Ensuring Fast Crash Recovery for Secure NVMs

Kazi Abu Zubair
University of Central Florida
kzubair@knights.ucf.edu

Amro Awad
University of Central Florida
amro.awad@ucf.edu

I. Introduction

Emerging NVM technologies (e.g., Phase Change Memory) are closing the gap between fast and volatile DRAM and slow but non-volatile storage technologies (e.g. Flash drive, SSDs). The near-zero idle power, scalability, and other desirable features of NVMs are driving the vendors to introduce NVM based main memories to the market (e.g. 3DXPoint [1]). Adopting such technology in the memory hierarchy requires careful extension or modification in the memory management architecture while taking persistency, crash consistency, wear leveling, reliability, and security into account [2]–[4]. While the non-volatile characteristics of the NVM memory cells facilitate persistent applications and fast accessible filesystem (e.g., DAX filesystem), they make the system more vulnerable to adversarial attacks. For instance, data remanence in NVM cells invite a wide range of attacks, demanding the encrypted presence of the stored data. Apart from encryption, secure architectures also aim to provides protection against data tampering and replay attacks [4].

State-of-the-art NVM security models make use of the Counter Mode Encryption for confidentiality and the Merkle Tree for integrity protection [2]. Although the encryption counters and tree nodes are stored in the NVM memory, they are cached in fast volatile cache inside the processor chip. While caching security metadata provides outstanding performance improvements and reduces NVM writes, it makes the system crash inconsistent as it loses the updates made in the volatile cache if a system crash happens. Osiris [5], a state-of-the-art, solves the crash consistency problem by re-purposing the ECC to recover the counters. However, it takes a long time to recover as the entire memory (can be terabytes in emerging NVM) needs to be scanned for possible corrections. Moreover, the SGX-style counter tree structure does not allow for recovery even with the cost of high recovery time. Anubis [6][1], addressing the problem, tries to reduce the recovery time for all integrity tree structures from hours to sub-seconds in a performance-friendly way.

II. Motivation

Having fast recoverability is as important as having security in a system. In the case of Bonsai Merkle Tree, Osiris [5] can recover the security metadata at the cost of extremely high recovery time. For instance, for a system with 8TB memory, Osiris takes more than 7 hours to recover both encryption counters and Merkle-tree. According to the 5 nine's rule of reliability, a high availability server needs to be functional 99.999% of its lifetime. Being unavailable for hours for recovery easily disqualifies the system as highly available. Table 1 compares existing methods in terms of recoverability and performance, and presents the aim of Anubis, which is achieving recoverability, fast recovery, and high performance across all security implementations and all Merkle-tree types.

TABLE I: Recoverability and recovery time for different schemes.

	BMT	**SGX Tree**	**Performance Overheads**
Traditional NVM Security	Not Recoverable	Not Recoverable	Low
Osiris	Recoverable; High Recovery Time	Not Recoverable	Low
Strict Persistence	Recoverable; Fast Recovery	Recoverable; Fast Recovery	Very High
Anubis	Recoverable; Fast Recovery	Recoverable; Fast Recovery	Low

III. Anubis

A. Tracking Security Metadata Updates

The core design philosophy of Anubis is that it is sufficient to persistently track the modifications in the security metadata done in the volatile cache. To achieve that, Anubis maintains a shadow structure in the NVM memory which acts as the mirror of the cache. However, to minimize the updates in the shadow structure, Anubis adopts some smart optimizations that can effectively track the Counter Cache and Merkle-tree Cache without incurring high write overhead.

B. Anubis for General Integrity Tree (AGIT)

AGIT provides mechanisms to recover a secure NVM system having a Bonsai-style integrity tree. AGIT Read (Figure 1a) only tracks the encryption counter and tree node reads due to a miss in counter cache or Merkle-tree cache. On the other hand, AGIT Plus (Figure 1b) asks the question of "what metadata were modified in cache?" rather than asking "what metadata were brought in cache?". To do so, it only writes the address of the cacheline to the shadow table when an encryption counter or Merkle-tree node is modified first time in the cache. Later modification to the same metadata in cache will not trigger any tracking unless it is evicted and new metadata is placed in its position inside the cache. The pink bar in Figure 2 shows that AGIT read is performance friendly in most applications, but can incur high overhead in read-intensive applications. The green bar in Figure 2 shows that AGIT plus is highly

[1]This work has been published in ISCA'19, June 22-26, 2019, Phoenix, AZ, US

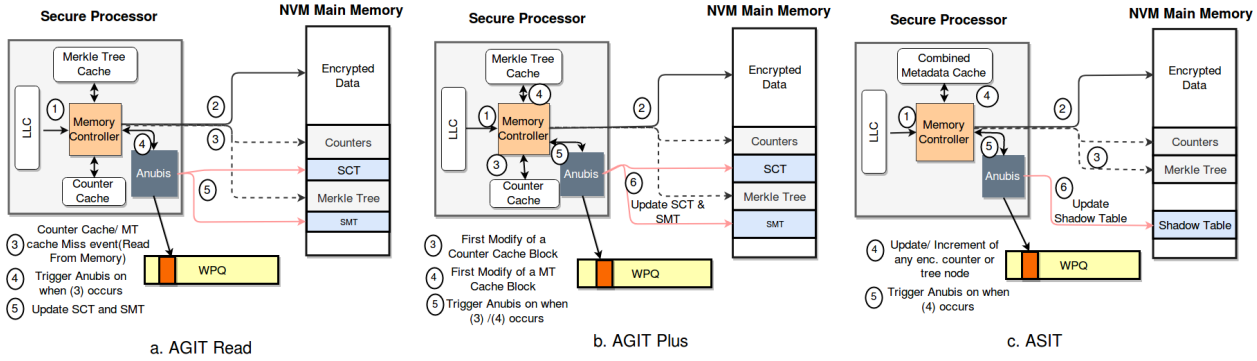

Fig. 1: Anubis Operation.

optimized and does not cause excessive performance overhead. Figure 4 shows the recovery time and cache size relationship.

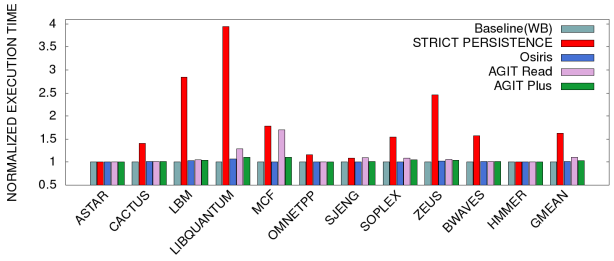

Fig. 2: AGIT Performance.

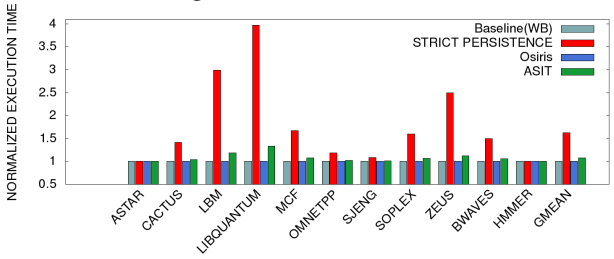

Fig. 3: ASIT Performance.

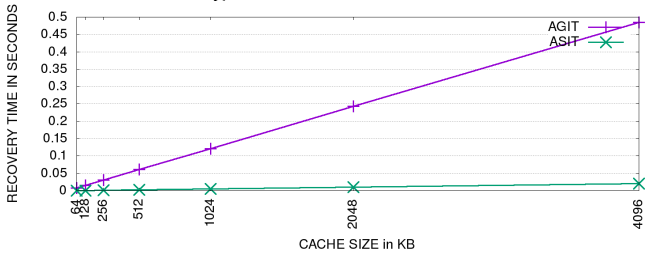

Fig. 4: Recovery Time.

C. Anubis for SGX Integrity Tree (ASIT)

ASIT (Anubis for SGX Integrity Tree) aims to provide recoverability and fast recovery for the SGX-style tree. Unlike BMT, which can be regenerated merely from the leaves, SGX tree recovery is highly complicated. To facilitate SGX tree recovery, ASIT stores the 49-bit LSBs of the tree-counter and a MAC value calculated over the counter in the shadow entry. Additionally, ASIT exploits Lazy tree update to reduce the overhead of having a recoverable tree and proposes to have an additional small tree to protect the integrity of the shadow table to enable secure recovery. Figure 1c shows the ASIT operation. Figure 3 shows that ASIT (green bar) effectively reduces the overhead of only other recoverable scheme Strict Persistence (red bar) significantly. While the overhead is slightly more than AGIT, some woks (Phoenix [7]) demonstrate that it is possible to further reduce overheads in SGX-tree.

IV. Conclusion and Potential Impact

Anubis allows secure NVM systems to recover quickly from a system crash. It reduces the recovery time significantly (less than a second), while incurring only 3.4% overhead in systems with BMT tree, and only 7.9% in complicated SGX-like tree of counters. Additionally, *Anubis* removes the memory size barrier in recovery time and makes the recovery time just a function of the metadata cache size. Such a fast recovery mechanism allows future server systems to seamlessly integrate NVM memory in highly reliable secure environment.

Acknowledgements

Part of this research was developed with funding from the Defense Advanced Research Projects Agency (DARPA) under contract number N66001-19-C-4017. The views, opinions and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. Approved for public release. Distribution is unlimited.

References

[1] F. T. Hady, A. Foong, B. Veal, and D. Williams, "Platform storage performance with 3d xpoint technology," *Proceedings of the IEEE*, vol. 105, no. 9, pp. 1822–1833, 2017.

[2] A. Awad, M. Ye, Y. Solihin, L. Njilla, and K. Abu Zubair, "Triad-nvm: Persistency for integrity-protected and encrypted non-volatile memories," in *Proceedings of the 46th International Symposium on Computer Architecture (ISCA)*, 2019.

[3] A. Awad, P. Manadhata, S. Haber, Y. Solihin, and W. Horne, "Silent shredder: Zero-cost shredding for secure non-volatile main memory controllers," in *ACM SIGARCH Computer Architecture News*, vol. 44, pp. 263–276, ACM, 2016.

[4] C. Yan, D. Englender, M. Prvulovic, B. Rogers, and Y. Solihin, "Improving cost, performance, and security of memory encryption and authentication," in *ACM SIGARCH Computer Architecture News*, vol. 34, pp. 179–190, IEEE Computer Society, 2006.

[5] M. Ye, C. Hughes, and A. Awad, "Osiris: A low-cost mechanism to enable restoration of secure non-volatile memories," in *51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2018)*, 2018.

[6] K. A. Zubair and A. Awad, "Anubis: ultra-low overhead and recovery time for secure non-volatile memories," in *Proceedings of the 46th International Symposium on Computer Architecture*, pp. 157–168, ACM, 2019.

[7] M. Alwadi, A. Mohaisen, and A. Awad, "Phoenix: Towards persistently secure, recoverable, and nvm friendly tree of counters," *arXiv preprint arXiv:1911.01922*, 2019.