IPADS
INSTITUTE OF PARALLEL AND DISTRIBUTED SYSTEMS

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

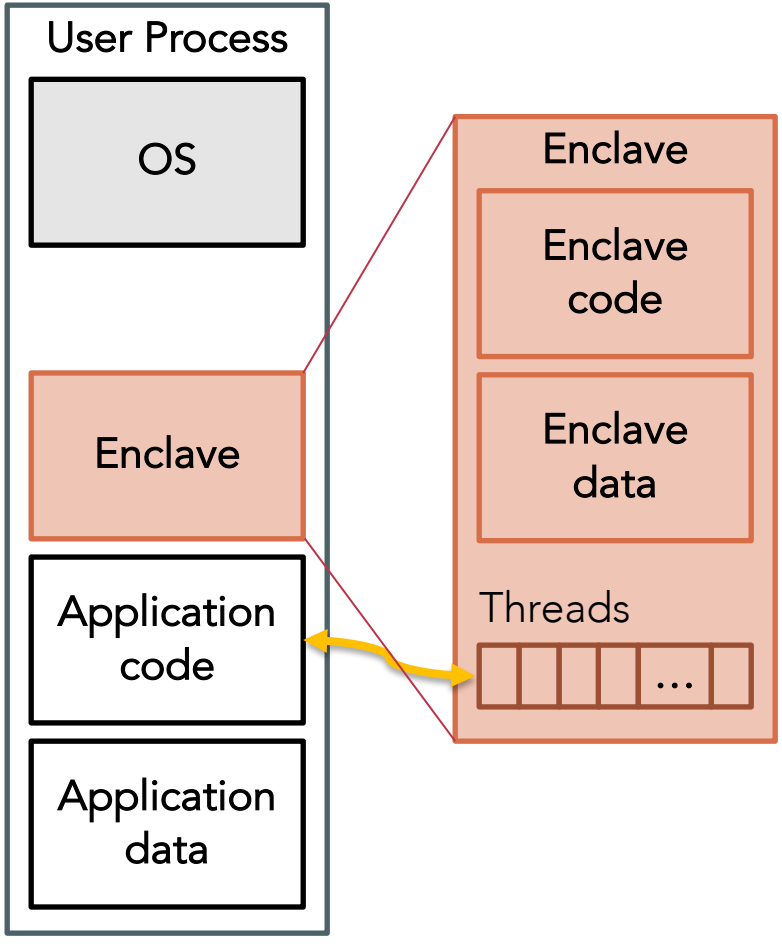
A Hardware-Software Co-design for Efficient Intra-Enclave Isolation

Jinyu Gu, Bojun Zhu, Mingyu Li, Wentai Li, Yubin Xia, Haibo Chen

Institute of Parallel and Distributed Systems (IPADS), Shanghai Jiao Tong University

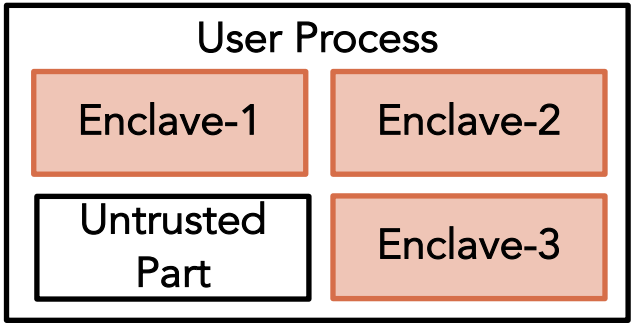
SGX Enclave

- **Trusted Execution Environment**
  - Build enclaves within the process
  - Protect security-sensitive code and data
  - Switch between non-enclave and enclave
- **Security**
  - Defend against untrusted software outside
  - Trusted Computing Base (TCB)
    - CPU and the enclave
    - Exclude privileged software including OS

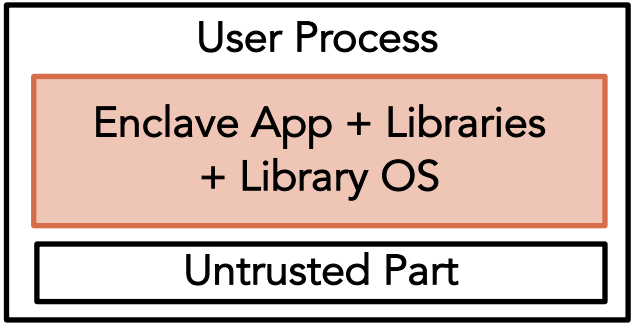


SGX Programming Model

- **Fine-grained separation**
  - Isolate the minimal security-sensitive part
  - Construct multiple enclaves
  - Issue: cross-boundary overhead; separation makes programming hard

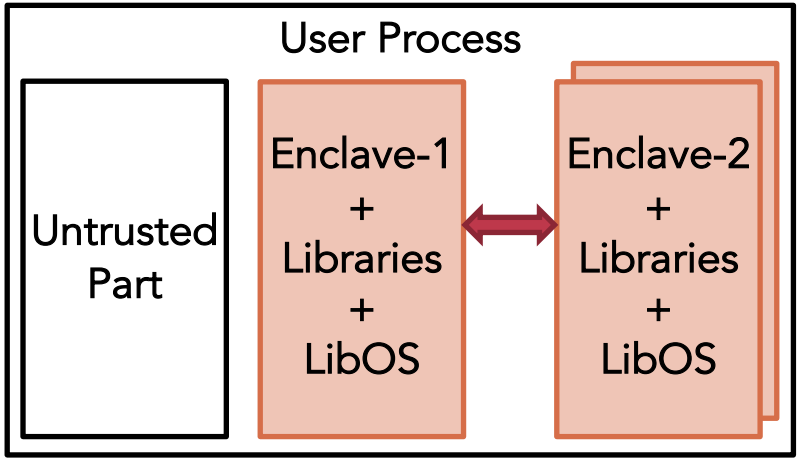


- **Monolithic enclave**
  - Run third-party libraries or LibOS in an enclave
  - No need to modify applications
  - Issue: the bloating of software TCB



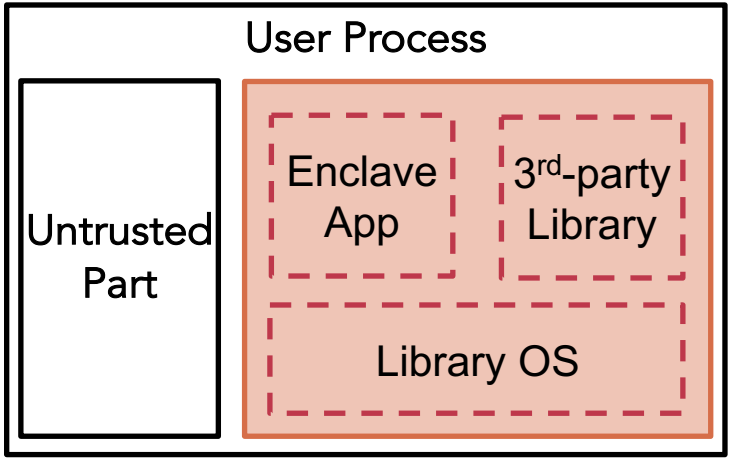
Candidate Solution: Multi-Enclave Isolation

- **How to achieve both ends**
  - Isolation: run modules/components in different enclaves
  - Ease-to-program: deploy libraries and LibOS in each enclave
  - Issues
    - Cross-enclave communication
    - Build enclave is not cheap
    - Enclave memory consumption



Candidate Solution: Intra-Enclave Isolation

- **How to achieve both ends**
  - Introduce isolation within one enclave by using SFI
    - SFI: software fault isolation (instrumenting memory accesses)
  - Issues
    - Extra runtime overhead
    - Enlarge the enclave code size
    - Inflexibility in the memory layout

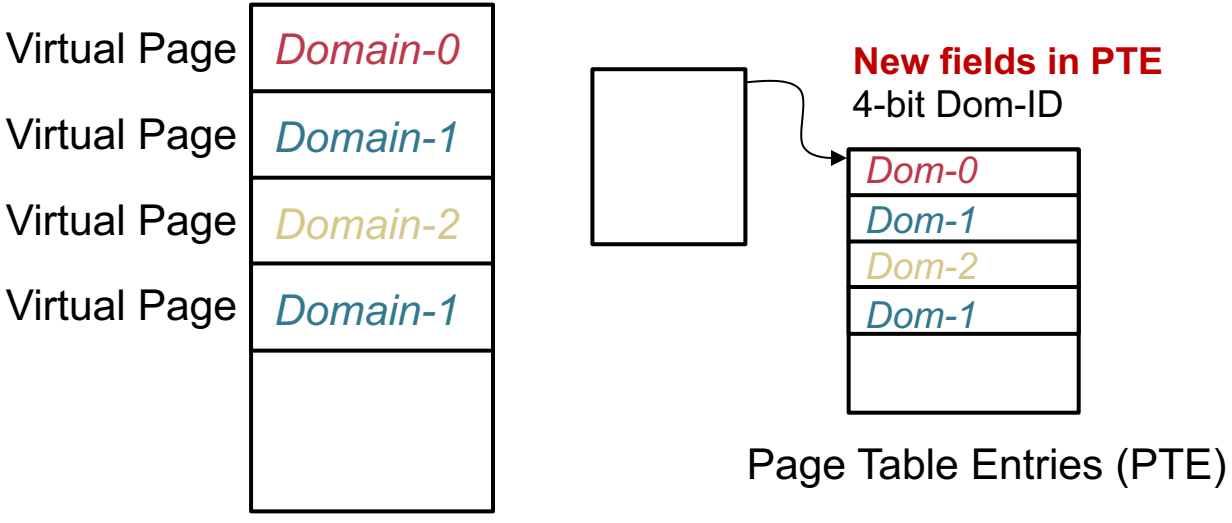


Opportunity: Intel MPK

- **Design goal: high-efficient intra-enclave isolation**
- **A hardware feature for intra-process memory isolation**
  - MPK: Memory Protection Key

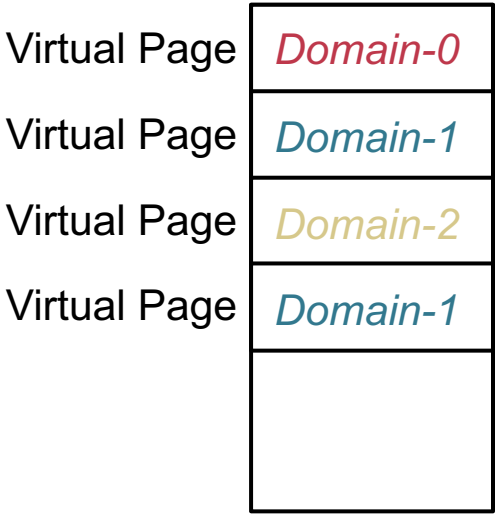
Opportunity: Intel MPK

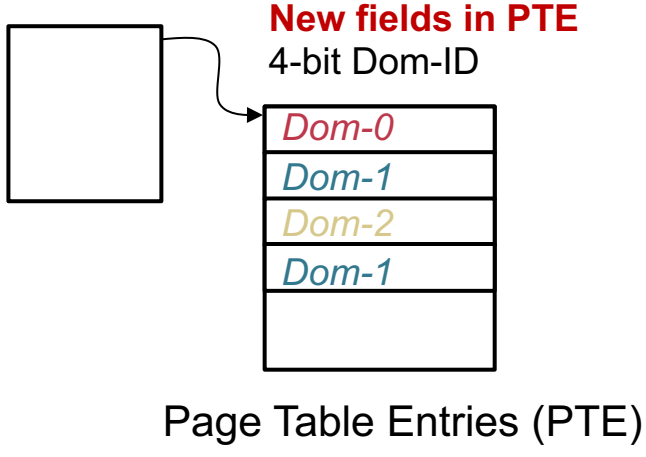
- **A hardware feature for intra-process memory isolation**
  - Partition different memory domains within one address space

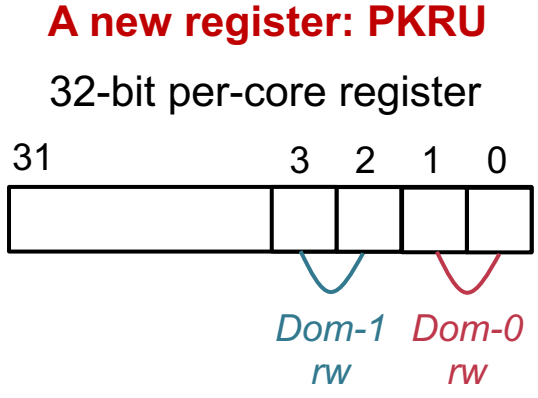


Opportunity: Intel MPK

- **A hardware feature for intra-process memory isolation**
  - Partition different memory domains within one address space
  - Allow different threads have different memory views

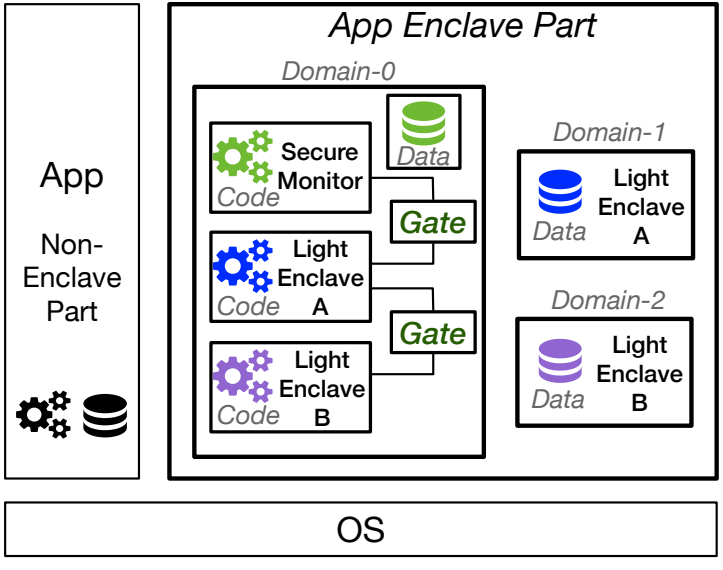






Our Work: LightEnclave

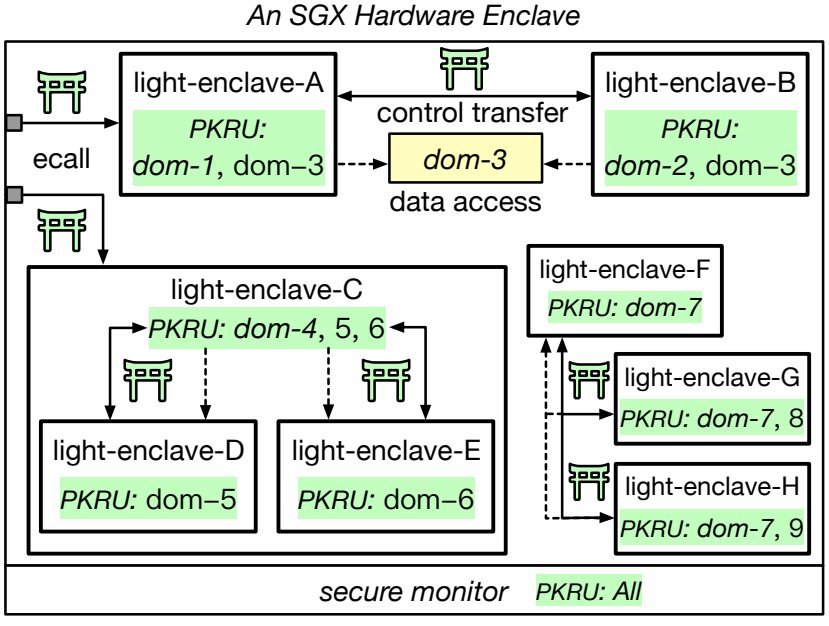
- **Intel SGX Enclave + Intel MPK Domain**
  - Use MPK to partition the enclave memory into multiple domains
  - Build multiple light-enclaves within one SGX enclave



- Each light-enclave contains one private memory domain (read-write) for hosting private data
- The code sections are located in domain-0 (execute-only)
- The interaction between light-enclaves are through lightweight gates (a piece of code)
- Secure monitor runs with the highest privilege and manages the light-enclaves

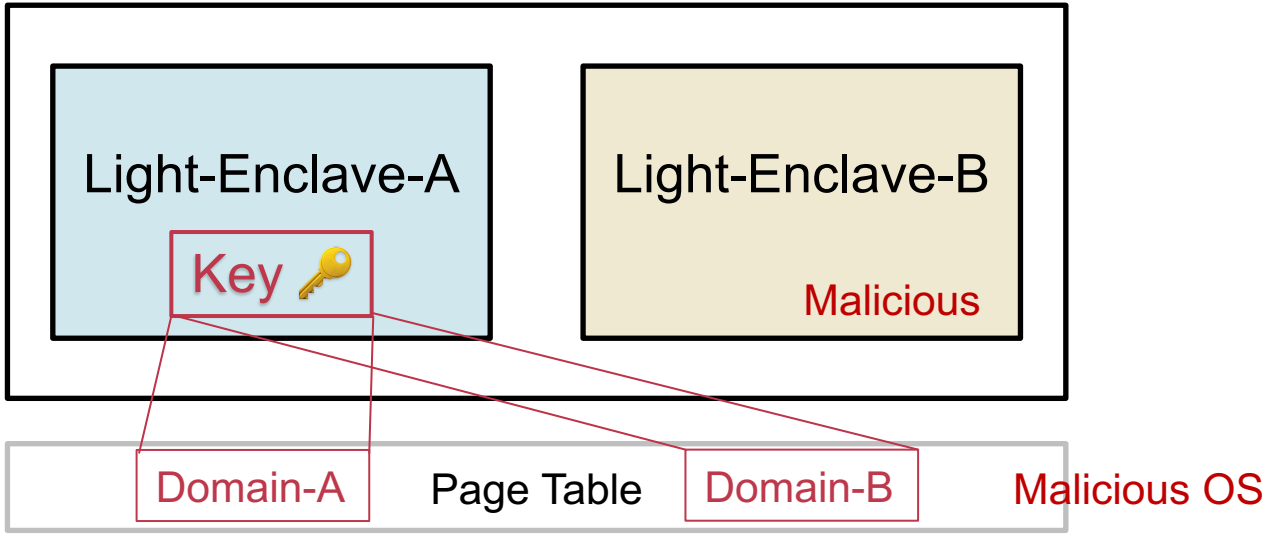
Light-Enclave

- **The identity of a light-enclave**
  - Domain access permission in PKRU
- **Interfaces**
  - ocall/ecall (also setting PKRU)
  - Gates between light-enclaves
- **Interaction of light-enclaves**
  - Switching PKRU for control transfer
  - Sharing a domain for data transfer
- **Flexible trust model**
  - Mutual-distrusted ones (e.g., light-enclave-A and light-enclave-B)
  - hierarchical light-enclaves (e.g., C, D, and E)



Challenge: Conflict Trust Model

- **SGX assumes untrusted OS while MPK trusts OS**
  - Compromised light-enclave or OS alone cannot steal the security-sensitive data
  - Collusion attack: OS modifies the domain mappings in the page table and then an inside malicious light-enclave can arbitrarily steal data from other light-enclaves



General Idea on Defending

- **The root cause of the attack**
  - The untrusted OS can manipulate the page table
  - MPK isolation needs the OS to faithfully configure the page table
- **Solution**
  - Deprived the OS of the ability to arbitrarily modify the MPK domain setting on the enclave pages
  - Non-intrusive hardware extensions for validating the domain setting at enclave initialization time and runtime

Hardware Extensions

Can be implemented in the SGX microcode

Procedures	Effects	Brief Description of the Extensions
Initialization	*EADD*	Set the page's domain-ID in EPCM according to the new *SECINFO*.
	ECREATE	Include the new fields in *SECS* into the enclave measurement.
Swapping	*EWB*	The domain-ID of an enclave page will be used to update the hash (*PCMD.MAC*).
	ELD	Ensure the domain-ID of a loaded-back page is unchanged.
Dynamic Paging	*EAUG*	Similar to *EADD*. Set the domain-ID for a new enclave page.
	EMODPE	Change the domain-ID of an enclave page in the EPCM at runtime.
	EACCEPT(COPY)	Check whether a page's domain-ID in EPCM matches the desired one.
Runtime	*EENTER/ERESUME*	Check whether MPK is enabled as required in *SECS*.
	Memory Access	Ensure the domain-IDs in the EPCM and in the page table match before accessing.
	ENCLU Execution	Check the capability table when executing sensitive *ENCLU* instructions.

Please refer to the paper if interested

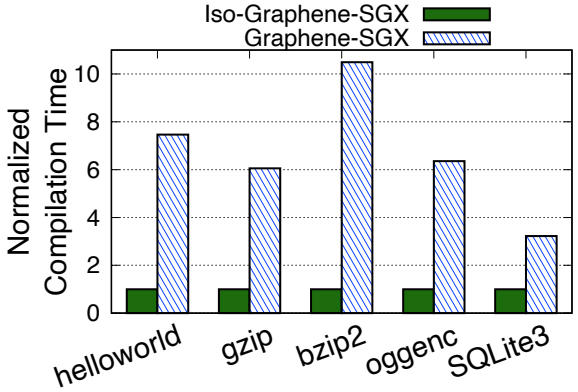
Experiments

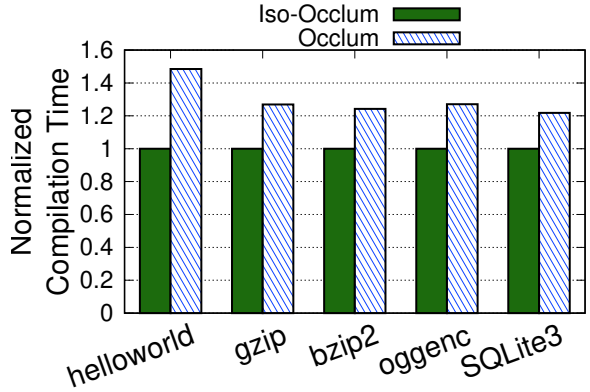
- **Intel i7-10700 IceLake CPU (SGXv1, MPK)**
- **Linux kernel 5.4.110**
- **Occlum (commit 0a06c898)**
  - Iso-Occlum: integrate LightEnclave into it
- **Graphene-SGX (commit 9c226c9a)**
  - Iso-Graphene-SGX: integrate LightEnclave into it

[1] **Occlum: Secure and Efficient Multitasking Inside a Single Enclave of Intel SGX (ASPLOS 20)**
[2] **Graphene-SGX: A Practical Library OS for Unmodified Applications on SGX (ATC 17)**

GCC

- **Setup**
  - GCC (v4.4.5) spawns and executes cc1, as, collect2 and ld for compiling different programs
  - Use ApacheBench to generate clients to fetch 10KB web pages





Conclusion

- **LightEnclave**
  - A hardware-software co-design for efficient intra-enclave isolation
  - Efficiency: MPK-assisted lightweight memory isolation and fast interaction
  - Security: high-security assurance for a light-enclave
  - Flexibility: ease-to-use and can be integrated into existing SDK

Thanks!

Email: gujinyu@sjtu.edu.cn