SMILE: Secure Memory Introspection for Live Enclave

IEEE S&P 2022

Lei Zhou[1], Xuhua Ding[2], Fengwei Zhang[1]

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

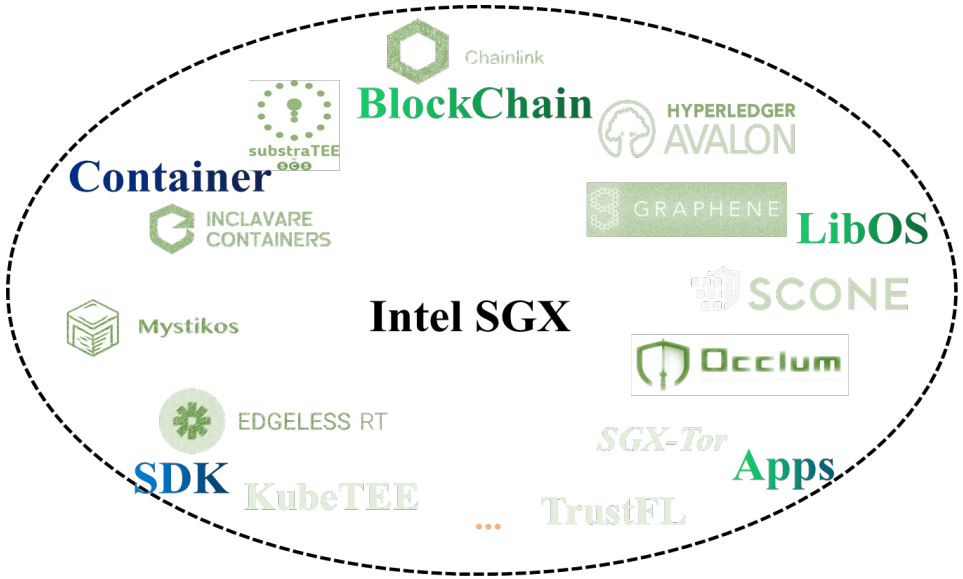
Outline

- **Motivation of SMILE**
- Overview of SMILE
- Design and Implementation
- Evaluation: Effectiveness and Performance
- Conclusion

Why SGX Enclave Need Introspection?

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.



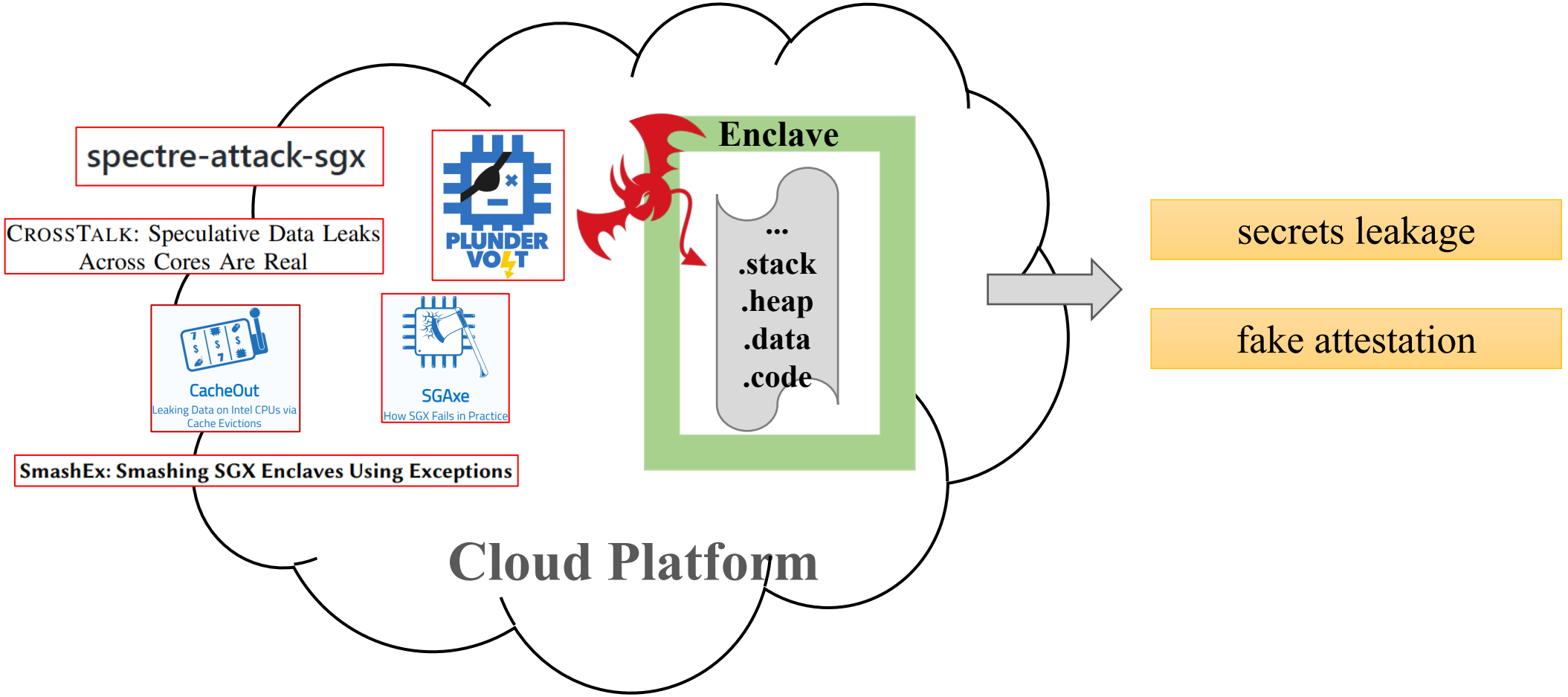
The question: does the SGX fully relieve the security concerns of users?

Why SGX Enclave Need Introspection?

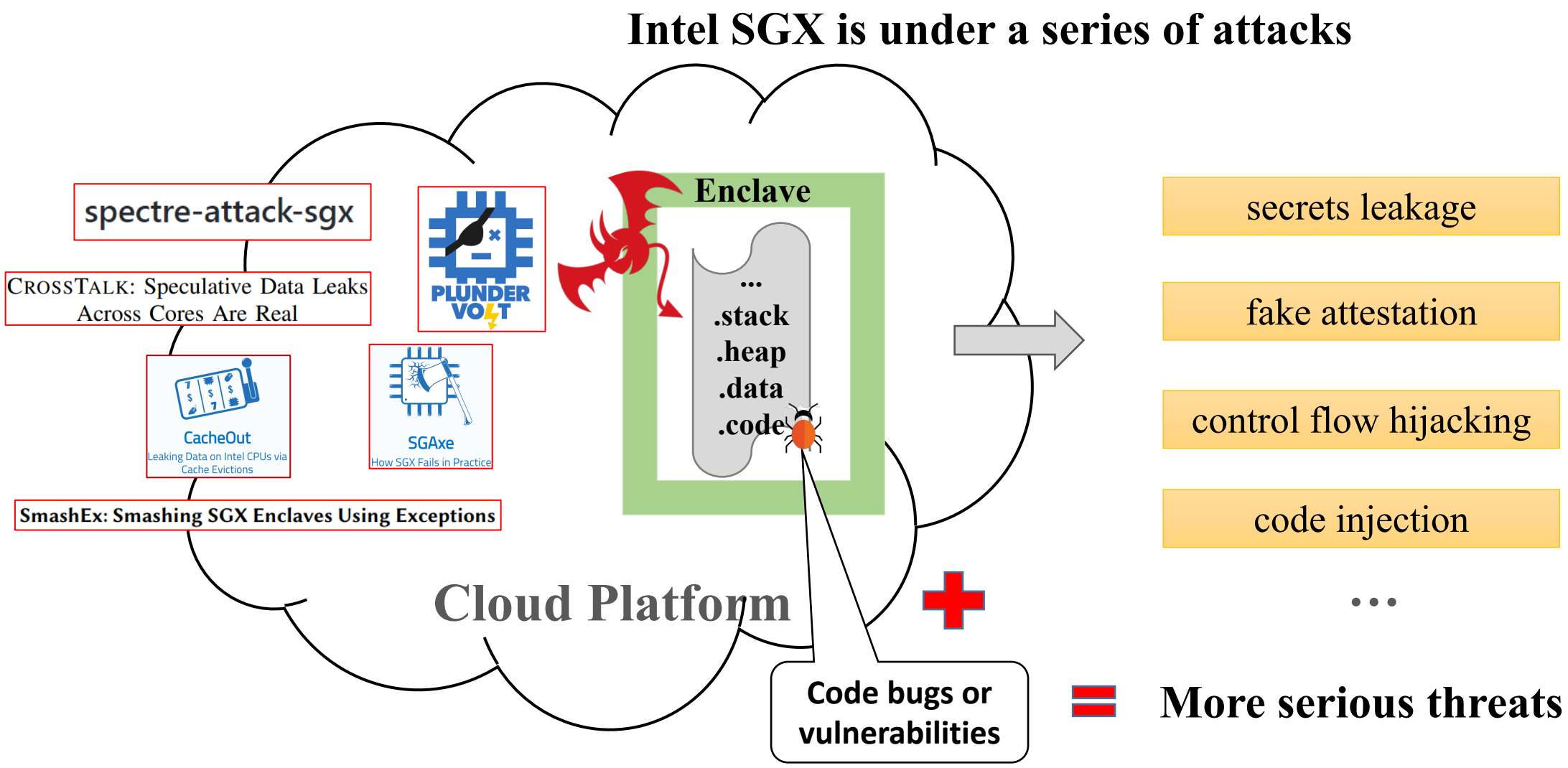
Does the SGX fully relieve security concerns?

The Answer is NO

Intel SGX is under a series of attacks

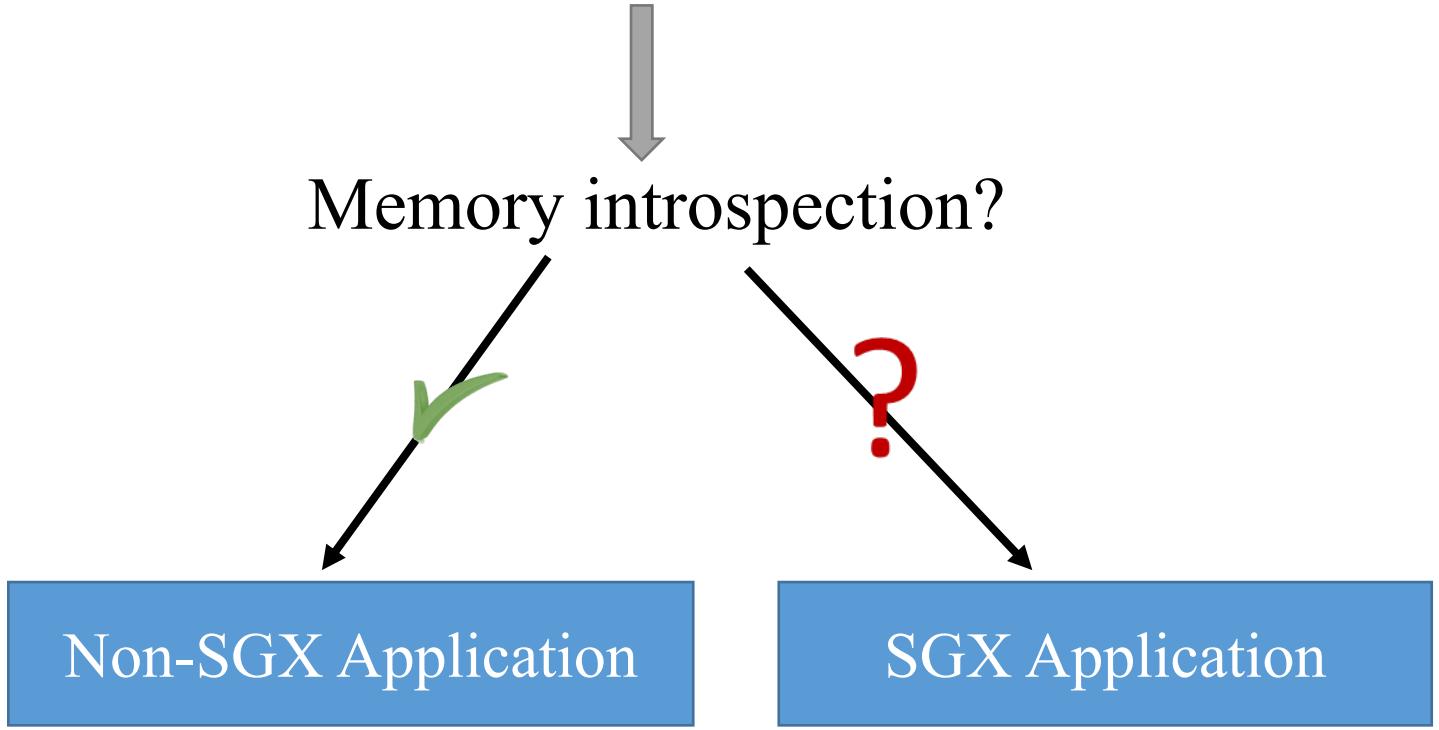


Why SGX Enclave Need Introspection?



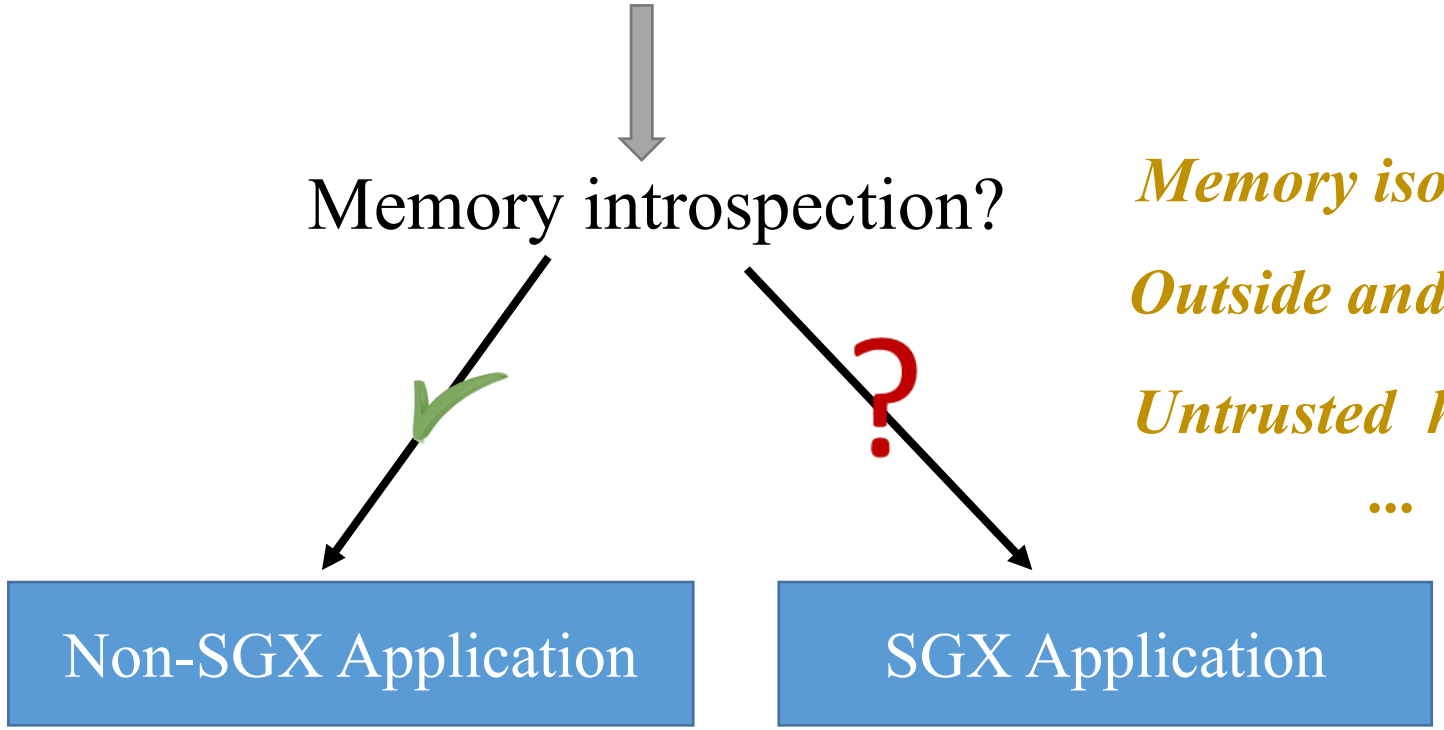
How to Securely Introspect SGX Enclave?

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



How to Securely Introspect SGX Enclave?

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



Memory isolation

Outside and inner attacks ⇨ **Error result !**

Untrusted host

...

Fake enclave !

Security issue!

False Introspection

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:

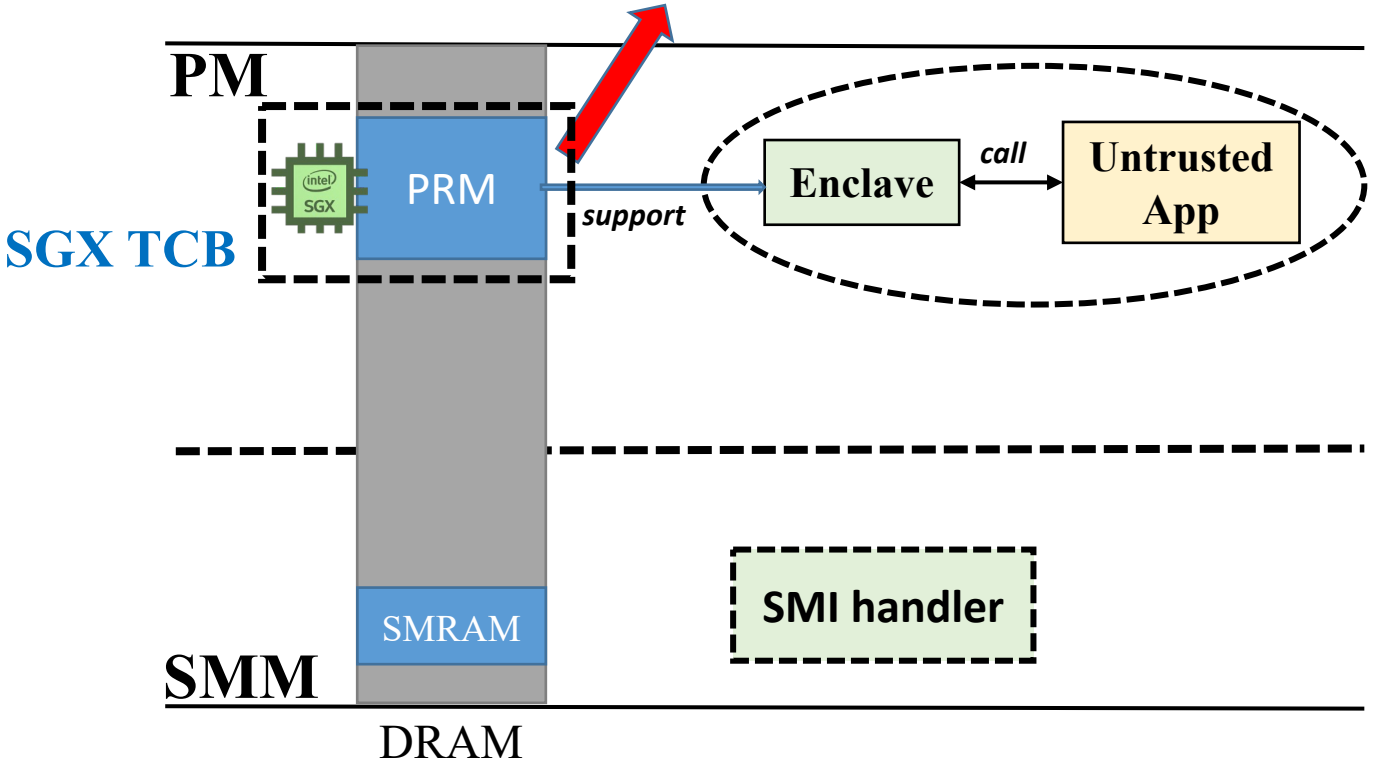
- ✓ ***Enclave authenticity***-introspection is upon the expected enclave.
- ✓ ***Introspection genuineness***-introspection results are not faked by corrupted enclave code.
- ✓ ***Security preserving***-introspection does not undermine the default enclave security.

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.

SMM for Introspection Assistance

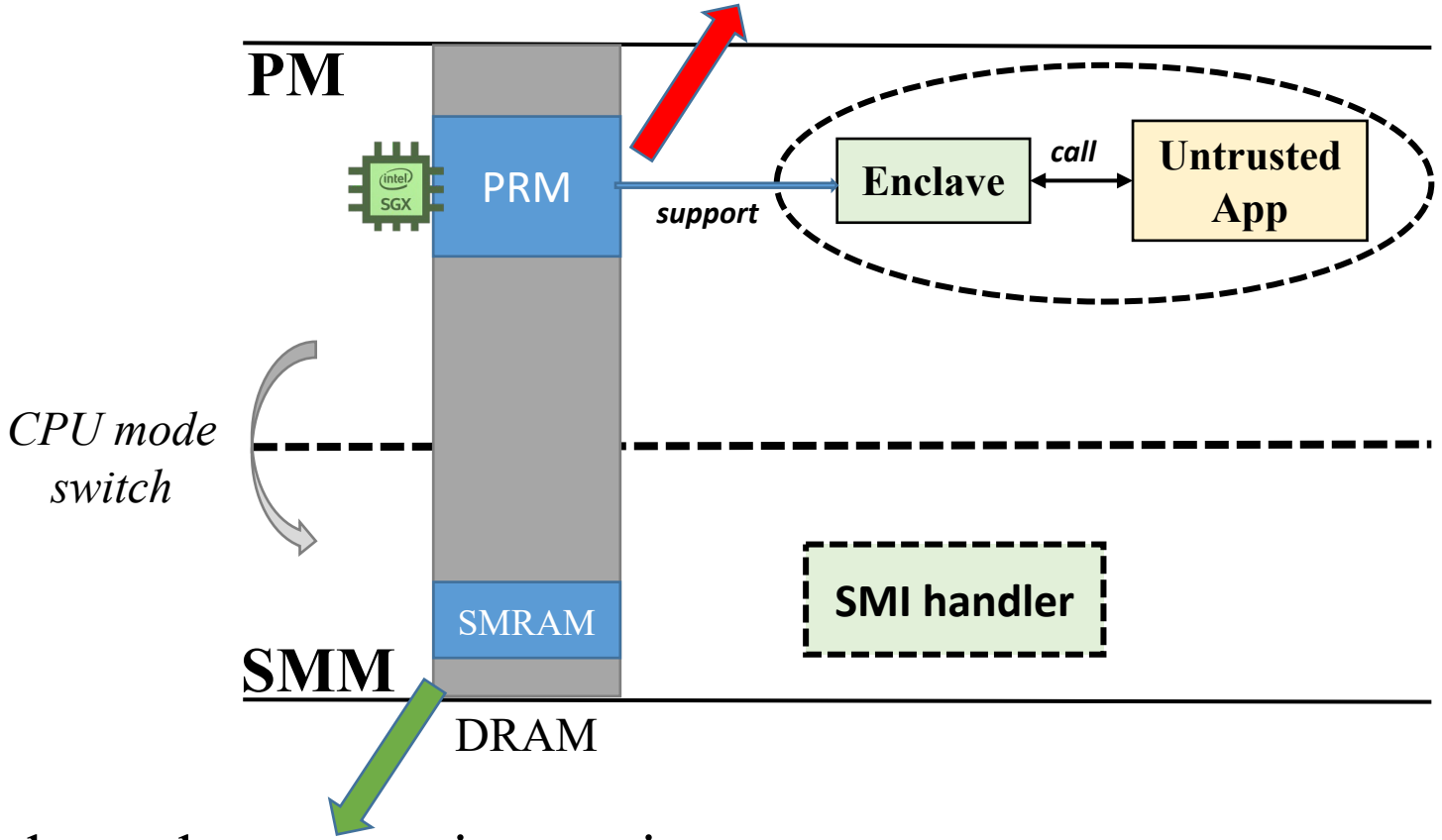
SGX Neither deals with enclave software compromise
nor supports runtime measurement



Full overview of the SGX enclave execution and SMM location in the system

SMM for Introspection Assistance

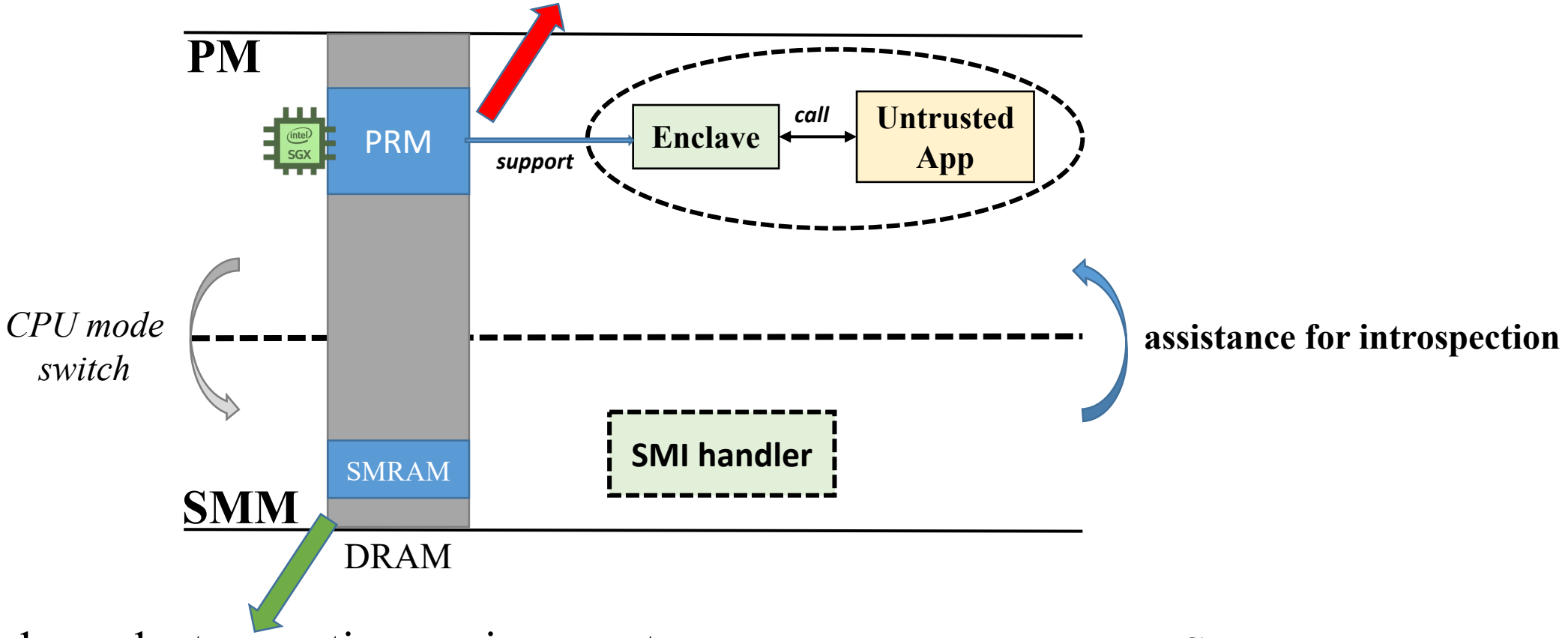
SGX Neither deals with enclave software compromise
nor supports runtime measurement



- ✓ 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



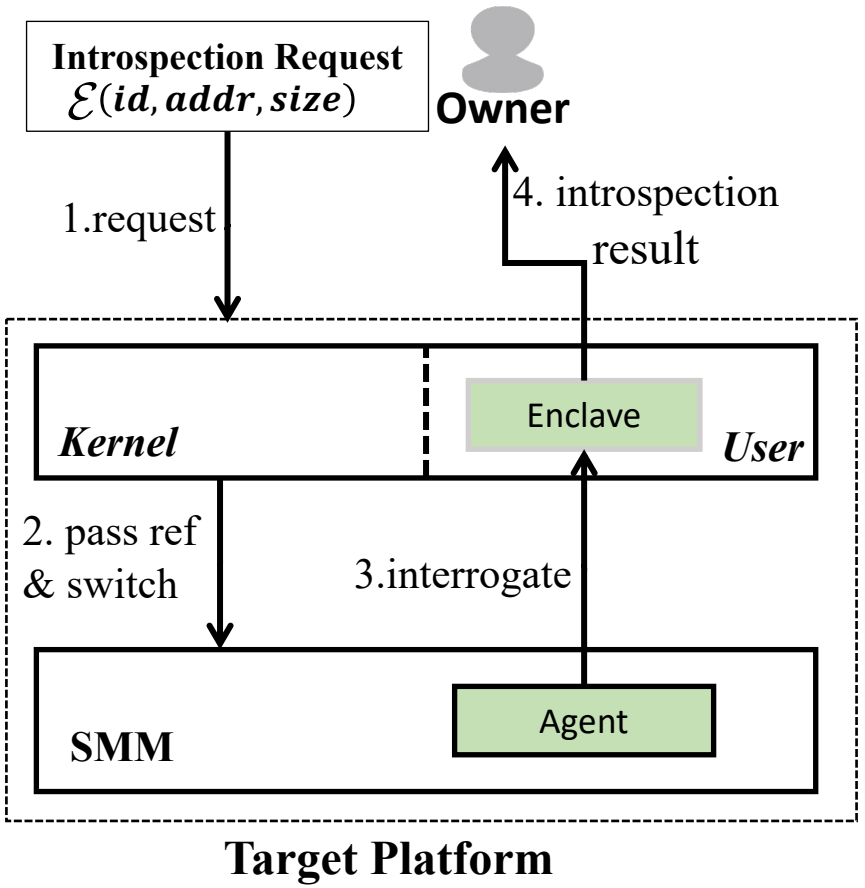
- ✓ Independent execution environment
- ✓ Halting and restoring host application
- ✓ Enable accessing the host memory and register value

Securely control enclave execution envrionment

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

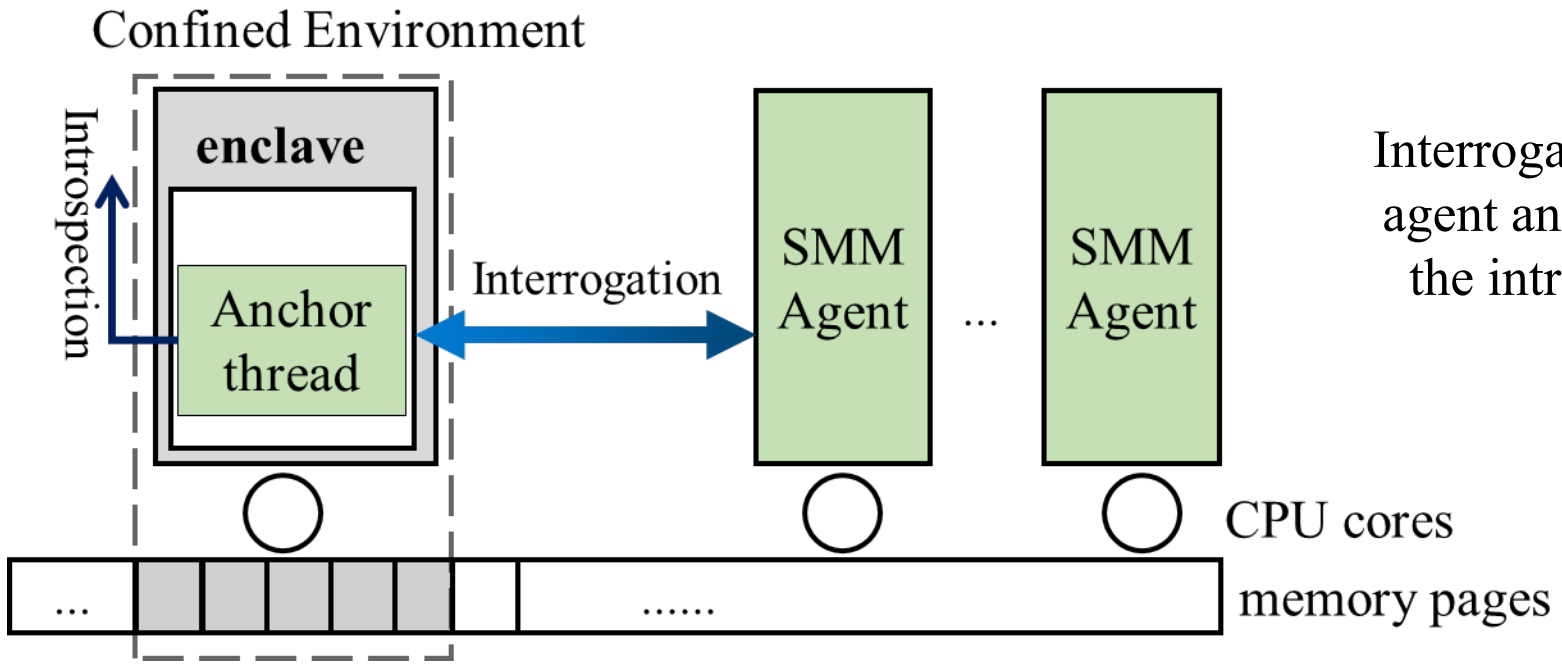
- Background of SGX
- Overview of SMILE
- **Design and Implementation**
- Evaluation: Performance and Effectiveness
- Conclusion

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.

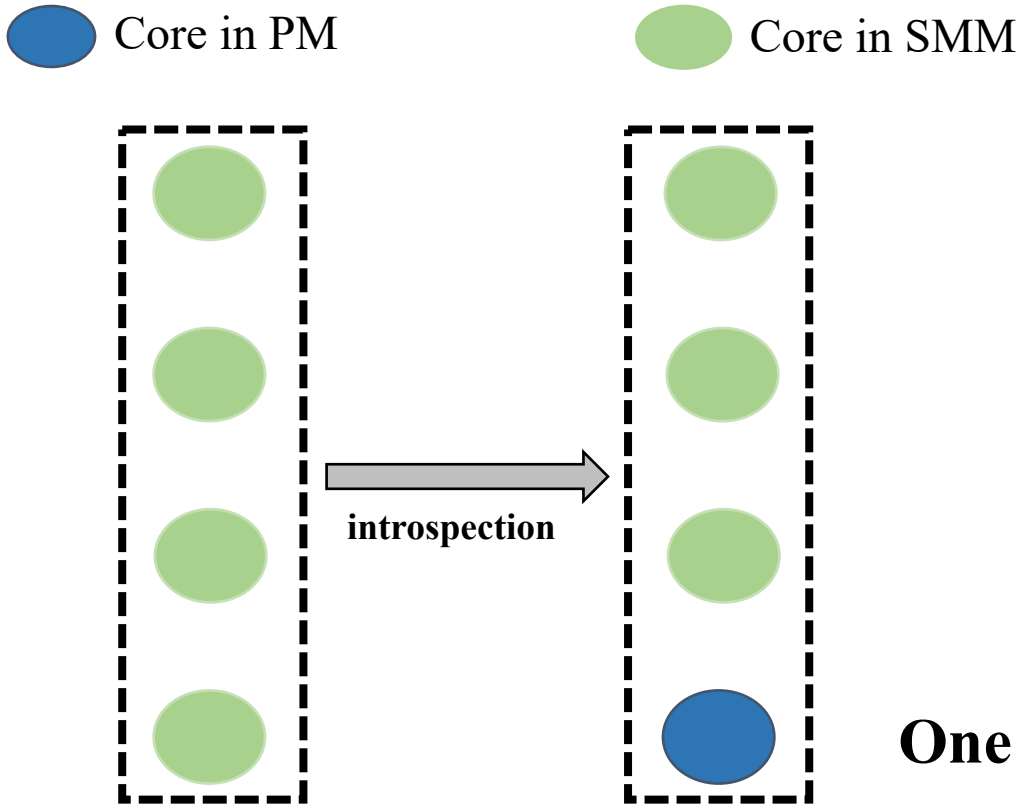
Design of SMILE



Interrogation session between SMM agent and anchor thread is to verify the introspection code in enclave

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

Confined Environment for Introspection



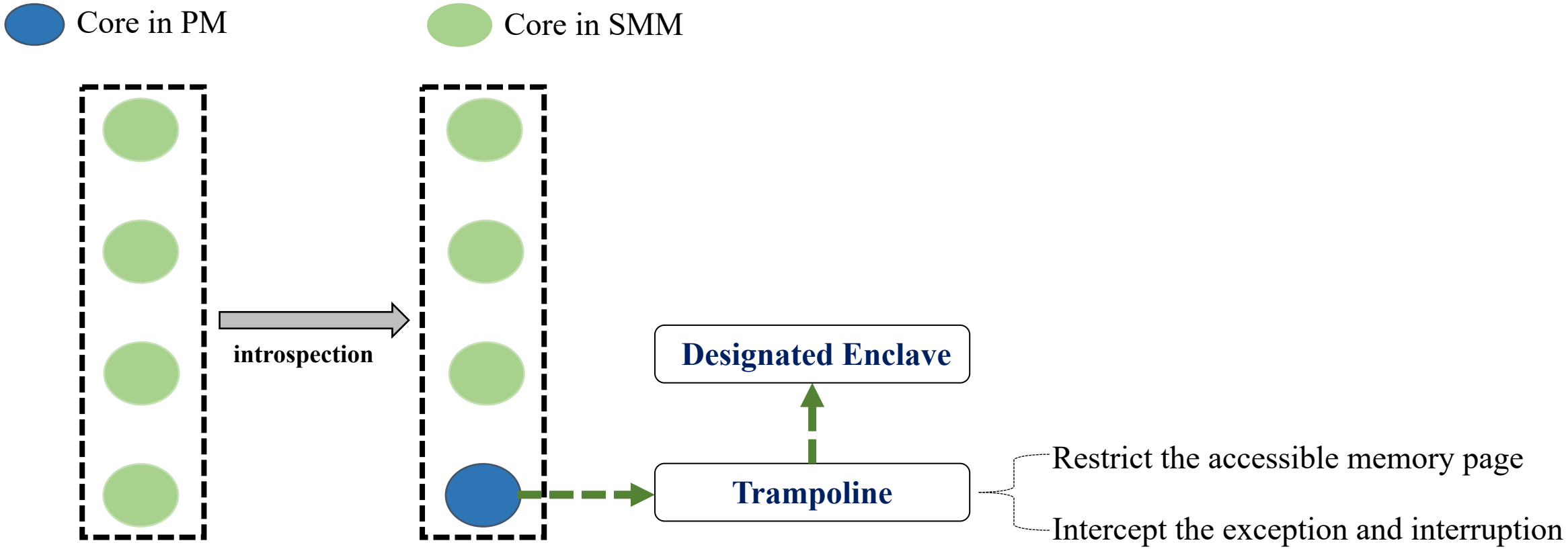
Illustrates:

- 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

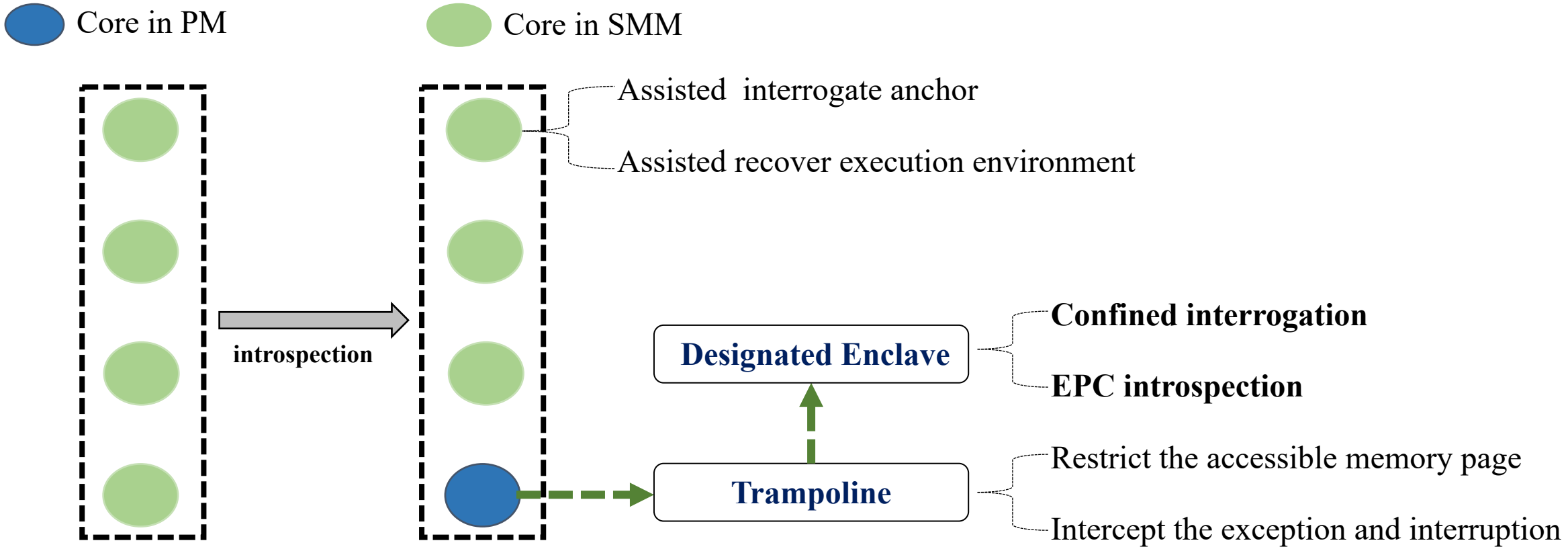
The state of CPU cores is controlled by SMM agent

Confined Environment for Introspection



The state of CPU cores is controlled by SMM agent

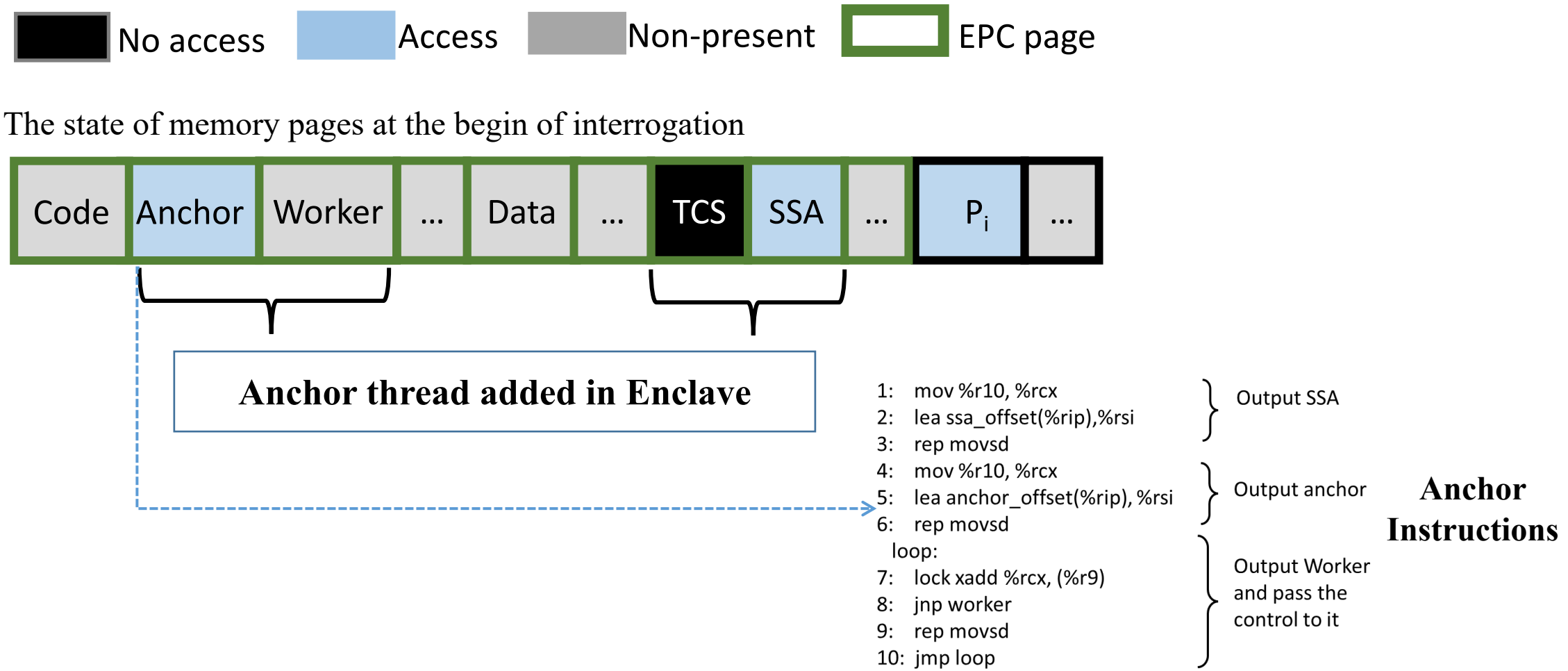
Confined Environment for Introspection



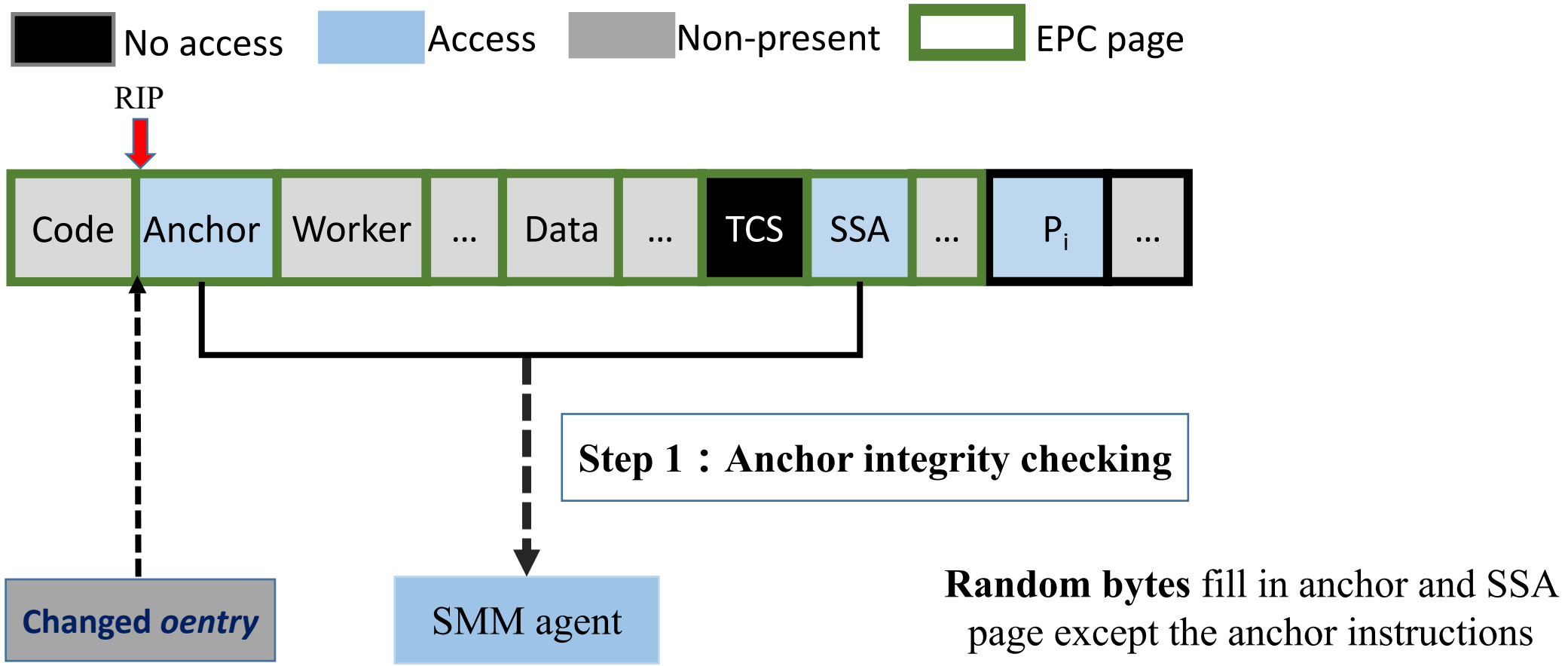
The state of CPU cores is controlled by SMM agent

Confined Interrogation

- Authenticity of Enclave is the prerequisite of a secure introspection



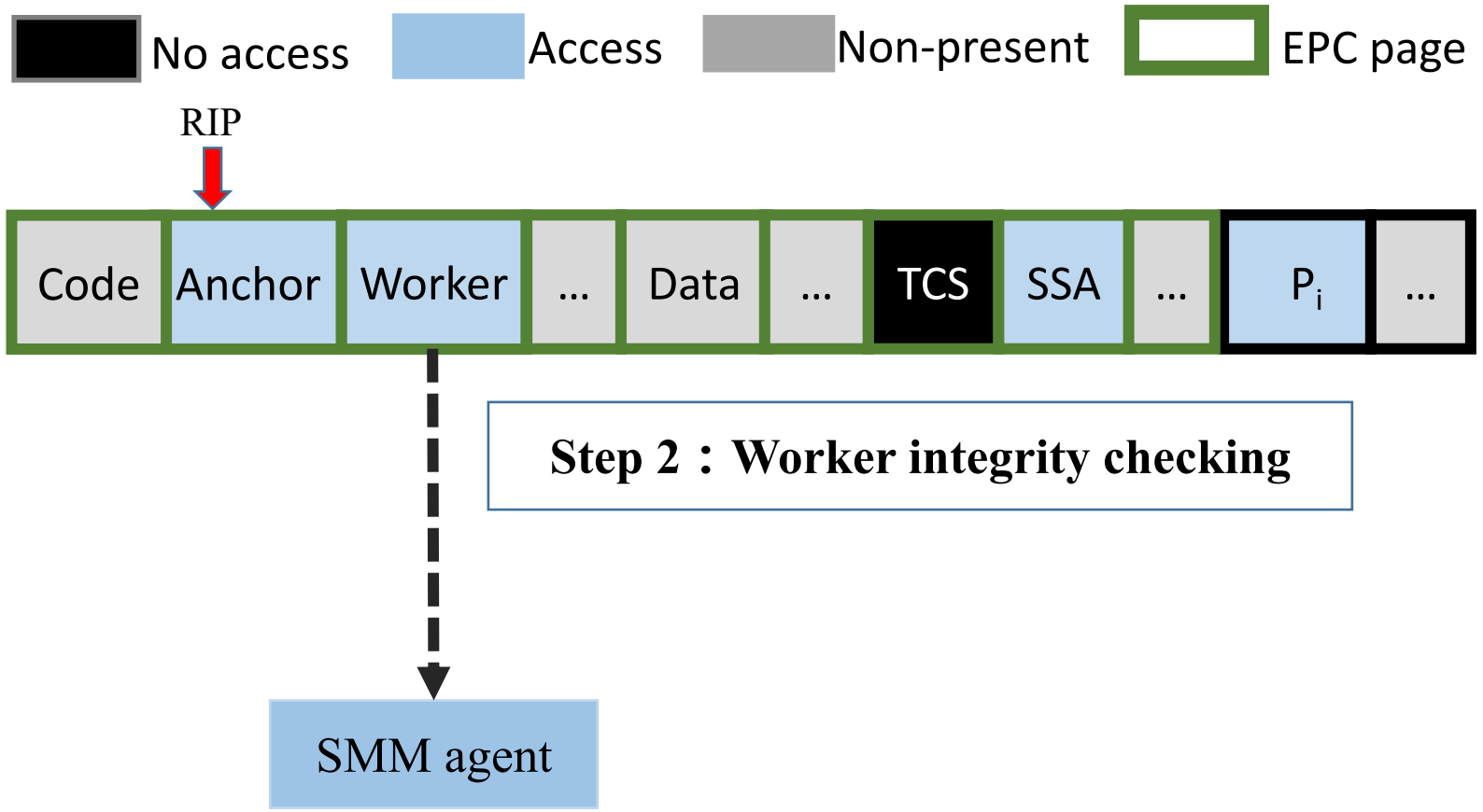
Confined Interrogation Protocol



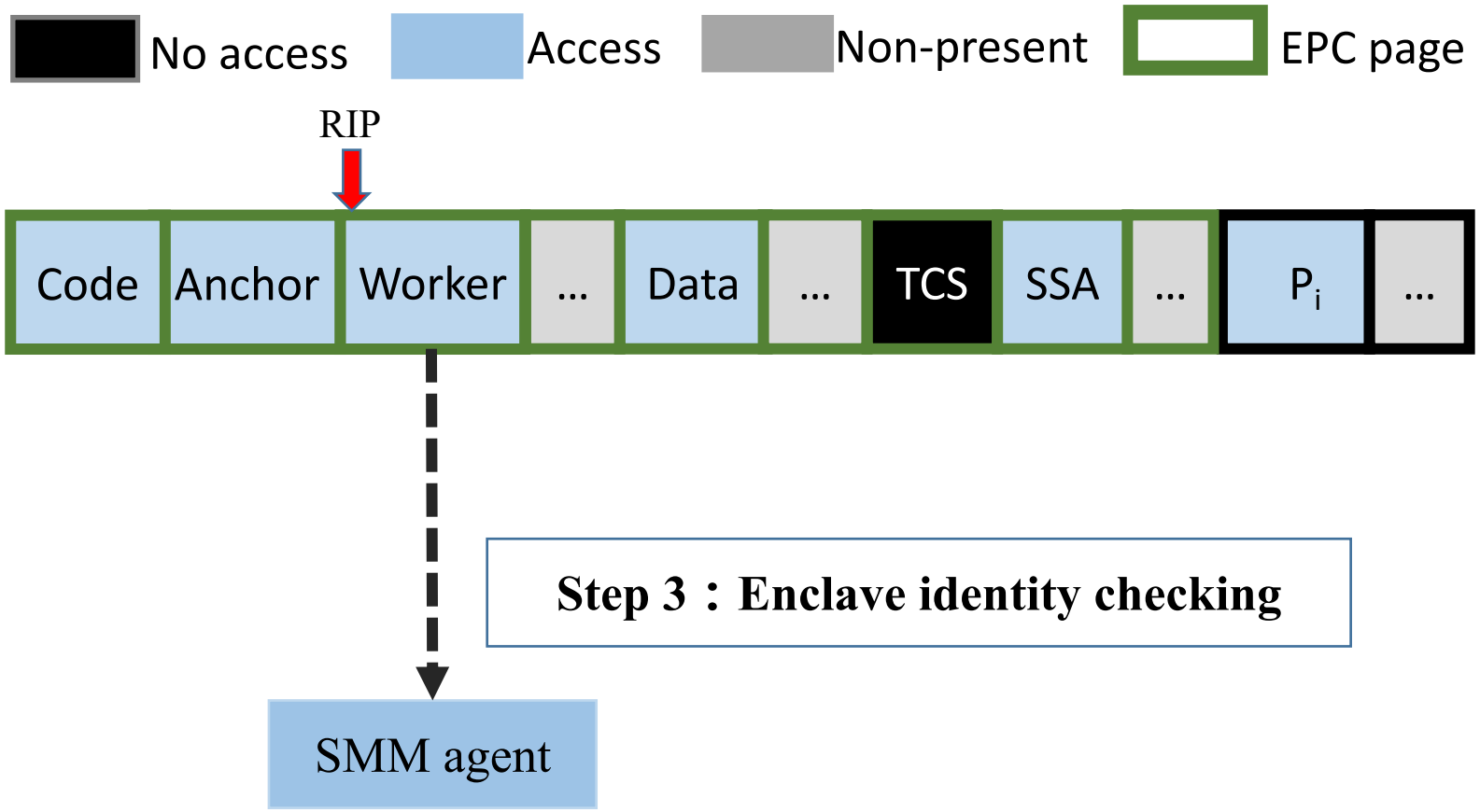
Random bytes fill in anchor and SSA page except the anchor instructions

Anchor is the first piece of code to run

Confined Interrogation Protocol

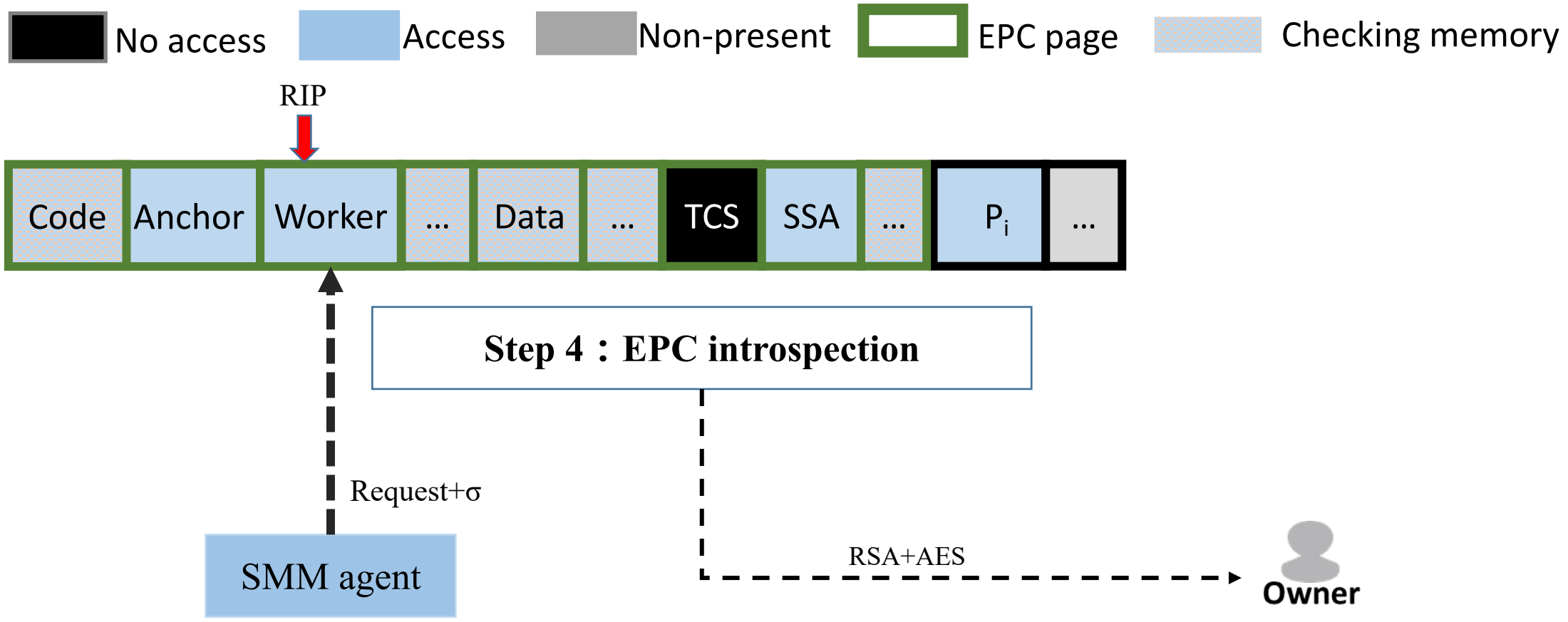


Confined Interrogation Protocol



The worker achieves the enclave id **under the confined environment**

EPC Introspection

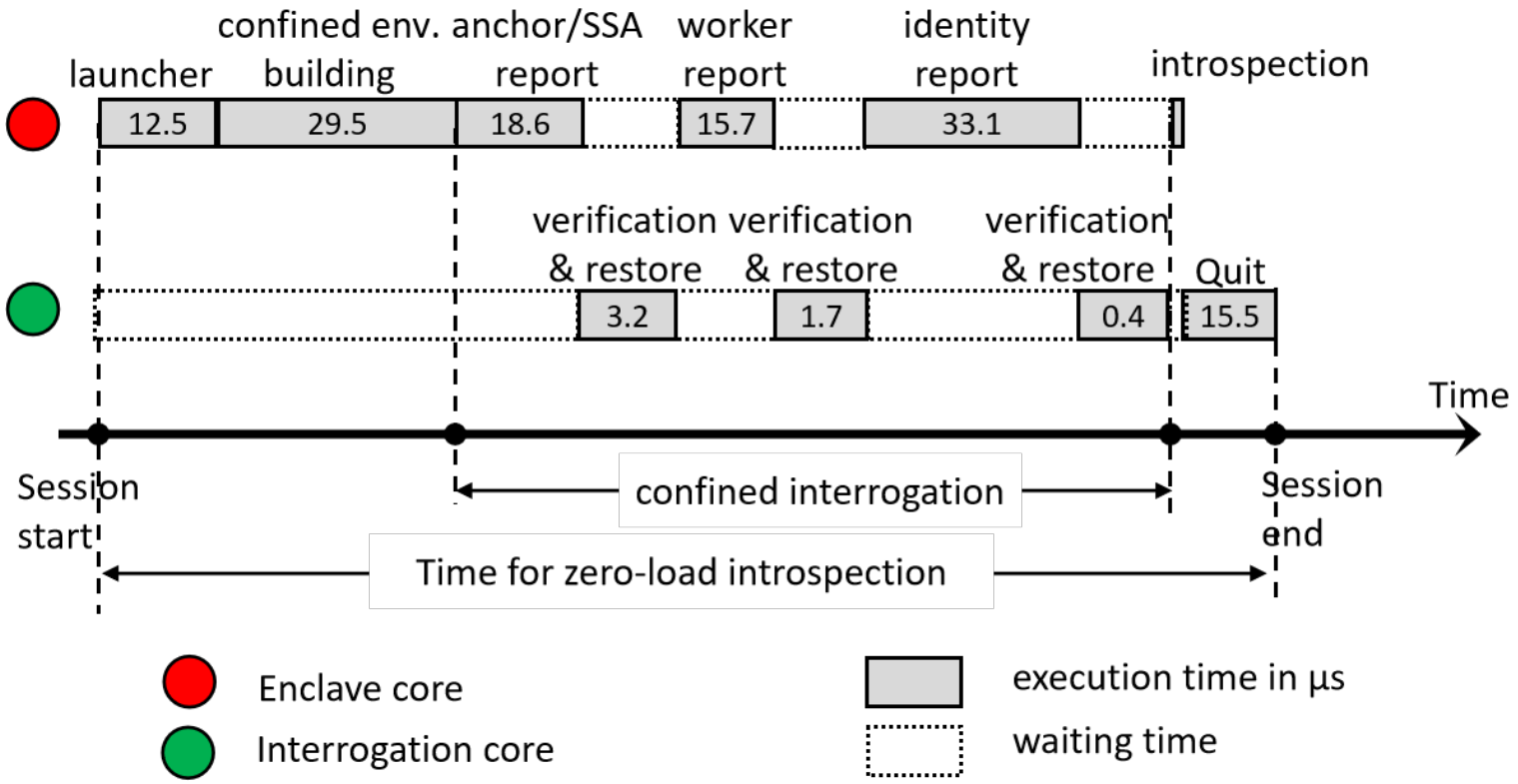


Neither a corrupted SMM agent nor an imposter can exploit SMILE

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.

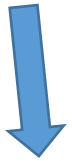
The Time Overhead for SMILE Introspection

One session of SMILE introspection:

- Interrogation 159.3 μs
+ RSA encryption 121.7 μs
+ 1-page AES encryption 2.1 μs.

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

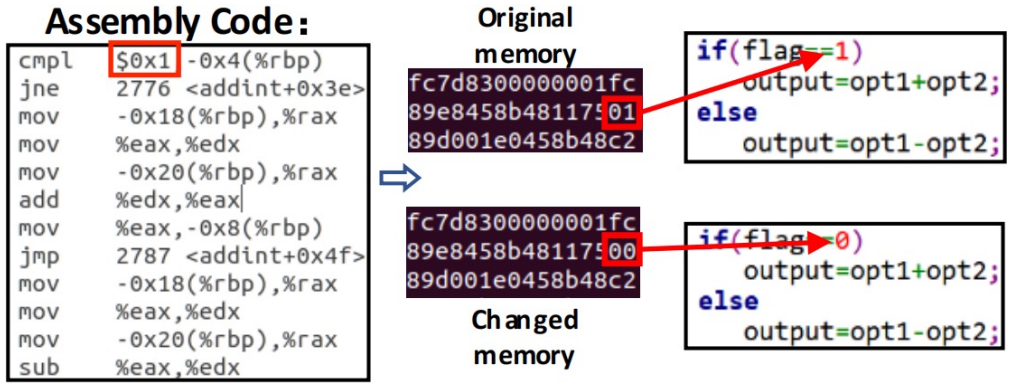
Improve



Sharing the AES key in multi-sessions with one-time RSA, costs $(121.7+159.3n/r+2.1n)$ μs

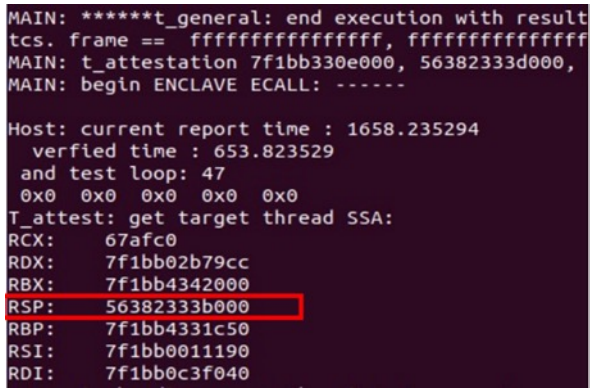
Applications of SMILE

Code Integrity Checking

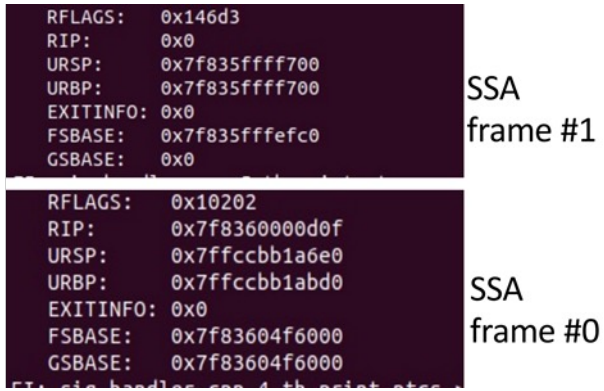


The attack swaps the conditions for two branches in the enclave, and SMILE finds the modifications at runtime.

SSA State Checking



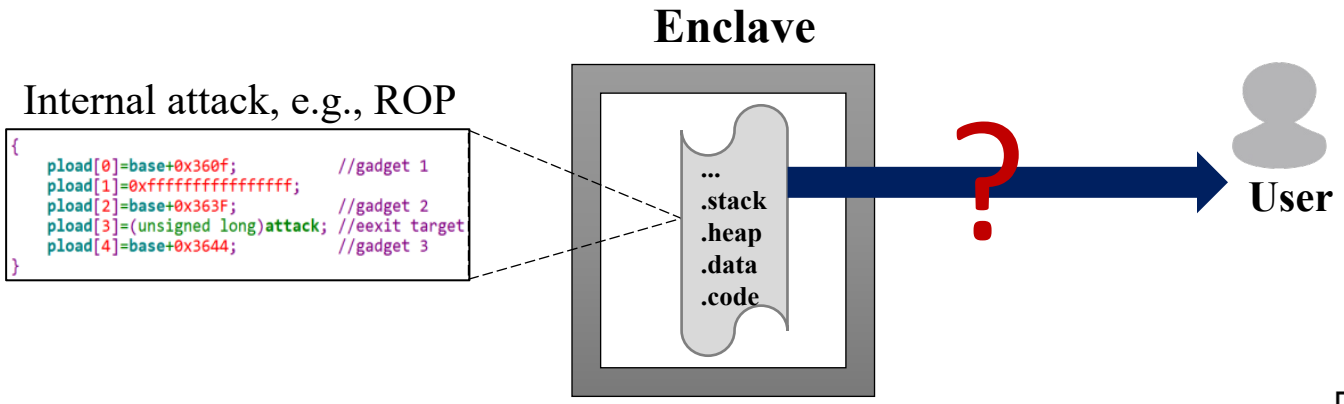
(a) Enclave using external stack.



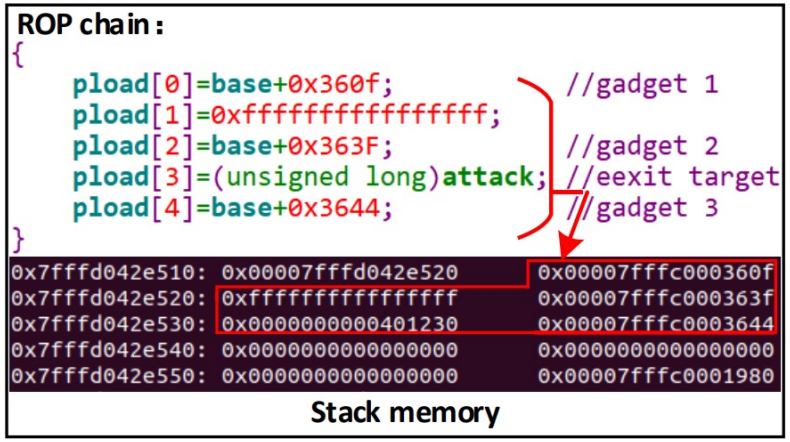
(b) Embedded AEX.

SMILE dumps the SSA frame data to verify if the saved register value is right.

Applications of SMILE – Stack Checking

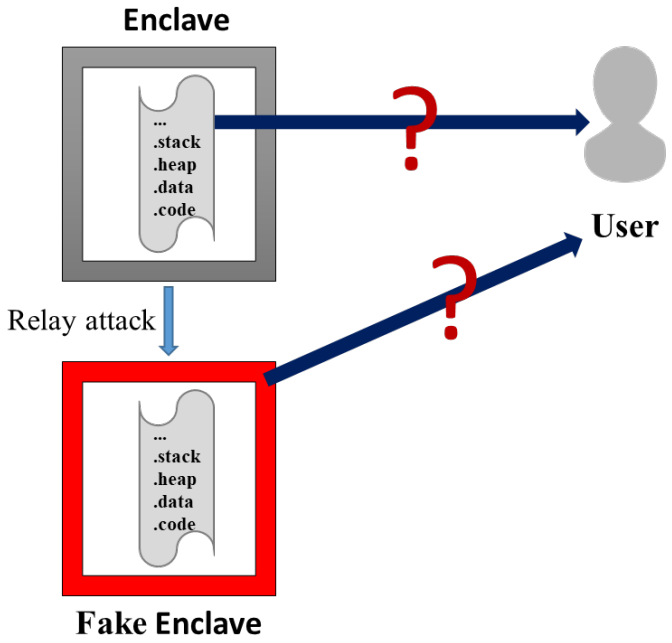


Similarly, the stack might be attacked by malware for code injection, e.g., ROP.

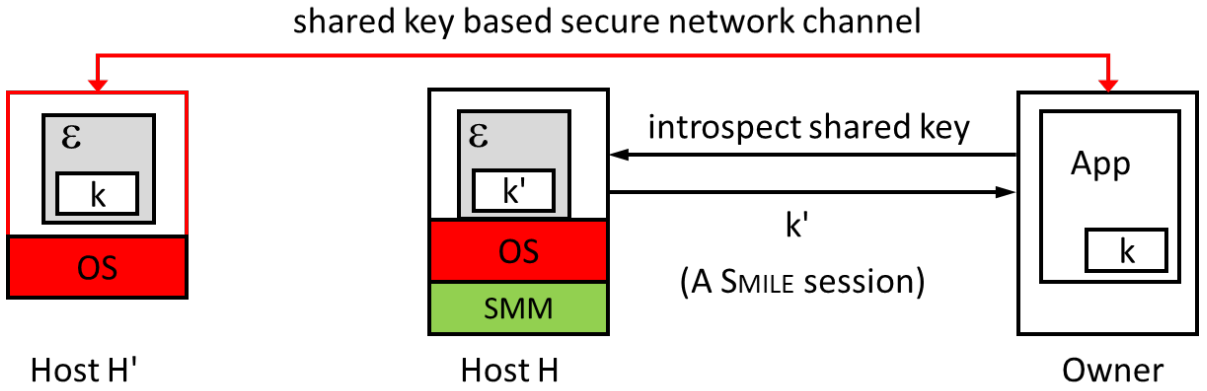


SMILE dumps the stack frame data to verify if any trace of attack.

Applications of SMILE – Enclave Location Verification

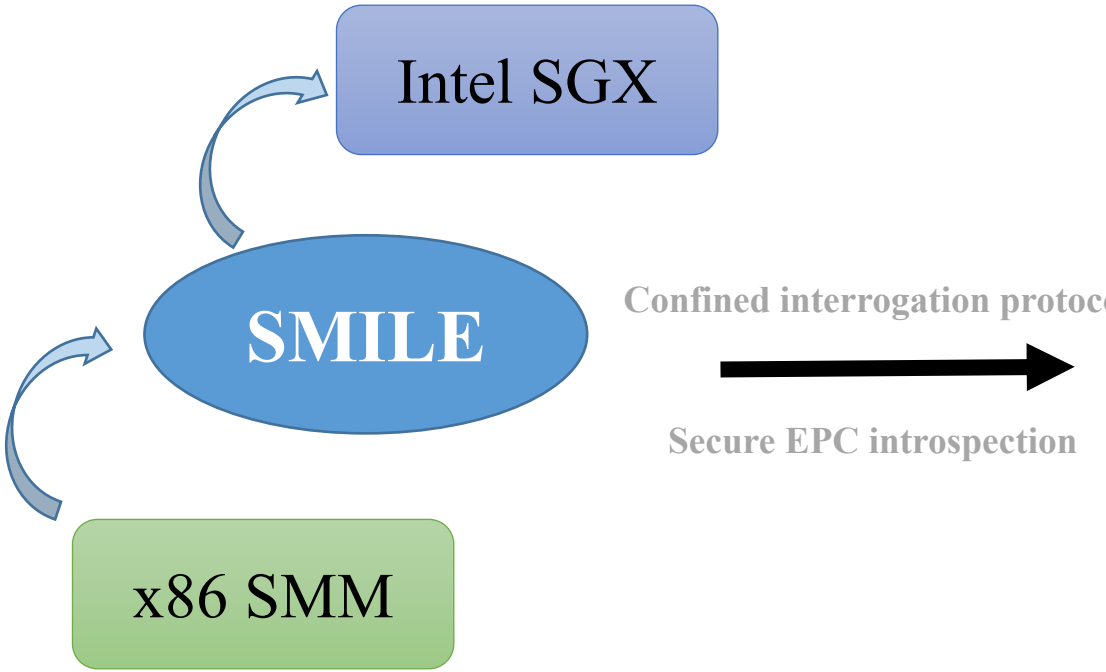


SGX attestation mechanism cannot perfectly verify the enclave location



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*

Conclusion



Confined interrogation protocol

Secure EPC introspection

Secure live introspection

enclave authenticity

genuineness

security-preserving

Acceptable overhead

281μs +

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