EnclaveFuzz: Finding Vulnerabilities in SGX Applications

Liheng Chen[1,2,3] * , Zheming Li[3] * , Zheyu Ma[3], Yuan Li[3,4], Baojian Chen[1,2], Chao Zhang[3,4†]

[1] Institute of Information Engineering, Chinese Academy of Sciences.

[2] School of Cyber Security, University of Chinese Academy of Sciences.

[3] Institute for Network Sciences and Cyberspace of Tsinghua University.

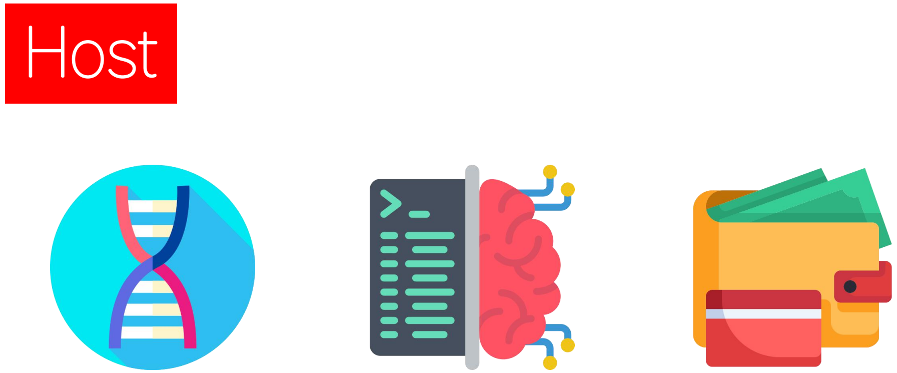
[4] Zhongguancun Laboratory.

*The first two authors contributed equally to this work.

†Corresponding author: chaoz@tsinghua.edu.cn

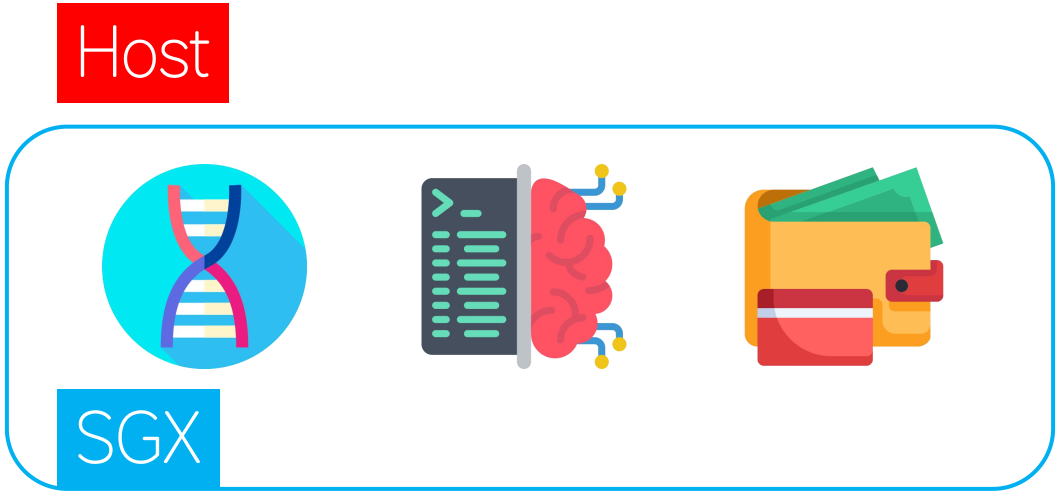
SGX Applications

Applications use Intel SGX to protect the confidentiality and integrity of data while performing computation on untrusted platforms.



SGX Applications

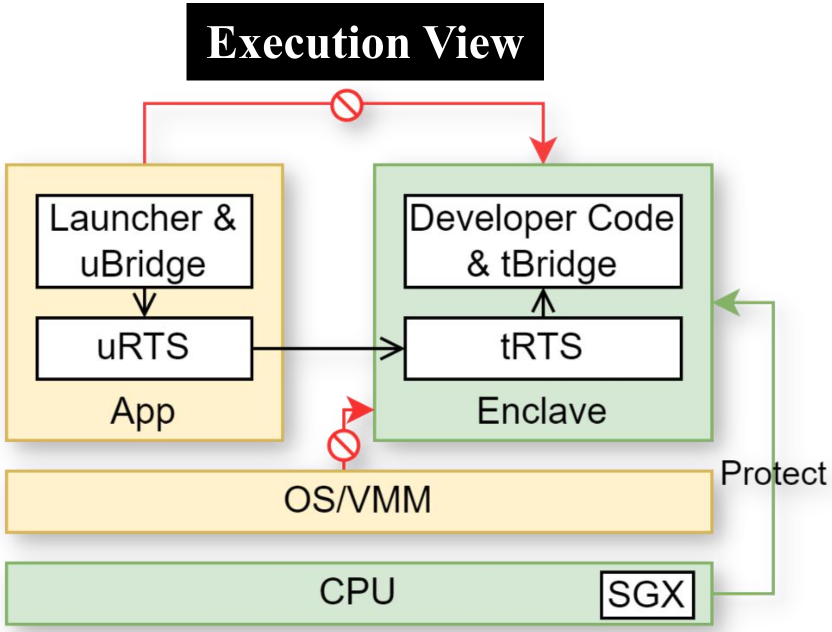
Applications use Intel SGX to protect the confidentiality and integrity of data while performing computation on untrusted platforms.

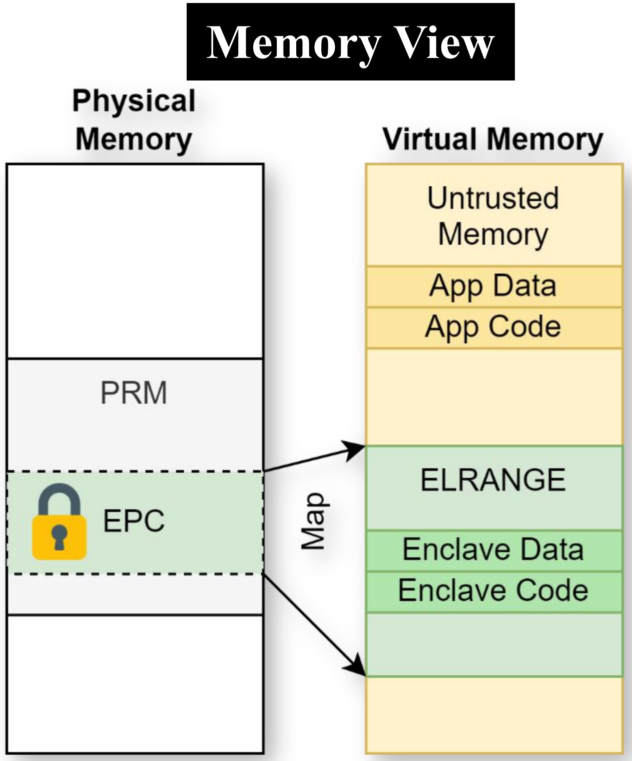


Intel SGX

Intel SGX leverages hardware resources to protect enclave instances from the host platform. One can enter enclave only via entry point.

Memory of enclave instance is independent and encrypted.





Related Works

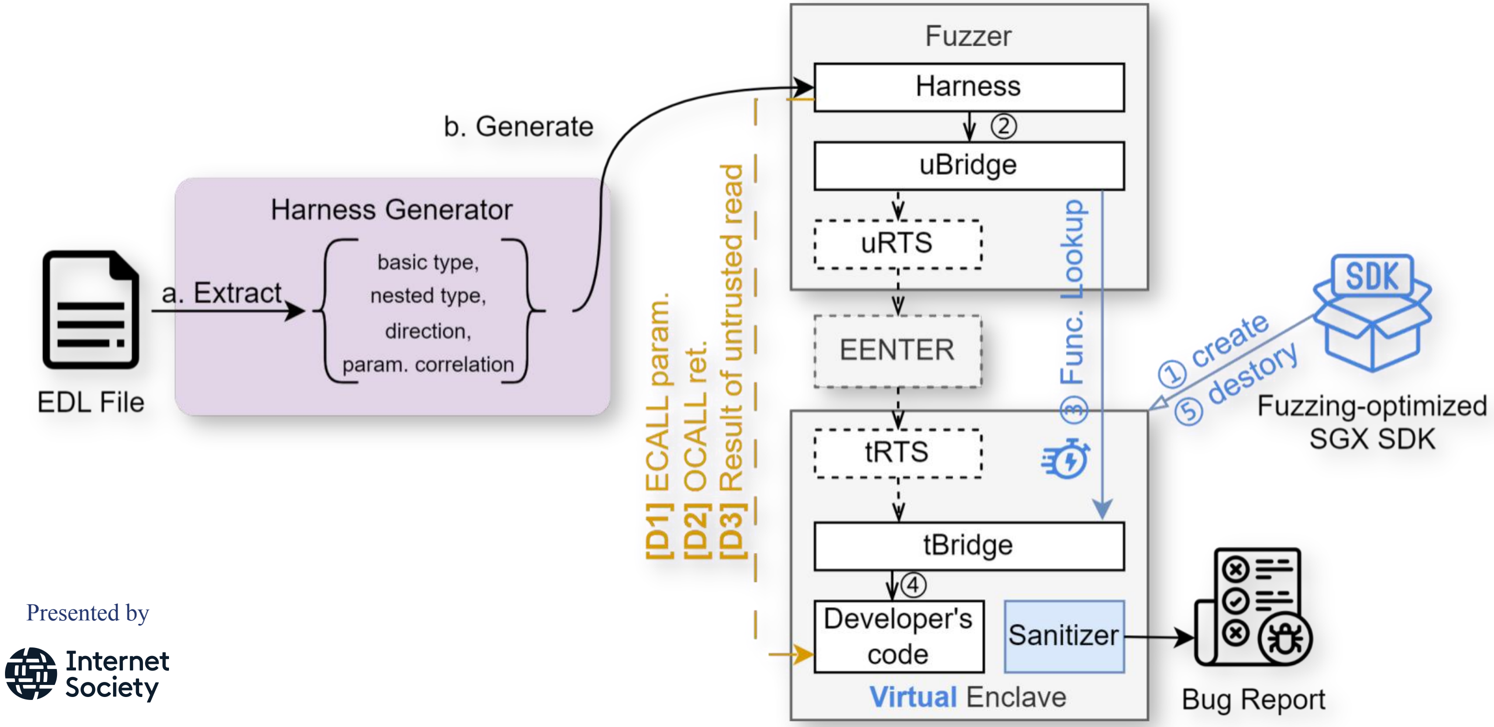
1. **TeeRex**[SEC'20] and **COIN attacks**[ASPLOS'20] exploit ***symbolic execution*** but face state explosion and unresolved constraints in large-scale applications.
2. **SGXFuzz**[SEC'22] is a black-box fuzzer that identifies input structures via ***page fault feedback***, but has difficulty handling complex parameters. It can only detect a limited number of vulnerability types without sanitizer.
3. **FuzzSGX**[EuroS&P'23] incurs overhead by ***mutating the host*** in a fuzz loop to test enclaves. It lacks untrusted memory input and SGX-specific sanitizers, limits vulnerability detection, and runs in a less efficient simulation mode.

Limitations

1. Insufficient understanding of the input structures and dimensions.
2. Limited bug oracle capabilities.
3. Slow fuzzing process due to redundant management routines.

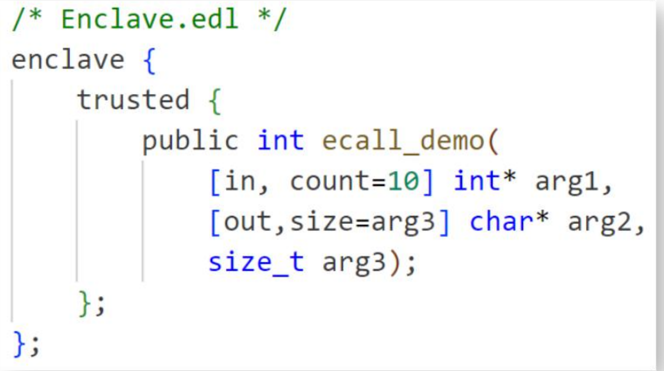
EnclaveFuzz Overview

1. A multi-dimensional structure-aware fuzzing harness.
2. An optimized SGX SDK to build a Virtual Enclave for faster fuzzing.
3. A sanitizer for SGX-specific and memory corruption vulnerabilities.



Multi-dimensional Structure-aware Fuzzing

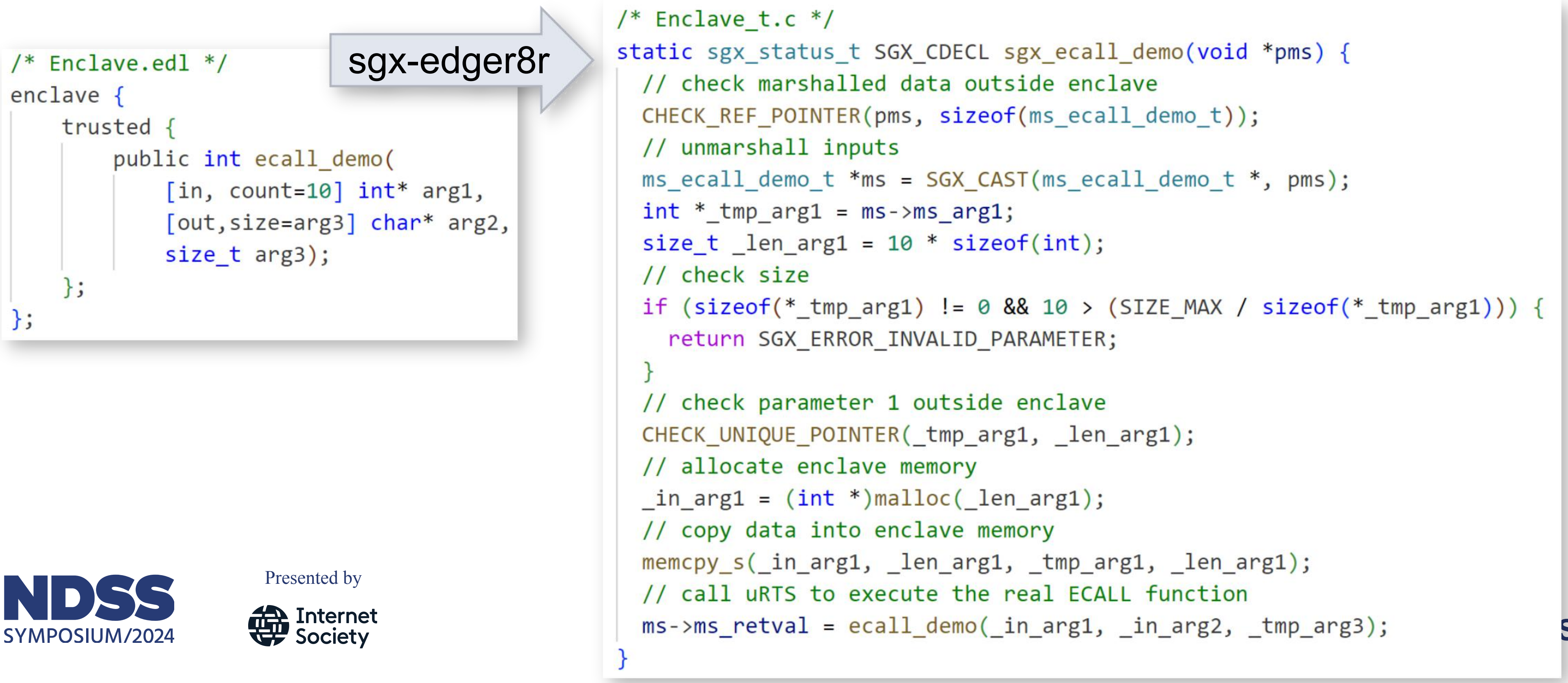
The enclave performs the necessary sanity checks in tBridge as described in the EDL.



```
/* Enclave.edl */
enclave {
    trusted {
        public int ecall_demo(
            [in, count=10] int* arg1,
            [out,size=arg3] char* arg2,
            size_t arg3);
    };
};
```

Multi-dimensional Structure-aware Fuzzing

The enclave performs the necessary sanity checks in tBridge as described in the EDL.



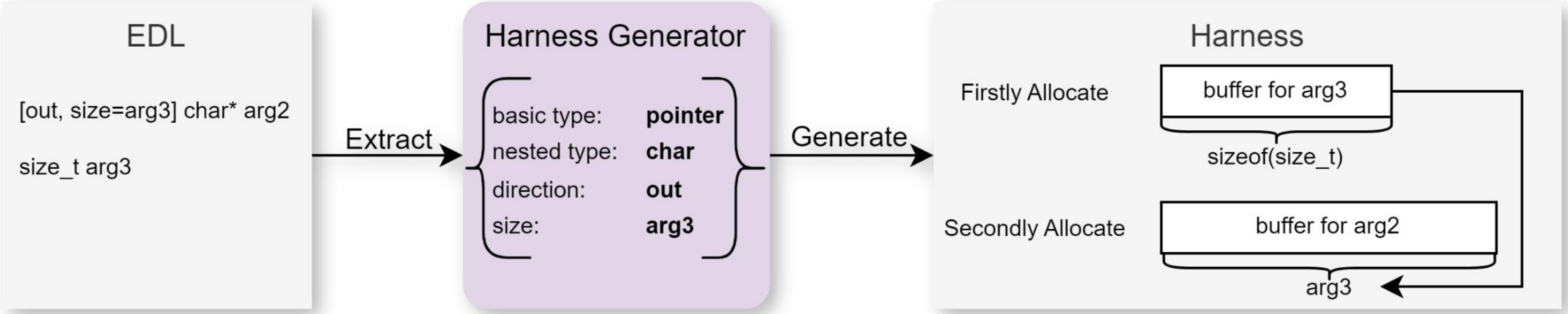
```
/* Enclave.edl */
enclave {
    trusted {
        public int ecall_demo(
            [in, count=10] int* arg1,
            [out,size=arg3] char* arg2,
            size_t arg3);
    };
};
```

sgx-edger8r

```
/* Enclave_t.c */
static sgx_status_t SGX_CDECL sgx_ecall_demo(void *pms) {
  // check marshalled data outside enclave
  CHECK_REF_POINTER(pms, sizeof(ms_ecall_demo_t));
  // unmarshall inputs
  ms_ecall_demo_t *ms = SGX_CAST(ms_ecall_demo_t *, pms);
  int *_tmp_arg1 = ms->ms_arg1;
  size_t _len_arg1 = 10 * sizeof(int);
  // check size
  if (sizeof(*_tmp_arg1) != 0 && 10 > (SIZE_MAX / sizeof(*_tmp_arg1))) {
    return SGX_ERROR_INVALID_PARAMETER;
  }
  // check parameter 1 outside enclave
  CHECK_UNIQUE_POINTER(_tmp_arg1, _len_arg1);
  // allocate enclave memory
  _in_arg1 = (int *)malloc(_len_arg1);
  // copy data into enclave memory
  memcpy_s(_in_arg1, _len_arg1, _tmp_arg1, _len_arg1);
  // call uRTS to execute the real ECALL function
  ms->ms_retval = ecall_demo(_in_arg1, _in_arg2, _tmp_arg3);
}
```

Multi-dimensional Structure-aware Fuzzing

To overcome enclave sanity checks, EnclaveFuzz extracts security boundaries from EDL, generating a structure-aware fuzzing harness.

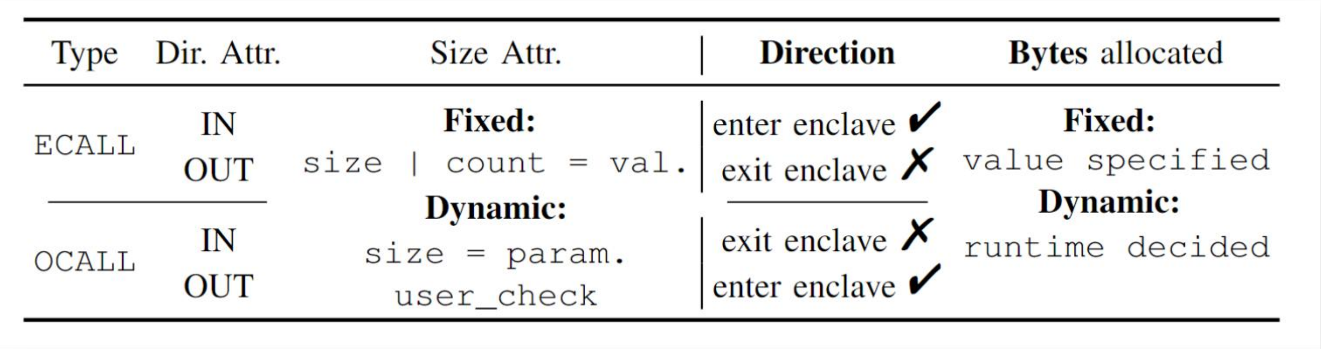


Presented by

#NDSSSymposium2024

Multi-dimensional Structure-aware Fuzzing

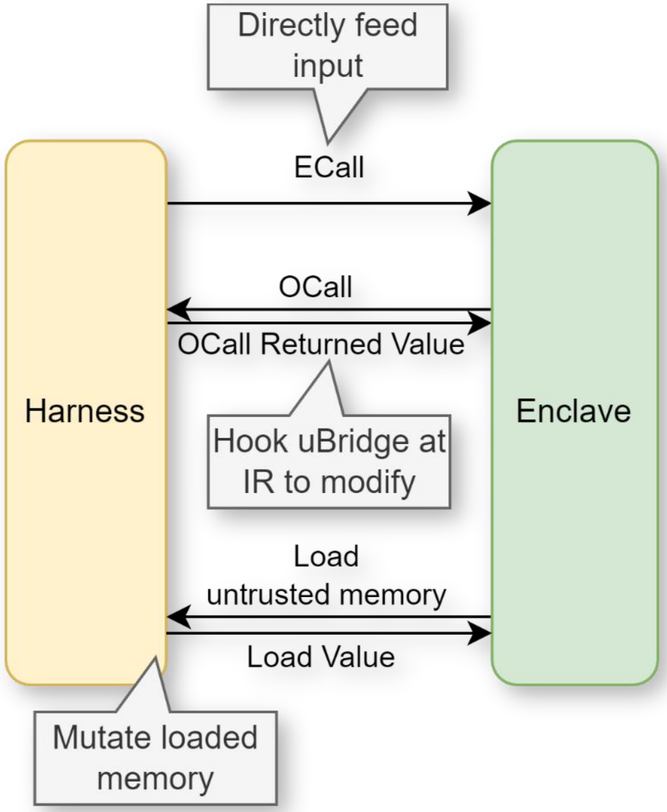
EnclaveFuzz analyzes parameters and handles data directions based on EDL attributes, and specifically manages *user_check* pointers for input preparation.



Type	Dir. Attr.	Size Attr.	**Direction**	**Bytes** allocated
ECALL	IN OUT	**Fixed:** size | count = val.	enter enclave ✓ exit enclave ✗	**Fixed:** value specified
OCALL	IN OUT	**Dynamic:** size = param. user_check	exit enclave ✗ enter enclave ✓	**Dynamic:** runtime decided

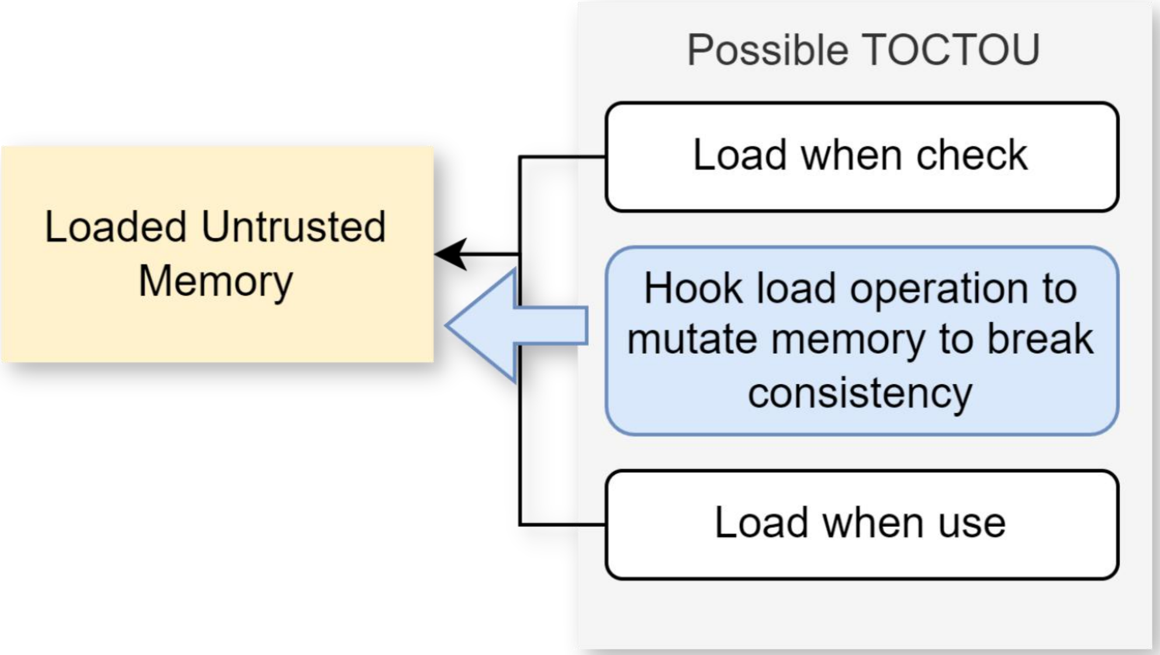
Multi-dimensional Structure-aware Fuzzing

EnclaveFuzz prepares data for ECALLs, OCALLs, and untrusted memory, boosting its efficiency in detecting enclave vulnerabilities.



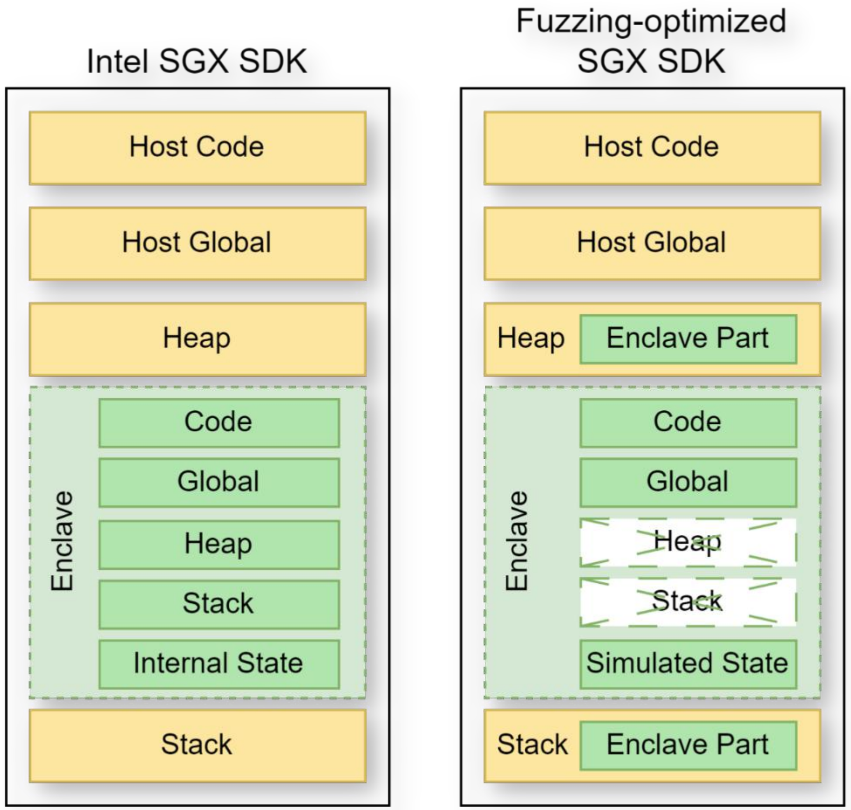
Multi-dimensional Structure-aware Fuzzing

E.g. Untrusted memory load dimension for testing TOCTOU bugs.



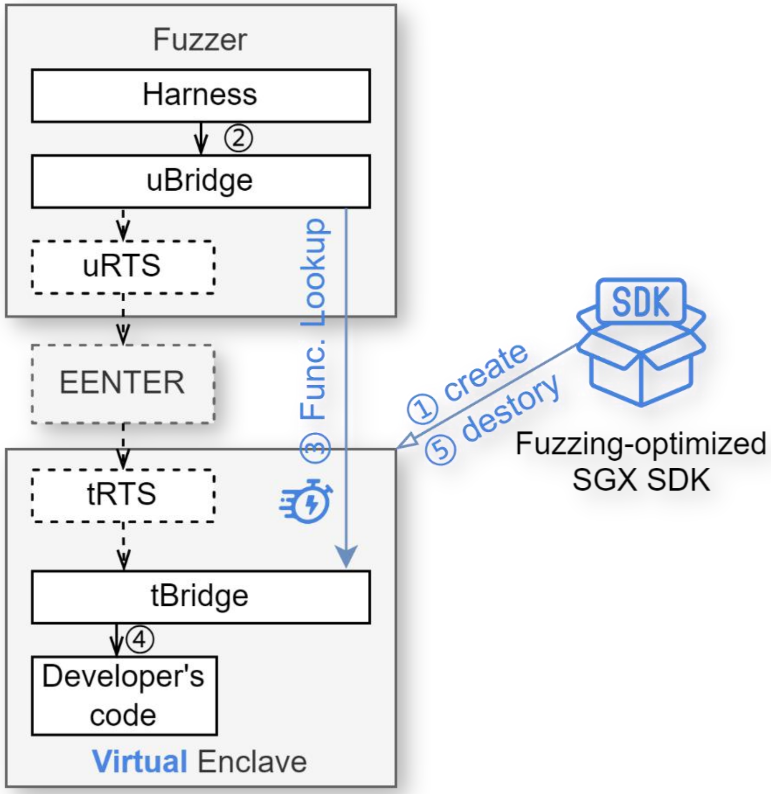
Optimized SGX SDK and Virtual Enclave

EnclaveFuzz loads enclave code as a traditional shared library, using shadow map for memory differentiation, simulating code execution to avoid SGX independent memory management and context switching.



Optimized SGX SDK and Virtual Enclave

This approach speeds up Virtual Enclave execution while maintaining critical sanity checks to ensure functional consistency.



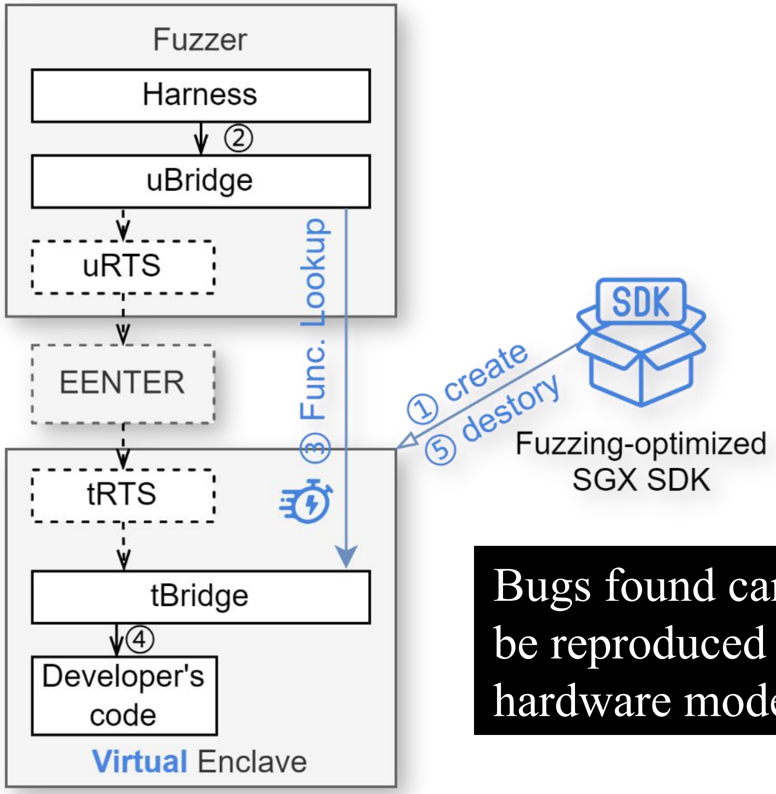
NDSS SYMPOSIUM/2024

Presented by Internet Society

#NDSSSymposium2024

Optimized SGX SDK and Virtual Enclave

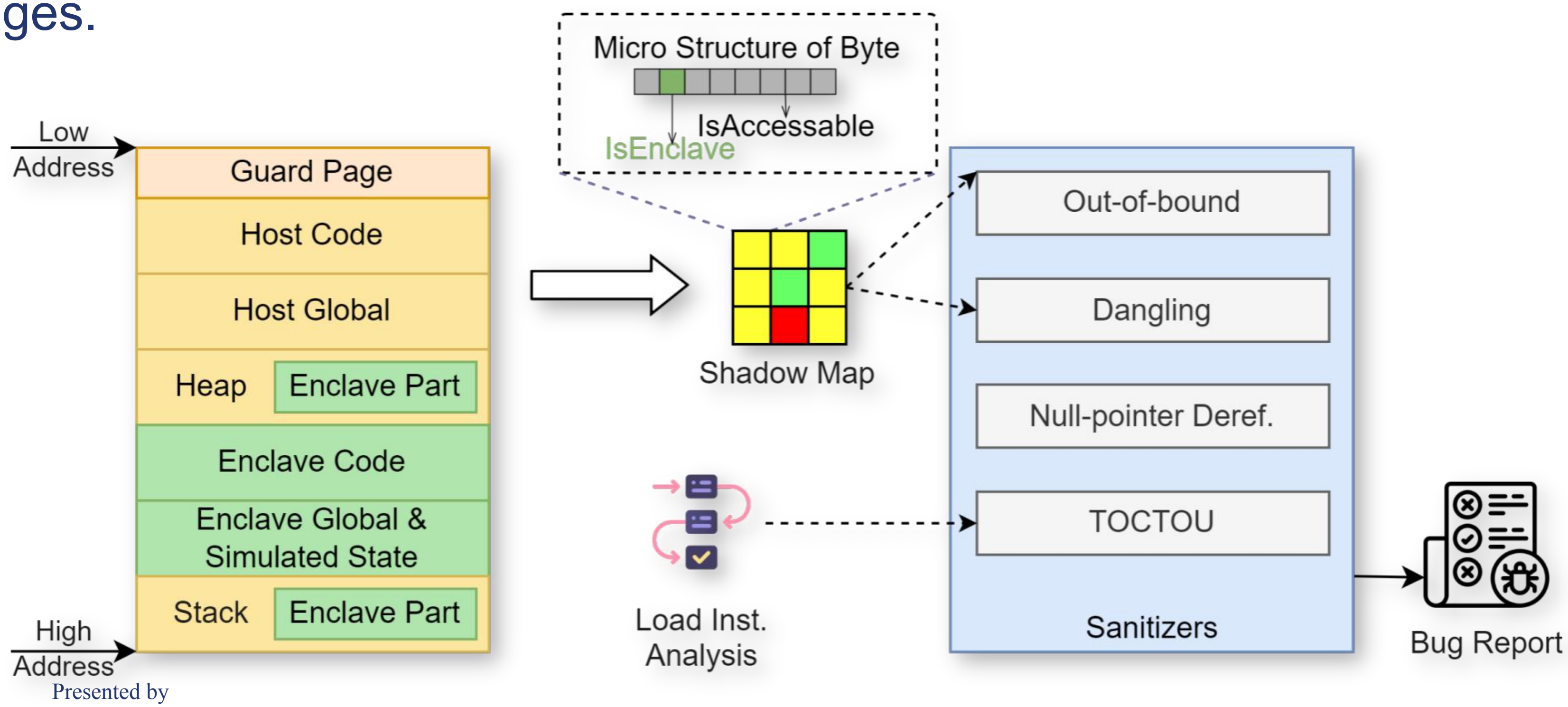
This approach speeds up Virtual Enclave execution while maintaining critical sanity checks to ensure functional consistency.



Bugs found can be reproduced in hardware mode

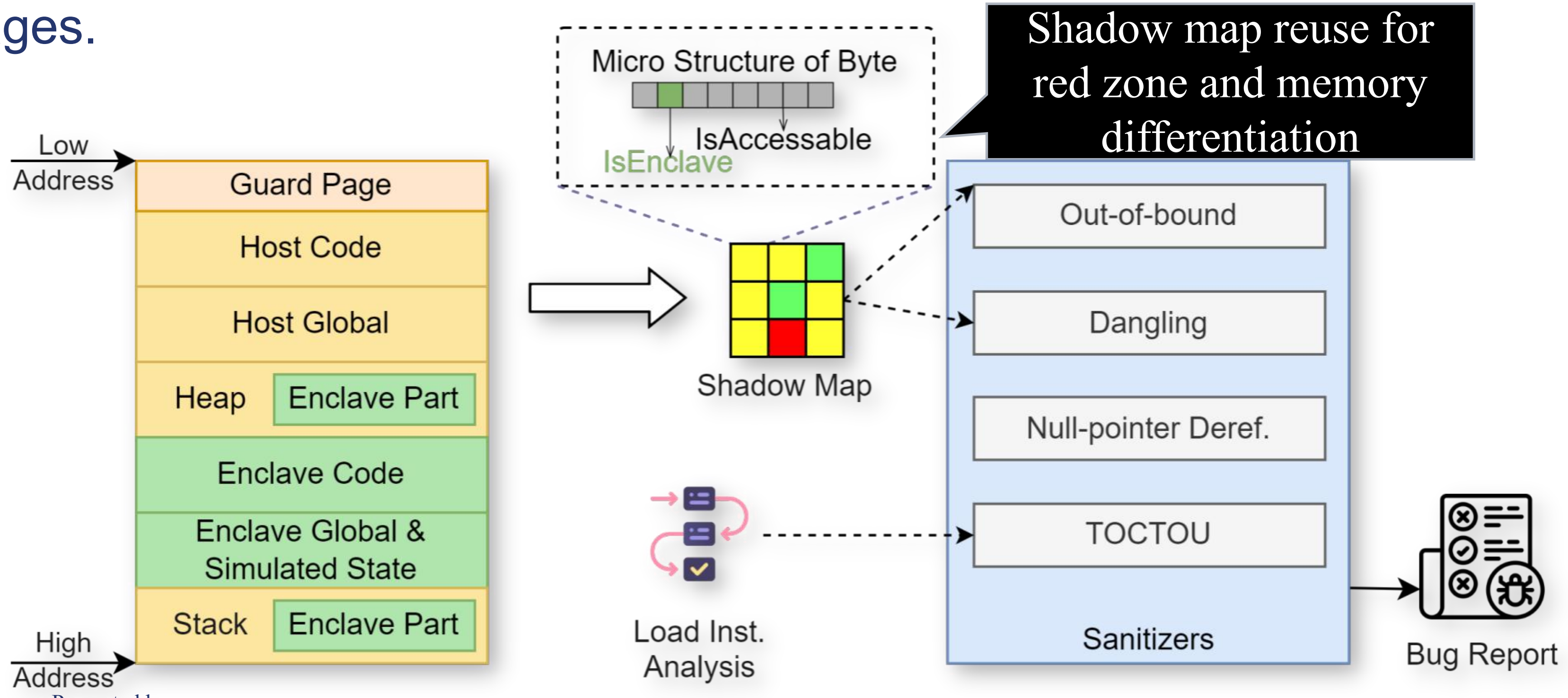
Vulnerability Detection

EnclaveFuzz detects out-of-bound and dangling pointer dereferences via redzone in shadow map, and null pointer dereferences via guard pages.



Vulnerability Detection

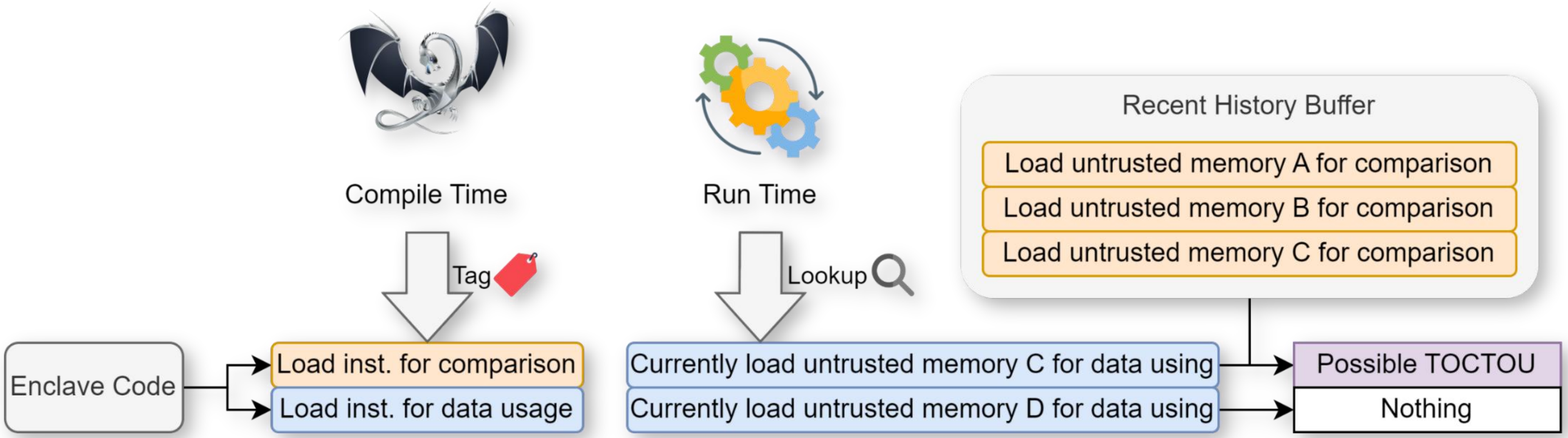
EnclaveFuzz detects out-of-bound and dangling pointer dereferences via redzone in shadow map, and null pointer dereferences via guard pages.



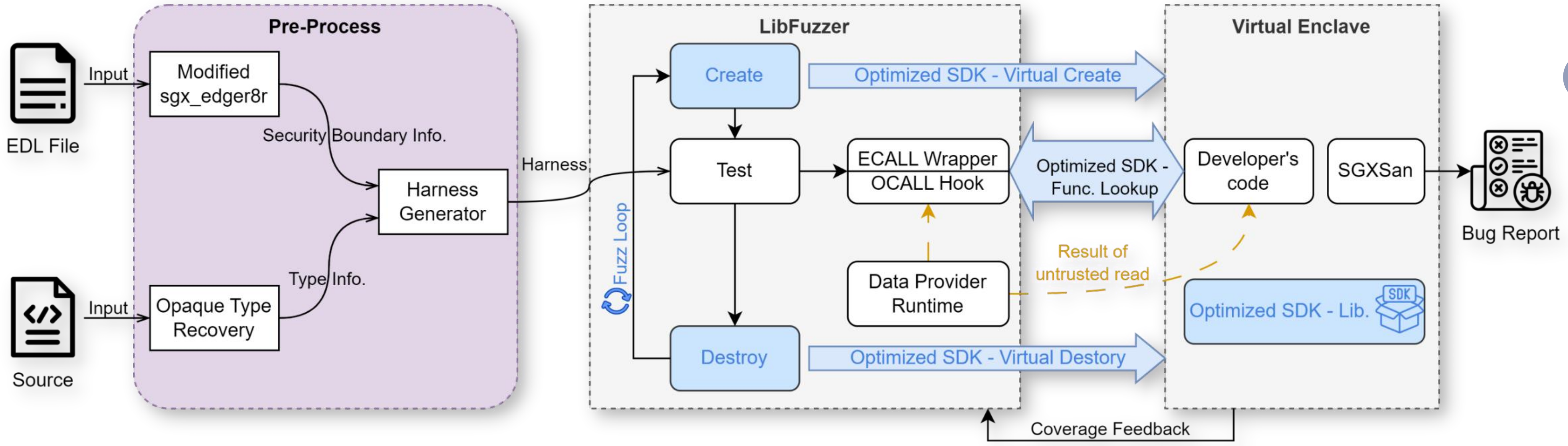
Shadow map reuse for red zone and memory differentiation

Vulnerability Detection

Detect TOCTOU: Analyze and tag load instructions when compiling, track load instructions at runtime when accessing the same memory. Then mutate the memory to break consistency.

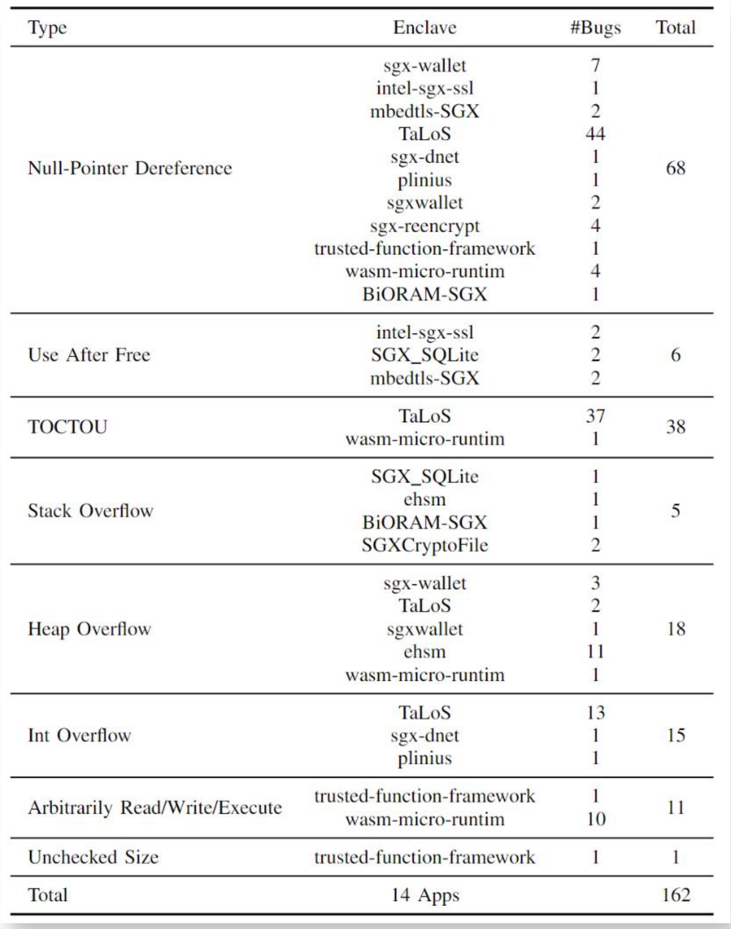

Compile Time
Run Time
Recent History Buffer
Load untrusted memory A for comparison
Load untrusted memory B for comparison
Load untrusted memory C for comparison
Tag
Lookup
Enclave Code
Load inst. for comparison
Load inst. for data usage
Currently load untrusted memory C for data using
Currently load untrusted memory D for data using
Possible TOCTOU
Nothing

Implementation Overview



Bugs Found

EnclaveFuzz found **162 bugs** in 20 real-world open source enclaves.

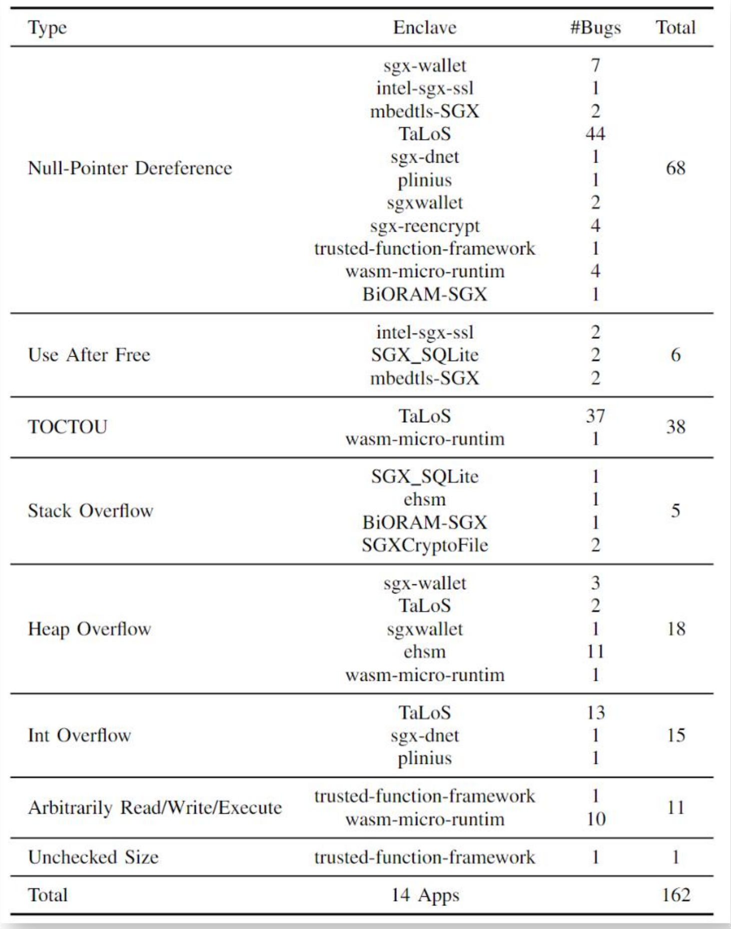


Type	Enclave	#Bugs	Total
Null-Pointer Dereference	sgx-wallet intel-sgx-ssl mbedtls-SGX TaLoS sgx-dnet plinius sgxwallet sgx-reencrypt trusted-function-framework wasm-micro-runtim BiORAM-SGX	7 1 2 44 1 1 2 4 1 4 1	68
Use After Free	intel-sgx-ssl SGX_SQLite mbedtls-SGX	2 2 2	6
TOCTOU	TaLoS wasm-micro-runtim	37 1	38
Stack Overflow	SGX_SQLite ehsm BiORAM-SGX SGXCryptoFile	1 1 1 2	5
Heap Overflow	sgx-wallet TaLoS sgxwallet ehsm wasm-micro-runtim	3 2 1 11 1	18
Int Overflow	TaLoS sgx-dnet plinius	13 1 1	15
Arbitrarily Read/Write/Execute	trusted-function-framework wasm-micro-runtim	1 10	11
Unchecked Size	trusted-function-framework	1	1
Total	14 Apps		162

Bugs Found

EnclaveFuzz found **162 bugs** in 20 real-world open source enclaves.

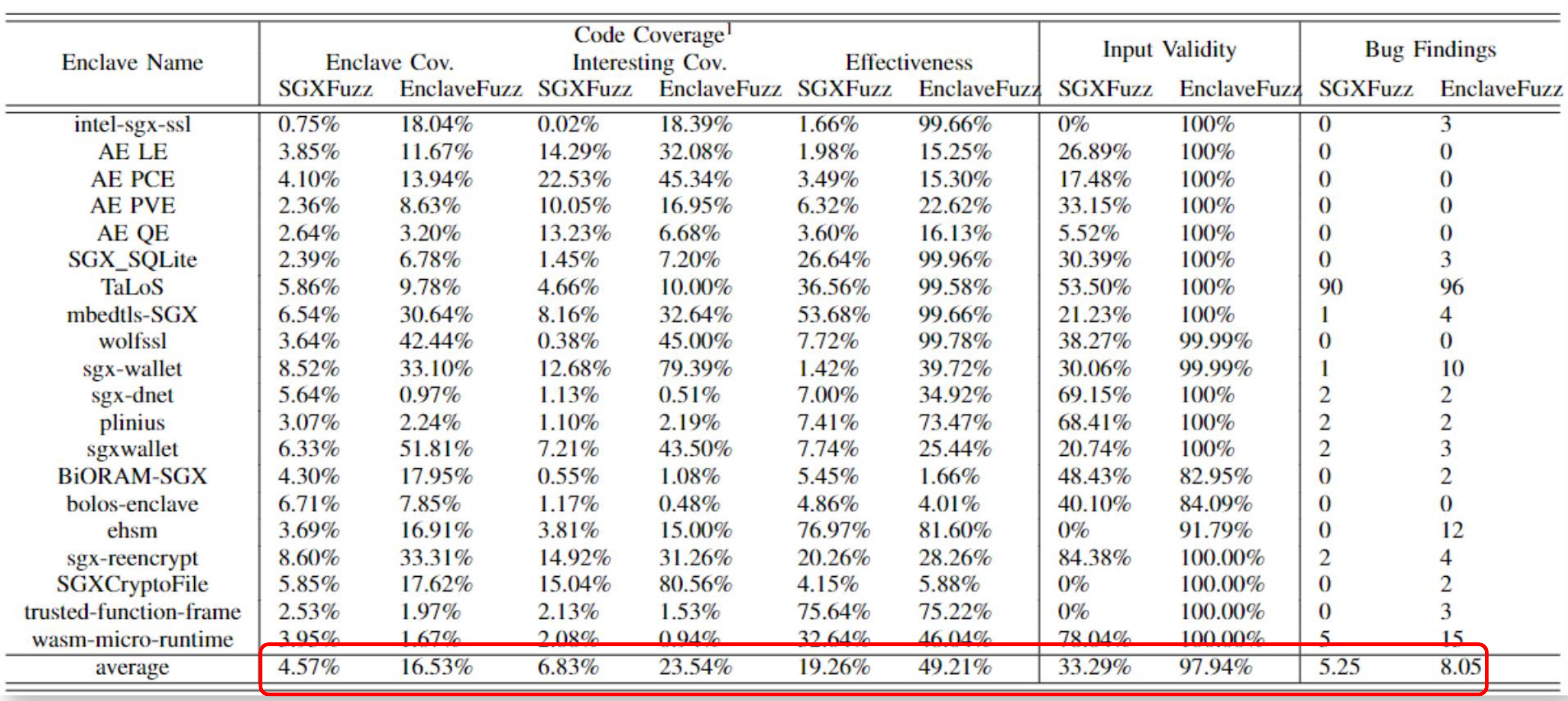
Most are **Null-Pointer Dereference** and **TOCTOU**. Developers overlook the nuances of SGX security, especially cross-bounds pointers.



Type	Enclave	#Bugs	Total
Null-Pointer Dereference	sgx-wallet	7	68
	intel-sgx-ssl	1	
	mbedtls-SGX	2	
	TaLoS	44	
	sgx-dnet	1	
	plinius	1	
	sgxwallet	2	
	sgx-reencrypt	4	
	trusted-function-framework	1	
	wasm-micro-runtim	4	
	BiORAM-SGX	1	
Use After Free	intel-sgx-ssl	2	6
	SGX_SQLite	2	
	mbedtls-SGX	2	
TOCTOU	TaLoS	37	38
	wasm-micro-runtim	1	
Stack Overflow	SGX_SQLite	1	5
	ehsm	1	
	BiORAM-SGX	1	
	SGXCryptoFile	2	
Heap Overflow	sgx-wallet	3	18
	TaLoS	2	
	sgxwallet	1	
	ehsm	11	
	wasm-micro-runtim	1	
Int Overflow	TaLoS	13	15
	sgx-dnet	1	
	plinius	1	
Arbitrarily Read/Write/Execute	trusted-function-framework	1	11
	wasm-micro-runtim	10	
Unchecked Size	trusted-function-framework	1	1
Total	14 Apps		162

Compare with SGXFuzz

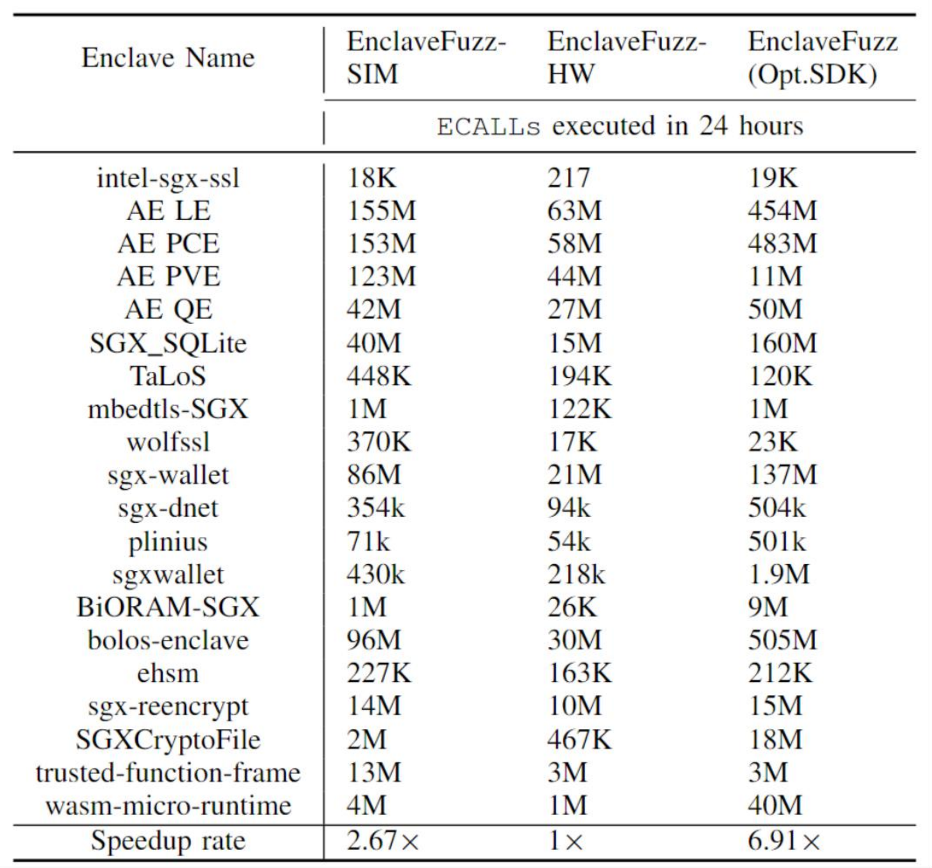
EnclaveFuzz covers more code coverage, improves input validity, and finds more bugs than the state-of-the-art SGXFuzz.



Enclave Name	Code Coverage[1]						Input Validity		Bug Findings	
	Enclave Cov.		Interesting Cov.		Effectiveness					
	SGXFuzz	EnclaveFuzz	SGXFuzz	EnclaveFuzz	SGXFuzz	EnclaveFuzz	SGXFuzz	EnclaveFuzz	SGXFuzz	EnclaveFuzz
intel-sgx-ssl	0.75%	18.04%	0.02%	18.39%	1.66%	99.66%	0%	100%	0	3
AE LE	3.85%	11.67%	14.29%	32.08%	1.98%	15.25%	26.89%	100%	0	0
AE PCE	4.10%	13.94%	22.53%	45.34%	3.49%	15.30%	17.48%	100%	0	0
AE PVE	2.36%	8.63%	10.05%	16.95%	6.32%	22.62%	33.15%	100%	0	0
AE QE	2.64%	3.20%	13.23%	6.68%	3.60%	16.13%	5.52%	100%	0	0
SGX_SQLite	2.39%	6.78%	1.45%	7.20%	26.64%	99.96%	30.39%	100%	0	3
TaLoS	5.86%	9.78%	4.66%	10.00%	36.56%	99.58%	53.50%	100%	90	96
mbedtls-SGX	6.54%	30.64%	8.16%	32.64%	53.68%	99.66%	21.23%	100%	1	4
wolfssl	3.64%	42.44%	0.38%	45.00%	7.72%	99.78%	38.27%	99.99%	0	0
sgx-wallet	8.52%	33.10%	12.68%	79.39%	1.42%	39.72%	30.06%	99.99%	1	10
sgx-dnet	5.64%	0.97%	1.13%	0.51%	7.00%	34.92%	69.15%	100%	2	2
plinius	3.07%	2.24%	1.10%	2.19%	7.41%	73.47%	68.41%	100%	2	2
sgxwallet	6.33%	51.81%	7.21%	43.50%	7.74%	25.44%	20.74%	100%	2	3
BiORAM-SGX	4.30%	17.95%	0.55%	1.08%	5.45%	1.66%	48.43%	82.95%	0	2
bolos-enclave	6.71%	7.85%	1.17%	0.48%	4.86%	4.01%	40.10%	84.09%	0	0
ehsm	3.69%	16.91%	3.81%	15.00%	76.97%	81.60%	0%	91.79%	0	12
sgx-reencrypt	8.60%	33.31%	14.92%	31.26%	20.26%	28.26%	84.38%	100.00%	2	4
SGXCryptoFile	5.85%	17.62%	15.04%	80.56%	4.15%	5.88%	0%	100.00%	0	2
trusted-function-frame	2.53%	1.97%	2.13%	1.53%	75.64%	75.22%	0%	100.00%	0	3
wasm-micro-runtime	3.95%	1.67%	2.08%	0.94%	32.64%	46.04%	78.04%	100.00%	5	15
average	4.57%	16.53%	6.83%	23.54%	19.26%	49.21%	33.29%	97.94%	5.25	8.05

Fuzzing-optimized SDK Brings Acceleration

The fuzzing-optimized SDK is **6.91x** faster than the hardware-mode SDK, while the simulation-mode SDK is only 2.67x faster.

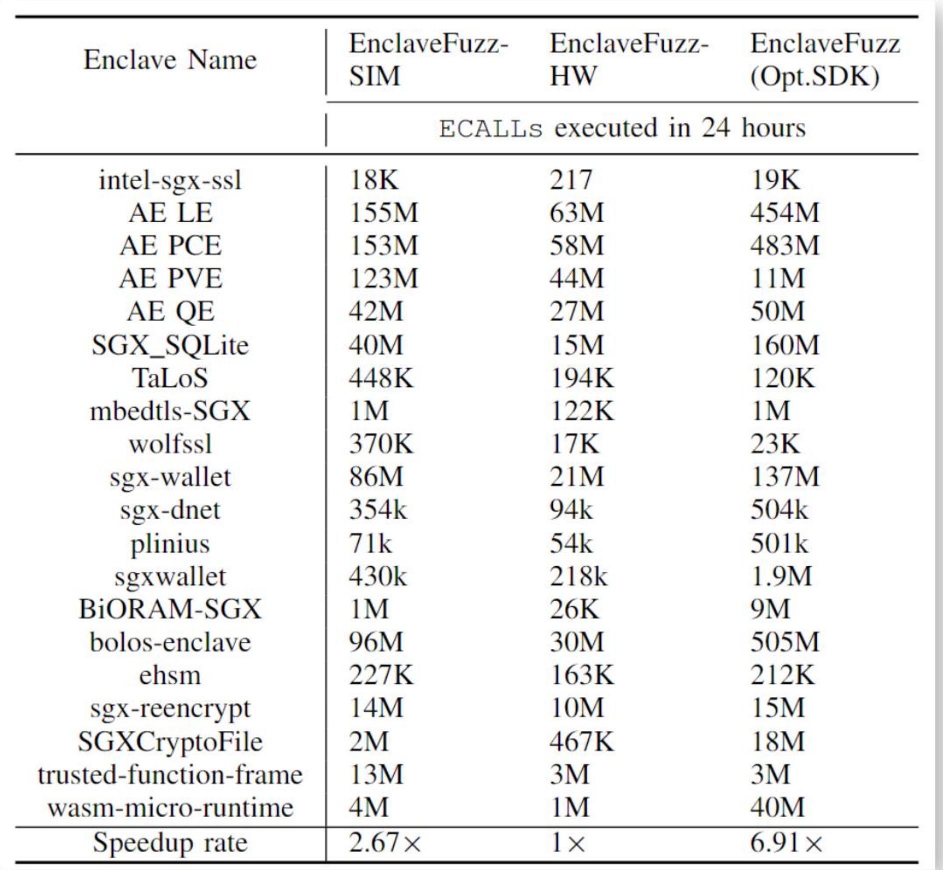


Enclave Name	EnclaveFuzz-SIM	EnclaveFuzz-HW	EnclaveFuzz (Opt.SDK)
	ECALLs executed in 24 hours		
intel-sgx-ssl	18K	217	19K
AE LE	155M	63M	454M
AE PCE	153M	58M	483M
AE PVE	123M	44M	11M
AE QE	42M	27M	50M
SGX_SQLite	40M	15M	160M
TaLoS	448K	194K	120K
mbedtls-SGX	1M	122K	1M
wolfssl	370K	17K	23K
sgx-wallet	86M	21M	137M
sgx-dnet	354k	94k	504k
plinius	71k	54k	501k
sgxwallet	430k	218k	1.9M
BiORAM-SGX	1M	26K	9M
bolos-enclave	96M	30M	505M
ehsm	227K	163K	212K
sgx-reencrypt	14M	10M	15M
SGXCryptoFile	2M	467K	18M
trusted-function-frame	13M	3M	3M
wasm-micro-runtime	4M	1M	40M
Speedup rate	2.67×	1×	6.91×

Fuzzing-optimized SDK Brings Acceleration

The fuzzing-optimized SDK is **6.91x** faster than the hardware-mode SDK, while the simulation-mode SDK is only 2.67x faster.

See paper for more ablation studies.



Enclave Name	EnclaveFuzz-SIM	EnclaveFuzz-HW	EnclaveFuzz (Opt.SDK)
	ECALLs executed in 24 hours		
intel-sgx-ssl	18K	217	19K
AE LE	155M	63M	454M
AE PCE	153M	58M	483M
AE PVE	123M	44M	11M
AE QE	42M	27M	50M
SGX_SQLite	40M	15M	160M
TaLoS	448K	194K	120K
mbedtls-SGX	1M	122K	1M
wolfssl	370K	17K	23K
sgx-wallet	86M	21M	137M
sgx-dnet	354k	94k	504k
plinius	71k	54k	501k
sgxwallet	430k	218k	1.9M
BiORAM-SGX	1M	26K	9M
bolos-enclave	96M	30M	505M
ehsm	227K	163K	212K
sgx-reencrypt	14M	10M	15M
SGXCryptoFile	2M	467K	18M
trusted-function-frame	13M	3M	3M
wasm-micro-runtime	4M	1M	40M
Speedup rate	2.67×	1×	6.91×

Takeaway

EnclaveFuzz is a multi-dimensional structure-aware fuzzer for SGX applications with a fuzzing-optimized SGX SDK and an SGX-specified sanitizer.

https://github.com/LeoneChen/EnclaveFuzz

https://netsec.ccert.edu.cn/vul337