

Memory Integrity

Cédric Lauradoux

October 20, 2011

Outline

▶ **Problems and challenges**

- Assumptions;
- Integrity and secure deletion;
- Metrics.

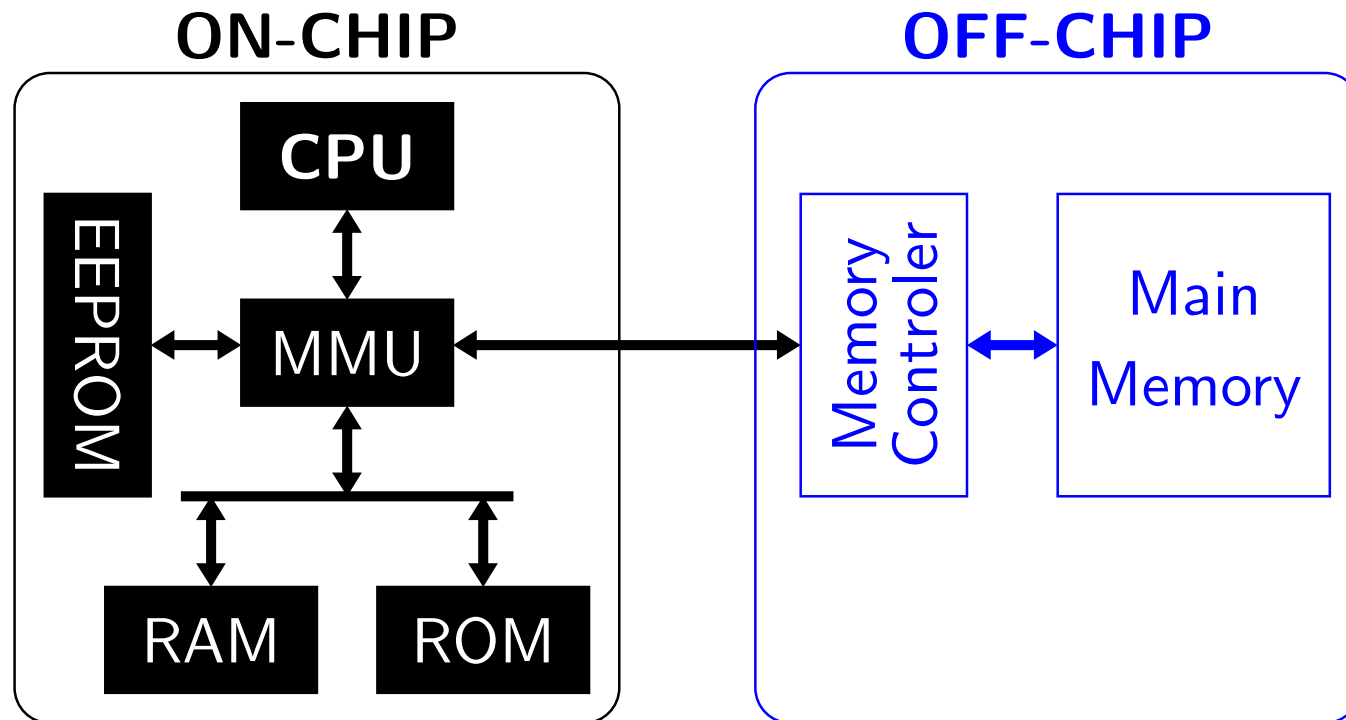
▶ **Integrity**

- HASH;
- COUNTER;
- MERKLE'S TREE.

▶ **Secure deletion** of SSD/PCM memories with wear-leveling

Problems and challenges

Assumptions



- ▷ Software is secure. (secrets cannot leak.)
- ▷ On-chip components are secure.
- ▷ Off-chip components are controlled at time t by the adversary.

Problems and challenges

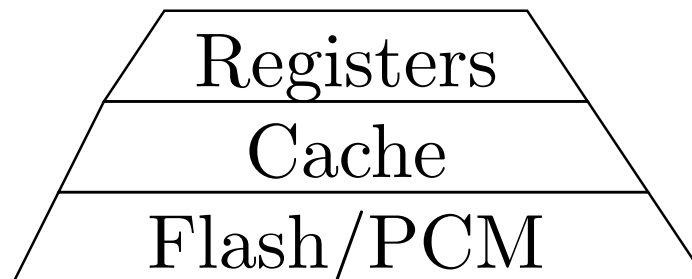
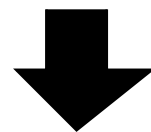
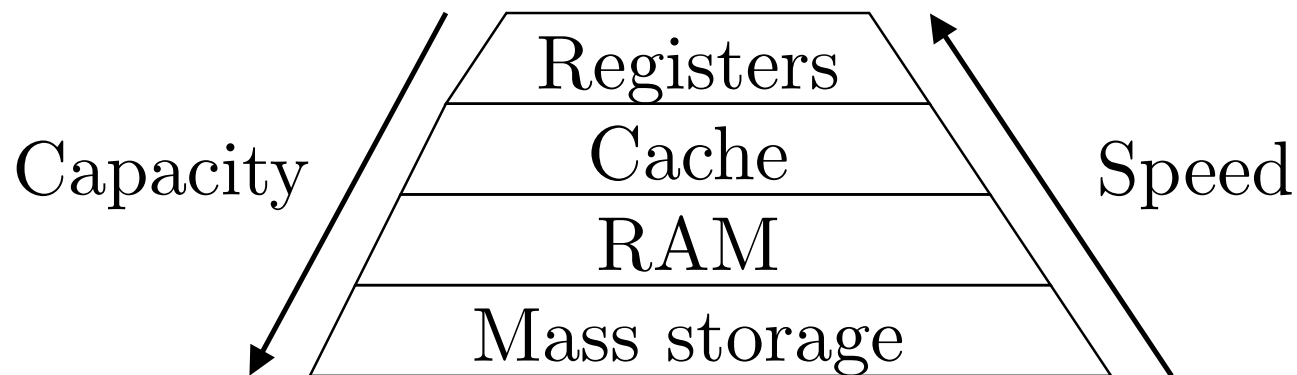
Problems. . .

- ▶ **Confidentiality:** prevents the adversary to *understand* the off-chip data. (solution:encryption)
- ▶ **Integrity:** prevents the adversary to *forge/tamper* with the off-chip data. (solution:?)
- ▶ **Secure erasure:** prevent the adversary to recover some data erased at time $t - i, i > 1$. (solution:?)

Problems and challenges

Challenges. . .

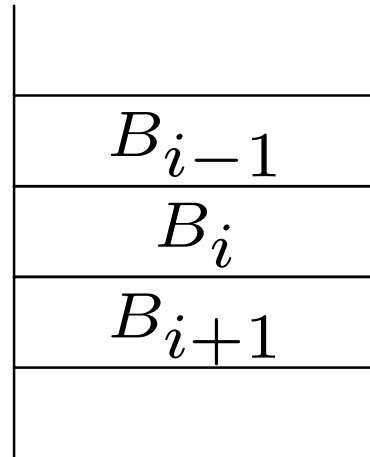
- ▶ Memory technologies are evolving quickly. . .



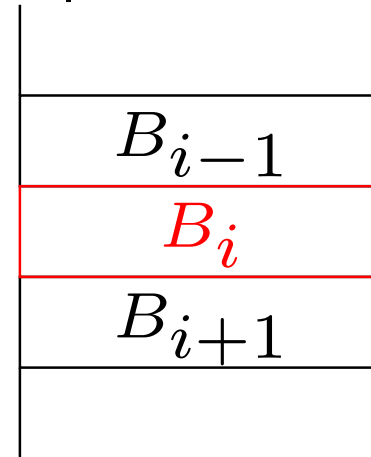
Integrity

Bloc injection attack

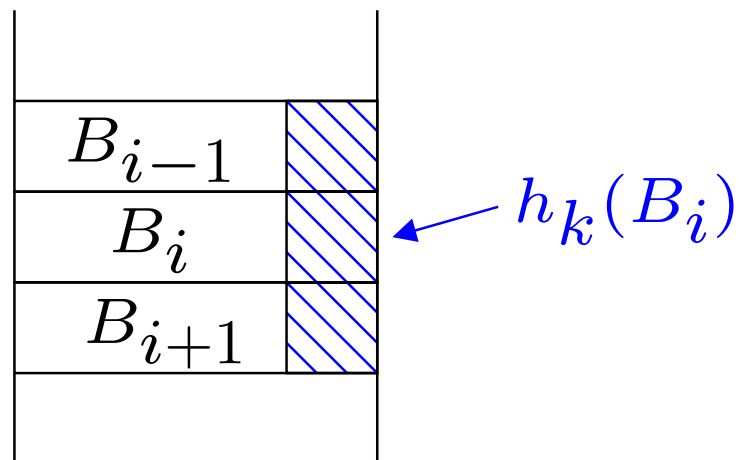
Correct memory



Corruption of Bloc i

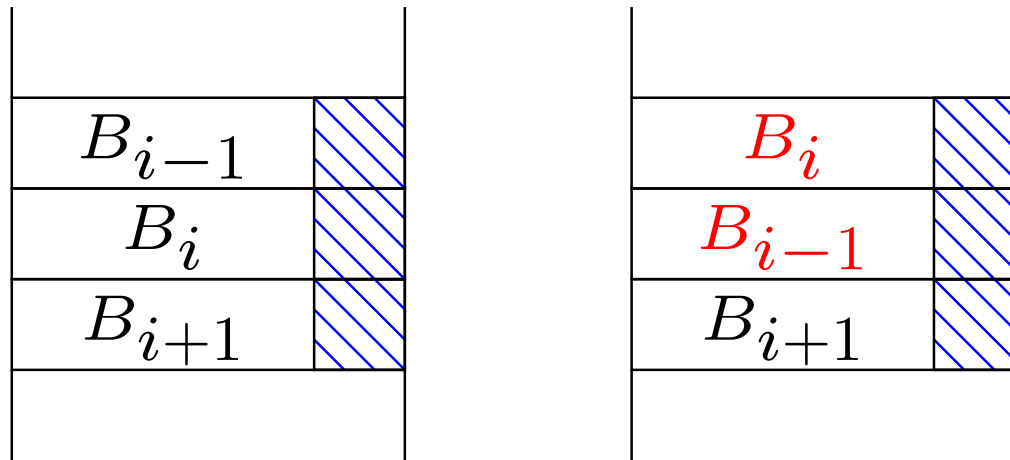


Solution: MAC

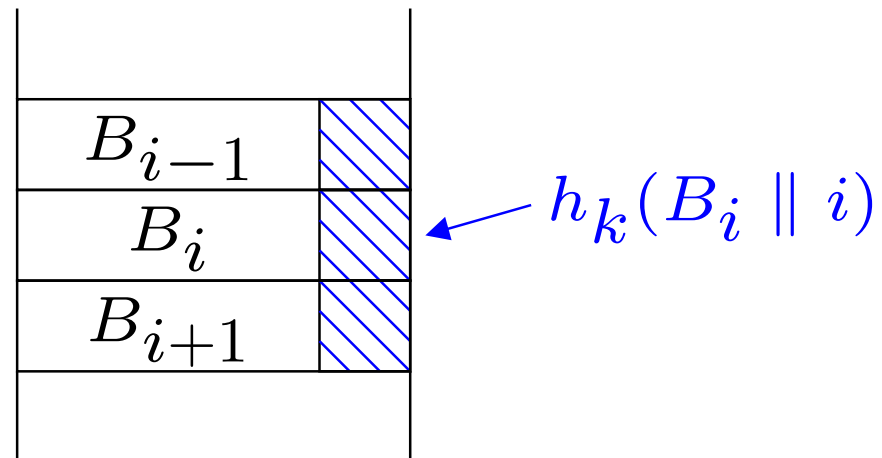


Integrity

Bloc movement attack

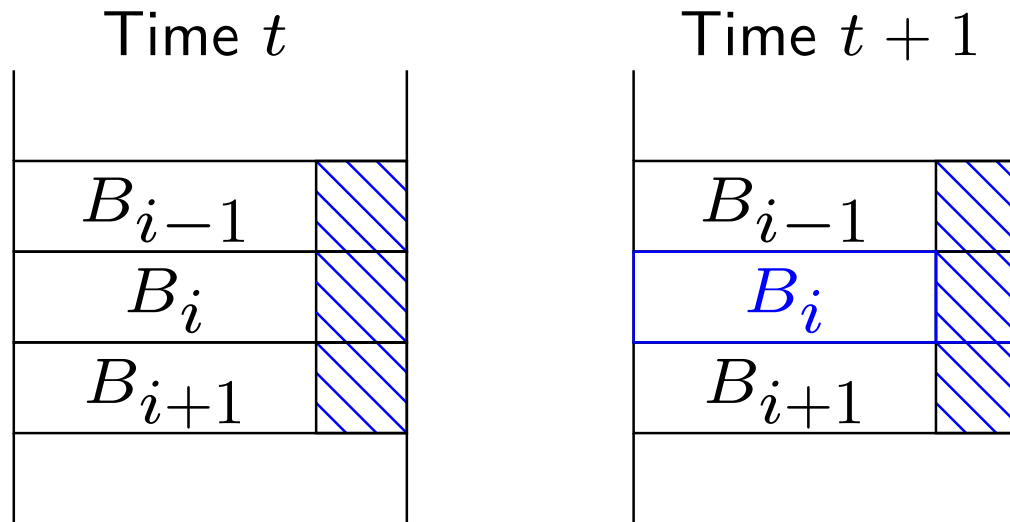


Solution

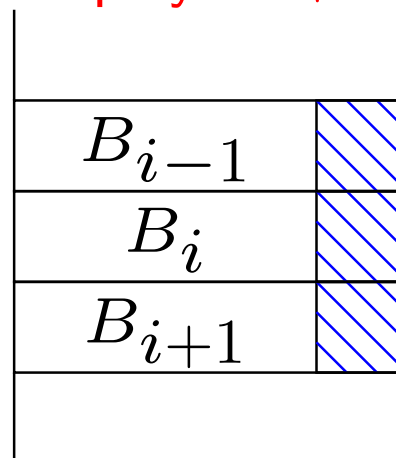


Integrity

Bloc replay attack



Replay: $t + 2$



Metrics

- ▶ **NCM**: Θ N-Chip Memory
- ▶ **FFCM**: Θ FF-Chip Memory (size n)
- ▶ **RoR**: number of memory accesses for reading a bloc
- ▶ **RoW**: number of memory accesses for writing a bloc

Integrity

Hash: a naive solution

- ▶ We compute:

$$H = f(h_k(B_1 \parallel 1) \parallel \cdots \parallel h_k(B_i \parallel i) \parallel \cdots \parallel h_k(B_n \parallel n)),$$

with f a cryptographic hash function (SHA-3).

- ▶ Complexity:

- **NCM:** $O(1)$ (not $O(n)$ because of Merkle-Damgard)
- **FFCM:** None
- **RoR:** $O(n)$
- **RoW:** $O(n)$

Integrity

Counter

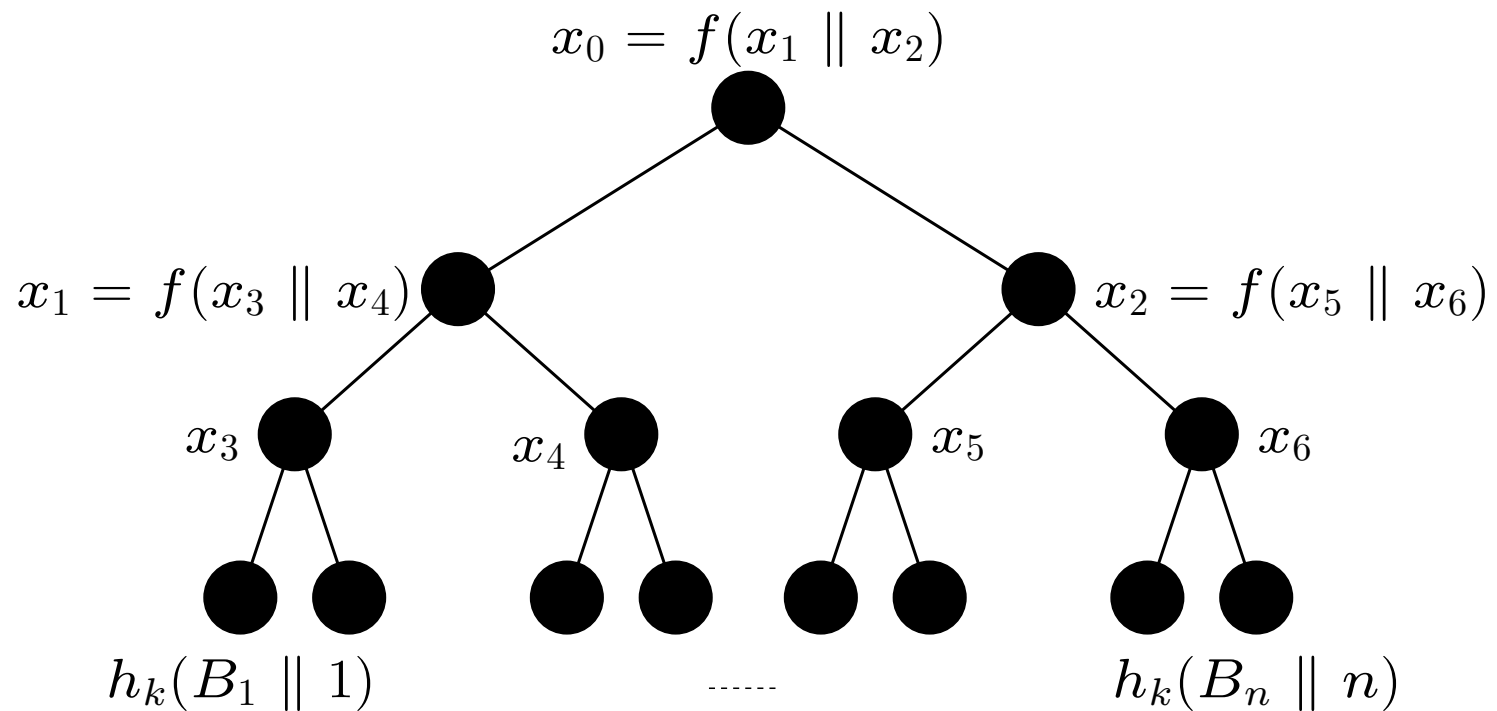
- ▶ We store a counter c_i for each block in the off-chip memory, and we modify the MAC:

$$h_k(B_1 \parallel 1 \parallel c_1)$$

- ▶ Complexity:
 - **NCM:** $O(n)$
 - **FFCM:** None
 - **RoR:** $O(1)$
 - **RoW:** $O(1)$

Integrity

Merkle's tree



$$x_3 = f(h_k(B_1 \parallel 1) \parallel h_k(B_2 \parallel 2))$$

Integrity

Merkle's tree

- ▶ The root x_0 and k are stored on-chip. All the other values are stored off-chip.
- ▶ To verify B_1 , read B'_2 and compute $x'_3 \dots$
At the end compare x'_0 to the stored value x_0 .
- ▶ Complexity:
 - **NCM**: $O(1)$
 - **FFCM**: $O(n)$
 - **RoR**: $O(\log n)$
 - **RoW**: $O(\log n)$

Integrity

Conclusion

	BB-Tree	Hash	Inc-Hash	Counter	Merkle	Bloom
NCM	$O(n)$	$O(1)$	$O(1)$	$O(n)$	$O(1)$	$O(n)$
FFCM	None	None	None	None	$O(n)$	None
RoR	$O(\log n)$	$O(n)$	$O(n)$	$O(1)$	$O(\log n)$	$O(1)^\dagger$
RoW	$O(\log n)$	$O(n)$	$O(1)$	$O(1)$	$O(\log n)^\ddagger$	$O(1)$

\dagger but $O(n)$ for reading the internal memory.

\ddagger also $O(\log n)$ write off-chip.

\S Need a counting filter otherwise $O(n)$.

Secure deletion

of SSD/PCM memories with wear-leveling

Problem

How to remove any versions of a data from the off-chip memory ?

- ▶ **Trivial** with classical memory
- ▶ **Difficult** with memory using wear-leveling
 - Flash;
 - Phase-Change Memory (PCM).

Flash and PCM memories

- ▶ Reading: block level (4 Kb).
- ▶ Erasure: page level only (256 Kb).
- ▶ Memory wear:
 - 10^6 program-erase cycles (expensive)
 - 10^3 program-erase cycles (cheapest and future)
- ▶ **Need to improve the memory endurance!**
 - garbage collector;
 - wear-leveling.

Flash and PCM memories

- ▶ The **locality** principle must **die!**

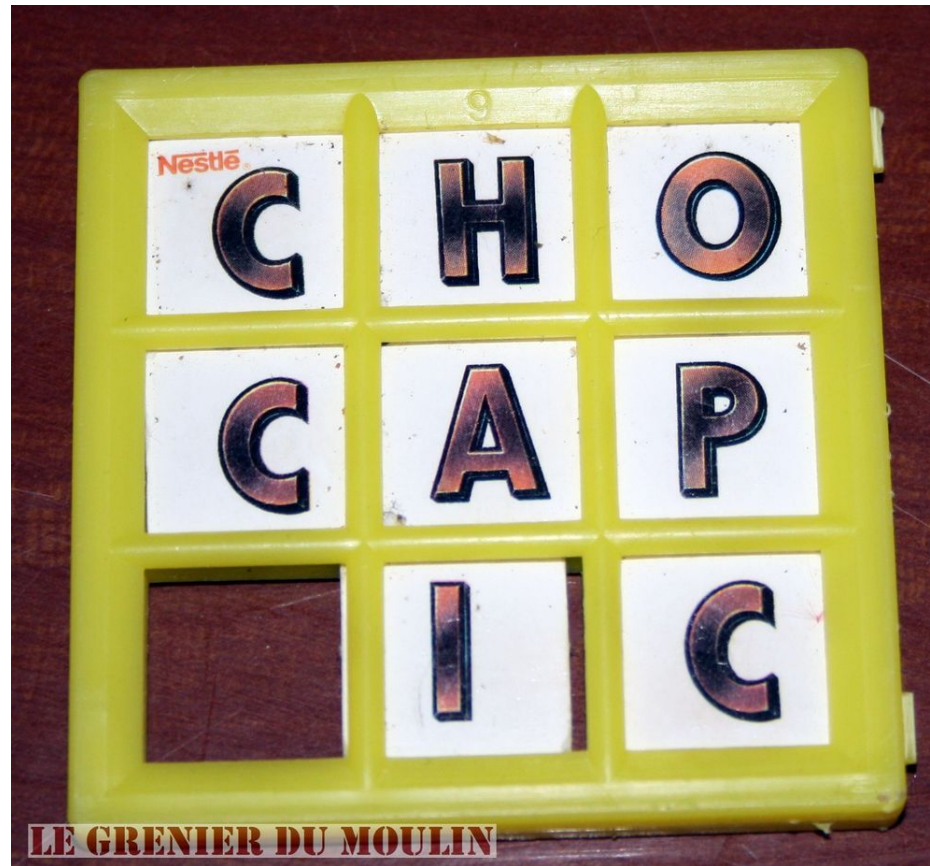
Wear-leveling

*Writes on Flash and PCM memory must be **heavenly distributed** over all the pages.*

- ▶ Can be done at:
 - OS level (YAFFS),
 - Driver, Firmware. . .
 - Memory-controller.
- ▶ The probability of writing a data twice at the same physical location is **negligible!**

Wear-leveling

Mystic Square



- ▶ Combinatorics: permutation, codes. . . (Rank modulation)

Kill 'em all [Metallica 1983]

Track 9: Seek & Destroy (Hetfield, Ulrich)



▶ . . . but in $\Theta(n \log n)$!

Conclusion

▶ Technological issue or long term research ?

▶ Is it meaningful to start to write:

Algorithmic on Strange Memory?

and/or

Security for Strange Memory?

▶ Difficult for industrials to answer. . . better to anticipate
their C__ P

References

Integrity

- ▶ R. Merkle. *A certified digital signature*. CRYPTO 1989, 218–238.
- ▶ M. Blum, W. S. Evans, P. Gemmell, S. Kannan, M. Naor. *Checking the Correctness of Memories*. FOCS 1991, 90-99.
- ▶ M. Bellare, O. Goldreich, S. Goldwasser. *Incremental Cryptography: The Case of Hashing and Signing*. CRYPTO 1994, 216-233.
- ▶ R. Elbaz, D. Champagne, R. Lee, L. Torres, G. Sassatelli, P. Guillemain. *TEC-Tree: A Low-Cost, Parallelizable Tree for Efficient Defense Against Memory Replay Attacks*. CHES 2007, 289-302.

References

Deletion

- ▶ P. Gutmann. *Secure Deletion of Data from Magnetic and Solid-State Memory*. USENIX Security 1996.
- ▶ M. Wei , L. Grupp, F. Spada , S. Swanson. *Reliably Erasing Data From Flash-Based Solid State Drives* . USENIX FAST 2011.
- ▶ J. Reardon, C. Marforio, S. Capkun, D. Basin. *Secure Deletion on Log-structured File Systems* . arXiv 2011.