AutoLock: Why Cache Attacks on ARM Are Harder Than You Think

Marc Green[W], Leandro Rodrigues-Lima[F], Andreas Zankl[F], Gorka Irazoqui[W], Johann Heyszl[F], and Thomas Eisenbarth[W]

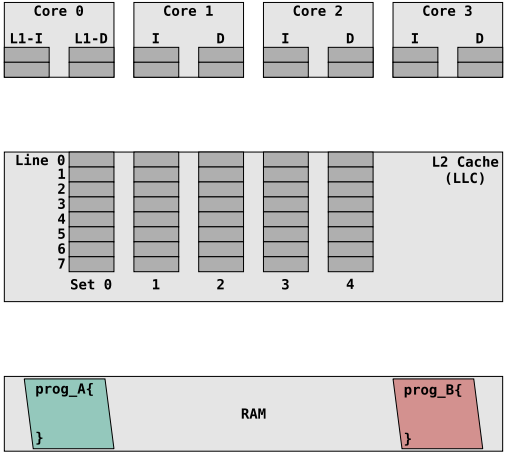
August 18th 2017 – USENIX Security Symposium

The Big Picture

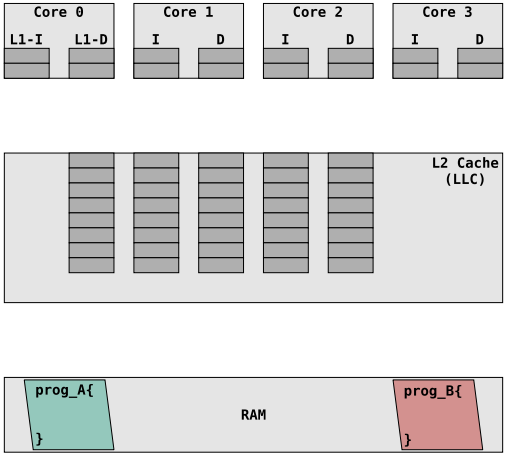
Cache Attacks		AutoLock
Transition of attacks: desktop & server → mobile	⟶	Dedicated countermeasures are still necessary for protection
Vulnerabililty and risk assessment are important, but challenging	⟶	Eviction-based attacks are harder than previously believed
Limited understanding of commercial microarchitectures	⟶	Undocumented performance feature of inclusive caches on ARM

Cache Basics

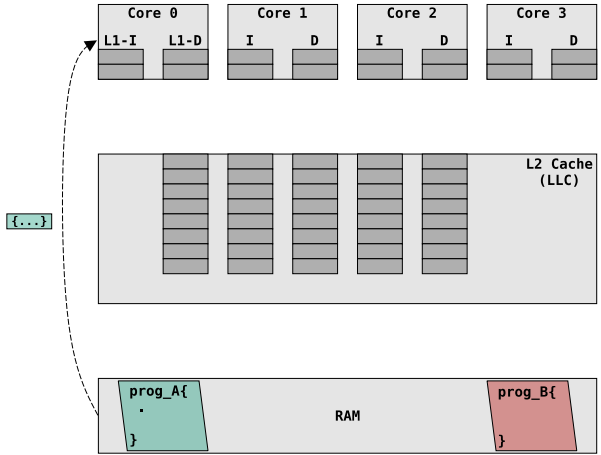
Cache Basics
Hierarchy



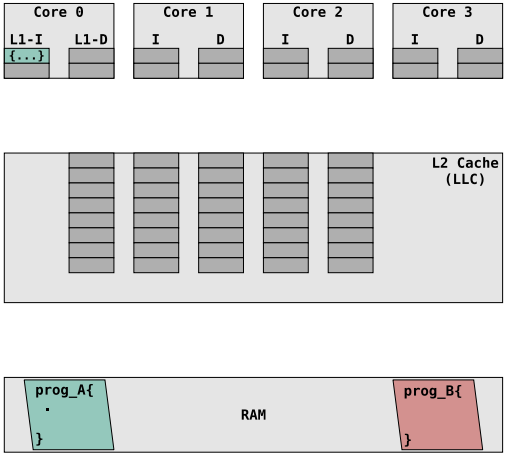
Cache Basics
Hierarchy



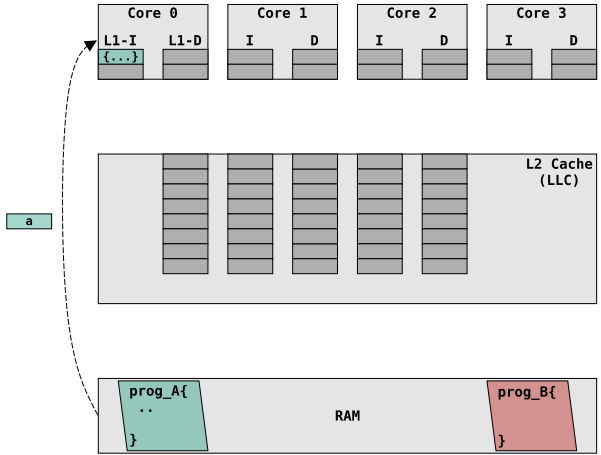
Cache Basics
Hierarchy



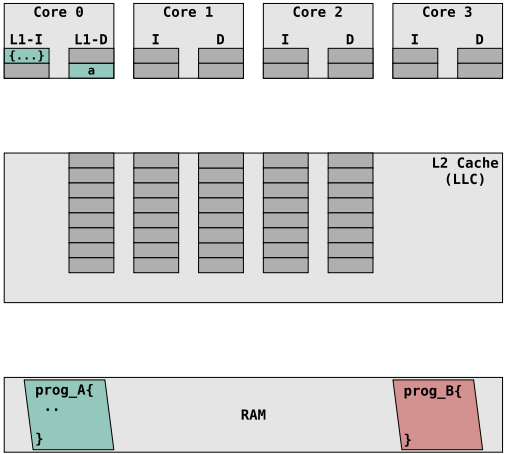
Cache Basics
Hierarchy



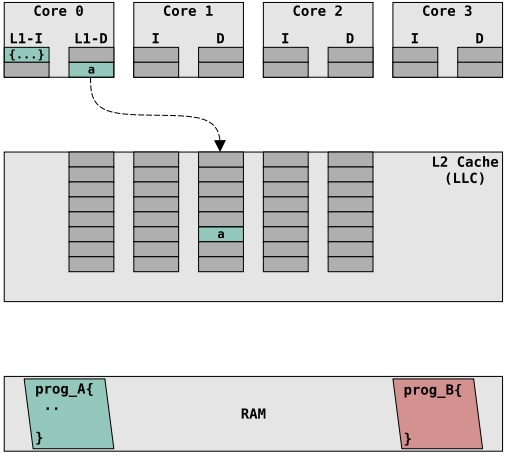
Cache Basics
Hierarchy



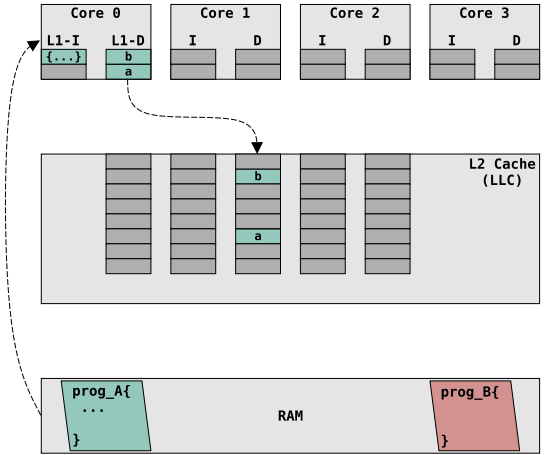
Cache Basics
Hierarchy



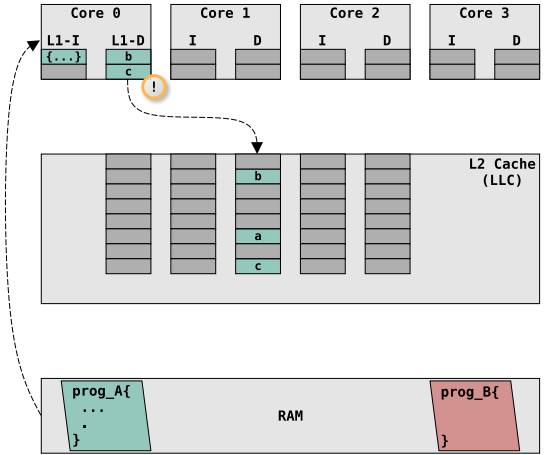
Cache Basics
Inclusiveness



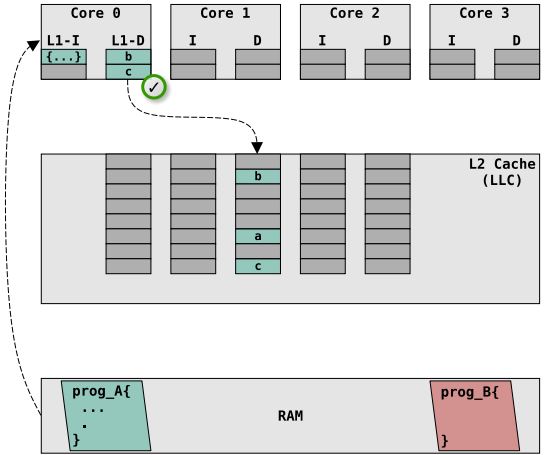
Cache Basics
Inclusiveness



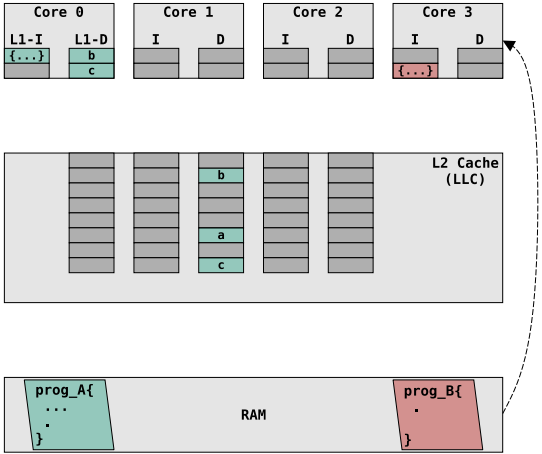
Cache Basics
Inclusiveness



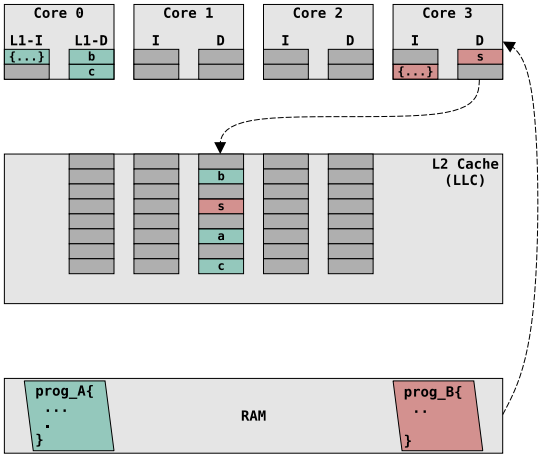
Cache Basics
Inclusiveness



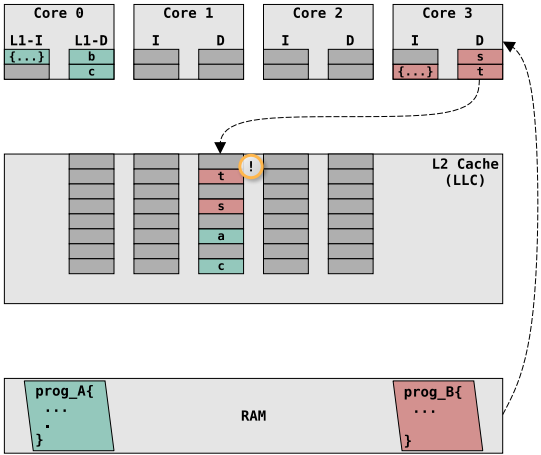
Cache Basics
Inclusiveness



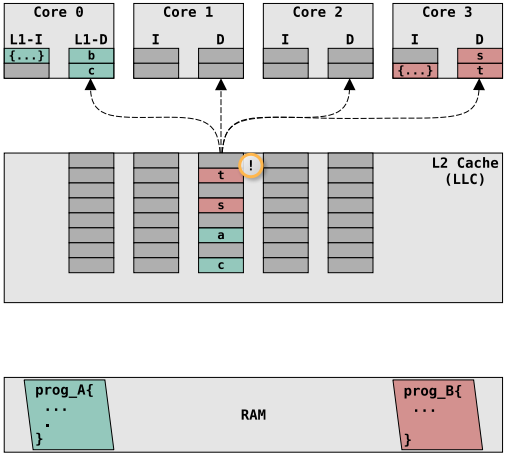
Cache Basics
Inclusiveness



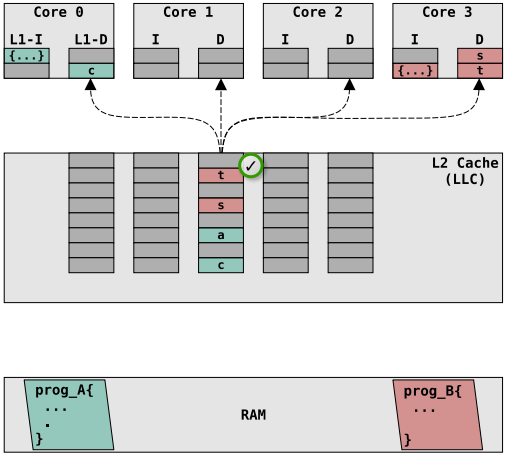
Cache Basics
Inclusiveness



Cache Basics
Inclusiveness

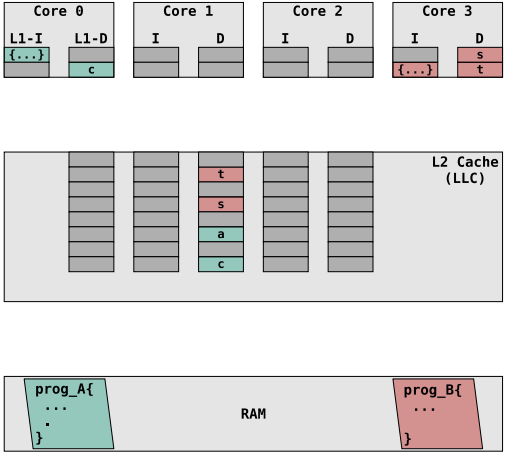


Cache Basics
Inclusiveness

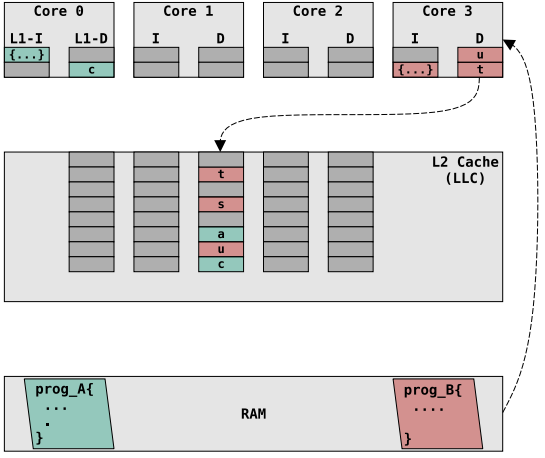


Cache Attacks

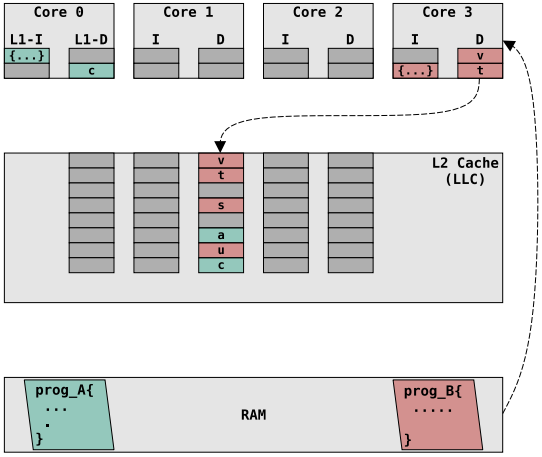
Cache Attacks
Cross-core Eviction



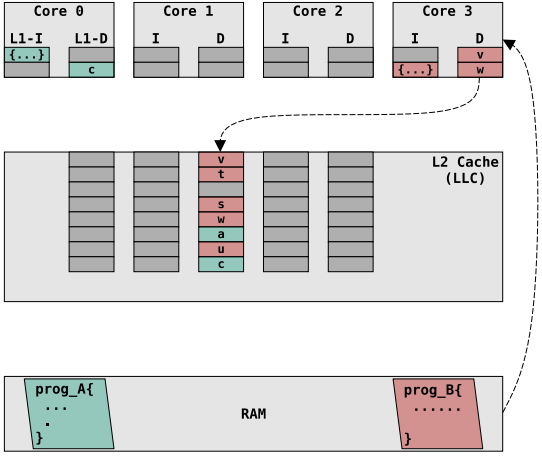
Cache Attacks
Cross-core Eviction



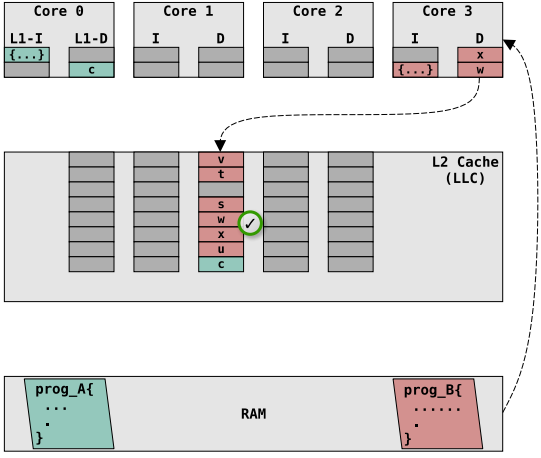
Cache Attacks
Cross-core Eviction



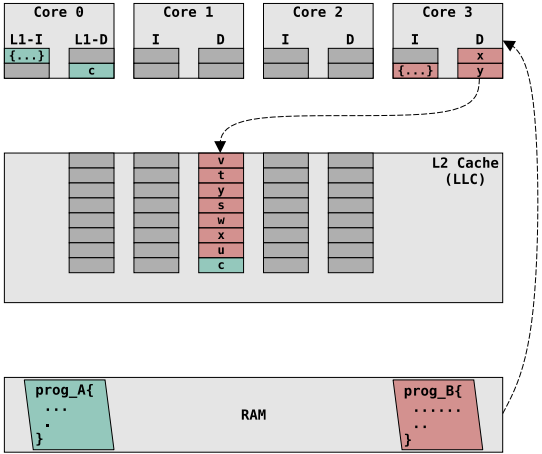
Cache Attacks
Cross-core Eviction



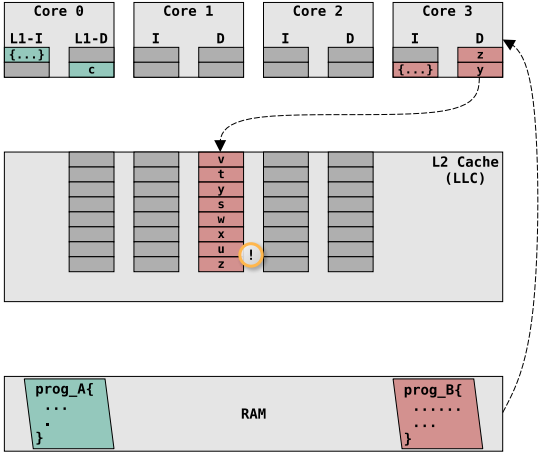
Cache Attacks
Cross-core Eviction



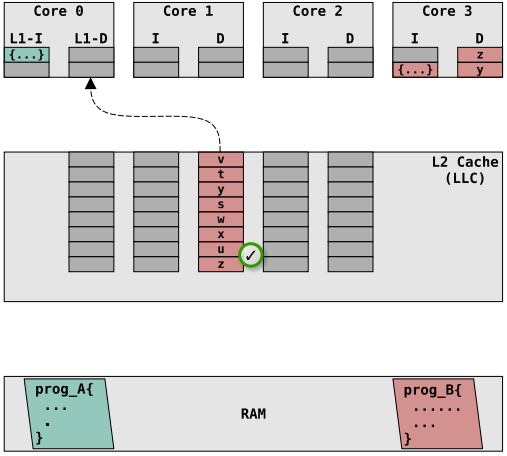
Cache Attacks
Cross-core Eviction



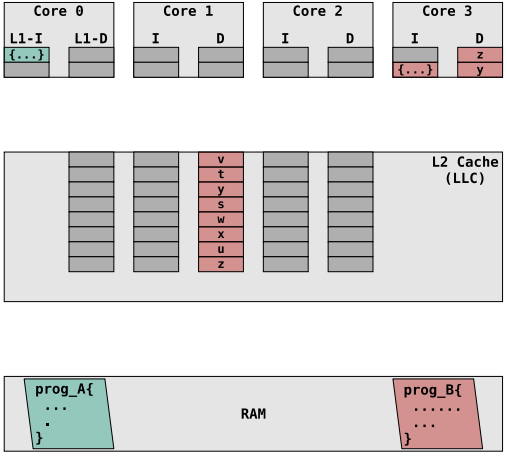
Cache Attacks
Cross-core Eviction



Cache Attacks
Cross-core Eviction

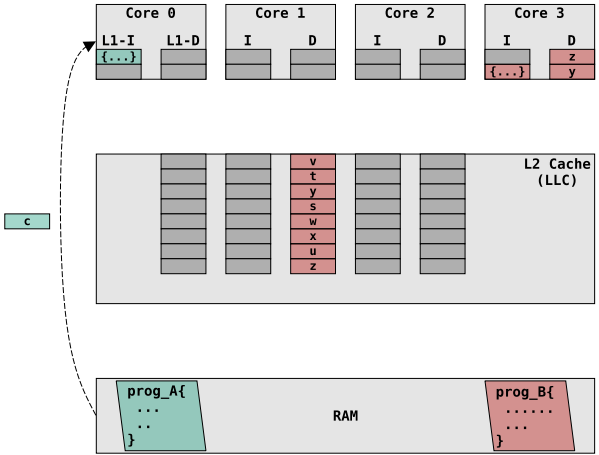


Cache Attacks
Cross-core Eviction

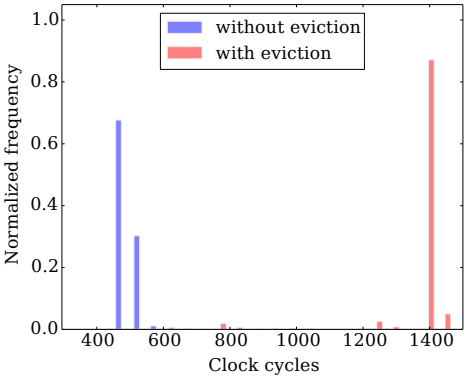


Cache Attacks

Cross-core Eviction



Cache Attacks
Cross-core Eviction



Qualcomm Krait 450

Cache Attacks
Literature

Method	Task	Reference
Evict + Time	Eviction	[OST06]
Prime + Probe	Eviction	[OST06]
Evict + Reload	Eviction	[GSM15]
Evict + Prefetch	Eviction	[GMF+16]
Flush + Reload	Flush	[YF14]
Flush + Prefetch	Flush	[GMF+16]
Flush + Flush	Flush	[GMWM16]

Cache Attacks
ARM Processors

Flush on ARM

- Easy, fast, and robust
- Only from ARMv8 onwards
- No guaranteed userspace access

Limited number of devices

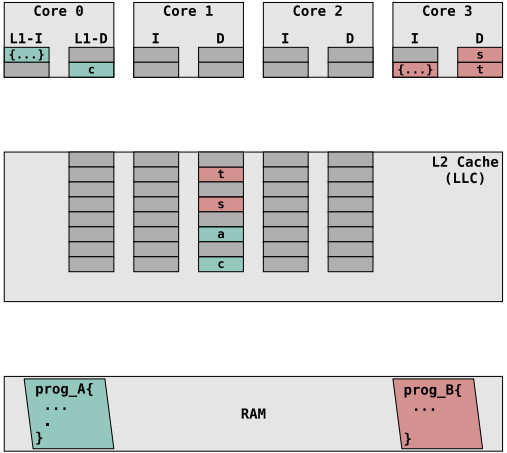
Eviction on ARM

- Complex, slow, and error-prone
- All ARM architectures: v6, v7, v8
- Userspace privileges are sufficient

Larger number of devices

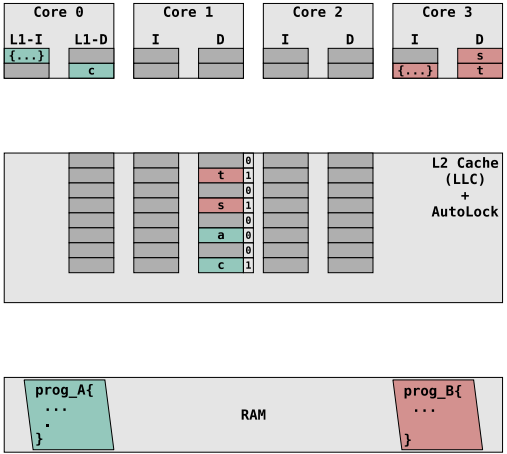
AutoLock

AutoLock
Cross-core Eviction



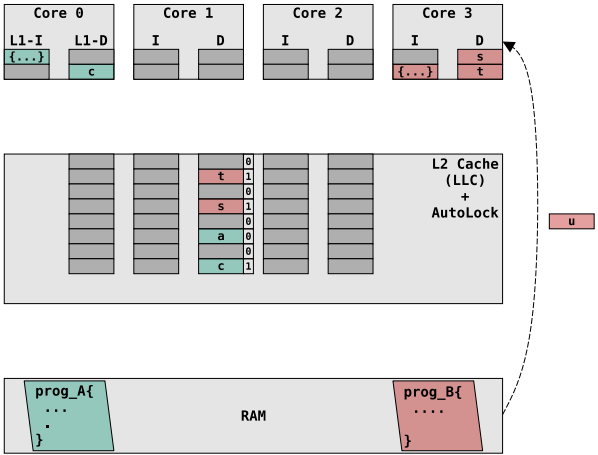
AutoLock

Cross-core Eviction



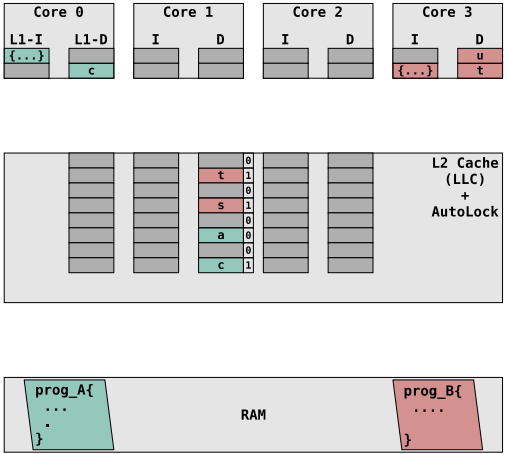
AutoLock

Cross-core Eviction



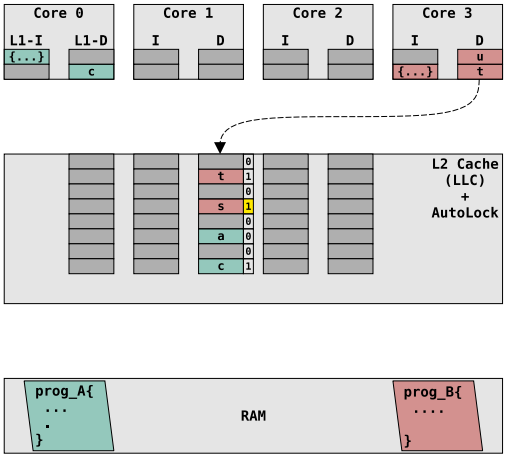
AutoLock

Cross-core Eviction



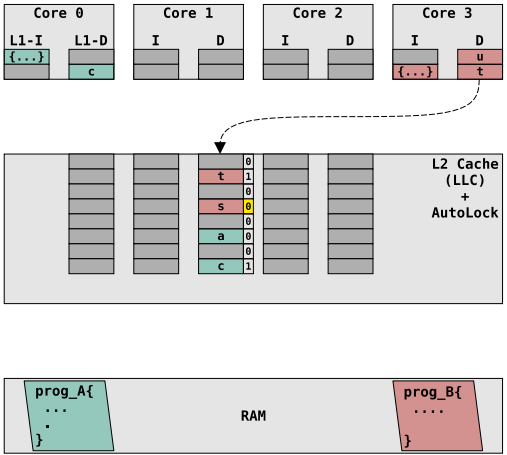
AutoLock

Cross-core Eviction



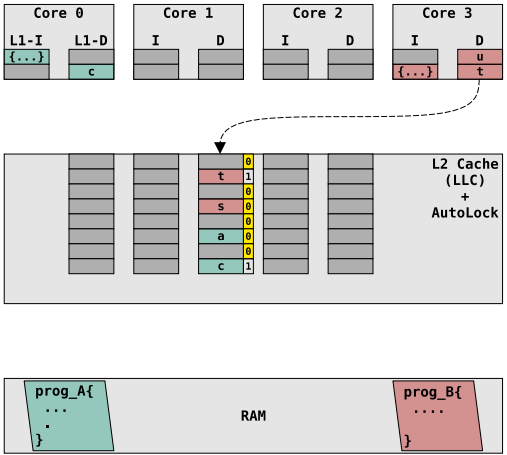
AutoLock

Cross-core Eviction



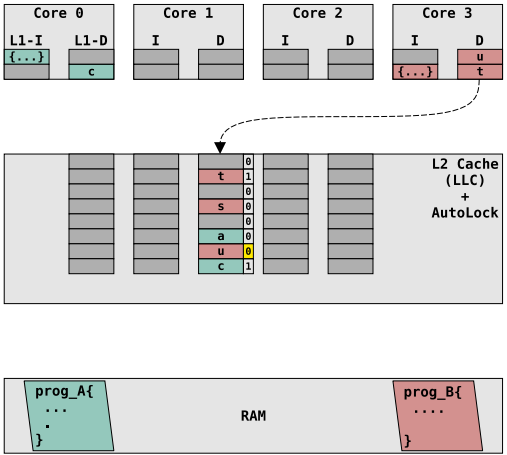
AutoLock

Cross-core Eviction



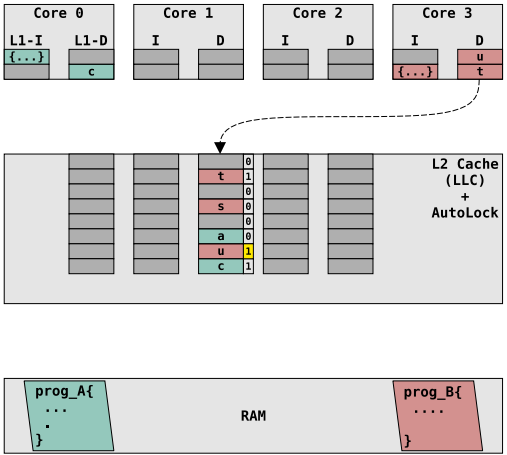
AutoLock

Cross-core Eviction



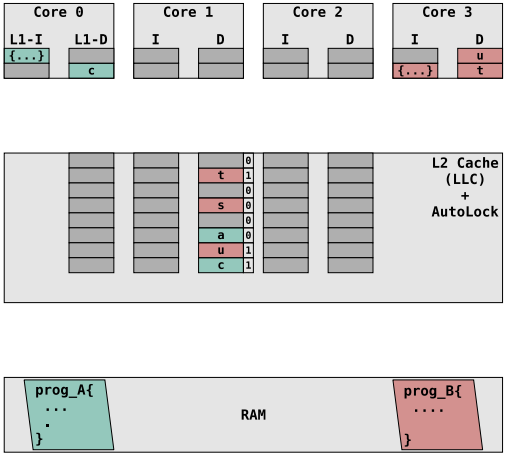
AutoLock

Cross-core Eviction



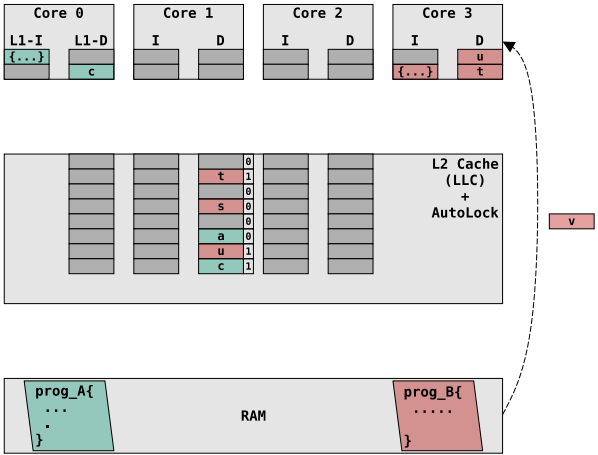
AutoLock

Cross-core Eviction



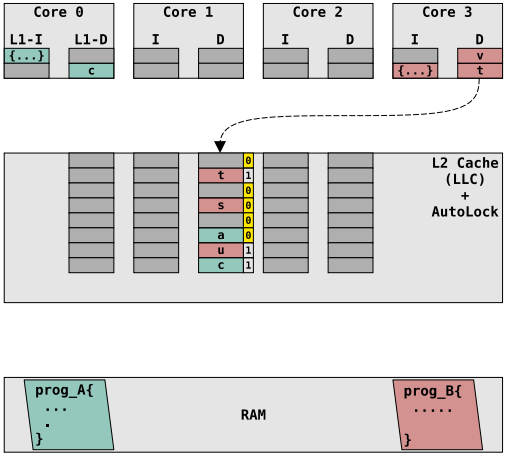
AutoLock

Cross-core Eviction



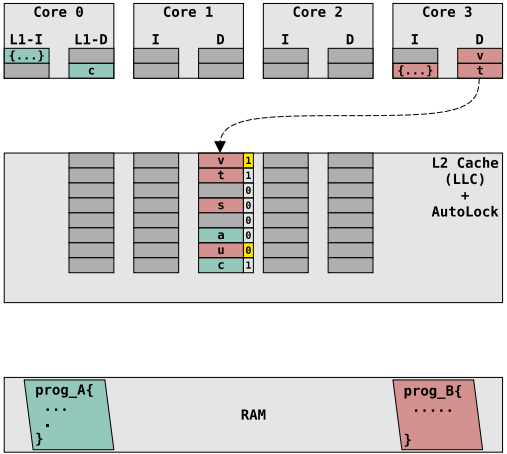
AutoLock

Cross-core Eviction



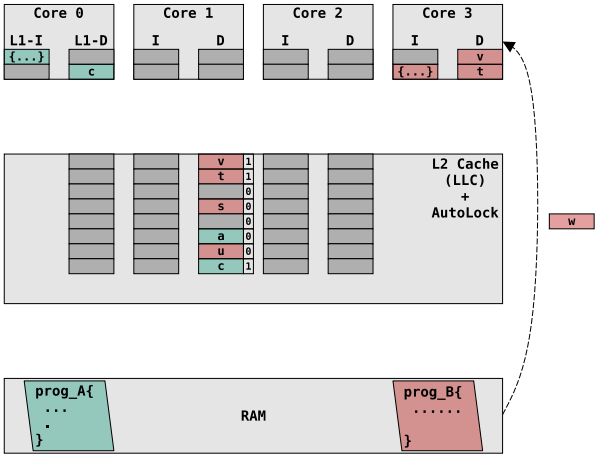
AutoLock

Cross-core Eviction



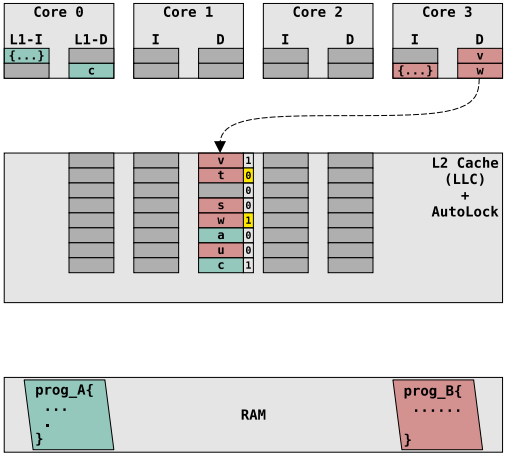
AutoLock

Cross-core Eviction



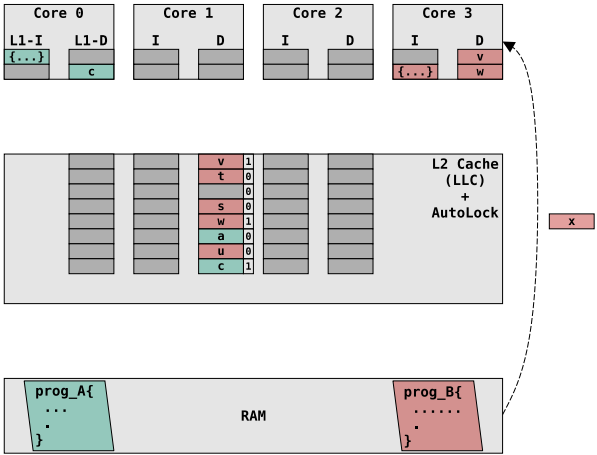
AutoLock

Cross-core Eviction



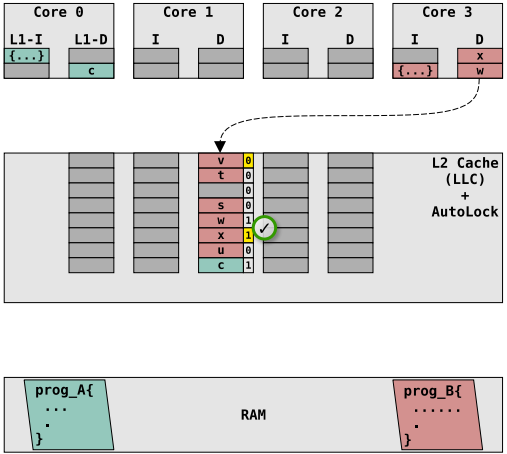
AutoLock

Cross-core Eviction



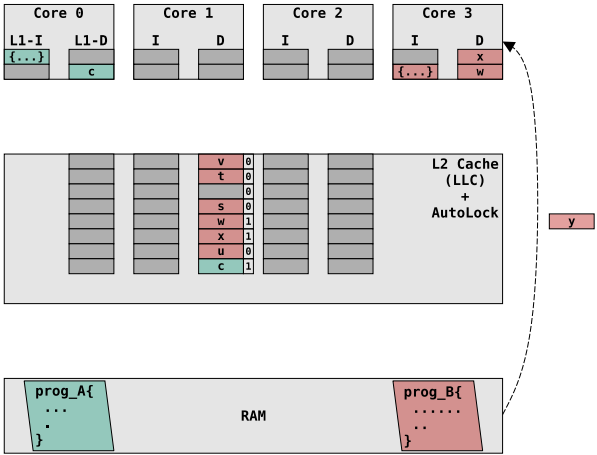
AutoLock

Cross-core Eviction



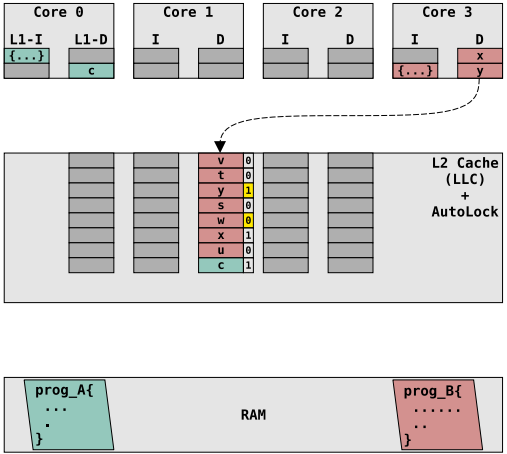
AutoLock

Cross-core Eviction



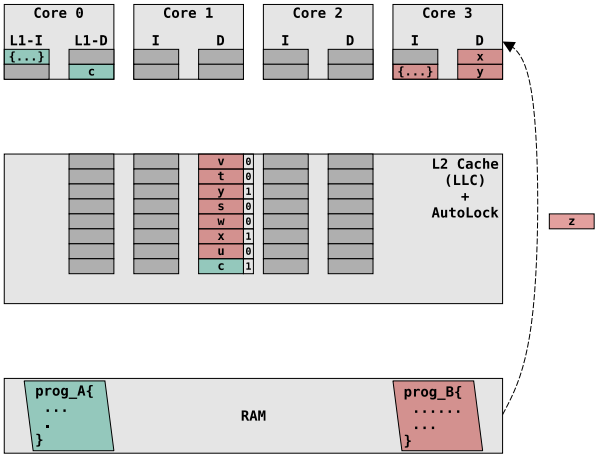
AutoLock

Cross-core Eviction



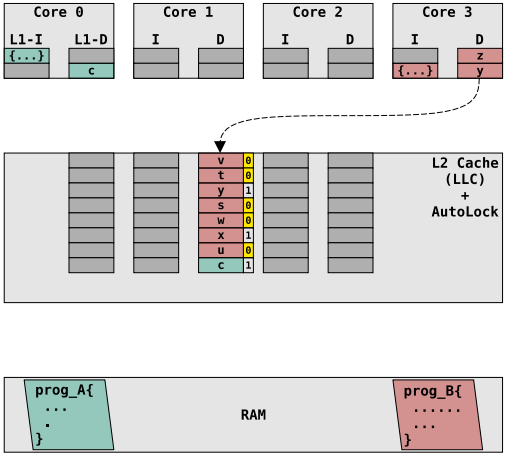
AutoLock

Cross-core Eviction



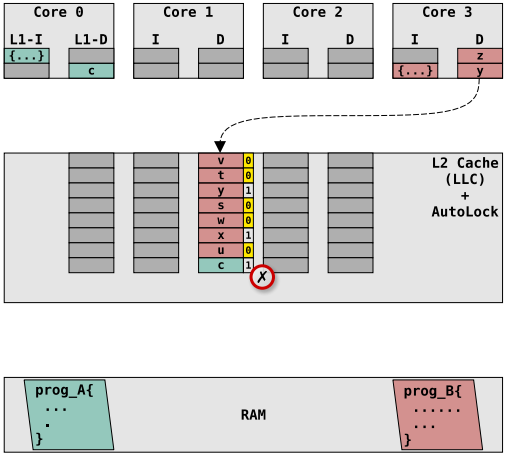
AutoLock

Cross-core Eviction



AutoLock

Cross-core Eviction



AutoLock
Definition

A patented and undocumented performance feature of inclusive cache levels that transparently prevents the eviction of cache lines, if they are contained in higher cache levels.

Automatic + Lockdown = "AutoLock"

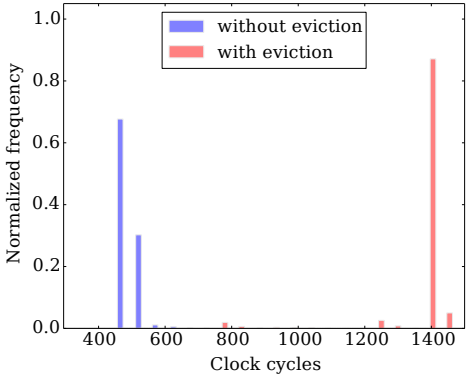
Implications of `AutoLock`

Implications
Impact in Theory

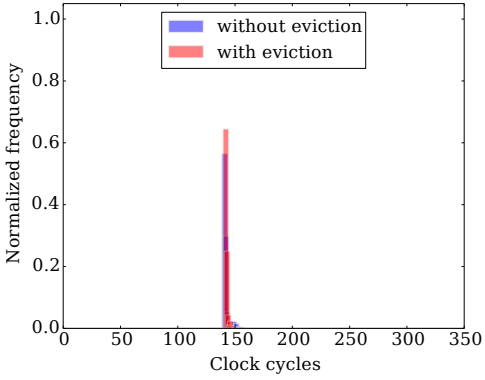
Method	Task	Same-core	Cross-core
Evict + Time	Eviction	✓	✗
Prime + Probe	Eviction	✓	✗
Evict + Reload	Eviction	✓	✗
Evict + Prefetch	Eviction	✓	✗
Flush + *	Flush	✓	✓

Implications

Impact in Practice



Qualcomm Krait 450



ARM Cortex-A57

Implications
SoC Evaluation

Processor	System-on-Chip (SoC)	AutoLock
ARM Cortex-A7	Samsung Exynos 5422	✓
ARM Cortex-A15	Samsung Exynos 5422/5250	✓
ARM Cortex-A53	ARM Juno r0	✓
ARM Cortex-A57	ARM Juno r0	✓
Qualcomm Krait 450	Snapdragon 805	✗

Implications
Smartphone SoCs

Manufacturer	SoC Family	% featuring A7,-15,-53,-57
Apple	A10, A9, A8, A7	0
HiSilicon	Kirin 9xx, 6xx	86
Mediatek	MT67xx, MT659x/8x	100
Nvidia	Tegra X, K, 4	71
Qualcomm	Snapdragon 8xx, 6xx, 4xx	47
Samsung	Exynos 9, 8, 7, 5, 4	79
Xiaomi	Surge S	100

Implications
Previous Work on ARM

"ARMageddon"
Lipp et al. [LGS+16]

⇒ Device Selection

- Qualcomm SoCs
- Userspace flush
- Cross-core eviction

"ROP Flush + Reload"
Zhang et al. [ZXZ16]

⇒ Attack Selection

- Flush + Reload
- `cacheflush` syscall
- No cross-core eviction

"TruSpy"
Zhang et al. [ZSS+16]

⇒ Device Properties

- ARM Cortex-A8
- Single-core setup
- No cross-core eviction

AutoLock = Countermeasure?

Countermeasure?
Not Ultimately

Vulnerable SoCs

- ARM-compliant cores
- Userspace flush instr.

Same-Core Attacks

- ARM TrustZone
- Compromised OS

Remote Evictions

- Trigger self-evictions
- Increase load and wait time

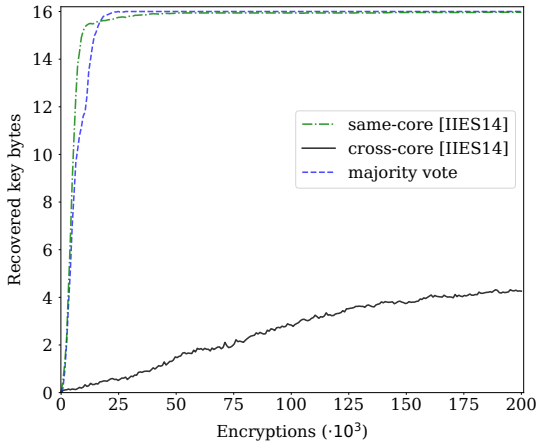
Redundant Targets

- Attack multiple cache lines
- e.g. entire AES T-tables

Countermeasure?

Wait-a-minute Attack

- Attack by Irazoqui et al. [IIES14]
  - Observes usage of AES T-tables
  - One cache line per table
- Simple redundant variant
  - Observe all lines of all tables
  - Majority vote on derived keys
- Test environment
  - ARM Cortex-A15 with `AutoLock`
  - Full Linux operating system



Conclusion

Conclusion
Takeaways

AutoLock: undocumented feature of inclusive LLCs on ARM

Inhibits cross-core eviction and adversely affects attacks

Predominantly implemented in Cortex-A designs by ARM

Countermeasures are still necessary to protect against attacks

Questions ?

Bibliography

[GMF+16] Daniel Gruss, Clémentine Maurice, Anders Fogh, Moritz Lipp, and Stefan Mangard.
Prefetch side-channel attacks: Bypassing smap and kernel aslr.
In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 368–379, New York, NY, USA, 2016. ACM.

[GMWM16] Daniel Gruss, Clémentine Maurice, Klaus Wagner, and Stefan Mangard.
Flush+flush: A fast and stealthy cache attack.
In Juan Caballero, Urko Zurutuza, and Ricardo J. Rodríguez, editors, *Detection of Intrusions and Malware, and Vulnerability Assessment: 13th International Conference, DIMVA 2016, San Sebastián, Spain, July 7-8, 2016, Proceedings*, pages 279–299, Cham, 2016. Springer International Publishing.

[GSM15] Daniel Gruss, Raphael Spreitzer, and Stefan Mangard.
Cache template attacks: Automating attacks on inclusive last-level caches.
In *24th USENIX Security Symposium (USENIX Security 15)*, pages 897–912, Washington, D.C., 2015. USENIX Association.

[IIES14] Gorka Irazoqui, Mehmet Sinan Inci, Thomas Eisenbarth, and Berk Sunar.
Wait a minute! a fast, cross-vm attack on aes.
In Angelos Stavrou, Herbert Bos, and Georgios Portokalidis, editors, *Research in Attacks, Intrusions and Defenses: 17th International Symposium, RAID 2014, Gothenburg, Sweden, September 17-19, 2014. Proceedings*, pages 299–319, Cham, 2014. Springer International Publishing.

[LGS+16] Moritz Lipp, Daniel Gruss, Raphael Spreitzer, Clémentine Maurice, and Stefan Mangard.
Armageddon: Cache attacks on mobile devices.
In *25th USENIX Security Symposium (USENIX Security 16)*, pages 549–564, Austin, TX, 2016. USENIX Association.

[OST06] Dag Arne Osvik, Adi Shamir, and Eran Tromer.
Cache attacks and countermeasures: The case of aes.
In David Pointcheval, editor, *Topics in Cryptology – CT-RSA 2006: The Cryptographers' Track at the RSA Conference 2006, San Jose, CA, USA, February 13-17, 2005. Proceedings*, pages 1–20, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

Bibliography

[YF14] Yuval Yarom and Katrina Falkner.
Flush+reload: A high resolution, low noise, l3 cache side-channel attack.
In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 719–732, San Diego, CA, 2014. USENIX Association.

[ZSS+16] Ning Zhang, Kun Sun, Deborah Shands, Wenjing Lou, and Y. Thomas Hou.
Truspy: Cache side-channel information leakage from the secure world on arm devices.
Cryptology ePrint Archive, Report 2016/980, 2016.
http://eprint.iacr.org/2016/980.

[ZXZ16] Xiaokuan Zhang, Yuan Xiao, and Yinqian Zhang.
Return-oriented flush-reload side channels on arm and their implications for android devices.
In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 858–870, New York, NY, USA, 2016. ACM.