Graminer: Fuzz Testing Gramine LibOS to Harden the Trusted Computing Base

Jaewon Hur, Byoungyoung Lee

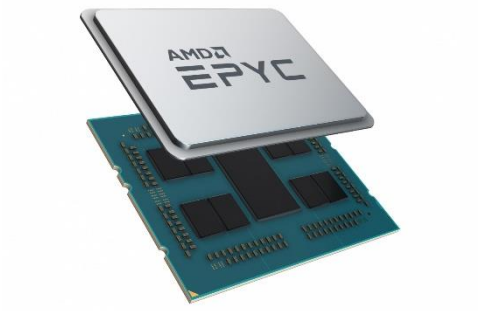
서 울 대 학 교
SEOUL NATIONAL UNIVERSITY

hurjaewon@snu.ac.kr

https://compsec.snu.ac.kr/people/jaewonhur/

Confidential Computing on the Rise

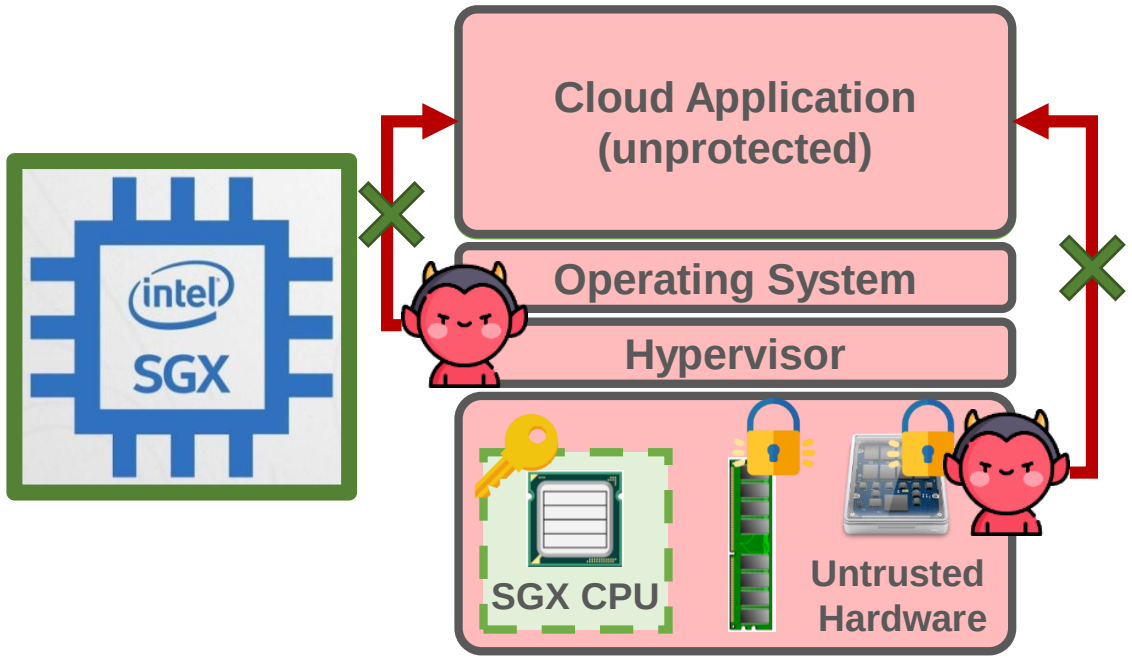
- **Intel SGX**, **AMD SEV**, **ARM CCA**, and **RISC-V Keystone**
- Protect valuable & private data-in-use
- Adopted by emerging cloud applications
  - Data-analytics, machine-learning, etc.



Intel SGX

- ISA extension of **Intel** introduced in 2015
- Construct a ***protected environment*** in the CPU

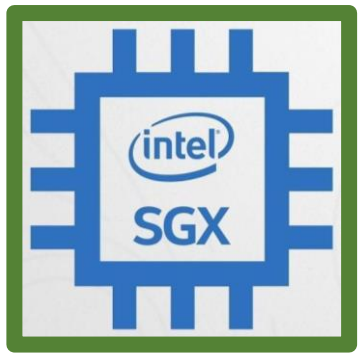
Enclave

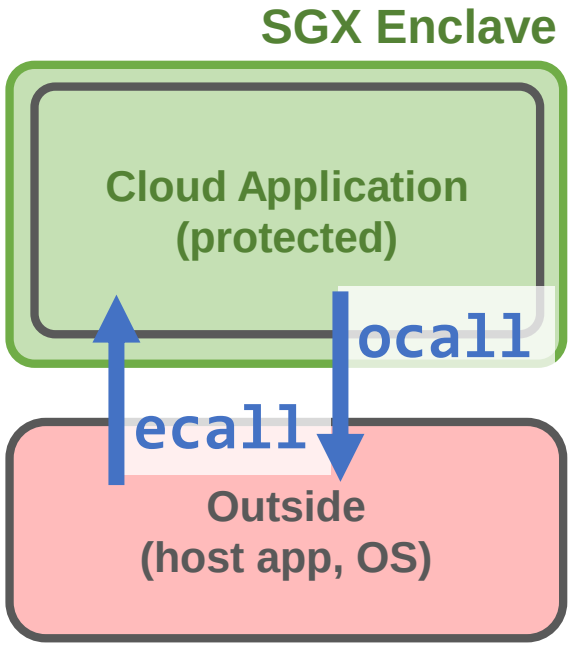


- A user-level application runs in the ***Enclave***
- ***Enclave*** protects the application against OS, hypervisor, and peripherals

Application in SGX Enclave

- Requires a **new interface** to communicate with **untrusted OS**
  - ***Syscalls*** *are not allowed in* ***SGX enclave***





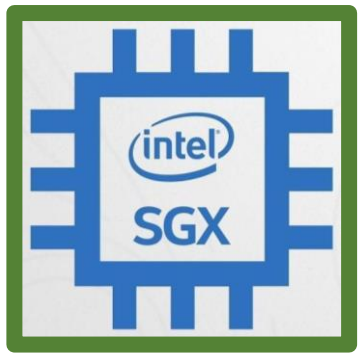
Legacy applications are not aware of ECALL & OCALL

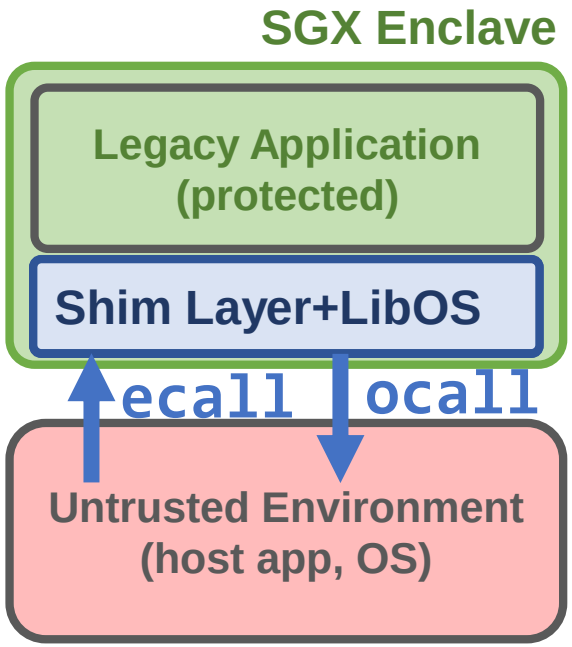
- ***ECALL***
  - Transfer the control flow from outside world to enclave
- ***OCALL***
  - Transfer the control flow from the enclave to the outside world
  - **Applications should use OCALL to receive a service from OS**

Gramine LibOS

- Run **legacy applications** in the SGX enclave
  - Provide a ***Shim Layer***, which connects the **application** and the **OS**
  - Provide a ***LibOS*** to mar

Trusted Computing Base (TCB)





- ***Shim Layer + LibOS***
  - Converts ***syscalls*** from the application into ***ocalls***
  - Sanitizes and forwards the return values of ***ocalls***
  - Memory management, IPC, etc.

Gramine LibOS

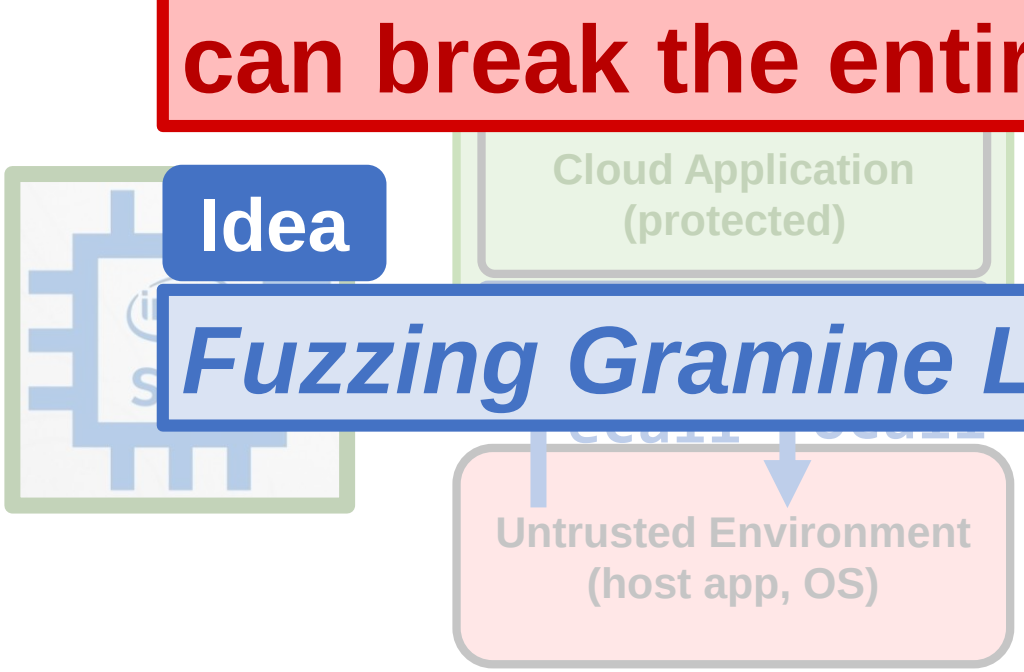
- Run **legacy applications** in the SGX enclave
  - ***n Layer***, which connects the **application** and the **OS**
  - **e (TCB)**

Motivation

Bugs in Gramine LibOS (i.e., TCB) can break the entire enclave

Idea

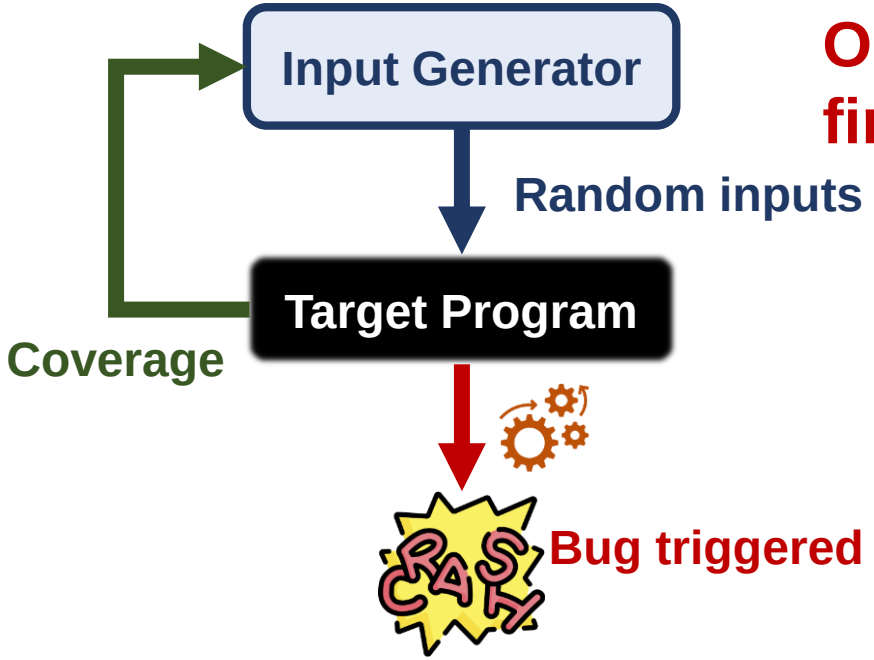
Fuzzing Gramine LibOS to Find Bugs



- Converts ***syscalls*** from the application to ***ocalls***
- e return
- Memory management, IPC, etc.

Fuzzing

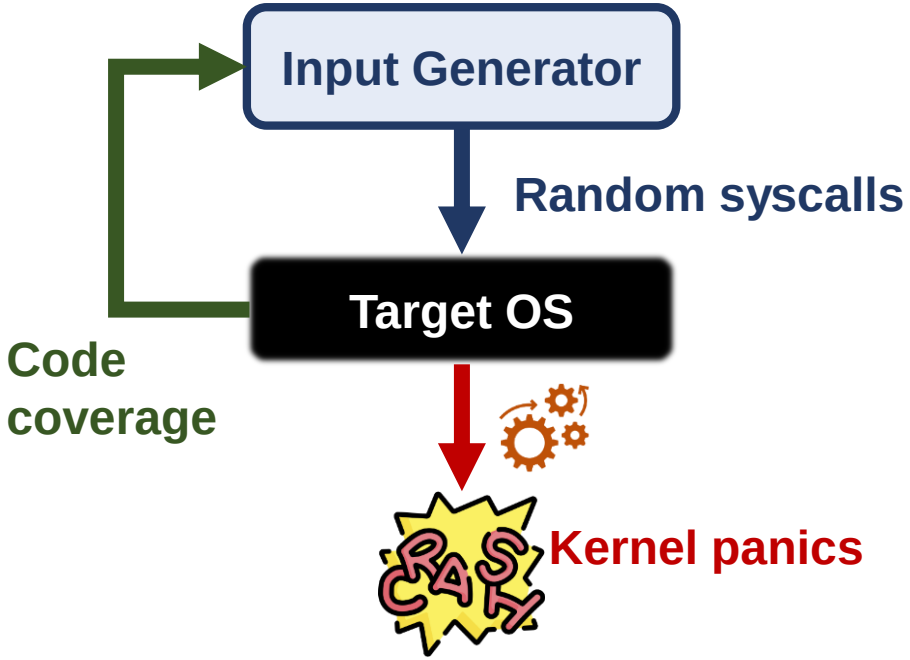
- Powerful tool to find bugs in programs
  - Keep **generating random inputs**
  - Keep **detecting bugs triggered**
  - Guide input generation **using coverage**



One of the most successful tools for finding bugs (e.g., AFL, Razzer, etc.)

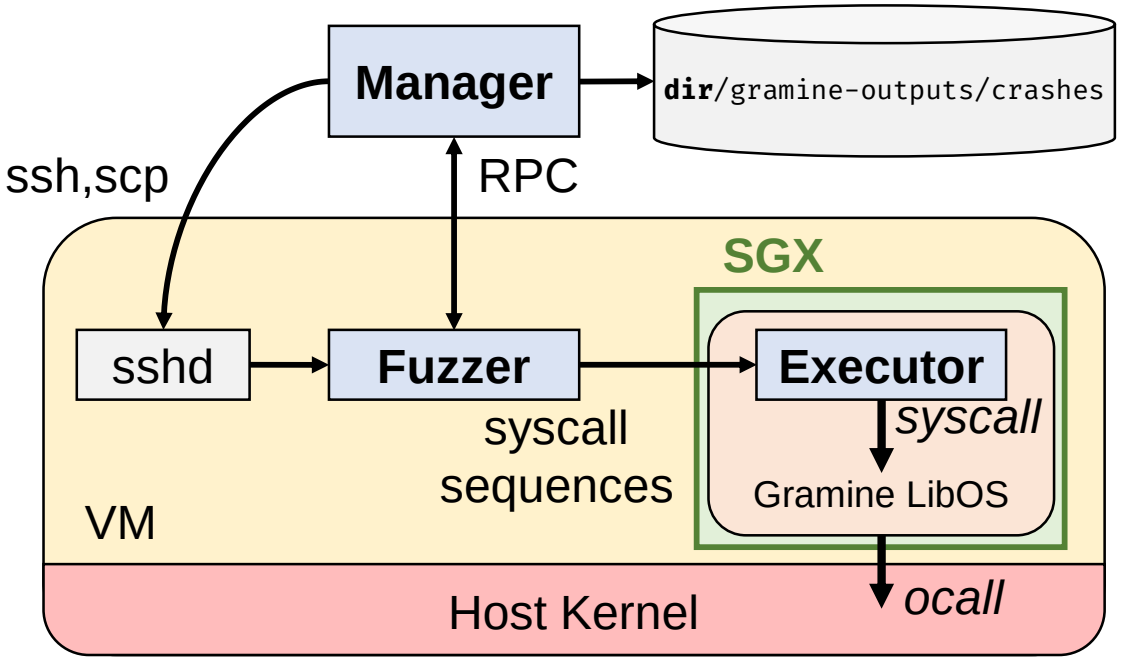
Fuzzing OS (e.g., syzkaller)

- In order to find bugs in **OS**, Fuzzer needs to…
  - Keep **generating random syscalls**
  - Keep **detecting kernel panics triggered**
  - Guide input generation **using code coverage**



Graminer: Fuzzing Gramine LibOS

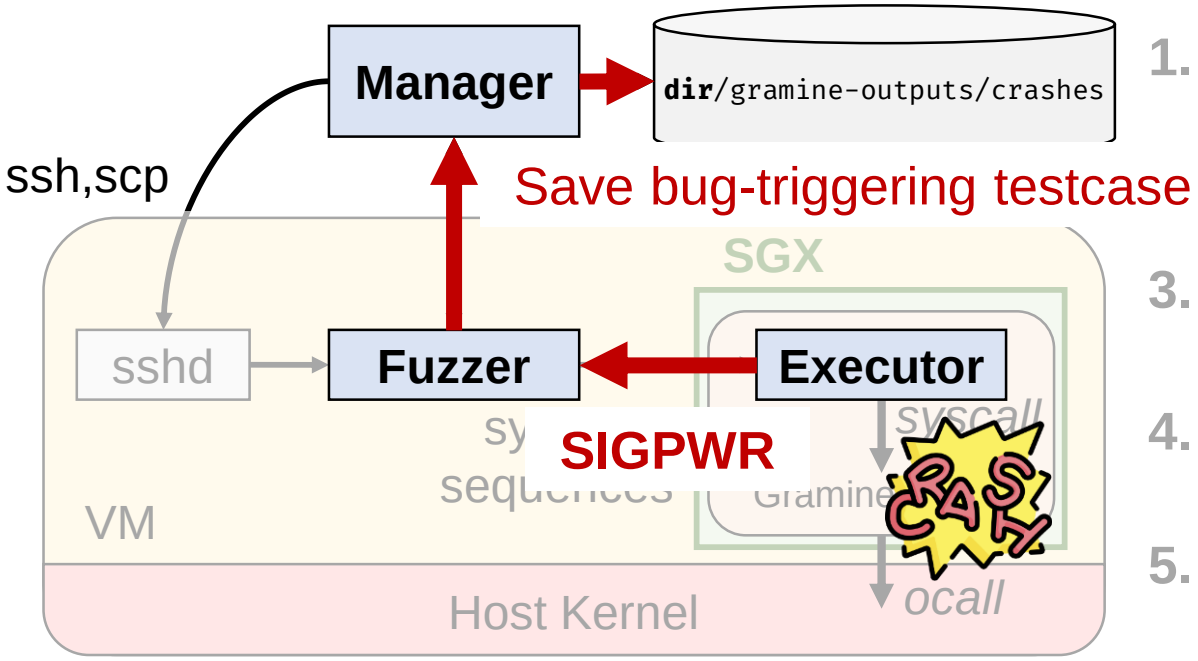
- Find bugs in **Gramine LibOS**
  - Generate **random syscalls**
  - Test on Gramine LibOS
  - Detect any **kernel panic** occurred in Gramine LibOS



1. **Manager** Launches a VM instance which contains Gramine LibOS.
2. **Manager** deploys **Fuzzer** in the VM.
3. **Manager** launches **Fuzzer**, and **Fuzzer** generates random syscall sequences.
4. **Fuzzer** launches **Executor**, and **Executor** invokes the syscalls in Gramine LibOS.
5. **Fuzzer** detects bugs, and **Manager** saves bug triggering testcases.

Graminer: Fuzzing Gramine LibOS

- Detecting kernel panics in **Gramine LibOS**
  - Exits with **SIGPWR** status when **SEGFAULT** is received



1. **Manager** Launches a VM instance which contains Gramine LibOS.
2. **Manager** deploys **Fuzzer** in the VM.
3. **Manager** launches **Fuzzer**, and **Fuzzer** generates random syscall sequences.
4. **Fuzzer** launches **Executor**, and **Executor** invokes the syscalls in Gramine LibOS.
5. **Fuzzer** detects bugs, and **Manager** saves bug triggering testcases.

Graminer: Evaluation

- Practical impact of Graminer
  - Found 6 new bugs within 12 CPU hours
  - All reported and confirmed by Gramine LibOS developers

Table 1. Disclosure of the found bugs and their status

ID	Found bugs	Status
1	Illegal instruction during Gramine internal execution at 0x7fffffee9879 (die_or_inf_loop at cpu.h)	fixed
2	Internal memory fault at 0x00000000 (libos_syscall_fchdir at libos_getcwd.c)	fixed
3	Assert failed ../libos/include/libos_flags_conv.h:25 WITHIN_MASK(prot, PROT_NONE | PROT_READ | PROT_WRITE | ...)	fixed
4	Assert failed ../libos/src/arch/x86_64/libos_context.c:113 IS_ALIGNED_PTR(xstate, LIBOS_XSTATE_ALIGN)	confirmed
5	Error: Internal memory fault with VMA at 0xfffffffffff600000 (libc.so.6+0x14a7d9)	confirmed
6	Internal memory fault at 0x21000000 (libos_syscall_writev at libos_wrappers.c)	fixed

Graminer: Future Extensions

1. Incorporating **ASAN (i.e., Address Sanitizer)**
   - Gramine supports building with ASAN enabled
2. **Coverage** guidance
   - Gramine can be built with AFL coverage instrumentation
   - Need to expose coverage value through pseudo file system
3. **Multi-dimensional input** generation
   - Generating inputs from host kernel also (i.e., return values of ocalls)
   - Need a new input format to interleave the syscalls and ocalls' return values

Graminer: Conclusion

- Bugs in Gramine LibOS, which is TCB, can break down the entire enclave
- We design **Graminer**, a fuzzing tool to find bugs in Gramine LibOS
- Graminer is open-sourced under https://github.com/JaewonHur/graminer.git, hope it will be the baseline for future researches

Thank you


hurjaewon@snu.ac.kr


https://compsec.snu.ac.kr/people/jaewonhur/