蓬莱

可信执行环境

Penglai-Enclave: Secure and Efficient RISC-V Enclave

Dong Du, Xu Lu, Erhu Feng, Bicheng Yang, Yubin Xia, Haibo Chen

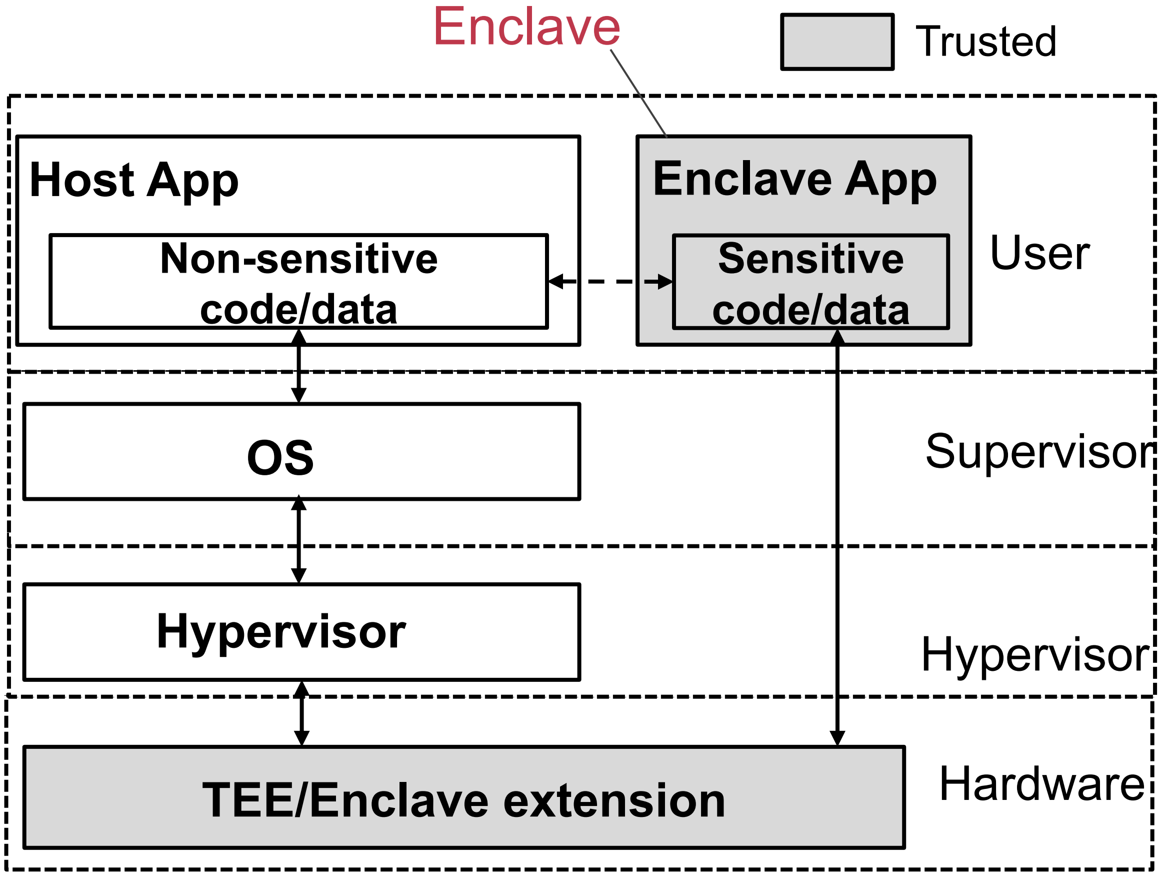
IPADS, Shanghai Jiao Tong University

IPADS
INSTITUTE OF PARALLEL
AND DISTRIBUTED SYSTEMS

TrustKernel
上海瓶钵信息科技有限公司

What’s Enclaves?

Hardware-assisted Trusted Execution Environment



Why Enclave for RISC-V?

Lessons learned:

Security is a necessary processor need

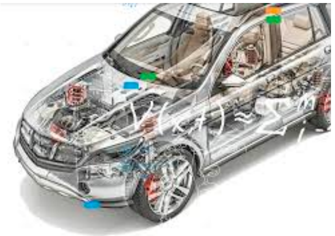
intel SGX

AMD SEV

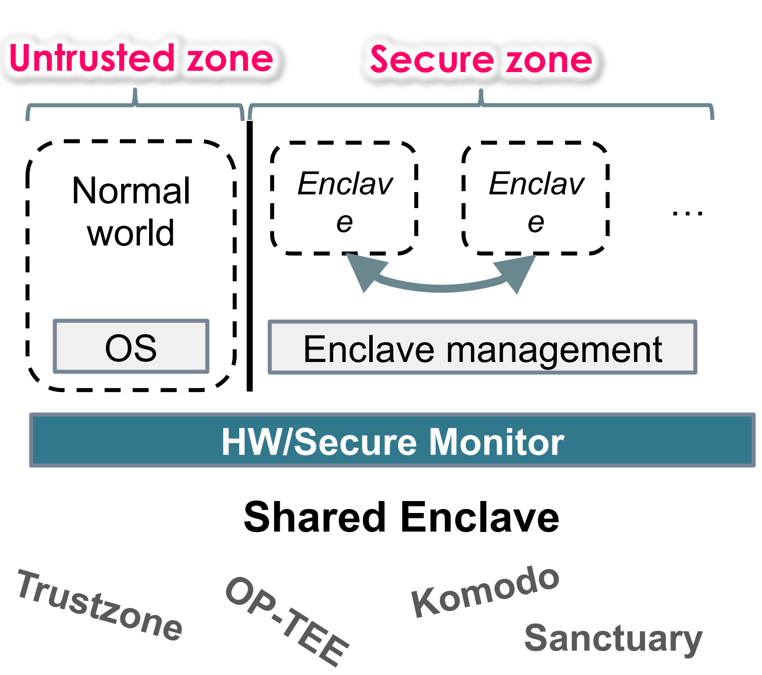
arm TRUSTZONE

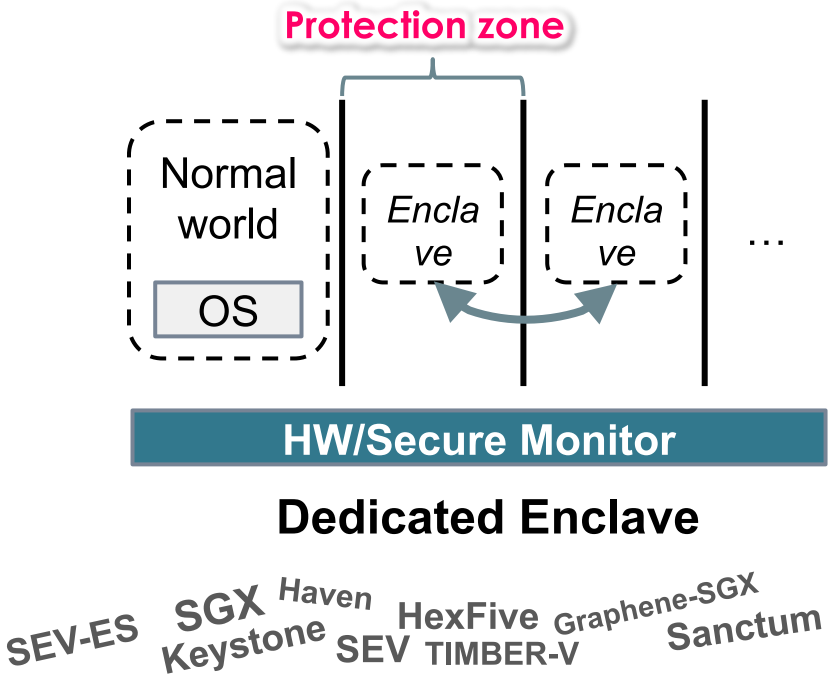
User needs:

Users will run sensitive code/data in RISC-V



Why Another Enclave?





Shared Enclave Architecture: A single HW-assisted protection zone for multiple enclaves

- Pros: low communication latency (intra-zone)
- Cons: higher TCB

Dedicated Enclave Architecture: A single HW-assisted protection zone for a single enclave

- Pros: small TCB → higher security-assurance
- Cons: long communication latency (inter-zone) & non-scalable

Why Another Enclave?

Systems			Performance				Security		
Type	Name	Arch	Communication	Fast boot	Instance number	Mem size & gran	Side channel (PT/Cache)	TCB	Mem enc./ integrity
Ded En	SGX	Intel	Slow	✗	Unlimited	256MB	✗	Small	√
	Haven	Intel	Slow	✗	Unlimited	256MB	✗	Large	√
	Graphene	Intel	Slow	✗	Unlimited	256MB	✗	Large	√
Shared Enclave	TrustZone	ARM	Fast	✗	Unlimited	All	Partial	Large	✗
	OP-TEE	ARM	Fast	✗	Unlimited	All	Partial	Large	✗
	Komodo	ARM	Fast	√	Unlimited	All	Partial	Medi.	✗
	Sanctuary	ARM	Fast	√	Unlimited	All	Partial	Large	✗
heavy.LIGHT	**Penglai**	RISC-V	**Fast**	√	**Unlimited**	All	√	Medi.	✗

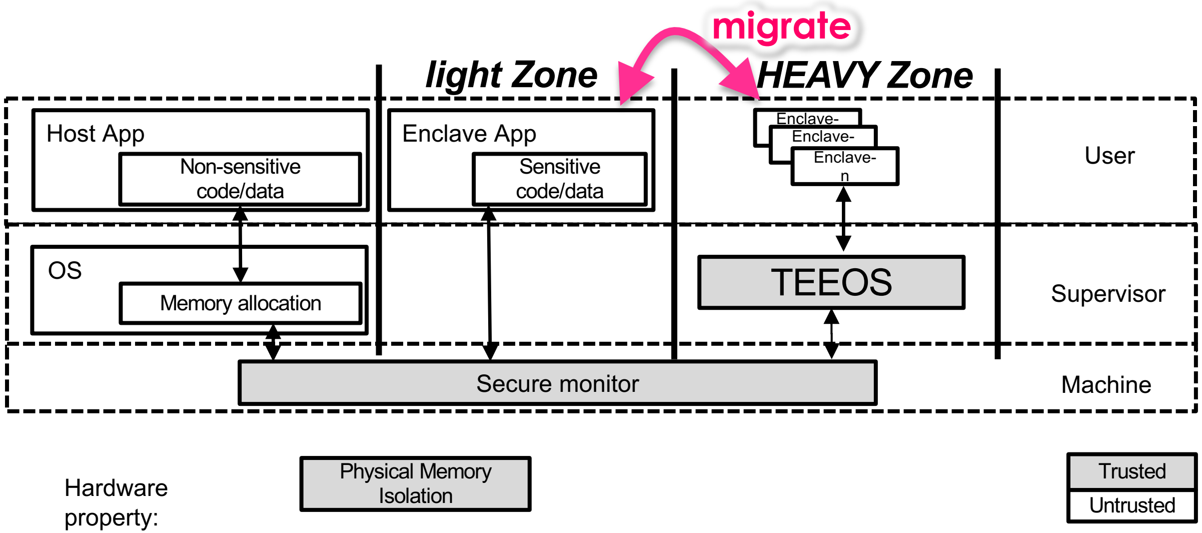
Existing enclave systems can not achieve **security** and **performance** simultaneously.

PENGLAI-ENCLAVE (蓬莱)

Overview

- SW-HW Co-design Trusted Execution Environment (TEE)
  - Based on RISC-V ISA
  - Can run on **any RISC-V core** that supports Privileged ISA v1.10
  - IoT and Cloud
- Trusted Code Base
  - RISC-V core (PMP/sPMP) + Verifiable security monitor (M-mode privilege) + TEEOS
- Secure Assurance
  - Strong isolation between enclave and other application or OS
  - Protect against a malicious or compromised OS
  - Secure boot and remote attestation for chain of trust
  - High performance and scalability

Penglai with heavy.LIGHT Architecture



heavy.LIGHT architecture

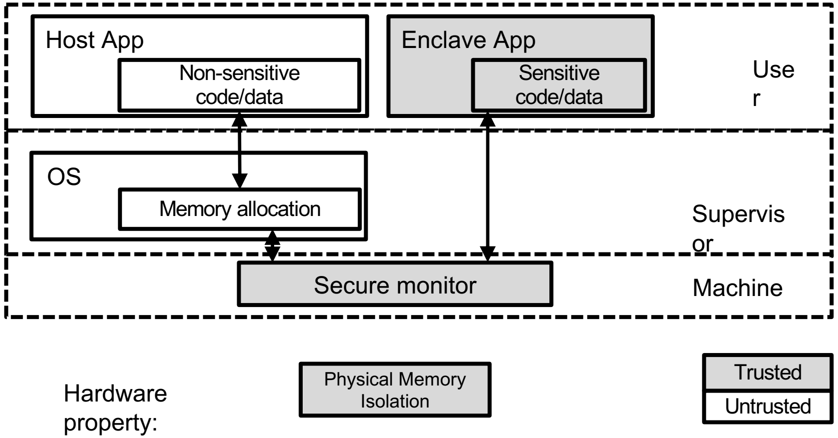
- **LIGHT Zone:** A dedicated HW-isolated box for a single enclave
- **heavy Zone:** Multiple-Enclaves isolated through TEEOS

- **TEEOS:** Leverage s-mode for enclave isolation （sPMP and PMP)
  - Fast cross-enclave communication (IPC)
  - Flexible resource management
  - Fast startup
  - Scalable instances

heavy.LIGHT Architecture

- **Security monitor:**
  - A small software running on M-mode
  - Enclave measurement and attestation
  - Manage enclave and provide isolation

    via physical memory isolation property
- **Physical Memory Isolation Property**
  - Restrict physical memory access of S- or U-mode software
  - Dynamically configurable by security monitor

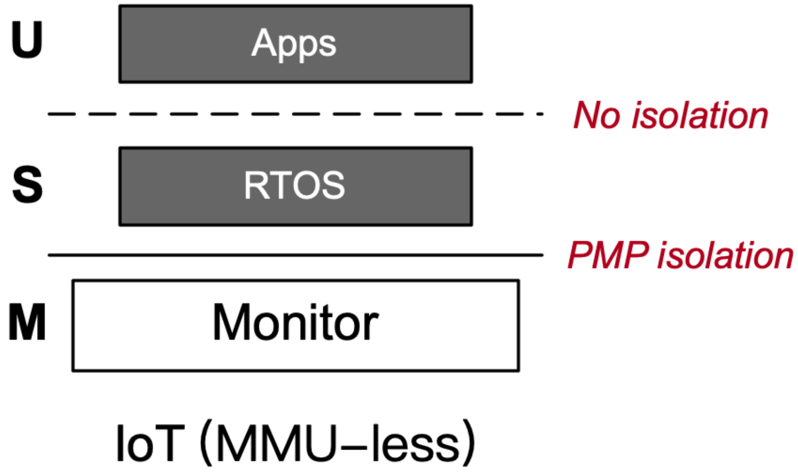


Hardware Requirement

- RV32 or RV64
- All of the three modes (M/S/U)
- Support RISC-V priv. ISA v1.10
  - need sPMP or PT support for performance
- Larger, tamper-proof boot ROM (~1MB)
  - Trusted bootloader should be added to initialize the system
- Physical memory isolation support
  - New hardware property for memory isolation
- IOPMP extension
  - Defend malicious I/O access

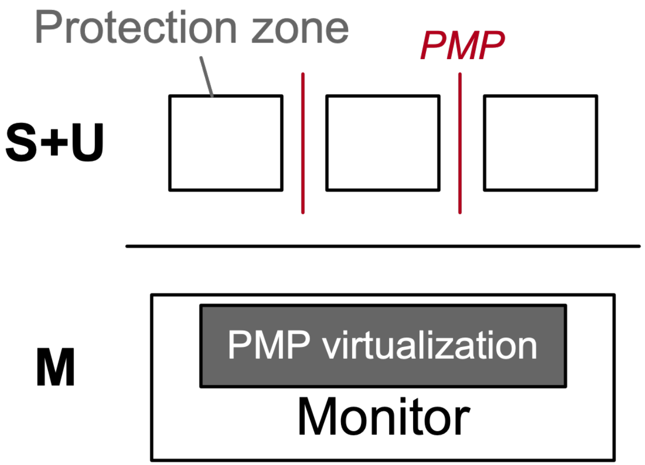
SPMP (S-mode PMP)

- For IoT devices (MMU-less)
  - it is desirable to enable S-mode OS to limit the physical addresses accessible by U-mode software

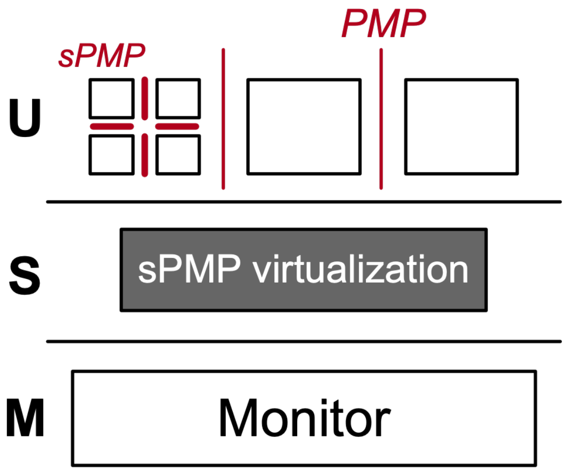


SPMP (S-mode PMP)

- S-mode virtualization for scalable enclaves



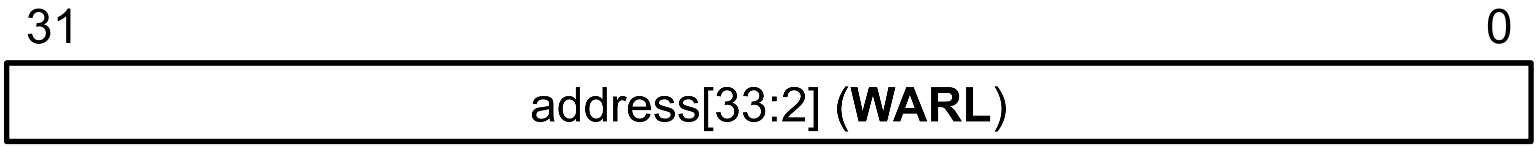
(a) PMP-based isolation



(b) sPMP

Supervisor-mode Physical Memory Protection Keys

- sPMP entries
  - 8-bit configuration register
  - XLEN-bit address register



31

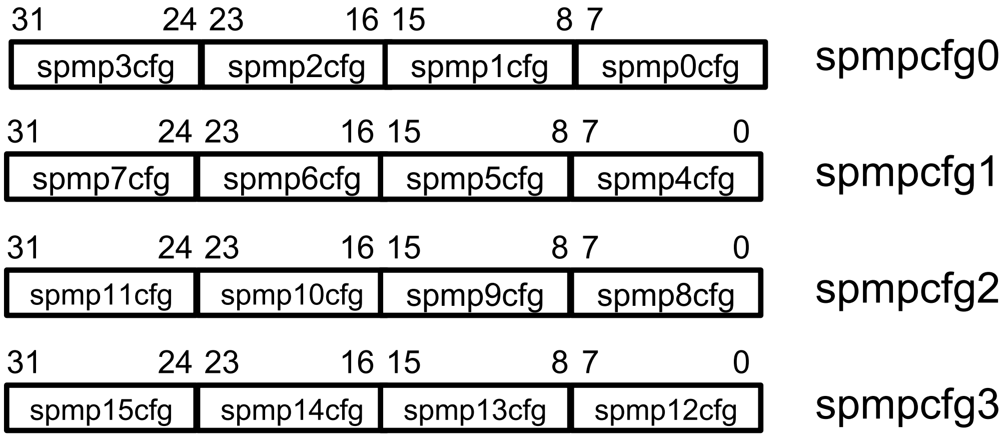
0

address[33:2] (**WARL**)

address register (RV32)

Supervisor-mode Physical Memory Protection Keys

- sPMP entries
  - 8-bit configuration register
  - XLEN-bit address register

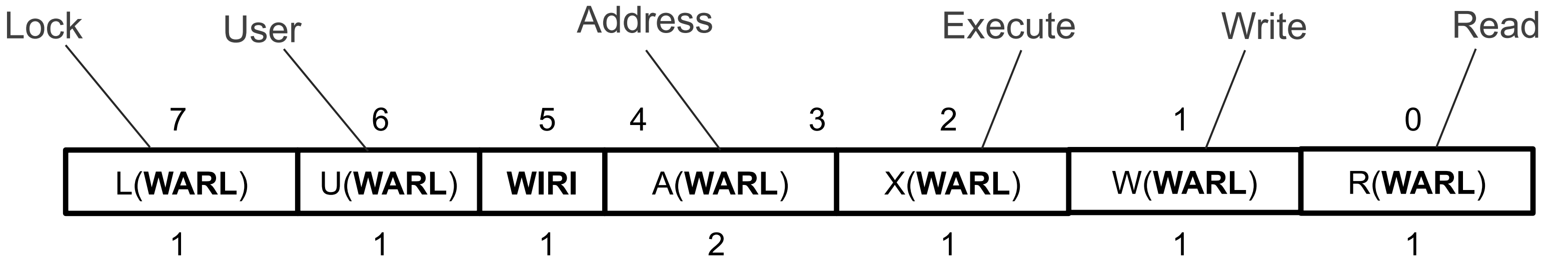


31 24	23 16	15 8	7	
spmp3cfg	spmp2cfg	spmp1cfg	spmp0cfg	spmpcfg0
31 24	23 16	15 8	7 0	
spmp7cfg	spmp6cfg	spmp5cfg	spmp4cfg	spmpcfg1
31 24	23 16	15 8	7 0	
spmp11cfg	spmp10cfg	spmp9cfg	spmp8cfg	spmpcfg2
31 24	23 16	15 8	7 0	
spmp15cfg	spmp14cfg	spmp13cfg	spmp12cfg	spmpcfg3

configuration register (RV32)

Supervisor-mode Physical Memory Protection Keys

- sPMP entries
  - 8-bit configuration register
  - XLEN-bit address register



Lock	User		Address	Execute	Write	Read
7	6	5	4 3	2	1	0
L(**WARL**)	U(**WARL**)	**WIRI**	A(**WARL**)	X(**WARL**)	W(**WARL**)	R(**WARL**)
1	1	1	2	1	1	1

configuration register format

Supervisor-mode Physical Memory Protection Keys

- Address matching
  - Same as PMP
- Locking and privilege mode
  - The *Lock* bit indicates: the sPMP is locked to S-mode
- Priority and Matching Logic
  - The lowest-numbered sPMP entry
  - Failed accesses generate a page fault exception

Refer our proposal in RISC-V/TEE group for details !

SMAP and SMEP

- SMAP (Supervisor Memory Access Prevention)
  - leverage the SUM bit in the status register
  - SUM=0:
    - S-mode memory accesses to memory for U-mode (U=1) will fault
  - SUM=1:
    - these accesses are permitted
- SMEP (Supervisor Memory Execution Prevention)
  - Do not allow the S-mode to execute codes in physical memory that are for U-mode (U=1)
- Violations will trigger page faults

Chain of Trust: (1) Secure Boot

- The manufacturer provisions:
  - device key pair {SK_{DEV}, PK_{DEV}}
  - endorses the certificate
- During CPU resets: load bootloader from boot ROM and execute it
- The bootloader measures and signs the security monitor
- The secure monitor measures and signs the TEEOS
- The user can remotely attest the security monitor and TEEOS by

  - trusting the manufacturer’s certificate

  - comparing the measurement

  - and verifying the signature with PK_{DEV}

Chain of Trust: (2) Remote Attestation for *light* Zone

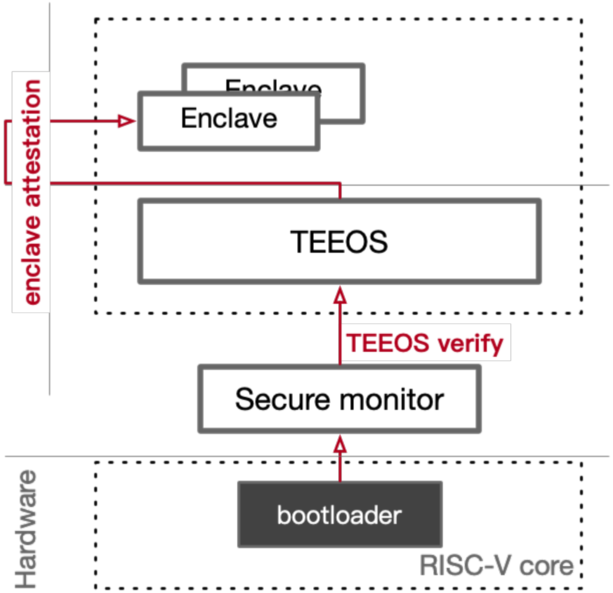
- (1) The bootloader provisions the attestation key pair $\{SK_{SM}, PK_{SM}\}$
- (2) The user uploads an executable to the system
- (3) The user asks SM to create an enclave and initialize it with the executable
- (4) The user asks SM to measure the initial state of the enclave
- (5) The SM measures the enclave, and signs it with the attestation private key
- The user can remotely attest the enclave by

  - comparing the measurement

  - and verifying it with PK_{SM}

Chain of Trust: (3) Remote Attestation for *HEAVY* Zone

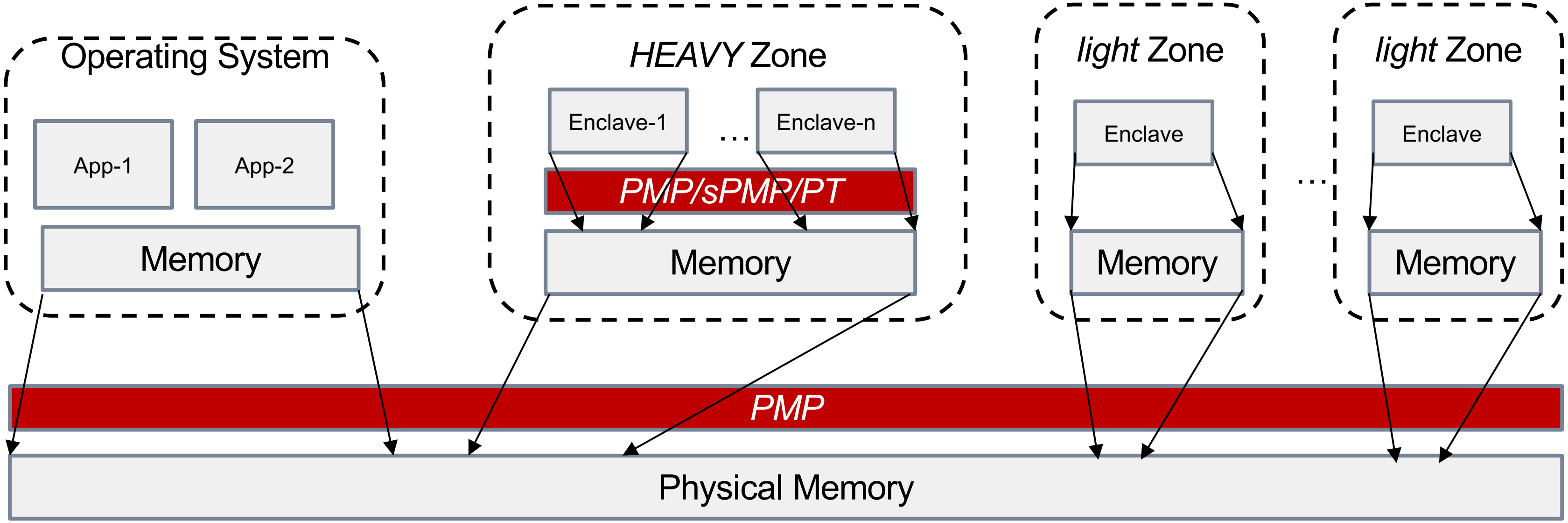
- Same (1)-(4)
- (5) The SM measures the enclave through **(verified) TEEOS**, and signs it with the attestation private key
- The user can remotely attest the enclave by
  - comparing the measurement
  - and verifying it with PK_{SM}



Layered Memory Isolation

- HEAVY.light provides a **strong** and **flexible** physical memory isolation
    - RISC-V Physical Memory Protection (PMP)
    - PT / S-mode PMP* for TEEOS memory isolation

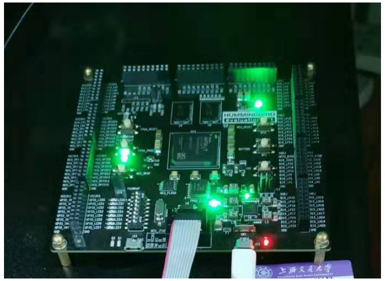
* : need HW extensions in HEAVY.light enclave design



DEMO

MCU case

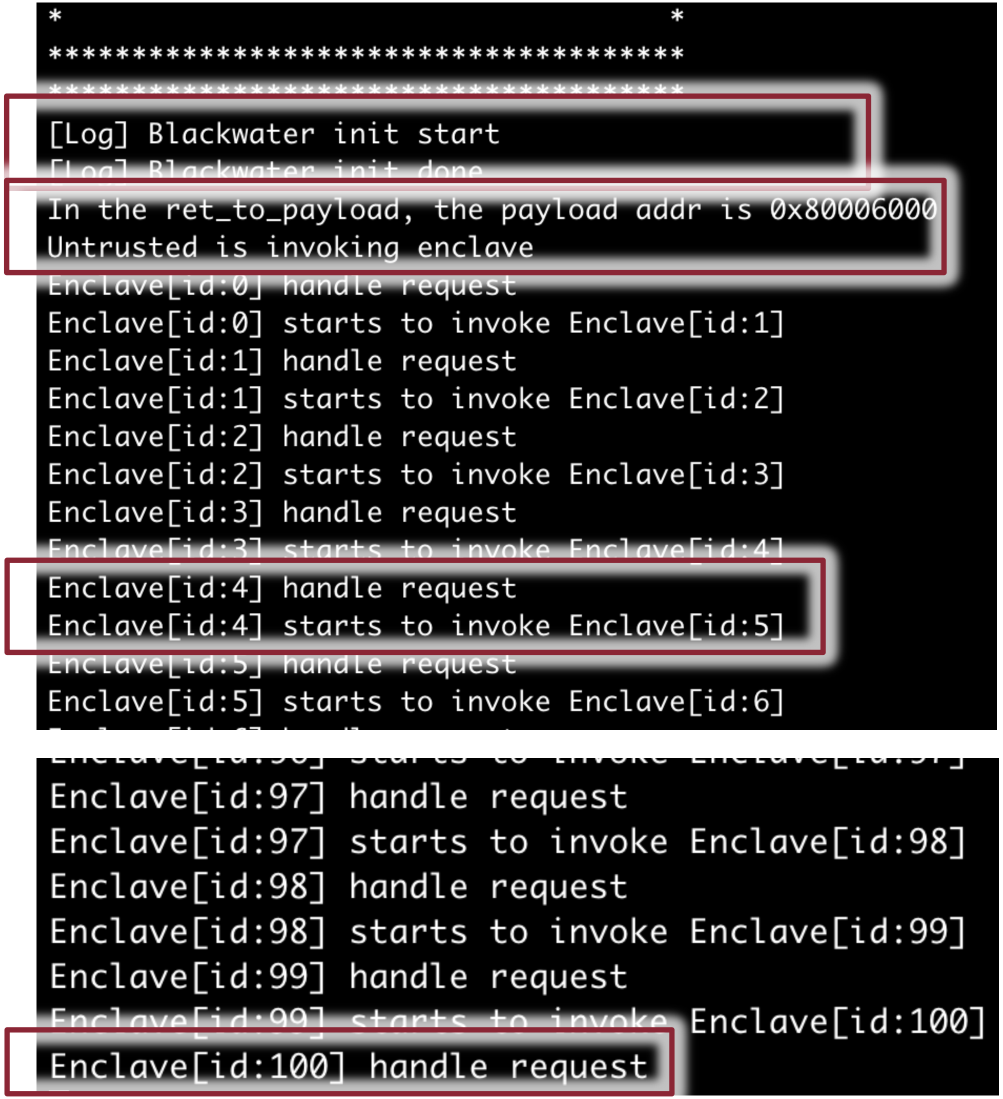
- **Platform**
  - N200 from NUCELI (芯来)
  - RV32 with sPMP support
  - No paging



芯来科技 NUCLEI

N200

- **Enclaves chain**
  - Up to 100 enclaves



Monitor starts

Untrusted App starts

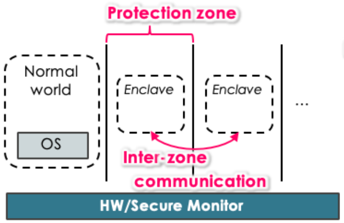
Enclave executes

100 Enclaves

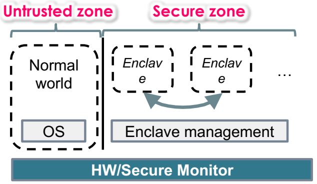
```
*                                    *
**************************************
**************************************
[Log] Blackwater init start
[Log] Blackwater init done
In the ret_to_payload, the payload addr is 0x80006000
Untrusted is invoking enclave
Enclave[id:0] handle request
Enclave[id:0] starts to invoke Enclave[id:1]
Enclave[id:1] handle request
Enclave[id:1] starts to invoke Enclave[id:2]
Enclave[id:2] handle request
Enclave[id:2] starts to invoke Enclave[id:3]
Enclave[id:3] handle request
Enclave[id:3] starts to invoke Enclave[id:4]
Enclave[id:4] handle request
Enclave[id:4] starts to invoke Enclave[id:5]
Enclave[id:5] handle request
Enclave[id:5] starts to invoke Enclave[id:6]
```

```
Enclave[id:96] starts to invoke Enclave[id:97]
Enclave[id:97] handle request
Enclave[id:97] starts to invoke Enclave[id:98]
Enclave[id:98] handle request
Enclave[id:98] starts to invoke Enclave[id:99]
Enclave[id:99] handle request
Enclave[id:99] starts to invoke Enclave[id:100]
Enclave[id:100] handle request
```

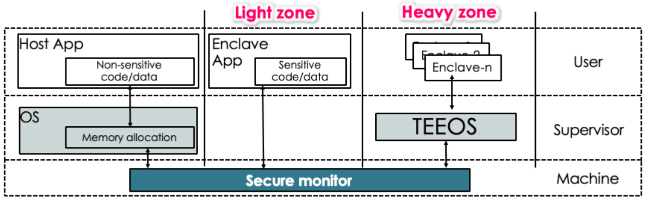
Conclusion



Dedicated Enclave



Shared Enclave



- **Background**
  - Existing enclave systems is either *dedicated enclave* or *shared enclave*
  - Cannot achieve both security & performance simultaneously
- **Penglai-Enclave** is based on heavy.LIGHT architecture
  - Using light-zone and heavy-zone to achieve both performance and security
  - TEEOS for scalability
- **Cases**
  - siFive U500 (Xilinx VC707)
  - Nuclei N200

IPADS INSTITUTE OF PARALLEL AND DISTRIBUTED SYSTEMS

TrustKernel 上海瓶钵信息科技有限公司

Thanks!