SMILE: Secure Memory Introspection for Live Enclave

IEEE S&P 2022

Lei Zhou¹, Xuhua Ding², Fengwei Zhang¹

¹ Southern University of Science and Technology, China
² Singapore Management University, Singapore
Outline

• Motivation of SMILE
• Overview of SMILE
• Design and Implementation
• Evaluation: Effectiveness and Performance
• Conclusion
Intel SGX is popularly deployed in current computing platform, especially in servers.

SGX provides a user-level trusted environment for security-sensitive code and data execution.

The question: does the SGX fully relieve the security concerns of users?
Why SGX Enclave Need Introspection?

Does the SGX fully relieve security concerns?

The Answer is **NO**

Intel SGX is under a series of attacks

- **spectre-attack-sgx**
- **CROSSTALK**: Speculative Data Leaks Across Cores Are Real
- **Plunder Volt**
- **CacheOut**: Stealing Data on Intel EPCC via Cache Exceptions
- **SGAxe**: Exploit SGX Flaws in Practice
- **SmashEx**: Smashing SGX Enclaves Using Exceptions

Cloud Platform

- **stack**
- **heap**
- **data**
- **code**

- secrets leakage
- fake attestation
Why SGX Enclave Need Introspection?

Intel SGX is under a series of attacks

- spectre-attack-sgx
- CROSSTALK: Speculative Data Leaks Across Cores Are Real
- CacheOut: Unlinking Data on Intel CPUs via Cache Flashes
- SmashEx: Smashing SGX Enclaves Using Exceptions
- PLUNDER: Under Attack: SGX in Practice

- .stack
- .heap
- .data
- .code

Cloud Platform

Code bugs or vulnerabilities

- secrets leakage
- fake attestation
- control flow hijacking
- code injection

More serious threats
How to Securely Introspect SGX Enclave?

Thus, users expect to check the enclave memory for attack diagnosis.

Memory introspection?

- Non-SGX Application
- SGX Application
Thus, users expect to check the enclave memory for attack diagnosis.

Memory introspection?

Non-SGX Application

SGX Application

Memory isolation

Outside and inner attacks

Untrusted host

...
Outline

• Motivation of SMILE

• Overview of SMILE

• Design and Implementation

• Evaluation: Effectiveness and Performance

• Conclusion
Secure Memory Introspection for Live Enclave (SMILE)

Meet following requirements:

✓ **Enclave authenticity** - introspection is upon the expected enclave.

✓ **Introspection genuineness** - introspection results are not faked by corrupted enclave code.

✓ **Security preserving** - introspection does not undermine the default enclave security.
Secure Memory Introspection for Live Enclave (SMILE)

Meet following requirements:

✓ **Enclave authenticity**-introspection is upon the expected enclave.

✓ **Introspection genuineness**-introspection results are not faked by corrupted enclave code.

✓ **Security preserving**-introspection does not undermine the default enclave security.

SMILE is designed to ensure the owner of an enclave – **and only the owner** – retrieves her enclave contents at runtime.

● **Focusing Scenario:** x86-based device with SMM and SGX hardware features.
Full overview of the SGX enclave execution and SMM location in the system
SMM for Introspection Assistance

SGX Neither deals with enclave software compromise nor supports runtime measurement

- Independent execution environment
- Halting and restoring host application
- Enable accessing the host memory and register value
SMM for Introspection Assistance

SGX Neither deals with enclave software compromise nor supports runtime measurement

- Independent execution environment
- Halting and restoring host application
- Enable accessing the host memory and register value

Securely control enclave execution environment
Workflow of SMILE

Introspection Steps:

I. Enclave owner sends introspection request to target platform.
II. Target OS passes the reference to SMM agent.
III. SMM agent interrogates enclave-inner introspection code.
IV. Enclave encrypts and passes the request memory to owner.

The SMM agent’s responsibility is to authenticate the enclave and assess the trustworthiness of the introspection code in enclave.
Outline

• Background of SGX
• Overview of SMILE
• Design and Implementation
• Evaluation: Performance and Effectiveness
• Conclusion
Design of SMILE

- **Deploy interrogation agent into SMM (SMM agent)**
  - build a confined environment for introspection.
  - pass the signature for introspection result encryption.

- **Add introspection code into enclave (Anchor thread)**
  - answer the integrity interrogation.
  - check the identity of the enclave.
  - achieve the request introspection memory.
Design of SMILE

With the Confined Environment, SMILE is expected to achieve **authenticity**, **genuineness**, and **security preserving** on introspection.

Interrogation session between SMM agent and anchor thread is to verify the introspection code in enclave.
Confined Environment for Introspection

Illustrates:
- Target $H$ having four CPU cores.
- Enclave occupies one core.
- SMM agent occupies all other cores.

One core runs to protected mode as enclave core

The state of CPU cores is controlled by SMM agent
The state of CPU cores is controlled by SMM agent.
Confinement Environment for Introspection

- Core in PM
- Core in SMM

Assisted interrogate anchor
Assisted recover execution environment

Designated Enclave

Confined interrogation
EPC introspection
Restrict the accessible memory page
Intercept the exception and interruption

The state of CPU cores is controlled by SMM agent.
Confined Interrogation

• Authenticity of Enclave is the prerequisite of a secure introspection
Confined Interrogation Protocol

Step 1: Anchor integrity checking

Random bytes fill in anchor and SSA page except the anchor instructions

Anchor is the first piece of code to run
Confined Interrogation Protocol

Step 2: Worker integrity checking

SMM agent
Confined Interrogation Protocol

Step 3: Enclave identity checking

The worker achieves the enclave id under the confined environment
EPC Introspection

Step 4: EPC introspection

Neither a corrupted SMM agent nor an imposter can exploit SMILE
Outline

• Motivation of SMILE
• Overview of SMILE
• Design and Implementation
• Evaluation: Performance and Effectiveness
• Conclusion
The Time Overhead for SMILE Zero-Load Introspection

Generally, SMILE takes 159.3 microseconds with zero-load introspection.
The Time Overhead for SMILE Introspection

One session of SMILE introspection:

- Interrogation \( 159.3 \mu s \)
- RSA encryption \( 121.7 \mu s \)
- 1-page AES encryption \( 2.1 \mu s \)

For \( n \)-page task, it costs \((281n/r +2.1n) \mu s\), \( r \) pages for each session.

Sharing the AES key in multi-sessions with one-time RSA, costs \((121.7+159.3n/r+2.1n) \mu s\)
Applications of SMILE

The attack swaps the conditions for two branches in the enclave, and SMILE finds the modifications at runtime.

SMILE dumps the SSA frame data to verify if the saved register value is right.
Applications of SMILE – Stack Checking

Enclave

Internal attack, e.g., ROP

User

Similarly, the stack might be attacked by malware for code injection, e.g., ROP.

SMILE dumps the stack frame data to verify if any trace of attack.
Applications of SMILE – Enclave Location Verification

SGX attestation mechanism cannot perfectly verify the enclave location

The owner initiates a SMILE session to introspect the shared secret key. The outcome is binding to trusted interrogation and can confirm if the enclave is in target.
SMILE empowers an enclave owner to collect on-demand runtime data from her enclave under a software exploitation attack.
Thank you!

zhoul6@sustech.edu.cn