ASSURE: Authentication Scheme for SecURE Energy Efficient Non-Volatile Memories

Joydeep Rakshit Kartik Mohanram
Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh

Abstract

Data tampering attacks threaten data integrity in emerging non-volatile memories (NVMs). Whereas Merkle Tree (MT) memory authentication is effective in thwarting data tampering attacks, it drastically increases cell writes and memory accesses, adversely impacting NVM energy, lifetime, and system performance. We propose ASSURE, a low overhead, high performance Authentication Scheme for SecURE energy efficient (ASSURE) NVMs. ASSURE synergistically integrates (i) smart message authentication codes (SMACs), which eliminate redundant cell writes by enabling MAC computation of only modified words on memory writes, with (ii) multi-root MTs (MMTs), which reduce MT reads/writes by constructing high performance static MMTs (SMMTs) or low overhead dynamic MMTs (DMMTs) over frequently accessed memory regions.

1 Introduction and Motivation

Resistance-class non-volatile memories (NVMs) such as phase-change memory (PCM) and resistive RAM (RRAM) [2, 3] are low power, dense (e.g., multi-/triple-level cell), and scalable DRAM replacement technologies. Whereas data persistence is a desirable property of NVMs, it poses security vulnerabilities that affect data privacy [4, 5], motivating NVM encryption. However, NVM encryption does not guarantee data integrity against active attacks, which must be addressed to realize a tamper-proof, secure NVM system.

Active attacks: An active attack on NVM entails an adversary *altering* the data stored in or being fetched from main memory. Active attacks can be further categorized into spoofing, splicing, and replay (occasionally referred to as rollback) attacks [6, 7]. In spoofing, the adversary replaces an existing valid memory block with fake data. In splicing, the attacker swaps the memory content between two locations. Finally, in replay, the content of a memory location is reverted back to an older value.

NVM authentication: It is widely accepted that active attacks can be thwarted by memory authentication [6–8]. State-of-the-art memory authentication solutions (including authentication supported in recent commercial solutions like Intel SGX [9]) maintain a logical data structure, Merkle Tree (MT), whose nodes are obtained by recursive computation of hash message authentication codes (HMACs) over memory blocks. An HMAC constitutes a cryptographic hash of the input data (i.e. memory block). In an MT, each parent node HMAC ensures integrity of all the child node HMACs [6–8]. MT memory authentication ensures the integrity of read (written) data by verifying (updating) its HMAC lineage upto the MT root. The secret HMAC key and the MT root is maintained on the secure processor, rendering valid HMAC generation for the spoofed/spliced/replayed data impossible.

NVM authentication overhead: HMACs demonstrate strong diffusion property, resulting in a high cell write rate. Hence, these HMACs incur significant NVM energy and lifetime overhead. Further, additional memory reads/writes are introduced for MT read/update on each memory access; these reads/writes stall critical data reads or writes to the same memory bank, degrading system IPC. We use simulations of SPEC CPU2006 workloads to show that deploying MT authentication over an encrypted TLC RRAM increases the cell writes (NVM energy) to 5.8× (5.3×) and degrades IPC to 0.65× in comparison to a nominal encrypted TLC RRAM. Although NVM authentication is indispensable for data integrity, it degrades NVM energy, lifetime, and system IPC, motivating development of low penalty NVM authentication solutions.

This research was supported by NSF Award CCF-1217738. Paper URL: https://goo.gl/9Zhi2v.
Citation: [1]. Authors' emails: joydeep.rakshit@pitt.edu, kartik.mohanram@gmail.com

2 ASSURE

ASSURE is a low overhead NVM authentication solution that deploys (i) smart MACs to decrease cell writes for HMAC computation and (ii) dynamic multi-root MTs to reduce memory accesses for MT authentication, without compromising the security of classical MT authentication. The detailed discussion of ASSURE architecture (SMAC and MMT) can be found in [1].

2.1 Smart Message Authentication Codes

Smart message authentication codes (SMACs) perform selective HMAC recomputation of the encrypted data by leveraging the observation that the majority of the words in a cache line remain unmodified during a memory write-back; this observation is also utilized in state-of-the-art NVM encryption schemes like DEUCE [4] and SECRET [5]. DEUCE (as well as SECRET) partitions the cache line into words of equal width, and re-encrypts only the modified words during a memory write. SMAC partitions the HMAC at word-level granularity and recomputes HMAC words corresponding to the re-encrypted data words; this eliminates cell writes due to the redundant HMAC computation of unmodified words.

Figure 1 illustrates SMAC design. To achieve selective HMAC computation of only the modified words, SMAC splits the original encrypted cache line into two decoupled intermediate messages (IMs) corresponding to the modified and unmodified words. The first (second) IM, IM_1 (IM_2) is constructed from the modified (unmodified) words, with the unmodified (modified) words zeroed out. IM_1 (IM_2) is then provided as input to a keyed cryptographic hash function, generating the intermediate HMACs, IH_1 (IH_2). Similar to IMs, the IHs are also partitioned at word-level granularity. The final HMAC (FH) is constructed with IH_1 (IH_2) words for the corresponding modified (unmodified) word positions.

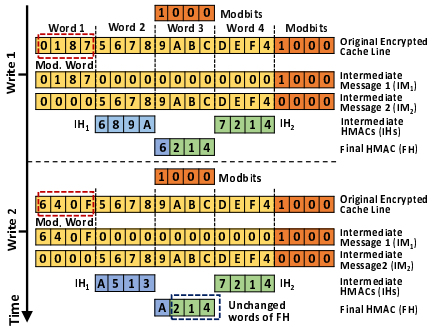

Figure 1: **SMACs eliminate redundant cell writes for unmodified words (2, 3, and 4) during HMAC computation. Between writes 1 (W_1) and 2 (W_2), word 1 gets modified in the original encrypted cache line, altering the intermediate HMAC 1 (IH_1), resulting in modification of word 1 in the final HMAC (FH); however, IH_2 is unaltered due to unmodified words 2, 3, and 4, eliminating redundant cell writes for unmodified words 2, 3, and 4 of the FH.**

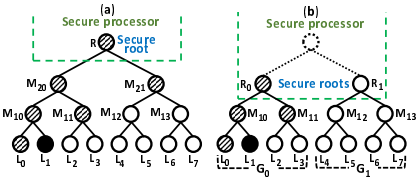

Figure 2: **(a) Classical single-root MT, with the traversal path for verification of leaf L_1 highlighted. (b) An SMMT with 2 memory block groups covered by 2 smaller MTs with 2 independent MT roots on the secure processor. In SMMT, the traversal path for L_1 verification has 1 less MT level, due to the smaller individual MTs.**

2.2 Multi-root Merkle Trees

Multi-root MTs (MMTs) maintain multiple smaller MTs having fewer levels (with multiple roots on the secure processor) as a novel alternative to the classical single-root MT. We discuss two MMT variants: (i) static MMT (SMMT) and (ii) dynamic MMT (DMMT).

Static MMT (SMMT): In SMMTs, we leverage the observation that a smaller MT that spans fewer leaf nodes, i.e., memory blocks is composed of fewer MT levels, reducing the reads (writes) to verify (update) a corresponding MT branch during a leaf read (write). SMMT partitions the memory into memory block groups (MBGs), assigning an MT to each MBG, and maintaining the corresponding MT roots in an MT-root RAM on the secure processor. The individual MTs spanning each MBG are smaller than a single-root MT spanning the entire memory, substantially reducing the MT reads/updates, thereby decreasing NVM energy and enhancing system IPC. Figure 2 illustrates the observation and design of SMAC. Whereas the advantages of SMMTs over classical single-root MTs scale logarithmically with the number of MBGs, it comes at the expense of linearly scaling processor MT-root RAM.

Dynamic MMT (DMMT): Dynamic MMTs (DMMTs) provide the NVM energy and system IPC improvements comparable to SMMTs without the significant overhead of processor MT-root RAM. DMMT leverages the spatial and temporal locality of memory accesses exhibited by practical workloads to maintain a smaller hot MT over the hot MBG, achieving SMMT-like reduction in the MT read/writes for authentication of a majority of the memory accesses. Since the remaining MBGs (cold MBGs) experience fewer memory accesses, the DMMT maintains a larger MT (cold MT henceforth) spanning the cold MBGs requiring only one root; therefore, the DMMT stores only two secure roots, independent of the number of MBGs. Figure 3 illustrates DMMT, which requires effective hot MBG prediction to capture the majority of memory accesses. DMMT tracks the memory access count of each MBG over a period of P_{PRED} accesses, and designates the MBG that accounts for the maximum accesses as the hot MBG for the next P_{PRED} accesses.

3 Results and Conclusions

We evaluate NVM energy, lifetime, and system IPC of ASSURE (both SMMT and DMMT) on a TLC RRAM architecture for comparison against state-of-the-art Bonsai Merkle Tree (BMT) [7], using integer and floating-point workloads from the SPEC CPU2006 [10] benchmark suite. For NVM energy and lifetime evaluations, we perform trace-based simulations using NVMain [11] and an in-house page-level lifetime simulator, respectively. For system IPC evaluations of ASSURE, we use MARSS [12]. Table 1 summarizes the NVM energy, memory lifetime, and IPC results of SMMT ASSURE and DMMT ASSURE (normalized to the baseline, i.e., BMT).

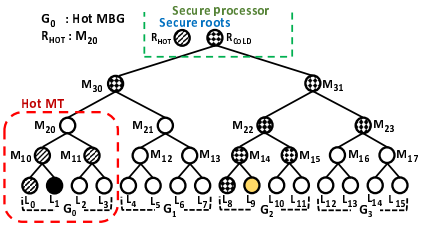

Figure 3: **DMMT organization with the hot MBG spanned by the hot MT, and a cold MT covering all remaining MBGs collectively, with their corresponding roots on the secure processor (R_{HOT} and R_{COLD}). Leaf L_1 (black) in the hot MBG requires lower MT level traversals than L_9 (yellow) in the cold MBG.**

Authentication Technique	NVM Energy	Memory Lifetime	System IPC
Baseline (BMT)	1	1	1
SMMT ASSURE	0.41	2.36	1.11
DMMT ASSURE	0.45	2.11	1.10

Table 1: **Summary of the NVM energy, memory lifetime, and IPC results of baseline (BMT), SMMT ASSURE, and DMMT ASSURE (normalized to the baseline).**

NVM energy and lifetime: SMMT ASSURE (DMMT ASSURE) reduces NVM energy, on average, by 59% (55%) over BMT authentication. SMMT ASSURE (DMMT ASSURE) extends the memory lifetime, on average, by 2.36× (2.11×) over baseline BMT. ASSURE leverages the dual advantages of SMACs and MMTs; SMACs significantly reduce cell writes for data HMACs and each MT node while MMTs decrease the number of MT node reads/writes.

System performance: SMMT ASSURE (DMMT ASSURE) improves system IPC, on average, by 11% (10%) over the baseline (BMT). ASSURE implements MMTs that diminish the number of MT node reads/writes, thereby reducing bank contention between critical data reads (writes) and MT node reads (writes). SMACs enable multiple power-constrained concurrent writes in one write slot by reducing the effective number of cell updates per write, thereby diminishing the effective latency of MT updates.

Conclusions: Memory authentication increases NVM energy, degrades lifetime, and deteriorates system IPC. ASSURE is the first work to address low cost NVM authentication. ASSURE integrates smart MACs (SMACs) and multi-root MTs (MMTs) to realize tamper-evident NVMs with low energy and improved lifetime as well as IPC. Whereas SMACs eliminate redundant HMAC computations of unmodified words on write-backs, MMTs maintain multiple smaller MTs that collectively span the memory, reducing MT reads/writes for authentication. ASSURE outperforms state-of-the-art BMT authentication with 55% lower NVM energy, 2.11× improved lifetime, and 10% better system IPC on average.

References

[1] J. Rakshit and K. Mohanram, "ASSURE: Authentication Scheme for SecURE energy efficient non-volatile memories," in *Proc. Design Automation Conference*, 2017.
[2] B. C. Lee *et al.*, "Architecting phase change memory as a scalable DRAM alternative," in *Proc. Intl. Symposium on Computer Architecture*, 2009.
[3] C. Xu *et al.*, "Understanding the trade-offs in multi-level cell ReRAM memory design," in *Proc. Design Automation Conference*, 2013.
[4] V. Young *et al.*, "DEUCE: Write-efficient encryption for non-volatile memories," in *Proc. Intl. Conference on Architectural Support for Programming Languages and Operating Systems*, 2015.
[5] S. Swami *et al.*, "SECRET: Smartly EnCRypted energy efficient non-volatile memories," in *Proc. Design Automation Conference*, 2016.
[6] G. E. Suh *et al.*, "Efficient memory integrity verification and encryption for secure processors," in *Proc. Intl. Symposium on Microarchitecture*, 2003.
[7] B. Rogers *et al.*, "Using address independent seed encryption and bonsai merkle trees to make secure processors OS- and performance-friendly," in *Proc. Intl. Symposium on Microarchitecture*, 2007.
[8] A. D. Hilton *et al.*, "PoisonIvy: Safe speculation for secure memory," in *Proc. Intl. Symposium on Microarchitecture*, 2016.
[9] S. Gueron, "A memory encryption engine suitable for general purpose processors.," *IACR Cryptology ePrint Archive*, vol. 2016, p. 204, 2016.
[10] J. L. Henning, "SPEC CPU2006 benchmark descriptions," in *ACM SIGARCH Computer Architecture News*, 2006.
[11] M. Poremba and Y. Xie, "NVMain: An architectural-level main memory simulator for emerging non-volatile memories," in *Proc. Computer Society Annual Symposium on VLSI*, 2012.
[12] A. Patel *et al.*, "MARSS: a full system simulator for multicore x86 CPUs," in *Proc. Design Automation Conference*, 2011.