TASecure: Temperature-Aware Secure Deletion Scheme for Solid State Drives

Bingzhe Li

University of Minnesota, Twin Cities
Minneapolis, MN, USA
lix1743@umn.edu

David H.C. Du

University of Minnesota, Twin Cities
Minneapolis, MN, USA
du@umn.edu

ABSTRACT

With the increasing concerns of security, the secure deletion for SSDs becomes very costly due to its out-of-place update (i.e., an update is performed in a new location leaving the old data un-touched). Some previous studies used a combined erase-based and cryptography-based method to find a heuristic deletion scheme. However, the deletion overhead is still large since a key may be associated with too many pages. Therefore, how to reduce the secure deletion overhead further is becoming an interesting research issue. In this paper, a temperature-aware secure deletion scheme (TASecure) is proposed. Data with the same temperature are stored in the same region in order to reduce the overhead of secure deletion. The temperature of a data is referred to the frequency of the data is updated. Moreover, the regions based on their temperature are associated with different numbers of keys and are applied with different secure deletion schemes. Finally, the experimental results indicate that our proposed secure deletion scheme can reduce the secure deletion cost about 1.2x to 19x compared to previous studies.

CCS CONCEPTS

• **Security and privacy** → *Database and storage security*; • **Computer systems organization** → *Architectures*;

KEYWORDS

Solid state drive; Secure deletion; Invalid data; Temperature

ACM Reference Format:

Bingzhe Li and David H.C. Du. 2019. TASecure: Temperature-Aware Secure Deletion Scheme for Solid State Drives. In *Great Lakes Symposium on VLSI 2019 (GLSVLSI '19), May 9–11, 2019, Tysons Corner, VA, USA*. ACM, New York, NY, USA, 4 pages. <https://doi.org/10.1145/3299874.3318016>

1 INTRODUCTION

With the rapid development of the electrical devices and Internet, various types of information are stored as digital

in storage media. How to keep data privacy and security has becoming more and more important. One important security issue is how to securely delete invalid data in the storage media [1]. Due to the high performance of solid state drives (SSDs), SSDs currently have been becoming prevailing across many computing environments from the mobile devices to the cloud systems. People expect the SSDs having the same security features as the traditional hard disks. However, due to the different granularity of write and erase operations, SSDs use out-of-place updates with the log-structure [2]. As a result, the updated data will not directly overwrite its original value and need to be written to free space (new location). The out-of-place updates keep the invalid data still in the SSDs until they are garbage collected. This provides an opportunity for adversaries or attackers to recover these invalid but probably sensitive data.

To overcome the secure issue of invalid data in SSDs, two basic methods were proposed. One is called erase-based deletion. The idea is to erase all invalid data for secure deletion. The other method is called cryptography-based scheme [3], which first encrypts data before writing them to the flash. Meanwhile, the keys used in the encryption are also stored in the flash. However, the above methods confront large overhead of migration when securely deleting invalid data. To reduce the migration overhead, some other works combined these two methods to lower erase and migration overhead. For example, ErasuCrypto [4] used either erase-based or cryptography-based method to delete blocks. The greedy heuristic algorithm is used to determine whether a block uses the erase-based or the cryptography-based deletion. However, above works did not consider the effect of the workload access patterns. According to the access patterns, the efficient I/O allocation in SSDs may help significantly reduce the migration and erase overhead during secure deletion.

In this paper, a new Temperature Aware Secure deletion scheme called TASecure is proposed to reduce overhead of the secure deletion. First, according to the historical accumulated update frequency (called temperature), the pages/data are classified to several regions and the data in the same region have similar update frequency (the same temperature). After that, chunks in the same region contain the same number of blocks and the pages/data across different blocks in the same chunk are encrypted by the same key. As a result, the TASecure scheme is capable of lowering the overhead of secure deletion with dynamic data allocation and different chunk configurations for different real workloads. Finally, the experimental results indicate that our proposed secure

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

GLSVLSI '19, May 9–11, 2019, Tysons Corner, VA, USA

© 2019 Association for Computing Machinery.

ACM ISBN 978-1-4503-6252-8/19/05...\$15.00

<https://doi.org/10.1145/3299874.3318016>

deletion scheme can out-perform existing schemes in terms of the secure deletion cost.

2 BACKGROUND AND MOTIVATION

Due to the out-of-place update as discussed above, the invalid data are kept in the SSD after updates and before erased. The purpose of secure deletion is to destroy invalid data permanently in order that adversaries cannot recover those invalid data. Two basic schemes, an erase-based and a cryptography-based, were proposed as discussed in the introduction section. For the erase-based scheme, during deletion process the blocks containing invalid pages will be erased. Meanwhile, all valid pages in the block must be migrated to free pages. The cryptography-based scheme encrypts data with security keys. The encrypted data and keys are stored in the flash memory. To securely delete the invalid data pages, the SSD only needs to delete the keys of the invalid data pages. The encryption algorithm normally uses the advanced encryption standard (AES) [5]. To save space, pages across multiple blocks can share the same key. These multiple blocks with shared keys comprise one **chunk**. The number of blocks in the chunk is called **chunk size** (C_z). Thus, if there are valid data pages sharing a key with invalid data pages, the valid pages must be migrated to free space before deleting the key. Therefore, the above two deletion schemes face large overhead of erase, decryption, migration, and re-encryption.

To overcome the high overhead, Liu *et al.* [4] proposed a secure deletion scheme called ErasuCrypto, which combines both erase-based and cryptography-based schemes together. The basic idea is that in one chunk ErasuCrypto uses a heuristic algorithm to find the minimum secure deletion cost (as seen in Eq. (1)) by applying either cryptography-based or erase-based scheme to each block.

$$cost = \#Mgr + k \times \#Era \quad (1)$$

where, $\#Mgr$ and $\#Era$ indicate the number of migrations and the number of erases separately for secure deletion. k is the coefficient to indicate the ratio of the cost between erase and migration operations.

Some other works [4][6][7][8] also focused on reducing the overhead of secure deletion. In [6], they partitioned the secure and insecure data in order to get rid of unnecessary migrations. However, the scenario of insecure and secure is not common. People always want to protect the data on the whole drive. Another technique called 'scrubbing' [9][7][10][11] is a page-level secure deletion scheme, which generally reprograms invalid pages in order to make those valid pages un-accessible. However, the scrubbing method may introduce unpredictable disturbs to the valid pages within the same blocks. Wang *et al.* [8] proposed a new encryption and secure deletion for mobile devices. The mobile host generates the keys for storage devices and saves the key block space. However, the drawback of this scheme is that the storage device is permanently attached to the host and cannot be used by other systems.

Overall, previous works [4][6][7][8] passively address the secure deletion issue without any intent to optimize data

Algorithm 1 TASecure Data Classification and Allocation

```

1: procedure DATA CLASSIFICATION AND ALLOCATION
2:   while  $t < T$  do // Collecting access frequency in  $T$ 
3:     Compute slice number  $S_i$  of  $Req_t$ 
4:      $Acc\_TBL[S_i] += 1$ 
5:     Compute max and min values in  $Acc\_TBL$ 
6:     Compute  $N$  centroids for  $N$  regions,  $centroid_j = \max - \min * j/N$ 
7:     New request coming ( $Req$ )
8:     Compute the slice number ( $S$ ) of  $Req$ 
9:     Compute Euclidean distance  $dS, centroid_j$ .
10:    if  $dS, centroid_j$  is the minimum for all  $j$  from 0 to  $N-1$  then
11:       $Region\#j \leftarrow Req$ 
```

Algorithm 2 TASecure Deletion

```

1: procedure SECURE DELETION
2:   if  $Region\#0$  then
3:     Erase-based deletion for the  $region\#0$ 
4:     ...
5:   else
6:     Do greedy heuristic algorithm [4]
```

scheduling and management. Therefore, there is an opportunity to further reduce the overhead of secure deletion by actively scheduling and managing incoming data.

3 TASECURE SCHEME

In this section, we mainly introduce the temperature-aware secure (TASecure) scheme. The basic idea is to separate potentially valid and invalid data in order to reduce the secure deletion overhead. There are two major steps in the TASecure scheme. The first step is to classify requests according to their historical access patterns and to assign them to the regions with the same temperature. The second step is to apply proper deletion schemes to different temperature regions. In Section 4, the configuration of the SSDs are provided with the constant key block overhead.

For the first step of the TASecure scheme, it is the data classification and allocation as seen in Algorithm 1. First, the SSD logical space is split into **slices**, which are consecutive logical spaces (the default size of a slice is 20MB), to record the historical access patterns. We do not use the page-level recording since it needs much larger space to store the historical information. During a period T , the number of writes are recorded in **Acc_TBL** as seen at Lines 2-4 of Algorithm 1. Then, the centroid values are computed, which are representative values for each region/classification category (Lines 5-6). For example, suppose that there are two regions ($N=2$, N indicates the number of regions), the maximum and minimum values in the Acc_TBL are the two representative values for these two groups. Once obtaining the centroids, a new coming request in the next period can be classified based on those centroids. The euclidean distances are compute between the slice number (S) of the request and each centroid value. Then, the request should be stored into the region of which centroid and its S value achieve the minimum euclidean distance. By

Figure 1: An example of the ErasuCrypto scheme for secure deletion with the chunk size of 4. Secure deletion causes 4 migrations and 3 erases.

Figure 2: An example of the TASecure scheme for secure deletion chunk size = 4. Secure deletion causes no migration and 3 erases.

doing that, data will be stored in the region which contains the same temperature data.

The second step is the TASecure deletion process as shown in Algorithm 2. As discussed above, data have been partitioned into different regions based on their temperature. In the hottest region (access frequencies of requests are high), the blocks contain the largest staleness value (number of invalid pages/total block size). Thus, we can only use the erase-based scheme with little migration overhead for the hottest region as seen at Lines 2-3 of Algorithm 2 and no encryption is needed in this region. There are two advantages. The first one is that without encryption and decryption processes data access latency will be reduced. The other is that no key block is needed and thus it saves space. For the other regions, they use the heuristic algorithm [4] to determine which deletion scheme is used for their blocks (Lines 5-6).

Figure 1 and Figure 2 are two examples to indicate the overhead of ErasuCrypto and TASecure schemes. Two schemes have the same key block overhead (data blocks : key blocks = 4:1). As seen in Figure 1, for the ErasuCrypto scheme the hot and cold data are mixed in the blocks. Since the hot data have the higher probabilities to be updated, most of hot data become invalid. Therefore, during the secure deletion process, ErasuCrypto scheme erases Block0 and Block2 due to its high staleness (erase-based) and deletes the keys of block1 and block3 (cryptology-based). Compared to ErasuCrypto, as seen in Figure 2, the proposed TASecure groups the data with the same temperature in the same blocks. As

Table 1: SSD configuration

Parameter	Value	Parameter	Value
SSD capacity	250GB	# of pages/block	128
Page Size	4KB	Read latency	25us [14]
Slice size	20MB	Erase latency	1.5ms [14]
# of key blocks	250	Program Latency	200us [14]

Table 2: MSR Cambridge trace configuration [12]

	Duration (hrs)	Number of IOs (Millions)		Total request size (GB)	
		Write	Read	Write	Read
prn_0	168	4.98	0.602	45.96	13.12
prn_1	168	2.77	8.46	30.78	181.35
proj_1	168	2.50	21.14	25.58	750.36
usr_1	168	3.86	41.43	56.13	2079.23
usr_2	168	1.99	8.58	26.47	415.28
hm_0	168	2.58	1.42	20.47	9.96
rsrch_0	168	1.30	0.13	10.82	1.39
stg_1	168	0.80	1.40	5.99	79.52
prxy_0	168	12.14	0.38	53.80	3.05
ts_0	194	1.49	0.32	11.34	4.13

a result, the pages with updates are grouped in the same block. Therefore, the TASecure scheme only needs 3 block erases without any migration. Moreover, after secure deletion, TASecure contains less number of invalid and keyless pages than ErasuCrypto which might cause large garbage collection overhead in future.

4 EXPERIMENTAL RESULTS

In this section, the performance of different secure deletion schemes are compared based on the simulation results.

4.1 Environment Setup

The SSD configuration is listed in Table 1. The SSD capacity is 250GB. The migration and erase latencies are 225us and 1.5ms, respectively. Thus, we set k value in Eq. (1) to 7. The number of key blocks is set to 250. It means that the chunk size is 8 for the ErasuCrypto scheme. The encryption algorithm uses the AES algorithm with 128 bits (AES-128). The real trace Cambridge MSR traces [12] is used. The experimental results are obtained based on the DiskSim simulator [13] with the SSD model extension. We assume that the secure deletion command is triggered after replaying each 168-hour trace.

4.2 Performance Comparison

In this subsection, the deletion performance comparison between TASecure, ErasuCrypto, Erase-based scheme and cryptography-based scheme is provided. Three metrics, the secure deletion cost in Eq. (1), number of migrations and number of erases, are used to indicate the secure deletion overhead. The chunks size is set to 8. It means that 8 pages are encrypted by the same key. Therefore, the total number of key blocks for TASecure, ErasuCrypto and Cryptography-based scheme is 250.

Figure 3, Figure 4 and Figure 5 indicate the normalized secure deletion cost, number of migrations and number of erases between TASecure, ErasuCrypto, Erase-based scheme and cryptography-based scheme, respectively. For the secure

Figure 3: Normalized secure deletion cost comparison between TASecure, ErasuCrypto, erase-based scheme and cryptography-based schemes.

Figure 4: Number of migrations between TASecure, ErasuCrypto, erase-based scheme and cryptography-based schemes. (y-axis is log-scale)

Figure 5: Number of erases between TASecure, ErasuCrypto, erase-based scheme and cryptography-based schemes. (y-axis is log-scale)

deletion cost, the cost of ErasuCrypto is normalized to one. As seen in Figure 3, the erase-based scheme obtains the worse secure deletion cost and the cryptography-based scheme is the second worse secure deletion scheme. The proposed TASecure achieves the lowest secure deletion cost. Compared to the erase-based scheme, TASecure achieves about 2.7x - 19.8x cost reduction. Compared to the cryptography-based scheme, TASecure obtains 1.3x - 12.7x secure deletion cost reduction. Also, TASecure gets 1.2x to 9.4x less cost than ErasuCrypto. In summary, the proposed TASecure obtains on average about 9.5x, 4.8x and 4.1x cost reduction compared to erase-based, cryptography-based and ErasuCrypto schemes, respectively.

The details of the secure deletion cost are provided based on the number of page migrations and block erases in Eq. (1). For

the migrations as seen in Figure 4, TASecure tremendously reduces the migration overhead compared to the other three schemes. This is because TASecure distinguishes the hot/cold data and separates them in the different regions. As a result, grouping hot data together causes less migration overhead.

5 CONCLUSION

In this paper, the temperature-aware secure deletion scheme is proposed. According to the temperature of the data pages in SSDs, the pages with the same or similar temperature are stored in the same block in order to reducing the migration overhead during deletion period. Meanwhile, both the erase-based and cryptography-based secure deletion methods are used for the blocks with different temperatures. Finally, the experimental results indicate that our proposed secure deletion scheme can reduce the secure deletion cost about 20% to 19x when compared with previous studies.

6 ACKNOWLEDGMENTS

This work was partially supported by NSF I/UCRC Center Research in Intelligent Storage and the following NSF awards 1421913, 1439622, 1525617, and 1812537.

REFERENCES

- [1] Cyril Guyot, et al. Implementing secure erase for solid state drives, August 21 2012. US Patent 8,250,380.
- [2] Peter Desnoyers. Analytic modeling of ssd write performance. In *Proceedings of the 5th Annual International Systems and Storage Conference*, page 12. ACM, 2012.
- [3] Joel Reardon, et al. Data node encrypted file system: Efficient secure deletion for flash memory. In *Proceedings of the 21st USENIX conference on Security symposium*, pages 17–17. USENIX Association, 2012.
- [4] Chen Liu, et al. Erasurecrypto: A light-weight secure data deletion scheme for solid state drives. *Proceedings on Privacy Enhancing Technologies*, 2017(1):132–148, 2017.
- [5] Royal Canadian Mounted Police. G2-003 hard drive secure information removal and destruction standards, 2003.
- [6] Hoda Aghaei Khouzani, et al. Architecting data placement in ssds for efficient secure deletion implementation. In *Proceedings of the International Conference on Computer-Aided Design*, page 42. ACM, 2018.
- [7] Wei-Chen Wang, et al. Scrubbing-aware secure deletion for 3-d nand flash. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(11):2790–2801, 2018.
- [8] Minshen Wang, et al. A novel data secure deletion scheme for mobile devices. In *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–8. IEEE, 2018.
- [9] Ping-Hsien Lin, et al. Achieving fast sanitization with zero live data copy for mlc flash memory. In *Proceedings of the International Conference on Computer-Aided Design*, page 41. ACM, 2018.
- [10] Michael Yung Chung Wei, et al. Reliably erasing data from flash-based solid state drives. In *Fast*, volume 11, pages 8–8, 2011.
- [11] Kyoungmoon Sun, et al. Models and design of an adaptive hybrid scheme for secure deletion of data in consumer electronics. *IEEE Transactions on Consumer Electronics*, 54(1), 2008.
- [12] Dushyanth Narayanan, et al. Write off-loading: Practical power management for enterprise storage. *ACM Transactions on Storage (TOS)*, 4(3):10, 2008.
- [13] Greg Ganger, et al. The disksim simulation environment (v4. 0). *Parallel Data Lab*, <http://www.pdl.cmu.edu/DiskSim/Online-document>, 2009.
- [14] Ping Huang, et al. Shiftflash: Make flash-based storage more resilient and robust. *Performance Evaluation*, 68(11):1193–1206, 2011.