OpenSGX: An Open Platform for SGX Research

Prerit Jain, Soham Desai, **Seongmin Kim***, Ming-Wei Shih, JaeHyuk Lee, Changho Choi, Youjung Shin, Taesoo Kim, Brent Byunghoon Kang, Dongsu Han

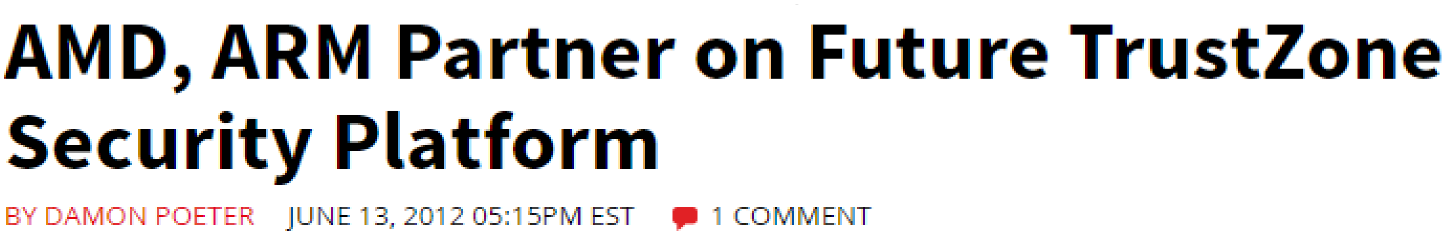
Georgia Tech

KAIST

Trusted Execution Environment (TEE)

- Hardware technologies for trusted computing
  - Isolated execution: integrity of code, confidentiality
  - To protect application from untrusted platform

Trusted Execution Environment (TEE)

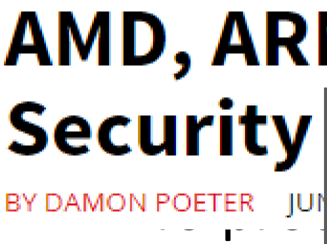


AMD, ARM Partner on Future TrustZone Security Platform

BY DAMON POETER JUNE 13, 2012 05:15PM EST 1 COMMENT

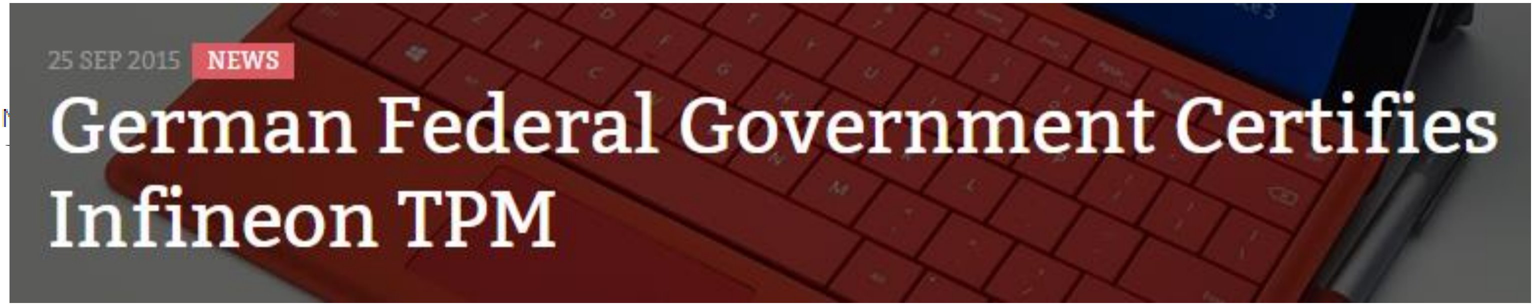
y

Trusted Execution Environment (TEE)



AMD, ARM Partner on Future TrustZone Security

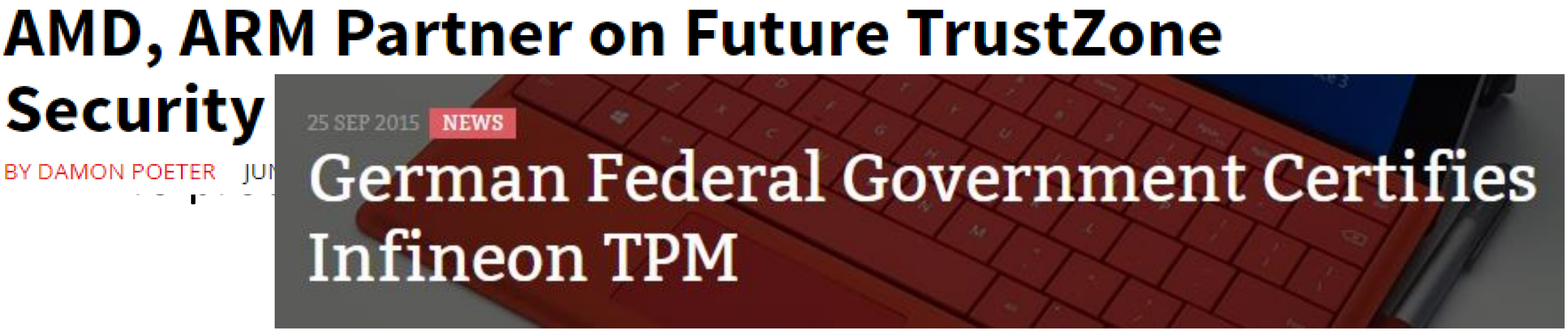
BY DAMON POETER JU



25 SEP 2015 NEWS

German Federal Government Certifies Infineon TPM

Trusted Execution Environment (TEE)



AMD, ARM Partner on Future TrustZone Security

BY DAMON POETER JU

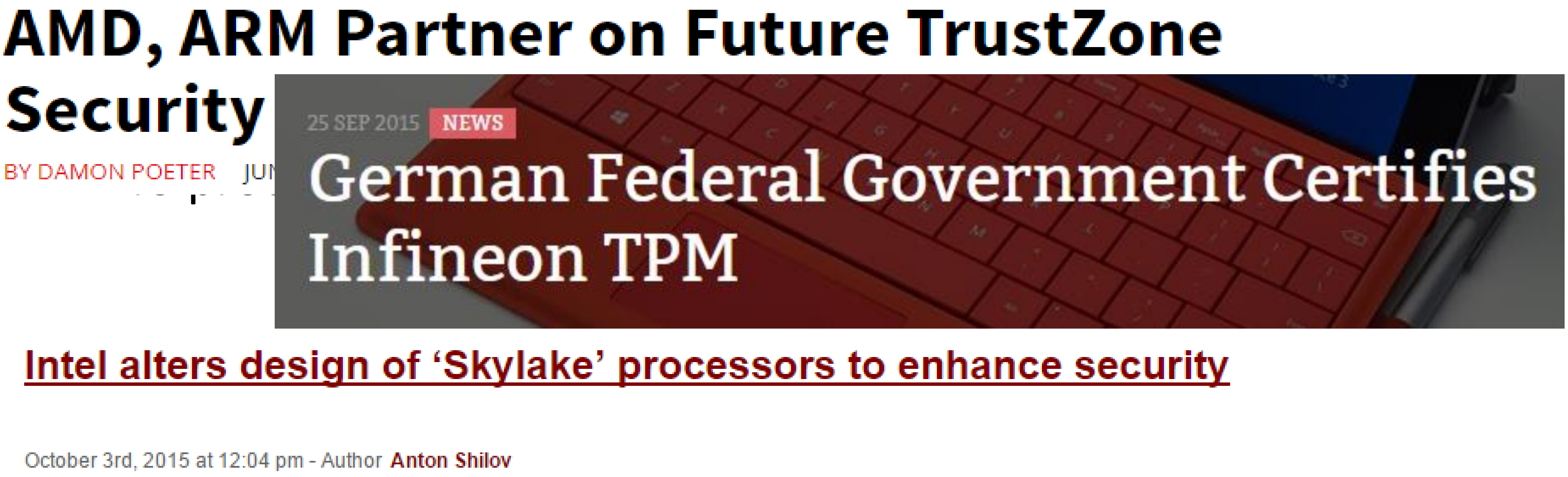
25 SEP 2015 NEWS

German Federal Government Certifies Infineon TPM

Intel alters design of ‘Skylake’ processors to enhance security

October 3rd, 2015 at 12:04 pm - Author Anton Shilov

Trusted Execution Environment (TEE)



AMD, ARM Partner on Future TrustZone Security

BY DAMON POETER JU

25 SEP 2015 NEWS

German Federal Government Certifies Infineon TPM

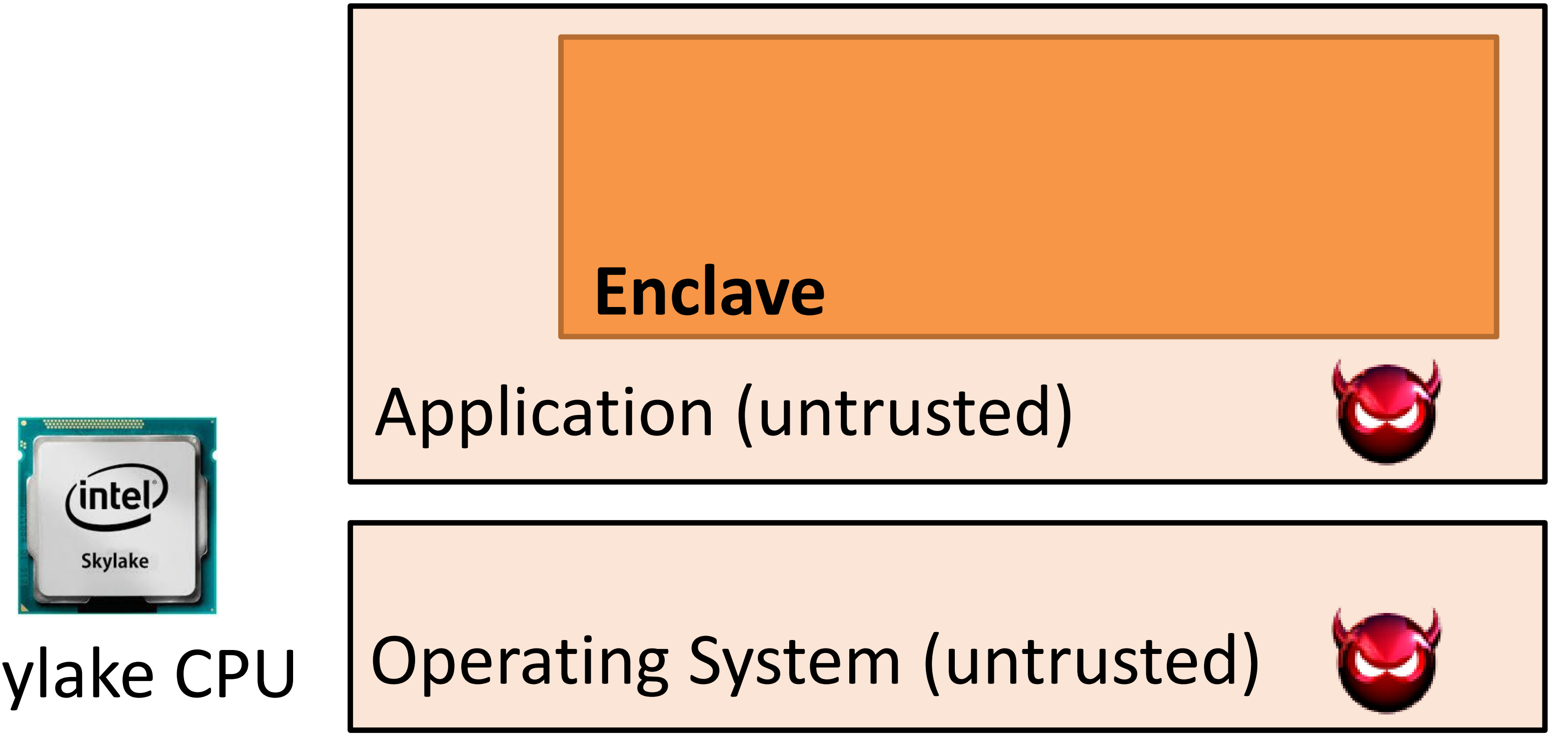
Intel alters design of ‘Skylake’ processors to enhance security

October 3rd, 2015 at 12:04 pm - Author Anton Shilov

- Practical limitations of TEEs
  - Trusted Platform Module (TPM) : Poor performance
  - ARM TrustZone : Compatibility (only for embedded devices)

Intel SGX

- An extension of x86 Instruction Set Architecture (ISA)
  - Offers native performance, Compatibility with x86
  - Application keeps its data/code inside the “**enclave**”



Enclave

Application (untrusted)

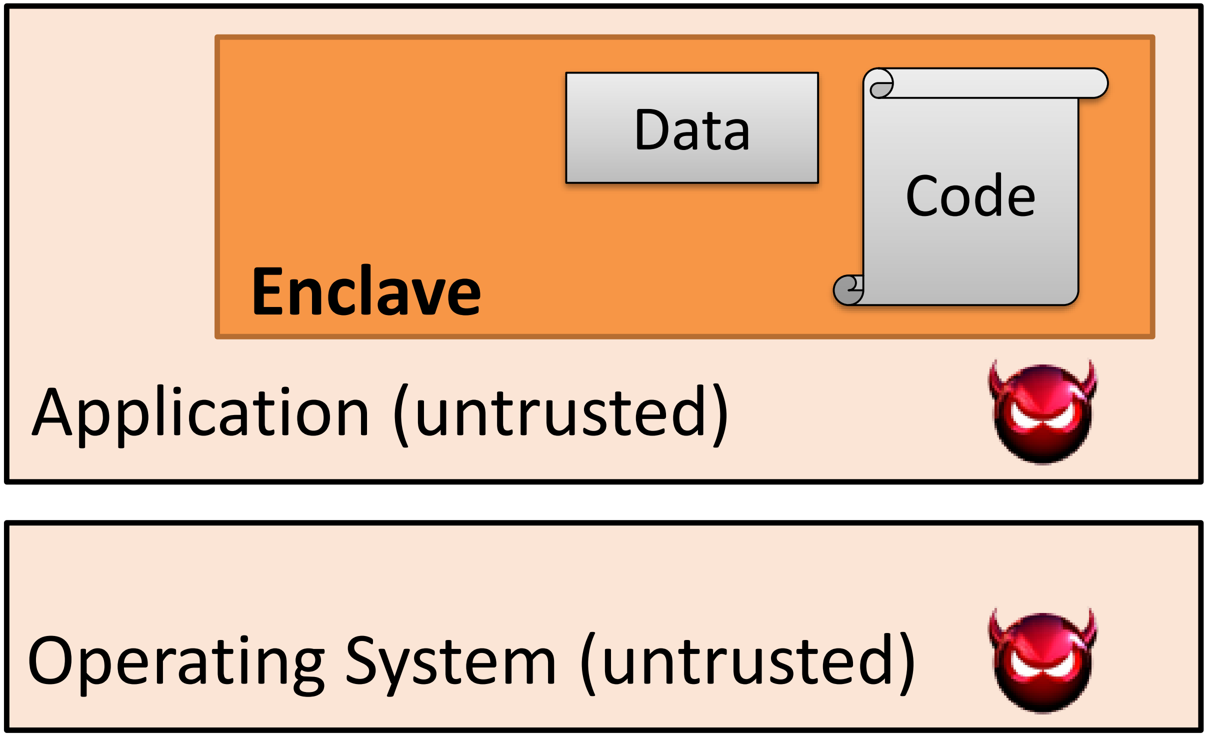
Operating System (untrusted)

Skylake CPU

Intel SGX

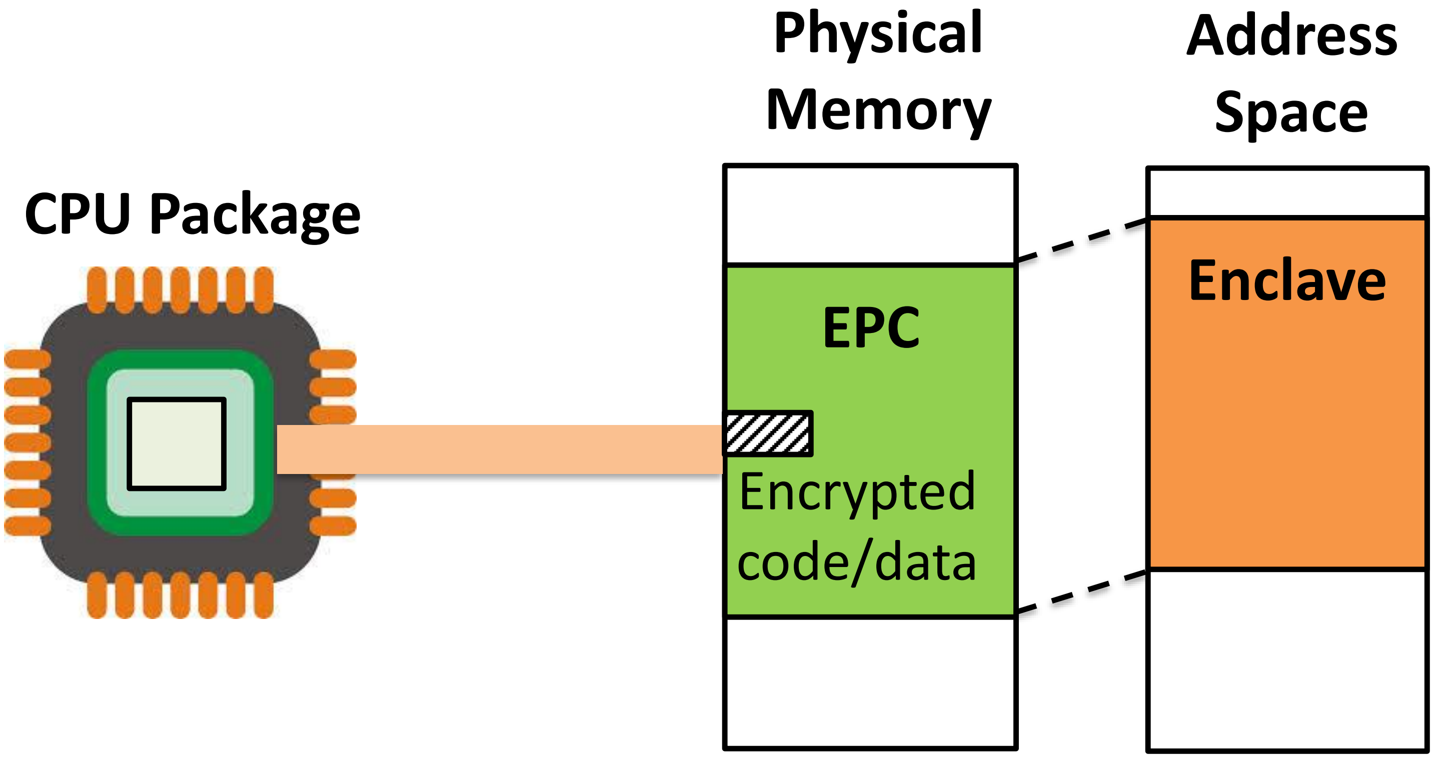
- An extension of x86 Instruction Set Architecture (ISA)
  - Offers native performance, Compatibility with x86
  - Application keeps its data/code inside the “**enclave**”

Skylake CPU



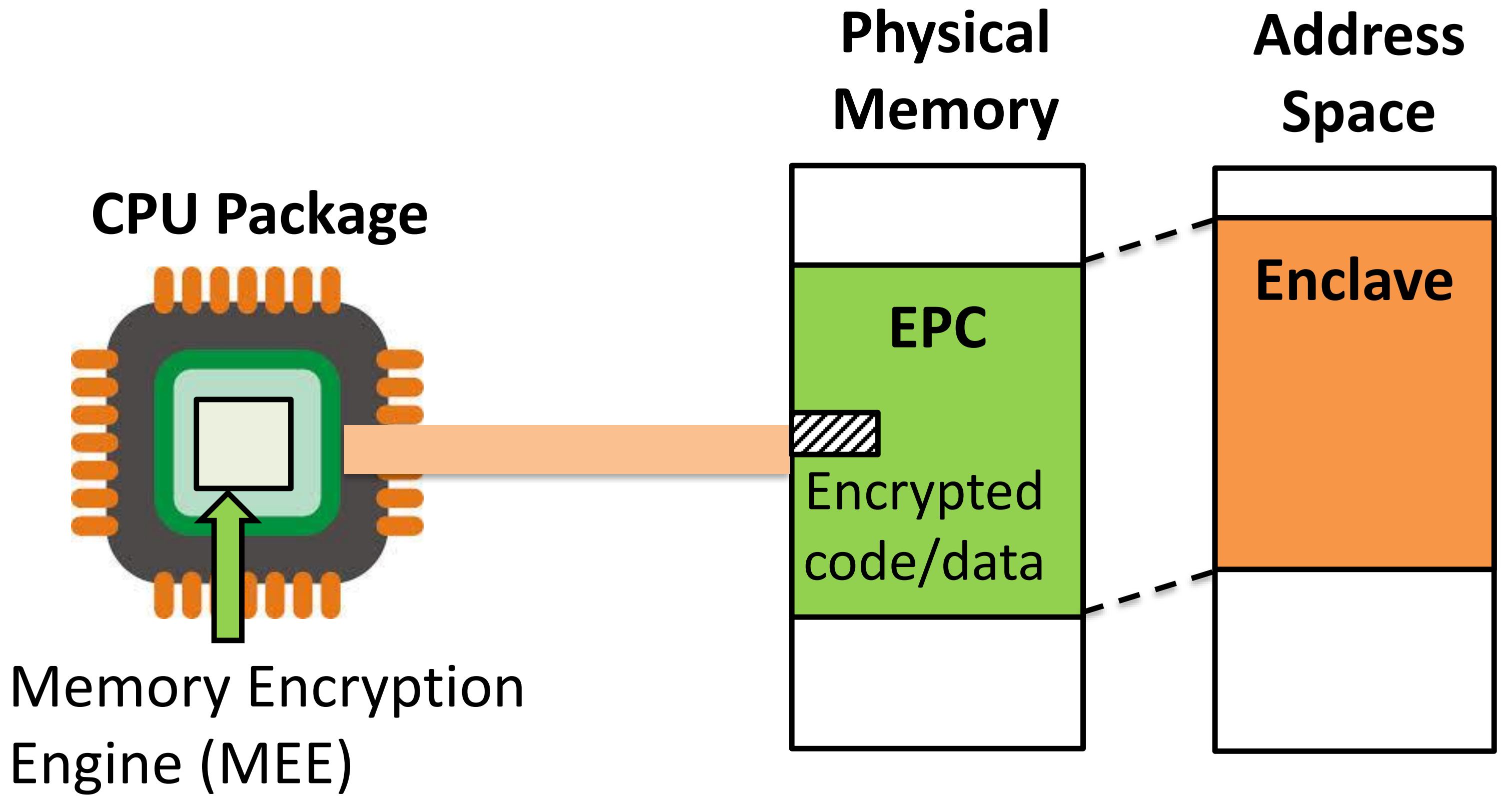
Intel SGX 101: Isolated Execution

- Smallest attack surface by reducing TCB (App + processor)
- Protect app’s secret from untrusted privilege software



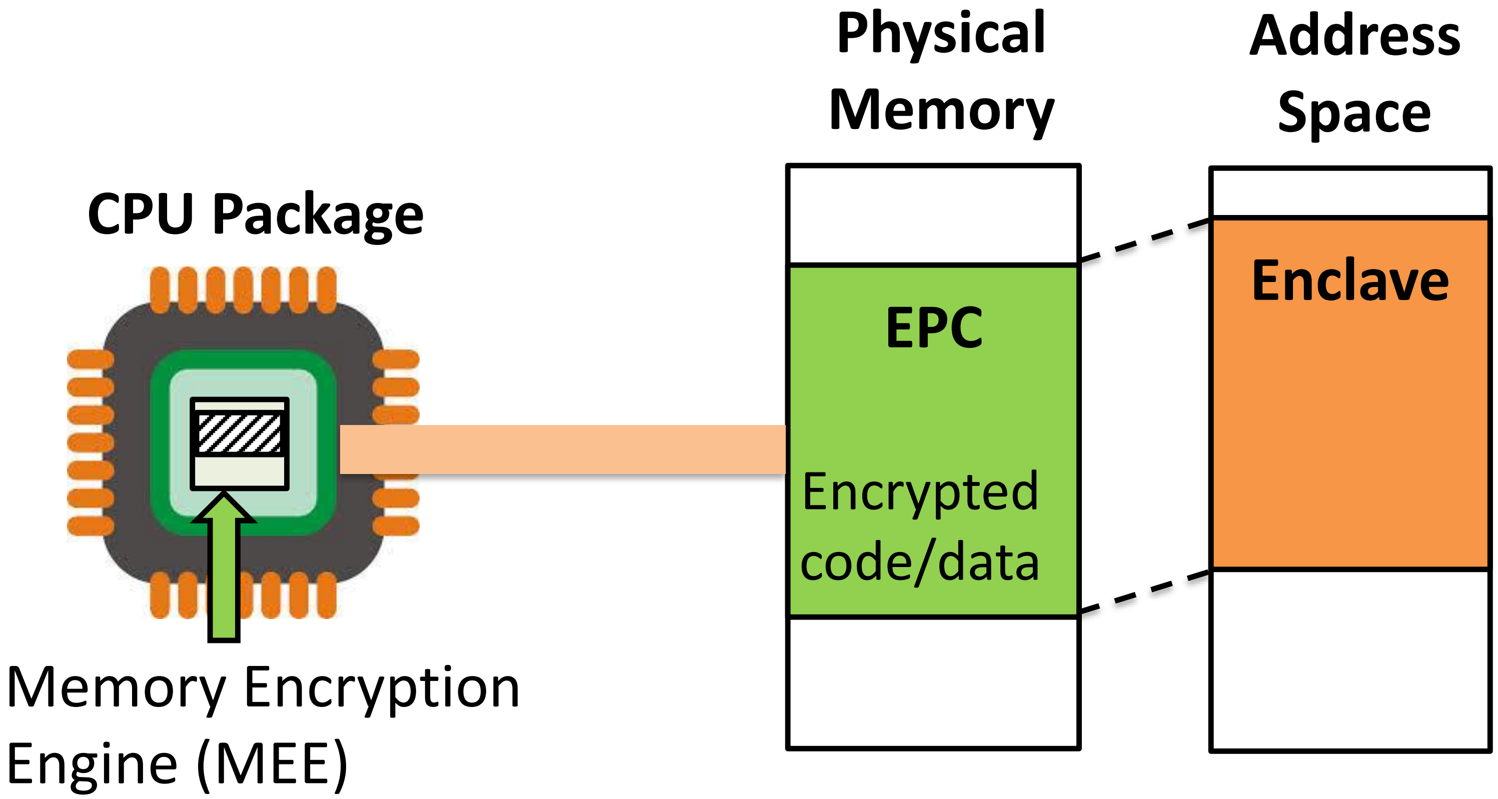
Intel SGX 101: Isolated Execution

- Smallest attack surface by reducing TCB (App + processor)
- Protect app's secret from untrusted privilege software



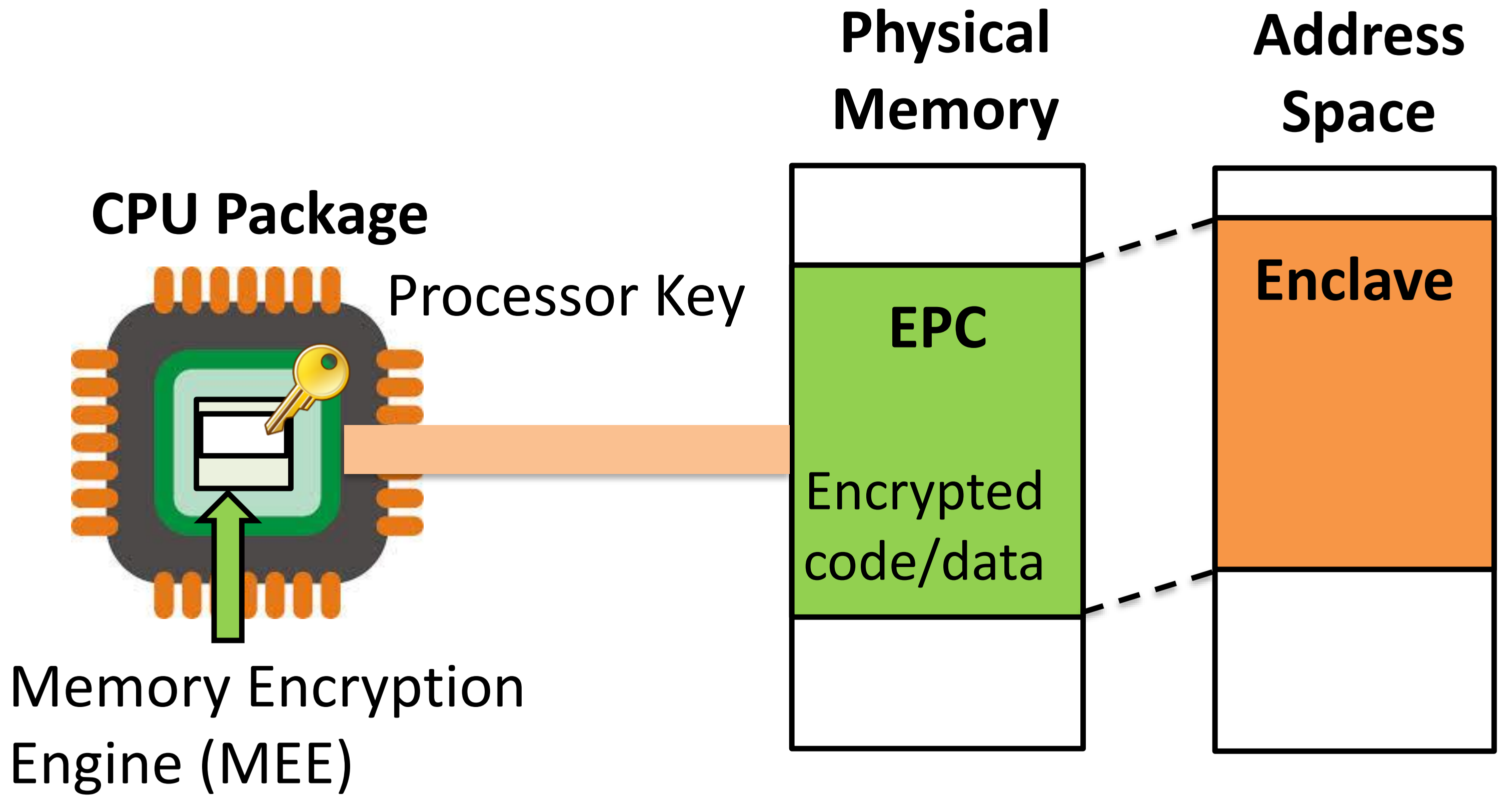
Intel SGX 101: Isolated Execution

- Smallest attack surface by reducing TCB (App + processor)
- Protect app's secret from untrusted privilege software



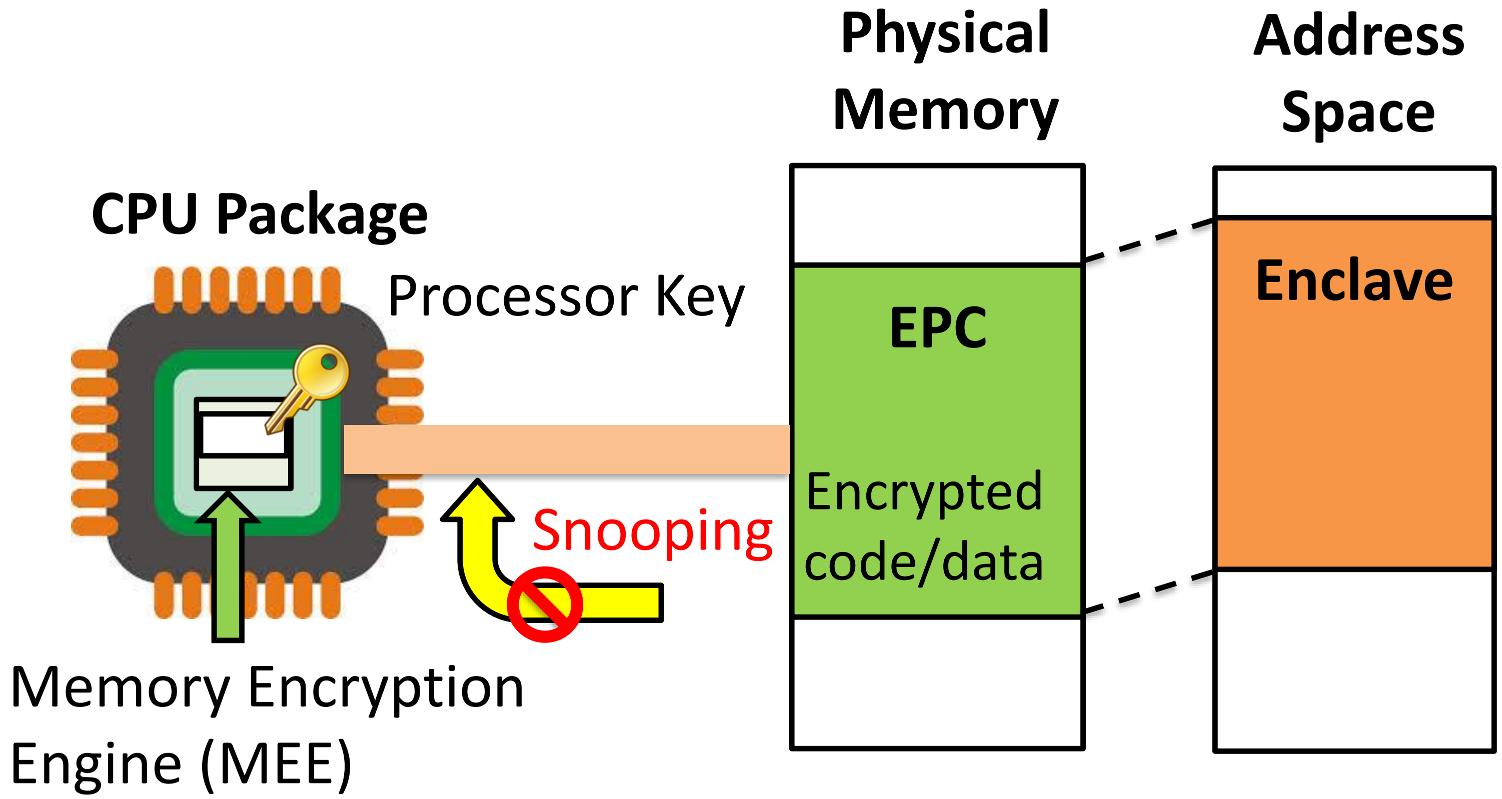
Intel SGX 101: Isolated Execution

- Smallest attack surface by reducing TCB (App + processor)
- Protect app’s secret from untrusted privilege software



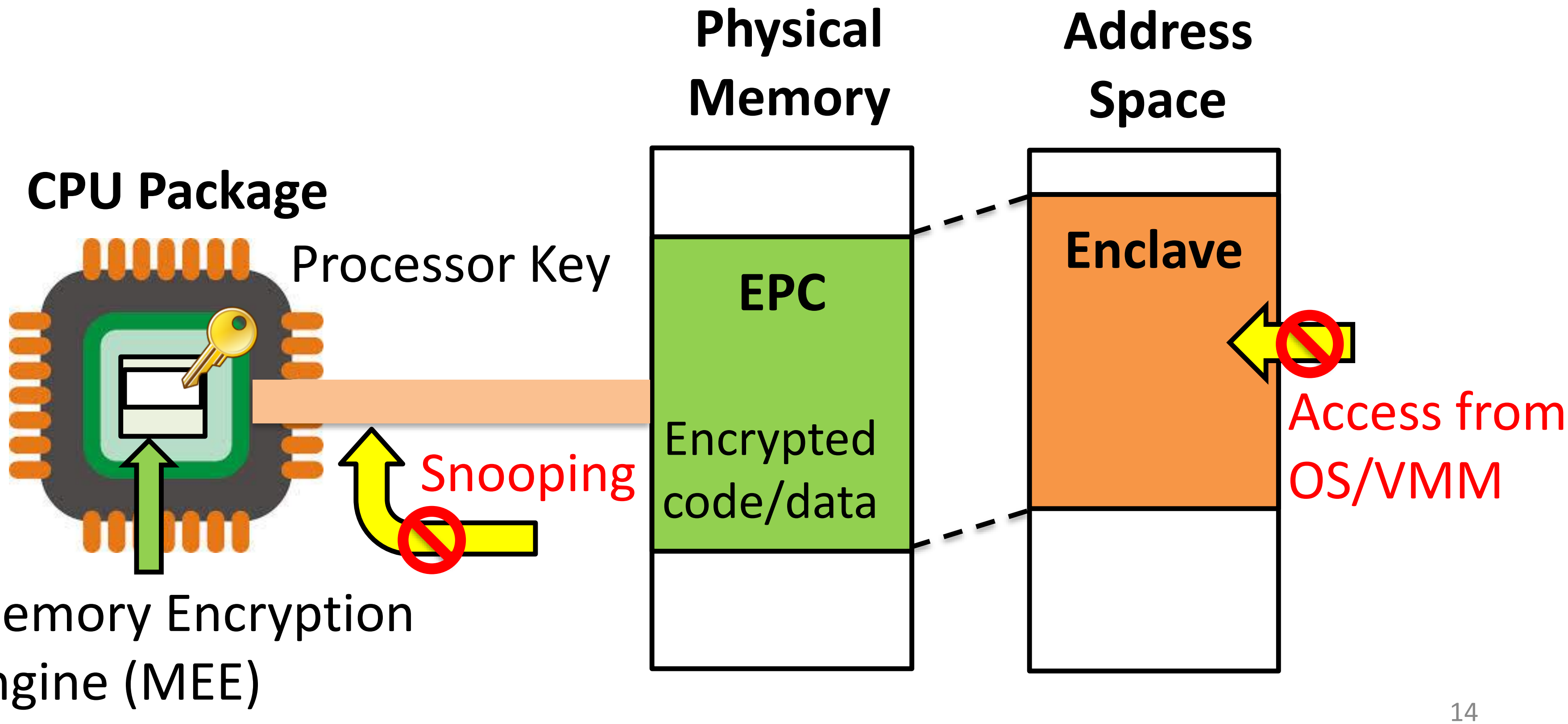
Intel SGX 101: Isolated Execution

- Smallest attack surface by reducing TCB (App + processor)
- Protect app's secret from untrusted privilege software



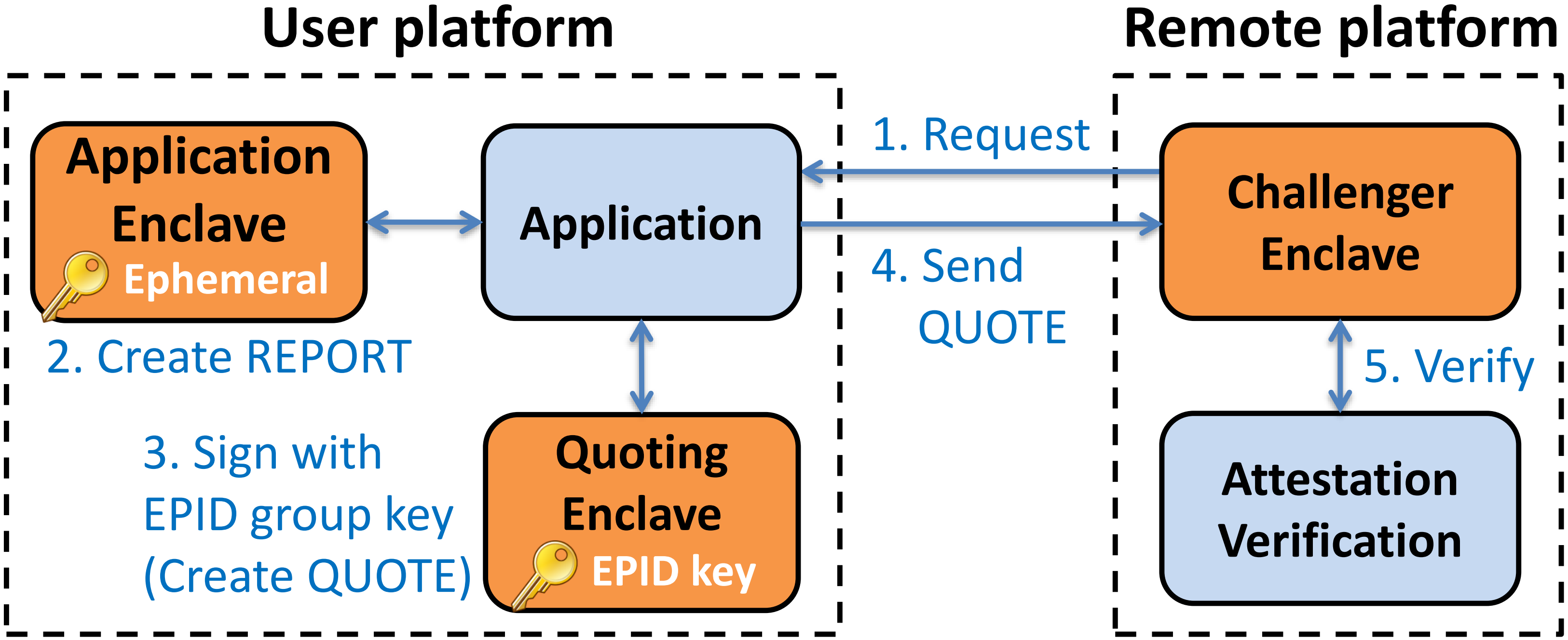
Intel SGX 101: Isolated Execution

- Smallest attack surface by reducing TCB (App + processor)
- Protect app’s secret from untrusted privilege software



Intel SGX 101: Remote attestation

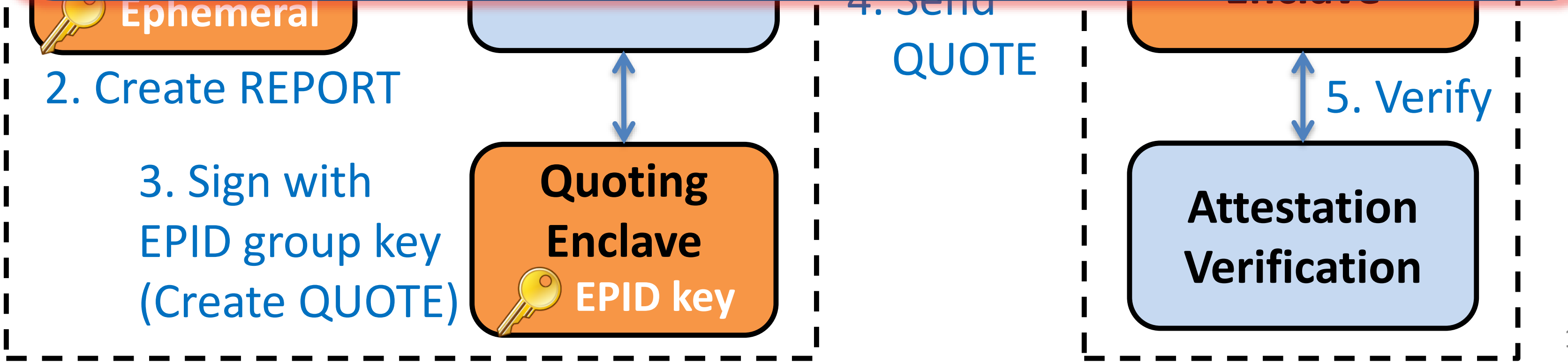
- Attest an application on remote platform
  - Check the **integrity of enclave** (hash of code/data pages)
  - Verify whether **enclave is running on real SGX CPU**
  - Can establish a "*secure channel*" between enclaves



Intel SGX 101: Remote attestation

- Attest an application on remote platform
  - Check the **integrity of enclave** (hash of code/data pages)
  - Verify whether **enclave is running on real SGX CPU**
  - Can establish a "*secure channel*" between enclaves

Intel SGX brings new opportunities for enhancing security of applications



SGX Research: Current Status

- Pioneering research: Adopting SGX on cloud computing (Haven [OSDI14], VC3 [S&P15])
- Confidentiality verification of SGX program (Moat [CCS15])
- Adopts SGX on networking [HotNets15]

SGX Research: Current Status

- However, software technologies for SGX lag behind their hardware counterpart

SGX CPU and SDK is now available! But..

- Specification for SGX [revision 1 & 2] is not fully available on the SGX hardware (only functionalities in revision 1)
- SGX technology has a complex license model

OpenSGX: Design Goal

- Offers a complete platform for SGX research
  - To explore software and hardware design space of SGX
  - To develop and evaluate SGX-enabled applications

OpenSGX: Design Goal

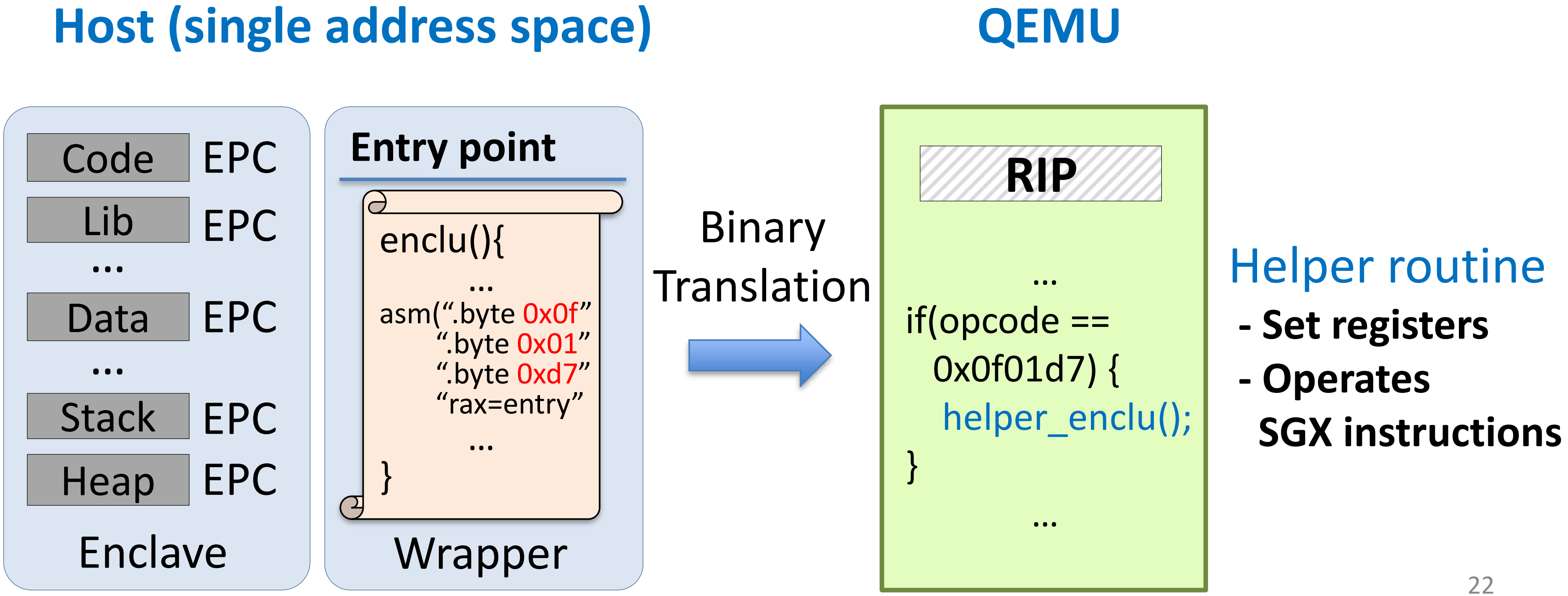
- Offers a complete platform for SGX research
  - To explore software and hardware design space of SGX
  - To develop and evaluate SGX-enabled applications
- Fills non-trivial issues on SGX software components
  - Support for system software and user-level APIs
  - Familiar programming model and interface
  - Secure design to defend against potential attack vectors (e.g., Iago attacks)

OpenSGX: Design Goal

- Offers a complete platform for SGX research
  - To explore software and hardware design space of SGX
  - To develop and evaluate SGX-enabled applications
- Fills non-trivial issues on SGX software components
  - Support for system software and user-level APIs
  - Familiar programming model and interface
  - Secure design to defend against potential attack vectors (e.g., Iago attacks)
- Non goal : security guarantee

OpenSGX: Approach

- Using userspace emulation of QEMU
  - Binary translation to support SGX instructions
  - QEMU helper routine to implement complex instructions

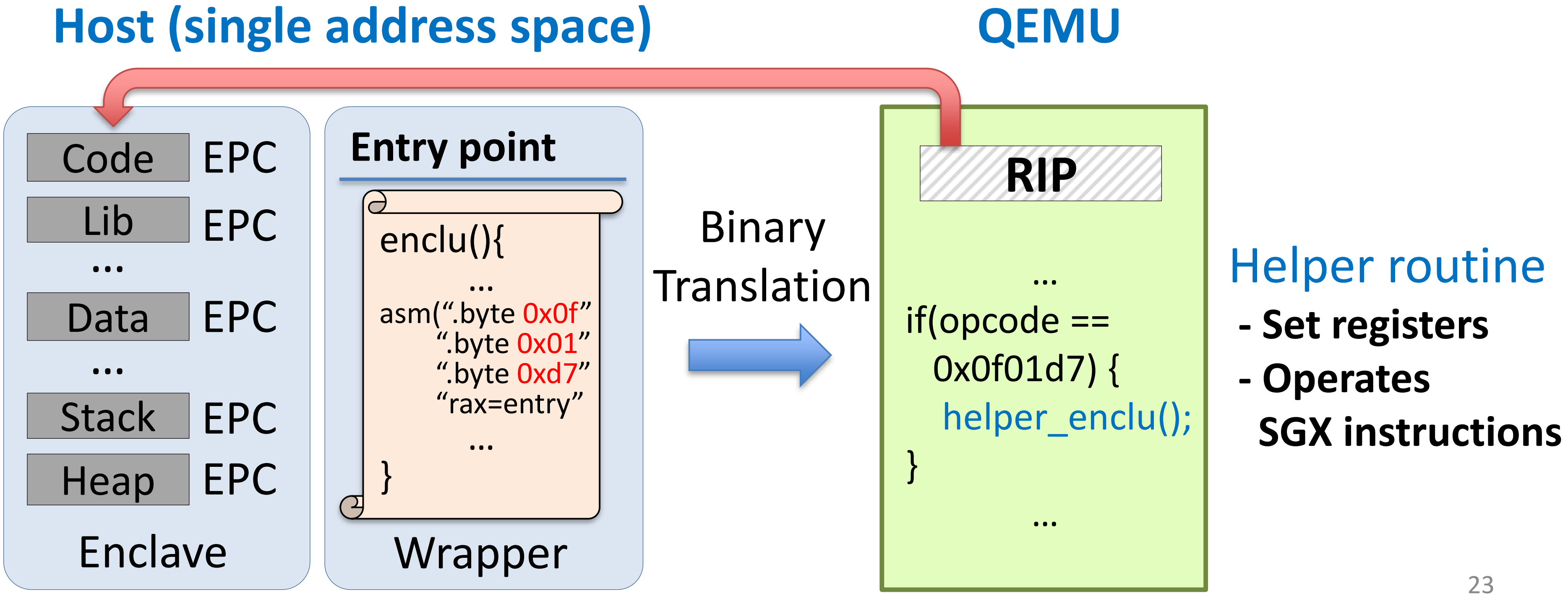


```
enclu(){
    …
asm(".byte 0x0f"
    ".byte 0x01"
    ".byte 0xd7"
    "rax=entry"
    …
}
```

```
…
if(opcode ==
  0x0f01d7) {
    helper_enclu();
}
…
```

OpenSGX: Approach

- Using userspace emulation of QEMU
  - Binary translation to support SGX instructions
  - QEMU helper routine to implement complex instructions

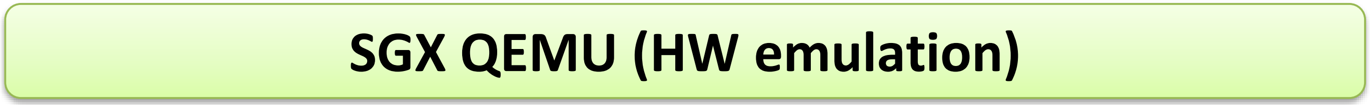


```
enclu(){
        …
asm(".byte 0x0f"
      ".byte 0x01"
      ".byte 0xd7"
      "rax=entry"
        …
}
```

```
        …
if(opcode ==
   0x0f01d7) {
    helper_enclu();
}
        …
```

OpenSGX: Component Overview

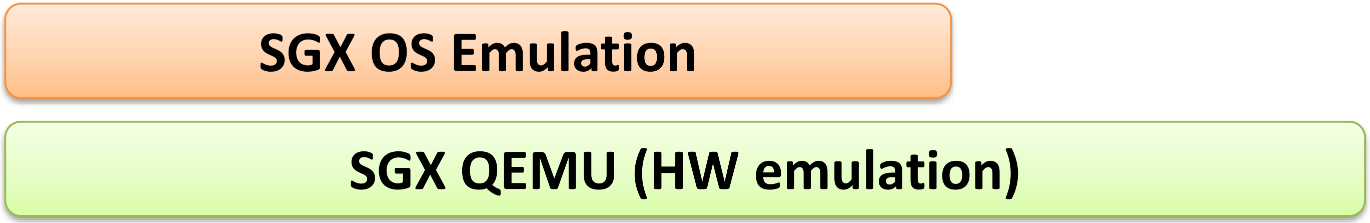
- Emulated SGX hardware



SGX QEMU (HW emulation)

OpenSGX: Component Overview

- Emulated SGX hardware
- OS emulation layer

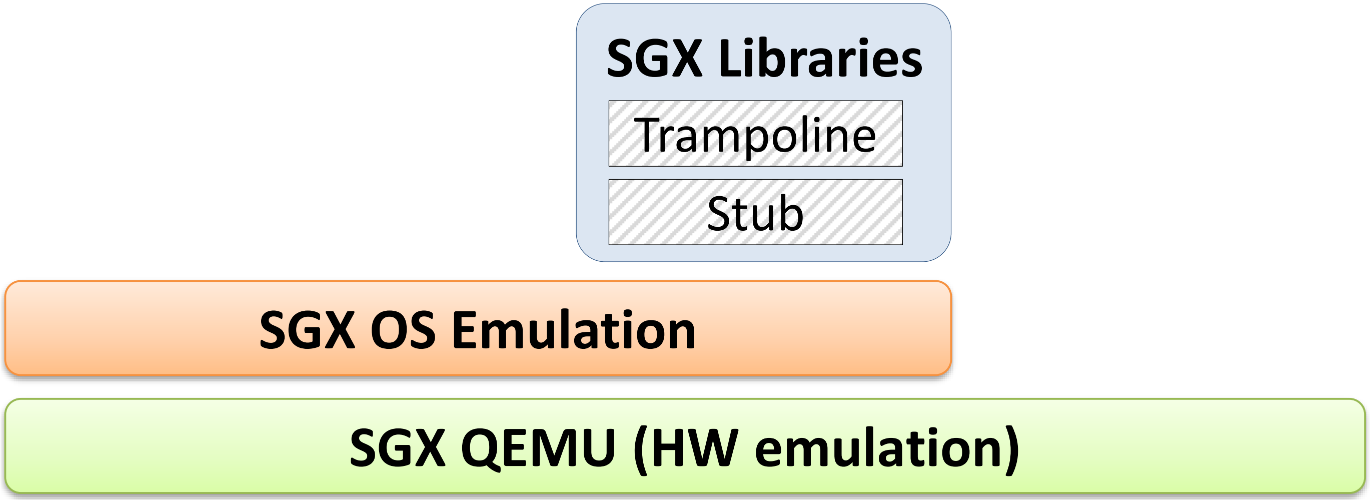


SGX OS Emulation

SGX QEMU (HW emulation)

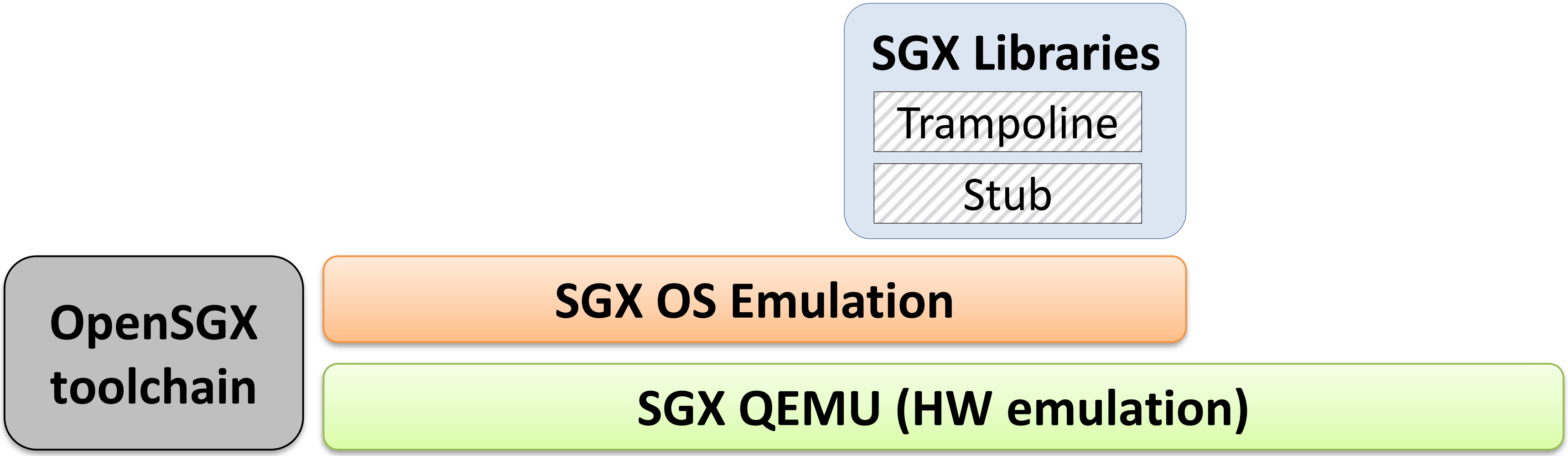
OpenSGX: Component Overview

- Emulated SGX hardware
- OS emulation layer
- OpenSGX user library



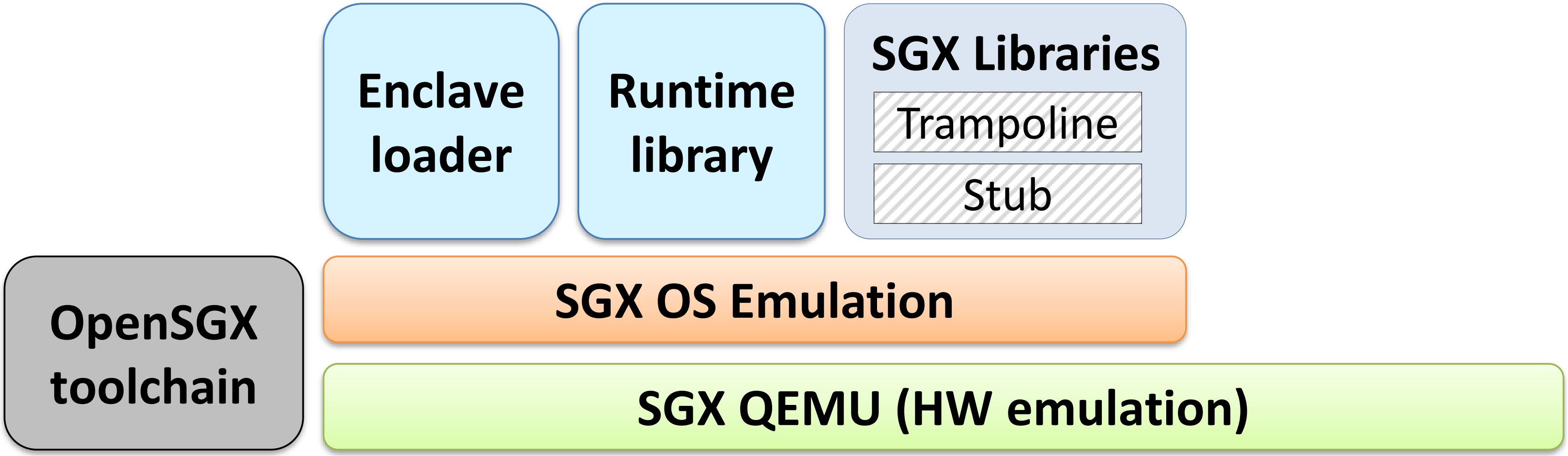
OpenSGX: Component Overview

- Emulated SGX hardware
- OS emulation layer
- OpenSGX user library
- OpenSGX toolchain



OpenSGX: Component Overview

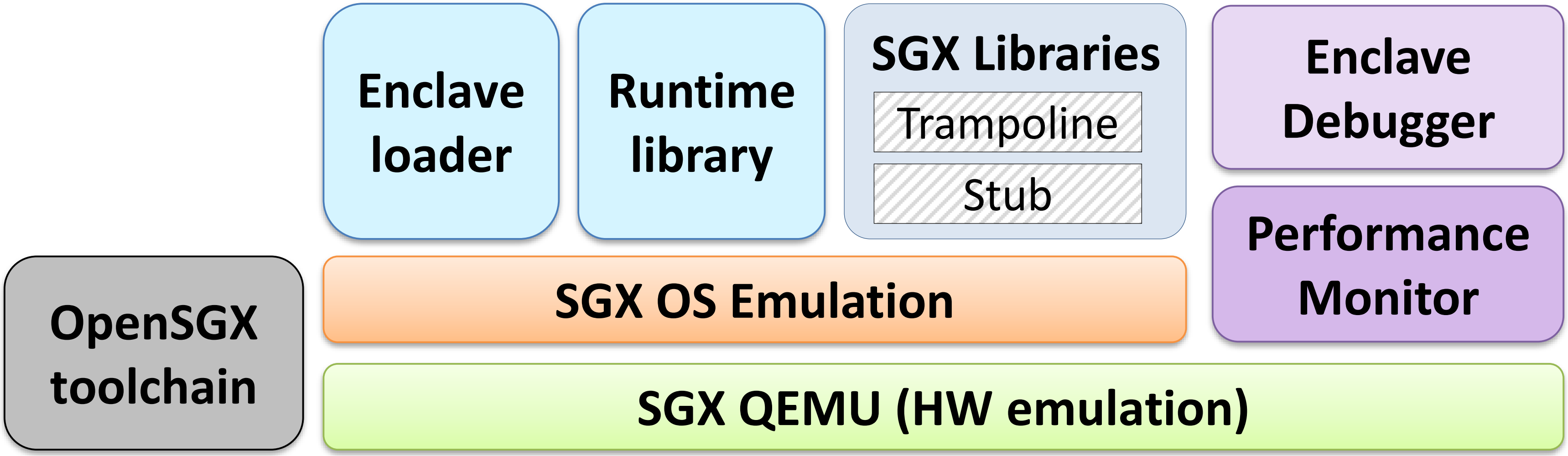
- Emulated SGX hardware
- OS emulation layer
- OpenSGX user library
- OpenSGX toolchain
- Enclave loader



OpenSGX: Component Overview

- Emulated SGX hardware
- OS emulation layer
- OpenSGX user library
- OpenSGX toolchain

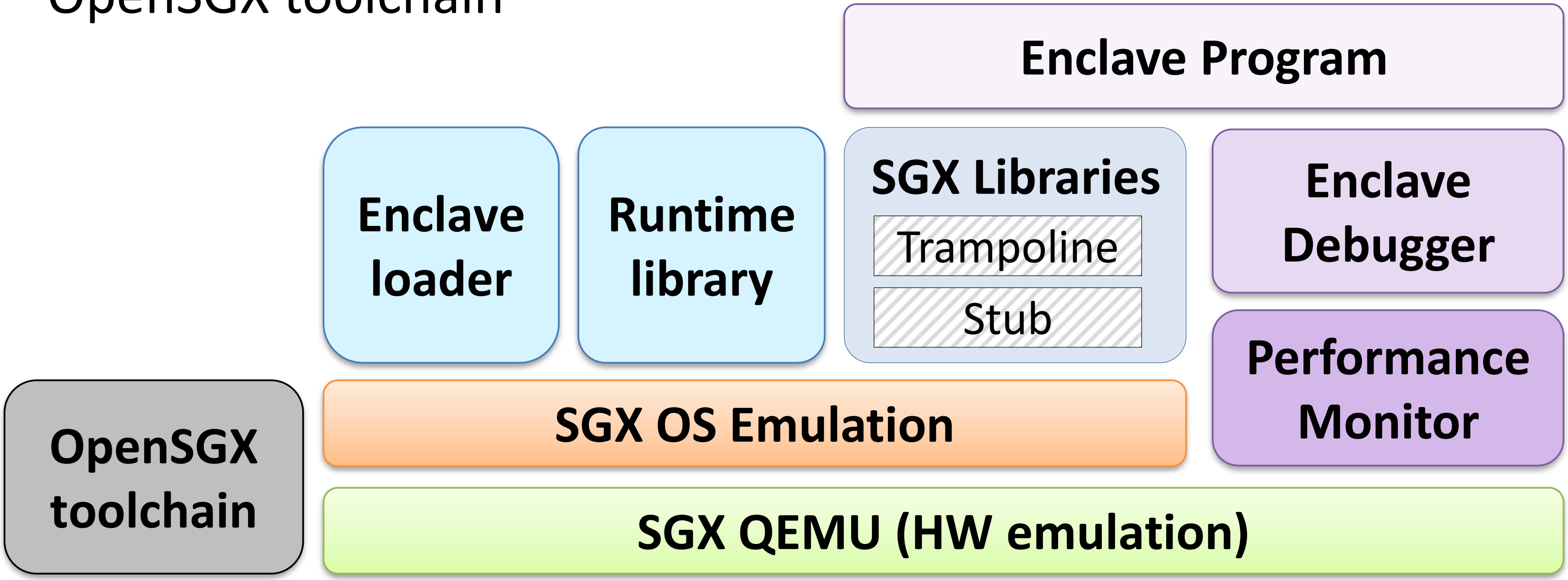
- Enclave loader
- Performance monitor
- Enclave debugger



OpenSGX: Component Overview

- Emulated SGX hardware
- OS emulation layer
- OpenSGX user library
- OpenSGX toolchain

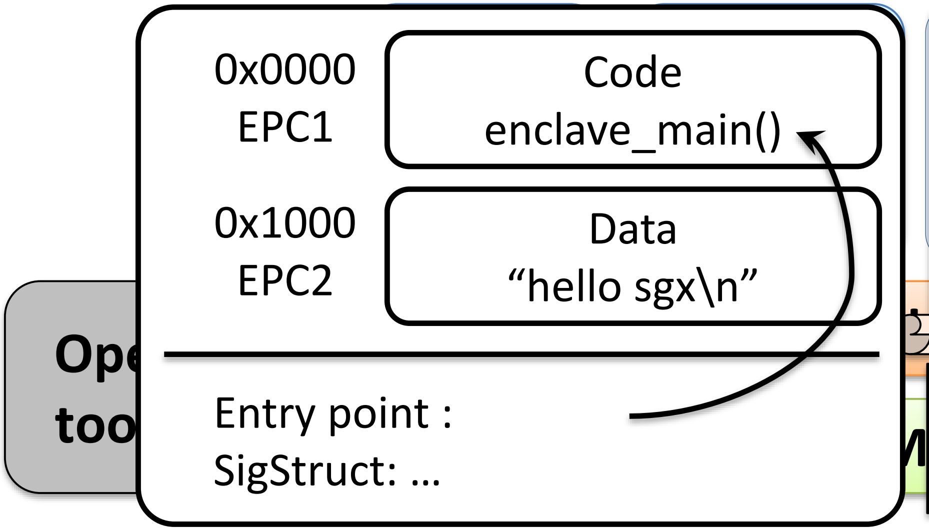
- Enclave loader
- Performance monitor
- Enclave debugger



OpenSGX: Component Overview

- Emulated SGX hardware
- OS emulation layer
- OpenSGX user library
- OpenSGX toolchain

- Enclave loader
- Performance monitor
- Enclave debugger



Enclave Program

```
void enclave_main(){
 char *hello = “hello sgx!\n”;
 sgx_enclave_wriate(hello, strlen(hello));
 sgx_exit(NULL);
}
```

```
$ opensgx hello.sgx hello.conf
hello sgx!
```

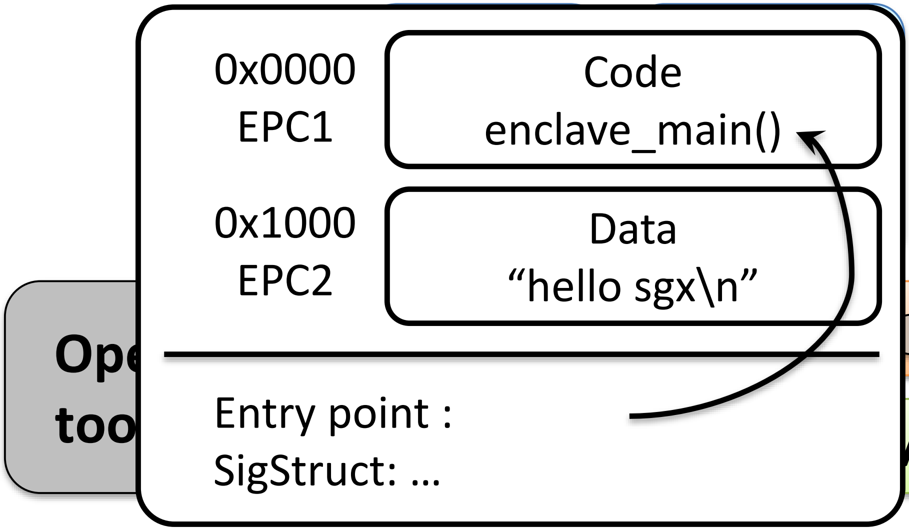
OpenSGX: Component Overview

- Emulated SGX hardware ✓
- OS emulation layer ✓
- OpenSGX user library ✓
- OpenSGX toolchain

- Enclave loader
- Performance monitor
- Enclave debugger

Enclave Program

```
void enclave_main(){
  char *hello = "hello sgx!\n";
  sgx_enclave_wriate(hello, strlen(hello));
  sgx_exit(NULL);
}
```



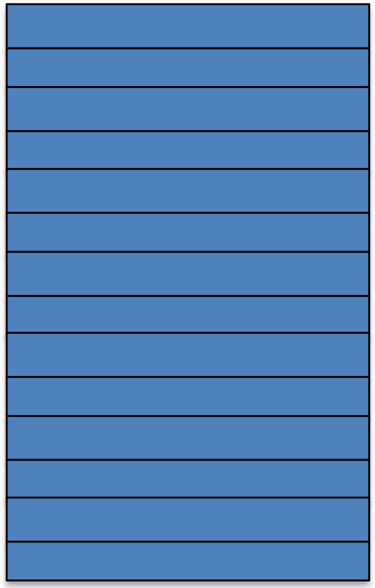
```
$ opensgx hello.sgx hello.conf
hello sgx!
```

Hardware Emulation

- Emulates all data structures(e.g., EPCM) and processor key
- EPC Memory management
  - Direct mapping on virtual memory
  - Access protection: Instrument memory access

Hardware Emulation

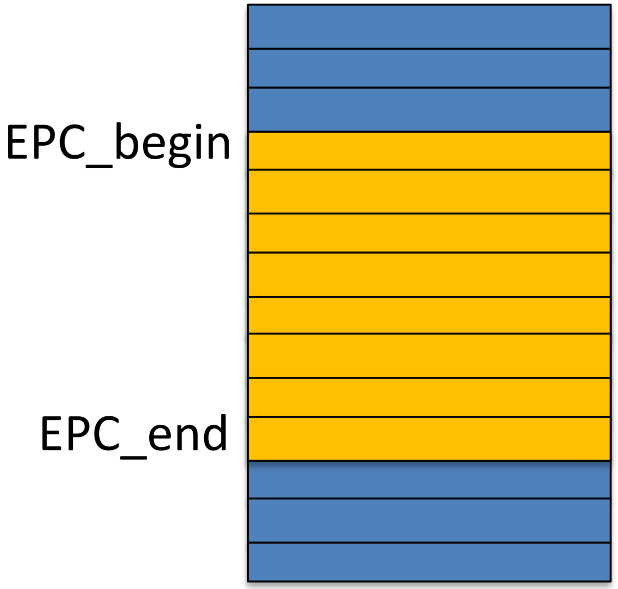
- Emulates all data structures(e.g., EPCM) and processor key
- EPC Memory management
  - Direct mapping on virtual memory
  - Access protection: Instrument memory access



Virtual address space

Hardware Emulation

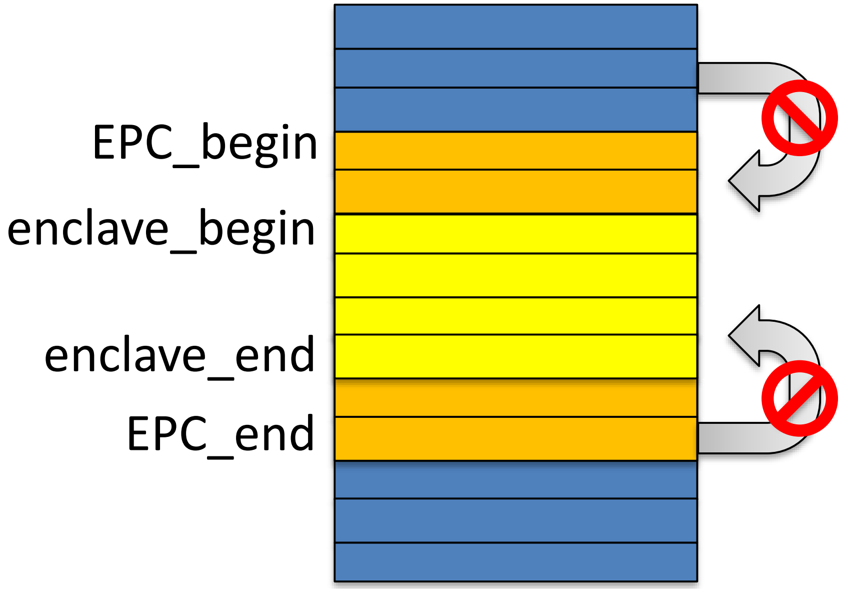
- Emulates all data structures(e.g., EPCM) and processor key
- EPC Memory management
  - – Direct mapping on virtual memory
  - – Access protection: Instrument memory access



Virtual address space

Hardware Emulation

- Emulates all data structures(e.g., EPCM) and processor key
- EPC Memory management
  - Direct mapping on virtual memory
  - Access protection: Instrument memory access



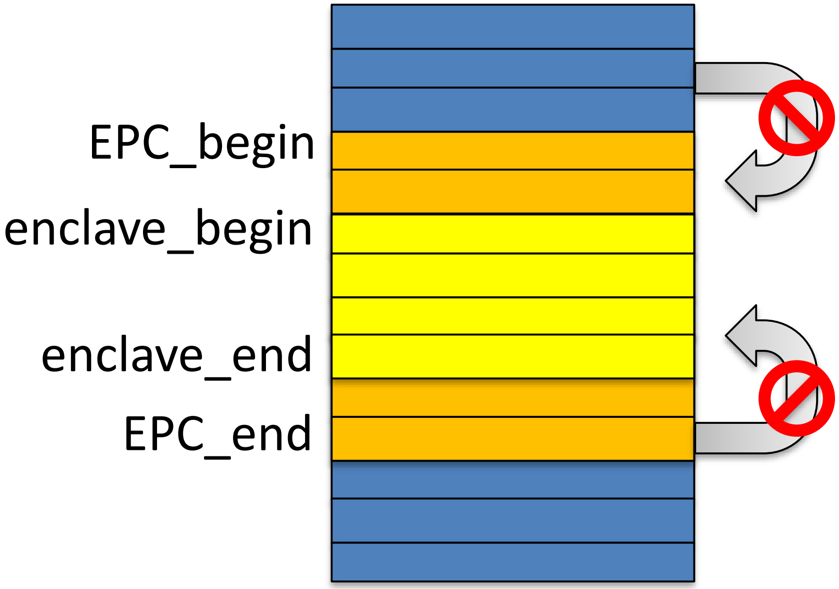
Virtual address space

1. Prohibit access from host to EPC

2. Prohibit others enclaves' EPC to current enclave's EPC

Hardware Emulation

- Emulates all data structures(e.g., EPCM) and processor key
- EPC Memory management
  - Direct mapping on virtual memory
  - Access protection: Instrument memory access



Virtual address space

```
...
Case (Load | Store) {
    1. Prohibit access
          from host to EPC
    2. Prohibit others enclaves'
        EPC to current enclave's EPC
}
...
```

QEMU's translation routine

Instruction Support

- OpenSGX supports most instructions specified
  - 21 out of 24 instructions
  - Except for debugging related instructions (e.g., EDBGRD)
  - Instead, it offers rich environment for debugging since it is a "**software emulator**" (e.g., GDB stub)

Instruction Support

- OpenSGX supports most instructions specified
  - 21 out of 24 instructions
  - Except for debugging related instructions (e.g., EDBGRD)
  - Instead, it offers rich environment for debugging since it is a "**software emulator**" (e.g., GDB stub)

- Provides simple C APIs which wraps assembly code
  - User-level instructions (ENCLU) : accessible to user-level APIs
  - Super-level instructions (ENCLS) : Requires system support

OS Emulation Layer

- Emulate OS to execute the privileged SGX instructions
  - Bootstrapping (EPC allocation)
  - Enclave initialization & page translation
  - Dynamic EPC page allocation

System call	Description
sys_sgx_init()	Allocate EPC memory region
sys_init_enclave()	Create an enclave, Add and measure EPC pages
sys_add_epc()	Allocates a new EPC page to the running enclave
sys_stat_enclave()	Obtains the enclave statistics

OS Emulation Layer

- Emulate OS to execute the privileged SGX instructions
  - Bootstrapping (EPC allocation)
  - Enclave initialization & page translation
  - Dynamic EPC page allocation

Planning to extend the emulated OS for the system-level layer

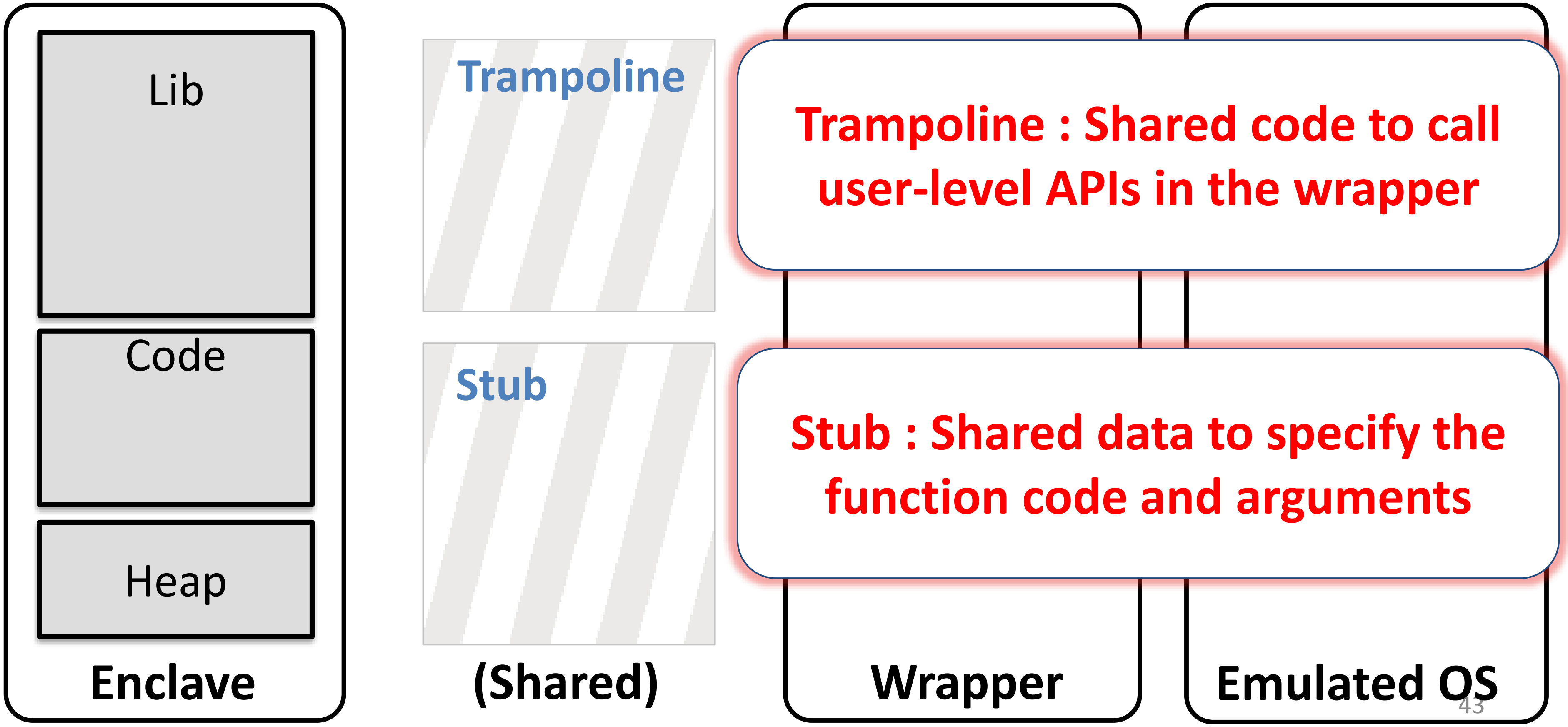
sys_init_enclave()	Create an enclave, Add and measure EPC pages
sys_add_epc()	Allocates a new EPC page to the running enclave
sys_stat_enclave()	Obtains the enclave statistics

Stub and Trampoline Interface

"A strict and narrow interface to handle enclave-host communication using shared data/code"

Stub and Trampoline Interface

"A strict and narrow interface to handle enclave-host communication using shared data/code"



Lib

Code

Heap

Enclave

Trampoline

Stub

(Shared)

Wrapper

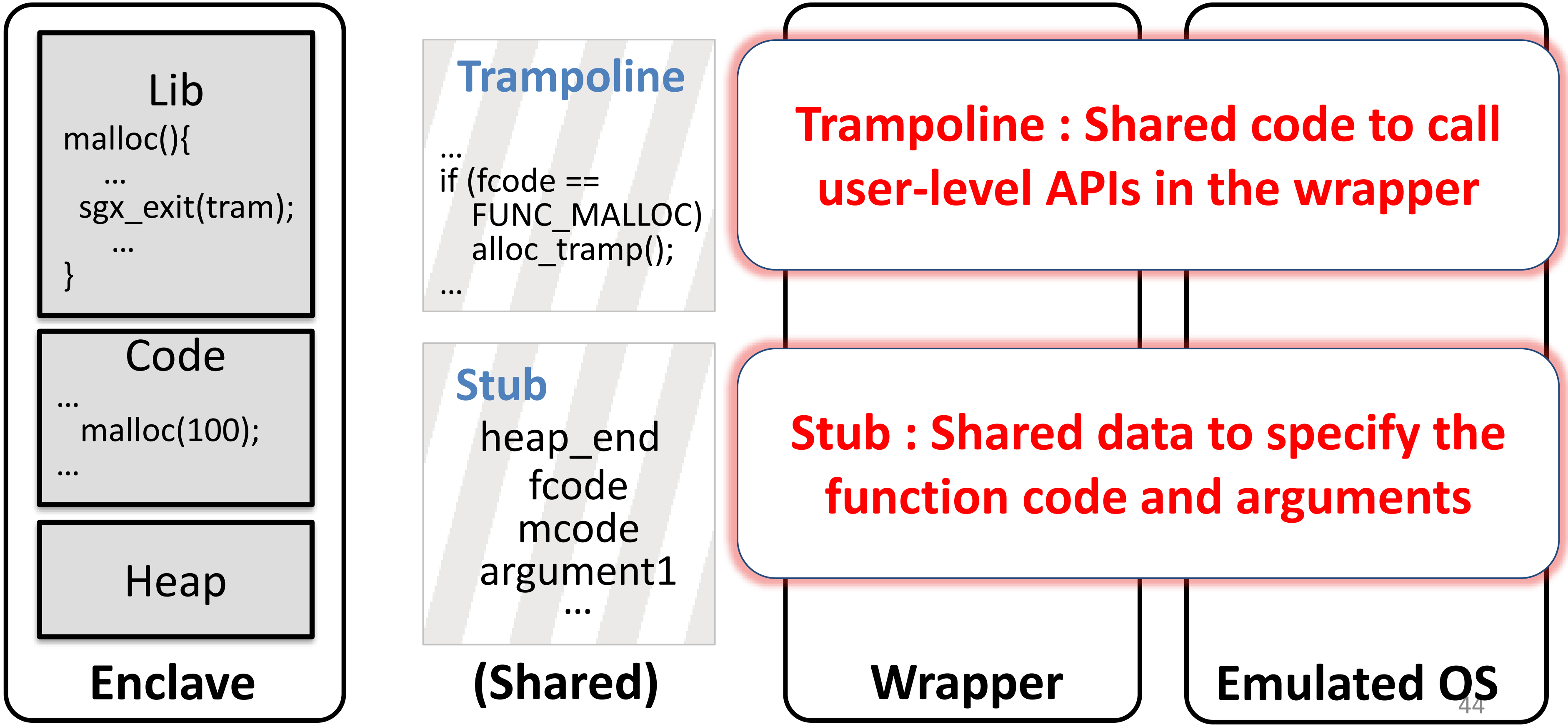
Emulated OS

Trampoline : Shared code to call user-level APIs in the wrapper

Stub : Shared data to specify the function code and arguments

Stub and Trampoline Interface

“A strict and narrow interface to handle enclave-host communication using shared data/code”



Lib

```
malloc(){
    …
  sgx_exit(tram);
     …
}
```

Code

```
…
  malloc(100);
…
```

Heap

Enclave

Trampoline

```
…
if (fcode ==
   FUNC_MALLOC)
   alloc_tramp();
…
```

Stub

heap_end
fcode
mcode
argument1
…

(Shared)

Trampoline : Shared code to call user-level APIs in the wrapper

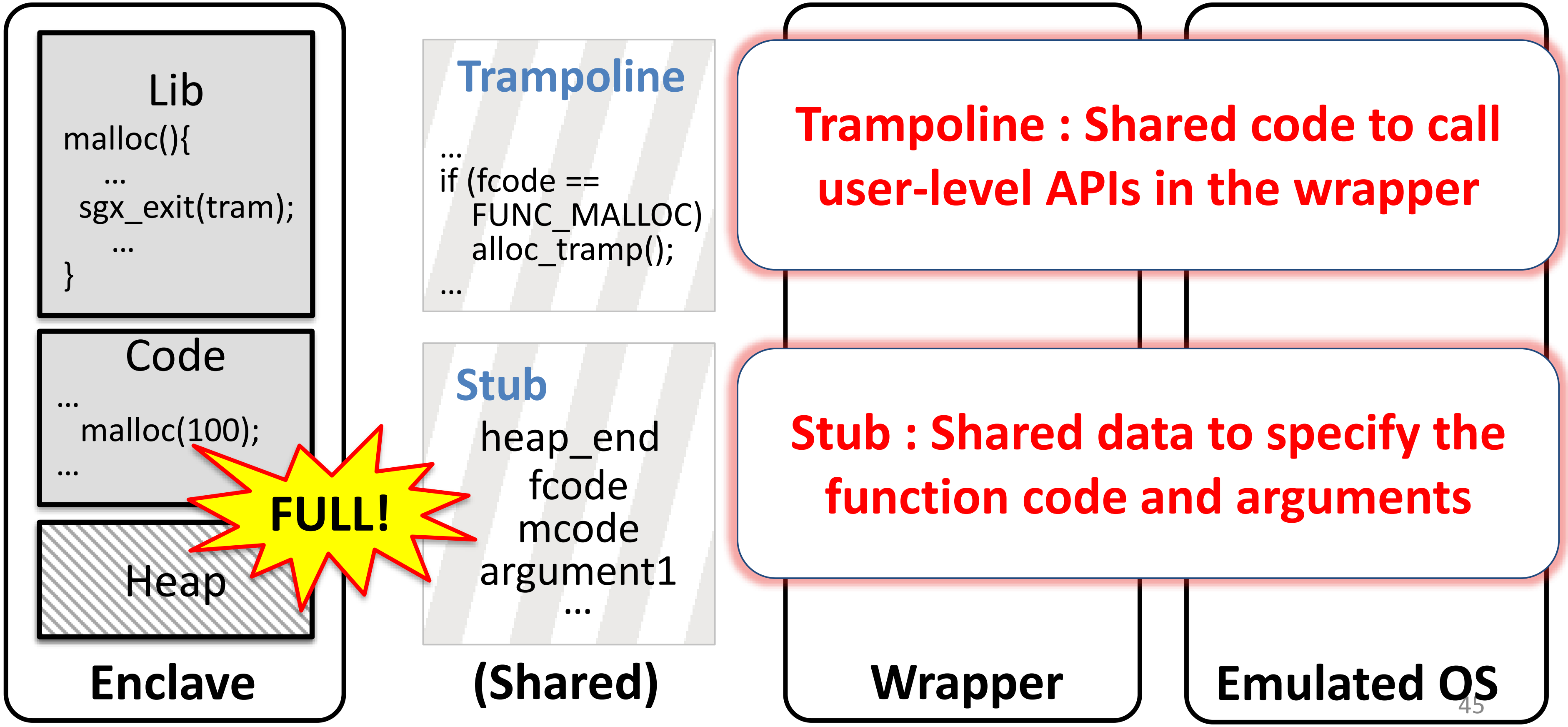
Stub : Shared data to specify the function code and arguments

Wrapper

Emulated OS

Stub and Trampoline Interface

> “A strict and narrow interface to handle enclave-host communication using shared data/code”



```
malloc(){
   …
  sgx_exit(tram);
   …
}
```

```
…
  malloc(100);
…
```

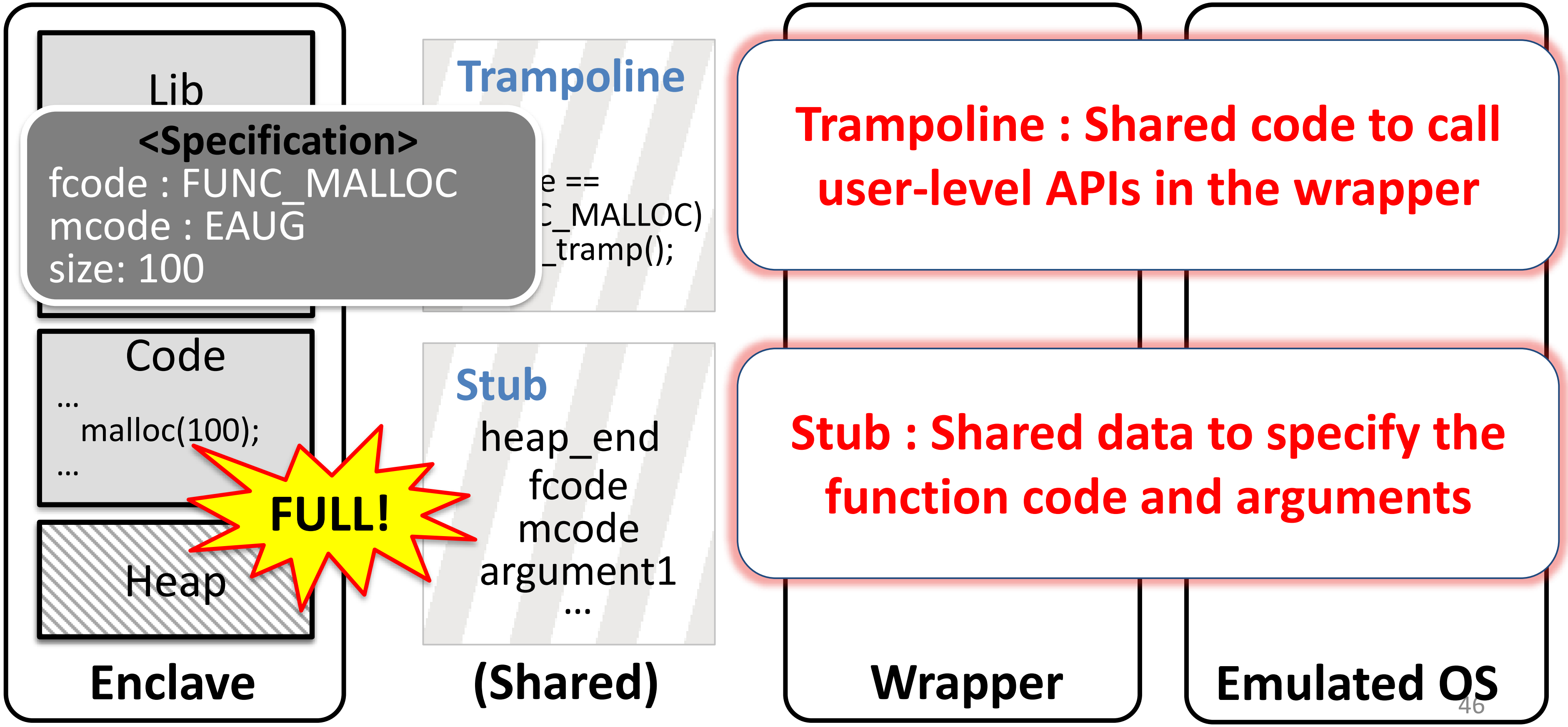
```
…
if (fcode ==
   FUNC_MALLOC)
   alloc_tramp();
…
```

Trampoline : Shared code to call user-level APIs in the wrapper

Stub : Shared data to specify the function code and arguments

Stub and Trampoline Interface

“A strict and narrow interface to handle enclave-host communication using shared data/code”

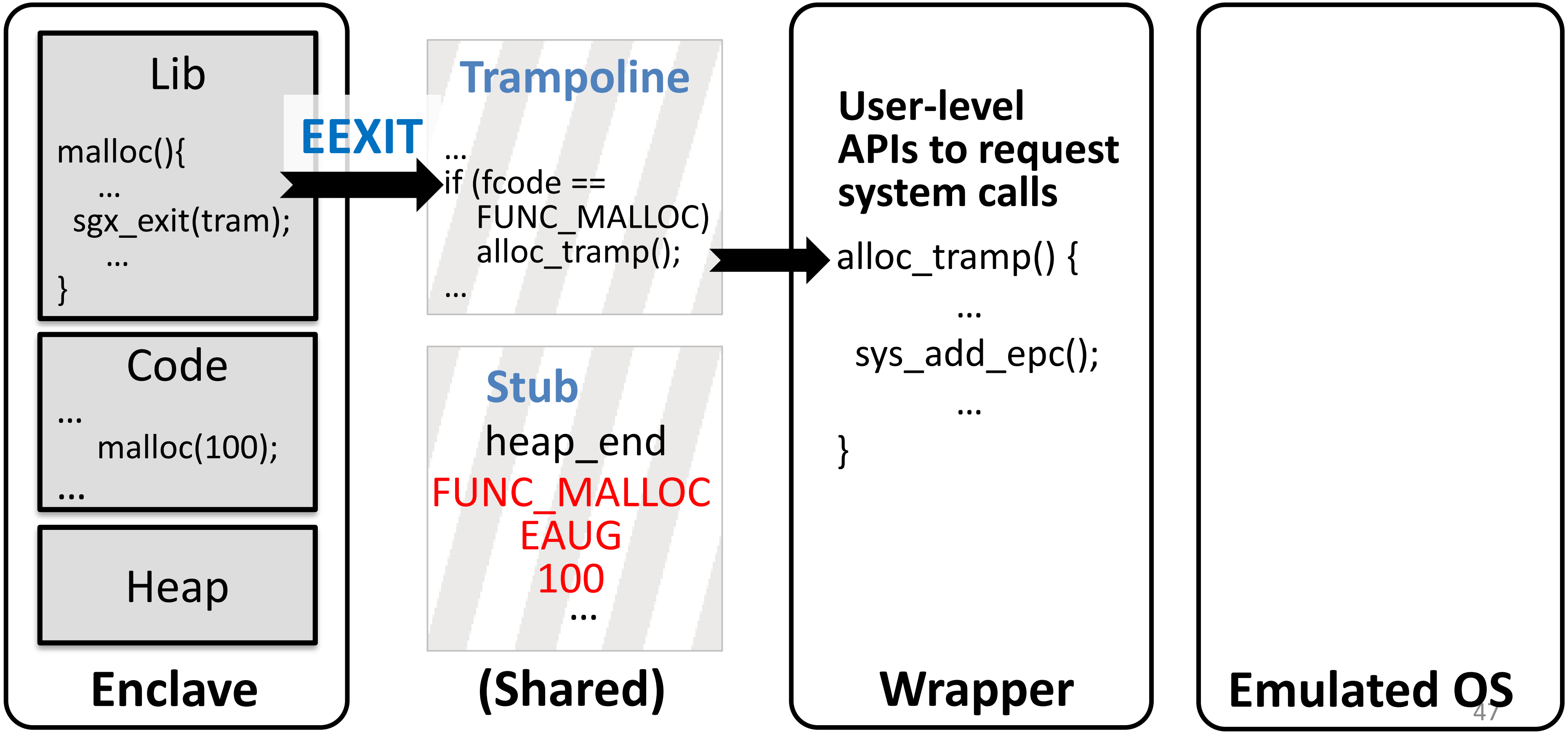


Trampoline : Shared code to call user-level APIs in the wrapper

Stub : Shared data to specify the function code and arguments

Trampoline and Stub Interface

> “A strict and narrow interface to handle enclave-host communication using shared data/code”



Enclave

Lib

```
malloc(){
    …
  sgx_exit(tram);
    …
}
```

Code

```
…
   malloc(100);
…
```

Heap

EEXIT

(Shared)

Trampoline

```
…
if (fcode ==
    FUNC_MALLOC)
    alloc_tramp();
…
```

Stub

heap_end

FUNC_MALLOC

EAUG

100

…

Wrapper

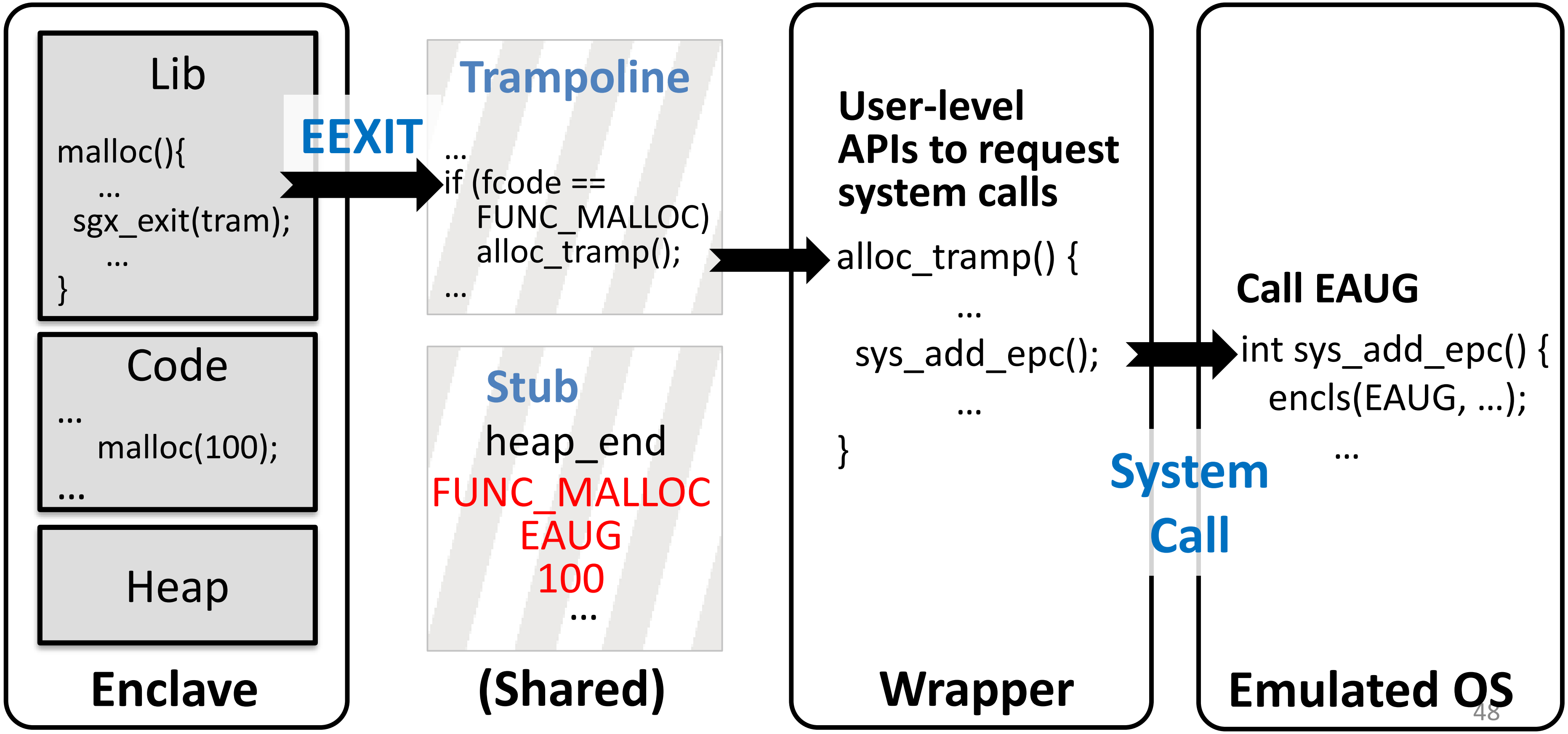
User-level APIs to request system calls

```
alloc_tramp() {
        …
  sys_add_epc();
        …
}
```

Emulated OS

Trampoline and Stub Interface

"A strict and narrow interface to handle enclave-host communication using shared data/code"



```
malloc(){
   …
 sgx_exit(tram);
    …
}
```

```
…
    malloc(100);
…
```

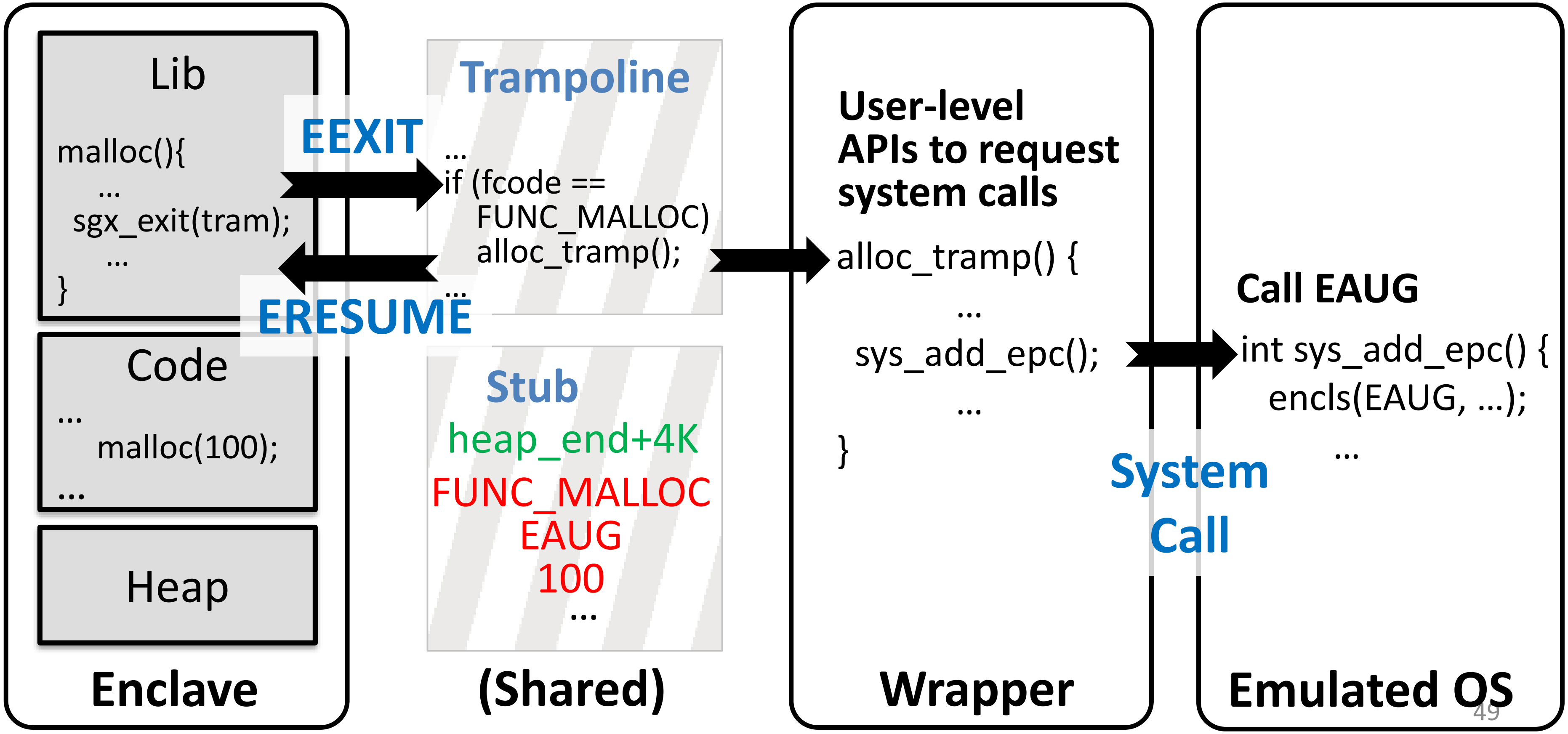
```
…
if (fcode ==
    FUNC_MALLOC)
    alloc_tramp();
…
```

```
alloc_tramp() {
        …
  sys_add_epc();
        …
}
```

```
int sys_add_epc() {
   encls(EAUG, …);
        …
```

Trampoline and Stub Interface

“A strict and narrow interface to handle enclave-host communication using shared data/code”



Lib

```
malloc(){
   …
 sgx_exit(tram);
    …
}
```

Code

```
…
   malloc(100);
…
```

Heap

Enclave

EEXIT

ERESUME

Trampoline

```
…
if (fcode ==
   FUNC_MALLOC)
   alloc_tramp();
…
```

Stub

heap_end+4K

FUNC_MALLOC

EAUG

100

…

(Shared)

User-level APIs to request system calls

```
alloc_tramp() {
        …
  sys_add_epc();
        …
}
```

Wrapper

System Call

Call EAUG

```
int sys_add_epc() {
   encls(EAUG, …);
        …
```

Emulated OS

Evaluation: Tor Network

- Redesigns non-trivial application to use OpenSGX
- Tor : volunteer-based anonymity network

Evaluation: Tor Network

- Redesigns non-trivial application to use OpenSGX
- Tor : volunteer-based anonymity network

“Defend possible attacks on Tor components when they are compromised by adversaries”

Evaluation: Tor Network

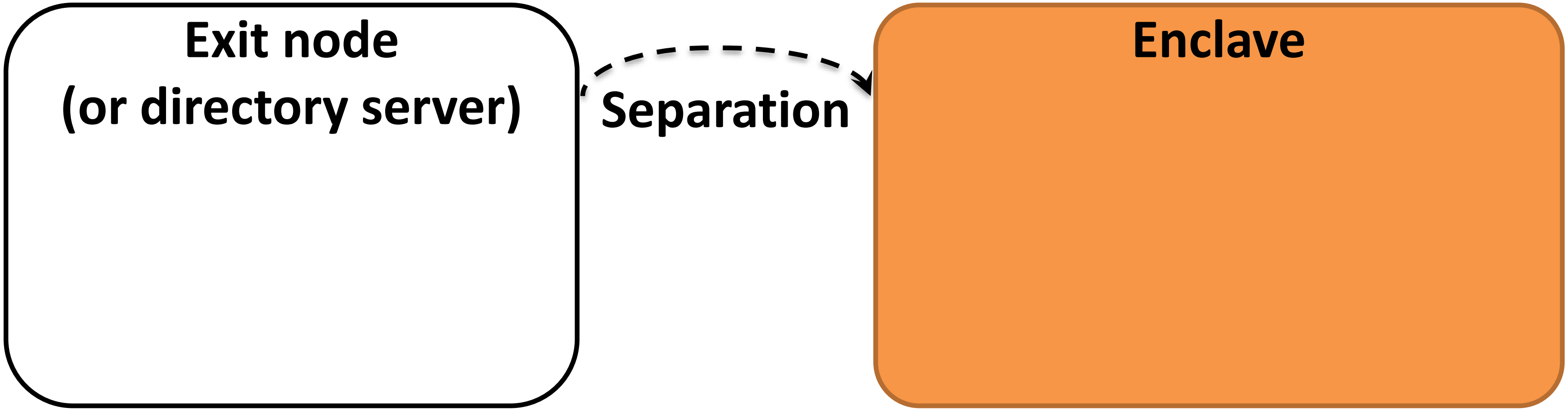
- Redesigns non-trivial application to use OpenSGX
- Tor : volunteer-based anonymity network

“Defend possible attacks on Tor components when they are compromised by adversaries”

- Here, defense against network-level attacks on Tor is out of scope

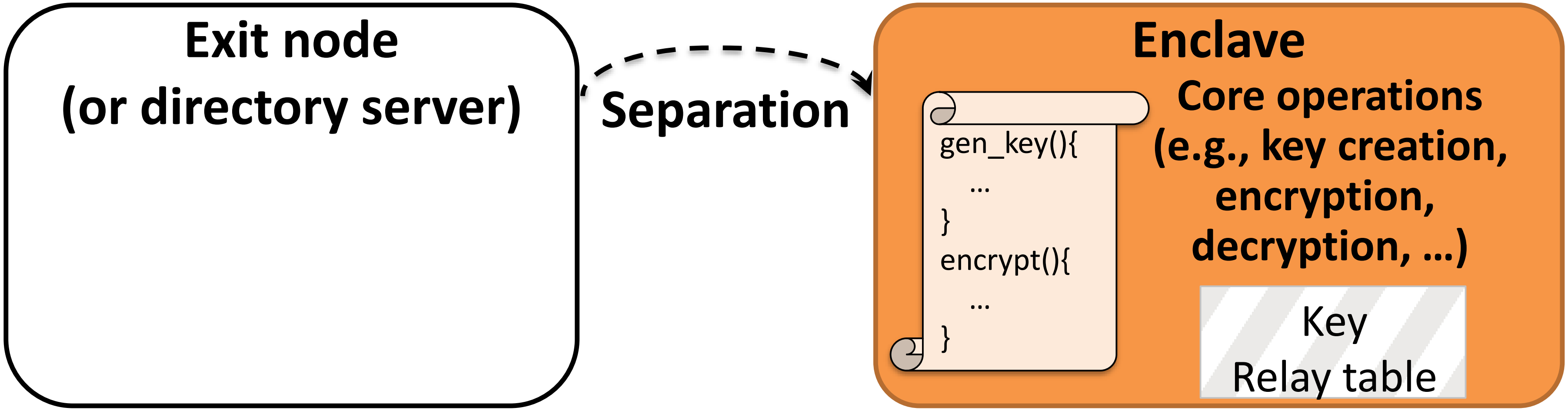
SGX-enabled Tor Design

- Design goal
  - **Protect data/code from adversary**
  - **Reducing Trusted Computing Base**



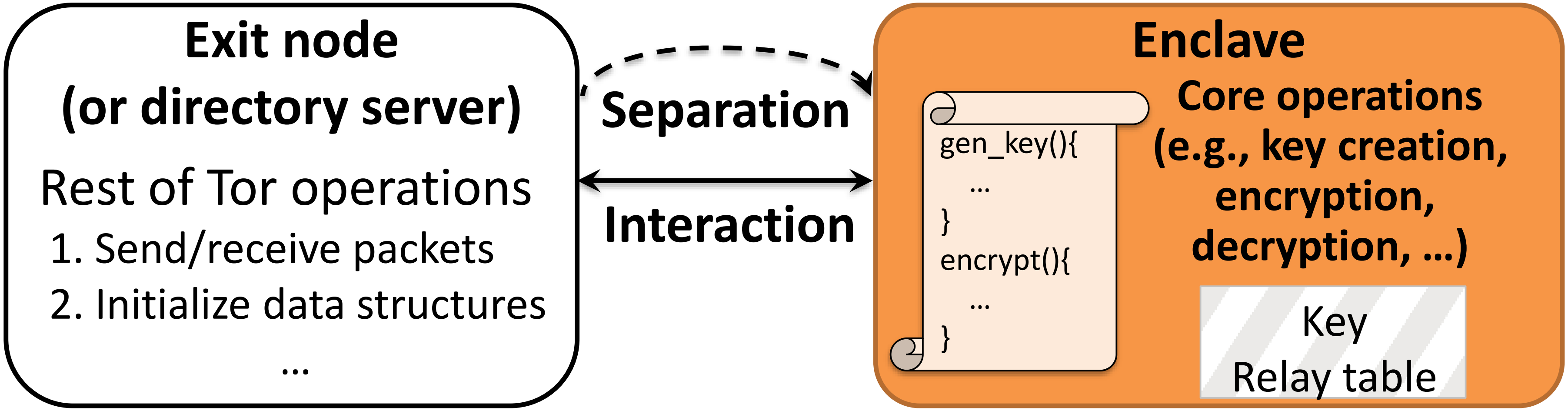
SGX-enabled Tor Design

- Design goal
  - **Protect data/code from adversary**
  - **Reducing Trusted Computing Base**



SGX-enabled Tor Design

- Design goal
  - **Protect data/code from adversary**
  - **Reducing Trusted Computing Base**



Exit node

(or directory server)

Rest of Tor operations

1. Send/receive packets
2. Initialize data structures

…

Separation

Interaction

Enclave

```
gen_key(){
   …
}
encrypt(){
   …
}
```

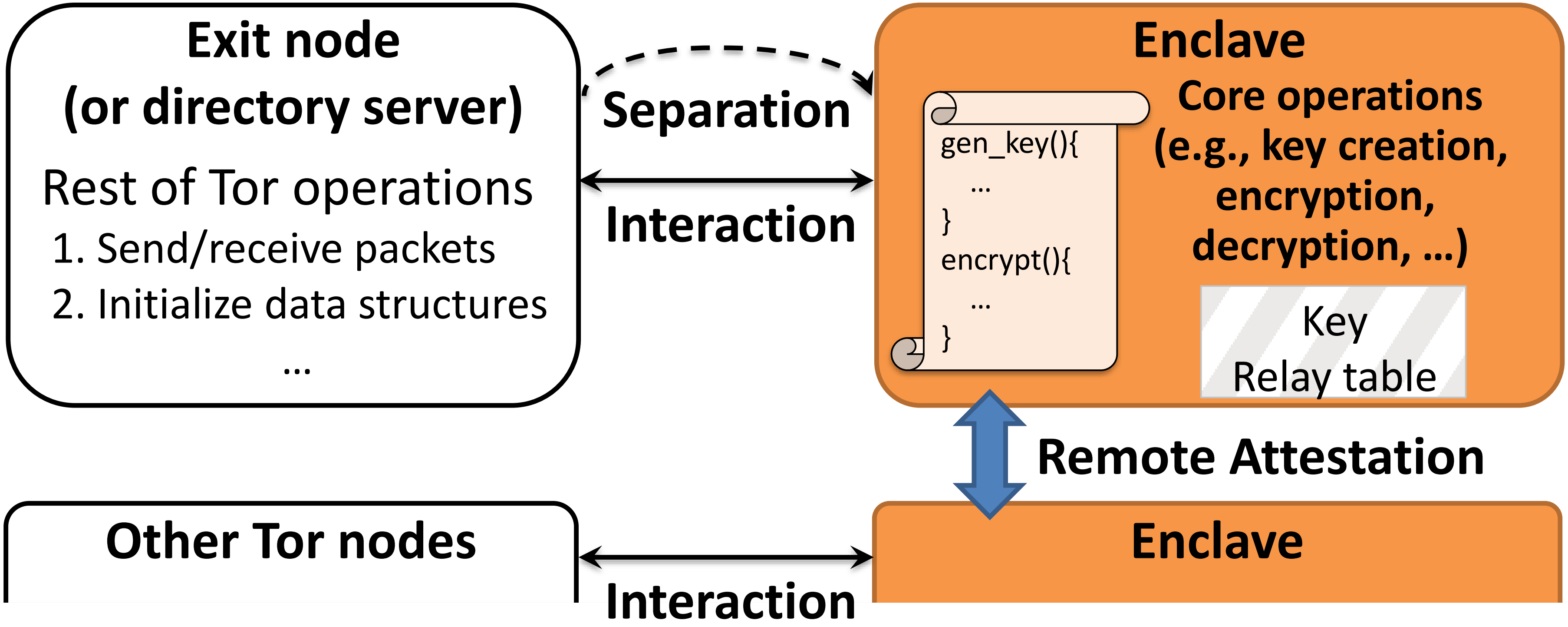
Core operations (e.g., key creation, encryption, decryption, …)

Key

Relay table

SGX-enabled Tor Design

- Design goal
  - **Protect data/code from adversary**
  - **Reducing Trusted Computing Base**



Exit node
(or directory server)

Rest of Tor operations

1. Send/receive packets
2. Initialize data structures

…

Separation

Interaction

Enclave

```
gen_key(){
    …
}
encrypt(){
    …
}
```

Core operations
(e.g., key creation,
encryption,
decryption, …)

Key
Relay table

Remote Attestation

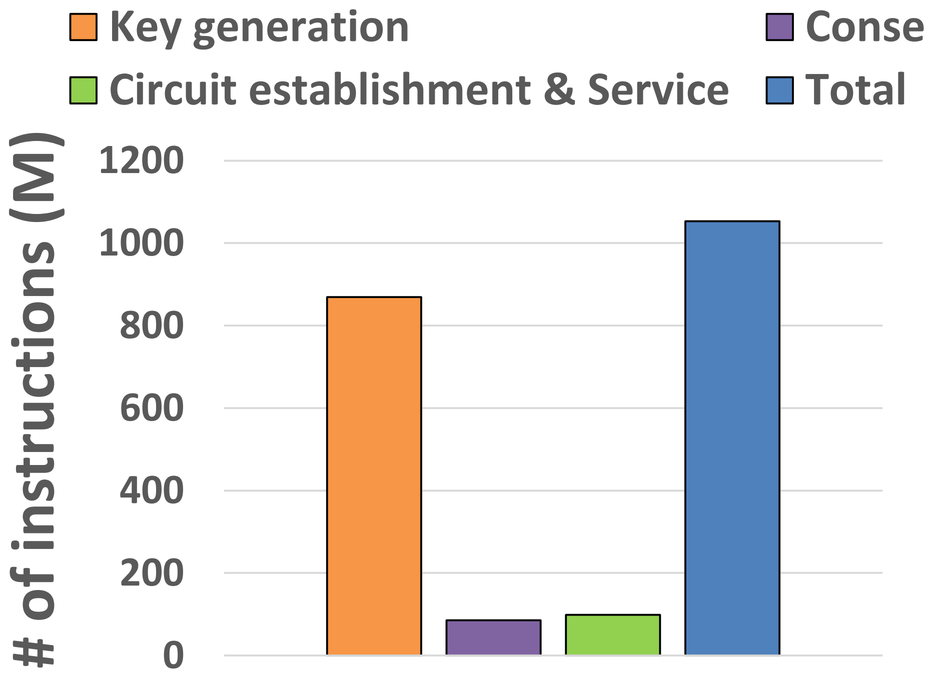
Other Tor nodes

Interaction

Enclave

Performance Profiling

- Performance profiling of Tor exit node
  - Using OpenSGX performance monitor



(Unit: Number of pages)

	Code	Data	Total
OpenSSL	271	89	360
SgxLib	3	1	4
Tor	4	1	5
Total	278	91	369

Required EPC : Less than 2MB

OpenSGX: Current Status

- Available at github, released in May 2015
  - Available in https://github.com/sslab-gatech/opensgx
  - 7 Contributors (Gatech, KAIST, Two sigma, MITRC, …)
  - 31 unique cloners, 1,645 Views (Until January, 2016)

- What's next?
  - Binary compatibility with Intel SGX hardware
  - Implement unsupported functionalities (e.g., multi-threading)

- Our current community

2σ TWO SIGMA

the programming language Lua

Our Early Lessons on SGX

- **Misconceptions on SGX**
  - SGX for desktop-like environment : Needs secure I/O channel (integration with hardware technology such as Intel IPT)
  - Need EPID support for the remote attestation

Our Early Lessons on SGX

- **Misconceptions on SGX**
  - SGX for desktop-like environment : Needs secure I/O channel (integration with hardware technology such as Intel IPT)
  - Need EPID support for the remote attestation

- **Malicious use of Intel SGX**
  - Malware might be possible by abusing the isolation property
  - Fails on traditional signature-based AV programs

Conclusion

- We design and implement OpenSGX, fully functional and instruction-compatible SGX emulator

Conclusion

- We design and implement OpenSGX, fully functional and instruction-compatible SGX emulator

- As a showcasing application, we develop SGX-enabled Tor to enhance the security and privacy

Conclusion

- We design and implement OpenSGX, fully functional and instruction-compatible SGX emulator

- As a showcasing application, we develop SGX-enabled Tor to enhance the security and privacy

- OpenSGX offers opportunities to explore all components of SGX research
  - Hardware semantics (e.g., encryption scheme of MEE)
  - System software, enclave loader and user-level APIs
  - Redesigning unforeseen security applications (e.g., Tor)

Thanks!
Any Questions?

SGX Threat Model

“An adversary has control over all software components (including OS and hypervisor) and hardware except the CPU package”

- Protection against denial-of-service is out of scope

Comparison: Intel SGX vs OpenSGX

	Intel SGX	OpenSGX
Type	Hardware	Software Emulator
Instructions	16 ENCLS, 8 ENCLU	13 ENCLS, 8 ENCLU (Except debugging)
Data structures	Specified	○
Paging	Page table	Direct mapping
System software	Not specified	User level emulation
User level APIs	SDK is available (Only for Windows)	○

OpenSGX User Library

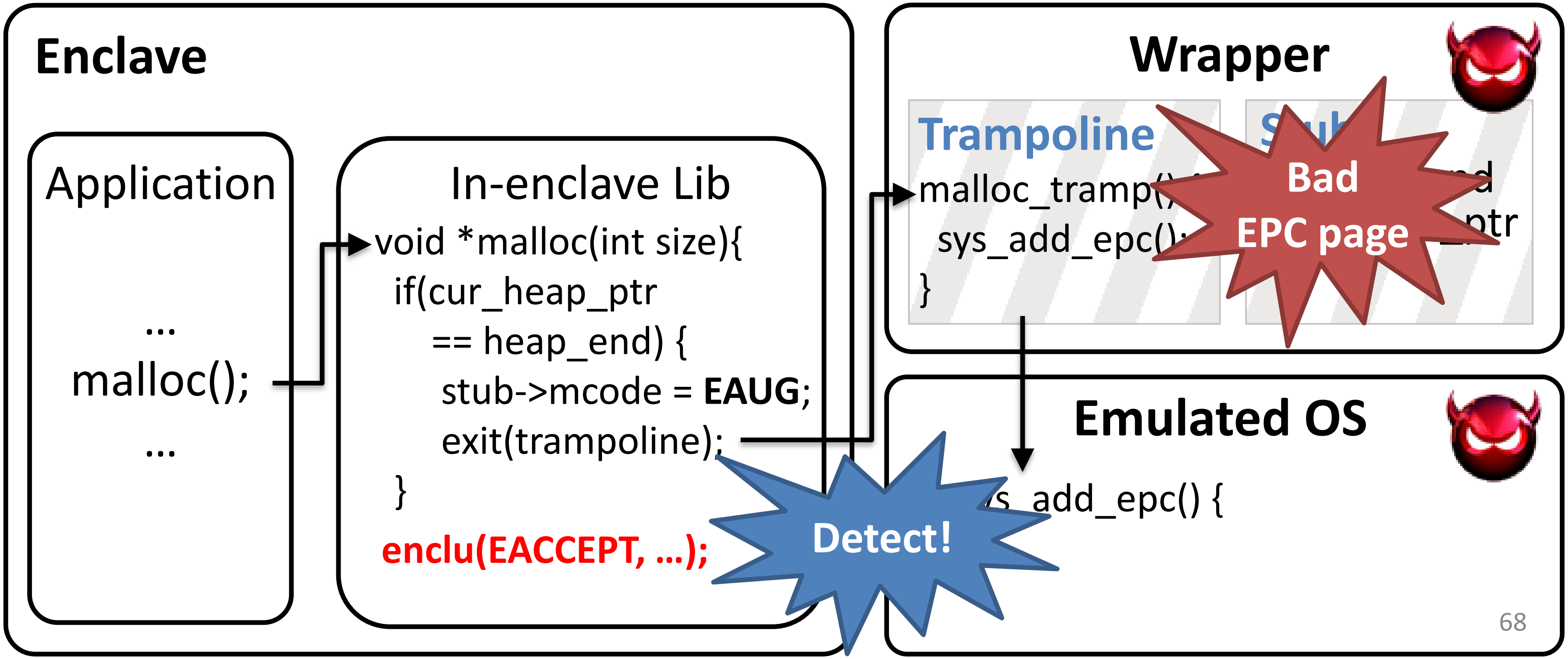
- **Challenge 1: Facilitate the enclave programming**
  - Custom in-enclave library : APIs for user-level SGX instructions
  - Porting standard C library (glibc)

- **Challenge 2: Minimize attack surface between enclave and the potentially malicious host process**
  - Function call relies on OS features will break an execution of enclave programs
  - Such functions open up new attack surfaces (e.g., Iago attacks)

Defense against Iago attacks

- Iago attacks [ASPLOS’13] : Malicious OS tries to subvert trusted application by incorrect behavior

  ex) adds incorrect EPC page for heap



Enclave

Application

```
…
malloc();
…
```

In-enclave Lib

```
void *malloc(int size){
 if(cur_heap_ptr
     == heap_end) {
      stub->mcode = EAUG;
      exit(trampoline);
 }
enclu(EACCEPT, …);
```

Wrapper

Trampoline

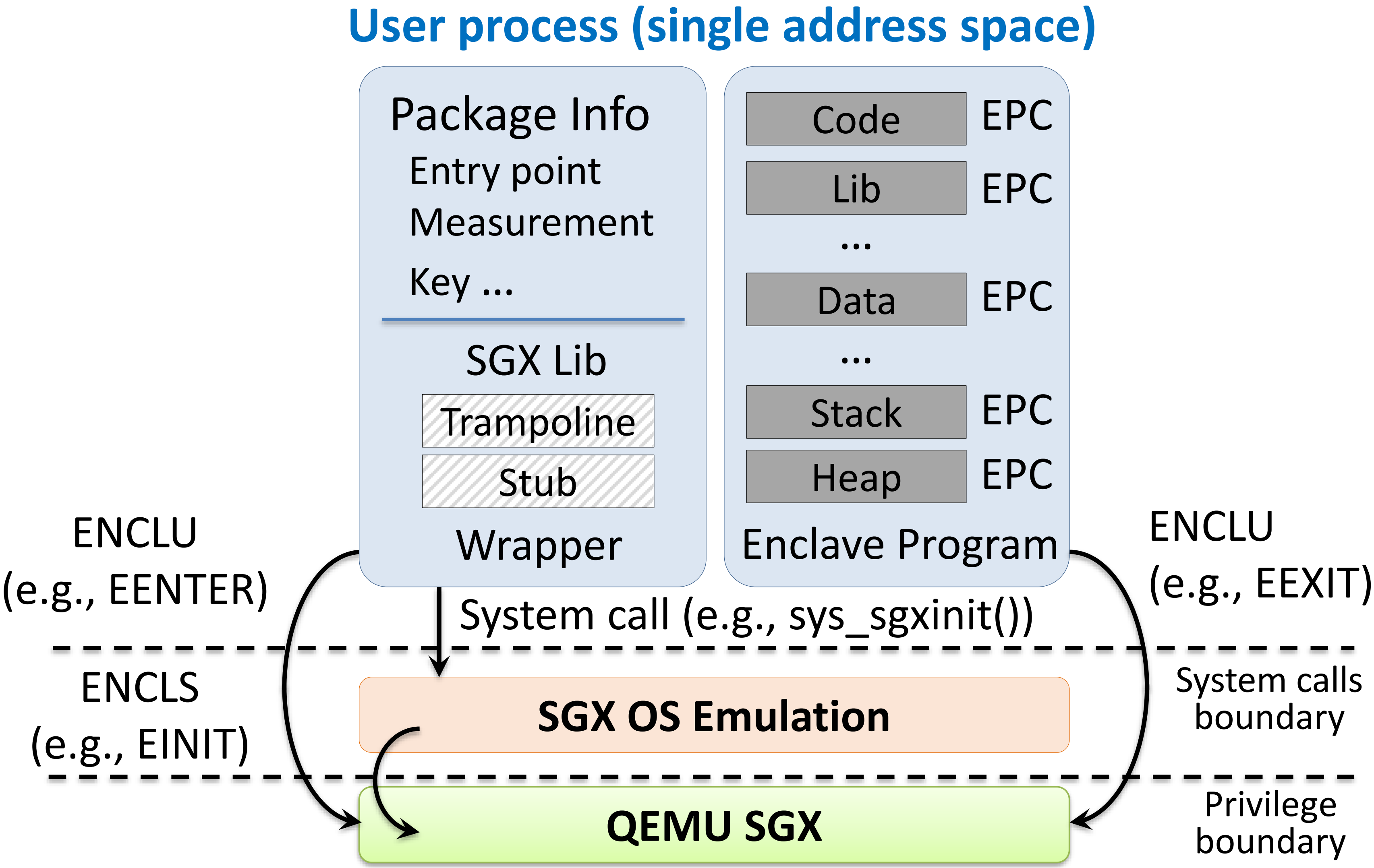
```
malloc_tramp() {
  sys_add_epc();
}
```

Stub

Emulated OS

```
sys_add_epc() {
```

Memory State of OpenSGX Program



User process (single address space)

Package Info

Entry point

Measurement

Key ...

SGX Lib

Trampoline

Stub

Wrapper

Code EPC

Lib EPC

...

Data EPC

...

Stack EPC

Heap EPC

Enclave Program

ENCLU (e.g., EENTER)

ENCLU (e.g., EEXIT)

System call (e.g., sys_sgxinit())

ENCLS (e.g., EINIT)

SGX OS Emulation

QEMU SGX

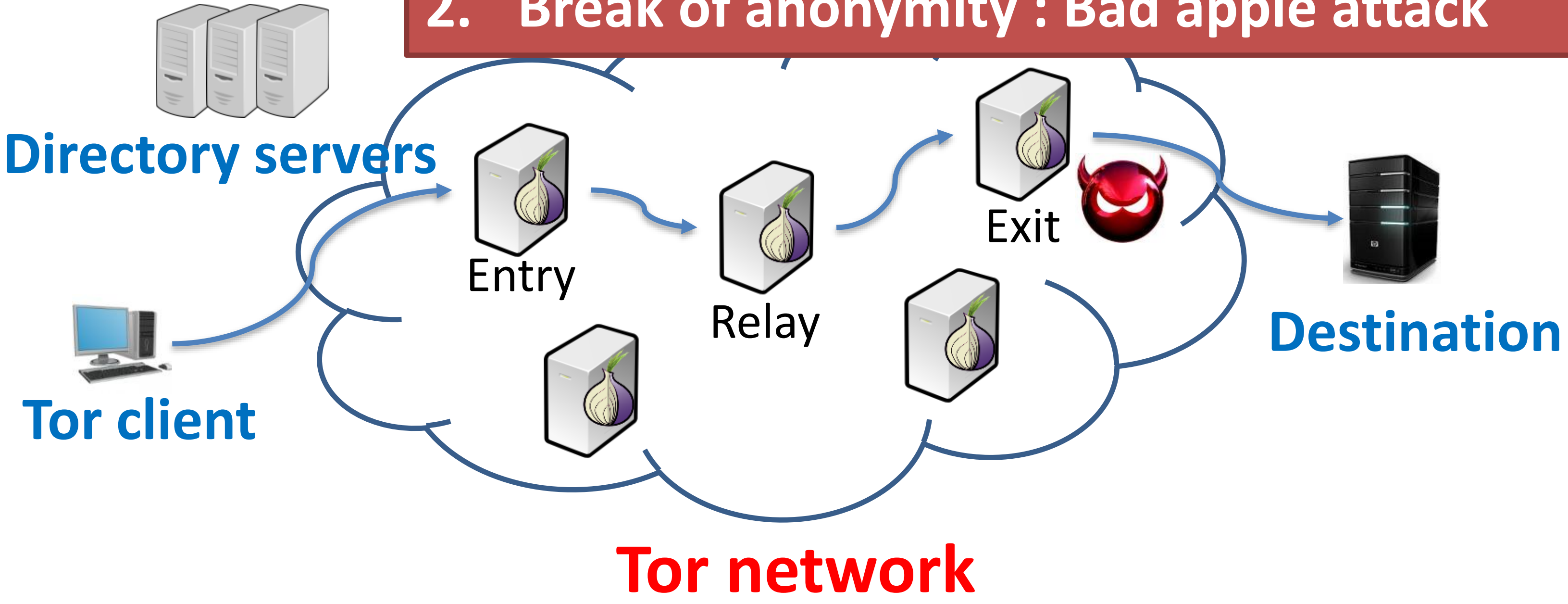
System calls boundary

Privilege boundary

Attacks on Tor Components

- Tor network : uses 3-hop onion routing
  - Directory servers : Advertise available onion routers (ORs)
    vote for bad

When exit node is compromised,
(unless end-to-end encryption is used)
1. Snooping or tampering of the plain-text
2. Break of anonymity : Bad apple attack

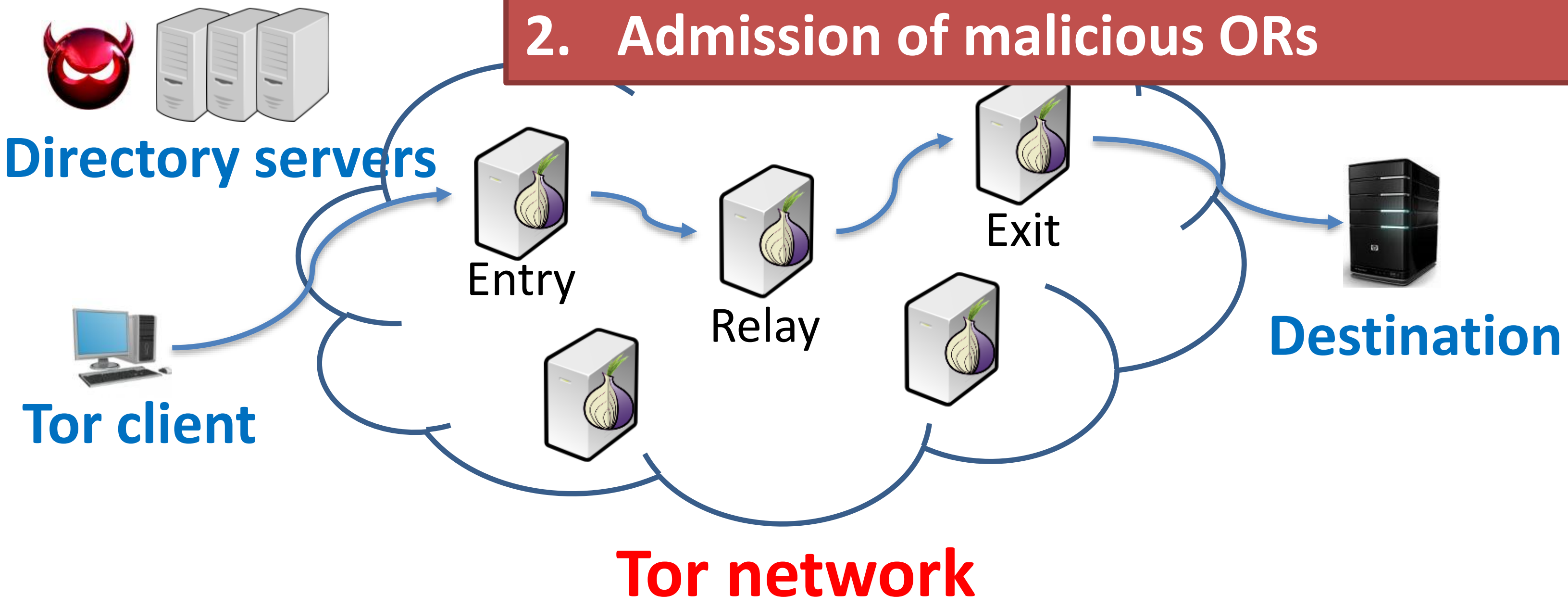


Attacks on Tor Components

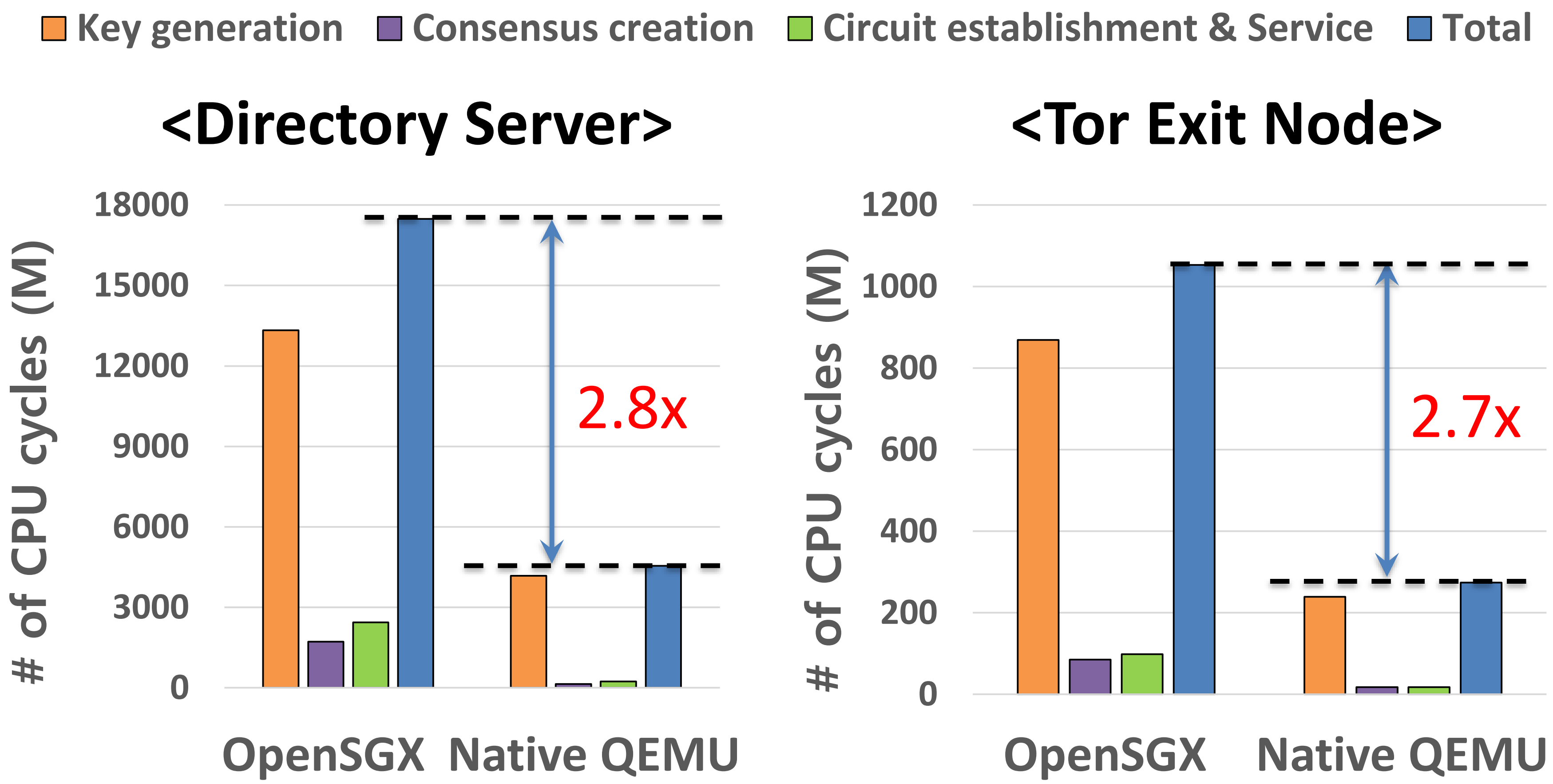
- Tor network : uses 3-hop onion routing
  - Directory servers : Advertise available onion routers (ORs), vote for bad exit nodes

When directory servers are compromised,

1. **Tie-breaking attacks while voting**
2. **Admission of malicious ORs**



Performance Profiling: CPU cycles



- ENCLU(EEXIT, ERESUME) calls
- In-enclave library code to handle stub & trampoline interface

Performance Profiling: TCB

<Directory Server>

	Code	Data	Total
OpenSSL	270	88	358
SgxLib	3	1	4
Tor	3	1	4
Total	276	90	366

<Tor Exit Node>

	Code	Data	Total
OpenSSL	271	89	360
SgxLib	3	1	4
Tor	4	1	5
Total	278	91	369

(Unit: Number of pages)

- Required EPC size: Less than 2MB for each process
- TCB size : 54% smaller than compared to Tor code base

OpenSGX implementation

- OpenSGX is an **open source project**!
  - Modified lines of code : 19K
  - First released in May, 2015
  - 7 Contributors (Gatech, KAIST)
  - 31 unique cloners, 1,645 Views (Until January, 2016)
  - Available at https://github.com/sslab-gatech/opensgx.git