Selective Cache Re-Mapping to Mitigate Cache Side Channel Attacks on RISC-V Processors

Pavitra Prakash Bhade
Indian Institute of Technology Goa
India

Olivier Sentieys
University of Rennes, INRIA
France

Sharad Sinha
Indian Institute of Technology Goa
India

Introduction and Contributions

- Cache side channel attacks (CSCA) have become a significant threat to shared cache security.
- Fortifying cache security by encrypting mappings to thwart eviction-based attacks (e.g., Flush+Reload, Evict+Abort).
- Encryption of mappings induces performance overhead.
- Since the entire address space is involved, predictive analysis can break the encryption.

- **Approach:** selective randomization that encrypts cache mappings of only specific sections clashing with protected memory regions.

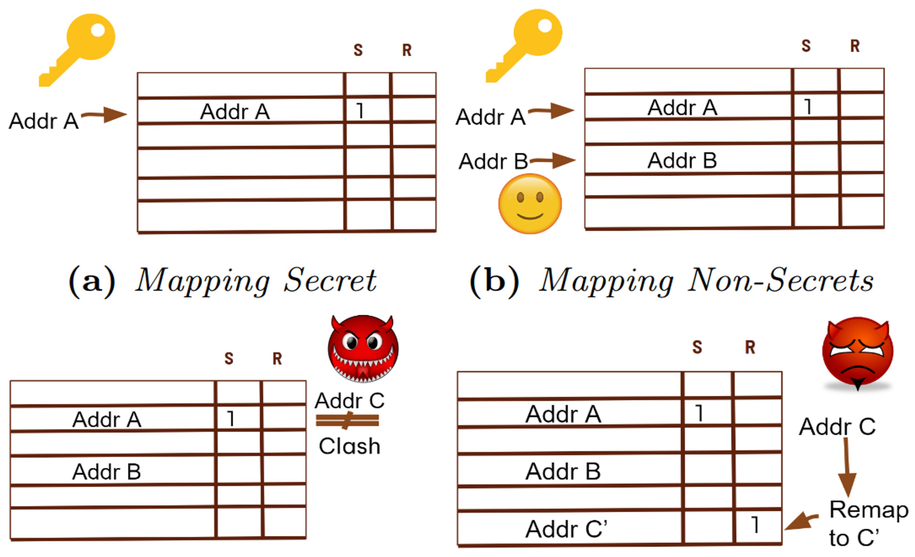
- Instead of mapping the entire address range, we propose a selective mapping technique performing encryptions only when and where needed.
- Reduces the impact on performance overhead and the possibility of breaking encryptions by predictive analyses.
- Only a marginal addition to the microarchitecture and cache replacement policy.

- Very suitable for RISC-V architecture specifically.
- Hardware implementation of the proposed mitigation technique in the *Comet*[1] RISC-V core.

Proposed Algorithms

Cache Replacement

- ***if Address A has to evict Address B then***
  - ***if SecretBit(B) == 0 then***
    - ***Replace B with A***
    - ***if A is secret :SecretBit(A) <= 1***
    - ***end if***
  - ***else***
    - ***Encrypt Address A to A'***
    - ***Search replacement location for A'***
    - ***Replace at the found location***
    - ***RemapBit(A') <= 1***
    - ***If A is secret : SecretBit(A') <=1, else SecretBit(A') <=0***
  - ***end if***
- ***end if***



(a) *Mapping Secret* (b) *Mapping Non-Secrets*

(c) *Conflict with secret section* (d) *Conflicting address remapped*

Figure 1: *Proposed mapping and re-mapping technique*

Cache Lookup

- ***if Address A present in cache then***
  - ***if RemapBit(A) == 0 then***
    - ***Cache Hit***
  - ***end if***
- ***else***
  - ***Encrypt A to A'***
  - ***if Address A' present in cache then***
    - ***if RemapBit(A') == 1 then***
      - ***Cache Hit***
    - ***end if***
  - ***end if***
- ***end if***

Results

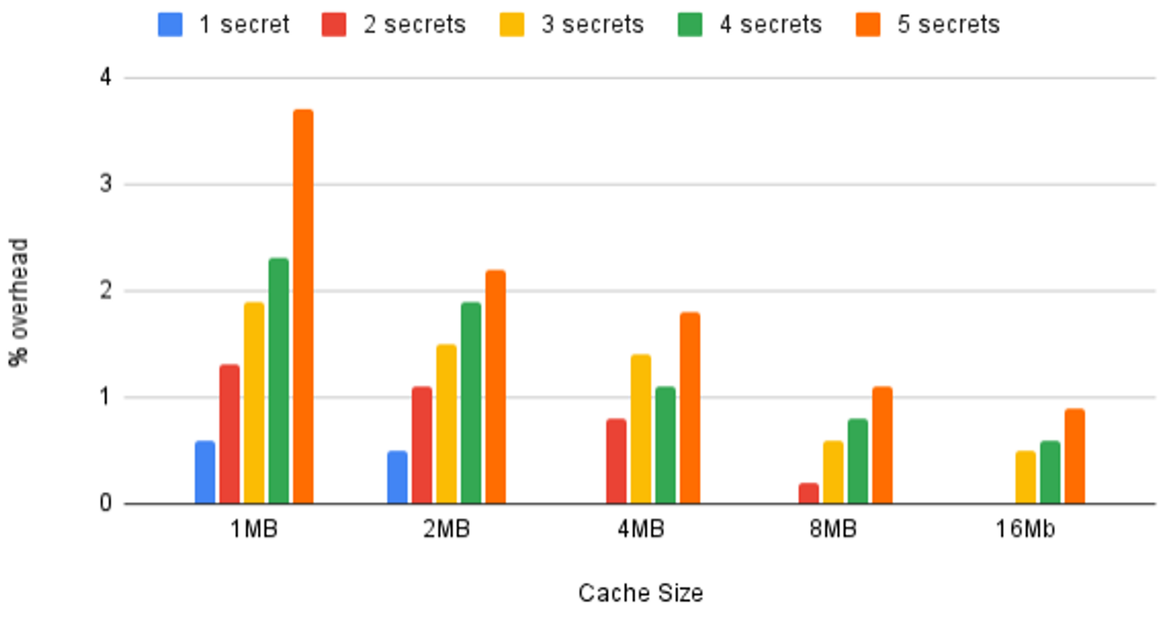
Overhead in % on RSA 512 execution

Parameters	Number of secrets (Cache = 1MB, Block = 64B)					
	0	1	2	3	4	5
Remaps	0	+0.8	+1.8	+3.98	+4.3	+4.7
Hits	0	+1.13	-0.88	-0.62	-2.7	-2.9
Misses	0	-0.7	+3.5	+4.2	+5.9	+6
Execution Time	+0.7	+0.79	+0.9	+1.16	+1.92	+2.1

Overhead in % on AES 128 execution

Parameters	Number of secrets (Cache = 1MB, Block = 64B)					
	0	1	2	3	4	5
Remaps	0	+0.6	+1.8	+3.98	+4.3	+4.7
Hits	0	+0.8	-0.88	-0.62	-2.7	-2.9
Misses	0	-1	+3.5	+4.2	+5.9	+6
Execution Time	+0.9	+1.2	+0.9	+1.16	+1.92	+2.1

Variation in number of Remaps, with increasing cache size



Conclusion

- A minimal enhancement in the cache microarchitecture and replacement policy to mitigate conflict-based CSCA.
- Measure performance overhead for 0 to 5 secret processes being protected simultaneously.
- Two additional bits per 64B cache line used as metadata for mapping and replacement (area overhead ~0.3%).

- Future works: synthesizing the proposed cache structure in various RISC-V microarchitectures to gain deeper insights into security and performance evaluations; studying the evaluation of hardware-level encryptors in RISC-V cores.

[1]S. Rokicki, D. Pala, J. Paturel, O. Sentieys, What You Simulate Is What You Synthesize: Designing a Processor Core from C++ Specifications, IEEE/ACM ICCAD, 2019. https://gitlab.inria.fr/srokicki/Comet

RISC V SUMMIT EUROPE 2024 (Munich, Germany)