Improving the Performance and Endurance of Encrypted Non-volatile Main Memory through Deduplicating Writes [*]

Pengfei Zuo, Yu Hua, Ming Zhao[†], Wen Zhou, Yuncheng Guo
Huazhong University of Science and Technology, China
[†]*Arizona State University, USA*

1 Motivation

Non-volatile memory (NVM) technologies are considered as promising candidates of the next-generation main memory. NVM has high scalability and low energy consumption but suffers from the problems of low endurance and security.

First, NVM is vulnerable to physical access based attacks, including stolen DIMM and bus snooping attacks [1, 3, 5], due to still retaining data after systems are powered down. For example, an attacker stealing the NVM DIMM or acting as a machine repairman can directly stream out all the data from the DIMM. Hence, memory encryption becomes important to ensure the data security in NVM. Second, NVMs typically have limited write endurance, e.g., $10^7 - 10^8$ writes for PCM. Writes on NVM also cause higher latency (i.e., $3-8\times$) and energy overhead than reads. The bit-level write reduction techniques, such as Data Comparison Write (DCW) [4] and Flip-N-Write (FNW) [2], are able to significantly reduce the number of bits written to NVM, based on the observation that only a small number of bits are modified for a write.

However, memory encryption renders existing *bit-level* write reduction techniques ineffective for encrypted non-volatile main memory (NVMM), due to the diffusion property of encrpytion. Because of this property, the change of a single bit in the plaintext leads to the change of about half of the bits in the ciphertext. Hence, existing techniques like DCW and FNW cannot achieve significant data reduction for encrypted NVMM.

There is, however, abundant data duplication at the *line level* which can be exploited to reduce the number of writes to secure NVMM. Specifically, we observe that a large number of cache lines written to the memory are identical to existing ones in the memory in many real-world applications. These observations motivate us to perform cache-line-level deduplication on secure NVMM. Eliminating a large number of duplicate writes not only extends the endurance of NVM-M, but also significantly improves the system performance. First, eliminating duplicate writes efficiently removes the high write latency off from the critical path of application execution. In the meantime, eliminating duplicate writes also speeds up read and non-duplicate write requests by reducing their waiting time. That is because, when a write request is served by an NVM bank, the following read/write requests to the same bank are blocked and wait until the current write request is completed. If many duplicate write requests are avoided, the waiting time of the following read/write requests is significantly reduced, thus improving the read/write performance.

[*]Original Paper: Pengfei Zuo, Yu Hua, Ming Zhao, Wen Zhou, Yuncheng Guo, "Improving the Performance and Endurance of Encrypted Non-volatile Main Memory through Deduplicating Writes", in *Proceedings of the 51st IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2018, pages: 442 – 454. `https://csyhua.github.io/csyhua/hua-MICRO2018.pdf`

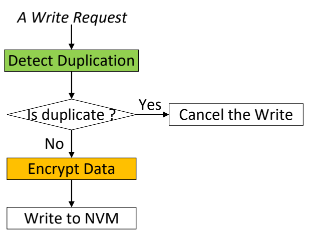
2 Our Approach

We propose DeWrite, a novel solution for enhancing both the lifetime and performance of secure NVMM with new in-line deduplication technique, i.e., write deduplication (§ 2.1), by leveraging the intrinsic read/write asymmetry of NVM and light-weight hashing. Moreover, DeWrite also opportunistically parallelizes the operations of deduplication and encryption (§ 2.2) and allows them to co-locate the metadata (§ 2.3) for high time and space efficiency.

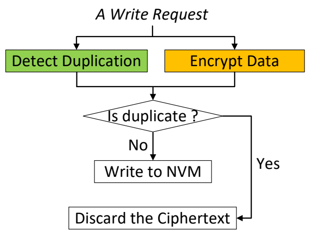
2.1 Write Deduplication

To explore how many cache lines written to main memory are duplicate, we have examined 20 applications from SPEC CPU2006 and PARSEC 2.1 benchmark suites. We observe that the percentage of duplicate lines vary from 18.6% to 98.4% across the 20 applications. Eliminating these duplicate lines via line-level deduplication would result in much high write reductions. Hence, in DeWrite, we study the use of deduplication for enhancing the endurance and performance of secure NVMM.

To perform in-line deduplication, DeWrite leverages the asymmetric property of NVMs, in which the write latency is much higher than read latency (i.e., $3-8\times$). To detect duplicate cache lines, DeWrite computes a light-weight hash to summarize the contents of cache lines, rather than the cryptographic hash with high computation latency. If



(a) The direct way



(b) The parallel way

Figure 1: Integrating deduplication and encryption.

the hash of the cache line matches that of an existing line in NVMM, DeWrite reads the line and compares the corresponding data byte by byte. Thus DeWrite eliminates a duplicate write at the cost of a read, thus improving system performance due to the asymmetry of reads and writes.

2.2 Prediction-based Parallelism

In the secure NVMM, the cache lines are first encrypted and then written to NVMM. DeWrite leverages deduplication to reduce the number of writes to the secure NVMM by eliminating the writes of duplicate cache lines. The direct way is to perform duplication detection and data encryption serially, as shown in Figure 1a. This way reduces write latency for duplicate lines due to canceling their writes, but increases write latency for non-duplicate lines, since duplication detection and data encryption are executed serially in the critical path of the memory write. To avoid the serialization of detecting duplication and encrypting data, we consider a parallel way that carries out cache line encryption and duplication detection in parallel, as shown in Figure 1b. This way reduces write latency for non-duplicate lines, but causes extra computation and energy overhead from the AES circuit for duplicate lines. In summary, the direct way is efficient for duplicate cache lines but inefficient for non-duplicate cache lines. The parallel way has the opposite effects. Intuitively, the best solution is to use the direct way for duplicate cache lines and the parallel way for non-duplicate cache lines, respectively.

In order to identify whether a cache line is duplicate beforehand, we propose a simple yet effective prediction scheme by exploiting the duplication states of the most recent memory writes recorded by using a history window. Specifically, DeWrite maintains a history window for the whole main memory. The history window records the duplication states of the most recent cache lines written into main memory. If a cache line is predicted to be non-duplicate, DeWrite encrypts data in parallel with duplication detection to reduce write latency. Otherwise, DeWrite detects duplication without encrypting data to reduce computation overhead and energy consumption from encryption. Our evaluation demonstrates that over 93% of average prediction accuracy is achieved.

2.3 Metadata Colocation

Existing counter mode encryption needs to store per-line counters to encrypt/decrypt data. The per-line counter is 28 bits for each line [3]. Deduplication also produces some metadata. To reduce the space overhead of metadata, we propose a co-located metadata storage scheme between deduplication and encryption, via embedding the per-line counters into the data structures for deduplication.

There are two kinds of memory lines in the NVMM, i.e., deduplicated lines and non-deduplicated lines. For the deduplicated lines, the real addresses containing their data contents are stored in the address mapping table. But their corresponding locations in the inverted hash table are null, since the deduplicated lines do not contain valid data. For the non-deduplicated lines, their storage locations in the address mapping table are null due to no deduplication, and their hash values are stored in the inverted hash table. Therefore, we observe that for each memory lines in the NVMM, either its real address location or its hash location is null. Based on the above observation, we embed the counter of each memory line into the null location either in the address mapping table or in the inverted hash table, thus reducing the space overhead of counter storage.

3 Experimental Results

DeWrite is implemented via revising the metadata store and metadata cache that already exist in secure NVMM and only adding a dedup logic into the memory controller, thus achieving the low design complexity. We have comprehensively evaluated the performance of DeWrite in gem5 with NVMain by using SPEC CPU2006 and PARSEC benchmark suites. Our experimental results demonstrate that DeWrite eliminates 54% of writes to secure NVMM, and speeds up memory writes and reads by 4.2× and 3.1× on average. Meanwhile, DeWrite improves the IPC by 82% and reduces 40% of energy consumption, while incurring only 6.25% metadata storage overhead.

References

[1] Awad, A., Manadhata, P., Haber, S., Solihin, Y., and Horne, W. Silent Shredder: Zero-cost shredding for secure non-volatile main memory controllers. In *Proc. of ASPLOS* (2016).

[2] Cho, S., and Lee, H. Flip-N-Write: a simple deterministic technique to improve PRAM write performance, energy and endurance. In *Proc. of MICRO* (2009).

[3] Young, V., Nair, P. J., and Qureshi, M. K. DEUCE: Write-efficient encryption for non-volatile memories. In *Proc. of ASPLOS* (2015).

[4] Zhou, P., Zhao, B., Yang, J., and Zhang, Y. A durable and energy efficient main memory using phase change memory technology. In *Proc. of ISCA* (2009).

[5] Zuo, P., and Hua, Y. SecPM: a secure and persistent memory system for non-volatile memory. In *HotStorage* (2018).