Hide-and-Seek: Hiding Secrets in Threshold Voltage Distributions of NAND Flash Memory Cells

Md Raquibuzzaman
The University of Alabama in Huntsville
mr0068@uah.edu

Aleksandar Milenkovic
The University of Alabama in Huntsville
milenka@uah.edu

Biswajit Ray
The University of Alabama in Huntsville
biswajit.ray@uah.edu

ABSTRACT

In this paper, we propose a new page-writing technique to hide secret information using the threshold voltage variation of programmed memory cells. We demonstrate the proposed technique on the state-of-the-art commercial 3D NAND flash memory chips by utilizing common user mode commands. We explore the design space metrics of interest for data hiding: bit accuracy of public and secret data and detectability of holding secret data. The proposed method ensures more than 97% accuracy of recovered secret data, with negligible accuracy loss in the public data. Our analysis shows that the proposed technique introduces negligible distortions in the threshold voltage distributions. These distortions are lower than the inherent threshold voltage variations of program states. As a result, the proposed method provides a hiding technique that is undetectable, even by a powerful adversary with low-level access to the memory chips.

CCS CONCEPTS

• **Hardware → Non-volatile memory**; • **Security and privacy → Hardware security implementation**.

KEYWORDS

Data hiding, 3D NAND, Threshold voltage variation

ACM Reference Format:
Md Raquibuzzaman, Aleksandar Milenkovic, and Biswajit Ray. 2023. Hide-and-Seek: Hiding Secrets in Threshold Voltage Distributions of NAND Flash Memory Cells. In *Proceedings of 15th ACM Workshop on Hot Topics in Storage and File Systems (HotStorage'23).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3599691.3603415

HotStorage '23, July 9, 2023, Boston, MA, USA

ACM 979-8-4007-0224-2/23/07…$15.00
https://doi.org/10.1145/3599691.3603415

1 INTRODUCTION

Steganography is a well-established technique for embedding secret information into digital objects such as images by intentionally adding small distortions [4, 19]. Steganography has a distinct advantage compared to encryption-based data protection as it hides the very presence of the secret data from the adversary. Encryption techniques can defend against a passive adversary trying to steal data storage devices' secrets. However, it cannot defend against an active adversary, who can find ways to coerce the device owner into disclosing the decryption key.

Plausibly deniable data storage solution in the solid-state drive has recently gained significant traction due to the pervasive usage of solid-state storage media (e.g., in mobile devices). Several solutions for achieving plausible deniability in the solid state drives have been proposed recently, such as INFUSE [18], PEARL [5], DEFTL [10], MDEFTL [11]. Similarly, several steganographic file system solutions have been proposed [1, 8, 14, 18]. However, most existing solutions incorporate deniability in the file system layer or the flash translation layer. In contrast, the proposed work incorporates the plausible deniability in the physical properties of the NAND storage media, which is immune to software-based deniability compromises.

Wang et al. [22] investigated data hiding in flash memory, proposing a covert channel using inherent variations of program times in memory cells to hide data. By manipulating the physical properties of selected cells, program times can be varied subtly to hide data. The process is slow and reduces device lifetime due to repeated program-erase cycling.

Zuck et al. [24] propose a new technique called "Stash in a Flash" to address the limitations of Wang et al.'s method. This method hides data in flash memory by manipulating the threshold voltage of randomly selected cells in the erased state. The threshold voltage variations of hidden and public data are indistinguishable, resulting in improved hiding and recovery throughput. However, the technique relies on special flash memory operations that are not commonly available to the end users. In addition, the erased memory state typically suffers from read disturb, program disturb, and cell

interference effects, making their technique vulnerable to NAND reliability issues [2, 3, 6, 7, 16, 21, 23].

This paper proposes a new method of steganographic storage in NAND flash media using the threshold voltage variation of programmed memory cells. Unlike previous methods, the proposed method offers more design space variables to control the bit accuracy of secret/public data with increased tolerance against memory disturbances. The approach requires no special memory operations and works with MLC (2-bits/cell), TLC (3-bits/cell), and QLC (4-bits/cell) flash memory configurations. Experimental evaluations show negligible distortions in threshold voltage distributions that are undetectable by an adversary. The proposed method achieves over 97% accuracy of recovered secret data with minimal impact on co-existing public data. Design trade-offs are also explored.

The rest of the paper is organized as follows. Section 2 covers the fundamental structure and functionalities of flash memory. Section 3 presents the threat model and provides a system overview of the proposed technique. Section 4 details the suggested approach for writing/programming and reading secret data. Section 5 delves into the experimental setup and evaluation results. Finally, Section 6 concludes the paper.

2 BACKGROUND

NAND flash memory chips are composed of dies containing planes, blocks, and pages [15]. A memory cell keeps information in the form of the charge stored on its floating-gate (FG) or charge-trap (CT) layer. In SLC memory (1-bit/cell), a flash cell is in the programmed state (logic 0) when electrons are stored on the FG/CT layer, whereas it is in the erased state (logic 1) when there are no electrons on the FG/CT layer. The read reference voltage is set between the erased and programmed states to identify the cells' states accurately.

The erase operation is performed at a block-level granularity, whereas the read and program operations are performed at a page level. The page program operation in a NAND array is carried out using the incremental step pulse program scheme (ISPP) [17]. The storage system typically uses a firmware layer called the flash translation layer (FTL) to manage the flash array's special characteristics efficiently. FTL provides a block access interface to the host file system by mapping the logical addresses in a block layer to physical addresses in NAND flash. In addition, FTL contains modules that perform garbage collection and wear leveling.

3 THREAT MODEL AND SYSTEM VIEW OF THE SOLUTION

We assume that the adversary has physical access to the storage device and he/she is capable of performing low-level memory operations such as page read/write, block erase, and cell V_t distribution measurement of the storage media. We also assume that the adversary knows the storage encryption key and can retrieve all the content in the storage device. However, we assume the adversary does not know the secret key to select memory cells containing the hidden data. Hence the adversary will not be able to recover the secrets and will not be sure if any hidden secrets are there.

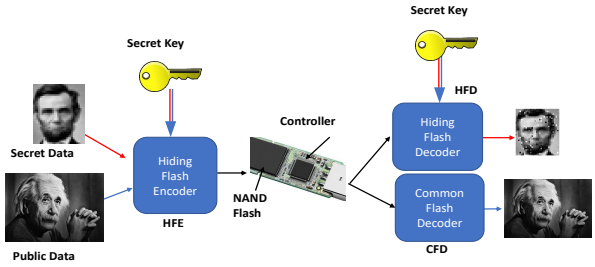

Figure 1: System overview of the proposed solution.

The system view of the proposed data hiding technique is described in Fig. 1. The Einstein image, representing the public data, is assumed to conceal the Lincoln image as the secret data. To encode the secrets, the memory bits of the public data are selected using a secret key. A hiding flash encoder (HFE) encodes the secret data. We assume the HFE can be implemented within the FTL of the storage device. If an adversary gains physical access to the storage device, they can only retrieve the Einstein image (public data) using the common flash decoder (CFD). The adversary would be unable to extract the secret data through separate probing of the NAND flash chip using its controller and conducting V_t analysis unless she or he possesses knowledge of the secret key. Furthermore, the adversary would be unable to ascertain whether any secret data has been encoded in the storage medium. The user with the secret key will be able to decode the secret message with a hiding flash decoder (HFD).

4 DATA HIDING AND READING TECHNIQUE

Fig. 2 illustrates our data-hiding technique. We use an SLC flash memory to simplify the discussion, though other configurations are possible (MLC, TLC). We start from a page in a memory block that was previously erased, and thus all cells are initially in the erased state (green dots). In our example, we assume we want to hide secret data, Lincoln's image, within the programmed (**0** bits) of public data, an Einstein's image. So, the size of the secret data needs to be lower than the number of **0** bits in the public image. For example, the

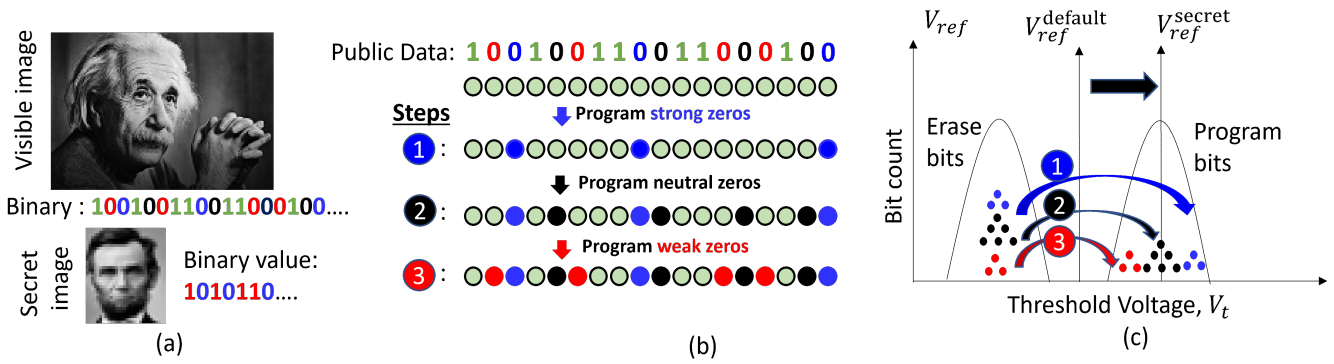

Figure 2: Illustration of data hiding method (a) Public and secret data. (b) Three-step programming scheme. (c) The reading method relies on distinguishing between strong and weak zeros by shifting the reference voltage.

size of the secret image in Fig. 1 is only 5.64% bits of the total number of 0s in the public image.

To hide Lincoln's image, we create subtle differences in the threshold voltages of the selected programmed flash memory cells. We encode strong 0s (colored in blue) that correspond to **0** bits of Lincoln's image and weak 0s that correspond to **1** bits of Lincoln's image (colored in red). The other programmed cells of Einstein's image have neutral zeros (colored in black), meaning they do not carry any secret data.

Flash memory inherently poses cell-to-cell V_t variation after a program operation. Several physical mechanisms, such as program noise, read noise, cell-to-cell process variation, and interference effects, all contribute to the V_t variation of the programmed cells. Thus, the proposed method hides secret data from the programmed memory cells' inherent V_t variation.

4.1 Hiding secret data

The process of hiding data involves three distinct steps (Fig. 2(b)), starting from an erased memory page shown as green cells on the top row. Based on the secret key, public data, and hidden data, we create the three binary contents strong zeros, weak zeroes, and neutral zeros.

Step 1. The first step is to write strong zeros. We exploit the neighbor wordline (WL) interference property of modern flash arrays [9], [15] to guarantee that the programmed cells will have threshold voltage in the upper portion of the V_t distribution for the programmed state. By writing into physically adjacent neighboring wordline (WL_{n+1}), the threshold voltage in the target wordline (WL_n) can be increased. Thus, step 1 involves regular page programming operation into the hiding page of the target wodline (WL_n) and a page from the neighboring wordline (WL_{n+1}), ensuring that programmed cells in the hiding page become strong 0s. The page in the neighboring wordline (WL_{n+1}) holds valid data eliminating the chance of leaving a clue for reverse steganography.

Step 2. This step involves programming cells that contain neutral zeros at the hiding page. We create a bit vector to program neutral zeros in the hiding page based on the public data and secret key. The bit vector is sent to the flash memory chip, and a regular page program operation is issued. In this step, most 0s from Einstein's image get programmed.

Step 3. The final step involves programming weak zeros. Whereas the previous two steps rely on regular page program operations, this step utilizes a partial program operation [20] that starts as a regular page program operation but gets terminated by prematurely issuing a RESET command. Partial program operations result in weak zero bits whose threshold voltage will reside in the lower tail of programmed V_t distribution, as shown in Fig. 2(c). The 1 bits in Lincoln's image are defined as these weak 0s.

4.2 Recovering secret data

The standard (or default) memory read operation cannot distinguish between the strong and weak zero bits; hence, the secret image remains invisible to the adversary. An elaborate memory read operation is needed to recover the secret image from the visible public image. Fig. 2(c) illustrates the secret image recovery scheme. The proposed reading method critically depends on the choice of the secret read reference voltage (V_{ref}^{secret}), which allows to distinguish between the strong and weak zero bits. We utilize the Read Offset features to distinguish between strong and weak zeros to recover the secret image [9]. A shared secret key can determine the bit position of the public image that holds the secret.

The overhead of the proposed hiding method involves two extra page program operations for implementing steps 2 and 3, as shown in Fig. 2(b). Overhead in the recovery process involves shifting the read reference voltage to V_{ref}^{secret} by read

offset command [13]. Except for the last wordline in a block, all other wordlines can be used for data hiding.

5 EXPERIMENTAL EVALUATION

5.1 Experimental setup

Fig. 3(a) shows our experimental setup consisting of a 132 BGA socket with a flash memory chip, an FT2232H mini module controller, and a workstation. A software package running on the workstation executes the ONFI commands and works as the FTL for the flash device. This hardware setup allows us to access raw memory bits without any error correction. We perform the experimental evaluation on several commercial-off-the-shelf (COTS) 64-layer 3D NAND TLC chips. The chips under test support so-called read offset operations that allow the flash controller to adjust read reference voltage, V_{ref} (Fig. 3(b)) by incrementally adding an offset of 7.5mV in each step. This way, we can extract the V_t distributions of all memory cell states [12].

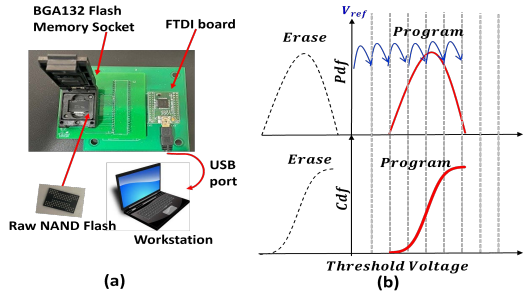

Figure 3: (a) Experimental set-up for interfacing raw NAND memory chips. (b) V_t distribution can be plotted with the probability density function (PDF) or cumulative distribution function (CDF).

5.2 Experimental evaluation of writing method

Fig. 4 shows experimental demonstrations of the three programming steps. After each programming step, the state of the Einstein image is shown by reading the memory content using the default read operation. Note that, after step 1, a small portion of the Einstein image is written corresponding to zero bits, coinciding with strong 0 bits of Lincoln's image. For simplicity, we chose the bit positions located on the left side of the Einstein image as the secret zeros. Hence, all the secret zeros are depicted in the left portion of the image after step 1 (colored in blue in Fig. 4). In practice, the location of the secret zeros can be randomly chosen from all possible locations of the public image. After step 2, most of the Einstein image is written. The zeros in this step do not carry

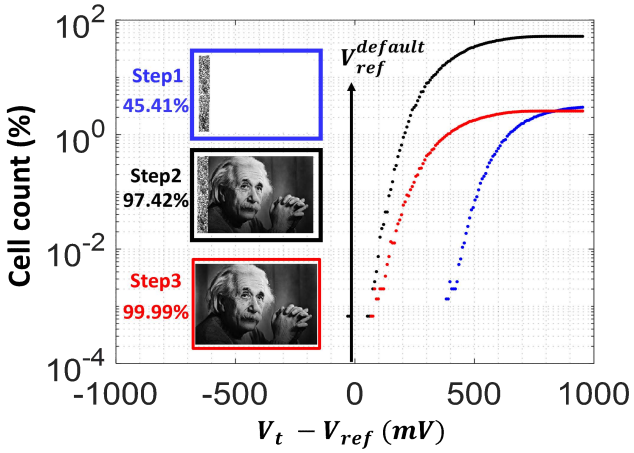

Figure 4: The evolution of the public image and corresponding V_t distribution of program bits after steps 1,2 and 3 are shown in blue, black, and red color, respectively.

any secrets. Finally, after step 3, the complete Einstein image is visible. This step's zeros are weak and represent secret '1' data. Partial programming is employed in step 3 to control the V_t values of the weak zero bits. The cell V_t distribution, measured after programming steps 1, 2, and 3, is shown in Fig. 4 using blue, black, and red colors, respectively. We use a cumulative distribution plot to show the measured cell V_t for the three sets of zero bits in the public data: (a) the strong zero bits (blue), which are secret zero bits (b) the neutral zero bits (black) which constitute the majority of the public zero bits and (c) the weak zero bits (red) which are secret ones. Note that there is a distinguishable difference between the V_t distribution of the strong and weak zero bits. We exploit this V_t difference to recover the secret data described in the following section.

5.3 Experimental evaluation of reading method

Fig. 5 shows the experimental evaluation results for the reading method of secret data. The figure shows the recovered Lincoln image for six different V_{ref}^{secret}. We quantify the reading efficiency using the bit accuracy percentage of the recovered image. The default read reference voltage ($V_{ref}^{secret} = V_{ref}^{default}$) reads the hidden image as an all-zero image (black), and hence bit accuracy is poor. Similarly, if $V_{ref}^{secret} > V_{ref}^{default} + 1V$, the secret image is read as an all-one image (complete white). Thus, there exists an optimal V_{ref}^{secret} for which the bit accuracy of the recovered image is the highest. In this particular example, the optimal read reference voltage is found to be $V_{ref}^{secret}(\text{opt}) = V_{ref}^{default} + 0.45\text{V}$.

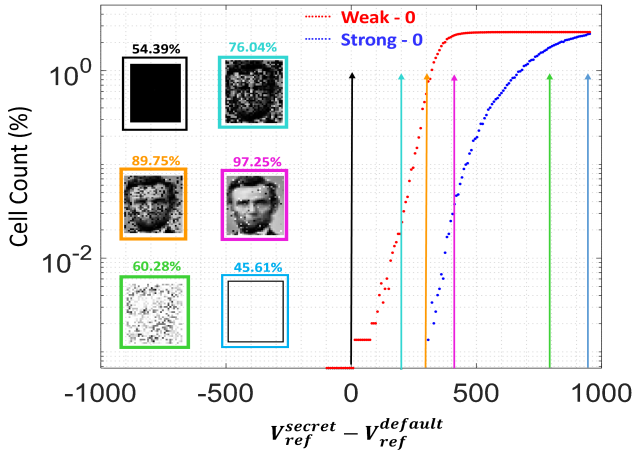

Figure 5: Experimental evaluation of hidden data recovery method.

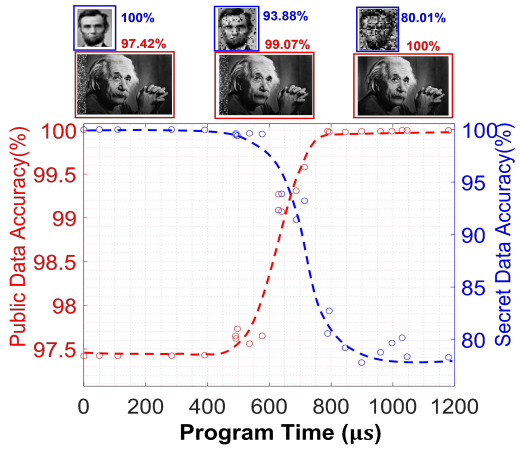

Figure 6: Trade-off between the accuracy of secret data and public data explored by varying partial program time in step 3 of the hiding method.

The optimum V_{ref}^{secret} is a design parameter that needs to be pre-characterized for a given chip. Note that the accuracy of the recovered image is not 100% even with optimum V_{ref}^{secret}. This is due to the overlap between the V_t distribution of strong and weak zero bits. It is very hard to precisely control the cell V_t even with a three-step programming method, and hence achieving very high bit accuracy of the hidden data will be challenging. Multiple redundant copies of the secret image should be stored, and a majority voting scheme might be employed to achieve close to 100% bit accuracy of the hidden data.

5.4 Trade-off between the accuracy of public and secret data

There is an inherent trade-off between the accuracy of the public data and the secret data. Fig.6 illustrates this trade-off. If step 3 of the writing scheme is skipped, the accuracy of the secret data will be approximately 100% with the default read. Since weak zeros are not programmed due to skipping step 3, distinguishing weak and strong zeros becomes very efficient. However, the corresponding public data will have very poor accuracy due to 0⟶1 bit flip errors, as illustrated in Fig. 6.

Next, if we perform step 3 with a partial program operation with increasing partial program time, the accuracy of the public data improves, as shown with red data points in Fig. 6. With longer partial program duration, more zero bits are programmed, and hence bit-accuracy of the public data improves. However, the accuracy of the secret data degrades with longer partial program time, as demonstrated with blue data points in Fig. 6. With increasing V_t values of the weak zero bits, it becomes increasingly difficult to distinguish between the strong and weak zeros, and hence bit accuracy of the secret data decreases.

Therefore, there exists an optimum partial program time that offers the best trade-off by ensuring high bit accuracy (>99%) of the public data as well as secret data. In our characterization, the optimum partial program time is found to be around ≈600 μs. The optimum partial program time is a design parameter in our technique that needs to be pre-characterized for a given chip.

We find that the accuracy of the secret data can be improved further if one employs a sequence of partial write operations with a fixed partial program time in step 3. The multiple rounds of issuing a partial page program operation followed by a read operation from the target page will provide the bit accuracy estimate of the public data after each partial write operation. The process is terminated once all weak zeros are programmed. We have implemented step 3 of our algorithm using five consecutive partial write operations with partial time = 400 μs and found that it offers a very good trade-off between the bit accuracy of public and secret data as described in the next section.

5.5 Trade-off between accuracy and detectability

The proposed method selectively modifies the threshold voltage (V_t) values of zero bits in the public data. Consequently, the V_t distribution of these zero bits could be slightly distorted compared to the distribution that doesn't contain any secrets. This slight alteration could potentially provide an observant adversary with a hint about the existence of concealed secrets.

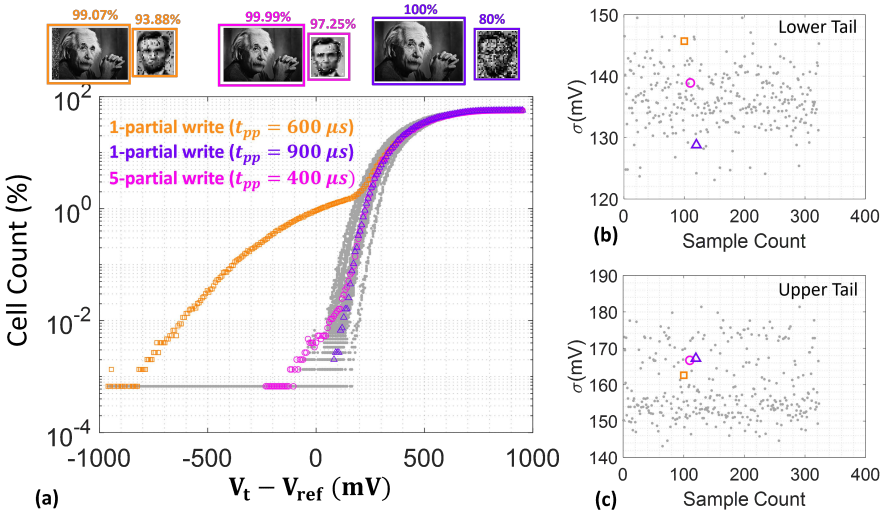

Figure 7: Comparison of V_t distribution of zero bits with secret encoding (proposed write) and without secret encoding (traditional write in gray color). V_t distribution with the proposed write is shown with three colors corresponding to three different programming conditions. The standard deviation of the V_t distribution in the (b) lower tail and (c) the upper tail obtained from several memory pages within a chip.

Fig. 7(a) presents a comparative analysis of the V_t distribution of zero bits, both with and without the presence of hidden secrets. The gray lines in the plot signify the V_t distribution in the absence of any secrets. The presence of several V_t distribution curves derived from the same chip emphasizes the inherent variability in the V_t distribution shape within a single memory chip.

Our findings also suggest that if step 3 of the proposed writing scheme is not optimized properly, it may inadvertently generate a significant signature of hidden secrets that can be revealed through meticulous V_t analysis. For instance, the V_t distribution that results from a lower partial write duration (at step 3) exhibits a long lower tail, suggesting the potential presence of hidden secret data. Increasing the partial write time reduces this lower tail but at the expense of lowering the bit accuracy of the secret data. Hence, a longer partial write time is beneficial for minimizing any anomalous signatures in the V_t distribution. Our study shows that the optimal balance between detectability and the bit accuracy of recovered data is achieved with multiple partial write operations.

The detectability can be quantified by measuring the standard deviations (σ) of the V_t distribution. Given the asymmetry in the upper and lower tails of the cell threshold voltage distributions, we calculate the σ of both tails separately. Fig. 7(b) and Fig.7(c) illustrate the σ values of the lower tail and upper tail, respectively. The grey dots represent the deviations in the V_t distributions that do not contain secret information. We conducted tests on several pages from different blocks and layers, and the scattered grey dots symbolize the inherent process variations across pages in a flash memory chip.

Comparatively, the colored symbols in Fig.7(b) and (c) represent the σ values of the lower and upper tails of distributions that contain hidden information. The alignment of the σ values for the hidden distributions confirms that the secret data is indistinguishable and merges seamlessly with the inherent process variations observed in the flash memory cell threshold voltage distributions.

6 CONCLUSIONS

In this paper, we have demonstrated a data hiding technique in the V_t variation of programmed flash memory bits using commercially available high-density 3D NAND flash memory. Our experimental evaluation results show that the proposed technique can hide secret data without significantly distorting the programmed state V_t distribution. Still, it offers more than 97% bit accuracy in hidden data and more than 99% bit accuracy in public data. In general, our proposed method is universally applicable to all NAND flash chips from any manufacturer, and it can be implemented within the FTL of the storage system.

7 ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under Grants #2145311 and #2007403.

REFERENCES

[1] Bieniasz, J., and Szczypiorski, K. Socialstegdisc: Application of steganography in social networks to create a file system. In *2017 3rd International Conference on Frontiers of Signal Processing (ICFSP)* (2017), pp. 76–80.

[2] Cai, Y., Ghose, S., Luo, Y., Mai, K., Mutlu, O., and Haratsch, E. F. Vulnerabilities in MLC NAND flash memory programming: Experimental analysis, exploits, and mitigation techniques. In *2017 IEEE International Symposium on High Performance Computer Architecture (HPCA)* (2017), pp. 49–60.

[3] Cai, Y., Mutlu, O., Haratsch, E. F., and Mai, K. Program interference in MLC NAND flash memory: Characterization, modeling, and mitigation. In *2013 IEEE 31st International Conference on Computer Design (ICCD)* (2013), pp. 123–130.

[4] Cheddad, A., Condell, J., Curran, K., and Mc Kevitt, P. Review: Digital image steganography: Survey and analysis of current methods. *Signal Process. 90*, 3 (mar 2010), 727–752.

[5] Chen, C., Chakraborti, A., and Sion, R. Pearl: Plausibly deniable flash translation layer using wom coding. *ArXiv abs/2009.02011* (2020).

[6] Fayrushin, A., Seol, K., Na, J., Hur, S., Choi, J., and Kim, K. The new program/erase cycling degradation mechanism of NAND flash memory devices. In *2009 IEEE International Electron Devices Meeting (IEDM)* (2009), pp. 1–4.

[7] Grupp, L. M., Caulfield, A. M., Coburn, J., Swanson, S., Yaakobi, E., Siegel, P. H., and Wolf, J. K. Characterizing flash memory: Anomalies, observations, and applications. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture* (New York, NY, USA, 2009), MICRO 42, Association for Computing Machinery, p. 24–33.

[8] Han, J., Pan, M., Gao, D., and Pang, H. A multi-user steganographic file system on untrusted shared storage. In *Proceedings of the 26th Annual Computer Security Applications Conference* (New York, NY, USA, 2010), ACSAC '10, Association for Computing Machinery, p. 317–326.

[9] Hasan, M. M., and Ray, B. Data recovery from "Scrubbed" NAND flash storage: Need for analog sanitization. In *29th USENIX Security Symposium (USENIX Security 20)* (Aug. 2020), USENIX Association, pp. 1399–1408.

[10] Jia, S., Xia, L., Chen, B., and Liu, P. DEFTL: Implementing plausibly deniable encryption in flash translation layer. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (New York, NY, USA, 2017), CCS '17, Association for Computing Machinery, p. 2217–2229.

[11] Jia, S., Zhang, Q., Xia, L., Jing, J., and Liu, P. MDEFTL: Incorporating multi-snapshot plausible deniability into flash translation layer. *IEEE Transactions on Dependable and Secure Computing 19*, 5 (2022), 3494–3507.

[12] Kumar, M. A., Raquibuzzaman, M., Buddhanoy, M., Wasiolek, M., Hattar, K., Boykin, T., and Ray, B. Total-ionizing-dose effects on threshold voltage distribution of 64-layer 3D NAND memories. In *2022 IEEE Radiation Effects Data Workshop (REDW) (in conjunction with 2022 NSREC)* (2022), pp. 1–5.

[13] Kumari, P., Surendranathan, U., Wasiolek, M., Hattar, K., Bhat, N. P., and Ray, B. Radiation-induced error mitigation by read-retry technique for MLC 3D NAND flash memory. *IEEE Transactions on Nuclear Science 68*, 5 (2021), 1032–1039.

[14] Lach, J. Sbfs - steganography based file system. In *2008 1st International Conference on Information Technology* (2008), pp. 1–4.

[15] Micheloni, R., Crippa, L., Zambelli, C., and Olivo, P. Architectural and integration options for 3D NAND flash memories. *Computers 6*, 3 (2017).

[16] Mizoguchi, K., Takahashi, T., Aritome, S., and Takeuchi, K. Data-retention characteristics comparison of 2D and 3D TLC NAND flash memories. In *2017 IEEE International Memory Workshop (IMW)* (2017), pp. 1–4.

[17] Nam, K., Park, C., Yoon, J.-S., Jang, H., Park, M. S., Sim, J., and Baek, R.-H. Origin of incremental step pulse programming (ispp) slope degradation in charge trap nitride based multi-layer 3D NAND flash. *Solid-State Electronics 175* (2021), 107930.

[18] Pang, H., Tan, K.-L., and Zhou, X. Stegfs: a steganographic file system. In *Proceedings 19th International Conference on Data Engineering (Cat. No.03CH37405)* (2003), pp. 657–667.

[19] Provos, N., and Honeyman, P. Hide and seek: an introduction to steganography. *IEEE Security Privacy 1*, 3 (2003), 32–44.

[20] Raquibuzzaman, M., Milenkovic, A., and Ray, B. Express: Exploiting energy−accuracy tradeoffs in 3D NAND flash memory for energy-efficient storage. *Electronics 11*, 3 (2022).

[21] Spinelli, A. S., Compagnoni, C. M., and Lacaita, A. L. Reliability of NAND flash memories: Planar cells and emerging issues in NAND devices. *Computers 6*, 2 (2017).

[22] Wang, Y., Yu, W.-k., Xu, S. Q., Kan, E., and Suh, G. E. Hiding information in flash memory. In *2013 IEEE Symposium on Security and Privacy* (2013), pp. 271–285.

[23] Zambelli, C., Micheloni, R., and Olivo, P. Reliability challenges in 3D NAND flash memories. In *2019 IEEE 11th International Memory Workshop (IMW)* (2019), pp. 1–4.

[24] Zuck, A., Li, Y., Bruck, J., Porter, D. E., and Tsafrir, D. Stash in a flash. In *16th USENIX Conference on File and Storage Technologies (FAST 18)* (Oakland, CA, Feb. 2018), USENIX Association, pp. 169–188.