Pandora: Principled Symbolic Validation of Intel SGX Enclave Runtimes

Fritz Alder[1], Lesly-Ann Daniel[1], David Oswald[2], Frank Piessens[1], Jo Van Bulck[1]

45th IEEE Symposium on Security and Privacy (S&P) – May 22, 2024

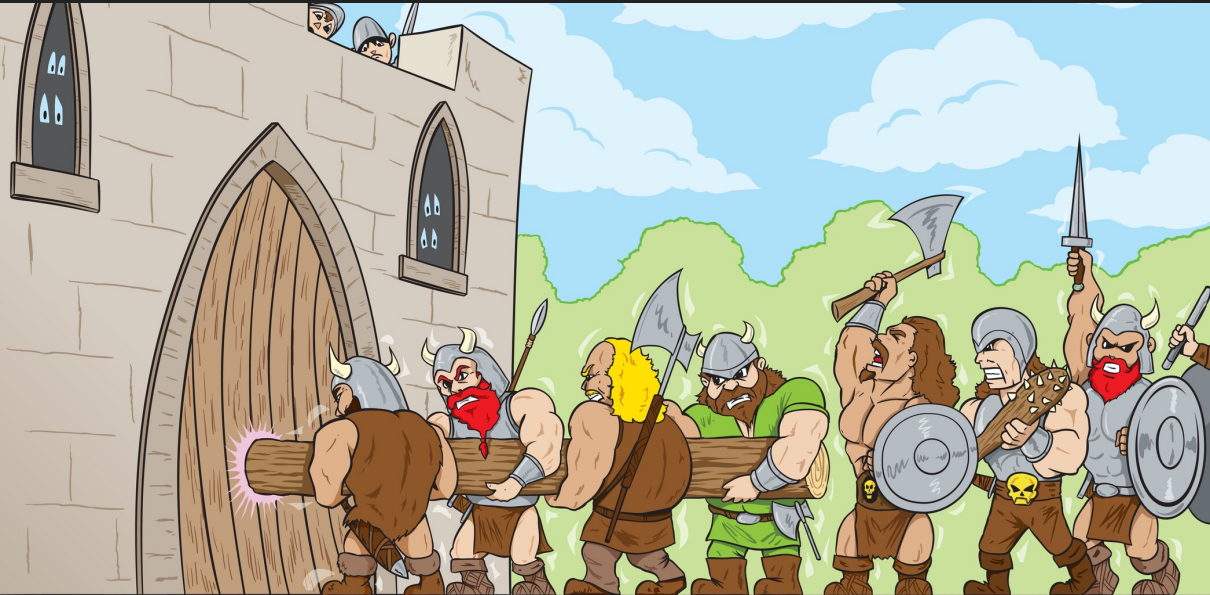
[1]DistriNet, KU Leuven, Belgium [2]University of Birmingham, UK

UNIVERSITY OF BIRMINGHAM

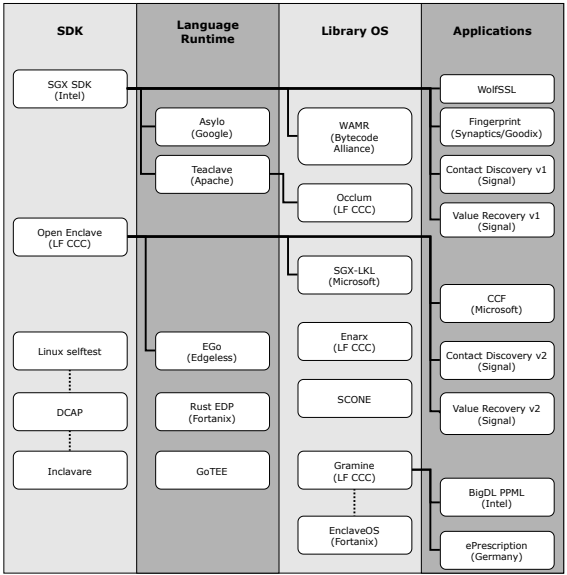
DistriNet

KU LEUVEN

Besieging the SGX Fortress: Software Interface Attacks

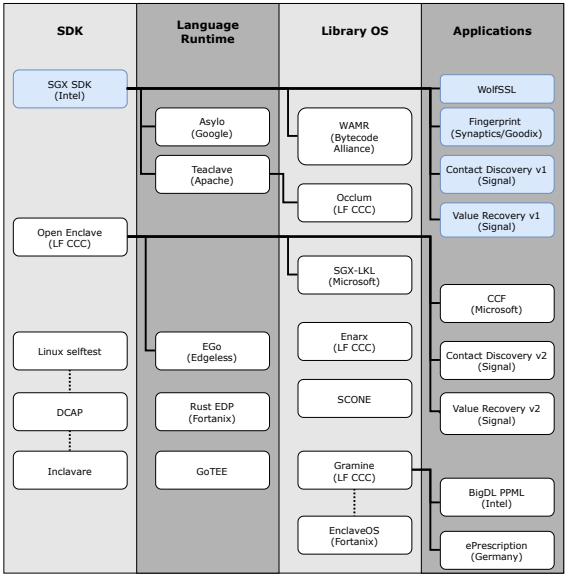


Challenge: Diverse Intel SGX Software Ecosystem



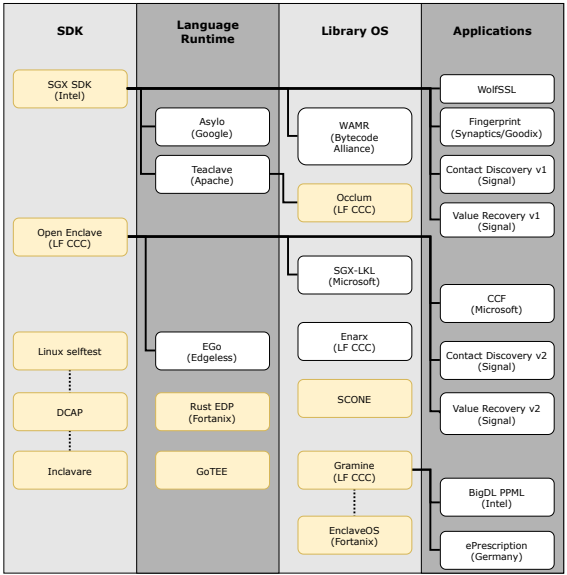
- **Ecosystem:** Diverse programming paradigms & abstractions

Challenge: Diverse Intel SGX Software Ecosystem



- **Ecosystem:** Diverse programming paradigms & abstractions
- **Prior work:** Selected applications on Intel SDK (e.g., `NULL` pointers)

Challenge: Diverse Intel SGX Software Ecosystem

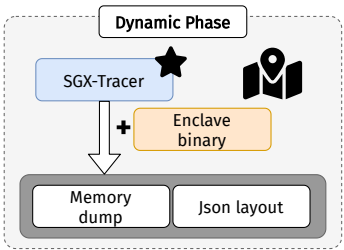


- **Ecosystem:** Diverse programming paradigms & abstractions
- **Prior work:** Selected applications on Intel SDK (e.g., `NULL` pointers)
- **Pandora:** Runtime-agnostic & truthful symbolic execution
  1. Exact attested memory binary
  2. Vulnerability detection plugins

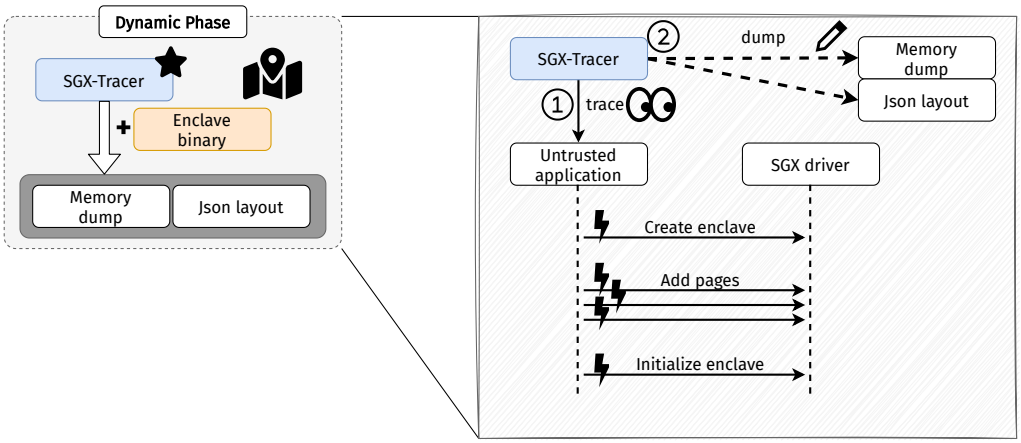
1. Truthful Symbolic Execution

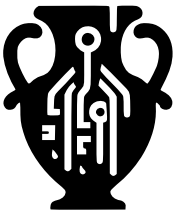
Pandora: Runtime-Agnostic Enclave Loading

Pandora: Runtime-Agnostic Enclave Loading



Pandora: Runtime-Agnostic Enclave Loading





2. Pluggable Vulnerability Detection

Pandora: Principled Symbolic Validation?

1. Extend `angr` with **enclave-aware breakpoints**
2. Validate **software invariants** during symbolic exploration!
3. Aggregate violations in human-readable rich **HTML reports**

Pandora: Principled Symbolic Validation?

1. Extend `angr` with **enclave-aware breakpoints**
2. Validate **software invariants** during symbolic exploration!
3. Aggregate violations in human-readable rich **HTML reports**

Challenge: Understanding attacks + specifying adequate invariants:

- API: Tainted *pointers*
- Control flow: Tainted *jumps*
- ABI: Tainted CPU *control registers*
- MMIO/ÆPIC: Cleansing + *alignment*

Experimental Results: > 200 New Vulnerable Code Locations

Runtime	Version	Prod	Src	Plugin	Instances
EnclaveOS	3.28	✓	✗ †	ABISan	1
EnclaveOS	3.28	✓	✗ †	PTRSan	15
EnclaveOS	3.28	✓	✗ †	ÆPICSan	33
EnclaveOS	3.28	✓	✗ †	CFSan	2
GoTEE	b35f	✗	✓	PTRSan	31
GoTEE	b35f	✗	✓	ÆPICSan	18
GoTEE	b35f	✗	✓	CFSan	1
Gramine	1.4	✓	✓	ABISan	1
Intel SDK	2.15.1	✓	✓	PTRSan	2
Intel SDK	2.19	✓	✓	ÆPICSan	22
↪ Occlum	0.29.4	✓	✓	ÆPICSan	11
Open Enclave	0.19.0	✓	✓	ABISan	2
Rust EDP	1.71	✓	✓	ABISan	1

Runtime	Version	Prod	Src	Plugin	Instances
Linux selftest	5.18	✗	✓	ABISan	1
↪ DCAP	1.16	✓	✓	ABISan	1
↪ Inclavare	0.6.2	✗	✓	ABISan	1
Linux selftest	5.18	✗	✓	PTRSan	5
↪ DCAP	1.16	✓	✓	PTRSan	17
↪ Inclavare	0.6.2	✗	✓	PTRSan	2
Linux selftest	5.18	✗	✓	CFSan	1
↪ Inclavare	0.6.2	✗	✓	CFSan	1
SCONE	5.7 / 5.8	✓	✗	ABISan	2 / 1
SCONE	5.7 / 5.8	✓	✗	PTRSan	10 / 3
SCONE	5.7 / 5.8	✓	✗	ÆPICSan	11 / 3
SCONE	5.8	✓	✗	CFSan	1

Issues reported at 0x22c3 1 do_encl_op_get_from_unmeasured CRITICAL Unconstrained read

Unconstrained read CRITICAL RIP=0x22c3

Plugin extra info

Key	Value
Address	<BV64 0x3000 + ((attacker_mem_66_32{UNINITIALIZED} .. 0x1) << 0x3)>
Attacker tainted	True
Length	8
Pointer range	[0x3008, 0xffffffff800003008]
Pointer can wrap address space	False
Pointer can lie in enclave	True
Extra info	Read address may lie inside or outside enclave

Execution state info

Disassembly

CPU registers

Backtrace

Basic block trace (most recent first)

Conclusions and Outlook

github.com/
pandora-tee

Truthful: Runtime-agnostic enclave memory model

→ *Exact attested memory layout (`MRENCLAVE`)*

Extensible: Validate vulnerability invariants via plugins

→ *ABISan, PTRSan, ÆPICSan, CFSan*

Evaluation: > 200 instances; 7 CVEs; 11 SGX runtimes

→ *Including low-level initialization & relocation logic!*