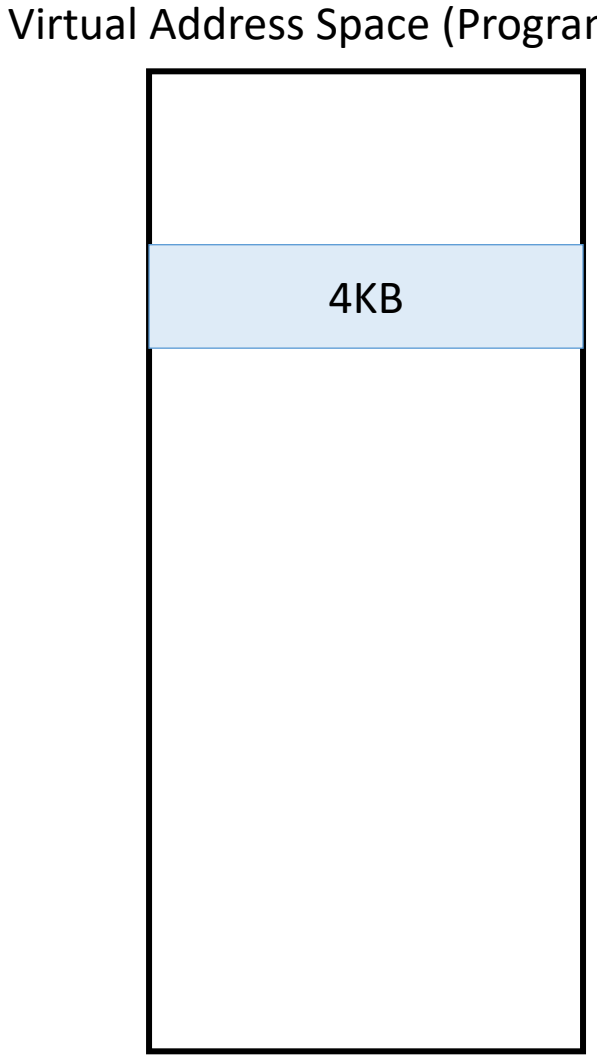
Intel SGX

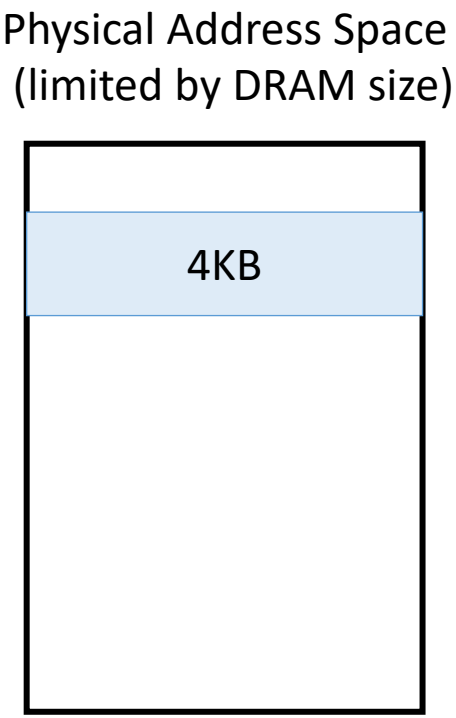
Mengjia Yan

Fall 2020

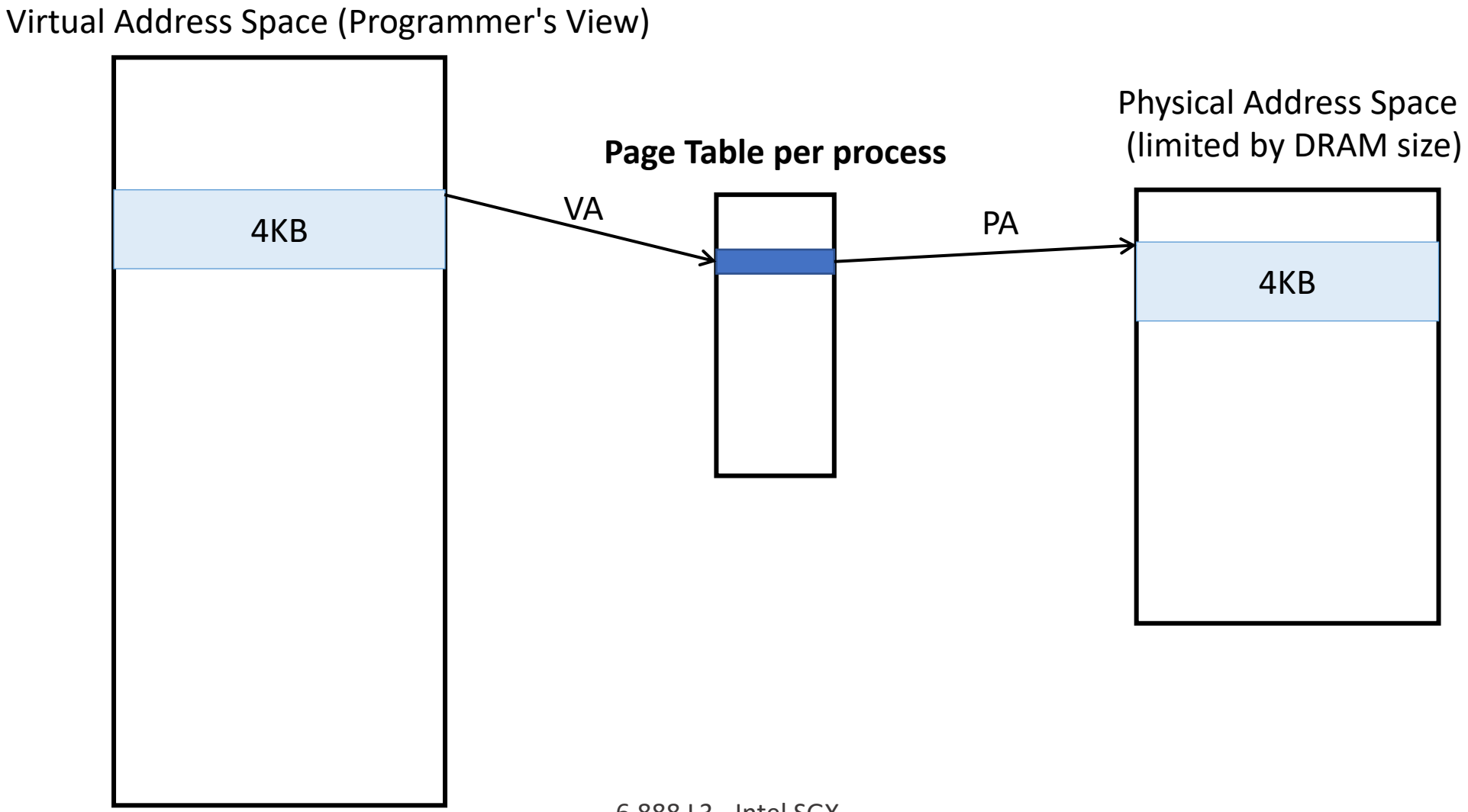
Based on slides of Intel SGX Tutorial

Recap: Address Translation

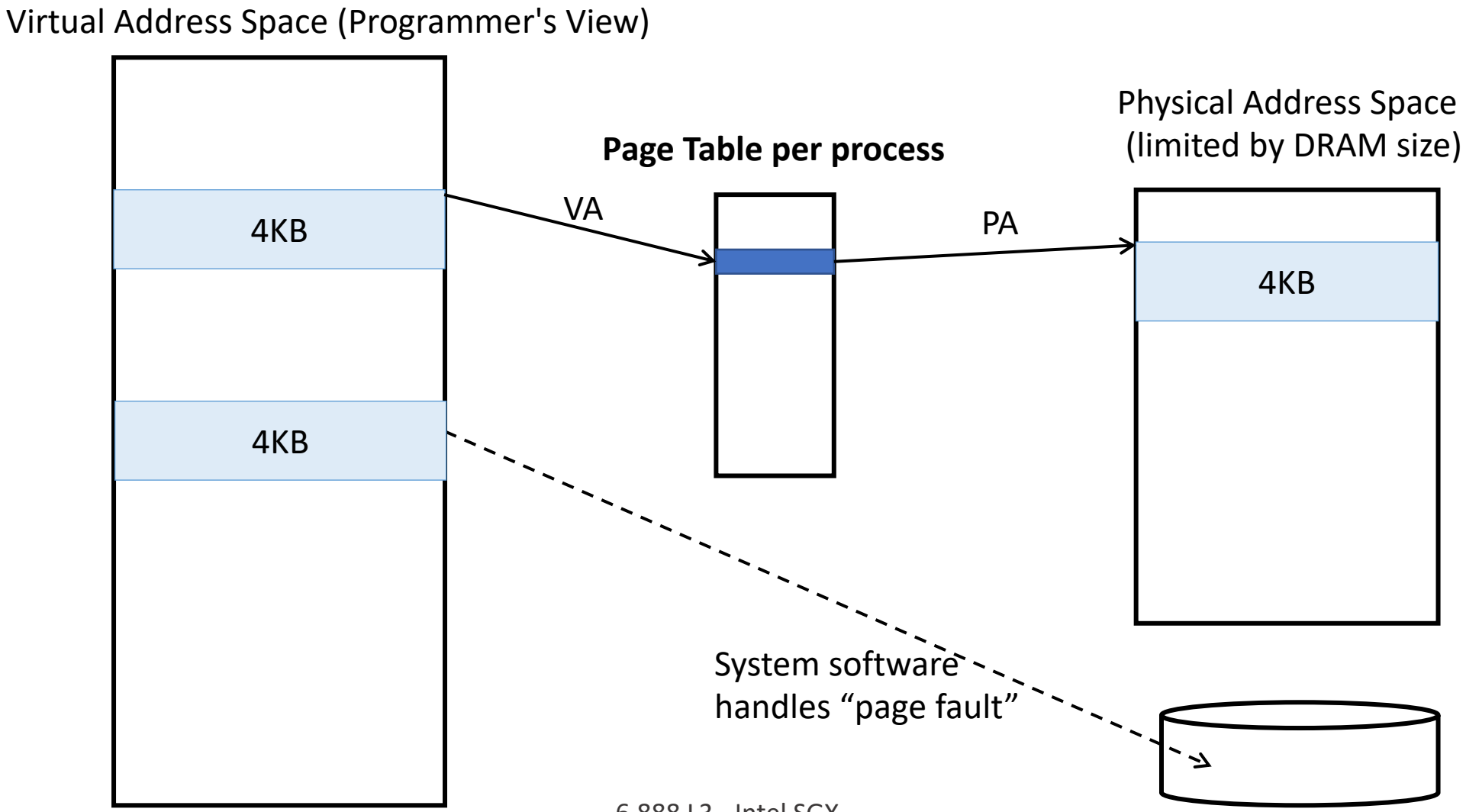




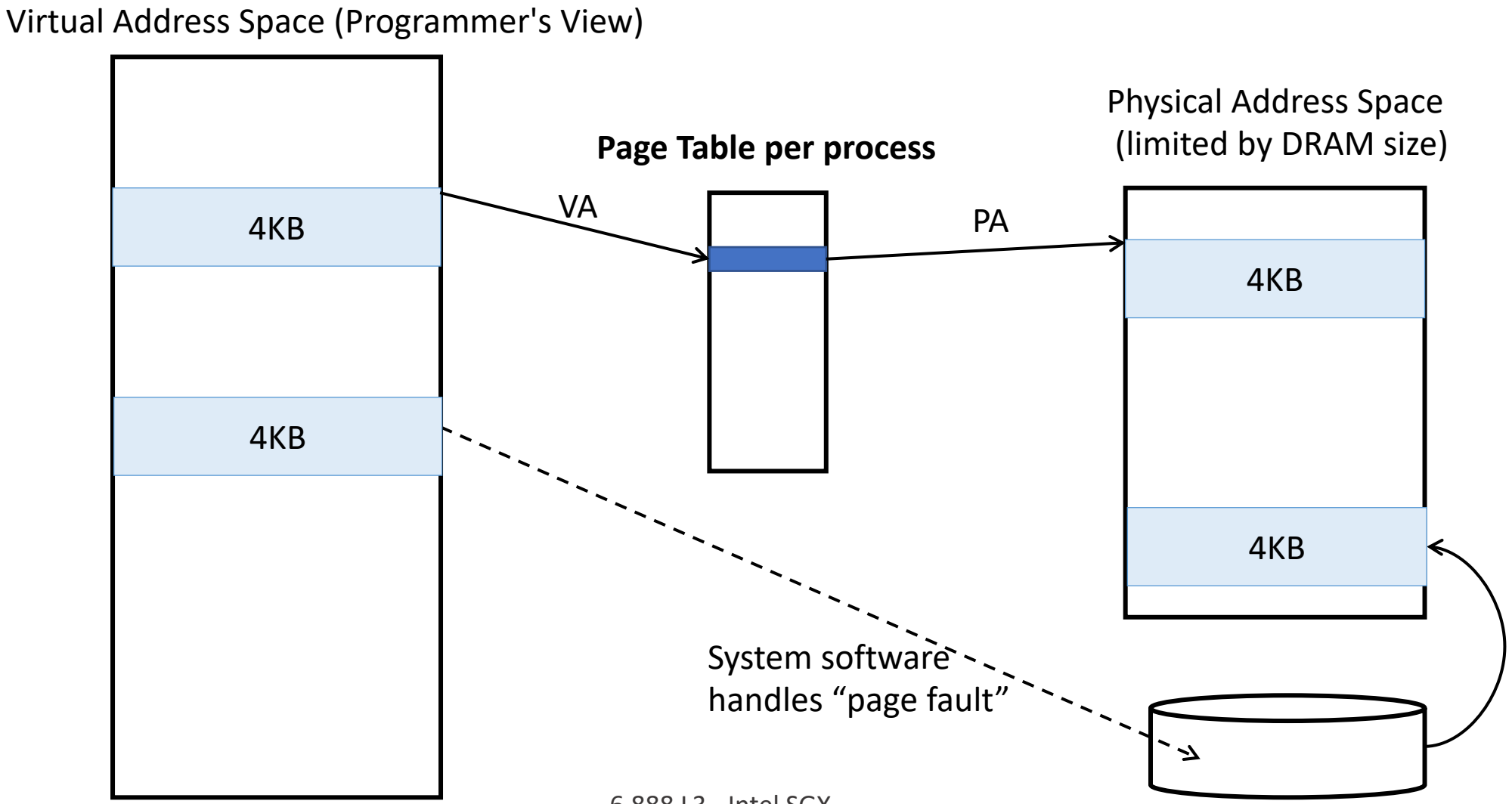
Recap: Address Translation



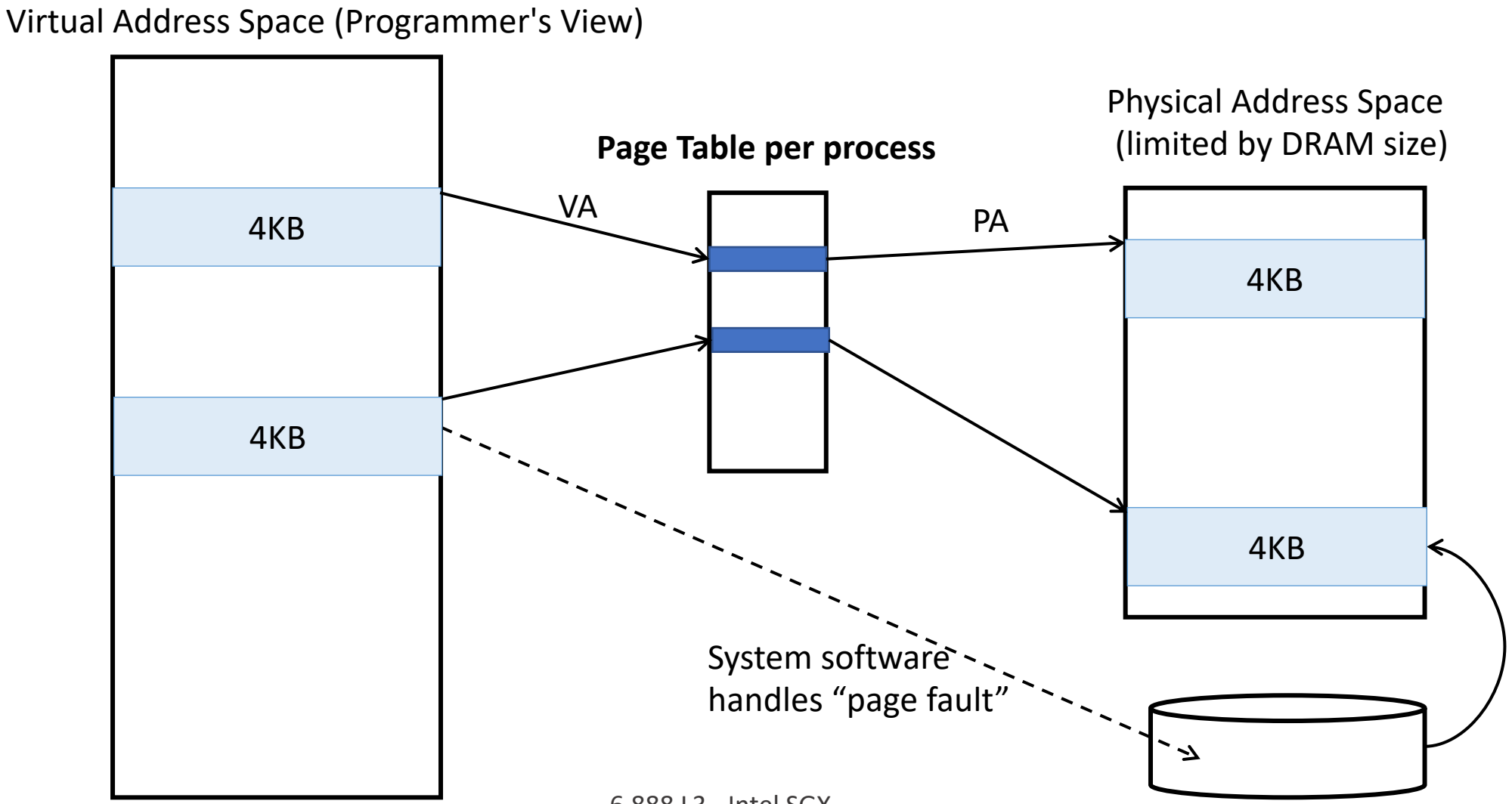
Recap: Address Translation



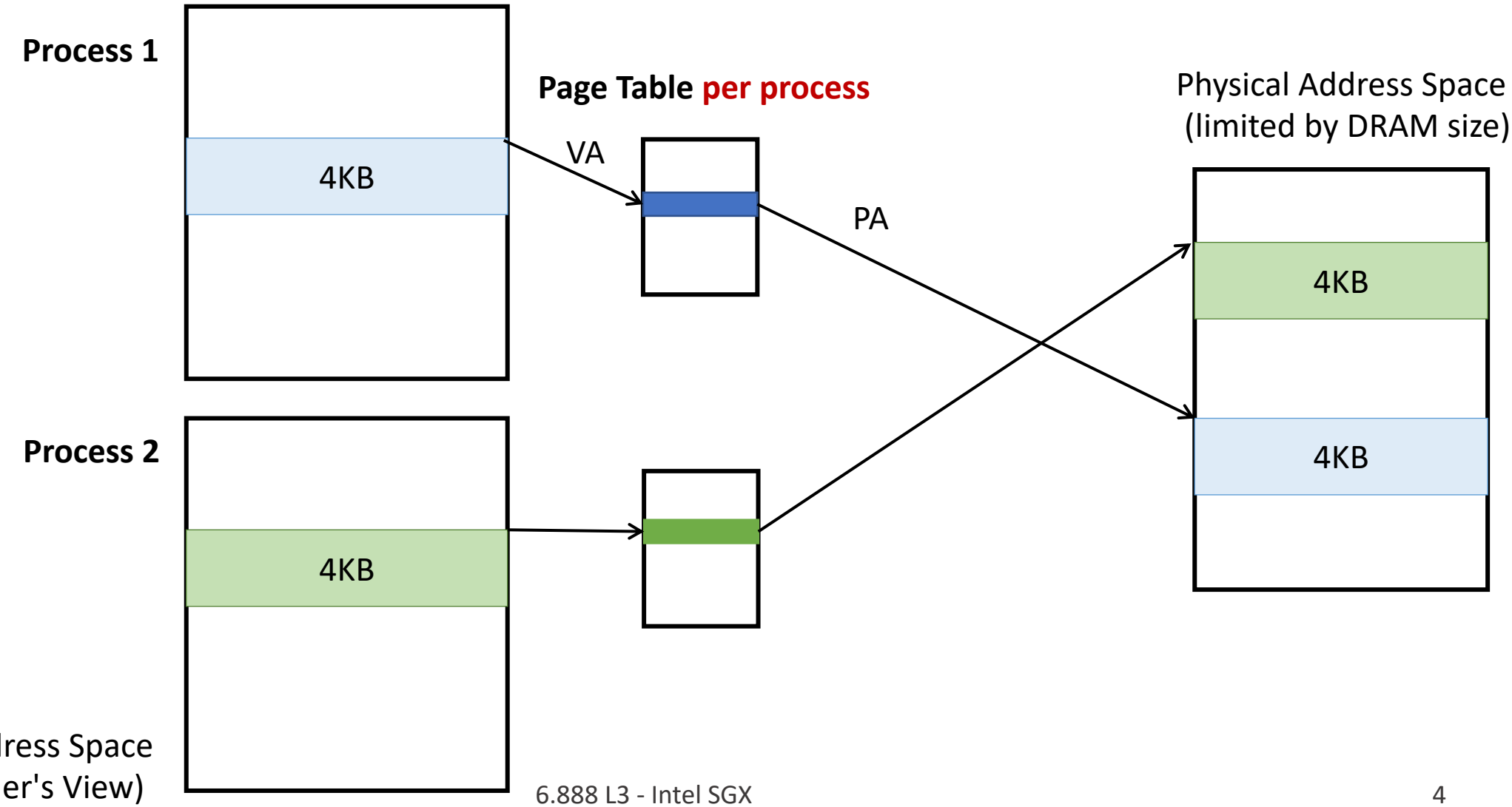
Recap: Address Translation



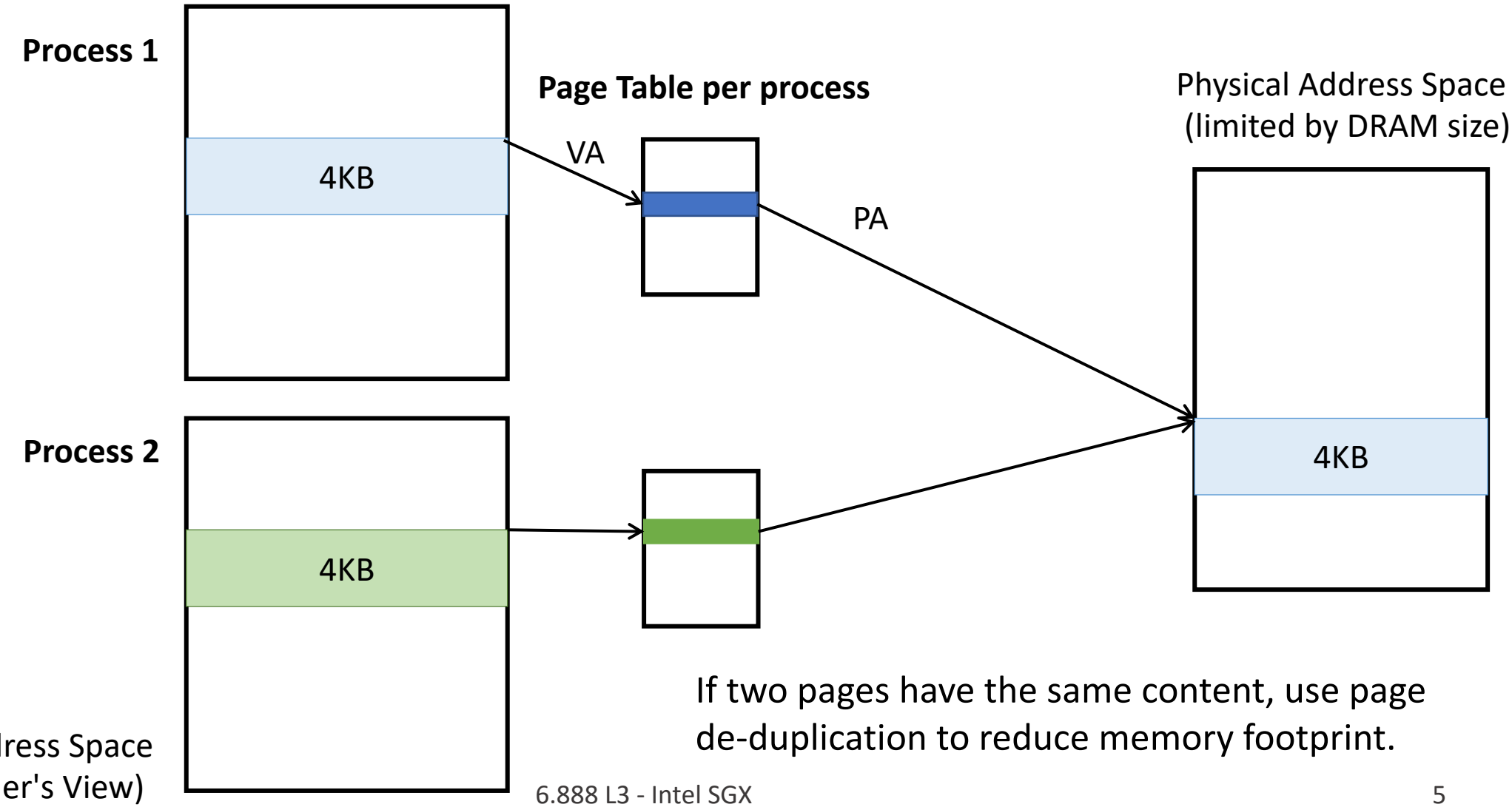
Recap: Address Translation



Recap: Process Isolation

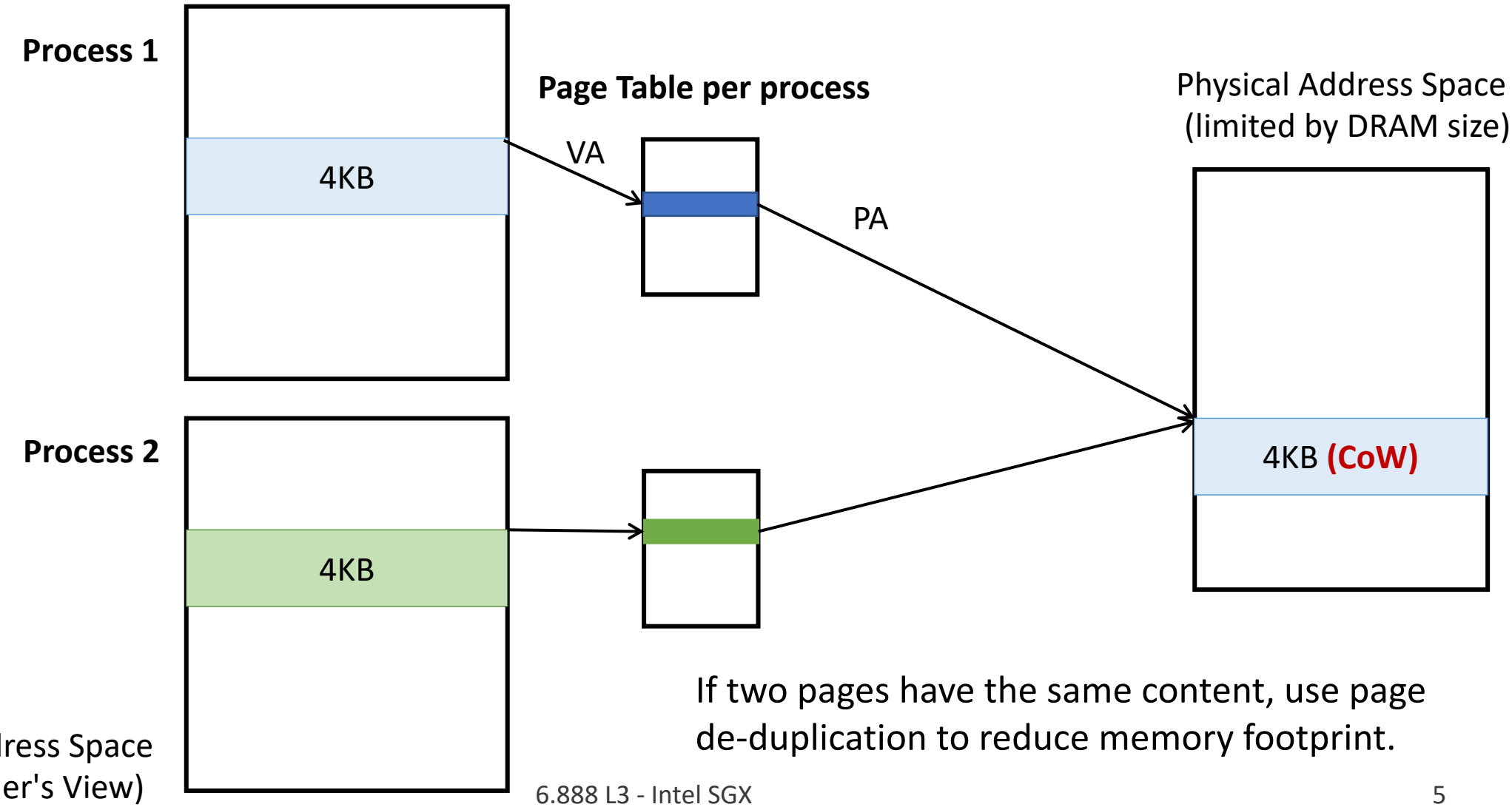


Page De-duplication and Copy-on-write



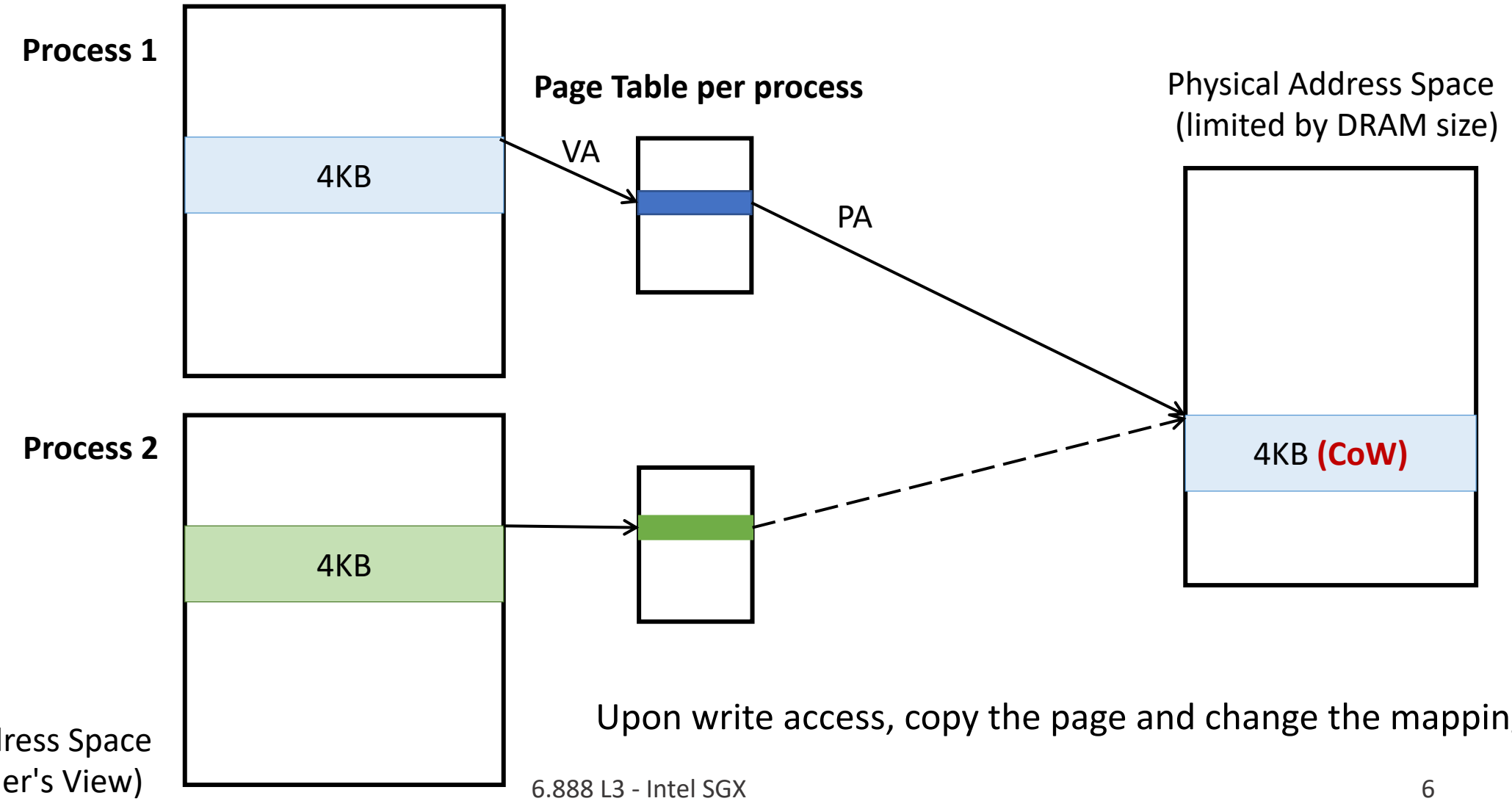
If two pages have the same content, use page de-duplication to reduce memory footprint.

Page De-duplication and Copy-on-write



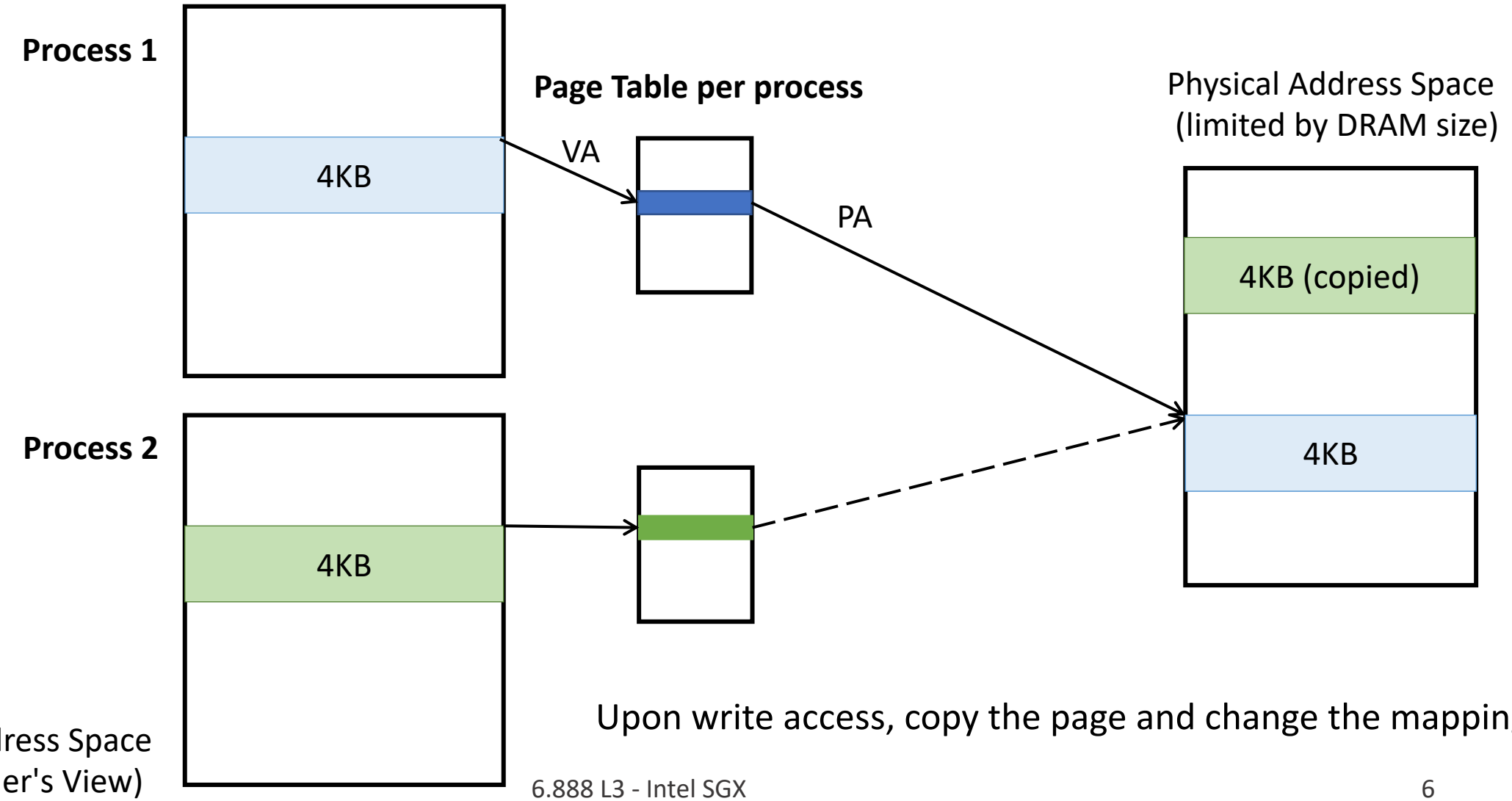
If two pages have the same content, use page de-duplication to reduce memory footprint.

Page De-duplication and Copy-on-write



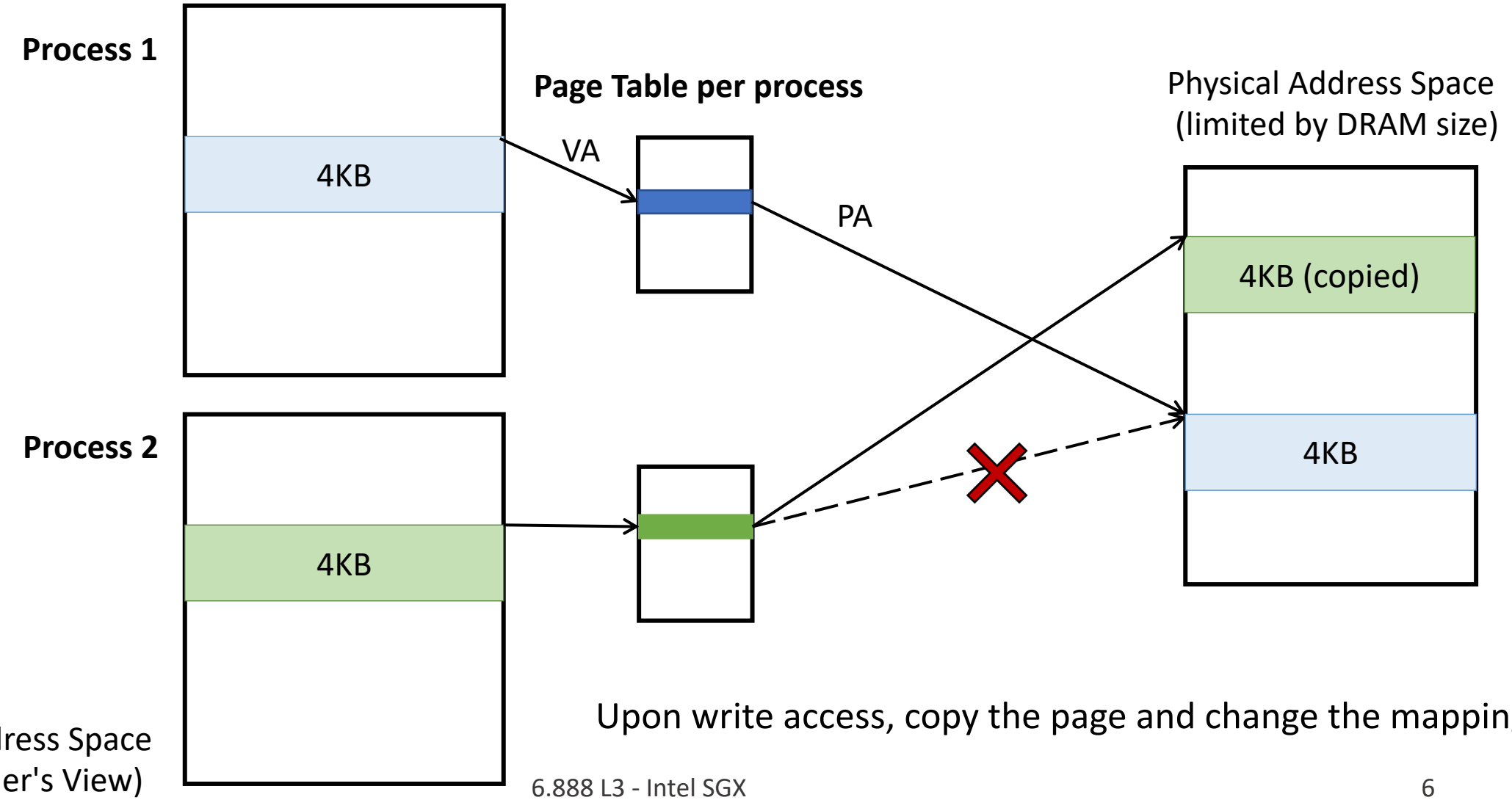
Upon write access, copy the page and change the mapping.

Page De-duplication and Copy-on-write



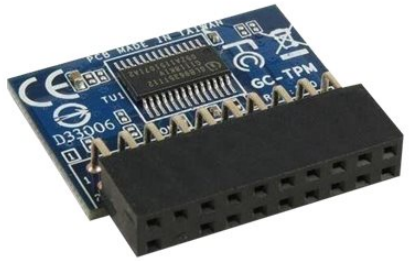
Upon write access, copy the page and change the mapping.

Page De-duplication and Copy-on-write

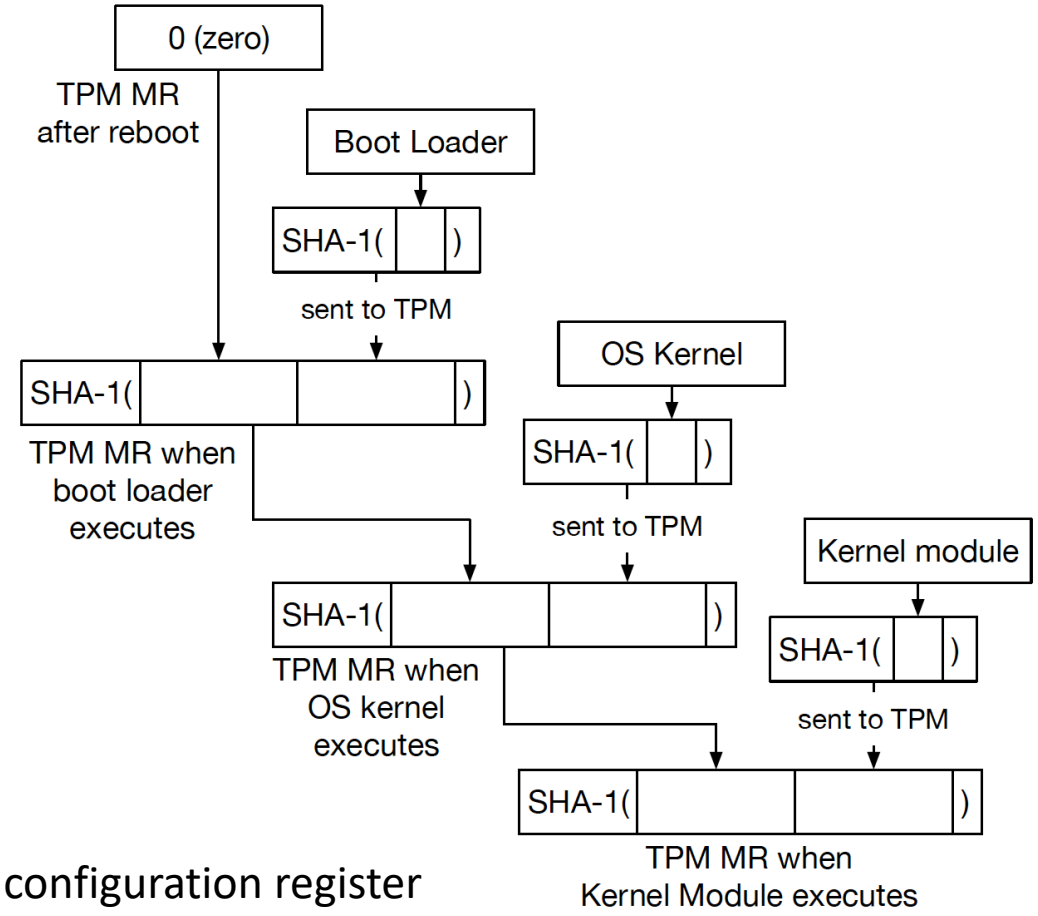


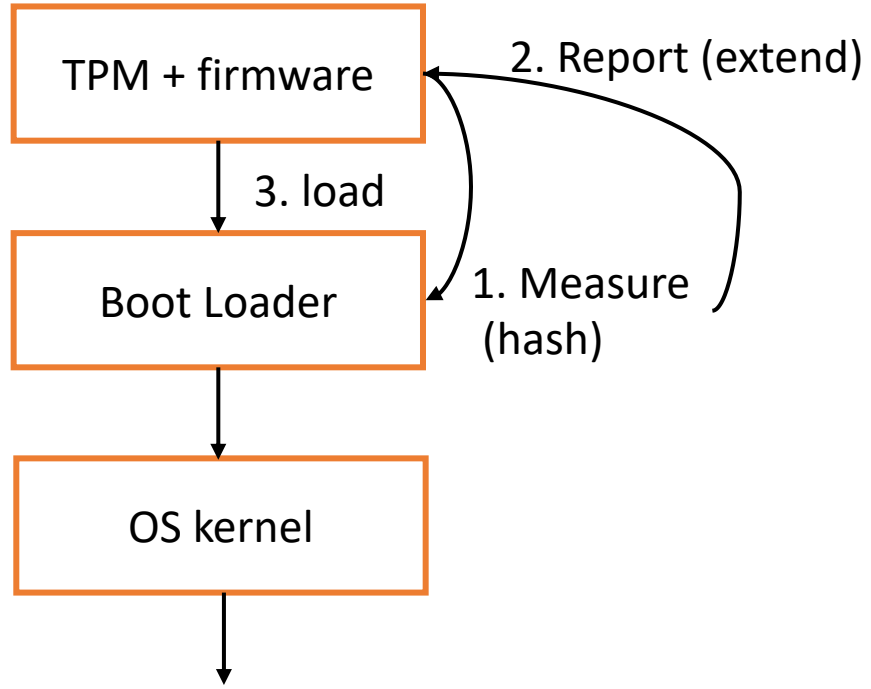
Upon write access, copy the page and change the mapping.

Recap: Secure Boot



- Static root of trust for measurement (SRTM)



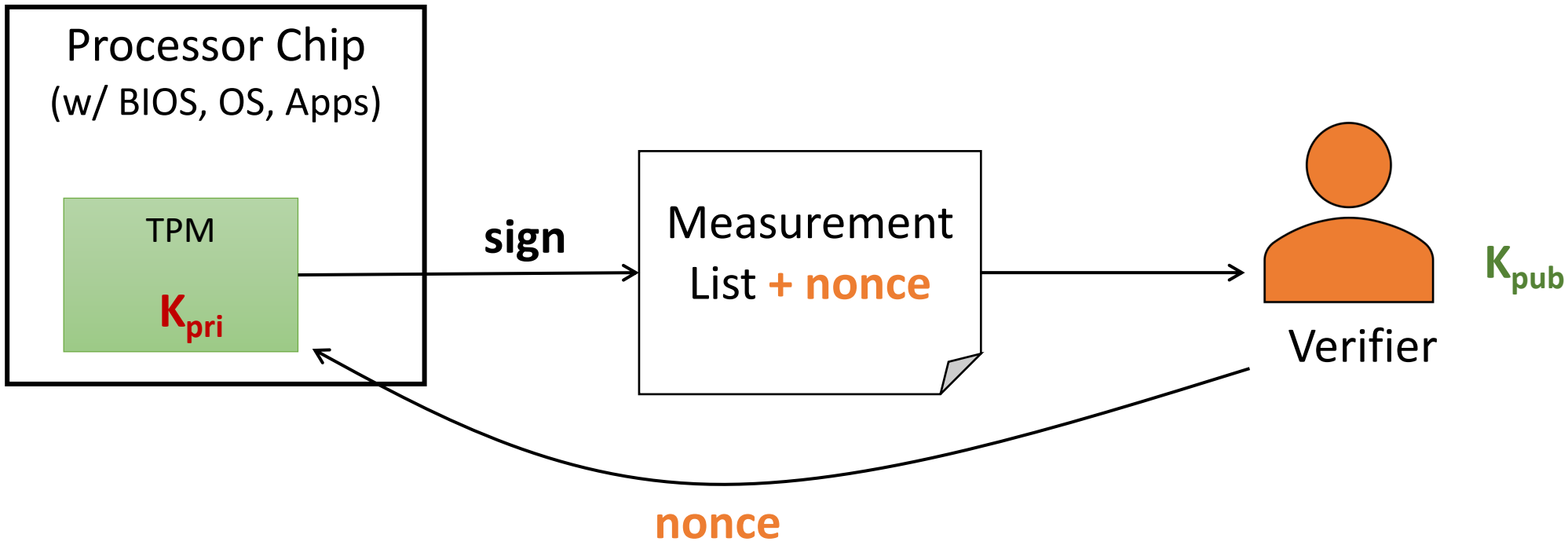


Compared to expected values locally or submitted to a remote attestor.

PCR: platform configuration register

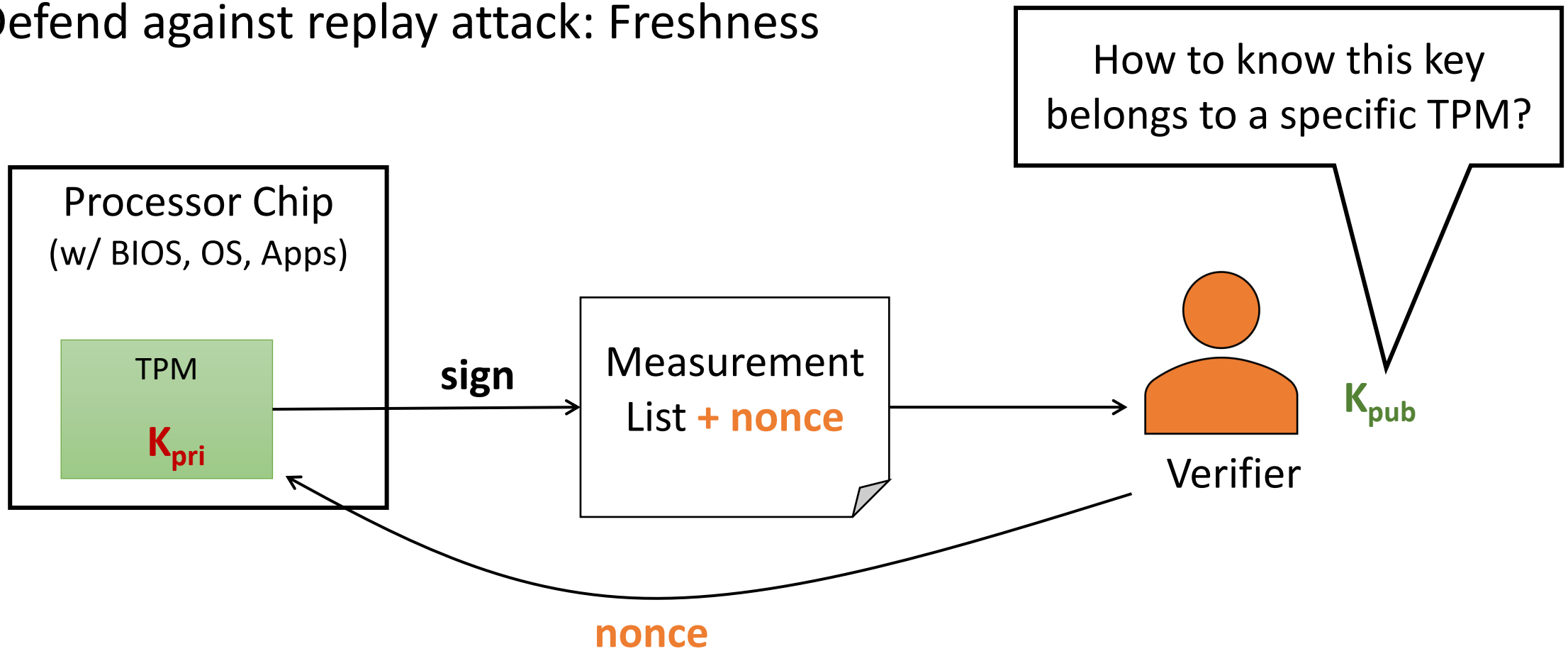
Software Attestation

- Defend against replay attack: Freshness



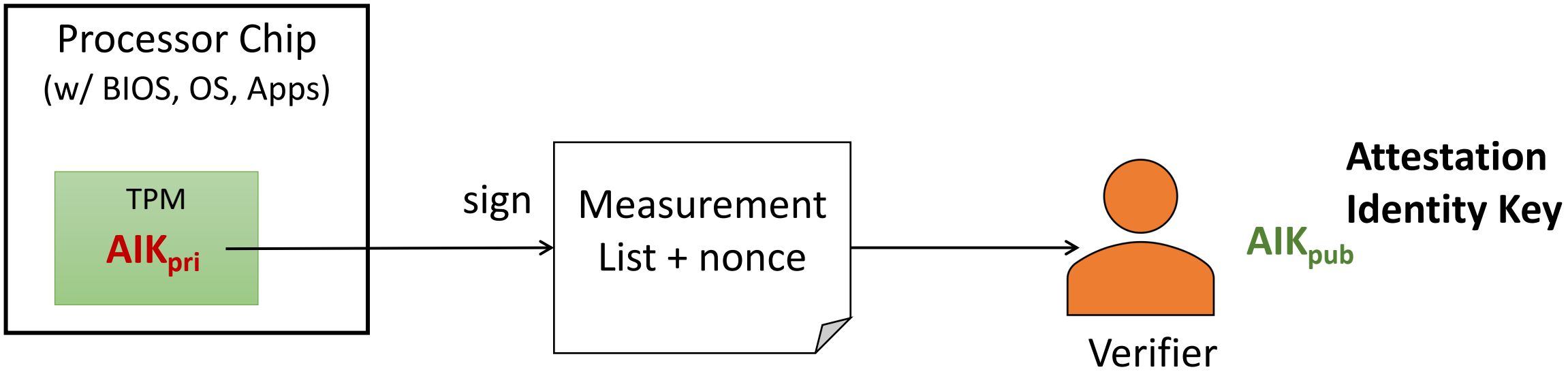
Software Attestation

- Defend against replay attack: Freshness



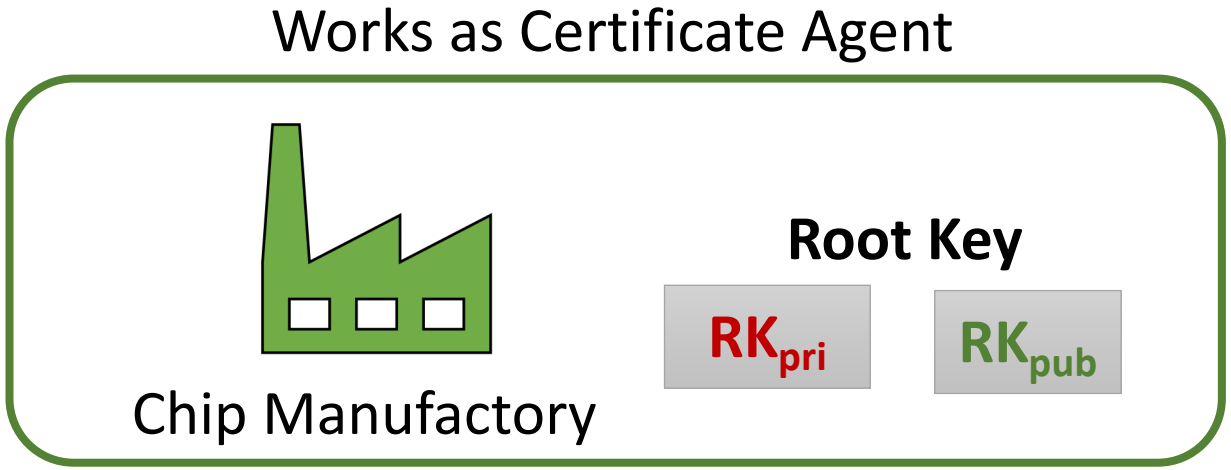
Software Attestation

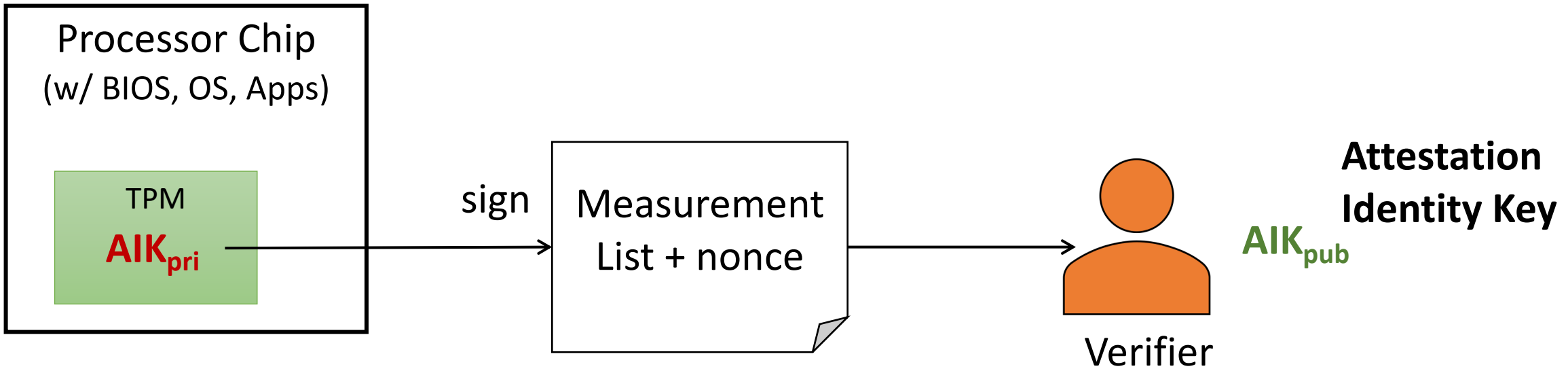
- Need public key infrastructure



Software Attestation

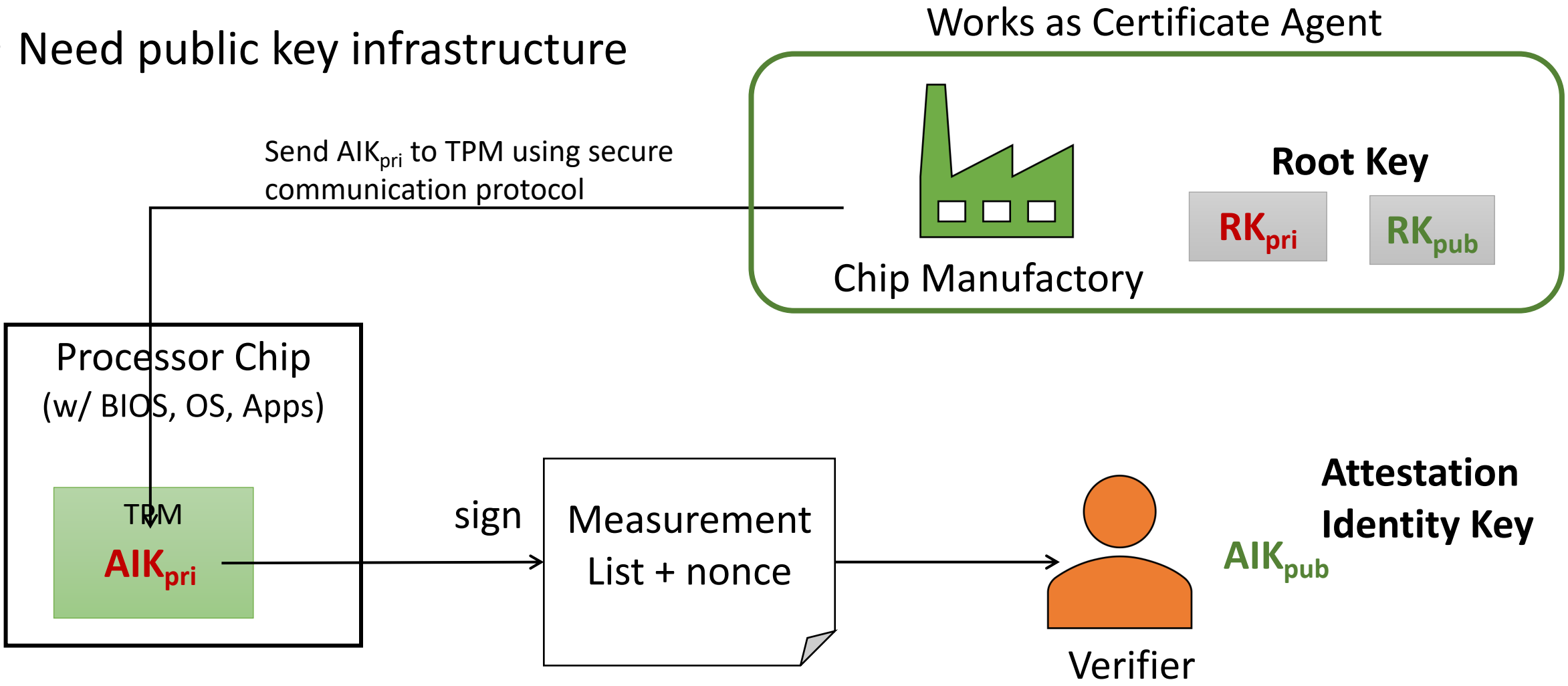
- Need public key infrastructure





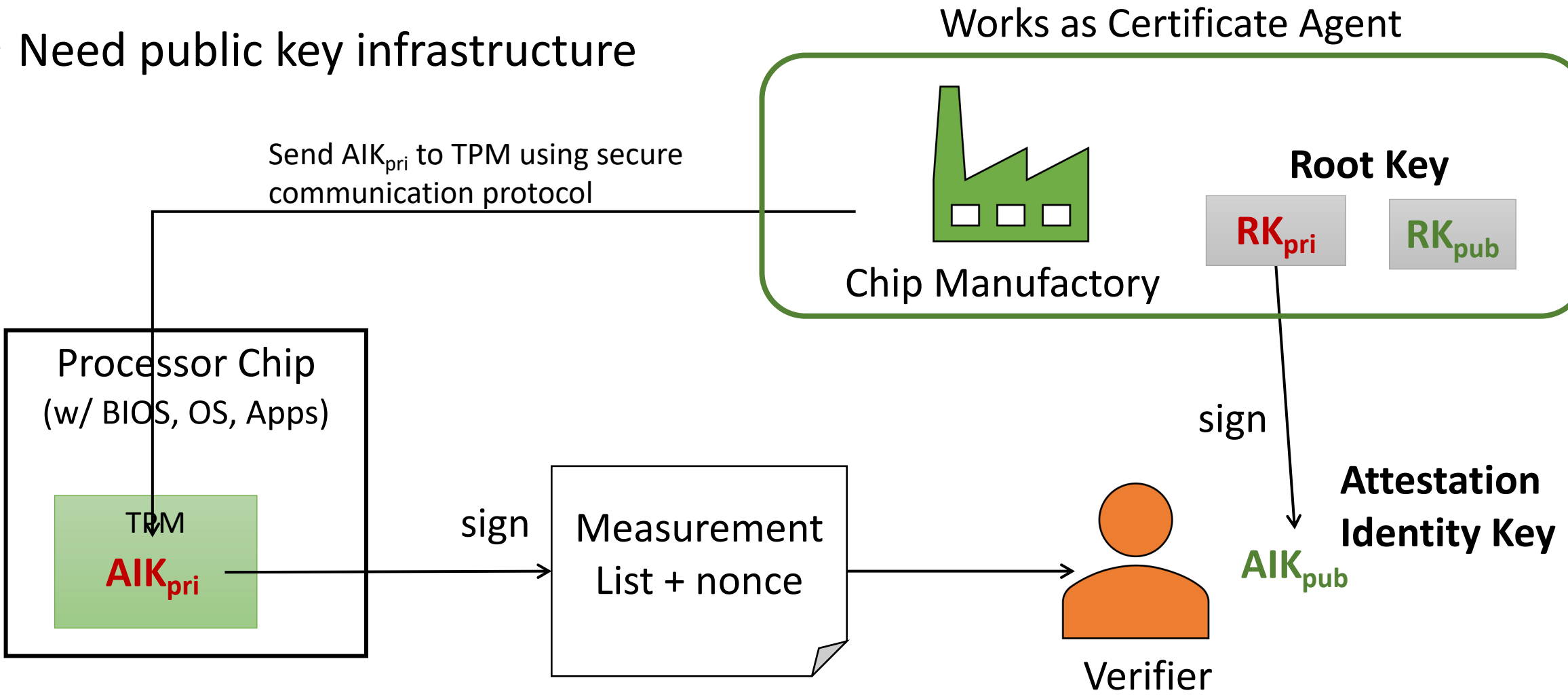
Software Attestation

- Need public key infrastructure

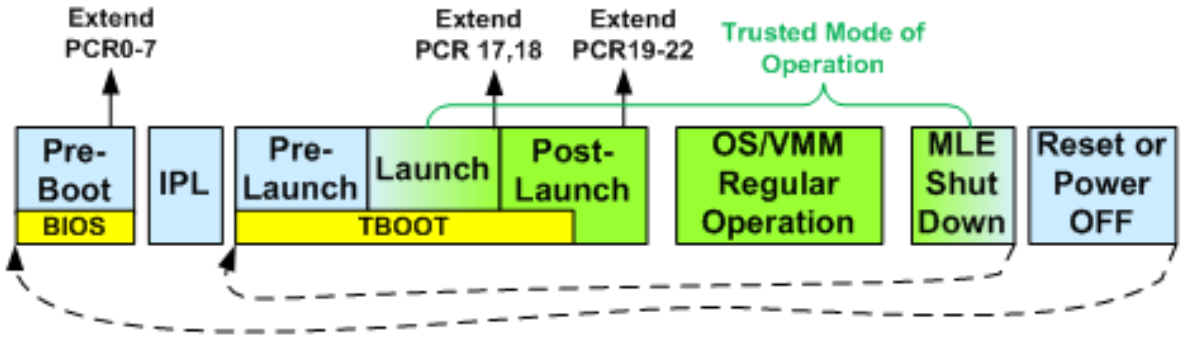


Software Attestation

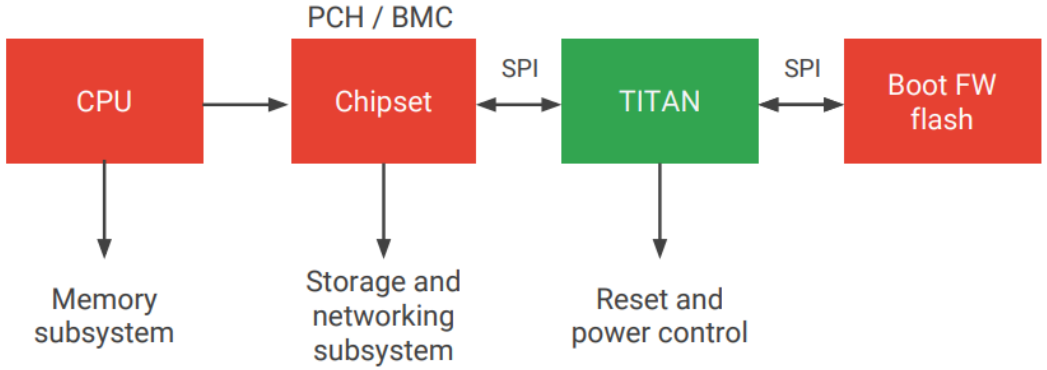
- Need public key infrastructure

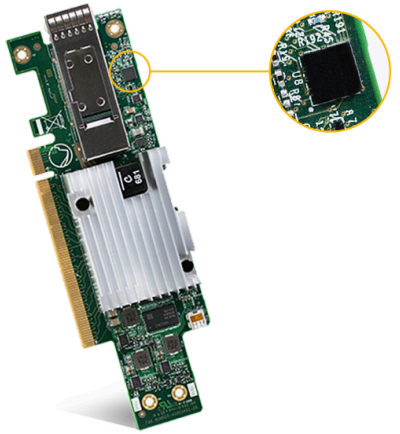


Intel TXT, AMD PSP, Google Titan



Intel TXT Dynamic trust of measurement

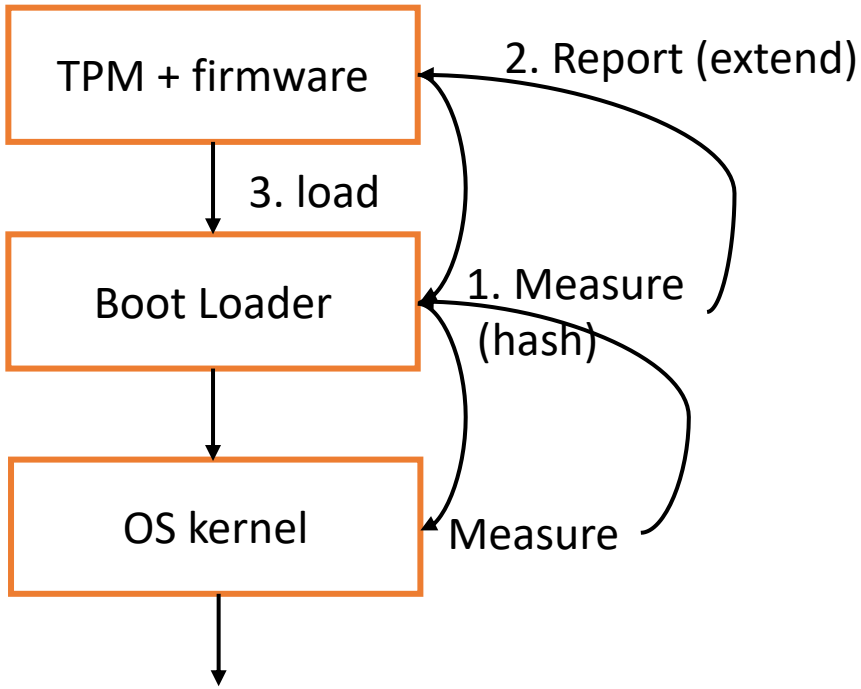




from https://www.hotchips.org/hc30/1conf/1.14_Google_Titan_GoogleFinalTitanHotChips2018.pdf

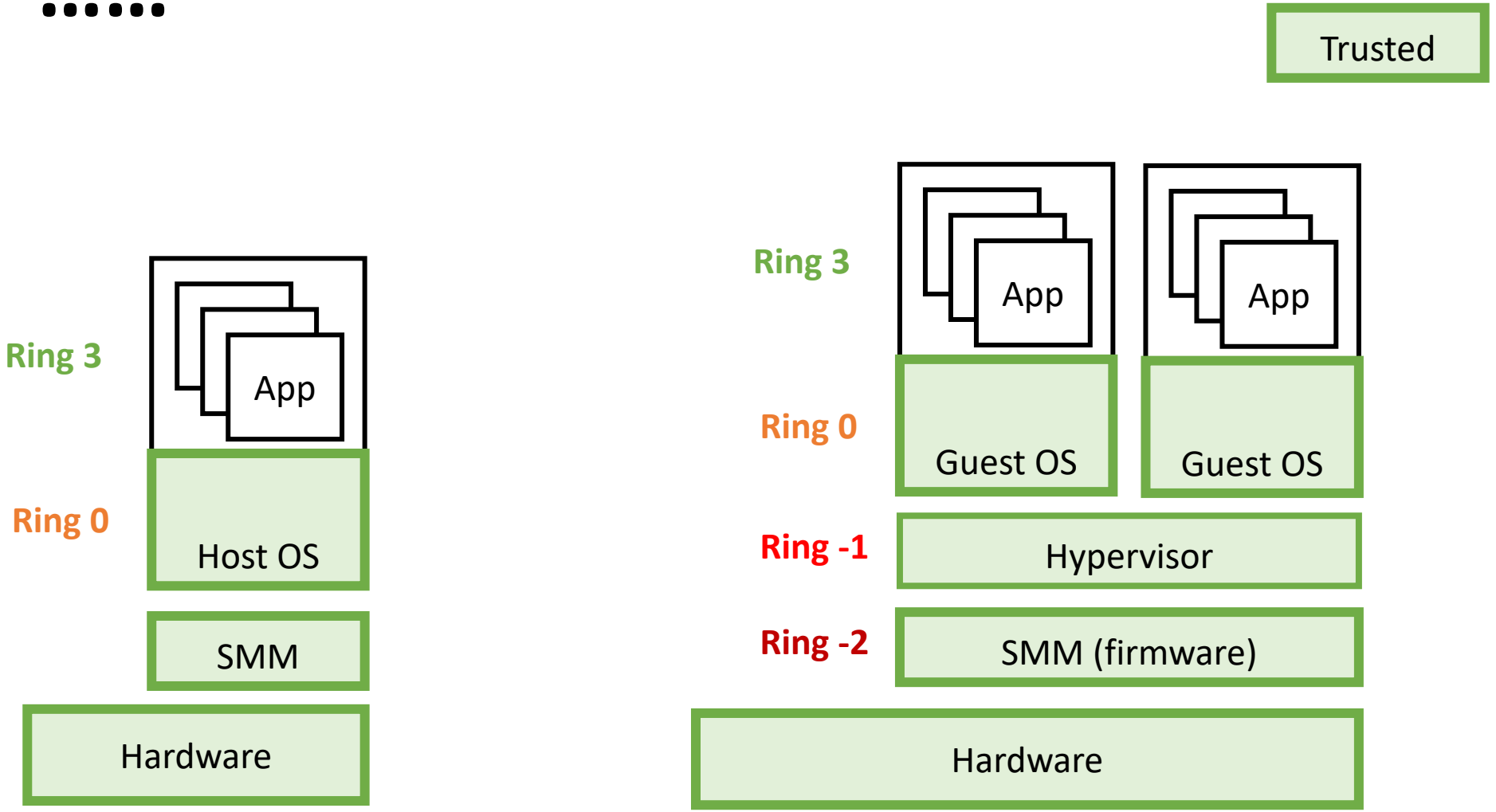
Security Vulnerabilities of Using TPM

- Vulnerable to bus tapping attacks
- TPM Reset attacks
  - SW reports hash values
- Bugs in the trusted software

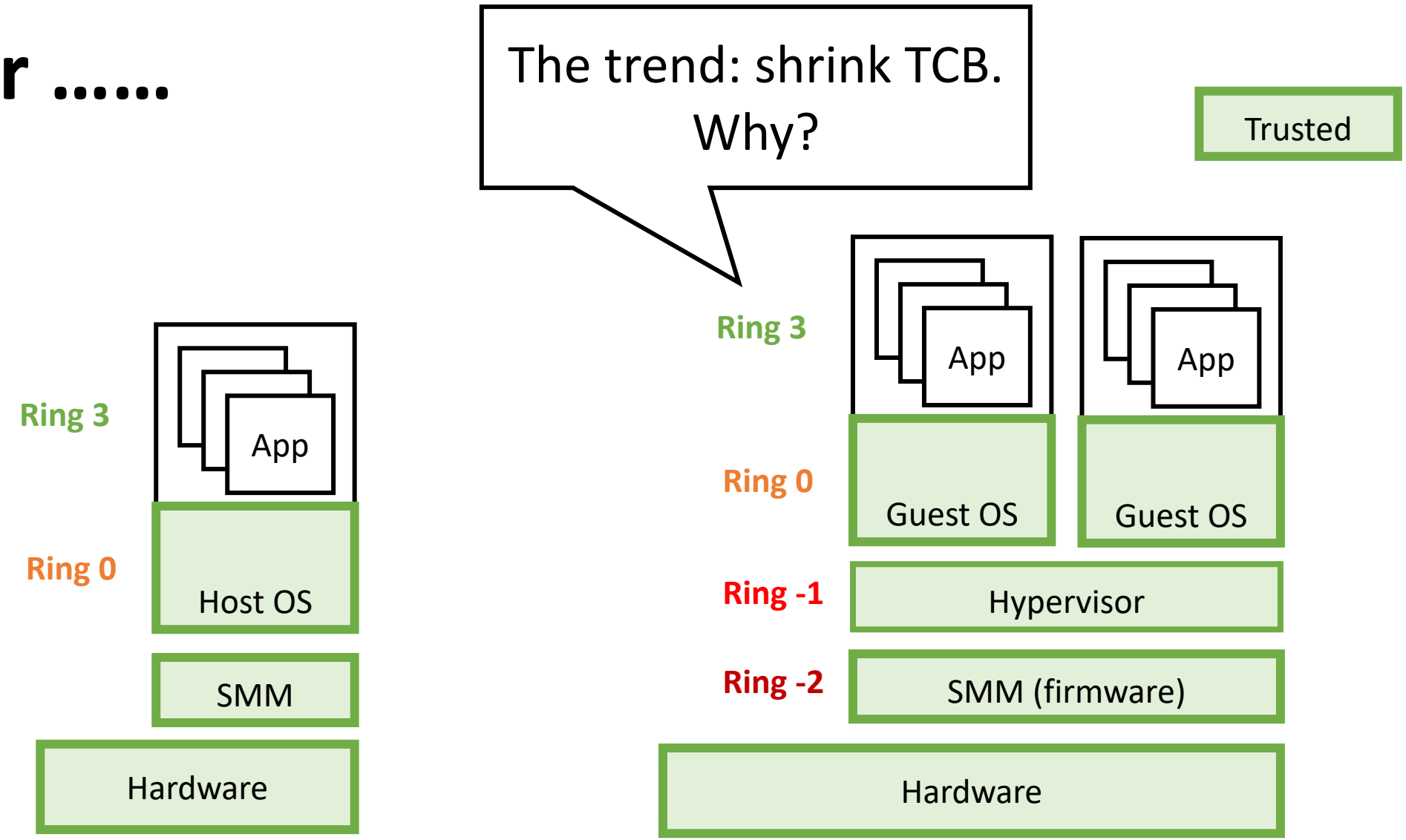


Han et al. A Bad Dream: Subverting Trusted Platform Module While You Are Sleeping. Usenix Security'18
Wojtczuk et al. Attacking Intel TXT® via SINIT code execution hijacking. 2011

So Far ……

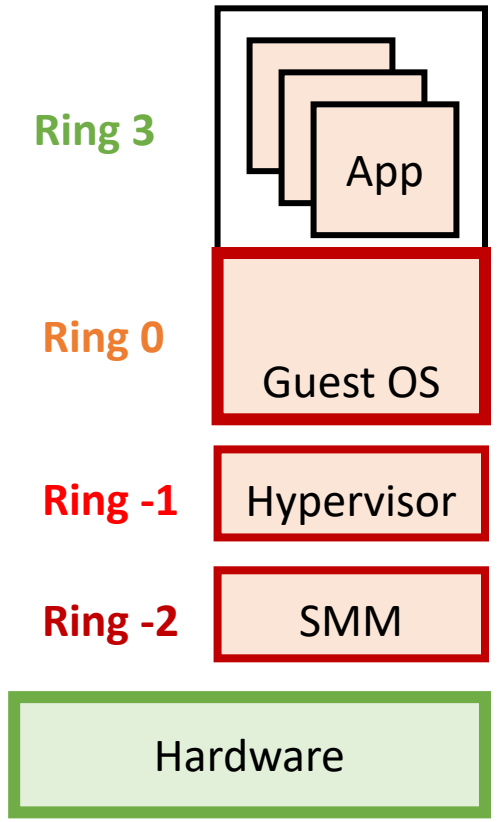

Trusted
Ring 3
App
Ring 0
Host OS
SMM
Hardware
Ring 3
App
App
Ring 0
Guest OS
Guest OS
Ring -1
Hypervisor
Ring -2
SMM (firmware)
Hardware

So Far ……


The trend: shrink TCB.
Why?
Trusted
Ring 3
App
Ring 0
Host OS
SMM
Hardware
Ring 3
App
App
Ring 0
Guest OS
Guest OS
Ring -1
Hypervisor
Ring -2
SMM (firmware)
Hardware

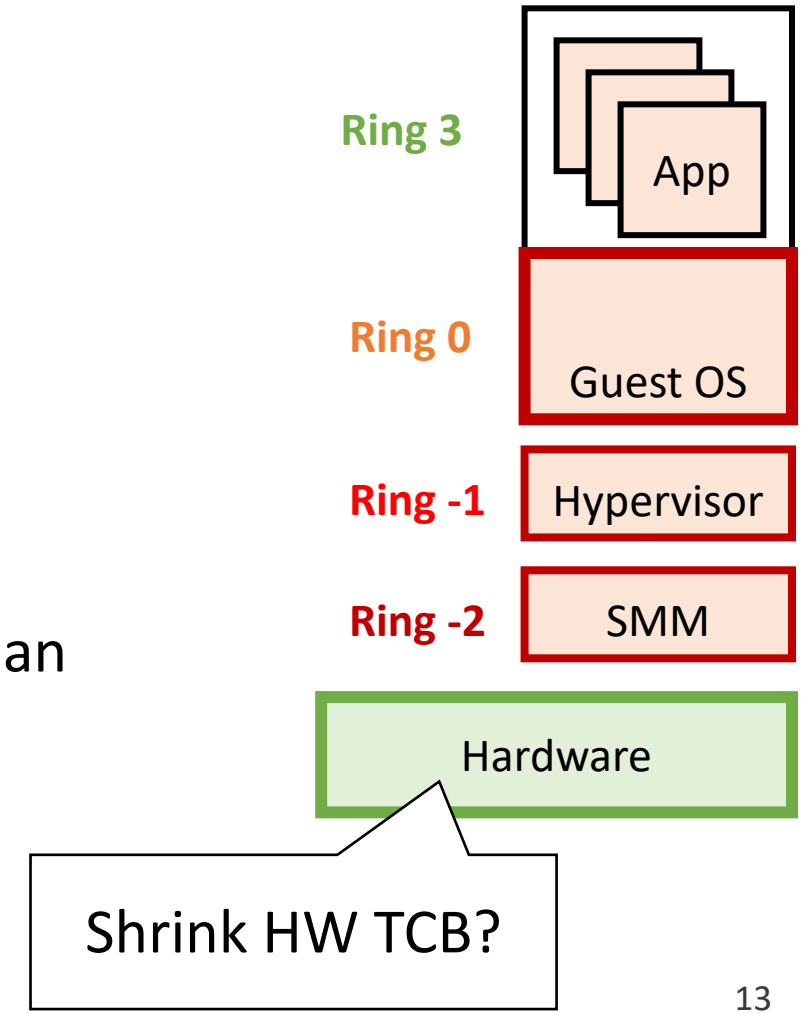
Why Shrink TCB?

- Software bugs
  - SMM-based rootkits
  - Xen 150K LOC, 40+ vulnerabilities per year
  - Monolithic kernel, e.g., Linux, 17M LOC, 100+ vulnerabilities per year
- Remote Computing
  - Remote computer and software stack owned by an untrusted party
  - Examples



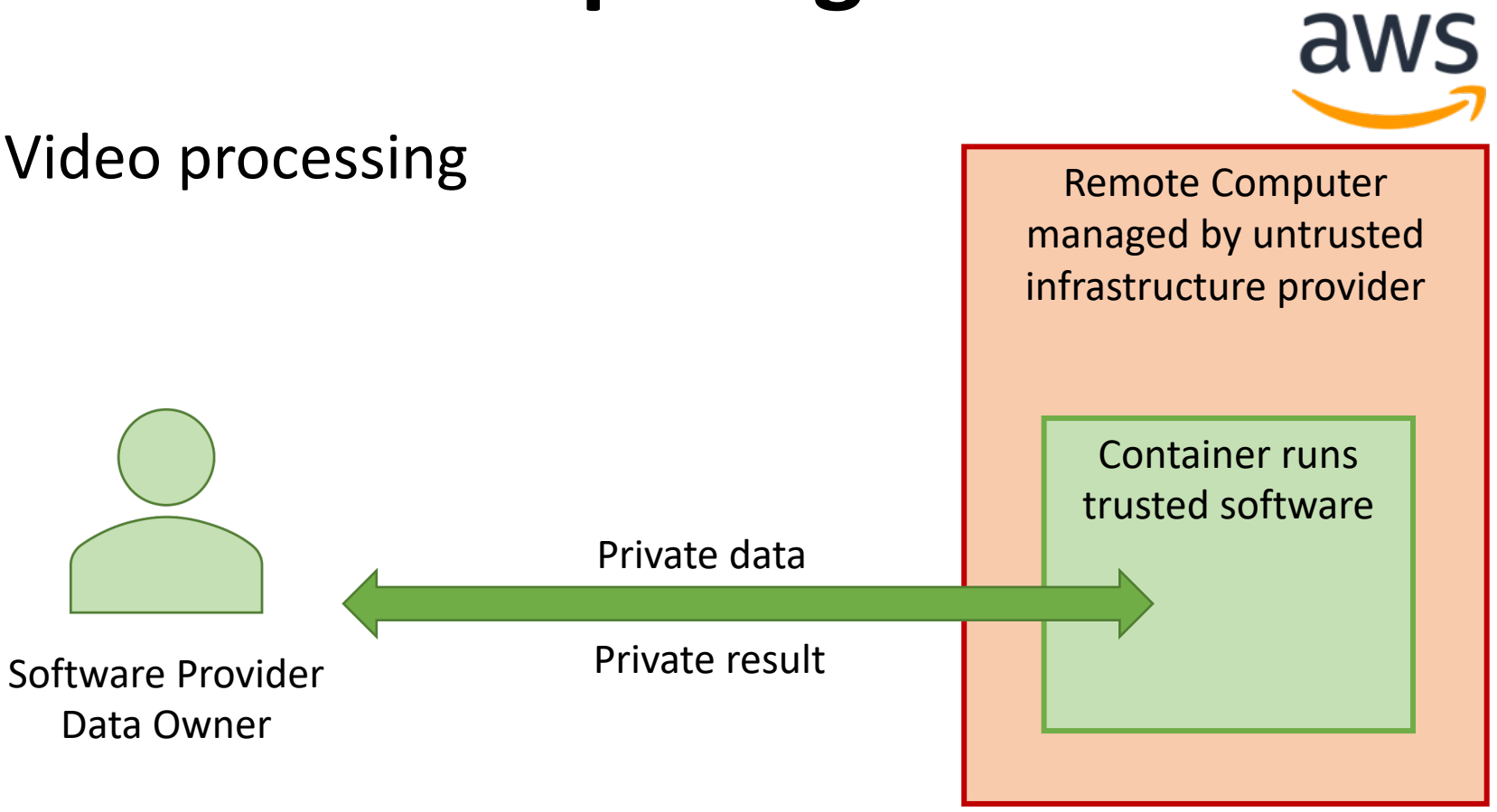
Why Shrink TCB?

- Software bugs
  - SMM-based rootkits
  - Xen 150K LOC, 40+ vulnerabilities per year
  - Monolithic kernel, e.g., Linux, 17M LOC, 100+ vulnerabilities per year
- Remote Computing
  - Remote computer and software stack owned by an untrusted party
  - Examples

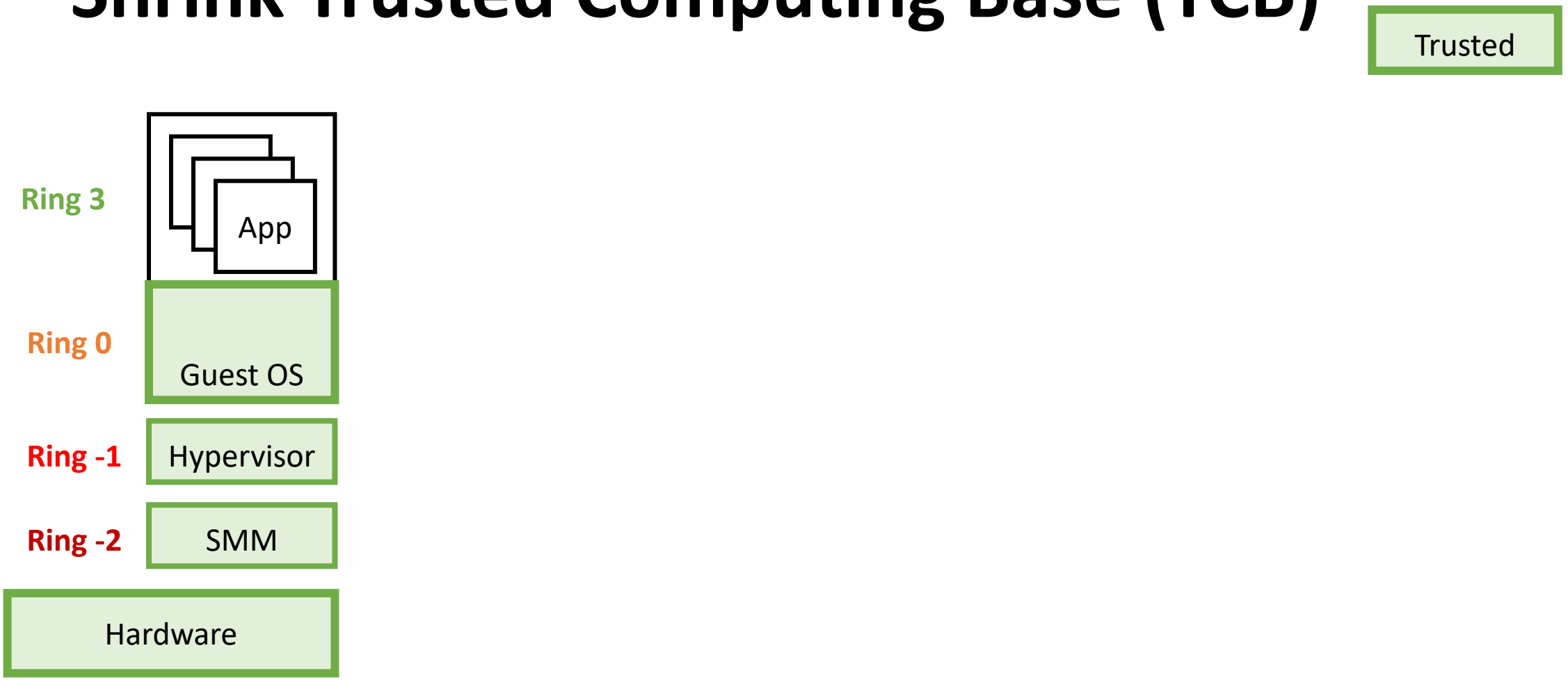


Secure Remote Computing

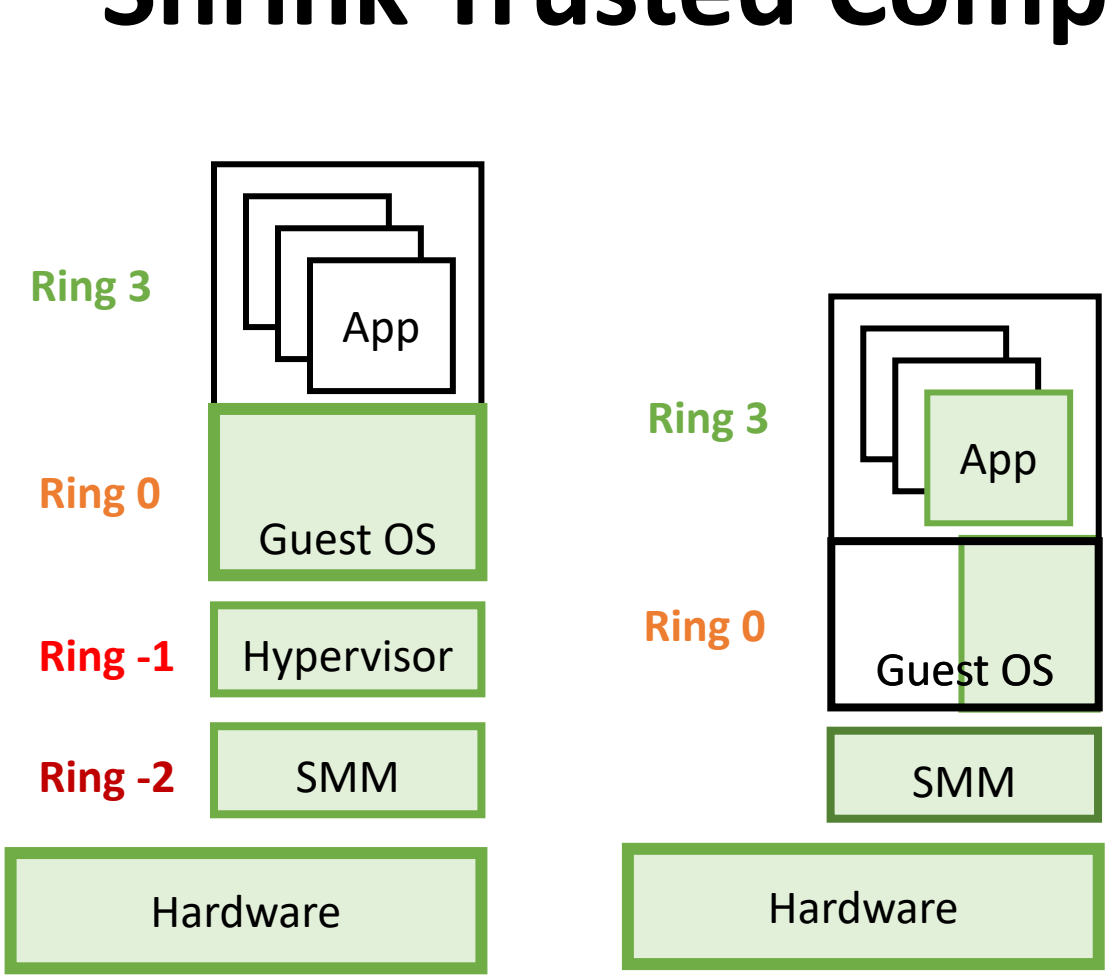
- Example: Video processing



Shrink Trusted Computing Base (TCB)

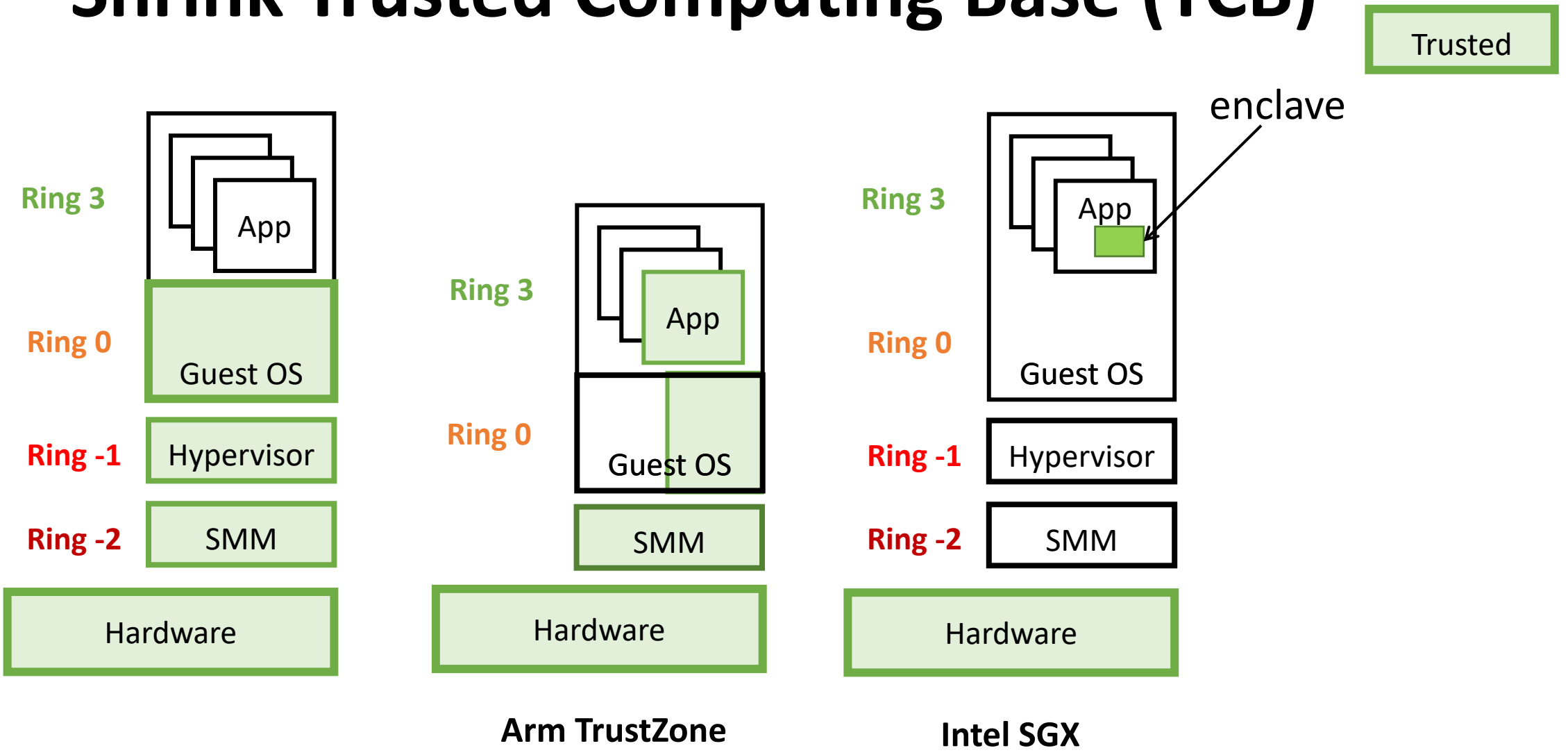


Shrink Trusted Computing Base (TCB)



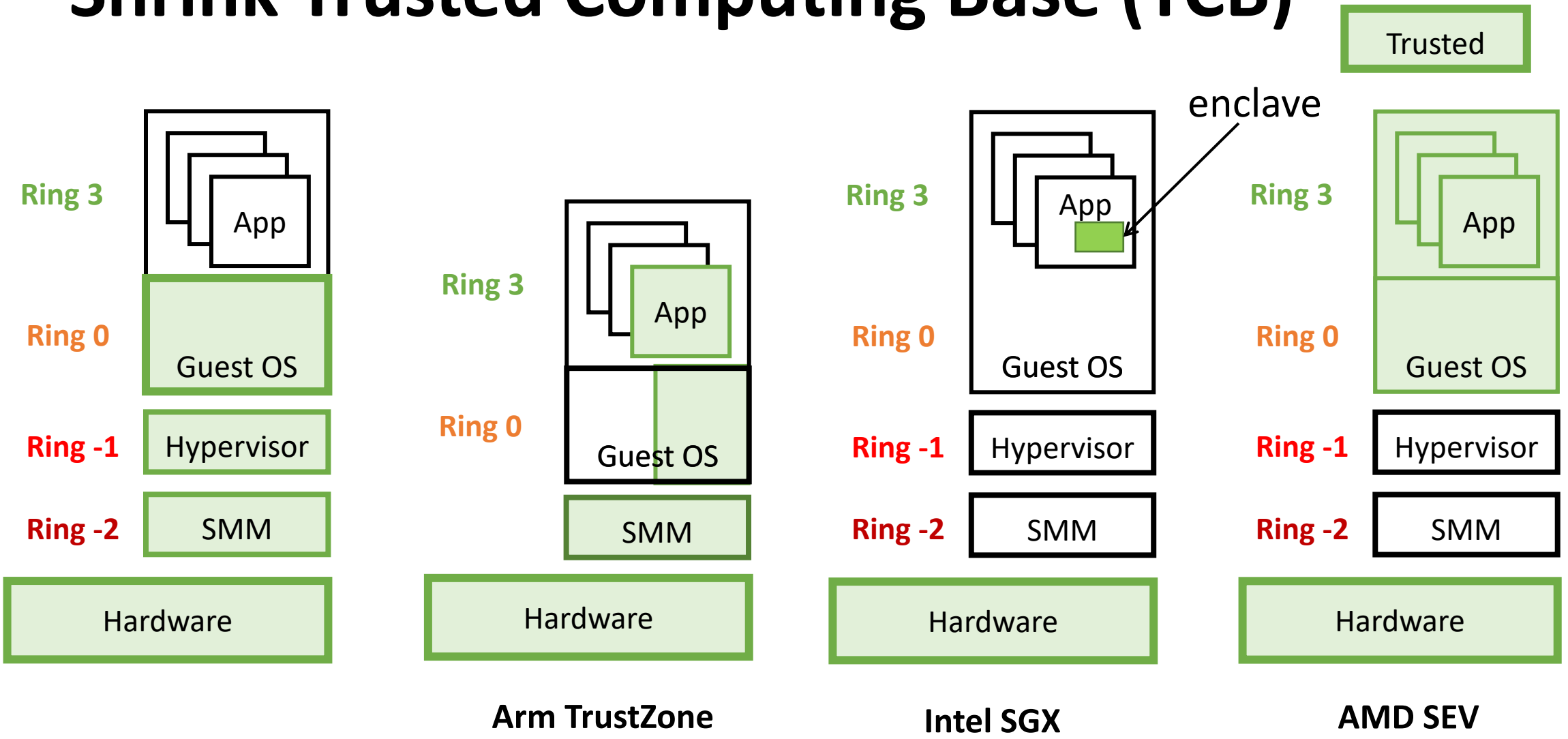
Arm TrustZone

Shrink Trusted Computing Base (TCB)

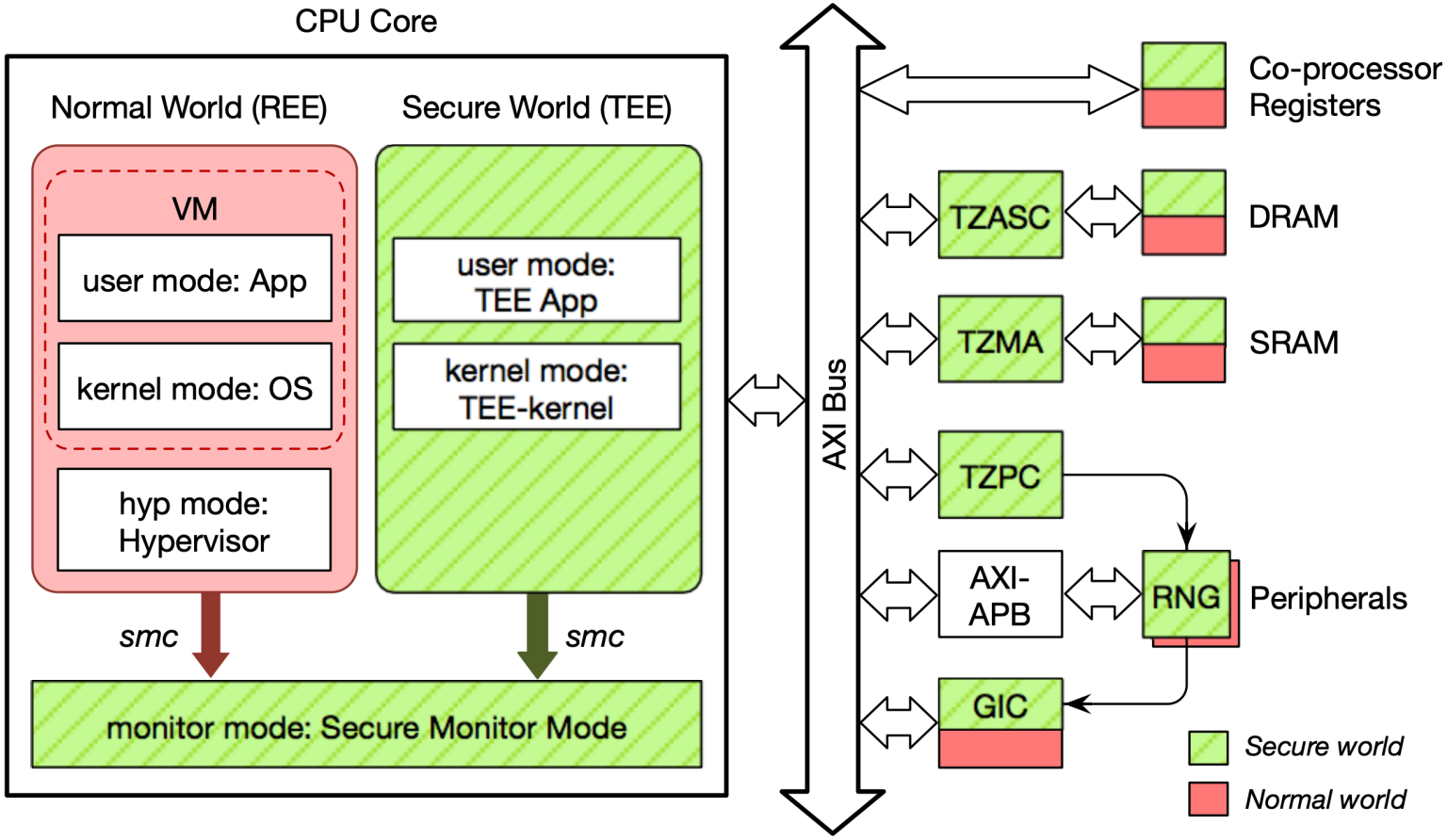


Arm TrustZone **Intel SGX**

Shrink Trusted Computing Base (TCB)

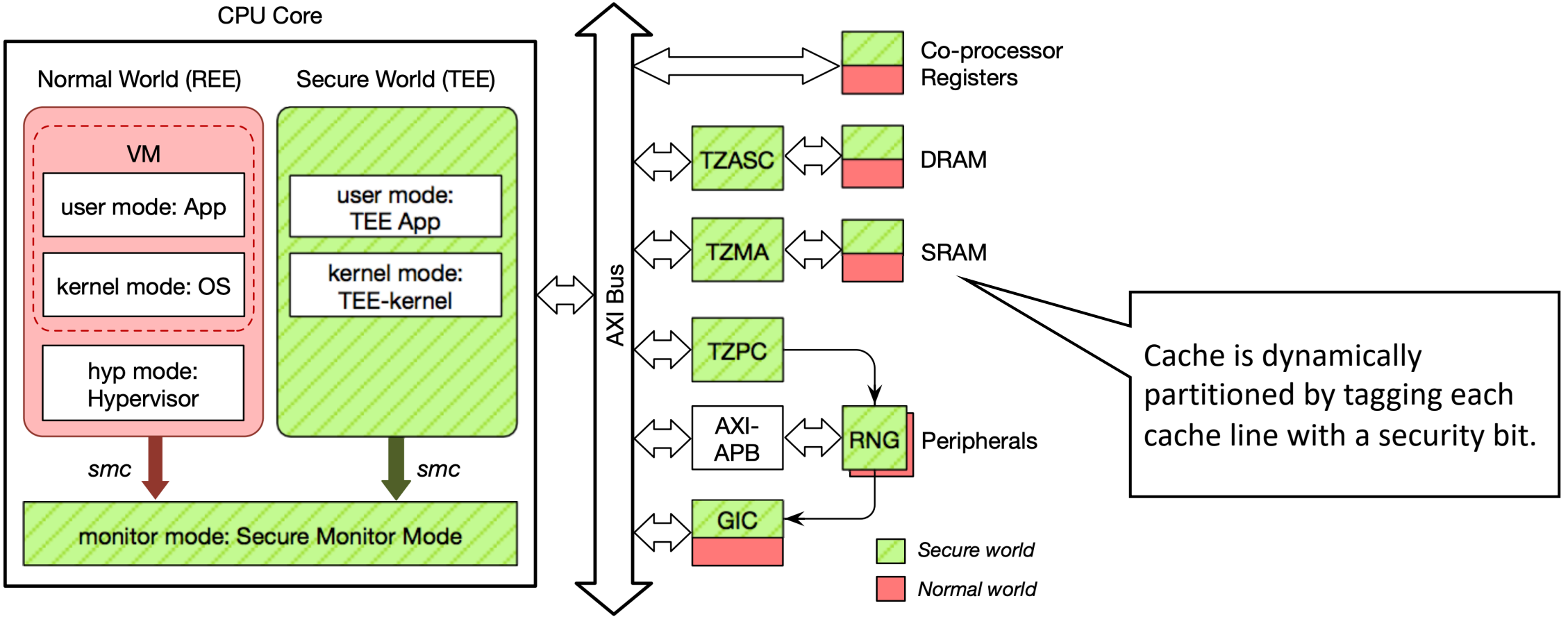


Arm TrustZone



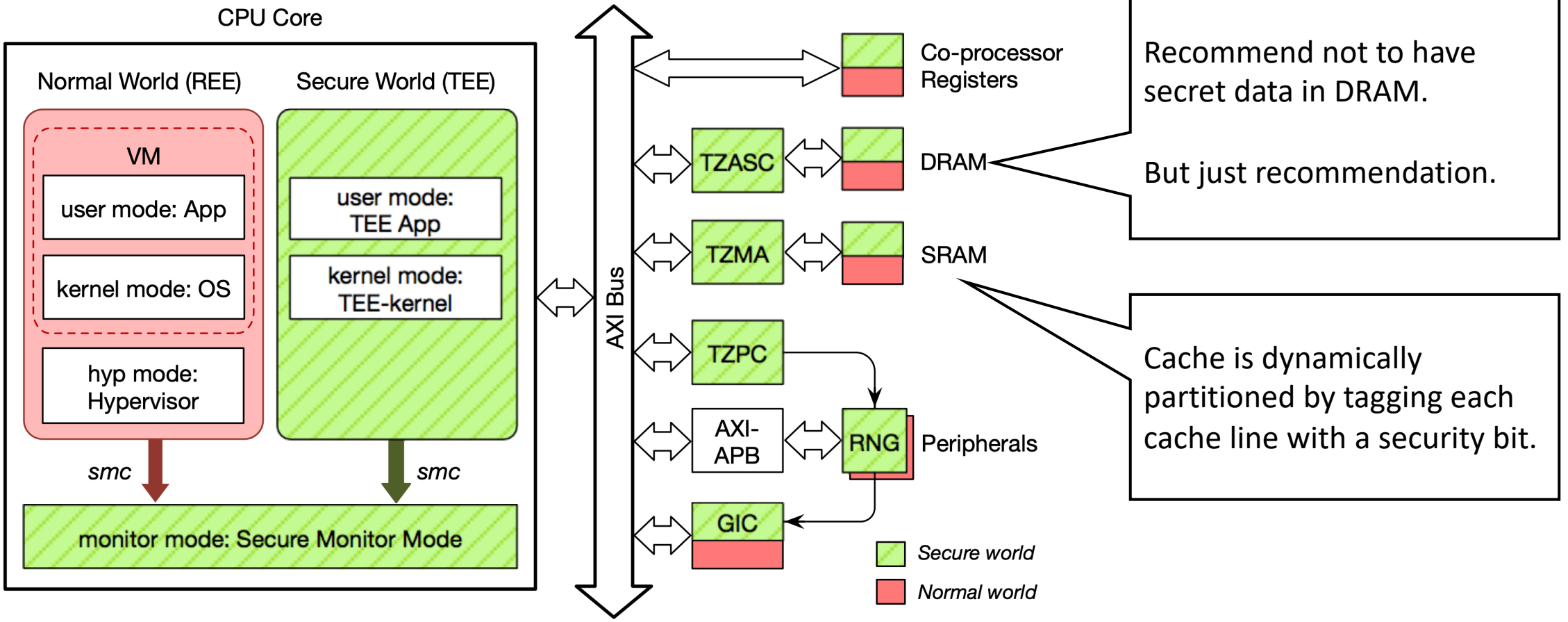
from Hua et al. vTZ: Virtualizing ARM TrustZone. Usenix'17

Arm TrustZone



from Hua et al. vTZ: Virtualizing ARM TrustZone. Usenix'17

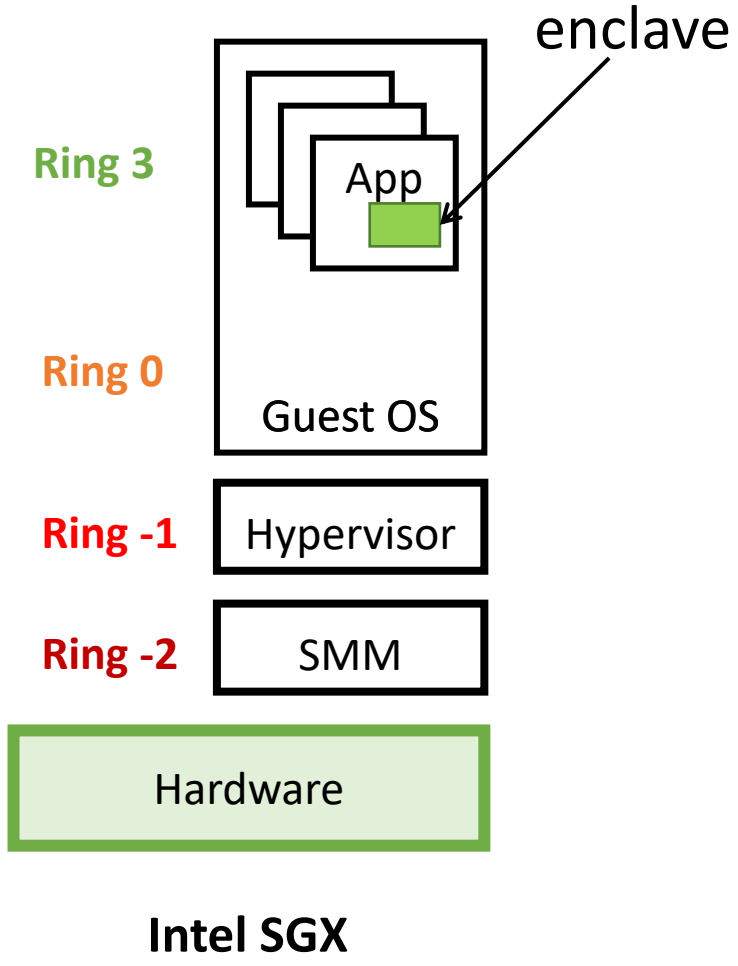
Arm TrustZone



from Hua et al. vTZ: Virtualizing ARM TrustZone. Usenix'17

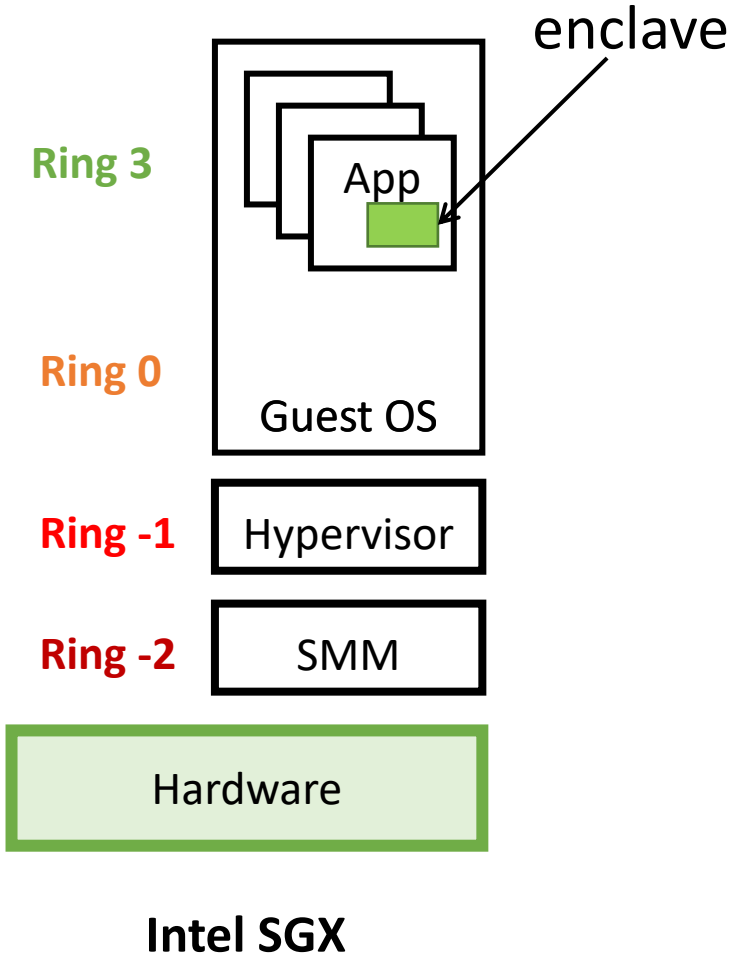
Privileged Software Attacks

- Manipulate everything



Privileged Software Attacks

- Manipulate everything
- Directly see and modify application code and data



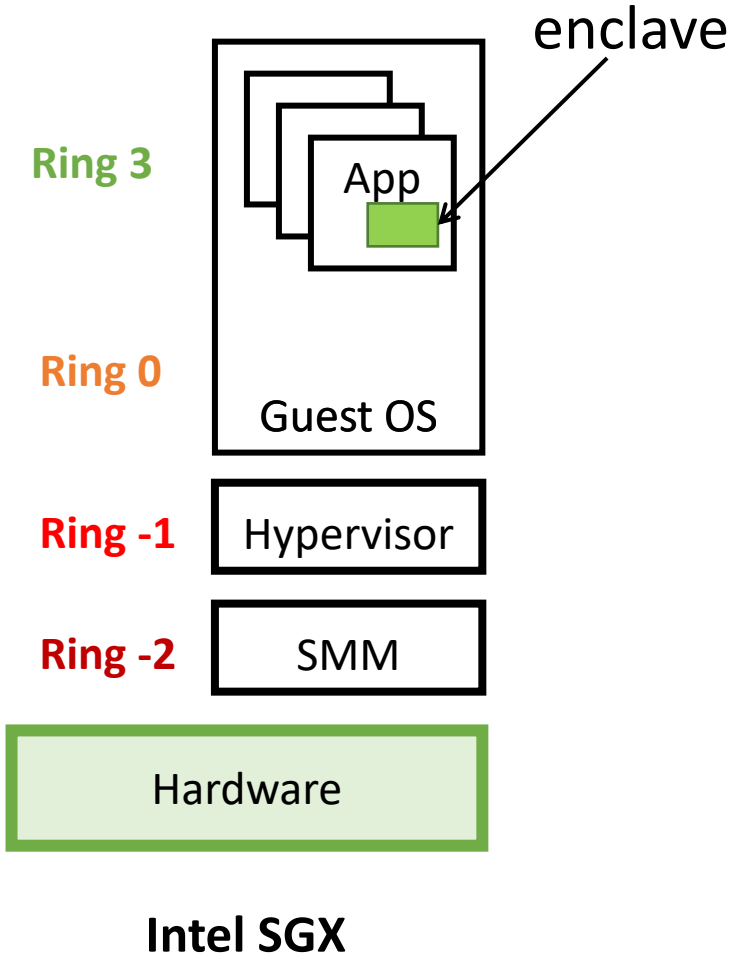
Privileged Software Attacks

- Manipulate everything

- Directly see and modify application code and data

  → Need to encrypt secret data

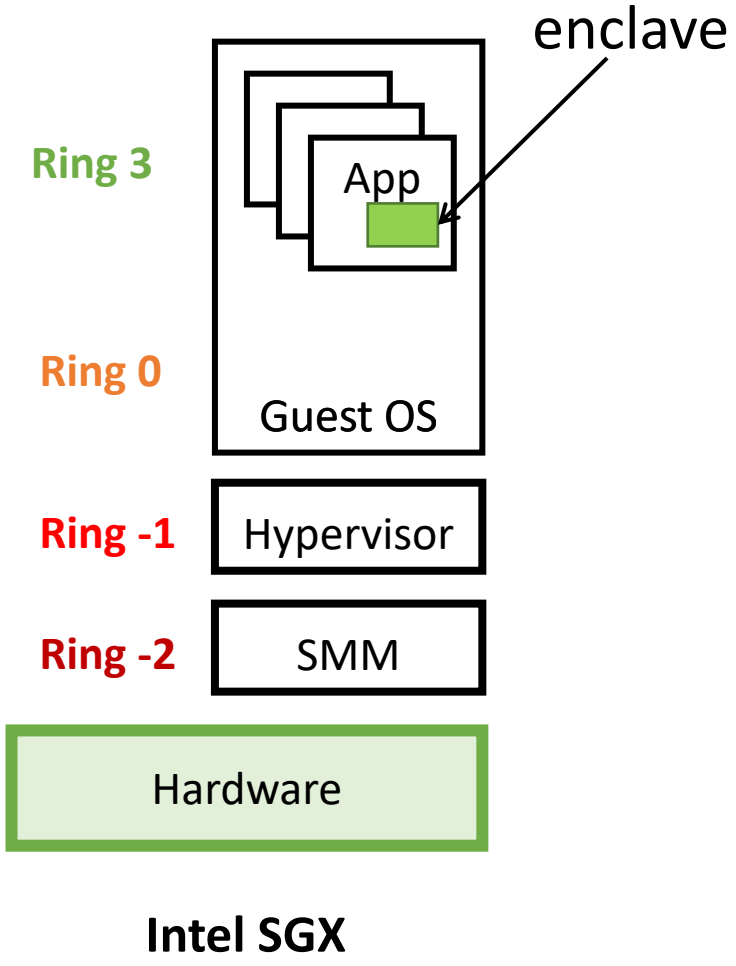
  → Need to verify integrity (software attestation)



Intel SGX

Privileged Software Attacks

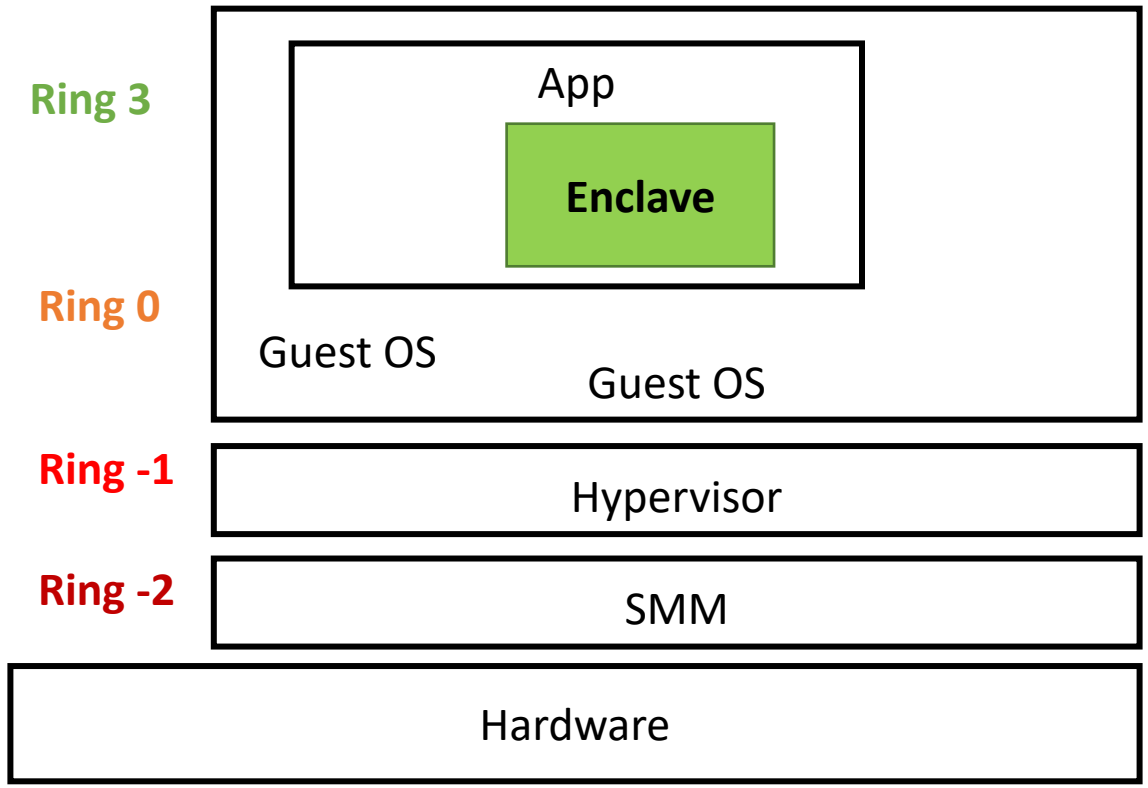
- Manipulate everything
- Directly see and modify application code and data
  - → Need to encrypt secret data
  - → Need to verify integrity (software attestation)
- Mess up with
  - Address translation
  - Process initialization and context switch
  - Interrupts, I/Os
  - etc.



Intel SGX

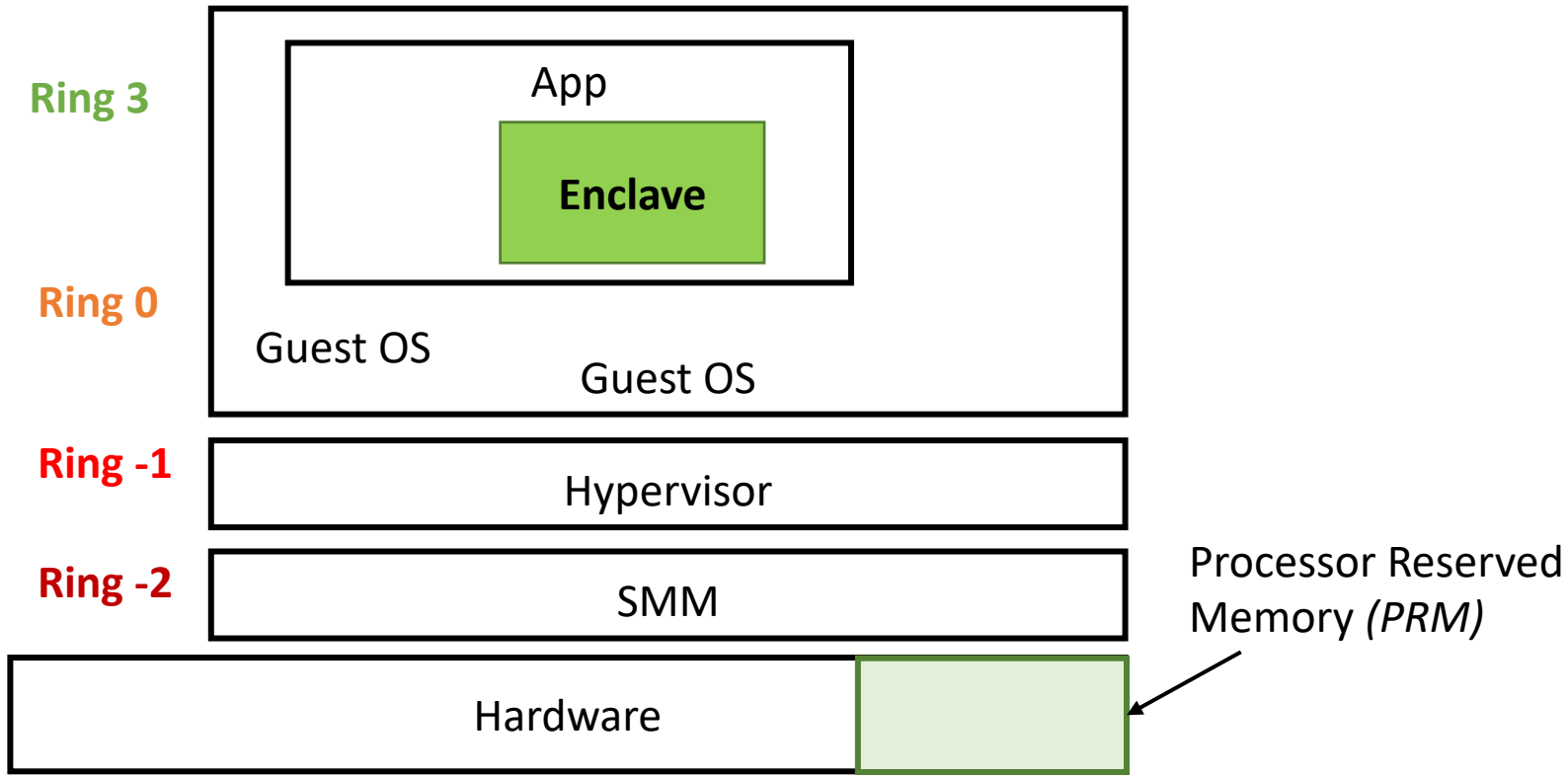
Enclave High-level View

- Goal: A protected environment that contains the code and data of a security-sensitive computation.



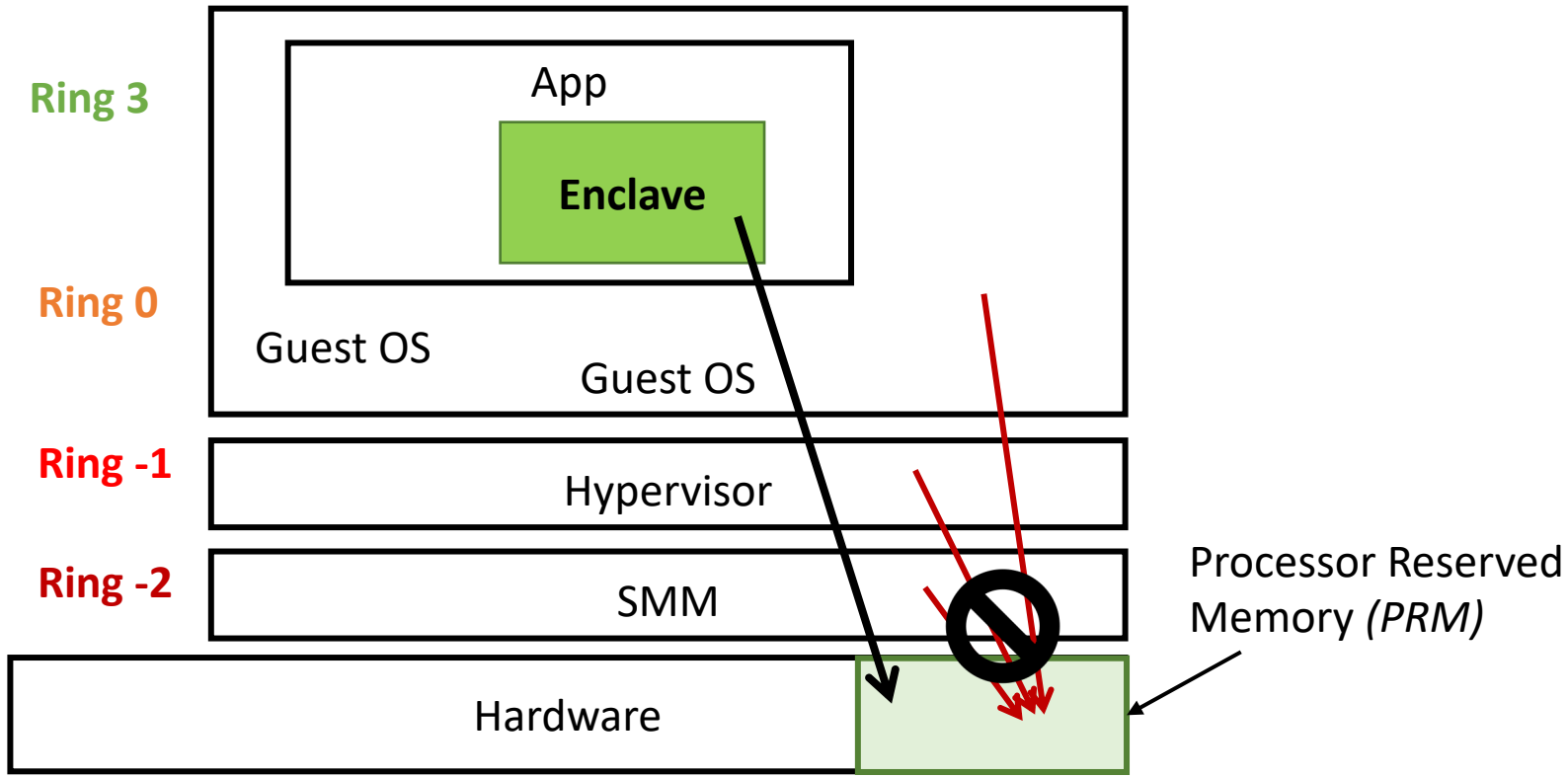
Enclave High-level View

- Goal: A protected environment that contains the code and data of a security-sensitive computation.



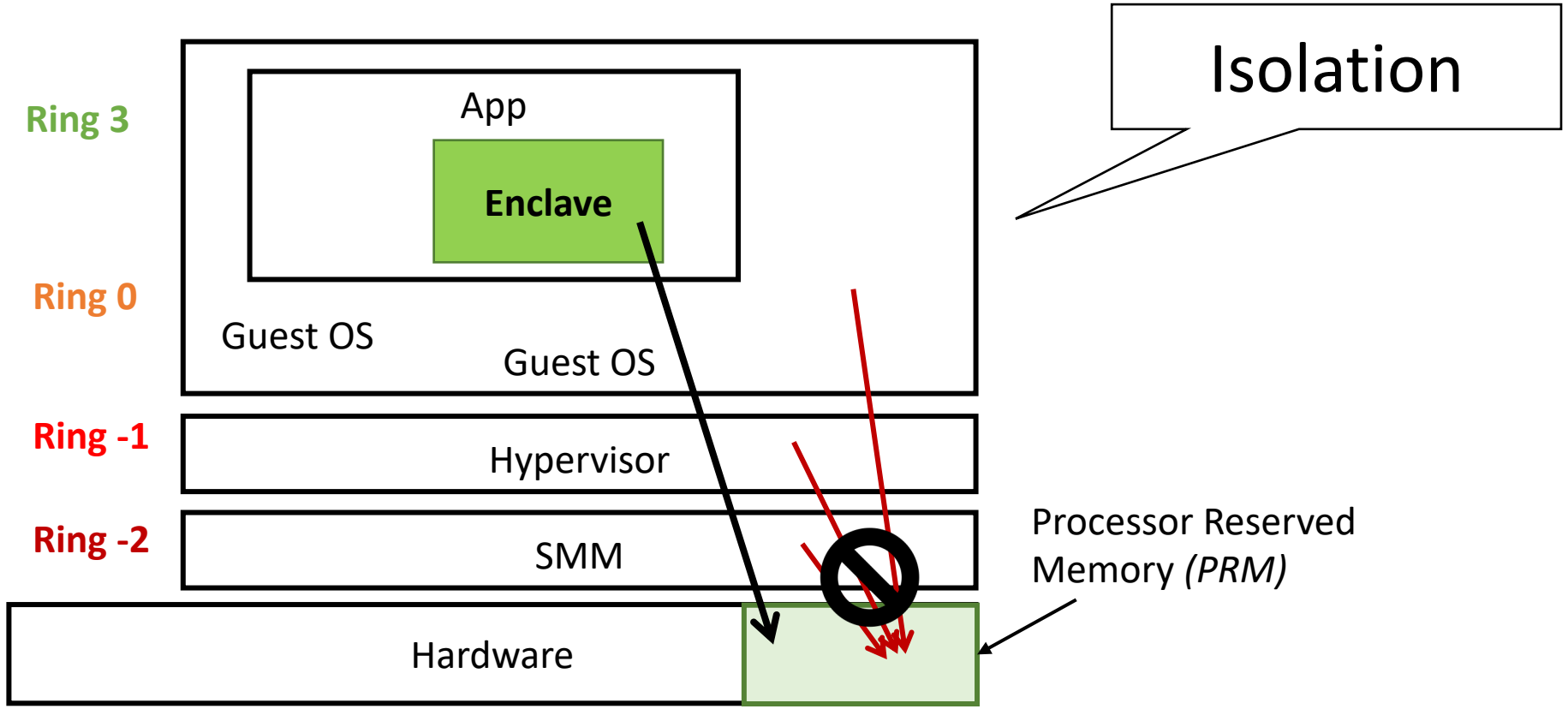
Enclave High-level View

- Goal: A protected environment that contains the code and data of a security-sensitive computation.



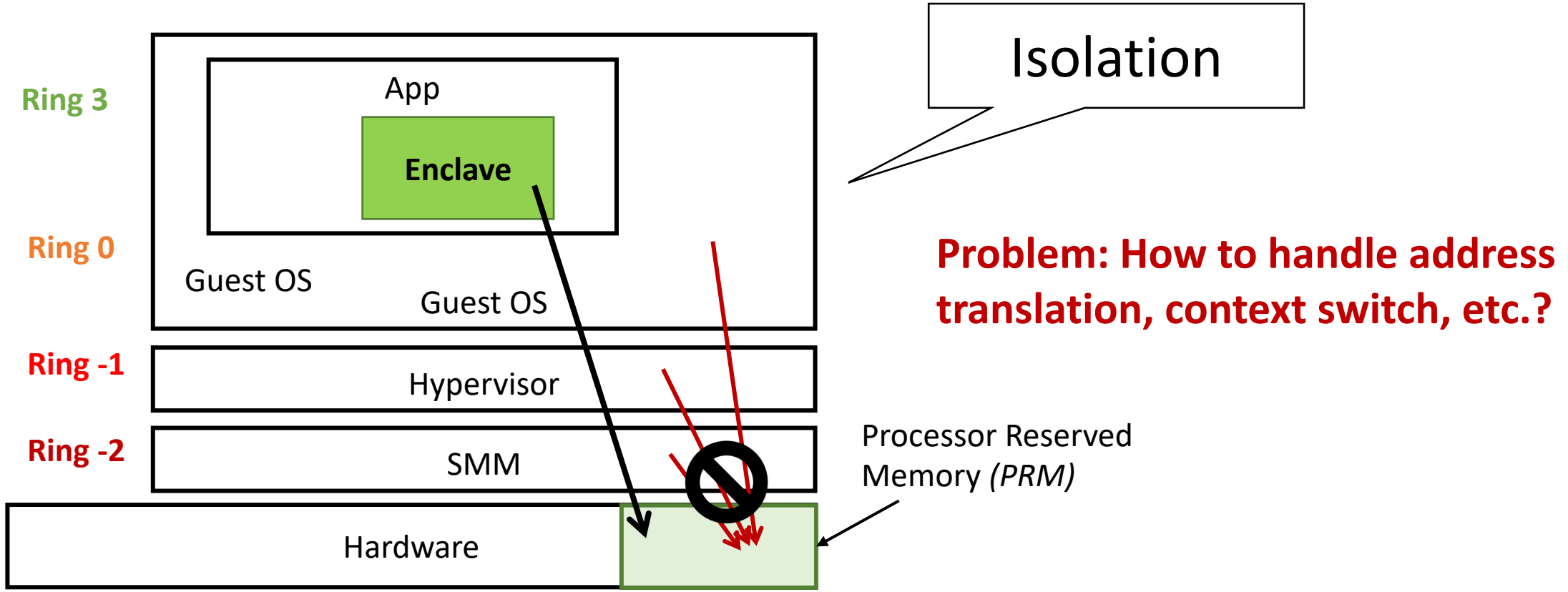
Enclave High-level View

- Goal: A protected environment that contains the code and data of a security-sensitive computation.

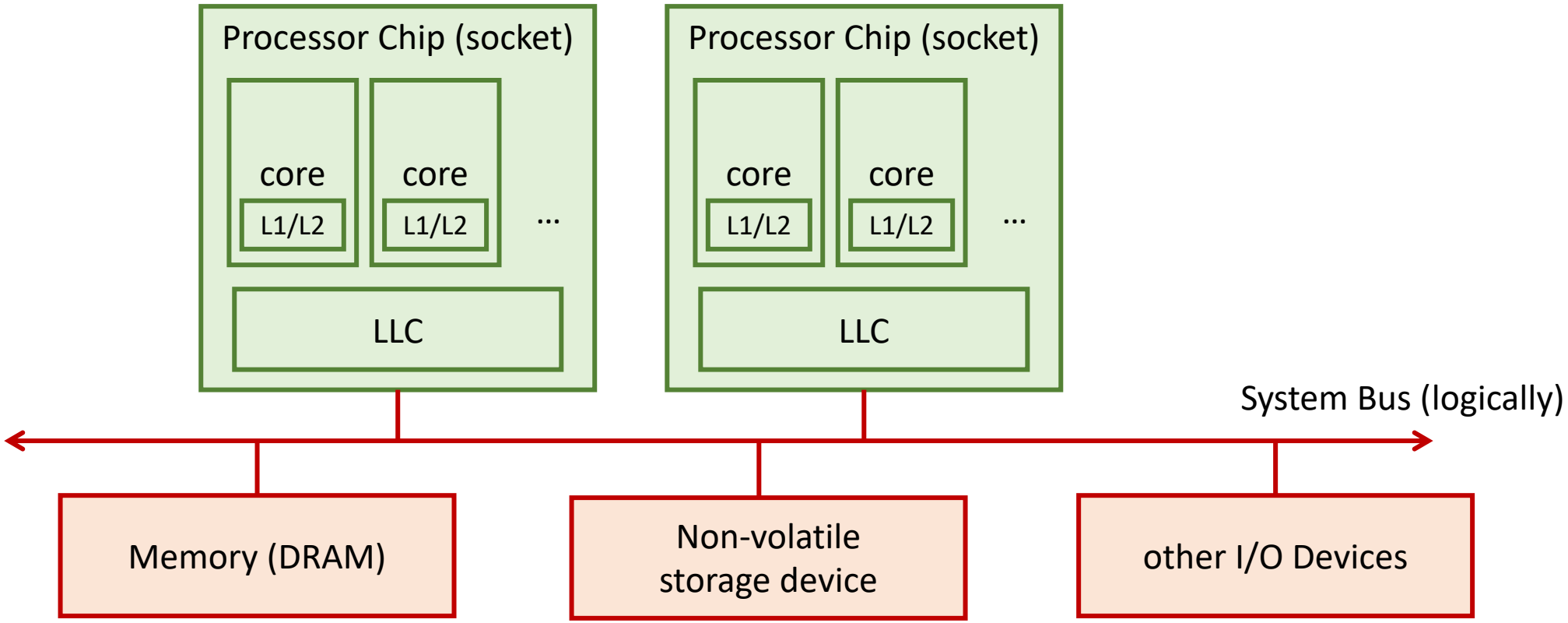


Enclave High-level View

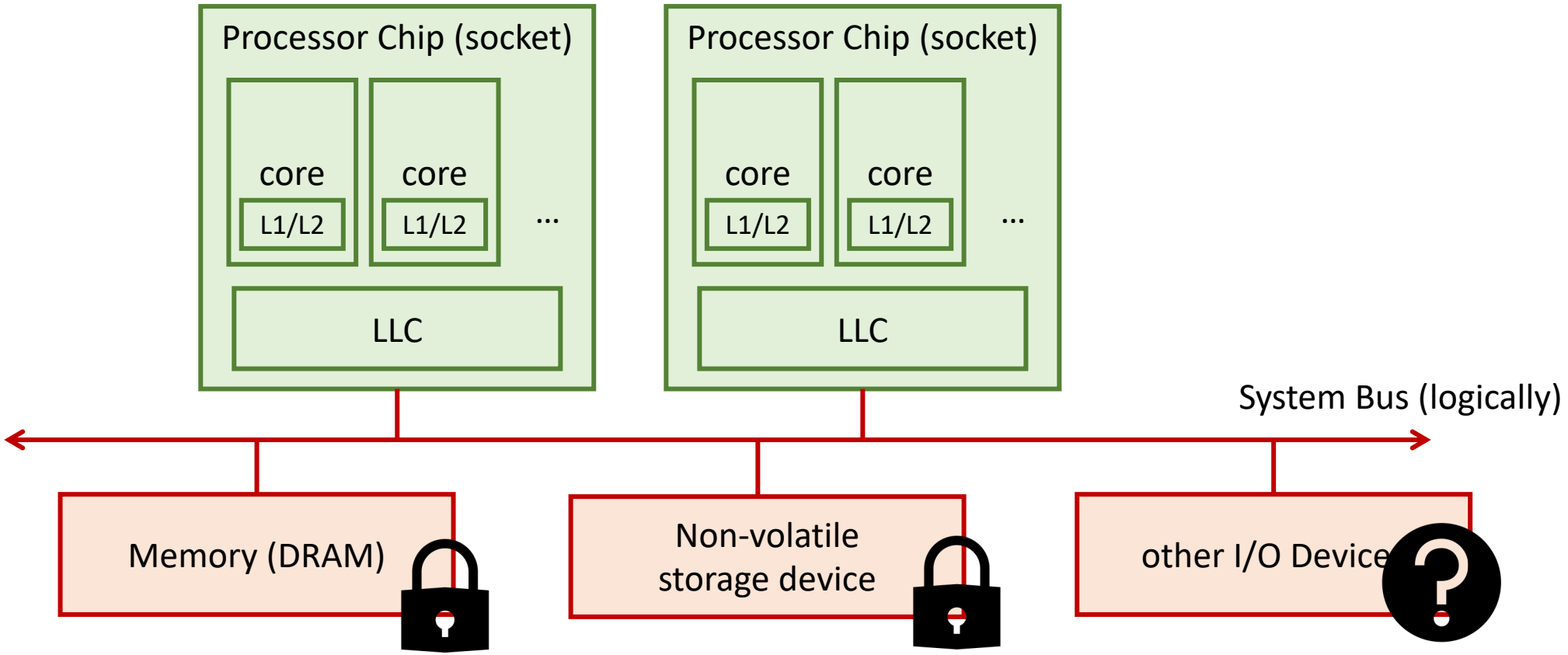
- Goal: A protected environment that contains the code and data of a security-sensitive computation.



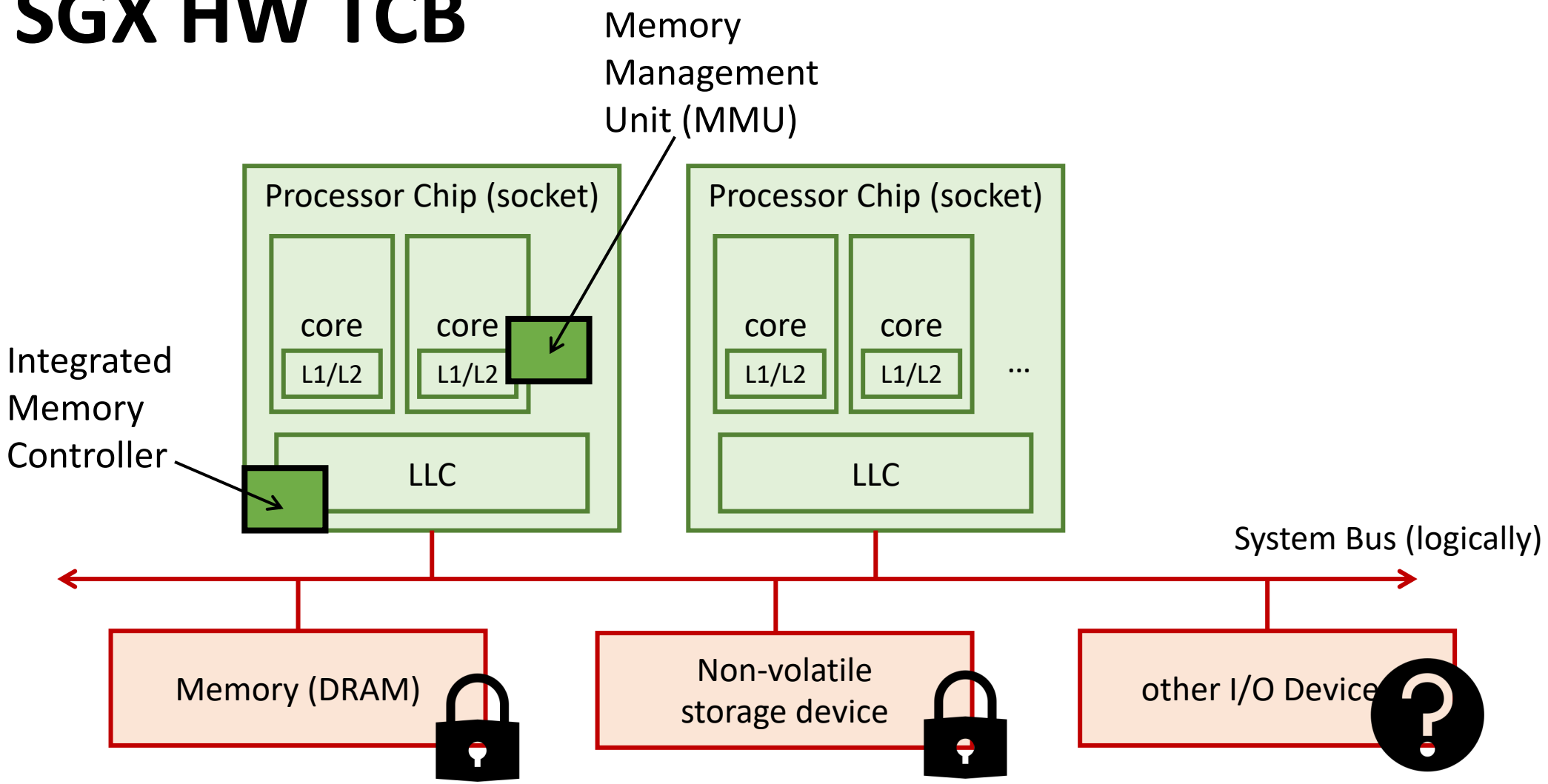
SGX HW TCB


Processor Chip (socket)
core
L1/L2
core
L1/L2
...
LLC
Processor Chip (socket)
core
L1/L2
core
L1/L2
...
LLC
System Bus (logically)
Memory (DRAM)
Non-volatile storage device
other I/O Devices

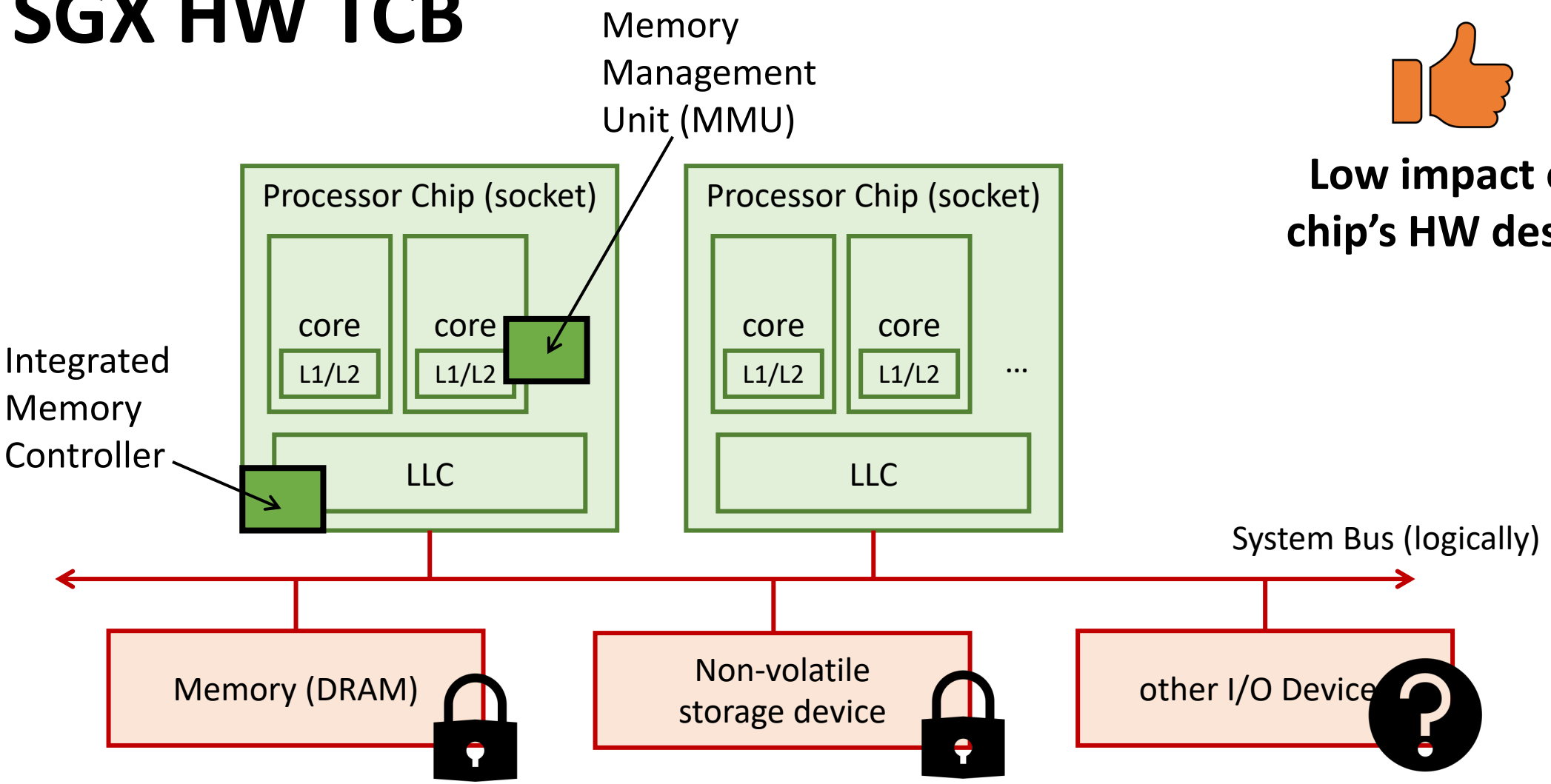
SGX HW TCB


Processor Chip (socket)
core
L1/L2
core
L1/L2
...
LLC
Processor Chip (socket)
core
L1/L2
core
L1/L2
...
LLC
System Bus (logically)
Memory (DRAM)
Non-volatile
storage device
other I/O Device
?

SGX HW TCB


Memory
Management
Unit (MMU)
Processor Chip (socket)
core
L1/L2
core
L1/L2
LLC
Processor Chip (socket)
core
L1/L2
core
L1/L2
...
LLC
Integrated
Memory
Controller
System Bus (logically)
Memory (DRAM)
Non-volatile
storage device
other I/O Device
?

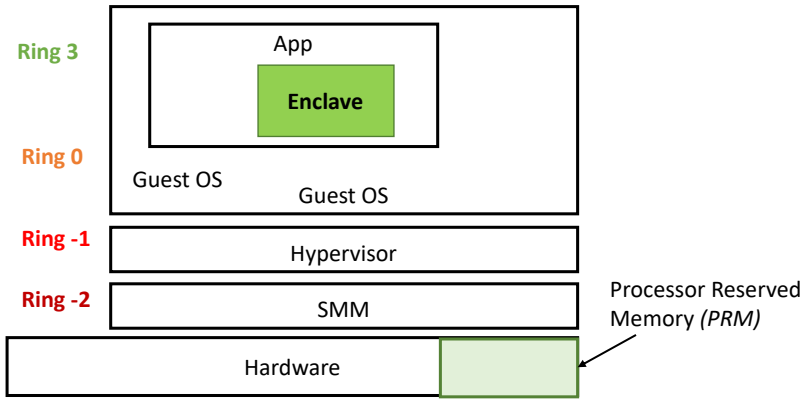
SGX HW TCB



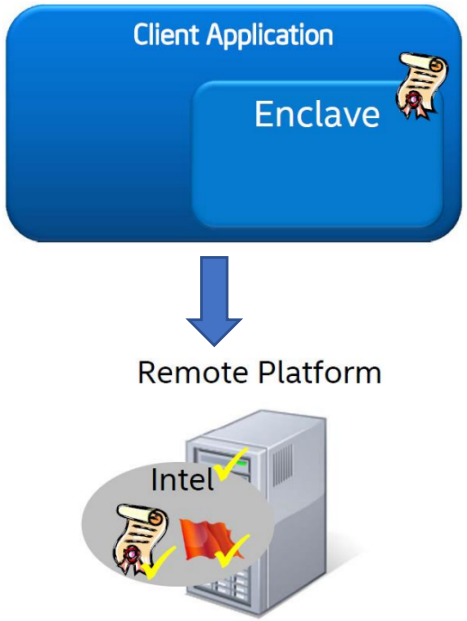
Low impact on chip's HW design

Intel SGX Security Mechanisms

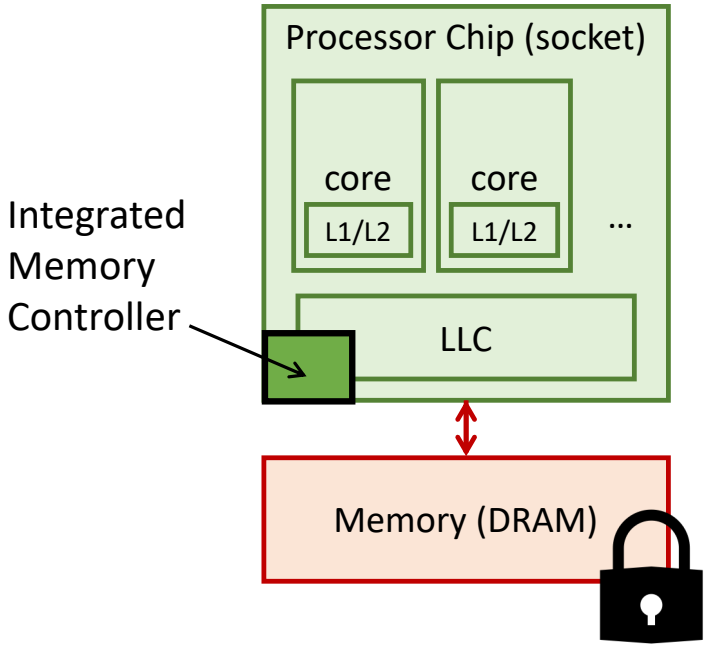
Isolation



Attestation

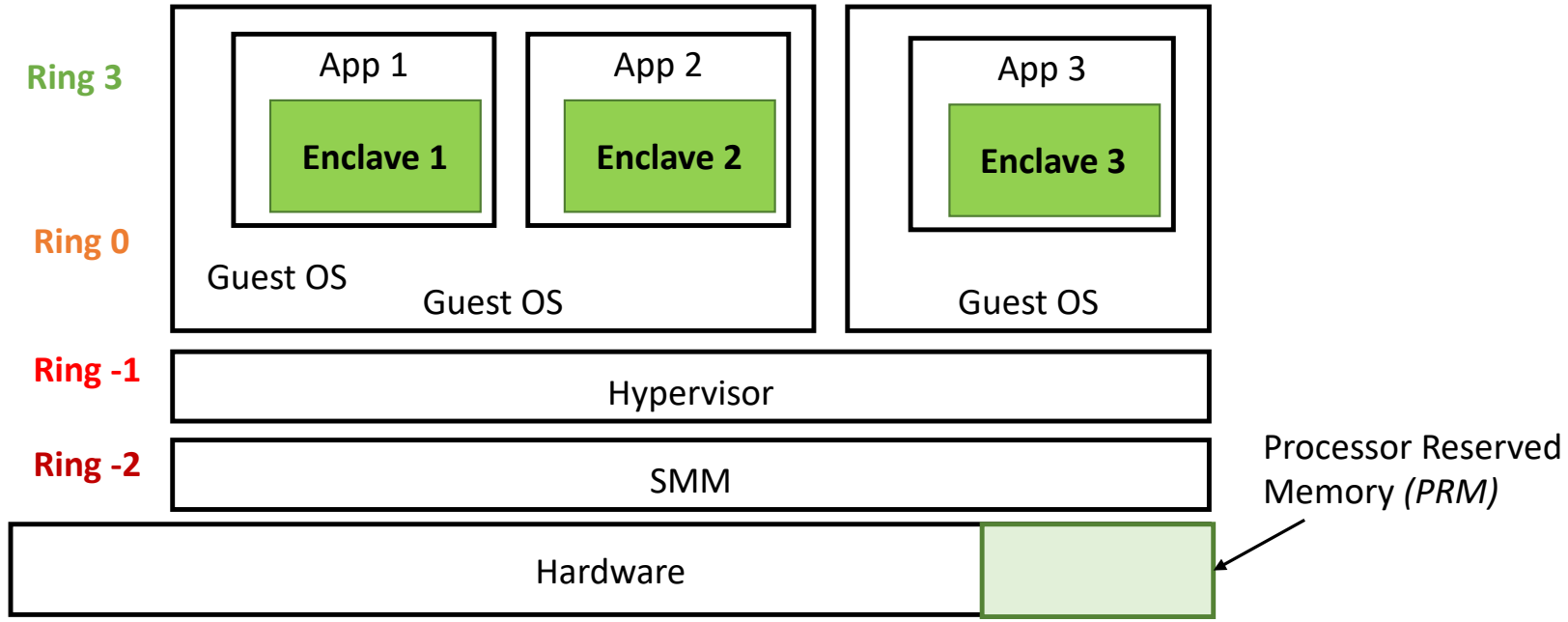


DRAM Protection



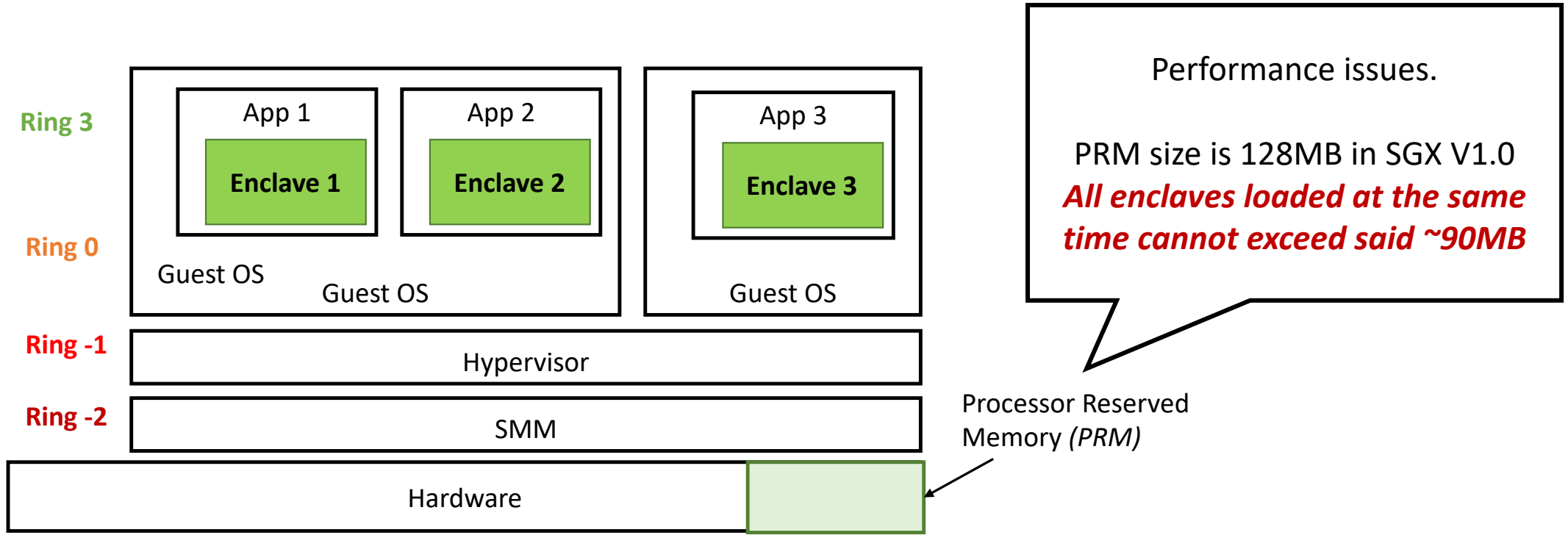
SGX Access Control

- Assume software attestation is done
- Can have multiple enclaves

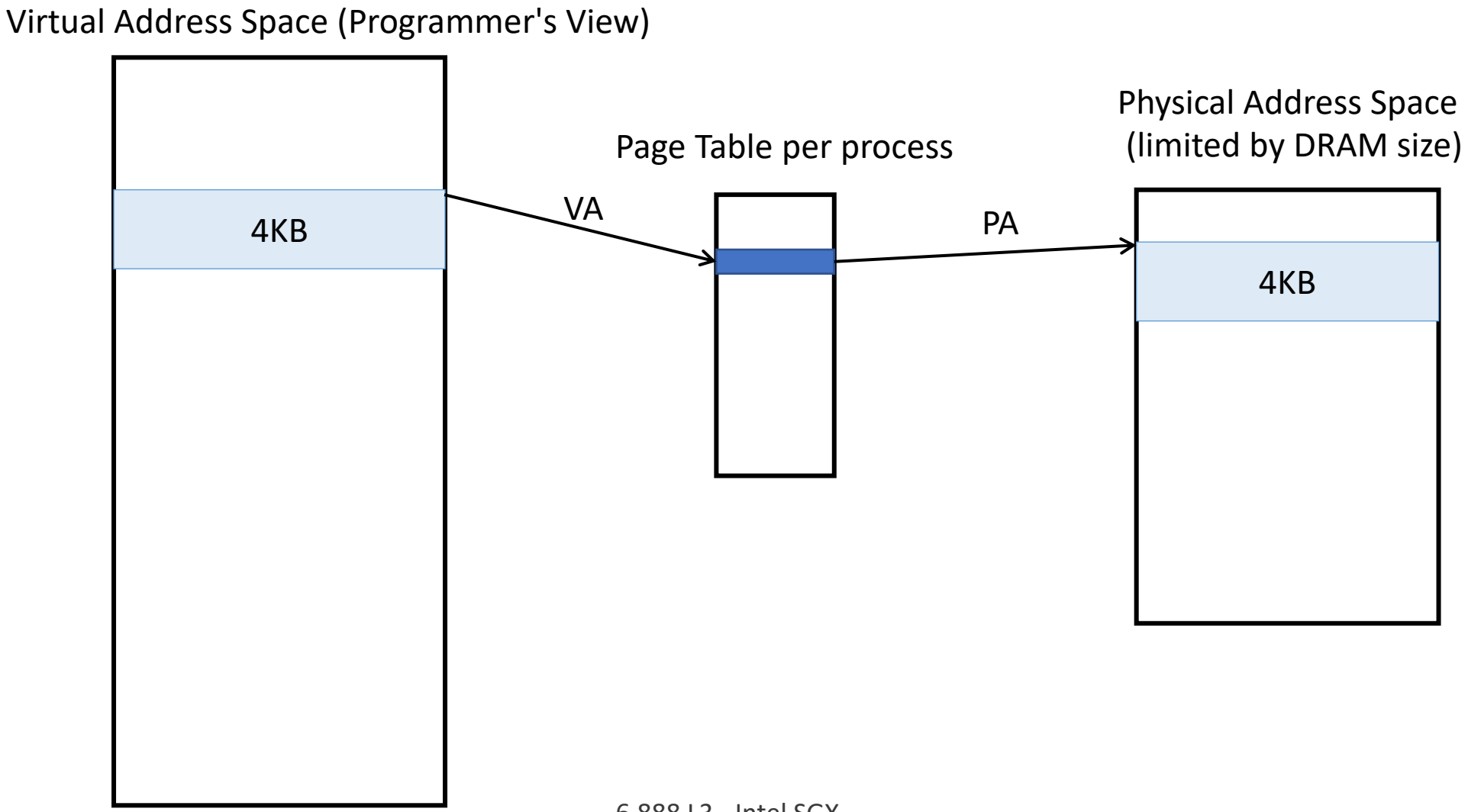


SGX Access Control

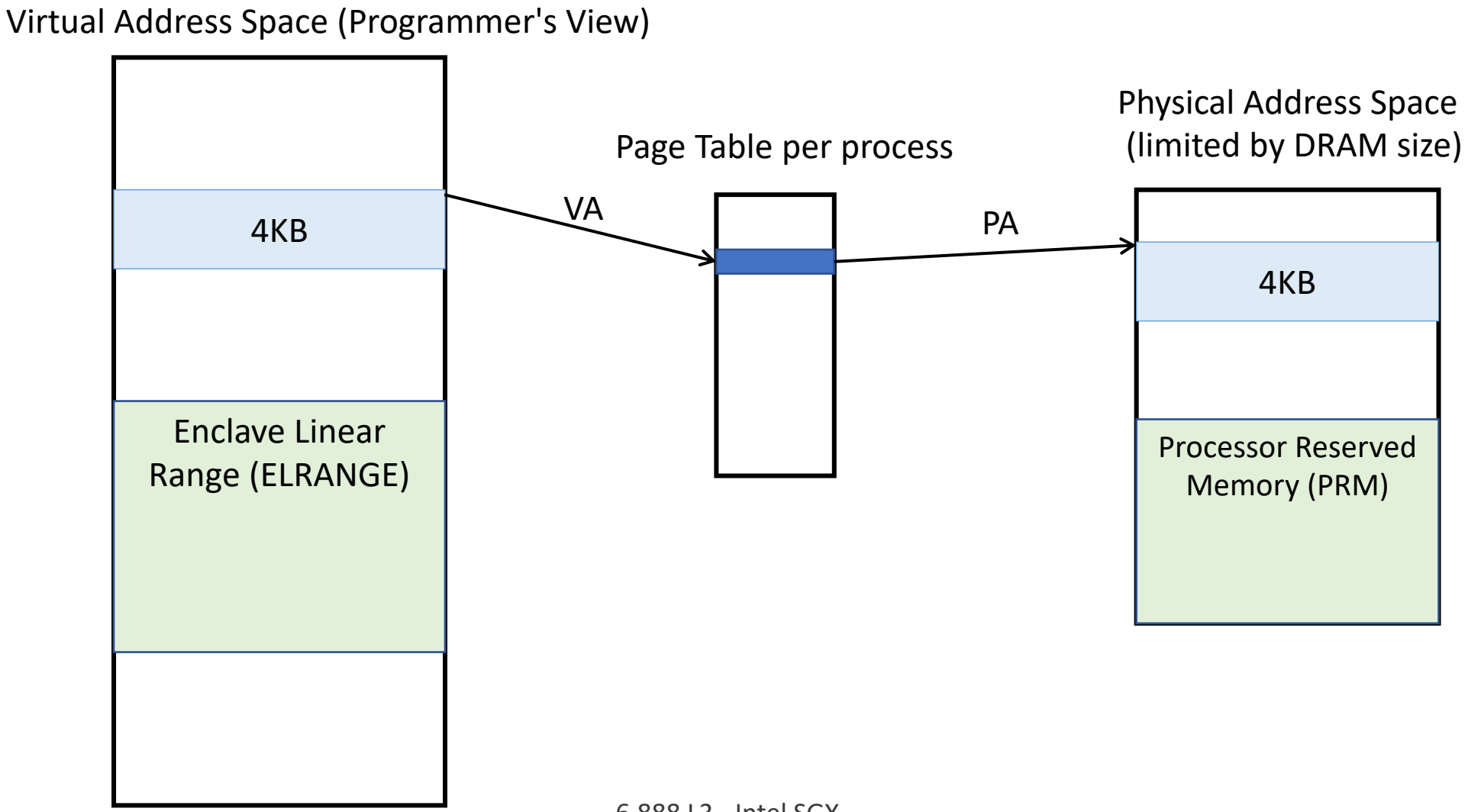
- Assume software attestation is done
- Can have multiple enclaves



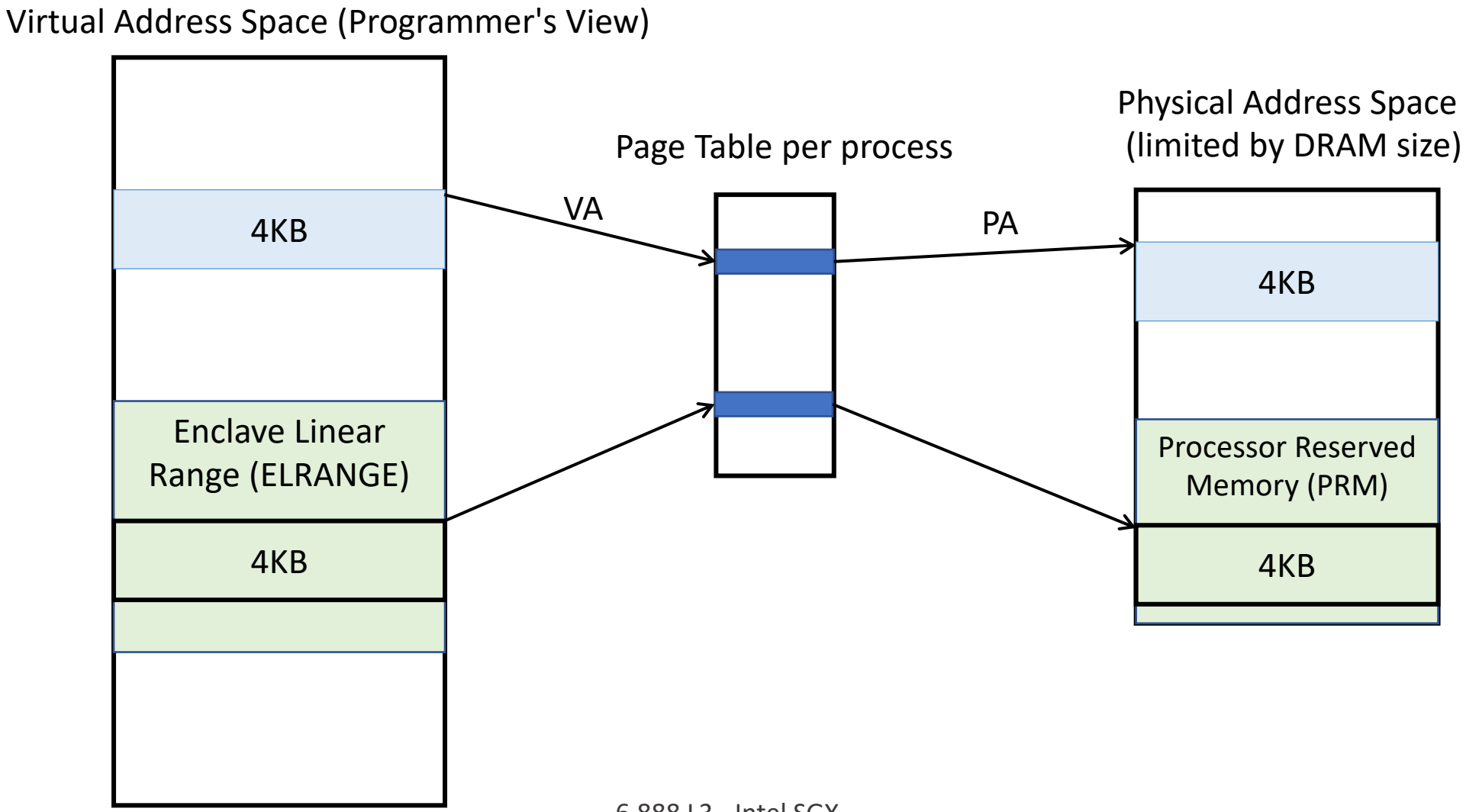
Enclave Address Translation



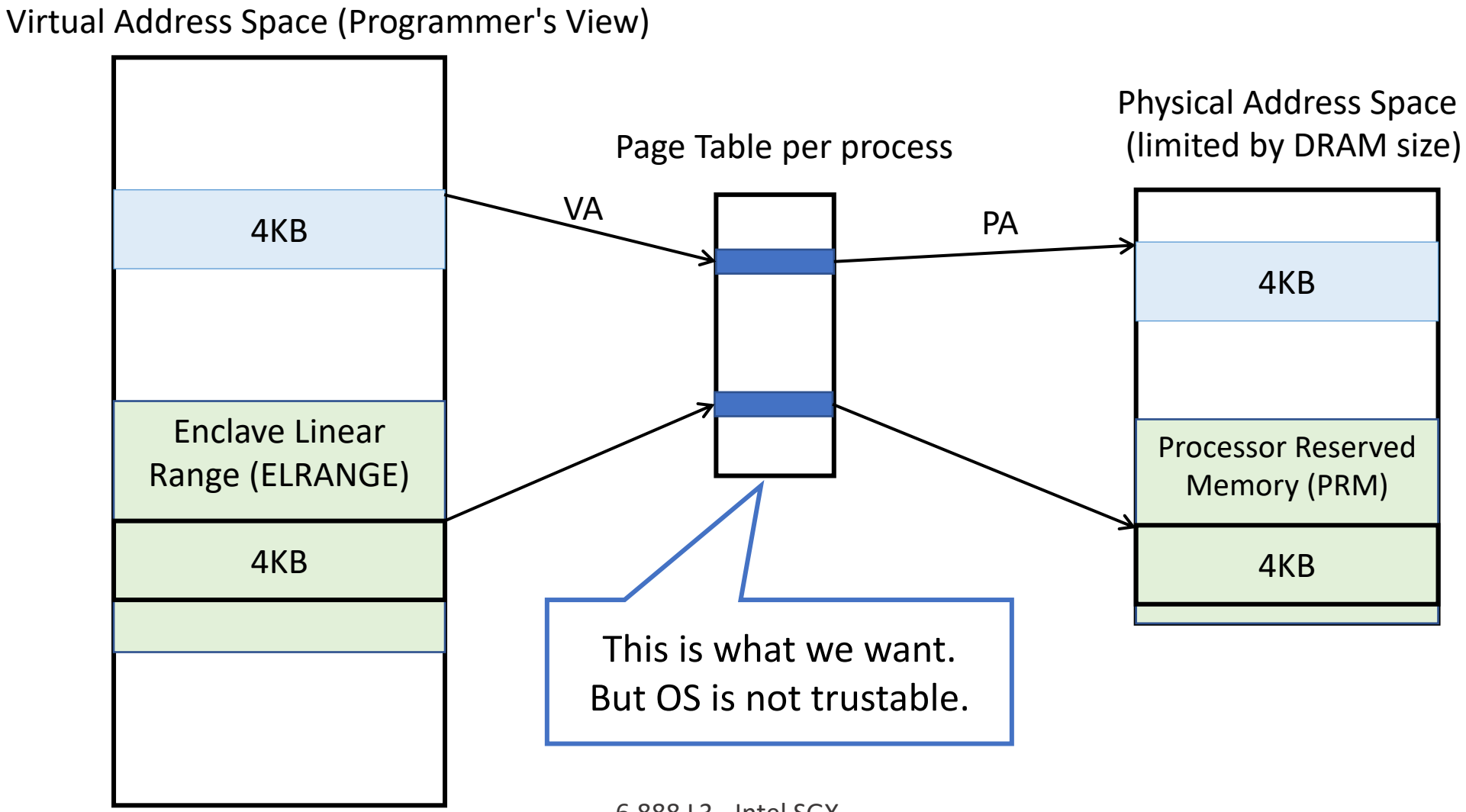
Enclave Address Translation



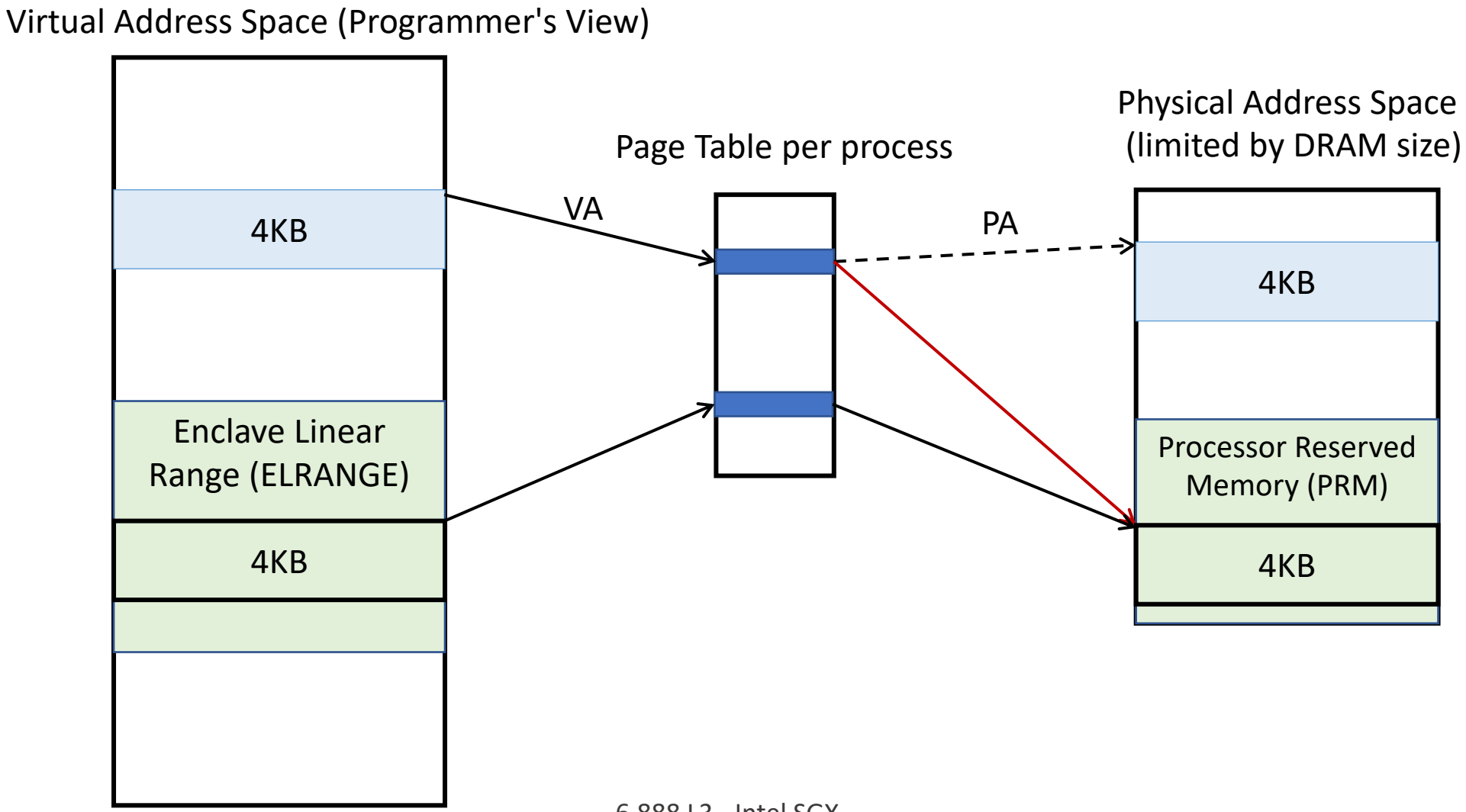
Enclave Address Translation



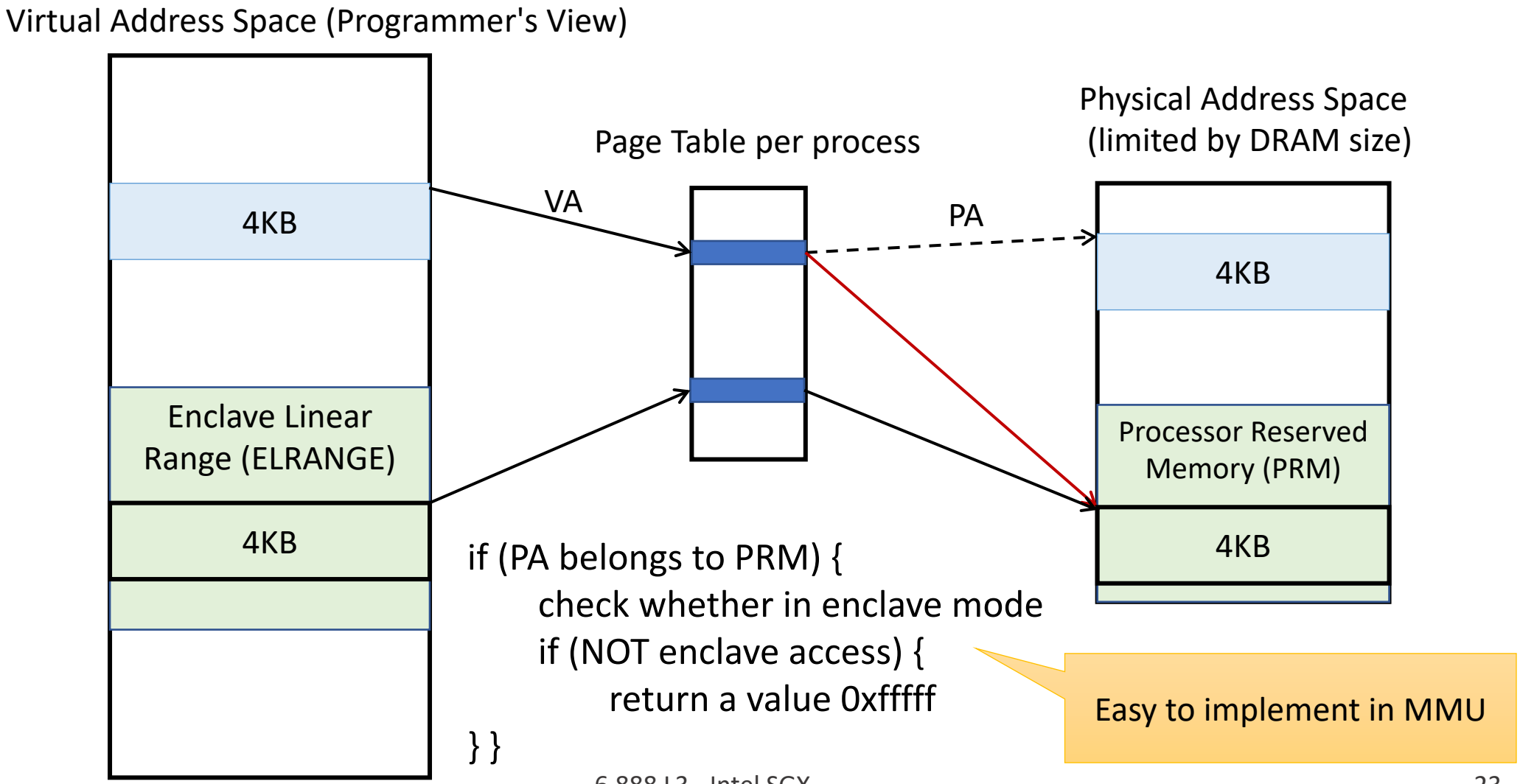
Enclave Address Translation



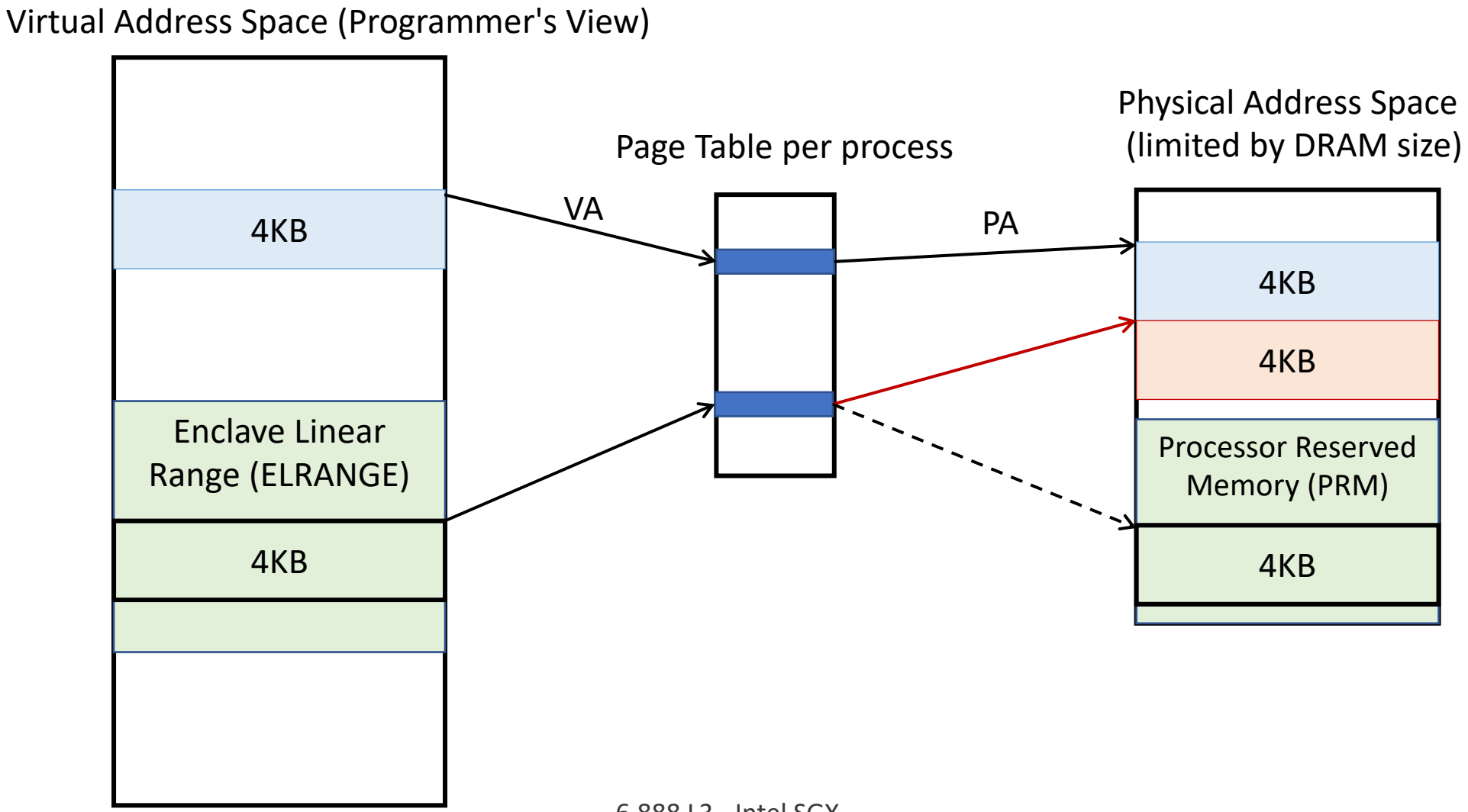
Malicious Address Translation



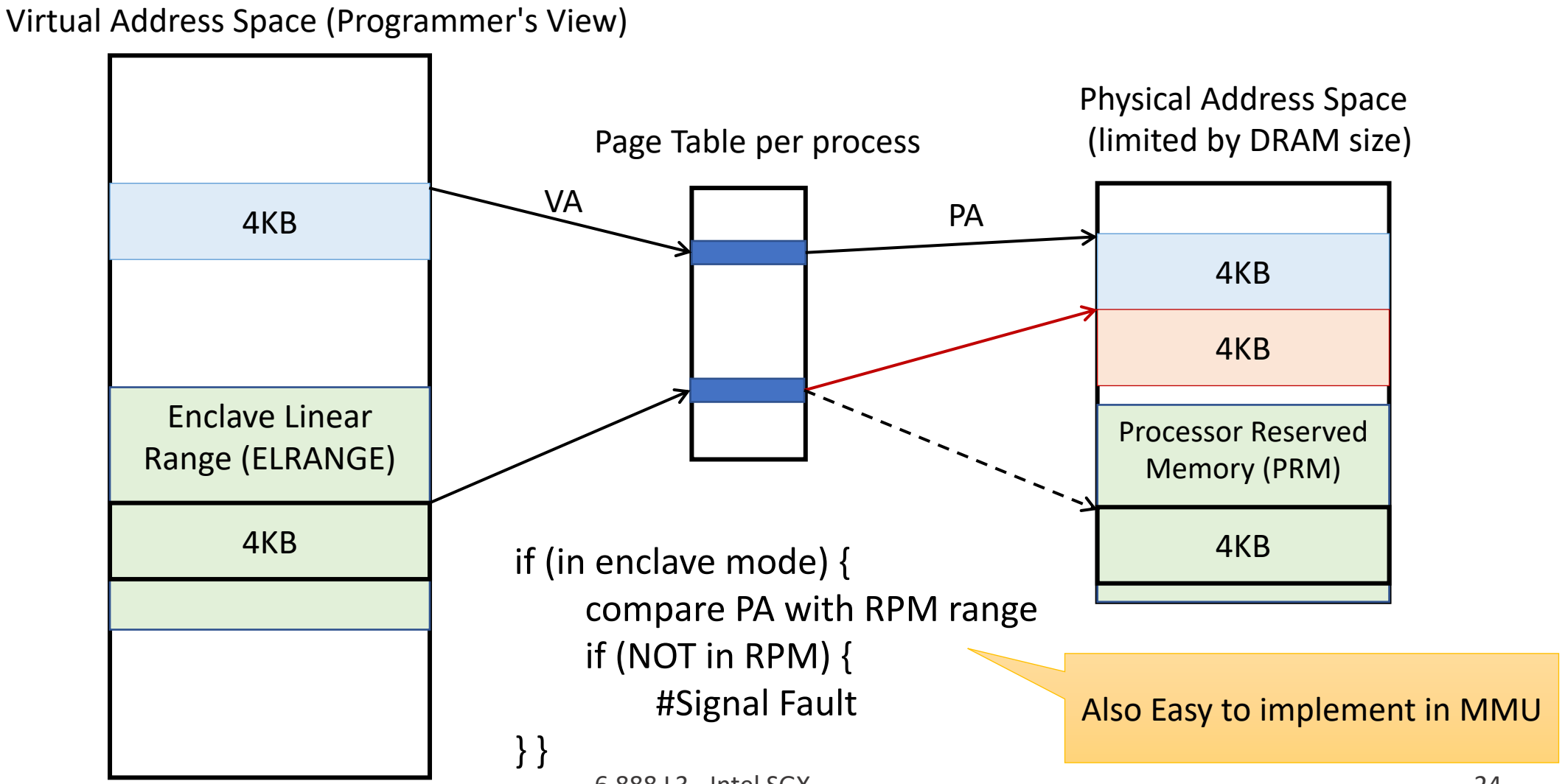
Malicious Address Translation



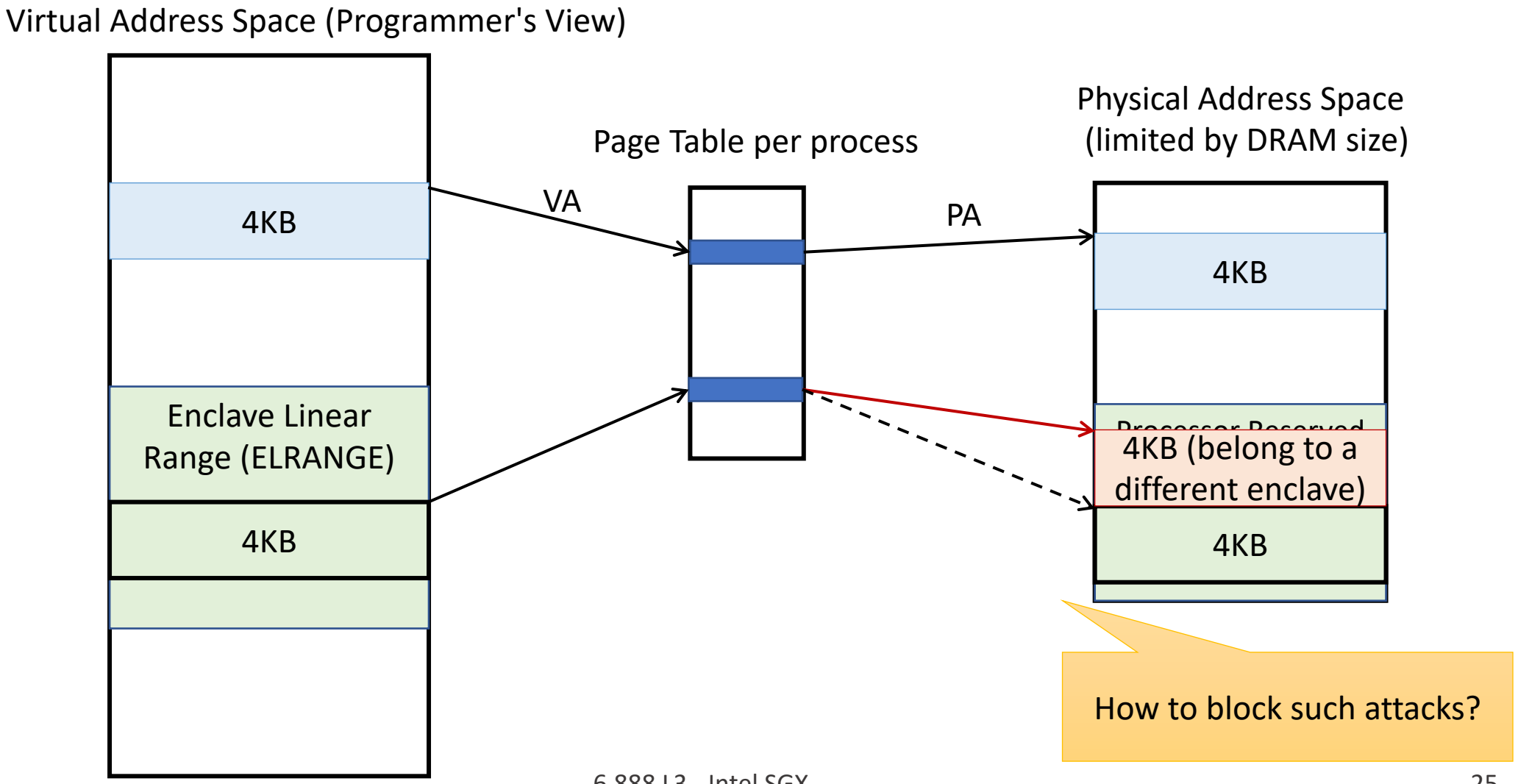
Malicious Address Translation



Malicious Address Translation

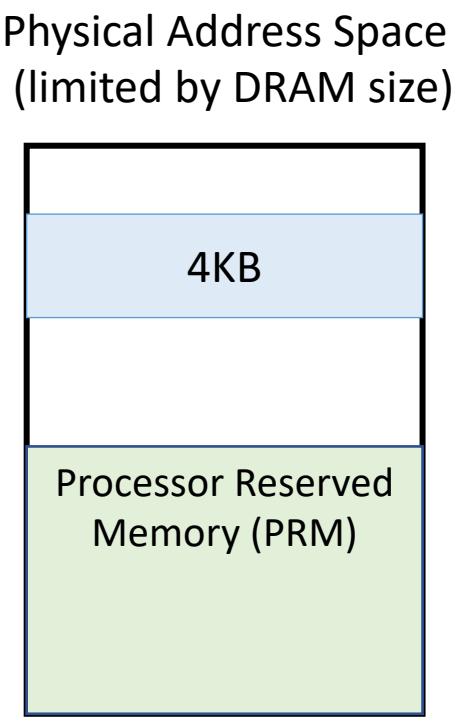


Malicious Address Translation



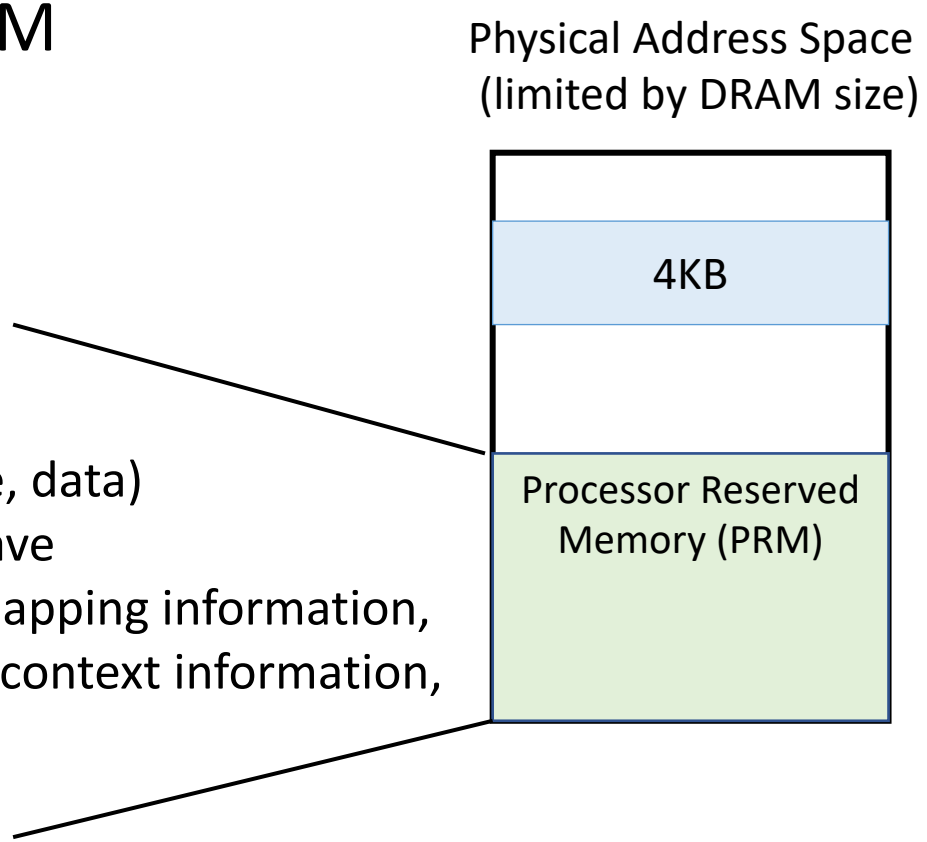
SGX Memory Organization

- Keep page mapping metadata in PRM
- MMU performs extra checks



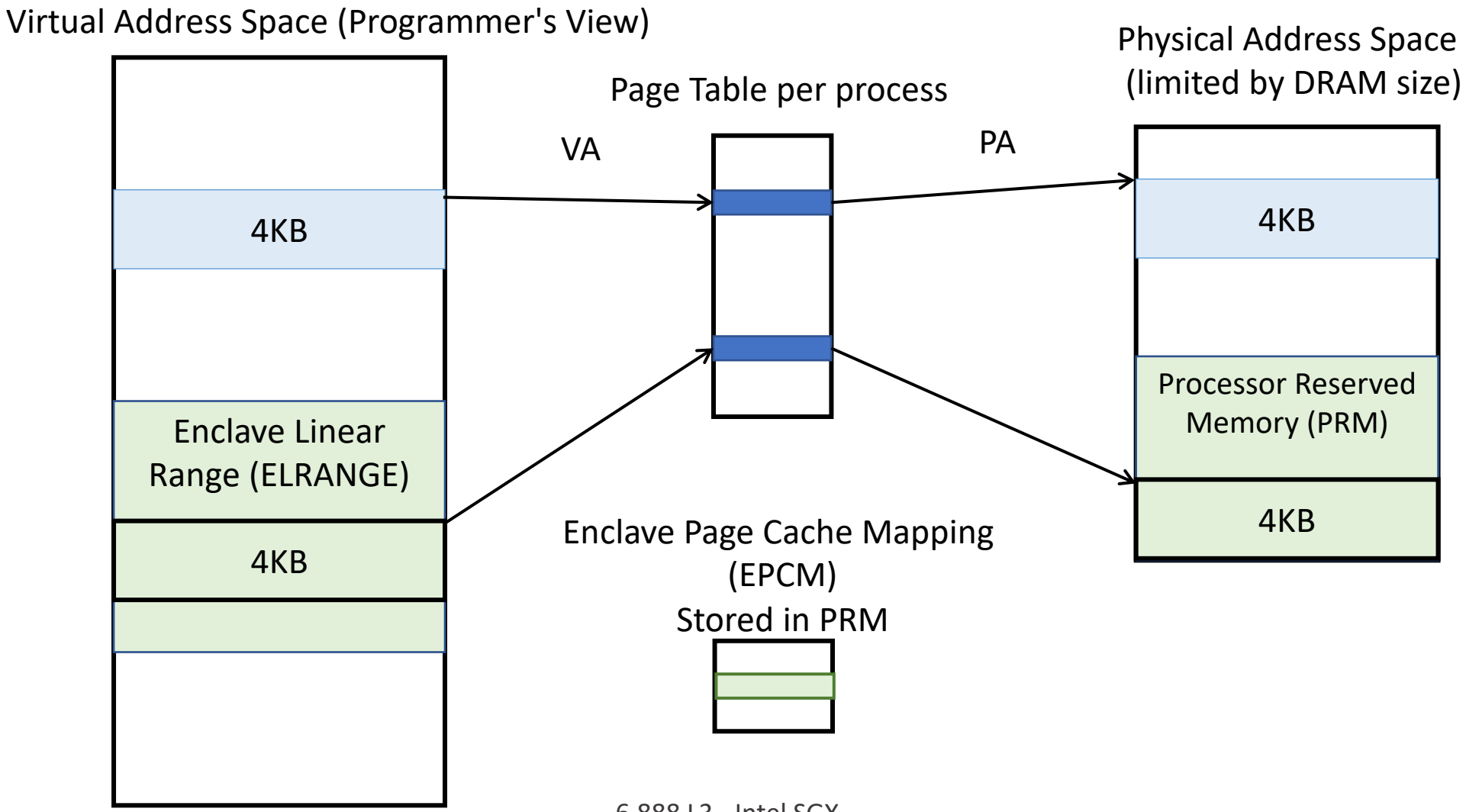
SGX Memory Organization

- Keep page mapping metadata in PRM
- MMU performs extra checks

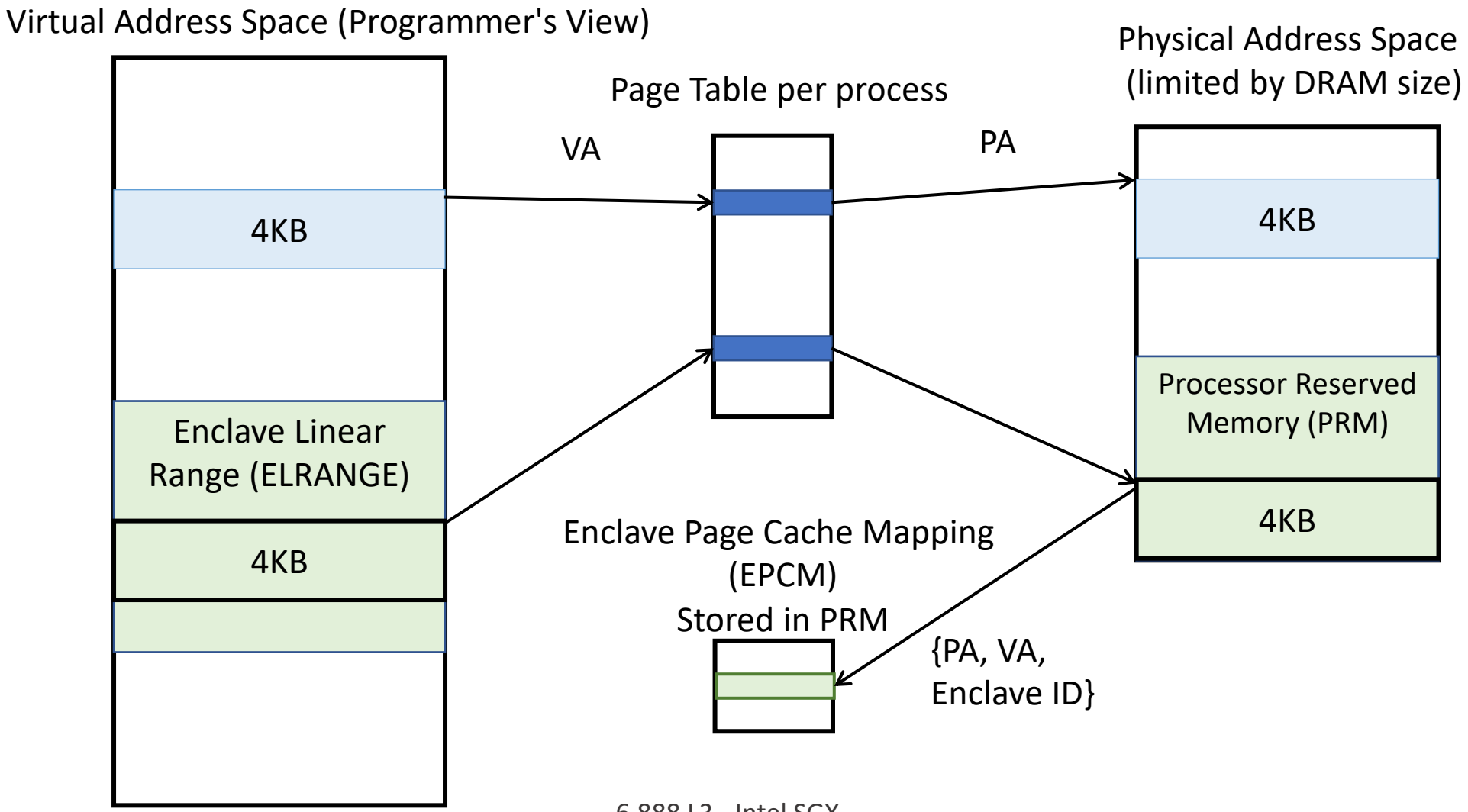


- Enclave pages (code, data)
- Meta data per enclave
  - enclave page mapping information, enclave thread context information, etc.

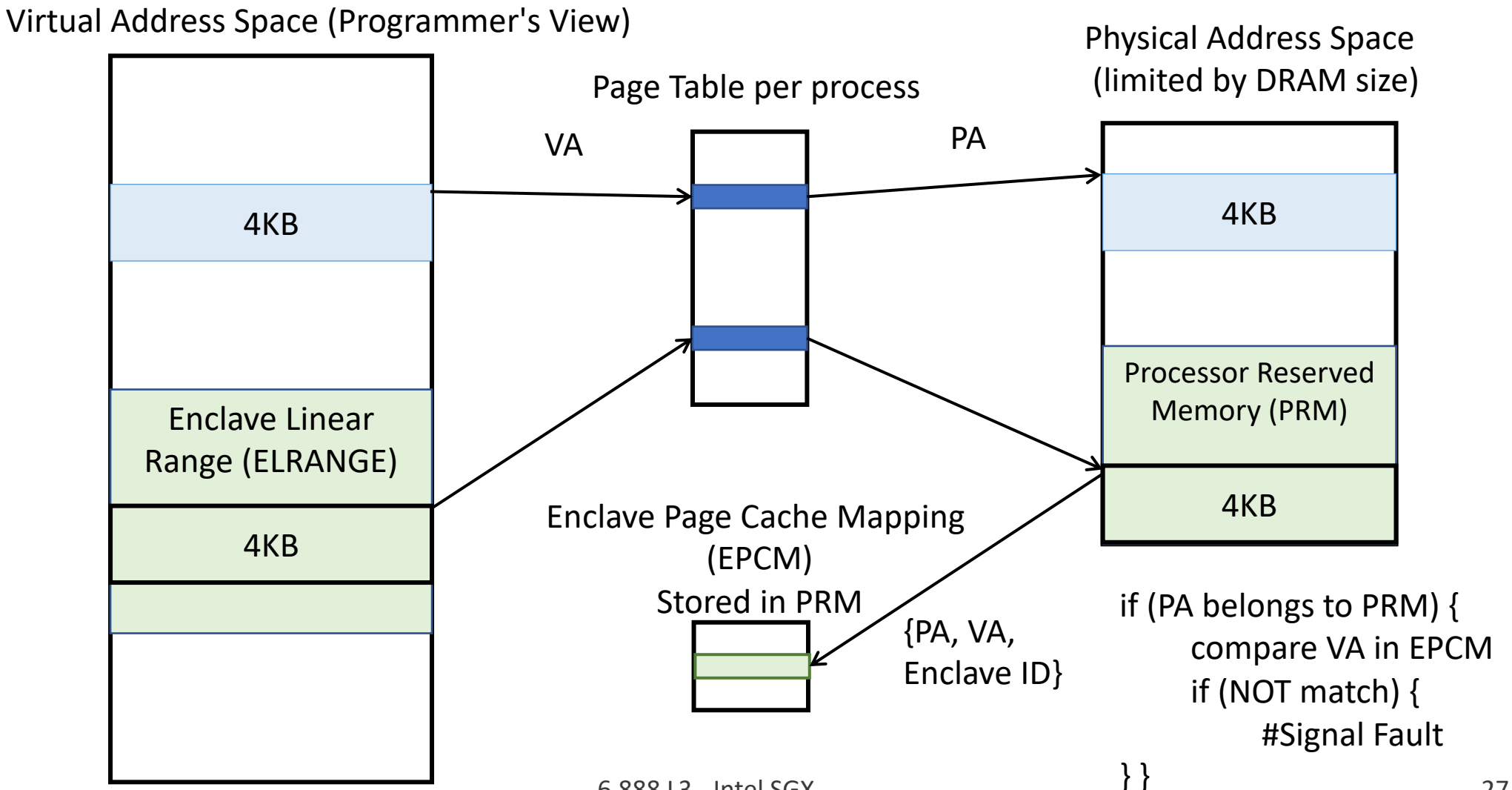
Enclave Page Mapping Information



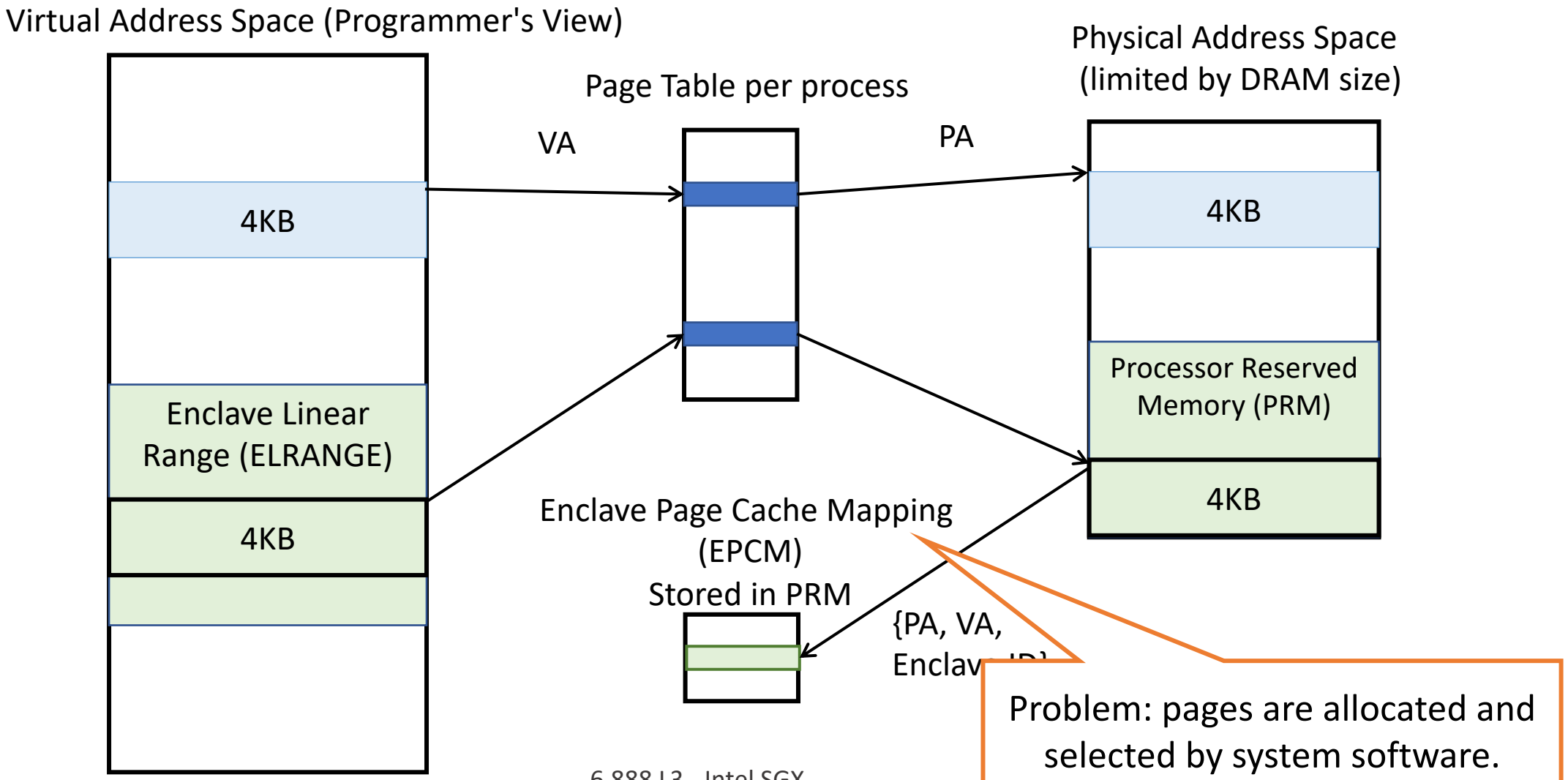
Enclave Page Mapping Information



Enclave Page Mapping Information



Enclave Page Mapping Information



So far ……

- Once the enclave is initialized correctly, it can be isolated from system software using
  - Hardware access control (supported by MMU)
  - Hardware support for secure context switch

- How to ensure the initialization is correct?

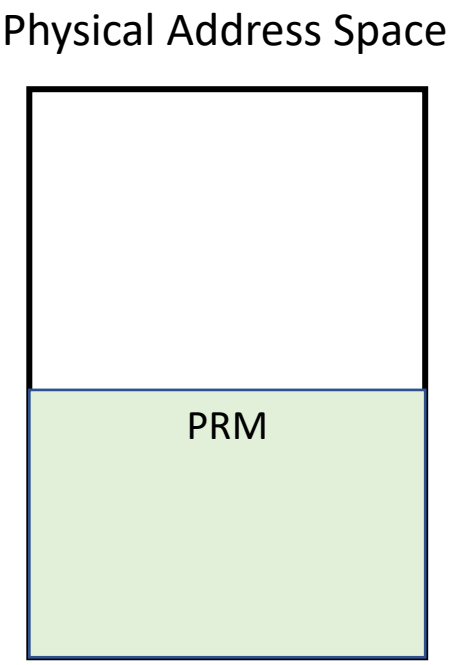
So far ……

- Once the enclave is initialized correctly, it can be isolated from system software using
  - Hardware access control (supported by MMU)
  - Hardware support for secure context switch

- How to ensure the initialization is correct?
  - Software Attestation (similar to secure boot)

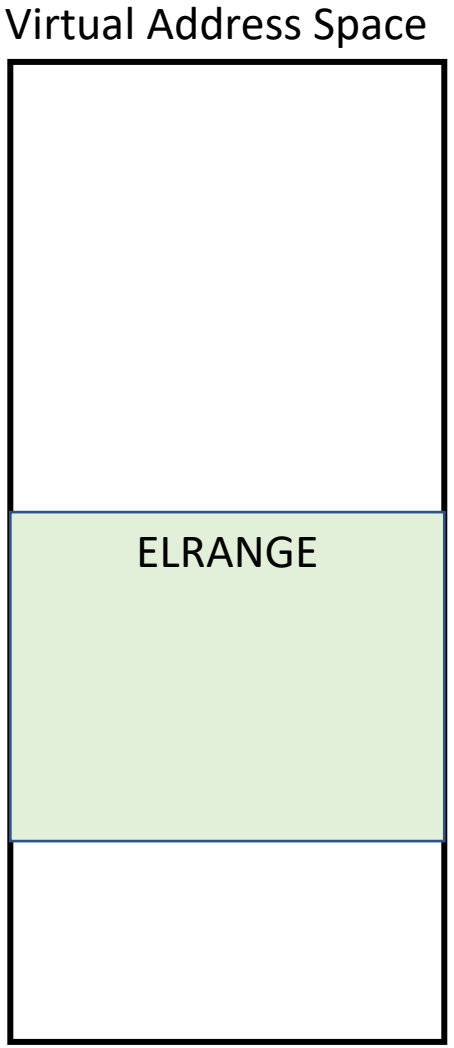
Enclave Initialization

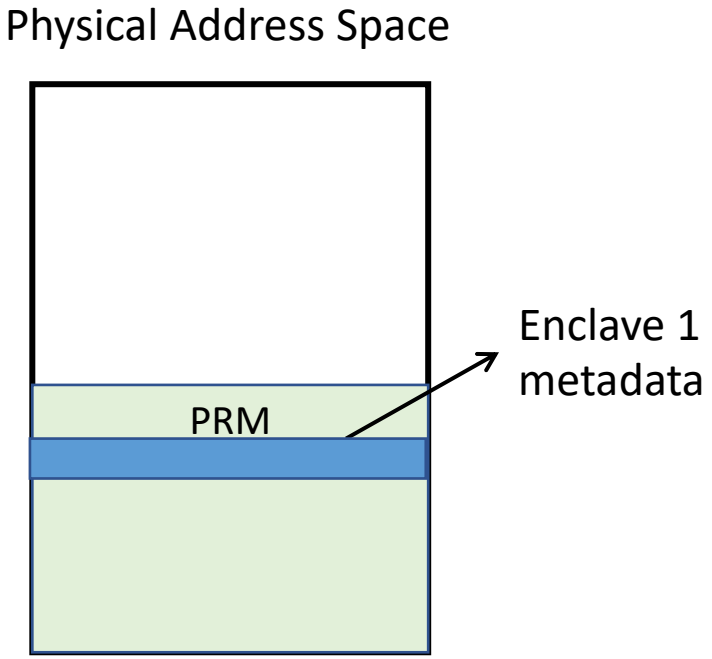
- BIOS setup PRM region



Enclave Initialization

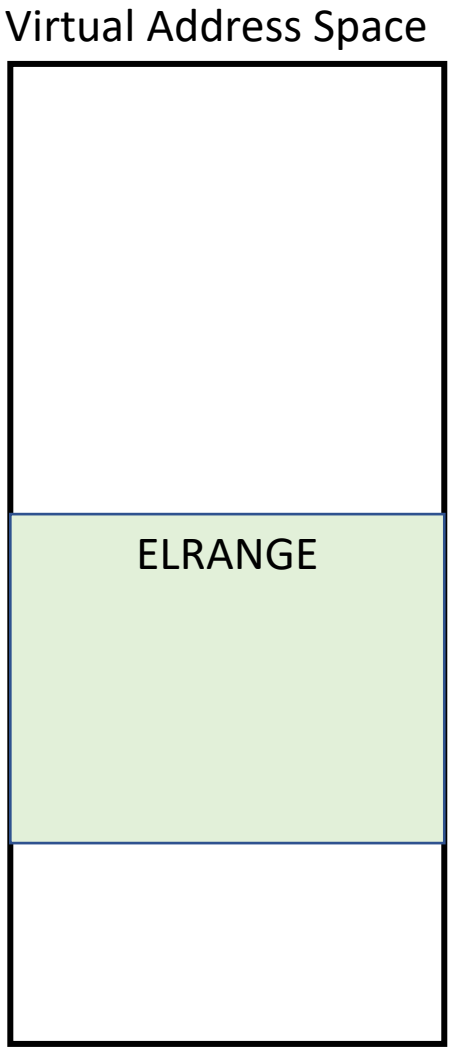
- Enclave creation (ECREATE)

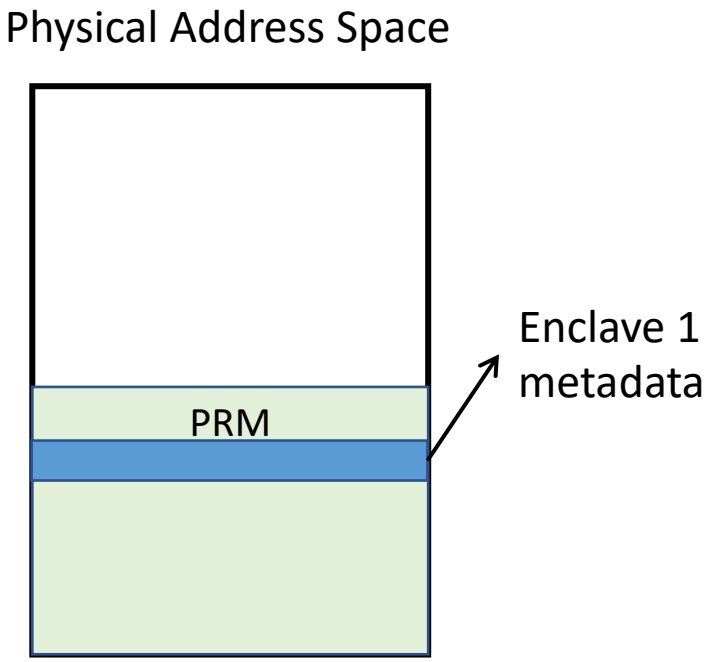




Enclave Initialization

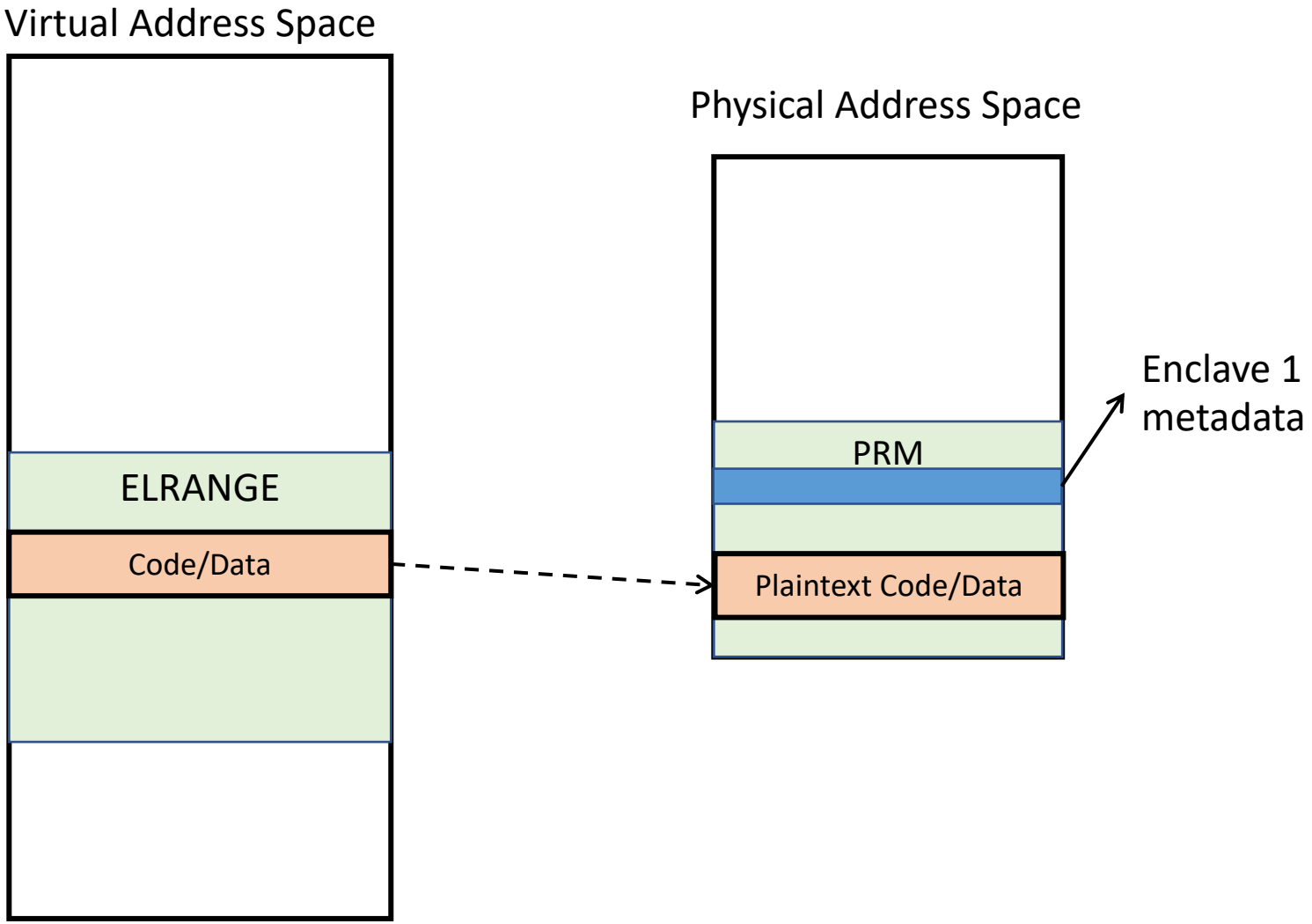
- Add page (EADD)
- Measure (EEXTEND)





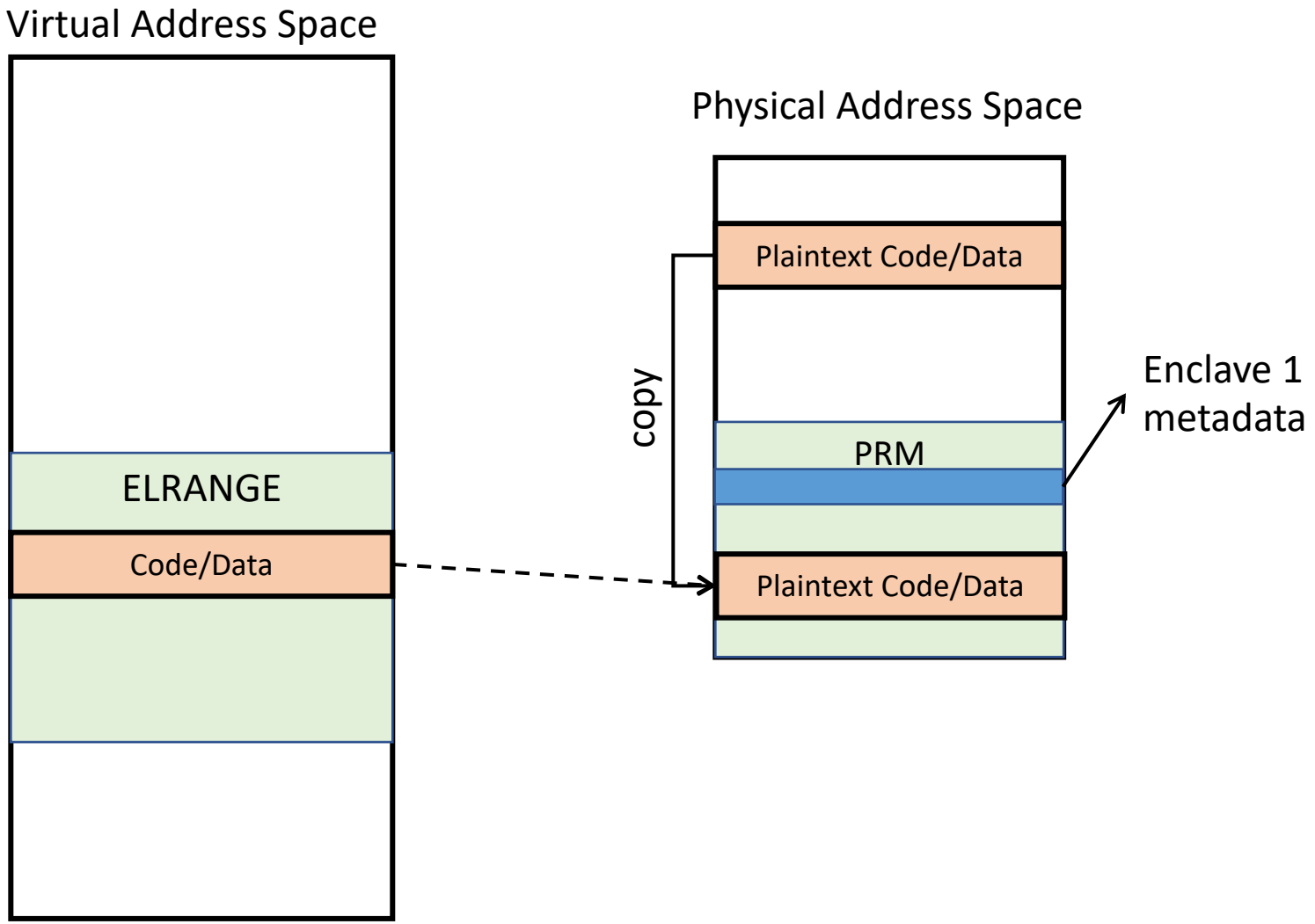
Enclave Initialization

- Add page (EADD)
- Measure (EEXTEND)



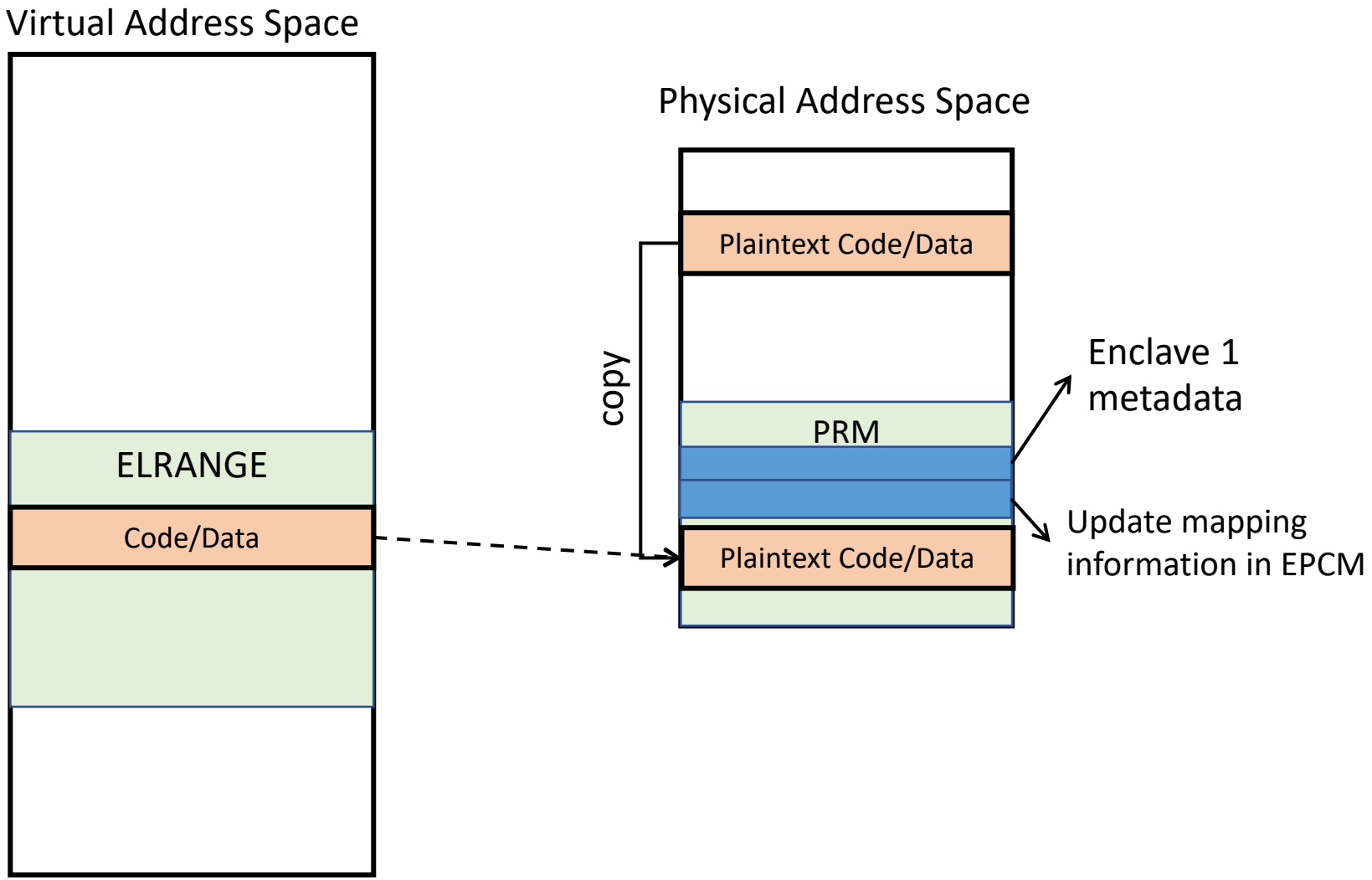
Enclave Initialization

- Add page (EADD)
- Measure (EEXTEND)



Enclave Initialization

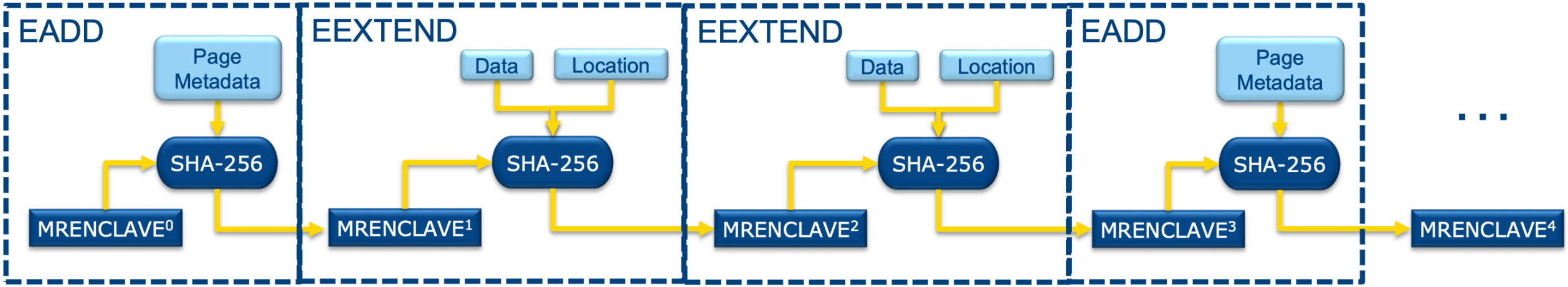
- Add page (EADD)
- Measure (EEXTEND)



Enclave Measurement

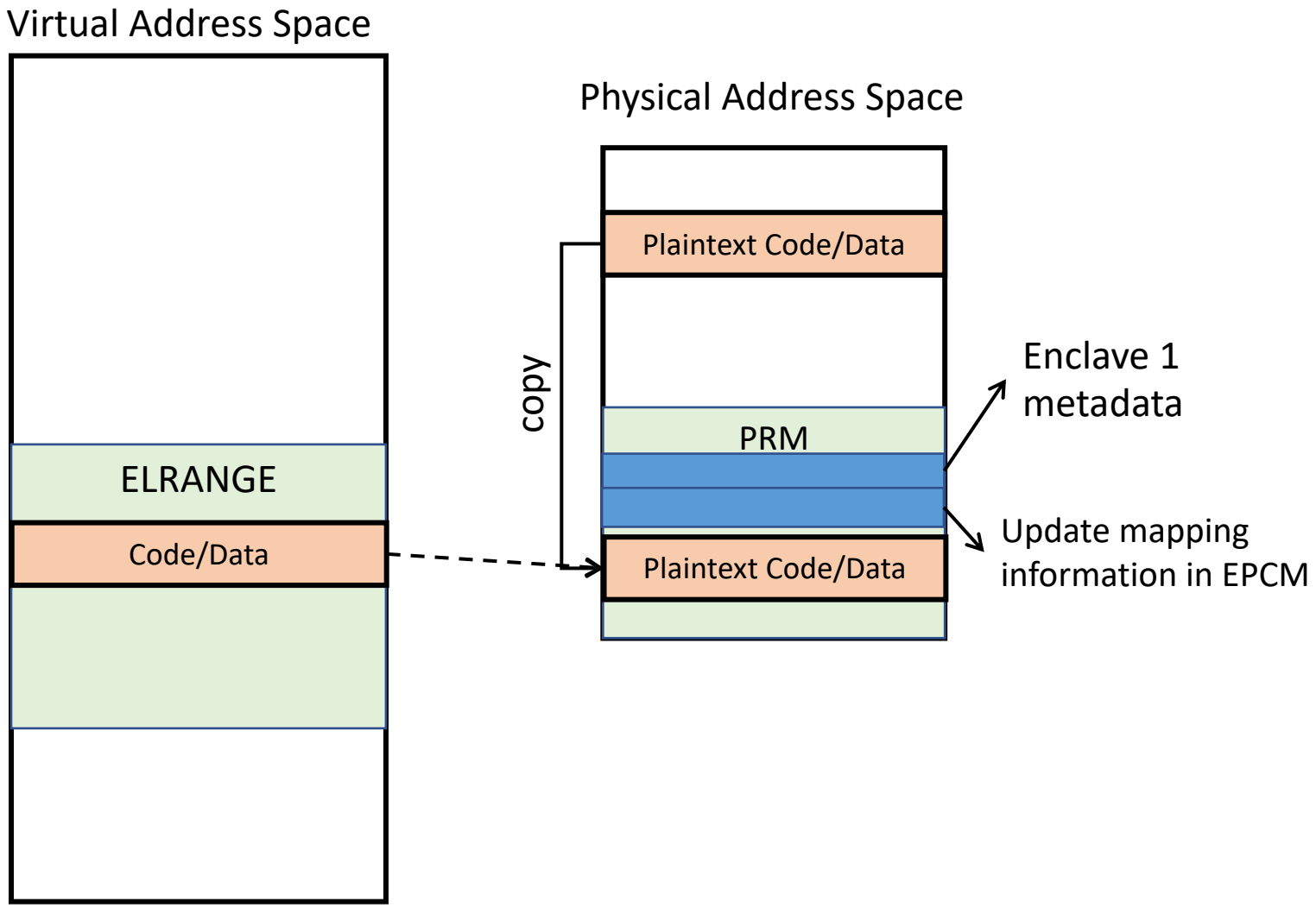
- Hardware generates a cryptographic log of the build process
  - Code, data, stack, and heap contents
  - Location of each page within the enclave
  - Security attributes (e.g., page permissions) and enclave capabilities

- Enclave identity (MRENCLAVE) is a 256-bit digest of the log that represents the enclave



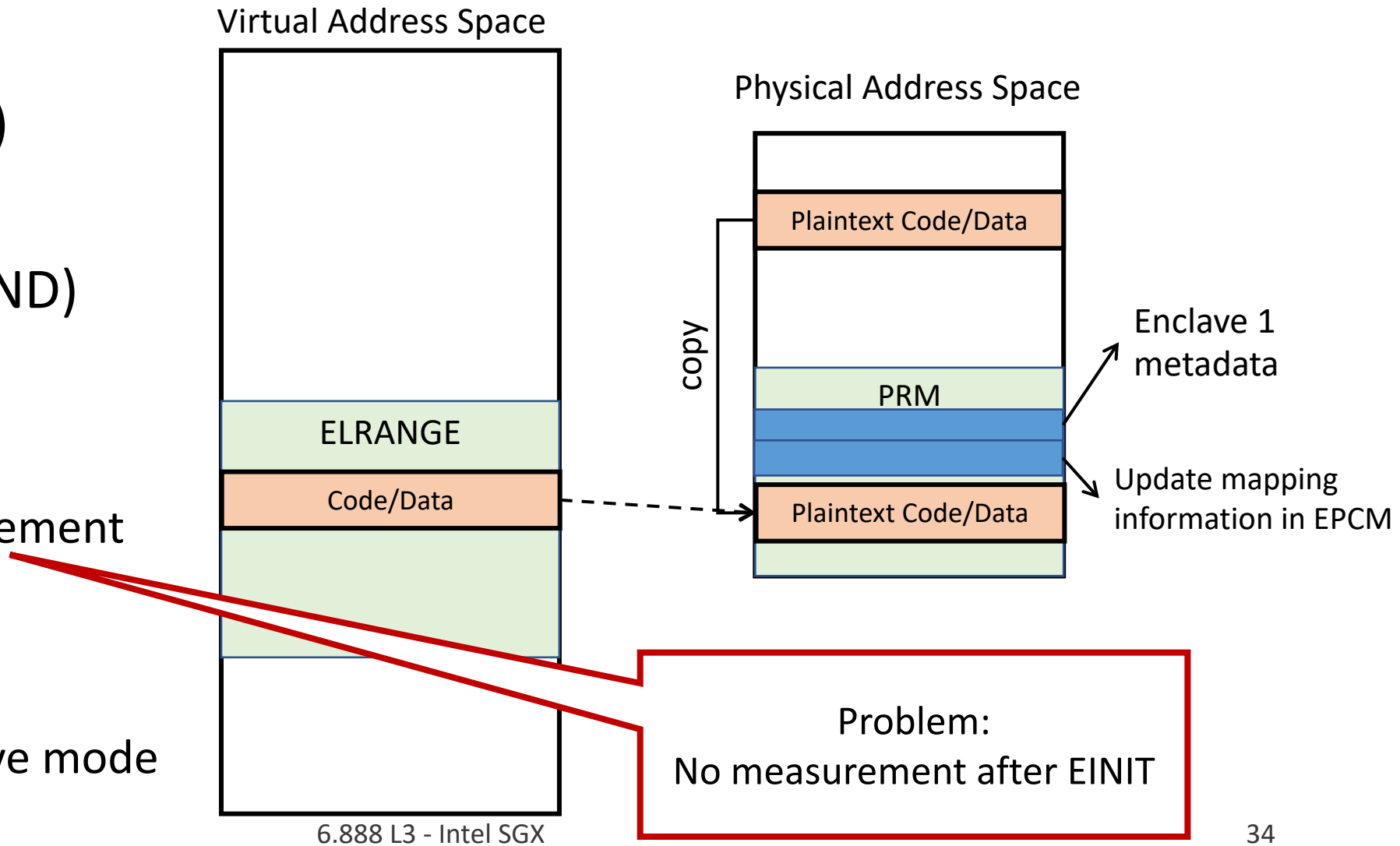
Enclave Initialization

- Add page (EADD)
- Measure (EEXTEND)
- Init (EINIT)
  - Finalize measurement
- Active (EENTER)
  - Switch to enclave mode



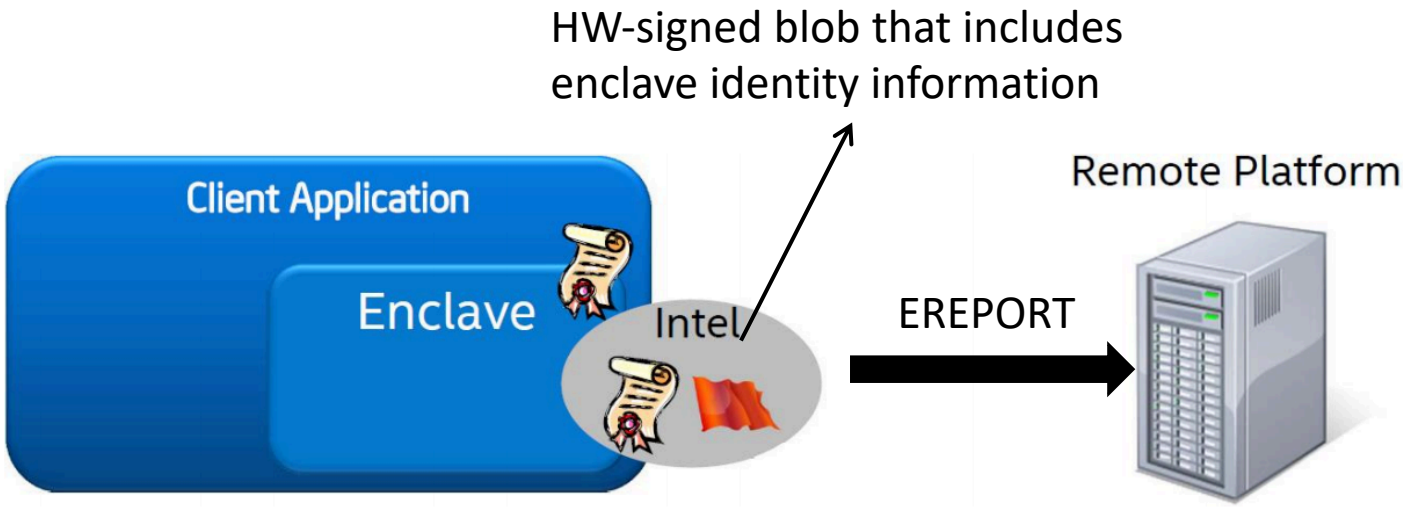
Enclave Initialization

- Add page (EADD)
- Measure (EEXTEND)
- Init (EINIT)
  - Finalize measurement
- Active (EENTER)
  - Switch to enclave mode



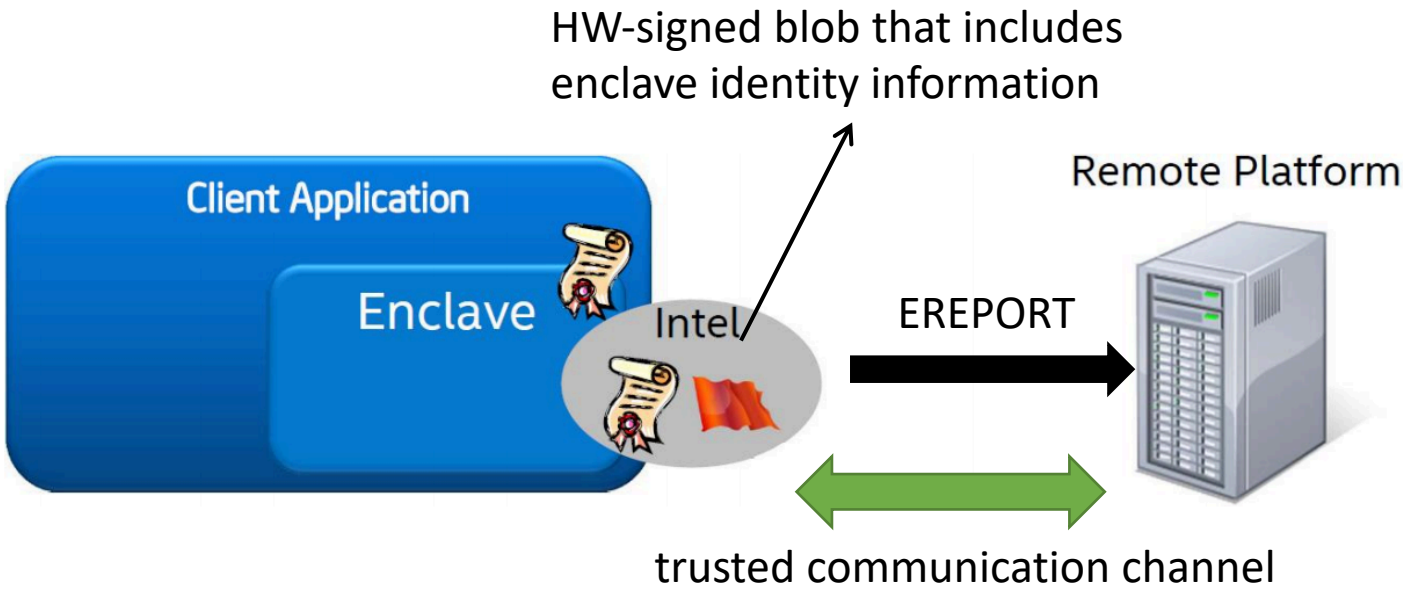
Enclave Attestation and Sealing

- HW based attestation provides evidence that “this is the right application executing on an authentic platform” (approach similar to secure boot attestation)

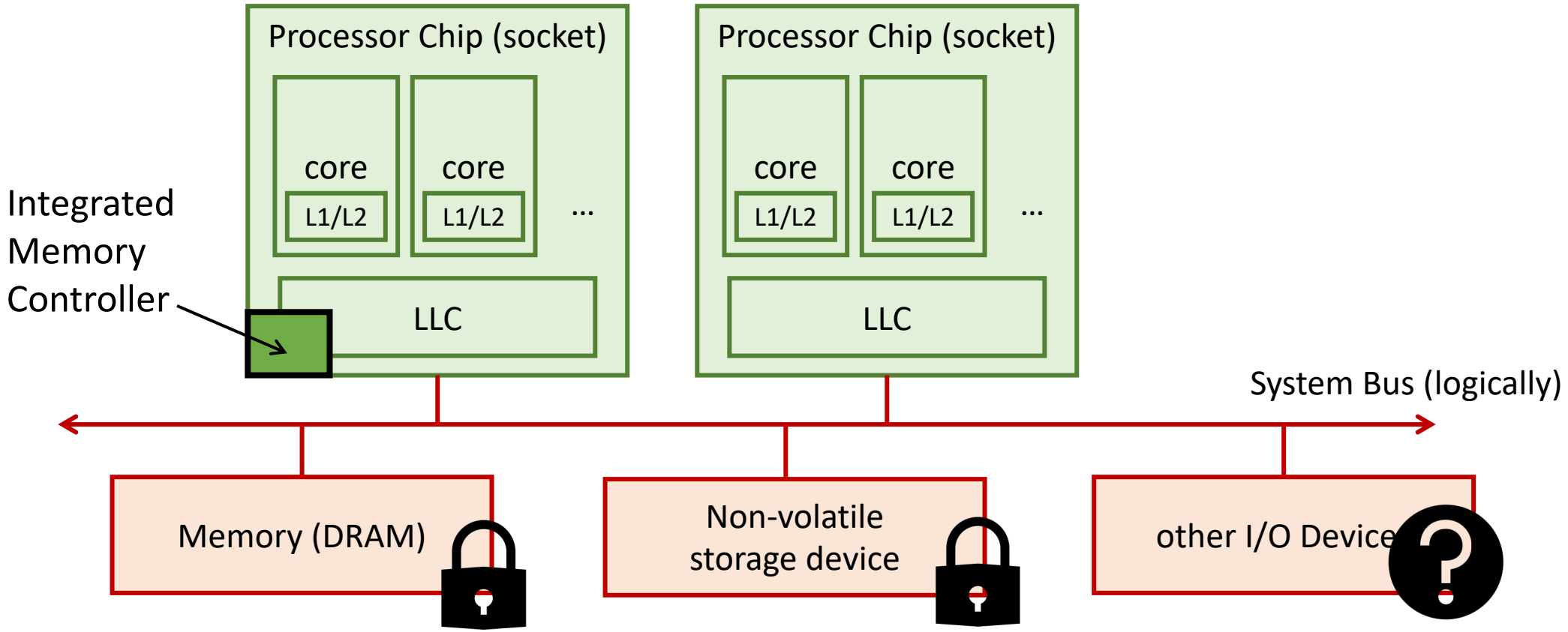


Enclave Attestation and Sealing

- HW based attestation provides evidence that “this is the right application executing on an authentic platform” (approach similar to secure boot attestation)

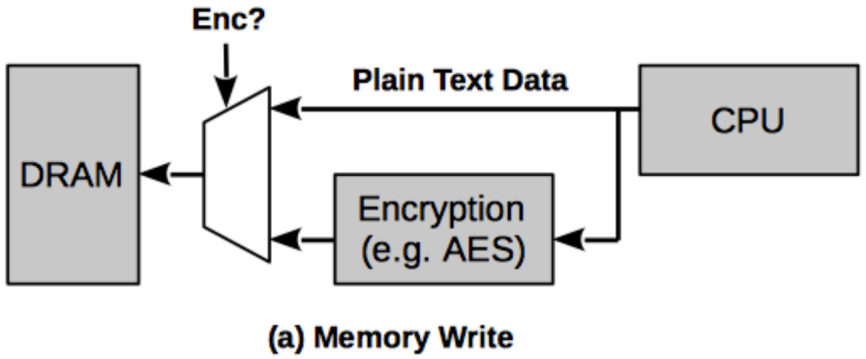


Protect Memory

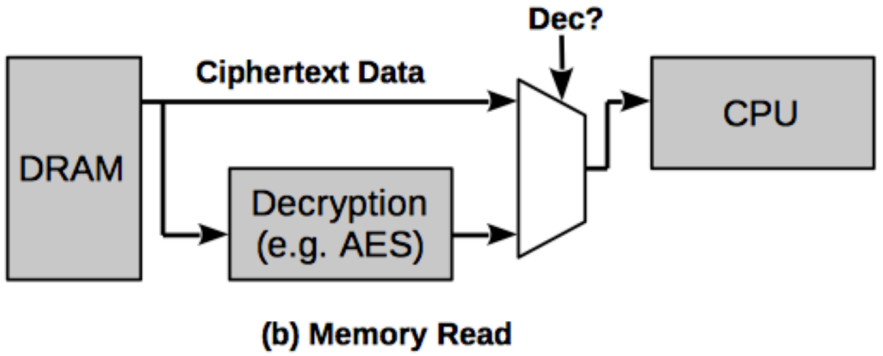


Confidentiality Protection with Encryption

- Secret key is stored inside chip
- For freshness, encrypt with nonce (counter)
- {nonce, ciphertext} per cache block are stored externally in DRAM



(a) Memory Write



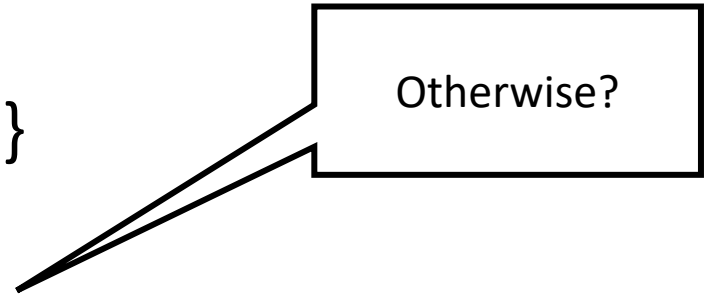
(b) Memory Read

Integrity Protection with Hash

- For each cache line: {ciphertext + nonce + hash}

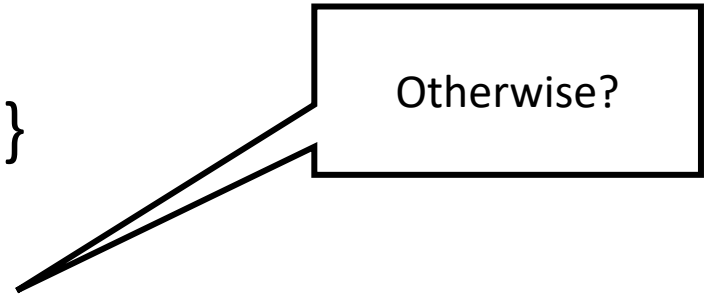
Integrity Protection with Hash

- For each cache line: {ciphertext + nonce + hash}
- Problem:
  - Need to store hashes or nonces on-chip → high on-chip storage requirement
  - Too much storage requirement (~64bits / block) → high off-chip storage requirement



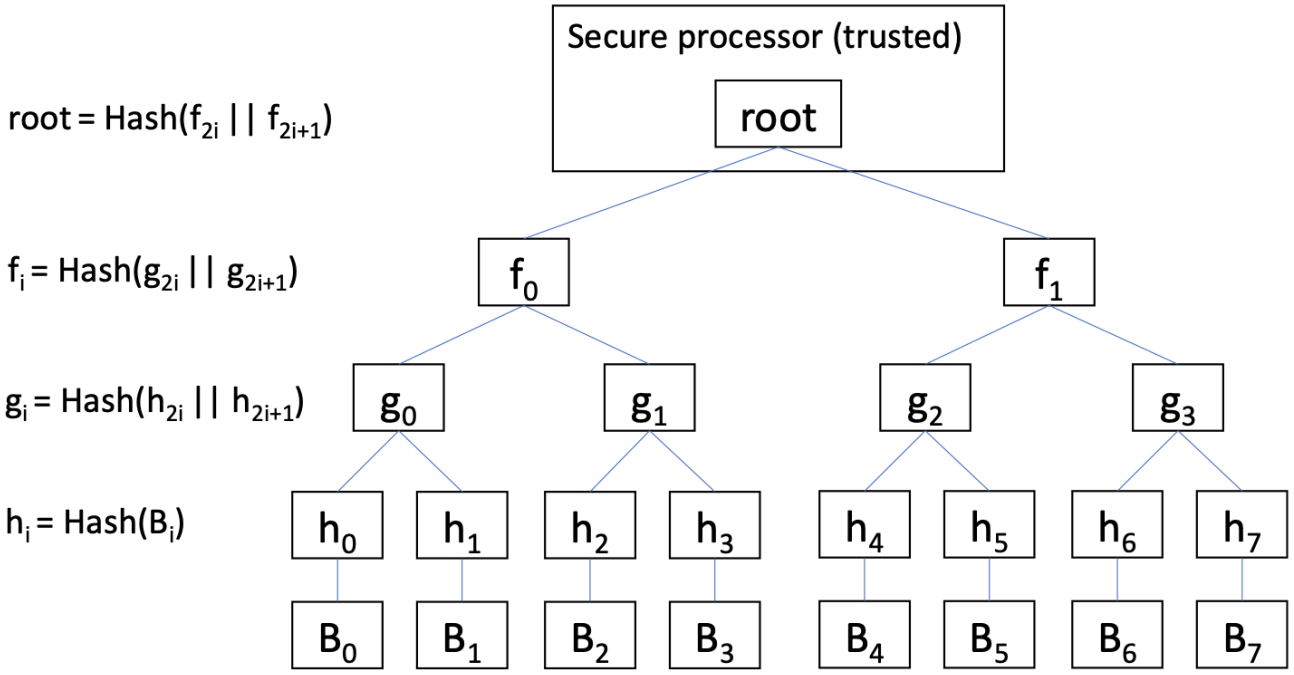
Integrity Protection with Hash

- For each cache line: {ciphertext + nonce + hash}
- Problem:
  - Need to store hashes or nonces on-chip → high on-chip storage requirement
  - Too much storage requirement (~64bits / block) → high off-chip storage requirement
- General solution:
  - Integrity Tree (Merkle tree)



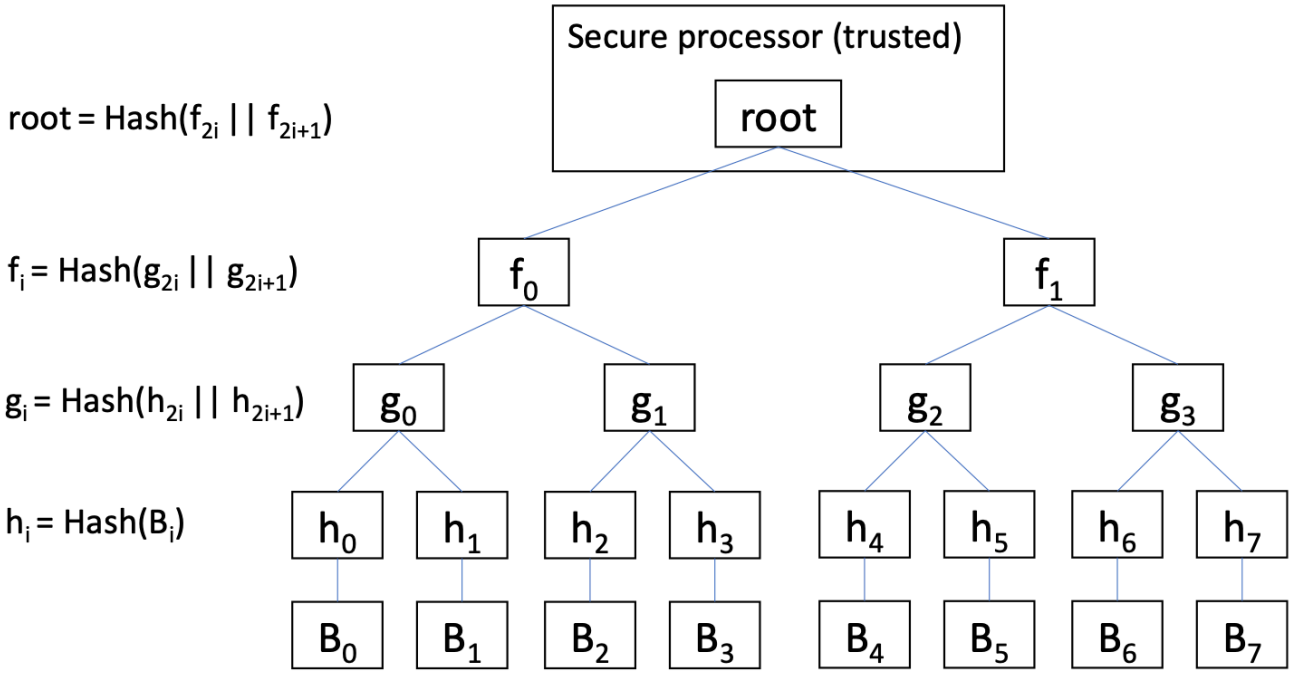
Operations on Merkle Tree

- Only need to store the root node on chip



Operations on Merkle Tree

- Only need to store the root node on chip
- How to verify block B1?
- Write to block B3?



Next Lecture:
Side Channel Introduction