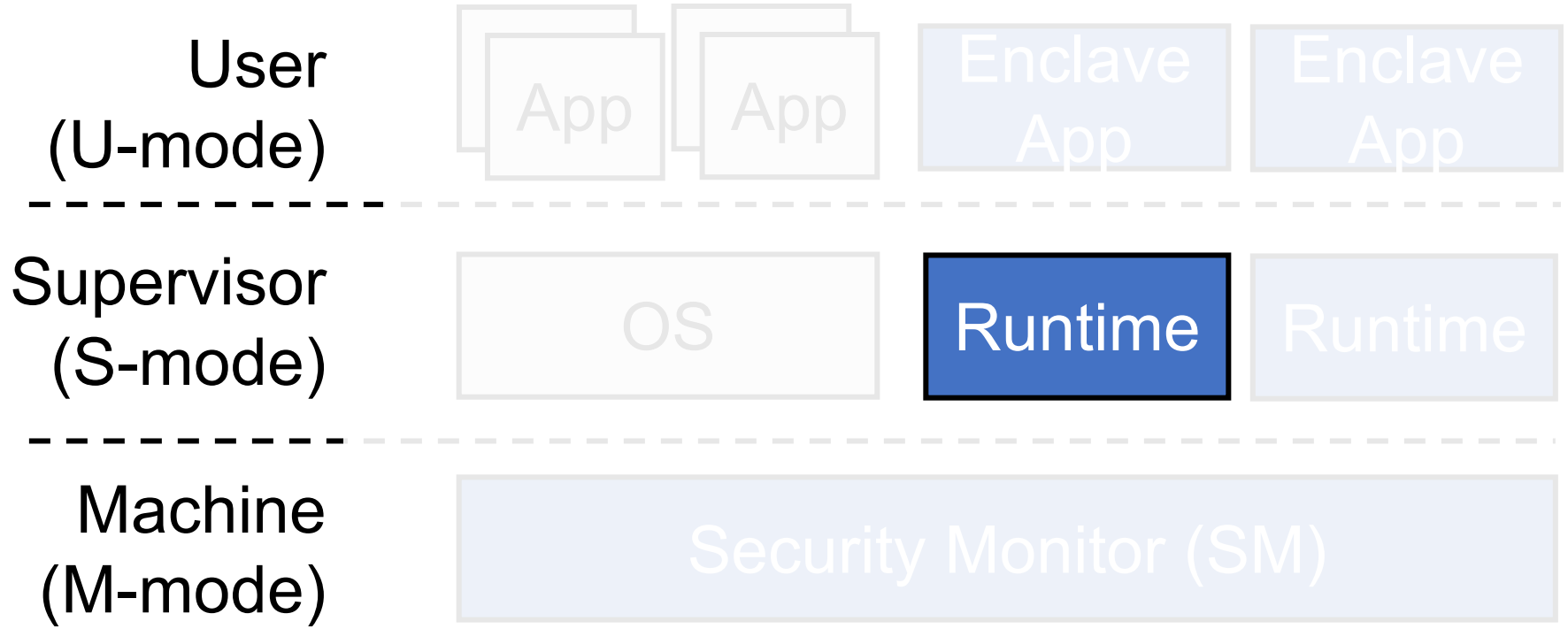
Memory Isolation via RISC-V PMP



Memory Isolation via RISC-V PMP



Keystone Architecture and Trust Model



What Does Keystone Runtime Do?

What does Keystone Runtime Do?

User (U-mode)

App

Enclave App

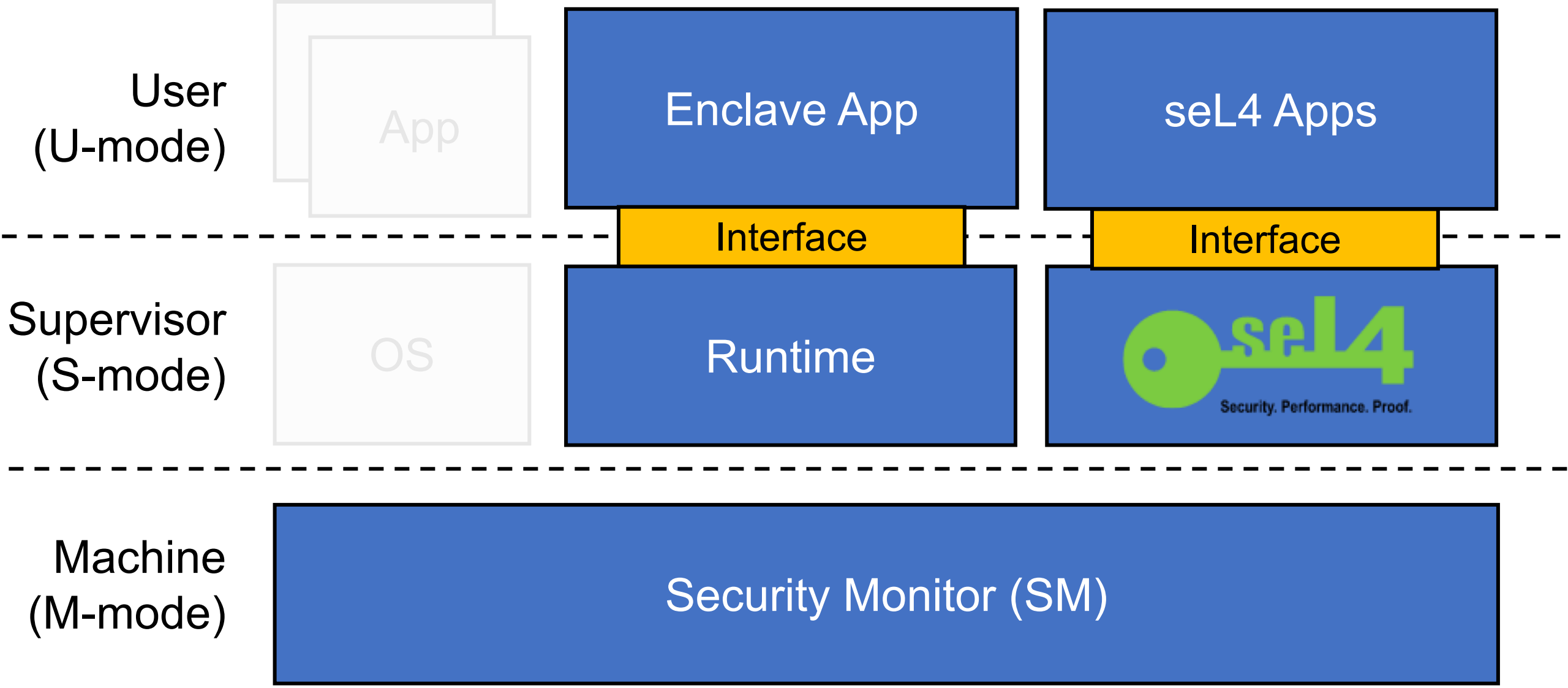
Supervisor (S-mode)

OS

Machine (M-mode)

Security Monitor (SM)

What does Keystone Runtime Do?



What does Keystone Runtime Do?

User (U-mode)

App

Enclave App

seL4 Apps

Supervisor (S-mode)

OS

freemem

paging

I/O

syscall

libc

Machine (M-mode)

Security Monitor (SM)

Use Cases

Multi-Platform, General-Purpose TEE

- Multi-platform
  - Supports both 32-bit (RV32) and 64-bit (RV64) Sub-ISA
  - Cores: RocketChip, BOOM, Ariane
  - Boards: SiFive Unleashed, Microsemi Polarfire
  - TEE in real-time OS

    Thomas et al., “ERTOS: Enclaves in Real-Time Operating Systems”, CARRV 2021
- General-purpose
  - Lightweight partitioned application
  - Full application with libc
  - Entire off-the-shelf OS (seL4) inside TEE
- Platform-specific extensions

Example Platform-Specific Extensions

- On-chip scratch pad memory
  - Software-base memory encryption

    Andrade et al., “Software-Based Off-Chip Memory Protection for RISC-V Trusted Execution Environments”, CARRV 2020
- Advanced cache controller feature (e.g., way masking)
  - Cache partitioning to defeat side channel
- AES/Ed25519 accelerator
  - Faster TEE initialization
  - Faster secure boot

    Hoang et al., “Quick Boot of Trusted Execution Environment with Hardware Accelerators”, IEEE Access, May 2020

Academic Users

ELASTICLAVE: An Efficient Memory Model for Enclaves

Zhijingcheng Yu
National University of Singapore

Shweta Shinde *
ETH Zurich

Trevor E. Carlson
National University of Singapore

Prateek Saxena
National University of Singapore

PIE: A Platform-wide TEE

Moritz Schneider*, Aritra Dhar*, Ivan Puddu, Kari Kostiainen, Srdjan Čapkun
Department of Computer Science
ETH Zurich

Quick Boot of Trusted Execution Environment With Hardware Accelerators

TRONG-THUC HOANG[1,2], (Graduate Student Member, IEEE),
CKRISTIAN DURAN[1], (Student Member, IEEE),
DUC-THINH NGUYEN-HOANG[1], (Student Member, IEEE),
DUC-HUNG LE[1], (Member, IEEE), AKIRA TSUKAMOTO[2],
KUNIYASU SUZAKI[2,3], AND CONG-KHA PHAM[1], (Member, IEEE)

When Oblivious is Not: Attacks against OPAM

Nirjhar Roy*
Indian Institute of Technology - Kanpur
nirjhar@iitk.ac.in

Nikhil Bansal
Indian Institute of Technology - Kanpur
nikhilba@iitk.ac.in

Gourav Takhar
Indian Institute of Technology - Kanpur
tgourav@cse.iitk.ac.in

Nikhil Mittal
Fortanix
nkmittal4994@gmail.com

Pramod Subramanyan†
Indian Institute of Technology - Kanpur
spramod@cse.iitk.ac.in

How To Contribute?

Key Repositories

- https://github.com/keystone-enclave
- **keystone** – top-level repo containing everything
- **sm** – Keystone security monitor firmware generator
- **keystone-runtime** – default runtime kernel (Eyrie)
- **keystone-seL4** – modified seL4 microkernel as a runtime
- **keystone-sdk** – useful libraries and example enclaves
- **keystone-demo** – demo word-counting application with remote attestation
- **linux-keystone-driver** – kernel driver for SBI calls

Security Monitor

- https://github.com/keystone-enclave/sm

.circleci	CI files
opensbi @ 0d49c3b	Bumped OpenSBI library
plat	Platform firmware (mpfs, fu540)
spec	Specification
src	All sources
tests	CMocka unit tests
tools	Enclave hash generator

Security Monitor Specification

- https://github.com/keystone-enclave/sm/blob/master/spec/v1.0.md

Keystone Security Monitor Specification

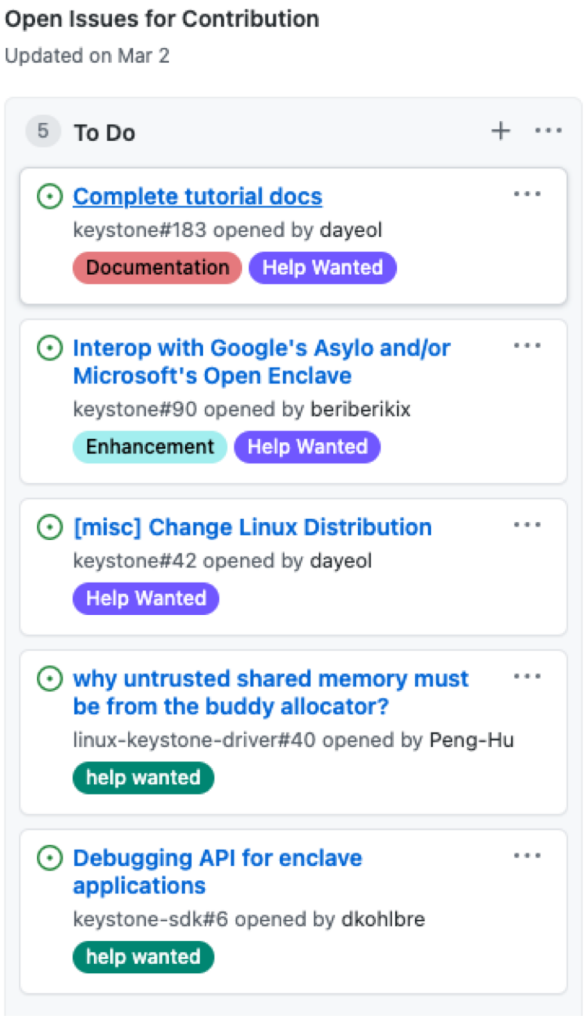
Current Spec Version
v1.0-rev1

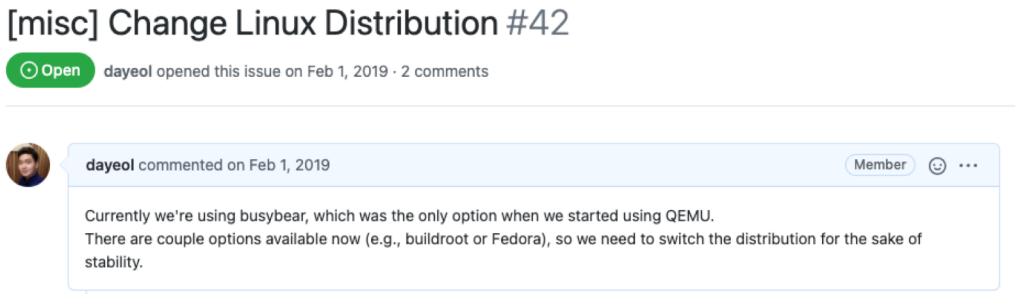
1. Introduction

This document describes the specification of Keystone security monitor (SM). Keystone SM is implemented as an experimental extension of OpenSBI. Keystone SM inherits RISC-V Supervisor Binary Interface (SBI) Specification. Given the specification, the EID of Keystone is `0x08424b45` which is `0x08` (experimental extension) + `0x424b45` (BKE, which stands for Berkeley Keystone Enclave).

Open Issues for Contribution

- https://github.com/orgs/keystone-enclave/projects/8





List of Open Projects

- https://github.com/keystone-enclave/keystone/blob/dev/CONTRIBUTING.md

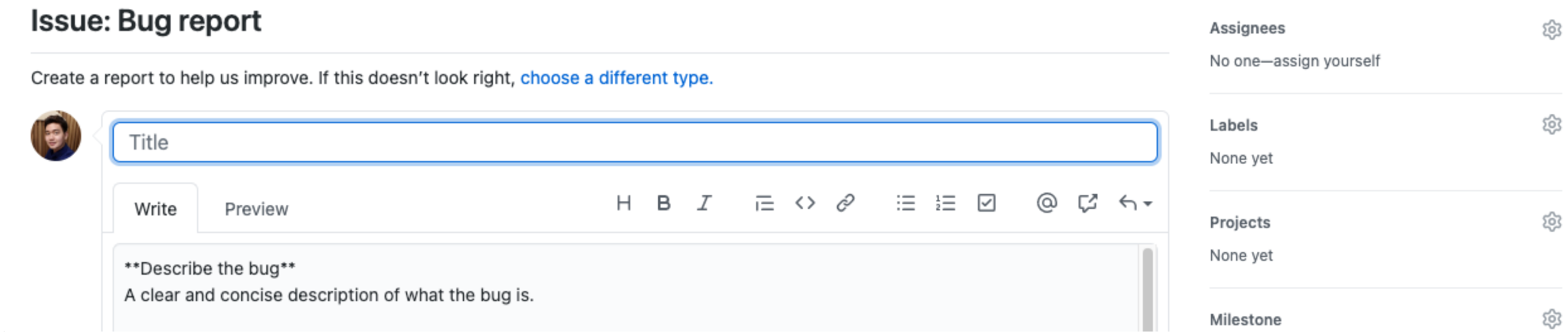
List of Projects

Hardware (Requirements are marked *)

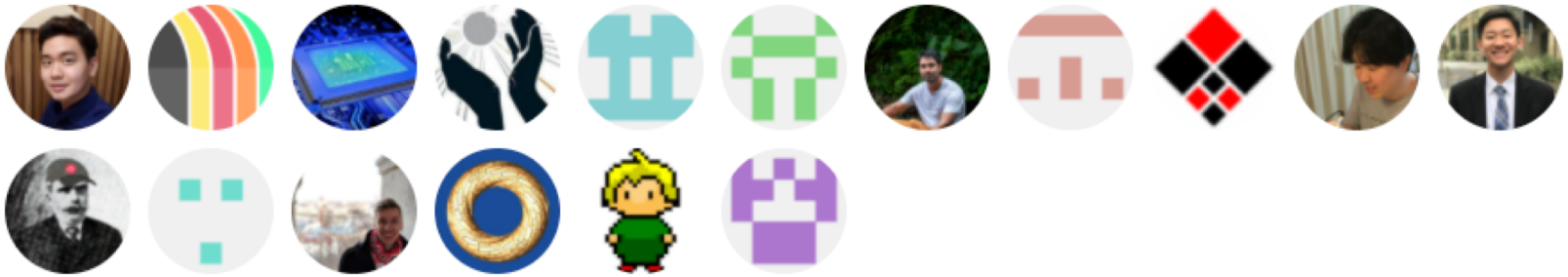
Name	Type	Assigned	Description
*Silicon root of trust	development		Currently, Keystone only implements software-based root of trust simulated via early-stage bootloader (e.g., ZSBL). This lacks hardware-based protection of the keys and the certificate. OpenTitan is a potential open-source project that can be integrated with Keystone.
*I/O protection	development		SoCs needs to also enforce the memory isolation for peripheral devices. This can be done by RISC-V IOPMP standard, which is still WIP. Some companies have already came up with non-standard IOPMP on their chip.
Interrupt Controller	development		Keystone doesn't have ability to allow enclaves to receive their own interrupts. This can be implemented on PLIC or CLIC interrupt controller.
Crypto Accelerator	research, development	Gui Andrade	Cryptographic accelerators may speed up secure booting, measurement, and attestation. Also, this could potentially make software-based memory encryption practical (ongoing research by Gui Andrade)

Bug Reporting & Feature Request

- https://github.com/keystone-enclave/keystone/issues



Contributors



Be Connected!

- Forum: https://groups.google.com/forum/#!forum/keystone-enclave-forum
- Documentation: http://docs.keystone-enclave.org/en/latest/
- GitHub: https://github.com/keystone-enclave
- Website: https://keystone-enclave.org/
- Contact: dayeol <at> berkeley <dot> edu

Summary

- Keystone is an open framework for TEE on RISC-V
- Goals:
  - Enable TEE on every processor
  - Make it easy to customize
  - Reduce the cost
- Open for discussion and contribution!

Thank You!

Dayeol Lee (dayeol <at> berkeley <dot> edu)