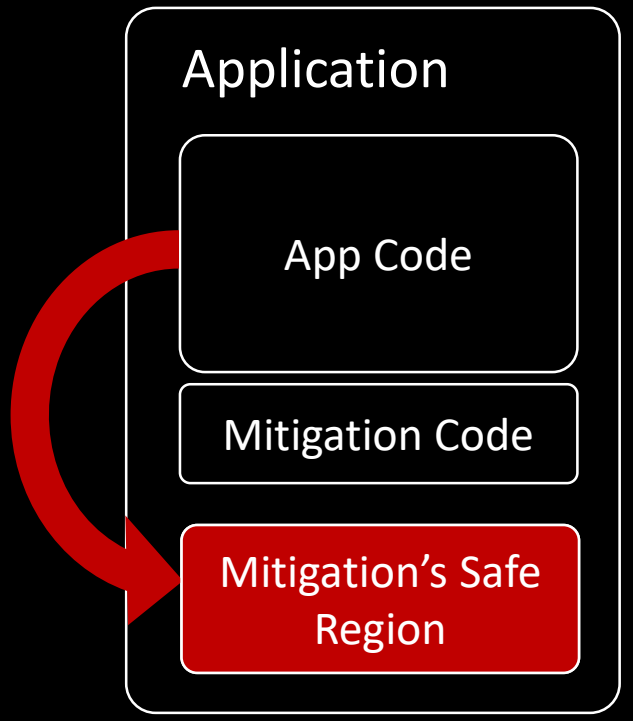
IMIX: Hardware-Enforced In-Process Memory Isolation

Tommaso Frassetto, Patrick Jauernig, Christopher Liebchen, Ahmad-Reza Sadeghi
Technische Universität Darmstadt

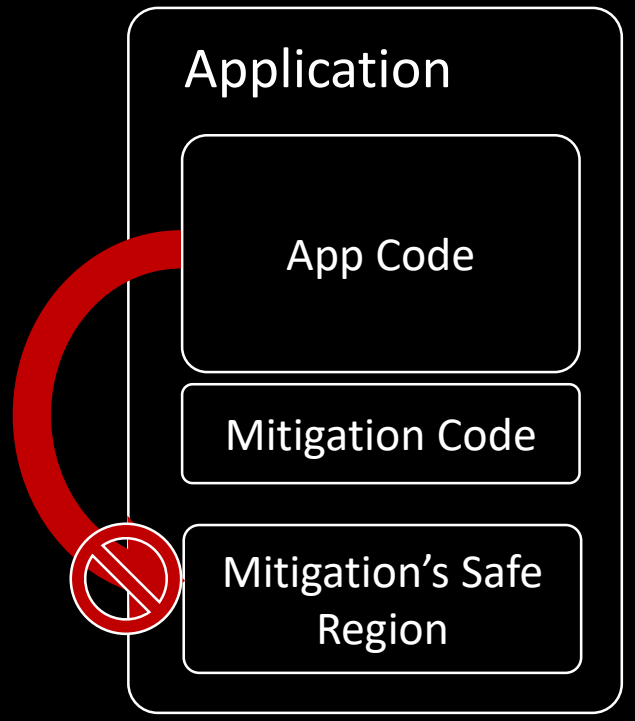
Research Institute for Collaborative Autonomous and Resilient Systems

TECHNISCHE UNIVERSITÄT DARMSTADT

IMIX



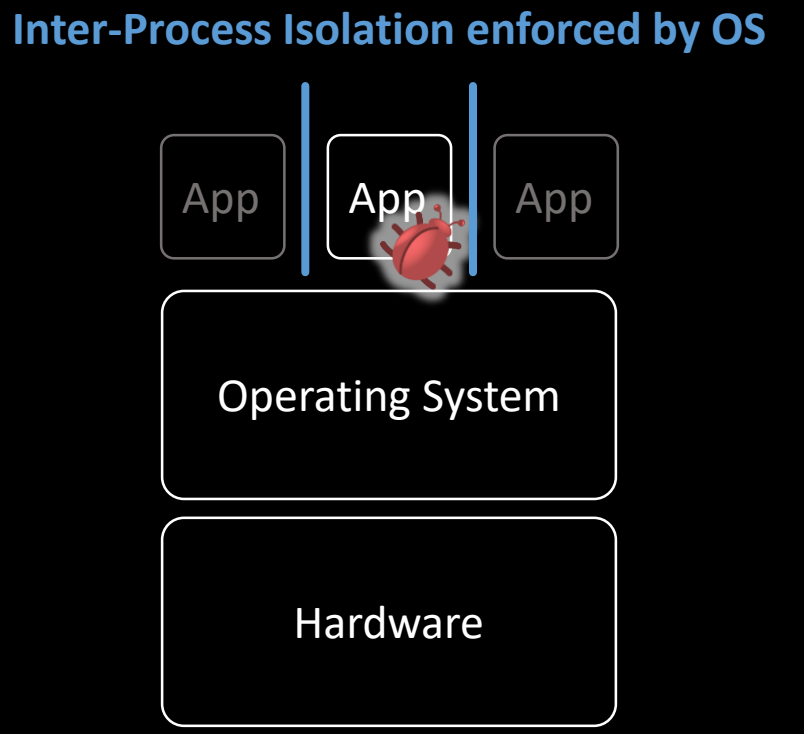
State of the Art



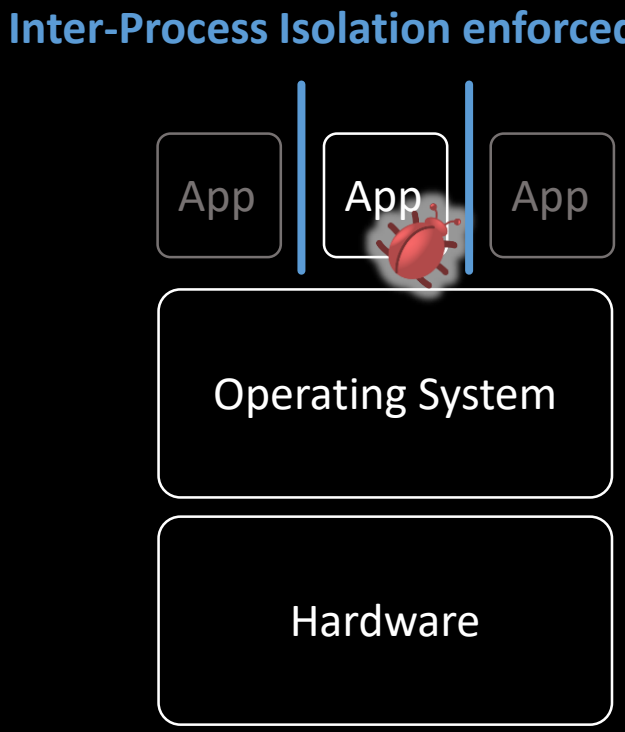
IMIX
Memory Isolation

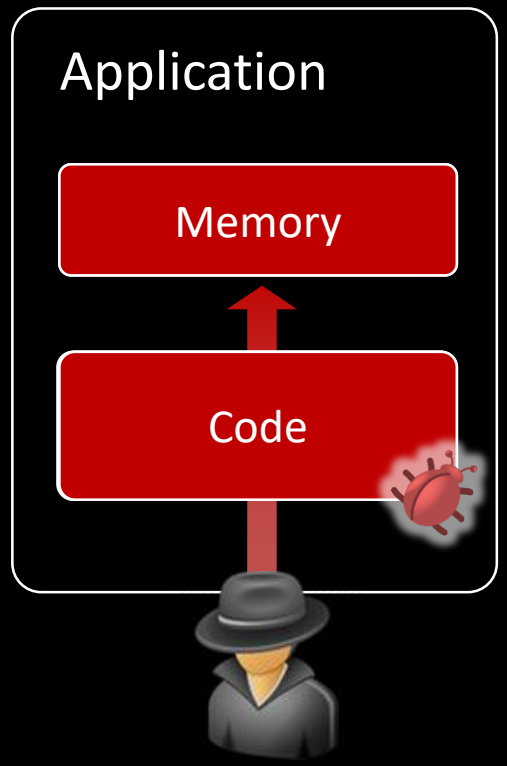
Why is in-process memory isolation a good idea?

Inter- & In-Process Isolation

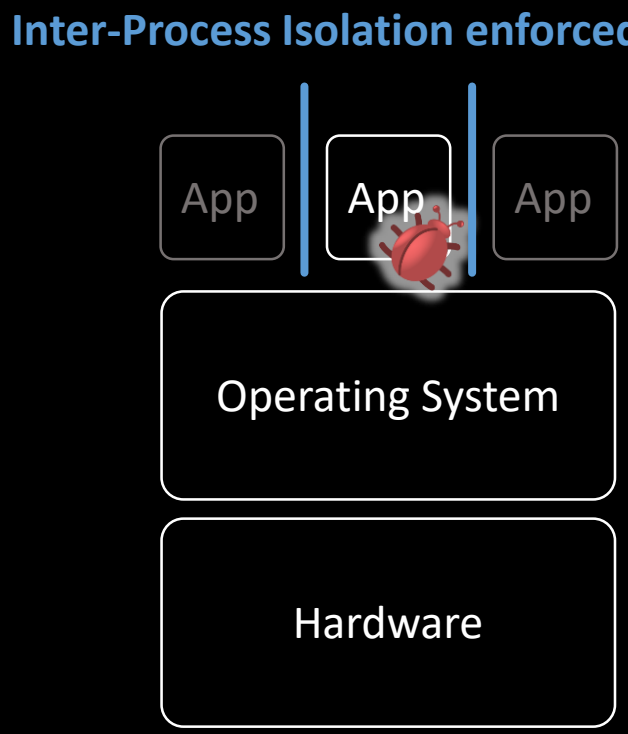


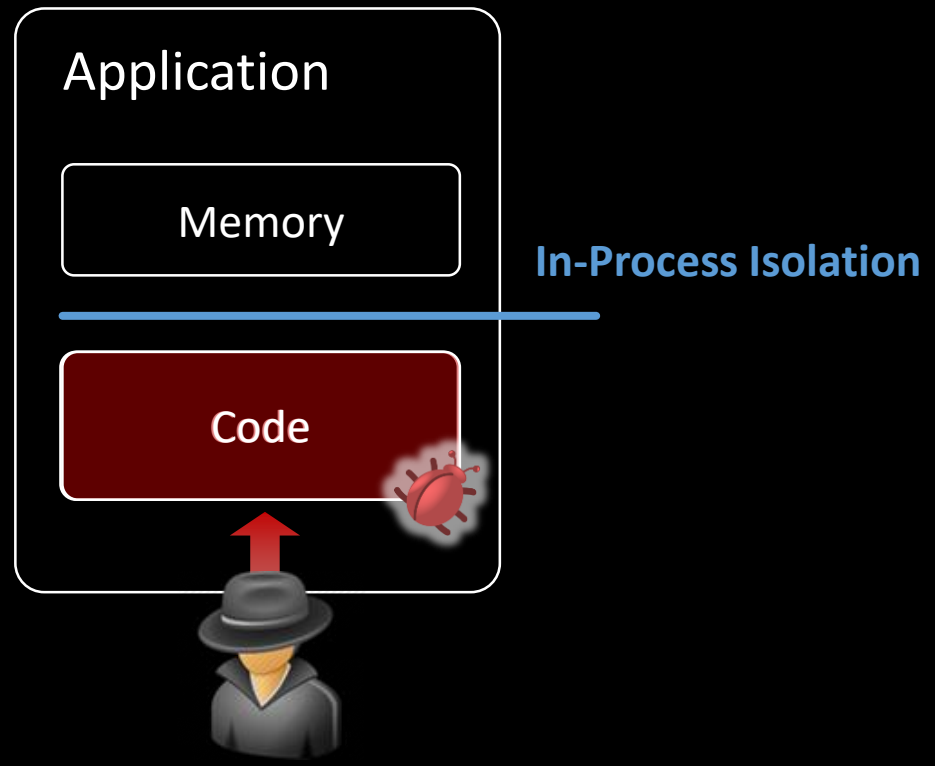
Inter- & In-Process Isolation





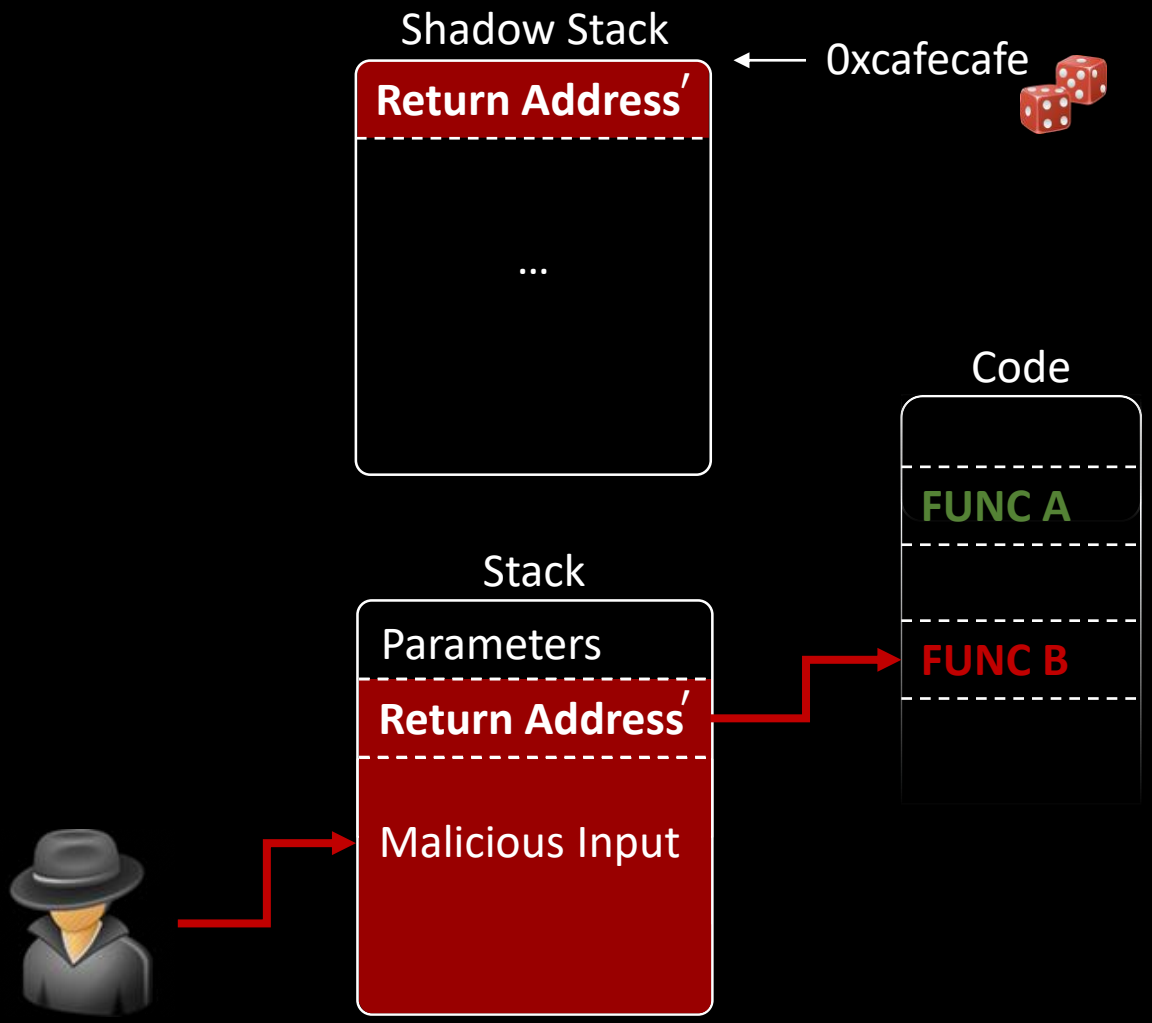
Inter- & In-Process Isolation





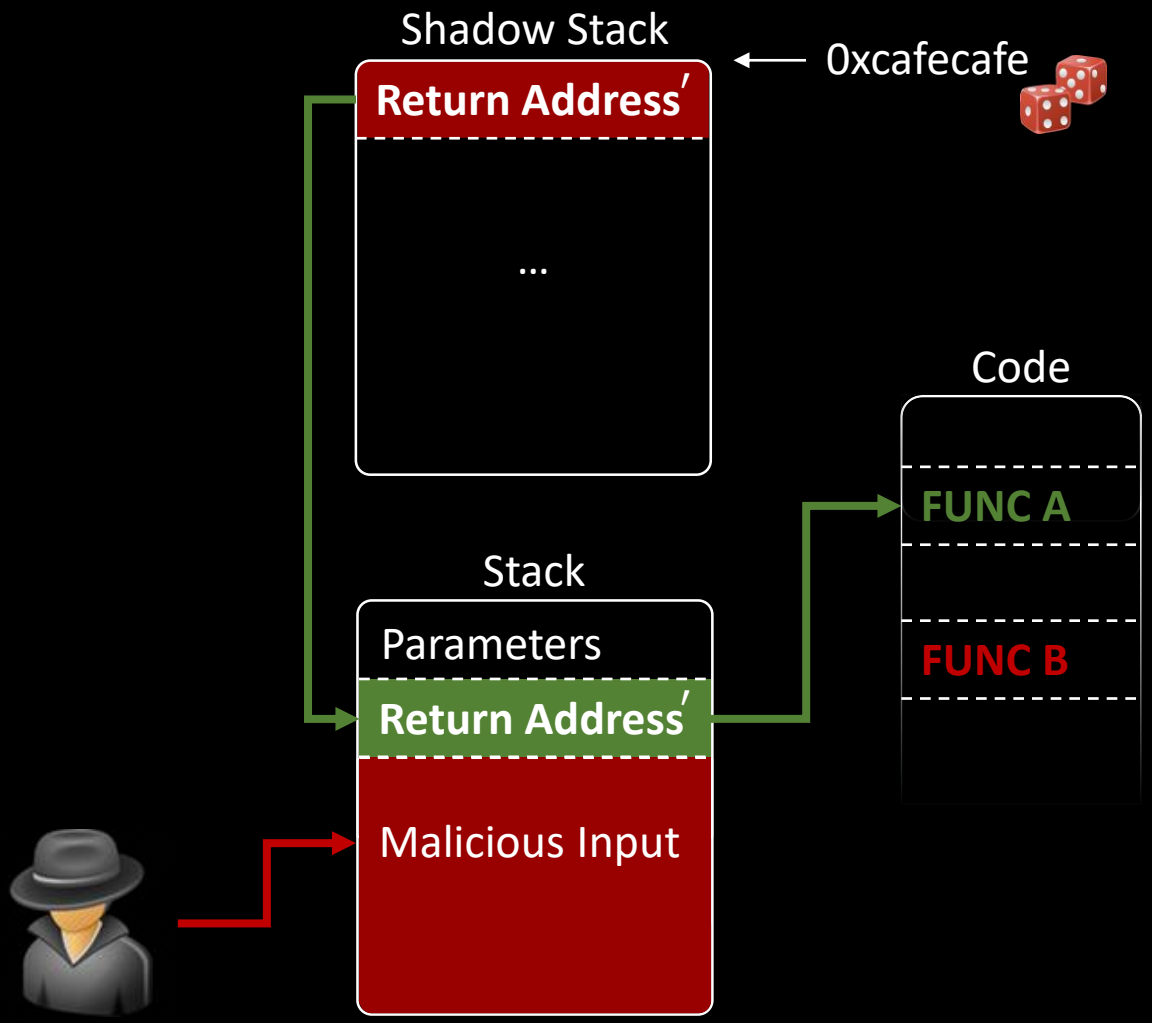
Shadow Stack

- Backup return addresses
- Address gets restored before *ret* is called



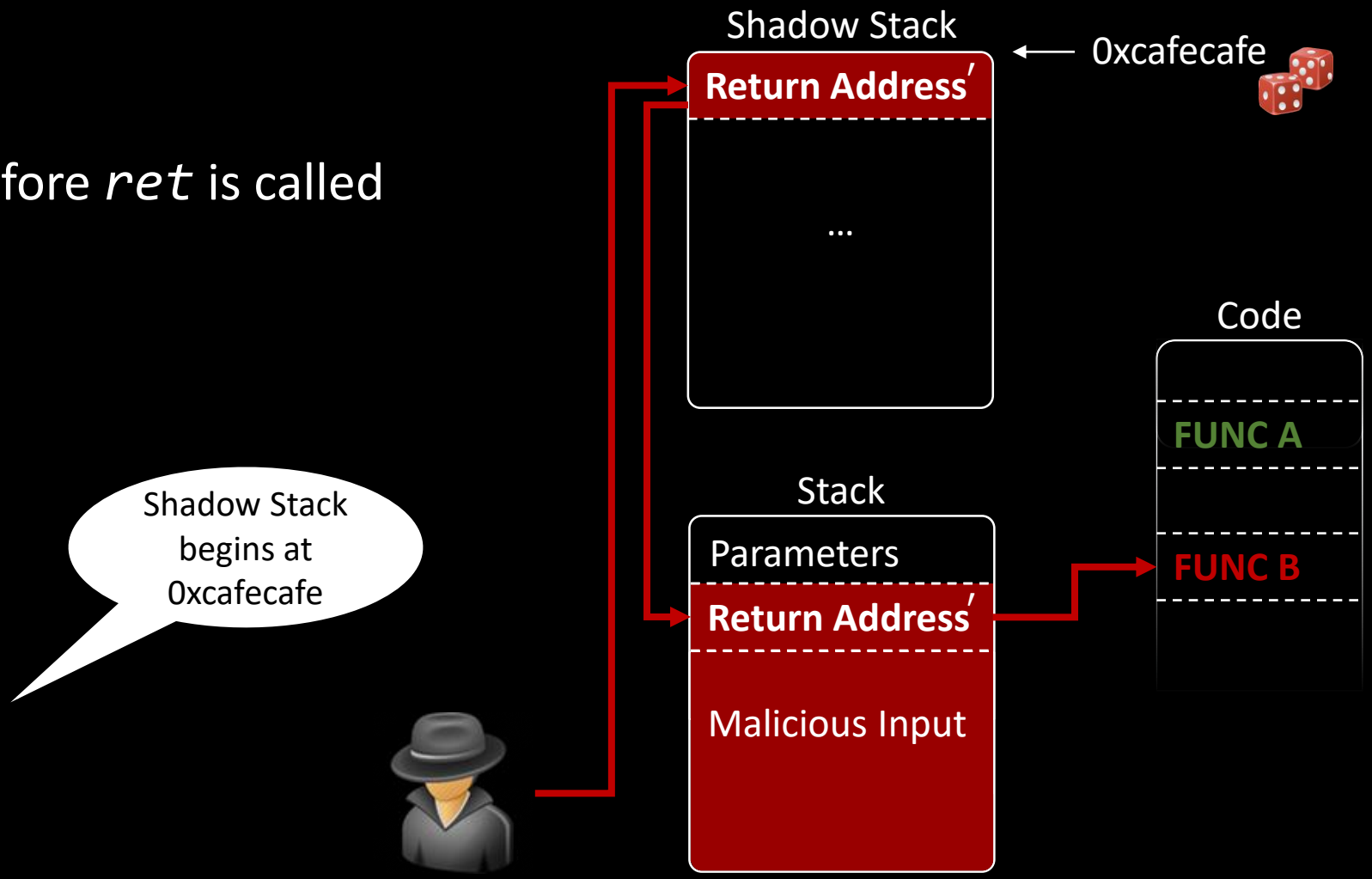
Shadow Stack

- Backup return addresses
- Address gets restored before *ret* is called



Shadow Stack

- Backup return addresses
- Address gets restored before *ret* is called



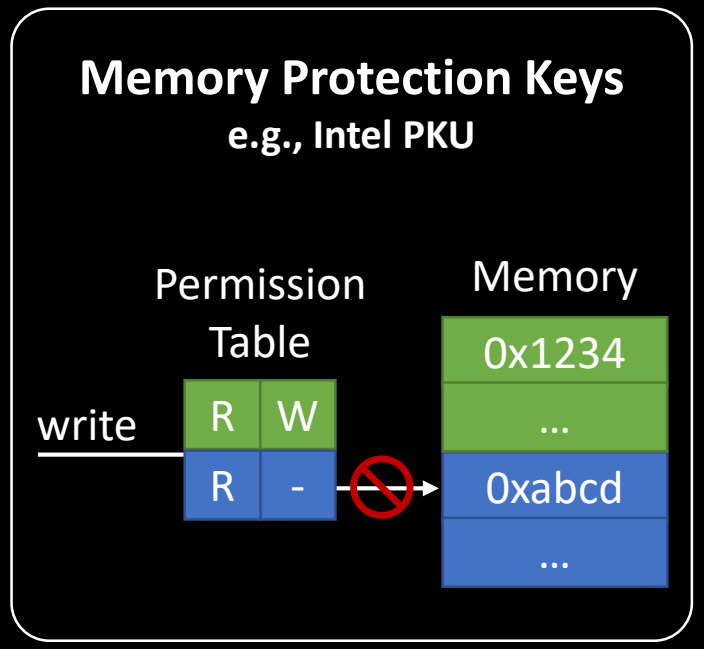
Our Contribution

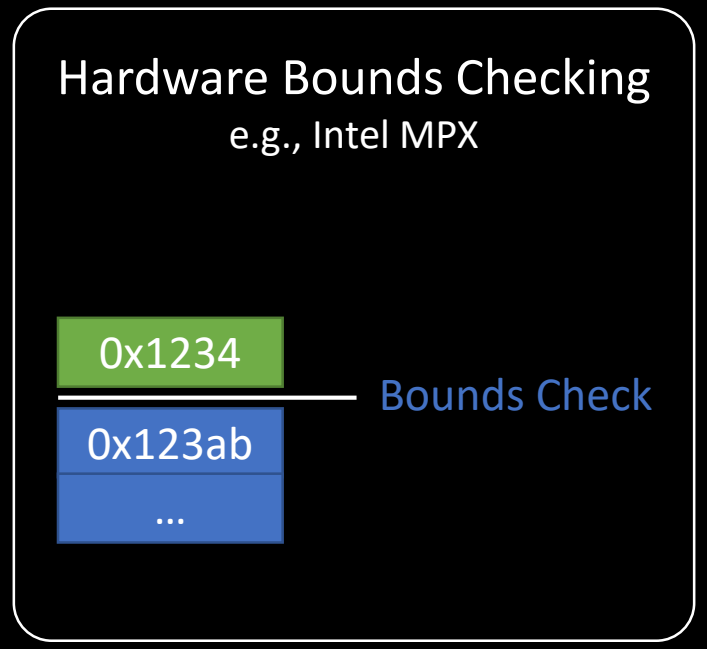
Memory isolation primitive — *Complete pipeline from compiler down to hardware*

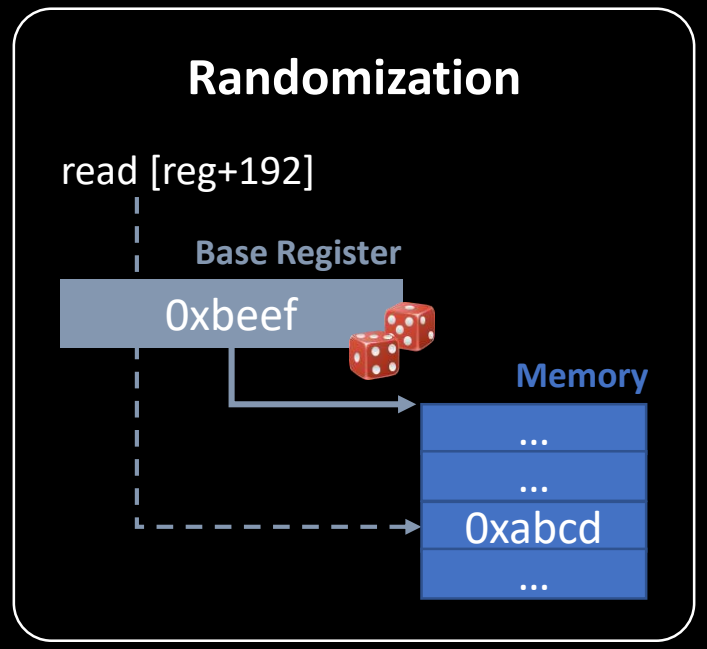
PoC implementation — *Implemented compiler support & use case*

Thorough evaluation — *Compared high-frequency memory domain switching performance to related work*

In-Process Isolation







In-Process Isolation

Memory Protection Keys
e.g., Intel PKU

Problems

- *High performance overhead for frequent switches*

Hardware Bounds Checking
e.g., Intel MPX

Problems

- *Excessive instrumentation*
- *High performance overhead*

Randomization

Problems

- *Entropy-based: single information leak breaks isolation*

What characteristics should a memory isolation primitive have?

Related Work

Policy-based Isolation

Hardware Enforced

Fast Interleaved Access

Fails Safe

Related Work

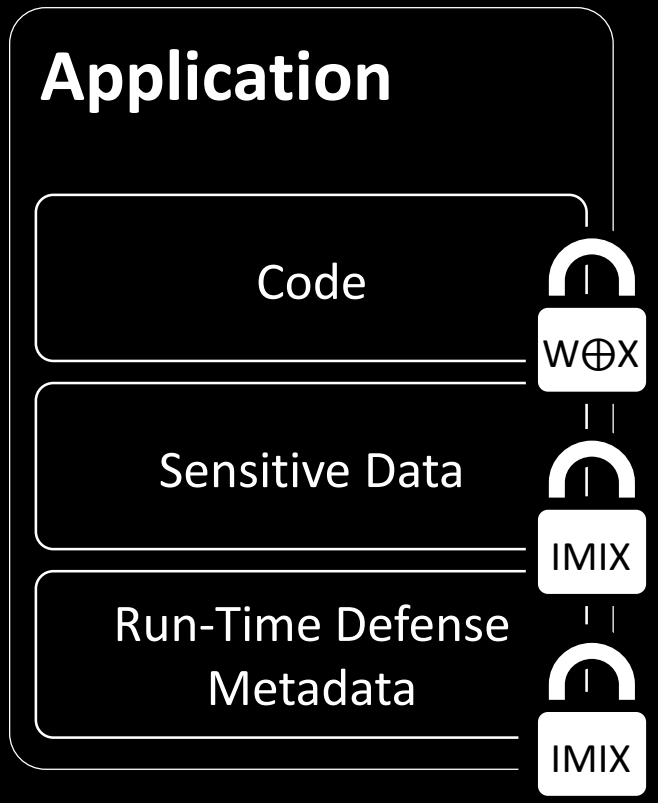
Technique	Policy-based Isolation	Hardware Enforced	Fast Interleaved Access	Fails Safe
SFI [1]	✓	✗	✓	✗
Segmentation	only for x86-32	✓	✓	✓
Memory Hiding	✗	✗	✓	✗
Paging / EPT	only single-threaded applications	✓	✗	✓
Intel MPK	✓	✓	✗	✓
Intel SGX	✓	✓	✗	✓
Intel MPX	✓	✓	(✓)	✗
Intel CET	only for Shadow Stack	✓	✓	✓

Goal: build a practical primitive that incorporates all aspects

[1] D. Sehr, R. Muth, C. Biffle, V. Khimenko, E. Pasko, K. Schimpf, B. Yee, and B. Chen.
Adapting software fault isolation to contemporary cpu architectures. In 18th USENIX Security Symposium, USENIX Sec, 2010.

Our Solution: IMIX

- Hardware-enforced in-process memory isolation
- Isolation primitive for mitigations at page granularity
- Two separate memory realms
  - *smov* instruction to load/store sensitive data
  - *mov* instruction for regular memory

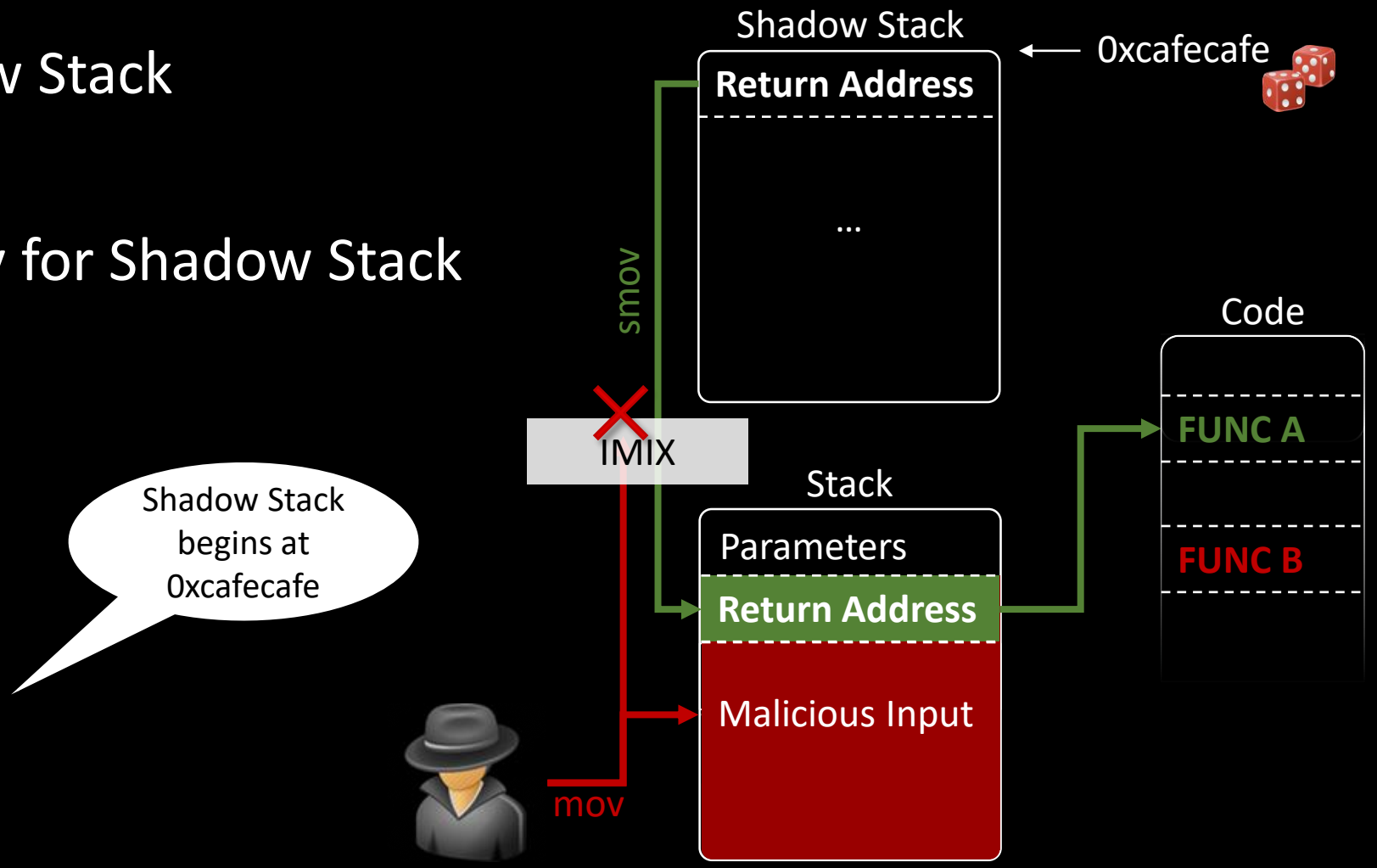


Related Work

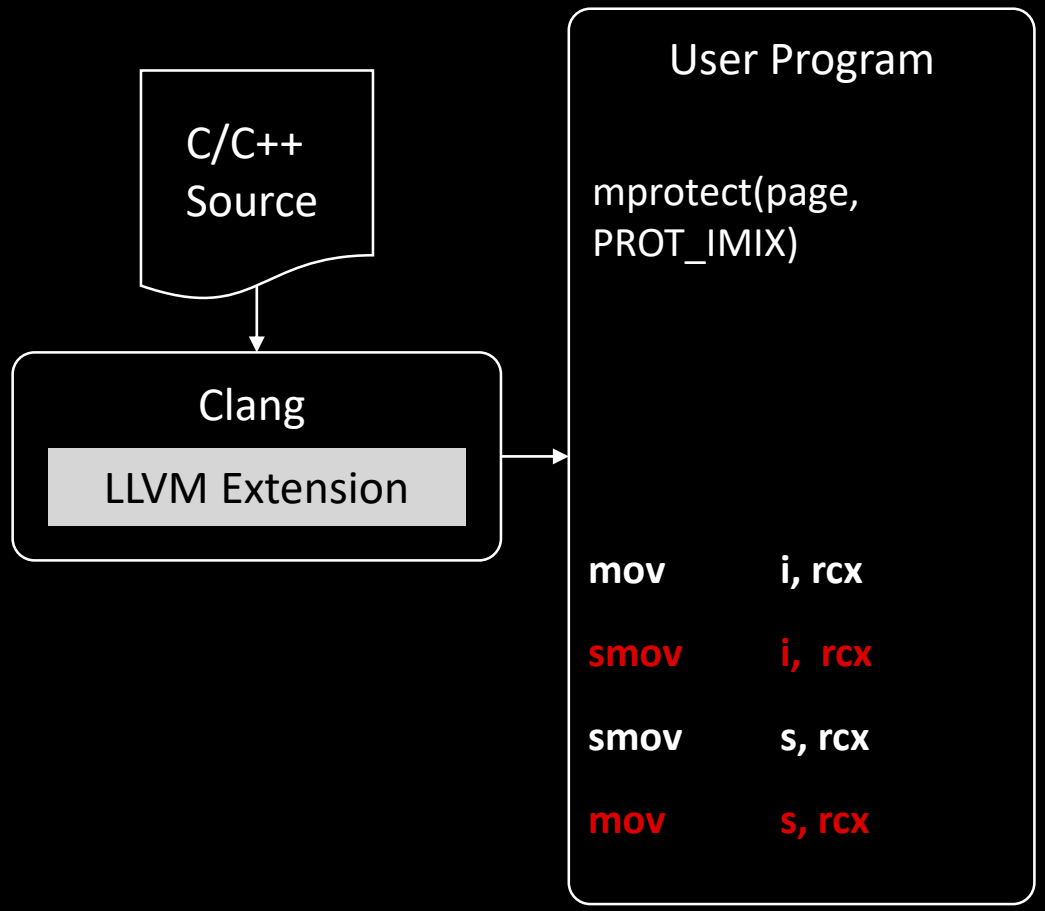
Technique	Policy-based Isolation	Hardware Enforced	Fast Interleaved Access	Fails Safe
SFI [1]	✓	✗	✓	✗
Segmentation	only for x86-32	✓	✓	✓
Memory Hiding	✗	✗	✓	✗
Paging / EPT	only single-threaded applications	✓	✗	✓
Intel MPK	✓	✓	✗	✓
Intel SGX	✓	✓	✗	✓
Intel MPX	✓	✓	(✓)	✗
Intel CET	only for Shadow Stack	✓	✓	✓
IMIX	✓	✓	✓	✓

IMIX in Action: Shadow Stack Revisited

- IMIX isolates Shadow Stack deterministically
- Exclusively use smov for Shadow Stack



Implementation

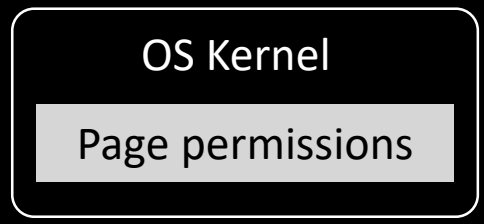


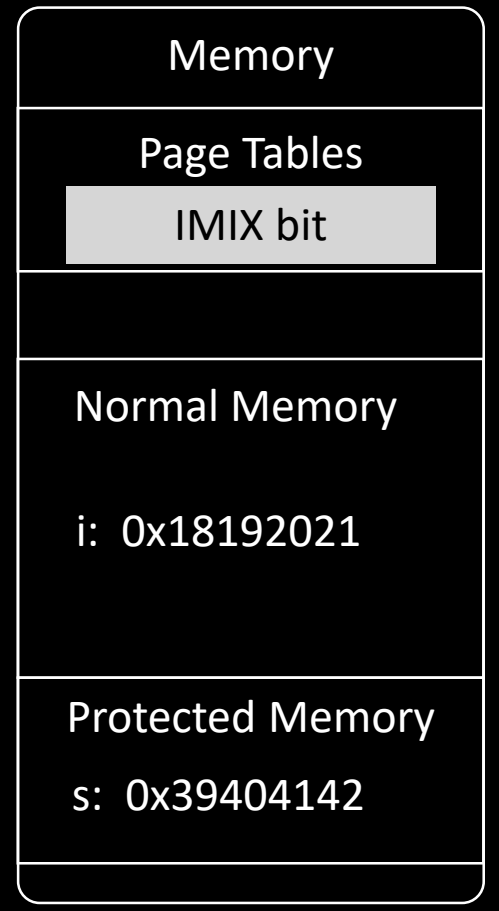
smov-based load/store for compiler IR

Implementation

Page table bit to mark page *sensitive*

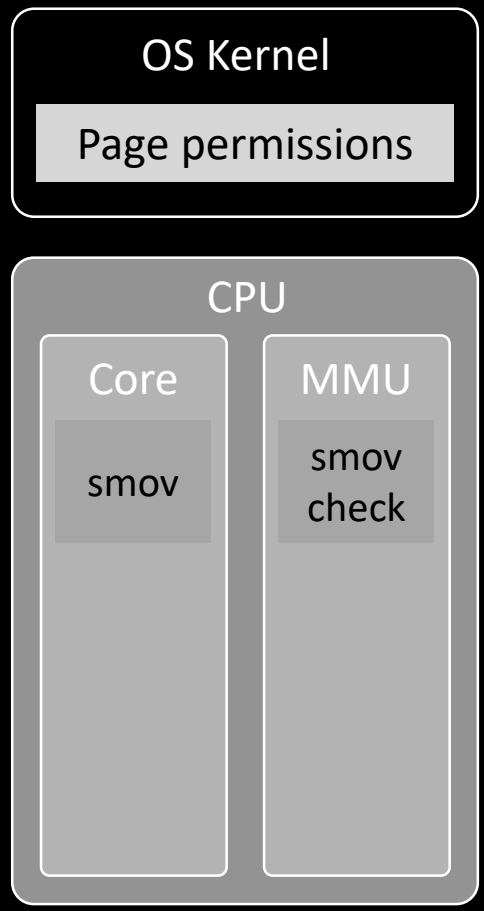
PT-bit management for Kernel



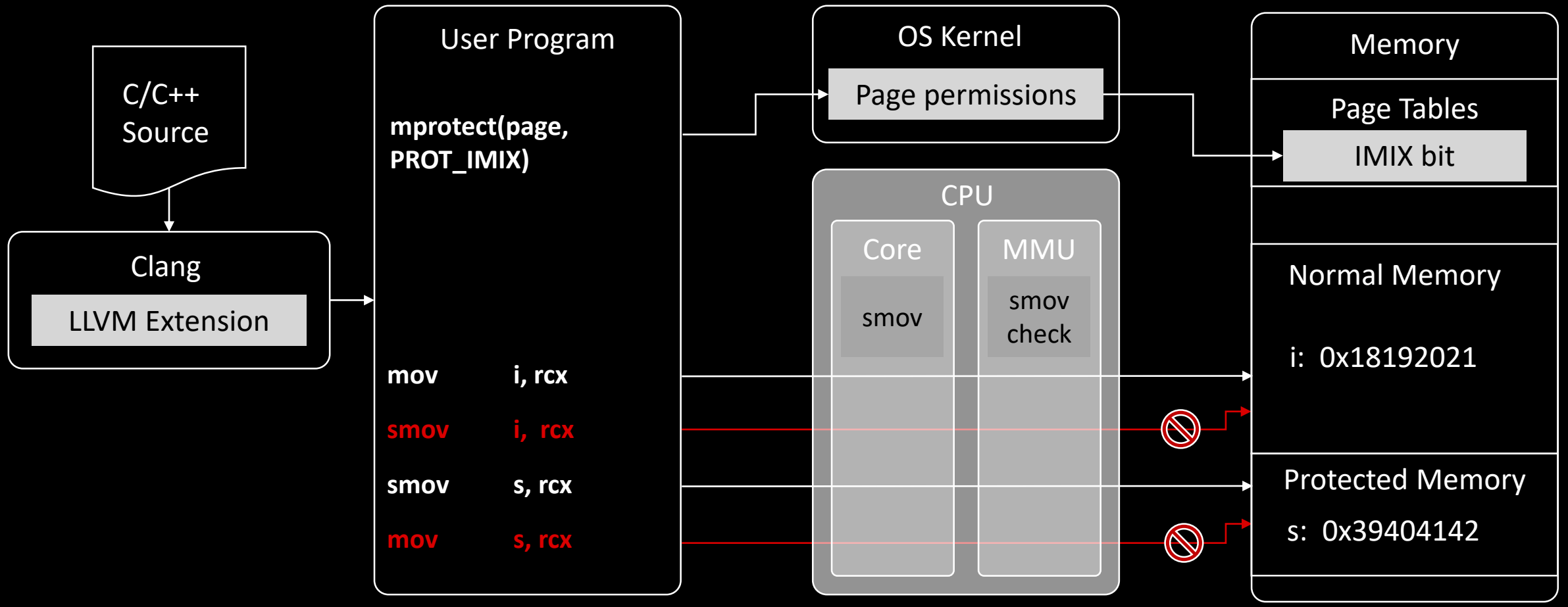


Implementation

ISA extension & MMU check

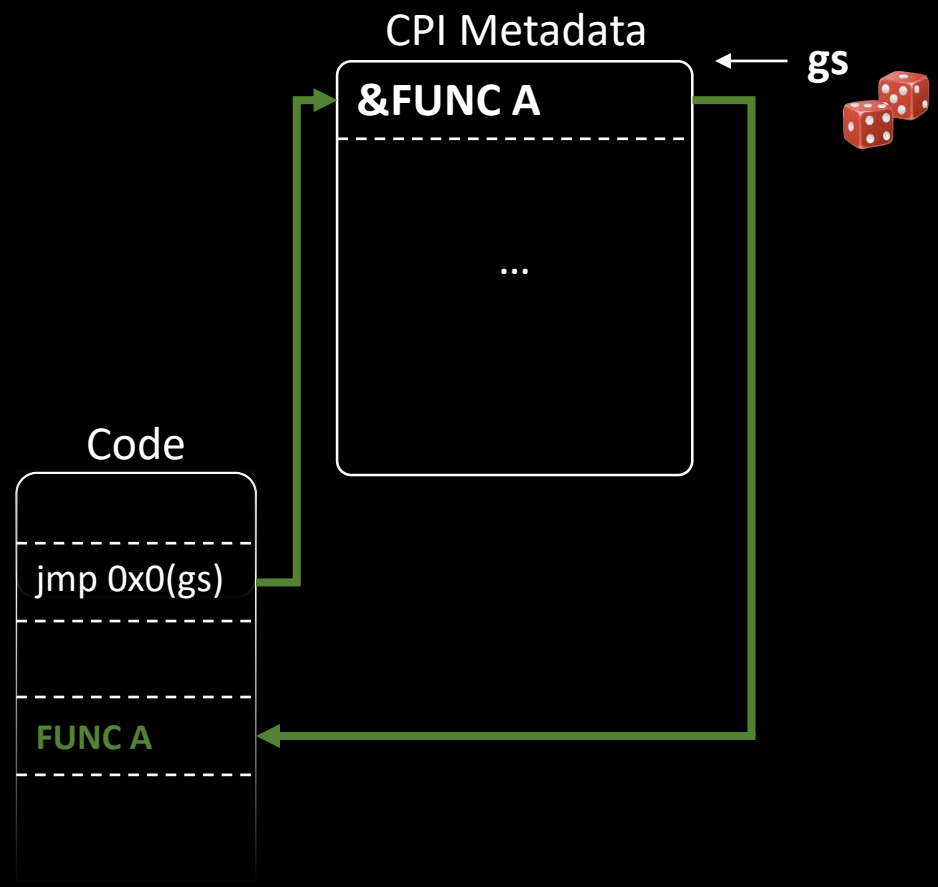


Implementation



Use-Case Evaluation: CPI

- CPI [2] prevents code-reuse attacks
- Move Code pointers and indirect code pointers to safe region → **integrity**
- BUT: safe region is only hidden – exploited by Evans et al. [3]



[2] V. Kuznetsov, L. Szekeres, M. Payer, G. Candea, R. Sekar, and D. Song. Code-pointer integrity. In 11th USENIX Symposium on Operating Systems Design and Implementation, OSDI, 2014.

[3] I. Evans, S. Fingeret, J. Gonzalez, U. Otgonbaatar, T. Tang, H. Shrobe, S. Sidiroglou-Douskos, M. Rinard, and H. Okhravi. Missing the point(er): On the effectiveness of code pointer integrity. In 36th IEEE Symposium on Security and Privacy, S&P, 2015.

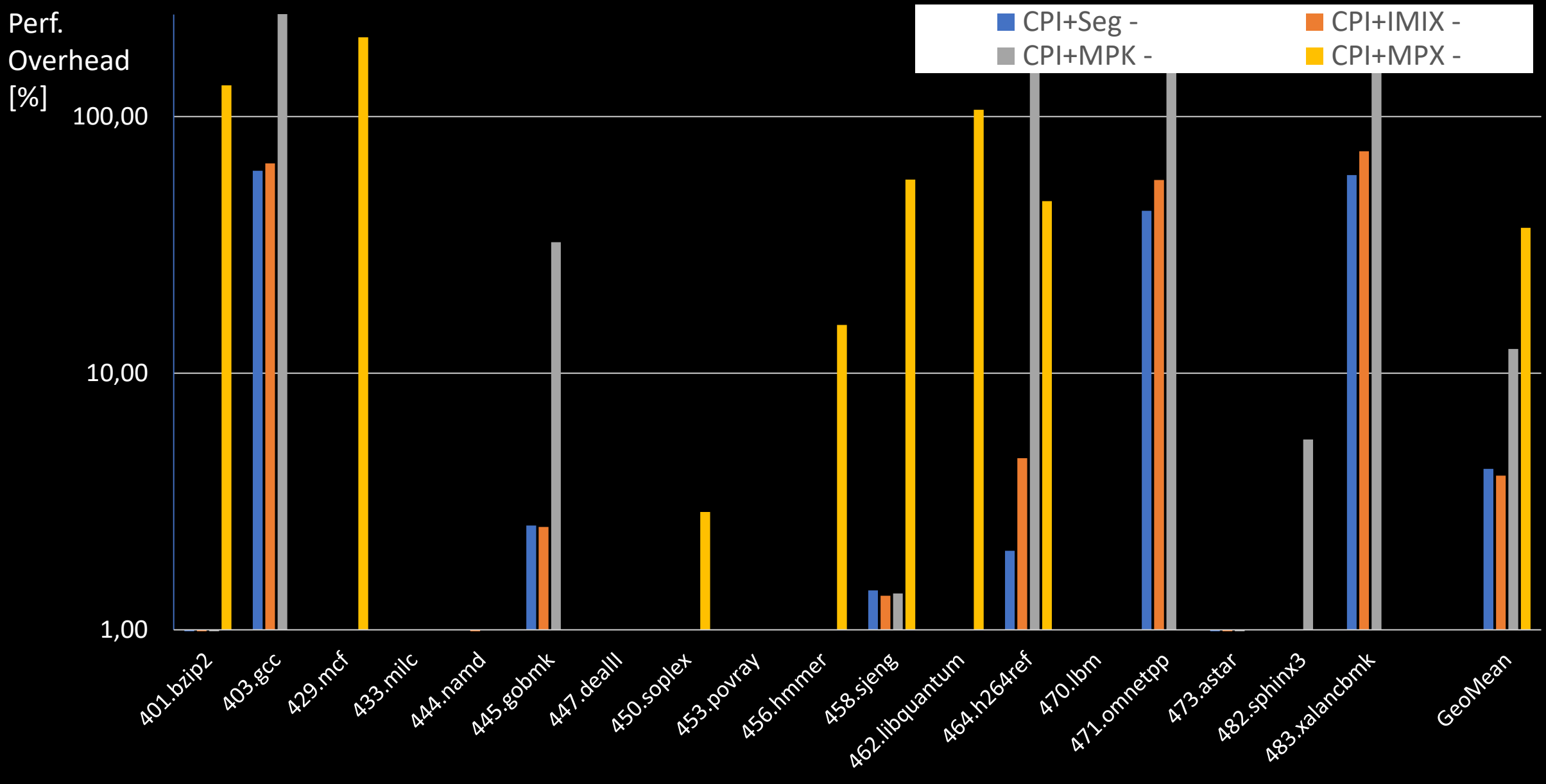
Use-Case: CPI - Replace Hiding with IMIX

- CPI evaluated different approaches for safe region protection
- Benefit: highly-modular implementation

- Added IMIX memory allocation
- Changed register-offset addressing to direct accesses

  mov 0x40(gs), ptr ➡ smov 0xcafecafe+0x40, ptr

Evaluation: CPI using IMIX



Conclusion

- IMIX is the first practical solution for in-process memory isolation
- Isolation is enforced at page granularity
- Existing approaches cannot be leveraged for CFI/CPI

Future Work

Can IMIX be adapted to protect the complete memory pipeline?

Are there new mitigation approaches that IMIX enables?

IMIX: Hardware-Enforced In-Process Memory Isolation

Tommaso Frassetto, Patrick Jauernig, Christopher Liebchen, Ahmad-Reza Sadeghi
Technische Universität Darmstadt

Research Institute for Collaborative Autonomous and Resilient Systems

TECHNISCHE UNIVERSITÄT DARMSTADT