KU LEUVEN

Validating Sancus Enclaves using Symbolic Execution

Gert-Jan Goossens

26 June 2024

Supervisors: Prof. dr ir. Frank Piessens
& Dr. ir. Jo Van Bulck

Assessor: Dr. Vera Rimmer

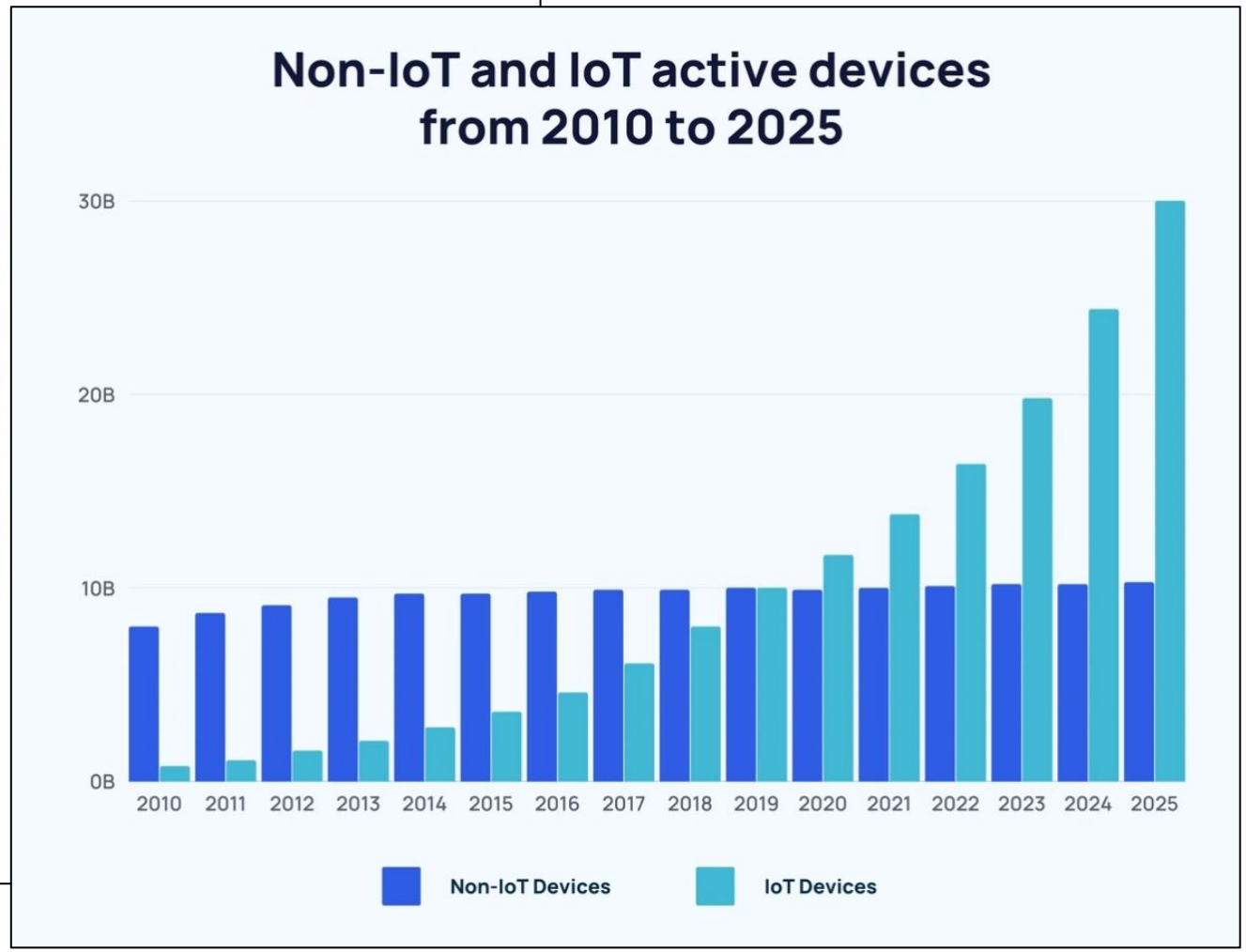
Mentor: Dr. ir. Fritz Alder

Casino Gets Hacked Through Its Internet-Connected Fish Tank Thermometer

Apr 16, 2018 Wang Wei

Internet-connected technology, also known as the Internet of Things (IoT), is now part of daily life, with smart assistants like Siri and Alexa to cars, watches, toasters, fridges, thermostats, lights, and the list goes on and on.

But of much greater concern, enterprises are unable to secure each and every device on their network, giving cybercriminals hold on their network hostage with just one insecure device.



https://explodingtopics.com/blog/iot-stats
https://thehackernews.com/2018/04/iot-hacking-thermometer.html

Three States of Data

AT REST

ENCRYPTION
e.g. disk encryption

IN TRANSIT

ENCRYPTION
e.g. TLS, SSL

IN USE

Trusted Execution Environments

Trusted Execution Environments (TEE)

- **Hardware** adaptation in CPU
- **Software** developed **on top**

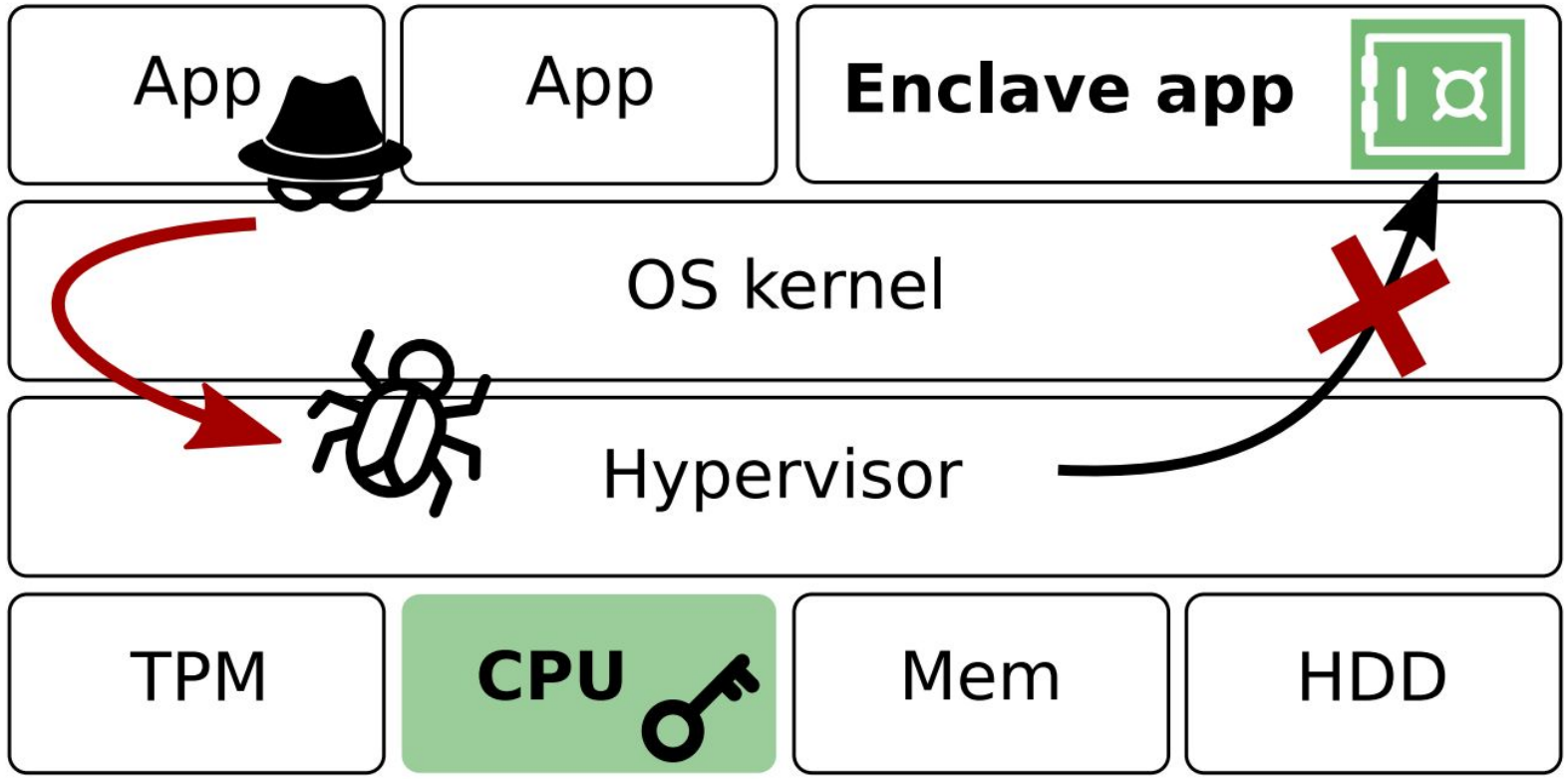


Image: Van Bulck, The Hitchhiker's Guide to Subverting Intel SGX Enclaves, https://jovanbulck.github.io/files/circuit22.pdf

Confidential computing with Intel® Software Guard Extensions (SGX)

Last updated 2024-06-17

Confidential computing with Intel® Software Guard Extensions (SGX) protects your data through hardware-based server security by using isolated memory regions that are known as encrypted enclaves. This hardware-based computation helps protect your data from disclosure or modification. Which means that your sensitive data is encrypted while it is in virtual server instance memory by allowing applications to run in private memory space. To use SGX, you must install the SGX drivers and platform software on SGX-capable worker nodes. Then, design your app to run in an SGX environment. For more information about confidential computing and IBM Cloud®, see Getting started with confidential computing.

Confidential computing with SGX

When you use confidential computing with SGX, your data is protected through the entire compute lifecycle. Which means that your data is accessible only to authorized code and is invisible to anyone or anything else, including the operating system and even IBM Cloud®.

Microsoft Selects Azure Confidential Computing Using Intel® SGX

Microsoft now hosts their e-commerce payment services on Azure Confidential Computing in the public cloud, having already moved US$25 billion in annual credit card transactions to the cloud as of November 2023.

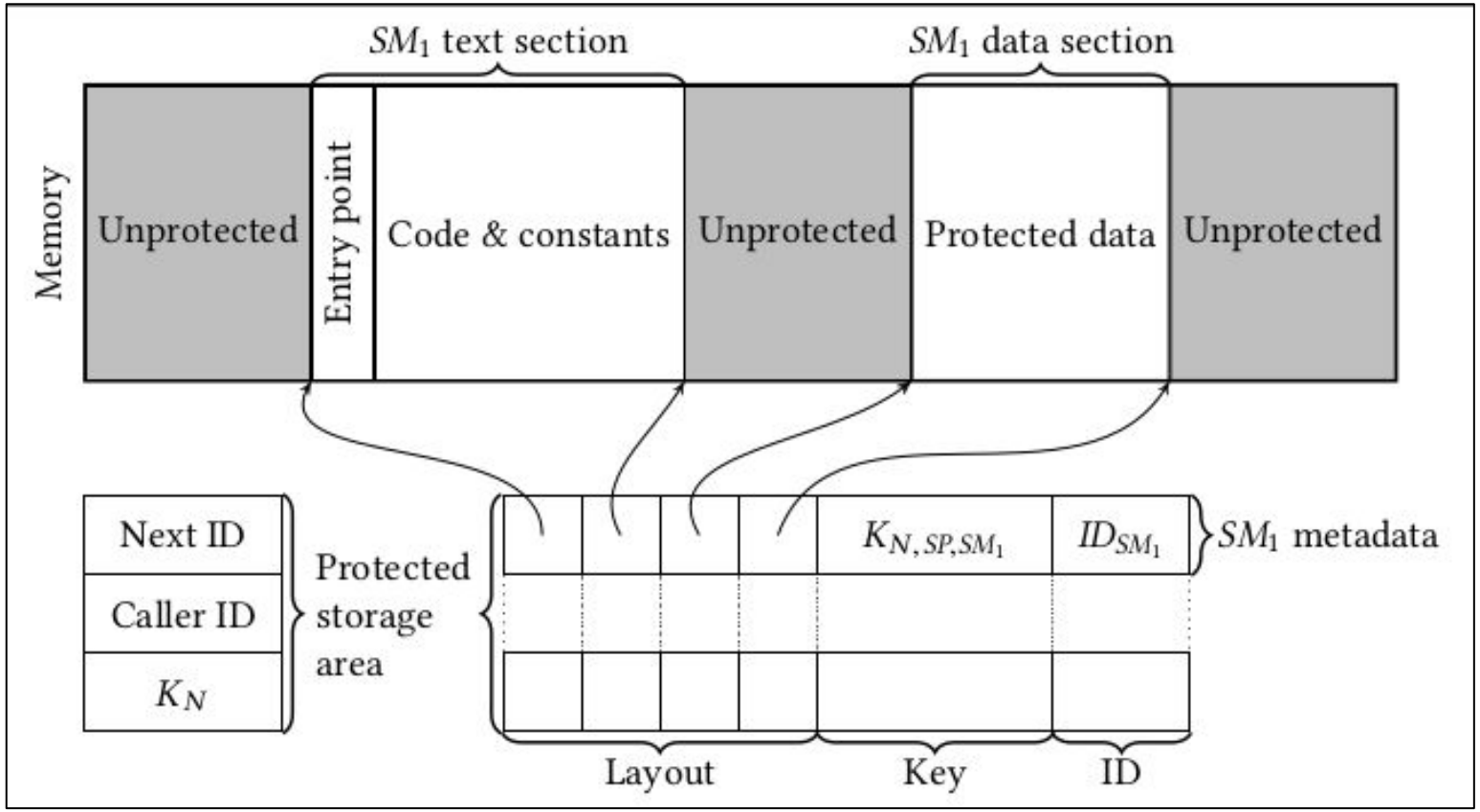
To help meet or exceed current PCI DSS standards, Microsoft uses Azure confidential computing and Intel® SGX application enclaves running Azure Kubernetes Service node pools.

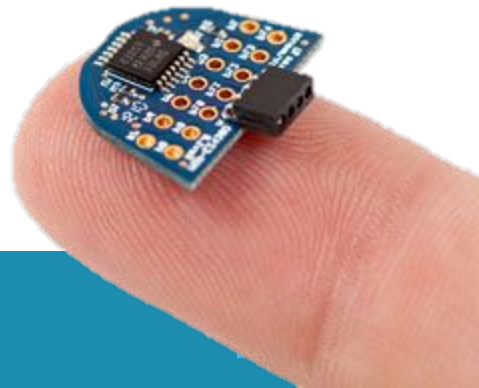
Learn why Microsoft trusts Azure with Intel® SGX

https://www.intel.com/content/www/us/en/products/docs/accelerator-engines/software-guard-extensions.html
https://cloud.ibm.com/docs/bare-metal?topic=bare-metal-bm-intel-sgx

Sancus

- Low-end
- Research TEE
- MSP430 architecture
- 16-bit





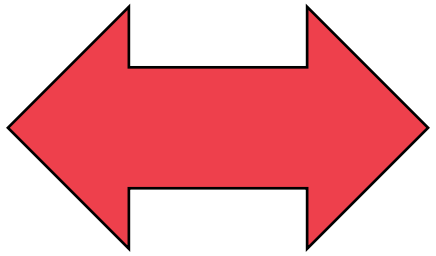
Noorman et al. "Sancus 2.0: A Low-Cost Security Architecture for IoT Devices." (2017).

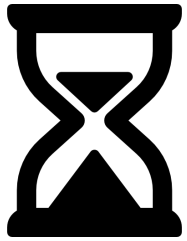
‘A Tale of Two Worlds’ (2019)

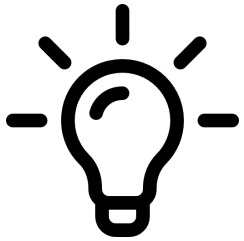
Vulnerability \ Runtime		SGX-SDK	OpenEnclave	Graphene	SGX-LKL	Rust-EDP	Asylo	Keystone	Sancus
		Intel SGX							
Tier1 (ABI)	#1 Entry status flags sanitization	★	★	◐	●	◐	●	○	○
	#2 Entry stack pointer restore	○	○	★	●	○	○	○	★
	#3 Exit register leakage	○	○	○	★	○	○	○	○
Tier2 (API)	#4 Missing pointer range check	○	★	★	★	○	●	○	★
	#5 Null-terminated string handling	☆	★	○	○	○	○	○	○
	#6 Integer overflow in range check	○	○	●	○	●	○	●	●
	#7 Incorrect pointer range check	○	○	●	○	○	●	○	●
	#8 Double fetch untrusted pointer	○	○	●	○	○	○	○	○
	#9 Ocall return value not checked	○	★	★	★	○	●	★	○
	#10 Uninitialized padding leakage	[23]	★	○	●	○	●	★	★

Van Bulck et al. “A Tale of Two Worlds: Assessing the Vulnerability of Enclave Shielding Runtimes”, CCS 2019

Vulnerability Analysis







- TeeRex
- Guardian
- SymGX
- Coin
- **Pandora**

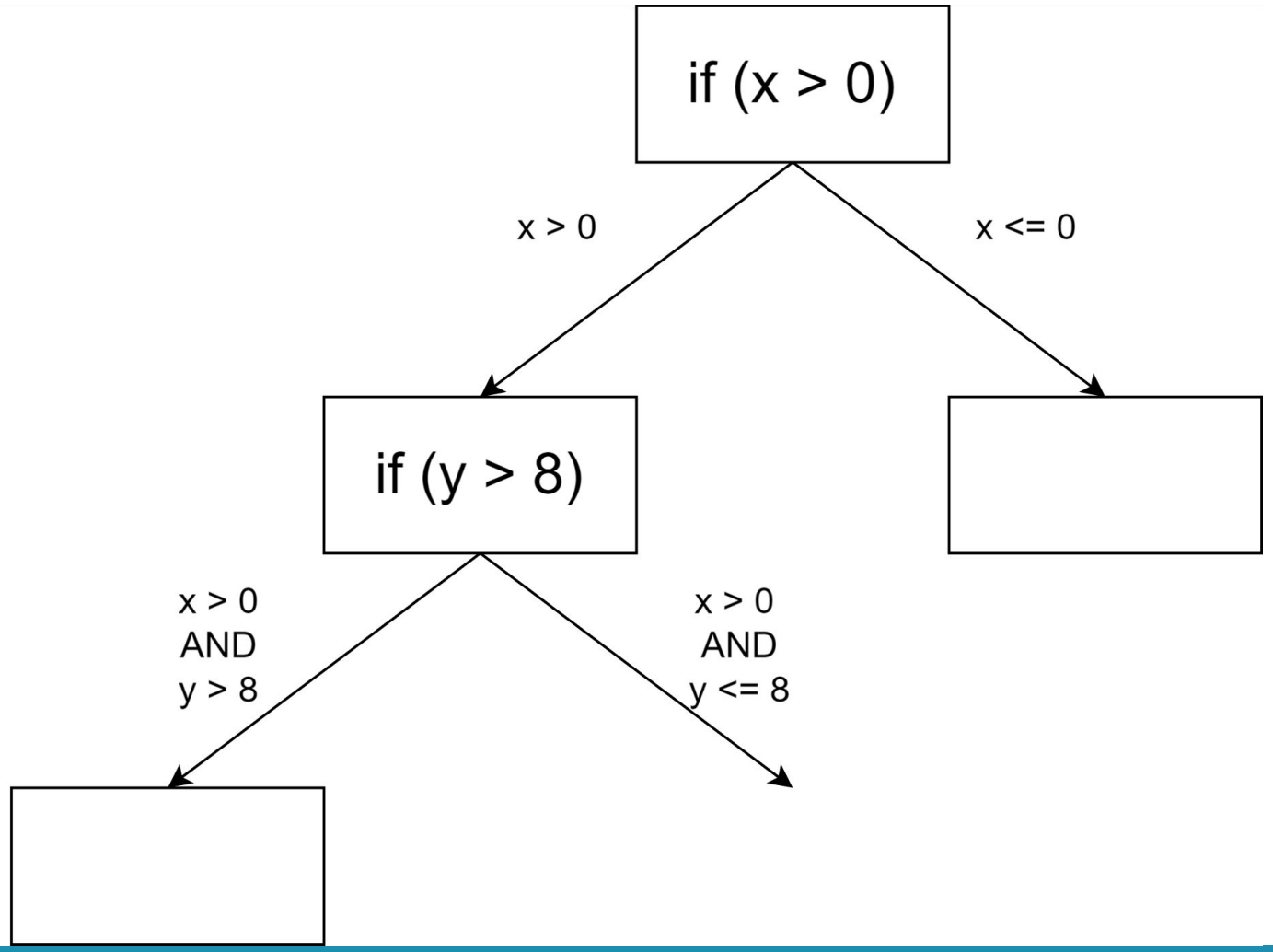
Cloosters et al. "TeeRex: Discovery and Exploitation of Memory Corruption Vulnerabilities in SGX Enclaves", Usenix '20.
Antonino et al. "Guardian: Symbolic validation of orderliness in sgx enclaves.", CCSW '21
Wang et al. "SymGX: Detecting Cross-boundary Pointer Vulnerabilities of SGX Applications via Static Symbolic Execution" CCS '23
Khandaker et al. "Coin attacks: On insecurity of enclave untrusted interfaces in sgx.", ASPLOS '20
Alder et al. "Pandora: Principled Symbolic Validation of Intel SGX Enclave Runtimes", IEEE S&P 2024

https://easydrawingguides.com/how-to-draw-a-bridge/

Pandora

Symbolic Execution

```
if(x > 0):
        ...
        if (y > 8):
                ...
        ...
...
```



KU LEUVEN

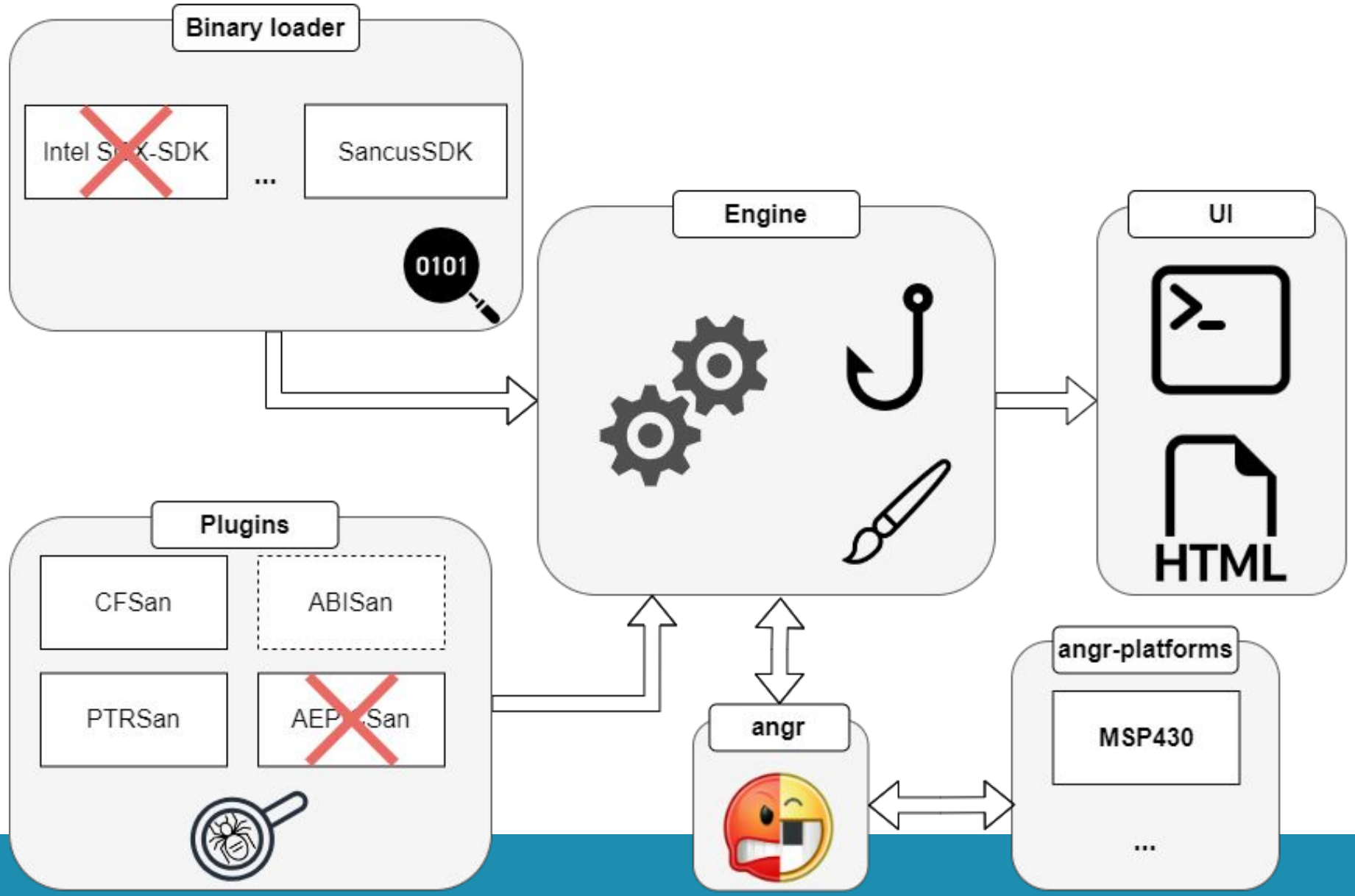
Pandora (2024)

- **angr**
  - **Binary** analysis
  - **Symbolic execution** framework

- **Intel SGX**
- **Taint tracking**
- **Plugin-based**

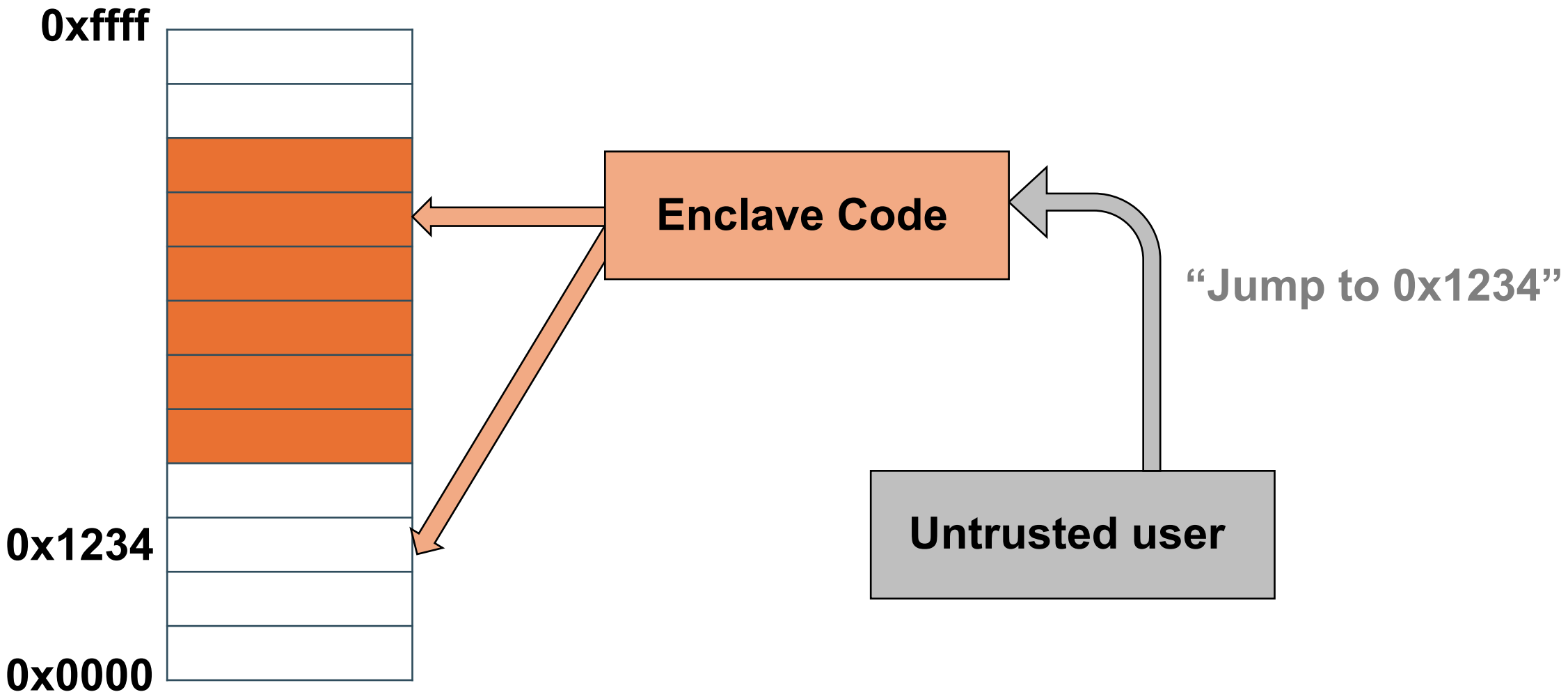
Alder et al. “Pandora: Principled Symbolic Validation of Intel SGX Enclave Runtimes”, IEEE S&P 2024

Pandora-Sancus

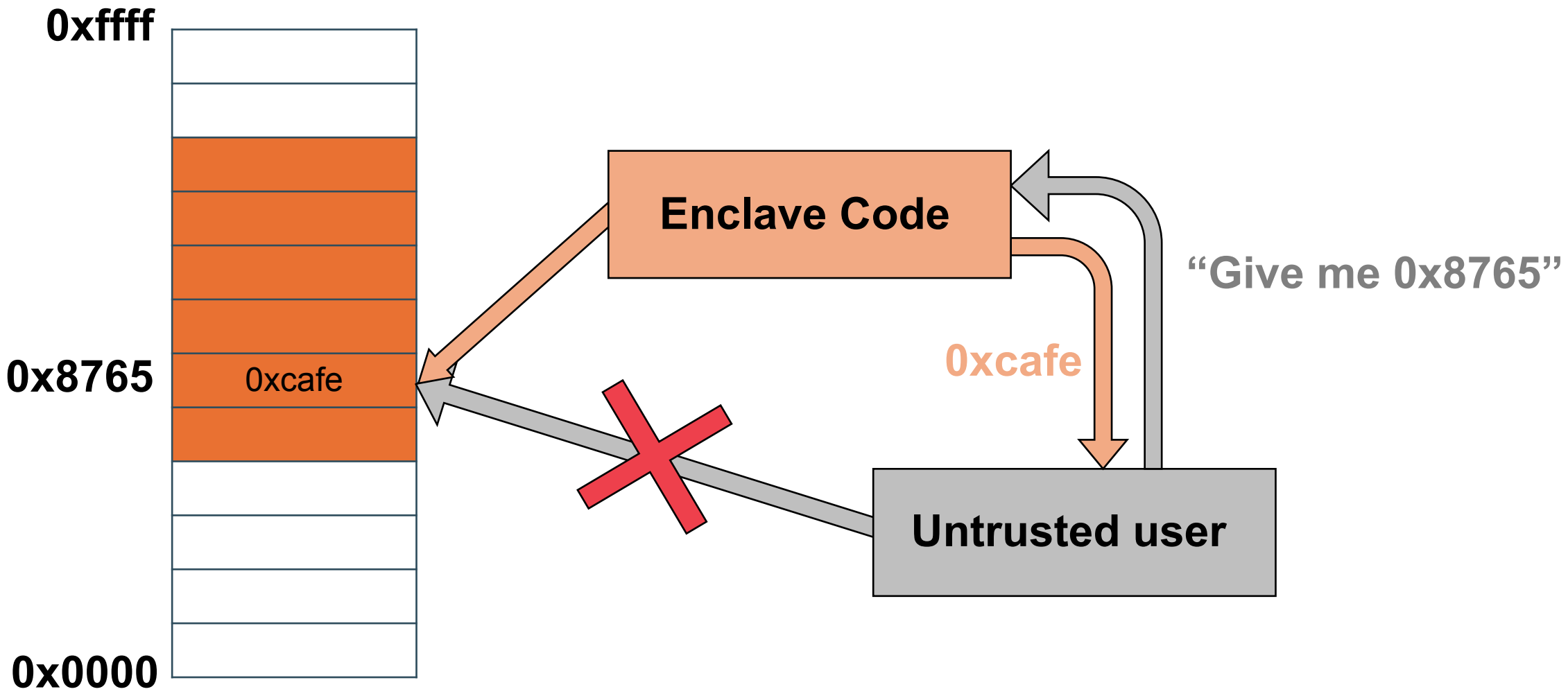


KU LEUVEN

CFSan vulnerabilities



PTRSan vulnerabilities



Evaluation

Unit Test Framework

CFSan Tests

- 21 assembly testcases

```
.text
__sm_foo_public_start:
enter_foo:
    br r15

__sm_foo_public_end:
    ret

.data
__sm_foo_secret_start:
__sm_foo_secret_end:
```

PTRSan Tests

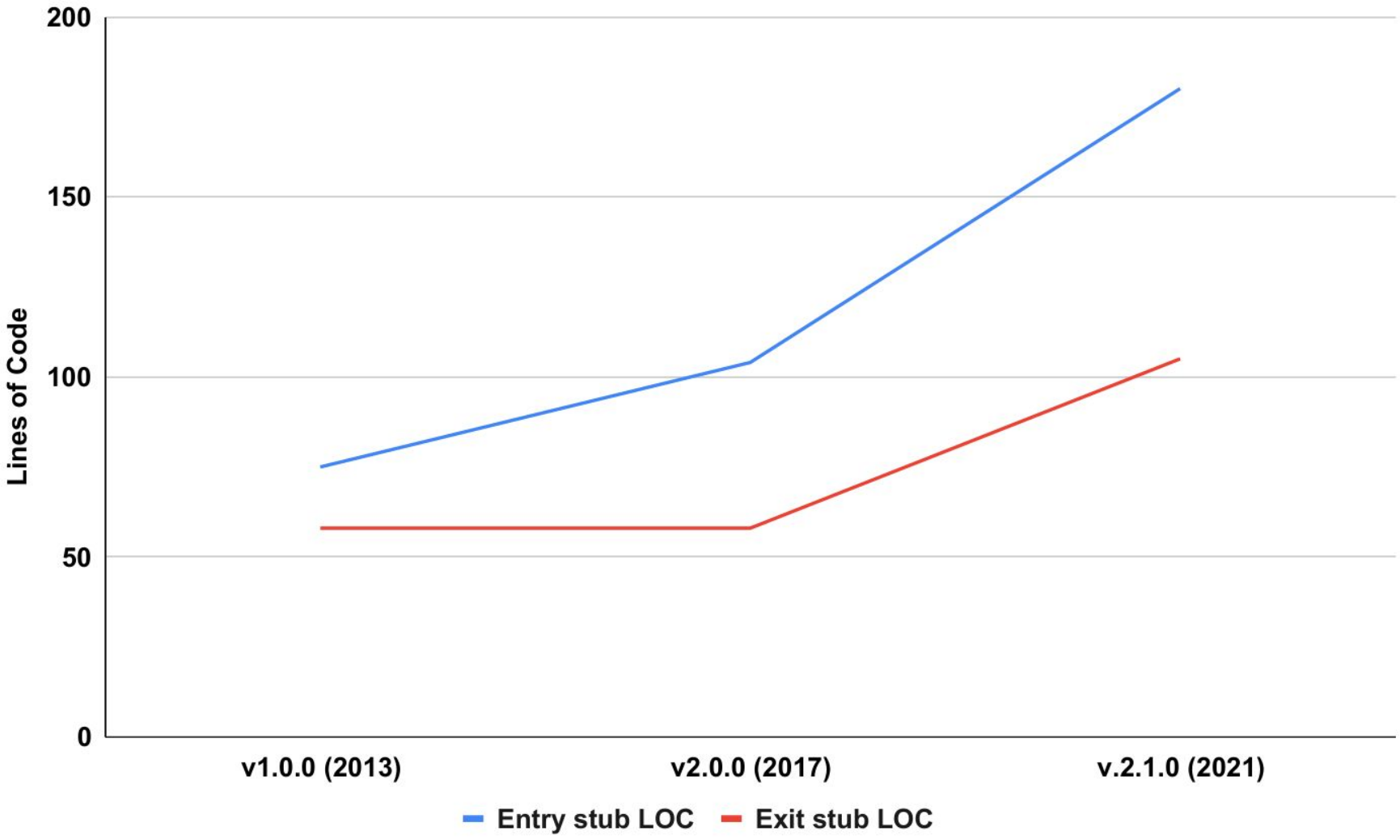
- 15 assembly testcases

```
.text
__sm_foo_public_start:
enter_foo:
    pop r13
    jmp __sm_foo_public_end

__sm_foo_public_end:
    ret

.data
__sm_foo_secret_start:
__sm_foo_secret_end
```

Sancus Stub Sizes



KU LEUVEN

‘A Tale of Two Worlds’ (2019)

Vulnerability \ Runtime		SGX-SDK	OpenEnclave	Graphene	SGX-LKL	Rust-EDP	Asylo	Keystone	Sancus
Tier1 (ABI)	#1 Entry status flags sanitization	★	★	◐	●	◐	●	○	○
	#2 Entry stack pointer restore	○	○	★	●	○	○	○	★
	#3 Exit register leakage	○	○	○	★	○	○	○	○
Tier2 (API)	#4 Missing pointer range check	○	★	★	★	○	●	○	★
	#5 Null-terminated string handling	⯪	★	○	○	○	○	○	○
	#6 Integer overflow in range check	○	○	●	○	●	○	●	●
	#7 Incorrect pointer range check	○	○	●	○	○	●	○	●
	#8 Double fetch untrusted pointer	○	○	●	○	○	○	○	○
	#9 Ocall return value not checked	○	★	★	★	○	●	★	○
	#10 Uninitialized padding leakage	[23]	★	○	●	○	●	★	★

Van Bulck et al. “A Tale of Two Worlds: Assessing the Vulnerability of Enclave Shielding Runtimes”, CCS 2019

Sancus Stubs: CFSan

Issues reported at 0x6c66 1 __sm_basic_enclave_entry CRITICAL Symbolic unconstrainted tainted jmp target

Symbolic unconstrainted tainted jmp target CRITICAL IP=0x6c66

Plugin extra info

Key	Value
Target	<BV16 r7_attacker_7_16{UNINITIALIZED}>
Attacker tainted	True
Symbolic	True
Target range	[0x0, 0xffff]
Target entirely inside enclave	False

Execution state info

Disassembly

```
6c60:    82 41 02 03    mov    r1,    &0x0302
6c64:    36 43          mov    #-1,   r6      ;r3 As==11
6c66:    00 47          br     r7
```


Van Bulck et al. “A Tale of Two Worlds: Assessing the Vulnerability of Enclave Shielding Runtimes”, CCS 2019

Sancus Stubs: PTRSan

Issues reported at 0x6cbc 1 _sm_basic_enclave_ret_entry CRITICAL Non-tainted read outside enclave

Non-tainted read outside enclave CRITICAL IP=0x6cbc

Plugin extra info

Key	Value
Address	<BV64 0x0>
Attacker tainted	False
Length	6
Pointer range	[0x0, 0x0]
Pointer can wrap address space	False
Pointer can lie in enclave	False

Execution state info

Disassembly

```
6cac:    3b 41    pop    r11
6cae:    3a 41    pop    r10
6cb0:    39 41    pop    r9
6cb2:    35 41    pop    r5
6cb4:    34 41    pop    r4
6cb6:    38 41    pop    r8
6cb8:    37 41    pop    r7
6cba:    36 41    pop    r6
6cbc:    30 41    ret
```

Pop under private stack

Van Bulck et al. “A Tale of Two Worlds: Assessing the Vulnerability of Enclave Shielding Runtimes”, CCS 2019

Wrap-up

Future Works

- Implementation of **hooks**, **reentry & ABISan** plugin for **Sancus**
- **Merging** *Pandora-Sancus* and *Pandora-SGX*
- Validate '**Secure Linking**'
- Validate **'real-time' guarantees**

- Other **MSP430 TEE architectures** e.g.
  - IPE
  - VRASED
  - SMART

Alder et al. "Aion: Enabling open systems through strong availability guarantees for enclaves", CCS '21
Bognár et al. "Mind the gap: Studying the insecurity of provably secure embedded trusted execution architectures.", IEEE S&P '22
Bognár et al. "Intellectual property exposure: Subverting and securing intellectual property encapsulation in texasinstruments microcontrollers", USENIX '24

Conclusion

⇒ **Techniques in Pandora** are **not exclusive** for Intel SGX and can be **generalized**

- **Sancus support** for **Pandora**
- **MSP430 angr backend** contributions
- **Extensive unit test framework** of 36 assembly test cases
- **Autonomous discovery** of 4 vulnerabilities described in **T2W**

Thank you for your attention!
Questions?

https://github.com/Gert-JanG/pandora-sancus

Appendix

Sancus vs. Intel SGX

Sancus

Intel SGX

= Multiple enclaves
= Entry points
= Single address space

- Low-end
- Research TEE
- MSP430
- Enter/Exit with regular instructions
- Linked enclaves

- High-end
- Production TEE
- X86
- Enter/Exit instructions
- Isolated enclaves

Hooks

Wrapper	Instruction	Hooked method	Hook Implemented
sancus_disable	SM_DISABLE	SimUnprotect	Partial
sancus_enable	SM_ENABLE	SimProtect	✗
sancus_verify_address	SM_VERIFY_ADDR	SimAttest	✗
sancus_wrap	SM_AE_WRAP	SimEncrypt	✗
sancus_unwrap	SM_AE_UNWRAP	SimDecrypt	Partial
sancus_get_id	SM_ID	SimGetID	Hardcoded: return 0 & constrain r7
sancus_get_caller_id	SM_CALLER_ID	SimGetCallerID	Hardcoded: return 0
sancus_stack_guard	SM_STACK_GUARD	SimNop	✗
N/A	SM_CLIX	SimNop	✗

MSP430 angr Backend

- angr includes x86 ⟶ MSP430 in ***angr-platforms***

  ↳ **Less mature!**

- Adaptations to: `br, ret, rra, rrc` instructions

- **No** proper **disassembly** support

Plugins

CFSan:

No jump to:

1. Non-executable memory
2. Arbitrary attacker-tainted address
3. Attacker-tainted address inside enclave
4. No explicit enclave leave

PTRSan

No read/write from/to:

1. Attacker-tainted address inside or outside enclave
2. Attacker-tainted address inside enclave
3. Non-tainted address outside enclave

- ABISan

1. End of ABI-Sanitization: all registers sanitized
2. Specified registers contain attacker-tainted values (e.g. SP)

Implicit Exit

```
1  __sm_foo_public_start:
2      nop
3      nop
4      mov #0xdead, r15
5
6  __sm_foo_public_end:
7      nop
8      nop
9      ret
```

1 basic block

```
1  __sm_foo_public_start:
2      nop
3      nop
4      mov #0xdead, r15
5      jmp __sm_foo_public_end
6
7  __sm_foo_public_end:
8      nop
9      nop
10     ret
```

2 basic blocks

Crashed States

```
.Lerror:
    ; caller provided poisoned arguments -> trigger an intentional
    ...
    mov #0, &__sm_ssa_base_addr
    mov #1, &__sm_entry
    ; should never reach here
1:
    jmp 1b
```

Limitations

- **Linked** enclaves
- No enclave **reentry**
- Incomplete **hooks**
- No **ABISan**
- Pandora-SGX limitations
  - angr sound
  - Encrypted code
  - (Path explosion)

Credits

- Slide 4:
  - https://www.flaticon.com/free-icon/man-working-on-a-laptop-from-side-view_49728?term=computer&page=1&position=51&origin=search&related_id=49728 designed by Freepik from Flaticon
  - https://www.flaticon.com/free-icon/folder_545336?term=folder&page=1&position=6&origin=search&related_id=545336 designed by Freepik from Flaticon
  - https://www.flaticon.com/free-icon/key_807292?term=key&page=1&position=2&origin=search&related_id=807292 designed by Freepik from Flaticon
- Slide 7: MSP430 device image: https://martybugs.net/electronics/msp430/
- Slide 8, 9, 14:
  - https://www.flaticon.com/free-icon/automated-process_4176850?term=automation&page=1&position=61&origin=search&related_id=4176850&k=1718712249009&sign-up=google designed by surang from Flaticon
  - https://www.flaticon.com/free-icon/people_14982203?term=human+analysing&page=1&position=2&origin=search&related_id=14982203 designed by juicy_fish from Flaticon
  - https://www.flaticon.com/free-icon/hourglass_483610?term=hourglass&page=1&position=4&origin=search&related_id=483610 designed by Those Icons from Flaticon
  - https://www.flaticon.com/free-icon/idea_566359?term=idea&page=1&position=4&origin=search&related_id=566359 designed by Freepik from Flaticon
- Slide 15: fishing hook: https://www.flaticon.com/free-icon/fishing_818953?term=fishing+hook&page=1&position=1&origin=search&related_id=818953 designed by Freepik from Flaticon