Data Recovery from "Scrubbed" NAND Flash Storage: Need for Analog Sanitization

Md Mehedi Hasan and Biswajit Ray, *The University of Alabama in Huntsville*

https://www.usenix.org/conference/usenixsecurity20/presentation/hasan

This paper is included in the Proceedings of the
29th USENIX Security Symposium.

August 12–14, 2020

978-1-939133-17-5

Open access to the Proceedings of the
29th USENIX Security Symposium
is sponsored by USENIX.

Data Recovery from "Scrubbed" NAND Flash Storage: Need for Analog Sanitization

Md Mehedi Hasan and Biswajit Ray
Electrical and Computer Engineering Department, The University of Alabama in Huntsville

Abstract

Digital sanitization of flash based non-volatile memory system is a well-researched topic. Since flash memory cell holds information in the analog threshold voltage, flash cell may hold the imprints of previously written data even after digital sanitization. In this paper, we show that data is partially or completely recoverable from the flash media sanitized with "scrubbing" based technique, which is a popular technique for page deletion in NAND flash. We find that adversary may utilize the data retention property of the memory cells for recovering the deleted data using standard digital interfaces with the memory. We demonstrate data recovery from commercial flash memory chip, sanitized with scrubbing, by using partial erase operation on the chip. Our results show that analog scrubbing is needed to securely delete information in flash system. We propose and implement analog scrubbing using partial program operation based on the file creation time information.

1. Introduction

Secure deletion of obsolete data from the storage medium is a topic of paramount importance to ensure the privacy and security of the data owner. According to the Data Protection Act (DPA) 2018, the deletion of information must be real i.e. the content should not be recoverable in any way. However, achieving true deletion of user data from the physical storage medium is not always straightforward and it depends critically on the analog characteristics of the specific non-volatile storage elements.

In this paper, we evaluate the secure deletion concept in NAND flash based non-volatile storage system. NAND flash memory finds a ubiquitous place in today's computing and storage landscape. Flash memory is widely used in personal electronic gadgets including smartphones, solid state drives, laptops, tablets, USB memory sticks, SD memory cards, etc. Due to the increasing popularity of flash as non-volatile storage media, the concept of secure deletion or sanitization of flash media is getting even more important.

NAND flash exhibits certain unique challenges for secure deletion due to its special characteristics. First, write operation in flash takes place at page level granularity while erase operation happens at block level which requires all the pages in a block to be deleted at the same time. Because of the mismatch in granularity between erase operations and program operations in flash, in-place update of a page is very resource expensive. Second, NAND flash requires erase-before-write constraint, which makes overwriting operation very unfriendly, given the mismatch in granularity between erase and write operation. Third, NAND flash has finite endurance meaning only a fixed number of program and erase operation are allowed on a NAND block. Thus, the flash controller is typically designed to minimize erase operation and ensure wear-leveling of all the memory blocks. In other words, erasing a block for in-place update of a page is not a common practice.

Since in-place updates are not possible in NAND flash, the standard overwrite-based erasure techniques, typically used for hard drives, does not work properly for NAND storage system. Instead, NAND storage usually perform logical sanitization (i.e., the data is not retrievable via the SATA or SCSI interface) by invalidating the page address of obsolete data. The page address mapping in NAND storage is handled by an intermediate firmware layer called Flash Translation Layer (FTL), which performs one-to-one mapping between logical page address and the physical memory address of the flash media. Thus, for any page update operation, FTL will write the new contents to another physical page (or sector) location and update the address table map so that the new data appears at the target logical address. As a result, the old version of the data remains in the physical storage medium, which can be retrieved by the adversary with advanced memory interfaces.

In order to achieve page-level deletion in flash based storage, the idea of "data scrubbing" was proposed by Wei et al[1]. The key concept behind "scrubbing" based sanitization is the creation of an all-zero page (or all cells programmed), which is equivalent to deletion of data from that page. Since it is not possible to remove charge from the floating gates at page level granularity, "scrubbing" provides an alternative route to digital sanitization by programming all the cells in the page. However, in this paper, we show that the deleted data is partially or completely recoverable after "scrubbing" due to the analog property of the programmed cells. More specifically, programmed cells in flash continuously lose charge due to fundamental data retention characteristics. As a result, the zero bits (or programmed bits) in the original data loses a portion of the stored charge at the time of erase operation. We

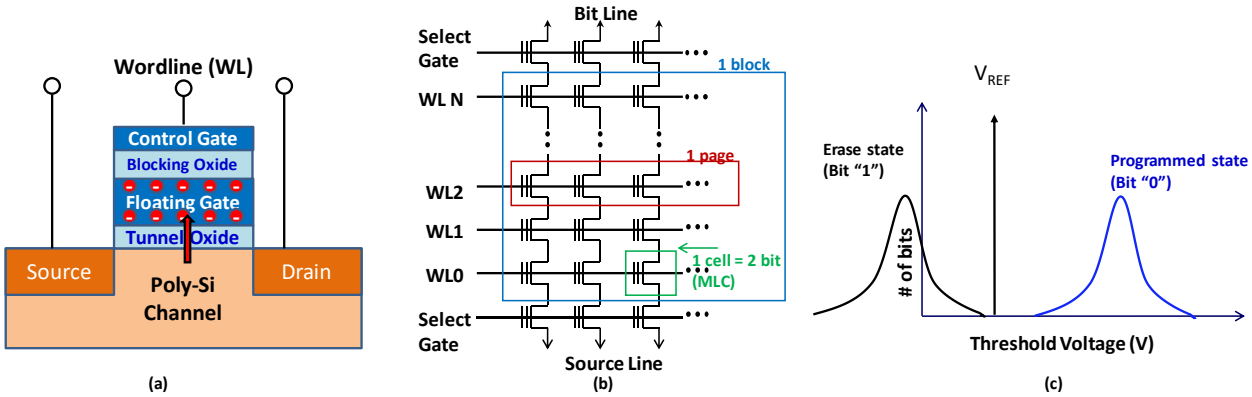

Figure 1: (a) A floating gate (FG) NAND flash memory cell which stores information in the form of charge on the FG. Metal word-line (WL) act as the control gate of the FG transistor. Charge is injected on the FG through tunneling of electron from Si-channel to FG. Blocking oxide prevents back tunneling of electron to control gate. (b) The hierarchical storage in NAND flash array consisting of kilo-bytes of memory cells and the WL electrically connects those cells (called a page of information). Each block consists of multiple WLs. The select gate transistors can be standard MOSFET or FG transistors, depending on manufacturer or technology node. (c) Typical threshold voltage distribution for erase state cell and programmed state cell.

call these zero bits as weak zeros since they have slightly lower threshold voltage compared to the freshly written bits. During "scrubbing" a new set of zero bits are created by newly programming the erased cells in the original data. We label the freshly written zeros as "strong zeros" because they have higher threshold voltage compared to the original zero bits (weak zeros) in the data. Thus, careful analysis of the analog threshold voltage of the memory cells in a scrubbed page will reveal the original data.

Contribution: Our key contributions in this paper are as follows:

1) We demonstrate that data is partially or fully recoverable from "scrubbing" based deleted page. We use partial erase operation on a "scrubbed" page to recover the deleted data.
2) We find that fundamental data retention (or charge loss) characteristics of flash cells should be taken into account during "scrubbing" to ensure true deletion of data.
3) We propose a new analog "scrubbing" technique in order to make sure data remains unrecoverable after deletion. The proposed technique utilizes the time difference between write and erase operation in order to program the erased bits. This will minimize the threshold voltage differences among the cells in the erased page.

2. Background

In this section, we will describe the fundamentals of flash memory cell, its operation, and NAND flash system design.

2.1 Flash Memory Organization and Operations

Flash Cell: Figure 1 (a) shows the device structure of a flash memory cell, which is essentially made of floating gate MOSFET (Metal Oxide Semiconductor Field Effect Transistor). Electrons placed on a floating gate are trapped because the floating gate is isolated electrically from the control gate and the transistor channel by blocking oxide and tunnel oxide respectively. Thus, a flash memory cell stores information in the form of charges (electrons) for an extended period of time without requiring any power supply.

NAND Array: Flash memory is organized as two-dimensional arrays of floating gate transistors. A number of cells connected in series, in a column, form a string (see Figure 1(b)), which is electrically connected to the metal bit line at one end and grounded at the other end. Cells in a row are electrically connected through a metal Word Line (WL) and constitute a page. The size of a page varies from 2-16K byte depending on manufacturer. There can be multiple pages per metal wordline depending on the storage technology. The number of bits per cell depends on the type of flash chip like SLC (single-level cell or 1 bit/cell), MLC (multi-level cell, 2bits/cell) etc.

Threshold Voltage Distribution: The threshold voltage (V_t) of a flash memory cell varies in analog-way depending on the amount of charge on the floating gate. Due to process variation, there is a cell-to-cell difference in threshold voltage, even though the cells are at programmed or erase state. Thus, program or erase state does not represent a single value of V_t. Instead, each state is represented by a V_t-distribution. Flash manufacturer generally keep enough voltage margin between the erase state V_t and the program state V_t so that they can be

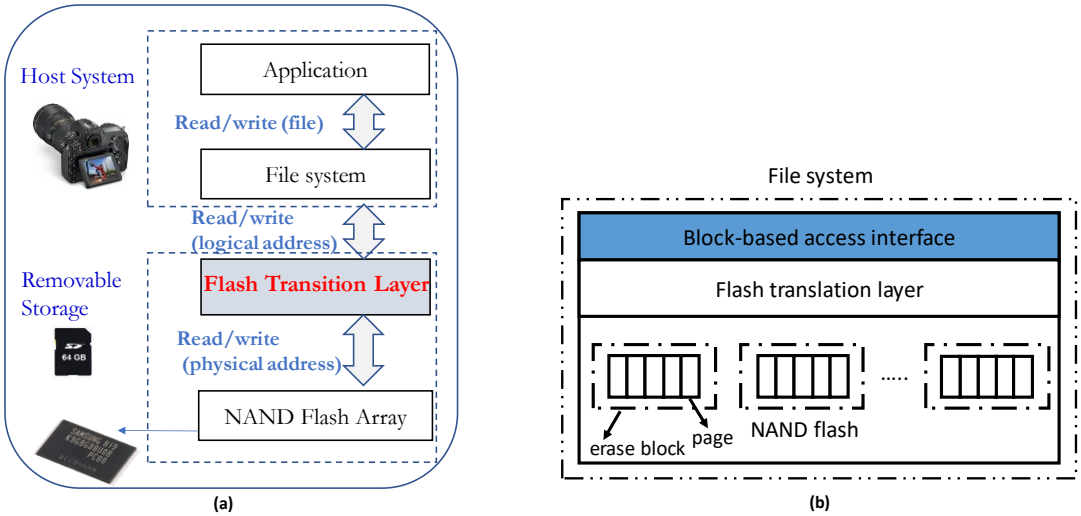

Figure 2: (a) The architecture of a flash-based storage system. (b) File system overview of storage system.

digitized accurately with a single reference voltage. Interestingly, the V_t distribution can be measured with standard digital interface by measuring bit error rate with shifted reference read level (Read Retry operation). The detailed V_t measurement procedure is discussed by Cai et al [2].

Memory Operation: Flash memory offers three basic operations: erase, program, and read. Among the program and read, operations take place on a page by page basis, while erase operation takes place on a block by block basis. During a program operation, a high voltage is applied on the WL which acts as a control gate of the MOSFET and attracts channel electrons into the floating gate by Fowler-Nordheim tunneling through the tunnel oxide. These trapped (negative) electrons increase the threshold voltage of the transistor. In erase operation, these trapped charges are removed from the floating gate by the application of high positive voltage on the substrate and the control gate is grounded. The erase state of flash cell represents logic "1". The programmed state has higher threshold voltage due to the presence of negative charges and it represents logic "0". Read operation involves sensing the threshold voltage of the flash cell by monitoring the current conduction. During a read operation, logic "0" & "1" are sensed by applying an intermediate (or reference) voltage to the control gate, which is less than the programmed threshold voltage. If the cell does not conduct current at the reference voltage, it is treated as a programmed cell or logic state "0". If the channel conducts current at the reference voltage, then the flash cell is considered in erase state and represents logic state "1".

2.2 Flash Translation Layer (FTL)

In order to efficiently manage the NAND array's special characteristics, a firmware layer called flash translation layer (FTL) [3]–[5] is typically used by the storage system which interfaces the host file system with the raw NAND memory. FTL provides a block access interface to the host file system by mapping the logical addresses in block layer to physical addresses in NAND flash. In addition, FTL contains firmware module for garbage collection and wear leveling[3]–[5]. The garbage collection module periodically reclaims all the invalid pages in the media in order to perform block erase operation, which will free-up memory space for new data. The wear-leveling module manages the limited endurance of the flash media by ensuring uniform program-erase operation on all the blocks.

3. Threat model and Assumptions

Adversary Model: We assume the adversary has one-time access to the flash memory device. In addition, we assume that adversary can perform multiple read and erase operations on the content of the flash. We also assume that adversary aims to illegitimately derive sensitive information which is not available through a "legitimate" interface. For example, we assume adversary has access to the raw NAND memory chip and he/she can perform low level memory operation, such as partial erase, shifted read or read retry, etc.

Assumption: We assume adversary can read the data from the NAND flash without any error correction. Most of the NAND chips do not include error correction engine on the memory chip. Instead, the ECC engine is typically included in the FTL. We also assume that adversary can access the NAND flash chip with bypassing the FTL.

4. Data Retrieval after Scrubbing

4.1 Data Retention (DR)

Flash memory technology has finite data retention characteristics as the stored charge on the floating gate (and trapped electrons in the oxides) continuously leaks through surrounding oxides [6]–[8]. Because of DR, when data is stored and kept for some time, the programmed state cell tends to lose its charge and its threshold becomes lower. Figure 3 describes

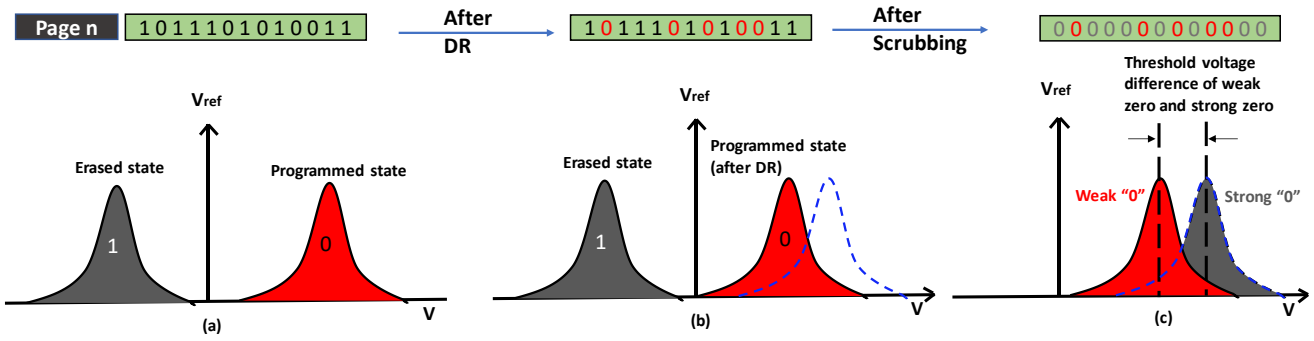

Figure 3: (a) Threshold voltage distribution of memory cells for stored '1' and '0'. Programmed state has a higher threshold than the erased state. (b) Down-shift of threshold distribution of programmed state due to data retention effect. (c) Threshold voltage distribution difference between weak '0' and strong '0'. The newly programmed cell has a higher threshold than the old one.

the data retention effects on the V_t distribution for SLC type of storage. In Figure 3(a), there are two states: erase state and programmed state. Flash manufacturers keep enough voltage margin between the two states. The read reference voltage is typically chosen in the middle of the voltage margin. When the cell voltage is greater than the reference voltage, it reads as logic zero and when the cell voltage is lower than this, the data is read as one. In Figure 3(b) we show the DR effect on the cell V_t distribution. Usually, program state V_t distribution moves down with DR, while erase state V_t distribution remains almost the same. If the data retention time is not large (less than 1-2 years), the program state will still be read as zero even after down-shift of cell V_t distribution.

4.2 Scrubbing after Data Retention

When data scrubbing is applied for page level sanitization after DR, all the data of the page becomes zero. It is important to note that even though all the bits are read as zero digitally, their threshold voltage distribution will have significant and detectable differences. In Figure 3(c), the data retention effect on the "scrubbing" process is explained. When scrubbing operation is applied to a page, which has gone through a finite time of data retention, only the erased cells are programmed. The zeros in the original data remain at the same V_t. However, the newly written zero has higher threshold than the old zeros as old zeros already lose some of their charges. In Figure 3(c), the red zeros are the old zeros which have a lower threshold distribution than the black zeros which have a higher threshold distribution. Thus, the data retention property is the key characteristic to identify the same logical zero as weak zero and strong zero based on their threshold voltage distribution. In other words, if scrubbing is done immediately after writing the data, the difference between old and new zeros will be minimal and it will be difficult to recover the data. If the time difference between write and scrubbing is high, there will be a higher chance that data will be recoverable.

4.3 Partial Erase

We utilized partial erase of a block in order to recover the scrubbing based deleted data. A full erase is the process where all the bits in a memory block turn into the logic state "1". The datasheet of the specific flash chip reports the typical erase time of a block. If the erase operation is interrupted in between by issuing a "RESET" command, then the operation is called partial erase. Partial erase will lower down the V_t distribution of the programmed cells in step by step, which provides a method to distinguish the strong vs weak zeros using standard digital interfaces. Alternative methods exist in order to determine the cell V_t by using digital interfaces, such as read retry, which involves counting fail bits with shifted read levels[9], [10]. However, many SLC NAND chips do not include read retry feature. Hence partial erase offers a more generic method to distinguish weak vs strong zeros.

4.4 Data Recovery with Partial Erase

The adversary can utilize the partial erase operation to distinguish between strong and weak zeros on a fully scrubbed page (meaning all the data being zeros). For example, if the adversary performs partial erase on a fully scrubbed page with fine resolution of erase time, the weak zeros will first turn into ones while the strong zeros will still be read as zeros. Thus, adversary can recover the original data by distinguishing the strong and weak zeros on a deleted page.

In Figure 4 we illustrate the data recovery process with an Einstein image (binary) as an example. We first store the binary image in a NAND block (Toshiba SLC Part # TC58NVG3S0F). The size of the image is 276,000 bytes and it requires 64 SLC pages for getting stored in the memory. In order to accelerate the data retention effects, we then bake the NAND chip at high temperature (120°C) for 3 hours and read back the image data. Figure 4(b) shows the post-bake Einstein image. From a digital viewpoint, there is no difference between the pre-bake Einstein image and the post bake one.

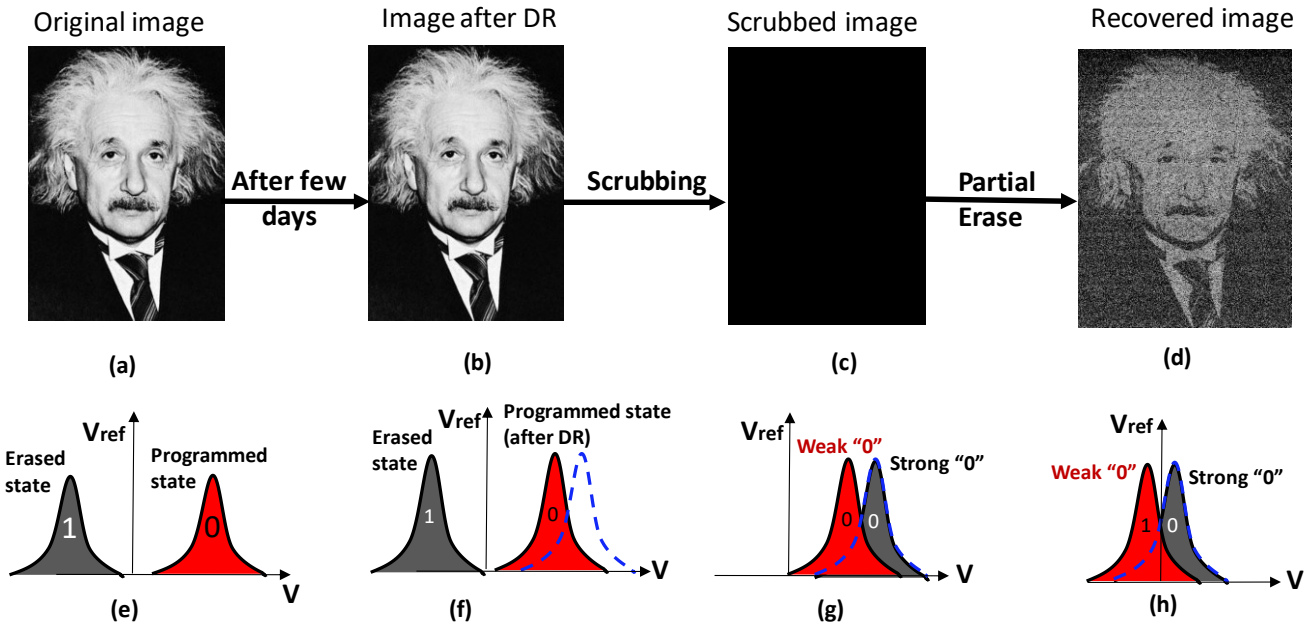

Figure 4: Data recovery from scrubbed pages. (a) Original Einstein image (460 × 600) which is stored to the NAND flash. (b) Original image after data retention takes place (c) Scrubbed image, this is all '0' image. We bake the chip for 3 hours in an oven of temperature 120°C in order to accelerate the data retention effect. (d) The raw image recovered using partial erase operation. (e), (f), (g) & (h) Threshold voltage distribution for corresponding image of (a), (b), (c) & (d).

Algorithm 1: Partial erase on scrubbed data

Initialize:

Target block where data has been stored and scrubbed previously (*TargetBlock*);
Measured partial erase time based on data retention information (PE_{time}) *and define delay* (t_d^e)
Number of pages in a single block ($Block_{page}$);

Perform:

Retrieve data from each page of target block using partial erase ($Retrieve_{page}$);
1: *Issue NAND block erase command;*
2: *Apply time delay* (t_d^e);
3: *Issue RESET command (FFh);*
4: *Issue READ command to read NAND flash data;*
5: *Save the retrieved bytes;*

However, the analog threshold voltage of the memory bits holding the data in Figure 4(a) and the Figure 4(b) are distinctively different as illustrated with the downshifted threshold voltage distribution of zero bits in Figure 4(f). Next, we perform the scrubbing based deletion operation and read the data back. Figure 4(c) shows the deleted image. As expected from a scrubbed NAND data, the image looks completely black or all the bits of the image are at zero state. However, in terms of analog threshold voltage, there is an important distinction between the zero bits. The original zero bits have slightly lower threshold voltage than the newly created zero bits, even though digitally both are read as zeros. Finally, we perform partial erase operation on the scrubbed image to recover back the original data. Partial erase operation will shift down the V_t distribution of both strong and weak zeros in such a way that the memory read operation with a fixed reference voltage will identify most of the weak zeros as one bits and most of the strong zeros as zero bits (see Figure 4(h)). We invert the bit map after partial erase operation and plot the recovered image in Figure 4(d).

Algorithm 1 briefly describes the command sequence used for data retrieval process. First, we select a block where data was stored for some time and then scrubbed recently. So, all the data is read as 0 with standard NAND read command. Then, according to the data storing information (i.e. the time when the data stored and scrubbed), we calculate the optimal partial erase time (PE_{time}). And, then upon applying the precise time delay for the partial erase process, we also issue the NAND Read operation to read data from a specified page. The data read after the partial erase is essentially the inverted version of original stored data. So, we invert the data and save as $Retrieve_{page}$ data. And the final stopping criteria requires 90% of the data become in the erased state. Until this requirement fulfills, the partial operation will continue for the specific block and keep saving data from the specified page for each partial erase operation.

5. Analog Scrubbing

In principle, analog sanitization of the flash media will ensure true destruction of the stored data. Analog sanitization of

semiconductor memory is always challenging because there are many electronic processes that leave imprints of remnant data on the device characteristics[11]–[13]. The block erase operation of the NAND flash device is closer to the analog sanitization of the flash media, as it ensures information is lost by removing the floating gate charge from the programmed cells. However, due to fundamental array architecture of NAND flash, there is no equivalent page-level erase command that converts all the bits in a page into erased bits. Hence developing analog scrubbing method is essential in order to securely delete page data in the NAND array. In this work, we propose the analog sanitization method of NAND memory pages using the history of data creation such as page creation time. The basic idea is to create an all-zero page (similar to digital scrubbing) with the additional constraint that all the zero bits have undistinguishable analog threshold voltage distribution. We implement this idea using partial program technique as described in the next section.

5.1 Partial Page Program

A NAND page generally takes ~100-1000 µs to be fully programmed based on different technology. The partial page program method on a NAND page is typically implemented by issuing a NAND RESET command after the NAND write command. The RESET command will forcibly stop the NAND write operation before its stipulated time. As a result, the memory cells get programmed to a lower threshold voltage level than the corresponding fully programed threshold level. In addition, introducing a time delay (t_d^p) between the NAND write command and the RESET command, it is possible control the analog threshold voltage of the partially programmed cells.

5.2 Analog Scrubbing with Partial Page Program

The goal of analog scrubbing is to match the threshold voltage distribution of the original zero bits and the newly created zero bits during page scrubbing. The challenge here is to estimate the partial program time during scrubbing which depends on the mean threshold voltage value of the original zeros of the page. Hence the knowledge of page creation time and an accurate model for data retention characteristics of the memory chip will be critical to implement this method.

Algorithm 2: Analog scrubbing with partial program

Initialize:

A randomly selected valid block where data has been stored previously;
A randomly selected page in the selected block;
Estimated approximate time delay for analog scrubbing based on data retention information (t_d^p);
Flash chip page program time (t_{PROG});

Perform:

Make all the data either strong zero or weak zero based on time delay;
1: *Issue NAND page write command;*
2: *Apply time delay (t_d^p);*
3: *Issue RESET command (FFh);*
4: *Issue READ command to read NAND flash data;*
5: ***If** scrubbed data < 97% **then***
6: *Repeat 1 to 5*

Algorithm 2 explains the process of analog scrubbing, where we estimate the program time for analog scrubbing process based on data retention information. Note that NAND flash has a default page program time t_{PROG} of ~100-1000 µs. We first select a block where some data has been stored previously and select a page to be read. In this method, depending on the data retention information, program time delay t_d^p is defined for a page ($t_d^p \leq t_{PROG}$). Then we issue the NAND write operation to implement the analog scrubbing. In this case, some of the bit might not be programmed, so if the percent of bit programmed is less than 97%, this process will take place again until the criteria fulfills. After performing scrubbing in this way, difference between 0's threshold voltage distribution is not noticeable, and an adversary is not able to recover the data from this page fully or partially.

6. Implementation and Evaluation

6.1 Experimental set-up

A custom design hardware board is used in order to interface the commercial off the shelf flash chips with the computer. The board contains a socket to hold the flash chip under test and an FT2232H (Future Technology Devices International Ltd) break-out board for USB communication. For the evaluation purpose, we have used SLC NAND flash memory chips from different flash manufacturers including Toshiba, Micron, and Samsung. The exact part number for all the chips used is listed in Table 1.

6.2 Data Retrieval Efficiency after Scrubbing

We evaluate the data recovery efficiency from a digitally sanitized all zero page in Figure 5. The key parameter in the data recovery process is the precise control of the partial erase time. If the duration of erasure is kept long, most of the bits will be erased (similar to standard block erase operation). On the flip side, if the erase duration is too small, then most of the bits will remain in zero states. Figure 5(a) illustrates the gradual data recovery process as a function of erase duration. For a clear illustration, we plot the impact of partial erase time on strong vs weak zeros separately in Figure 5(b). For complete recovery of the scrubbed image, it is required that all the weak zeros (zero bits of the original image) are converted to ones while all the strong zeros (one bits of the original image) remain at zero state. Due to overlap in the V_t distribution of the strong vs weak zeros, a partial recovery of the original image is possible in practice. For example, in the Figure 5(b) we found that at the beginning of erase operation weak zeros turned into ones much faster than the strong zeros.

Part #	Manufacturer	Block erase time (t_{ER})	RESET time (Erase)	Bit recovery efficiency
TC58NVG3S0F	Toshiba	3ms	500µs	77.54%
MT29F8G08ABACA	Micron	2ms	500µs	53.72%
K9F2G08X0A	Samsung	1.5ms	500µs	-
MT29F4G08ABADA	Micron	700µs	500µs	-

Table 1: Evaluation of bit recovery efficiency on different NAND chips.

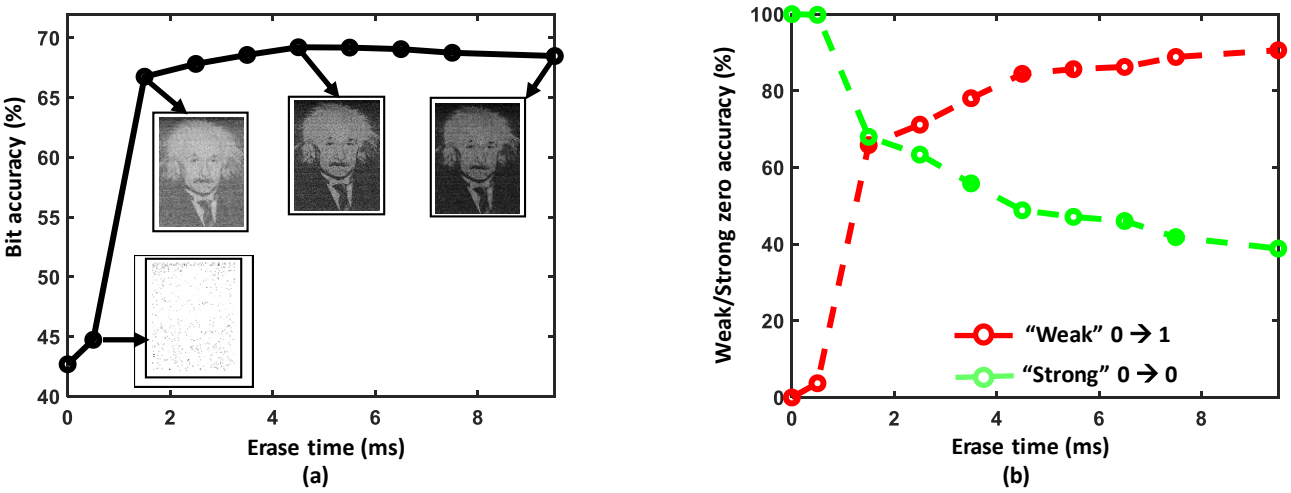

Figure 5: (a) Illustration of bit accuracy of a recovered image vs partial erase time. The image was scrubbed with zero-overwrite method after 3 hours of bake at 120°C. (b) For clear illustration, we plot the percentage of weak zeros that flipped into ones as well as the percentage of strong zeros that remained at zero state with respect to erase time.

However, a significant percentage of strong zeros also flip to ones before all the weak zeros are flipped. Hence there is a narrow window of opportunity for the partial erase duration that ensures recovery of most of the original data.

We define bit accuracy as a new metric for recovered image which measures the percentage of correctly identified bits of the original image data after partial erase operation. We find that there is an optimum erase duration that gives the highest bit accuracy. Please note that a completely scrubbed image (all zero bits or all one bits) will also have a bit accuracy approximately 50% assuming equal number of zeros and ones in the image. However, in order to recognize an image, the correctly identified positions also play a significant role. Hence in Figure 5(a) we find that even though the bit accuracy of the recovered image is low for certain erase duration, the image is correctly recognizable. The other important point to note here is that the maximum bit accuracy is a function of data retention time or the storage history of the image. Typically, the longer the duration of high temperature bake in our experiment (or older the stored data), the separation between V_t distribution of strong vs weak zeros will be wider. This will increase the bit accuracy of the recovered image. In Figure 6 we plot the maximum bit accuracy of the recovered image for different high temperature bake time. We find that the longer the bake duration, the higher the bit accuracy. Note that the maximum bit accuracy of the recovered image corresponding to different bake time is a function different partial erase duration.

We have performed the evaluation of a partial erase based image recovery on chips from different flash manufacturers and found that the partial erase method works for those chips which have longer block erase time. The block erase time of NAND flash chip is defined in the datasheet by the manufacturer i.e. t_{ER} and typically t_{ER} varies from ~1-10 ms. The longer the block erase time, it is easier to control the partial erase operation using the digital interface. Note that the partial erase operation is implemented using our measurement set-up as follows: we issue a block erase operation for a specified NAND block and then we issue the NAND RESET (FFh) command after certain pre-defined time delay (t_d^e). The RESET command takes a finite time (few hundreds of microseconds) to forcibly terminate the erase operation. In addition, the effective time for the pre-mature termination of erase operation depends on the delays associated with issuing commands by the digital interface. Hence the minimum value of the partial erase duration is limited by the time corresponding to the RESET command and the delays associated with the digital interface. For some of the chips as listed in Table 1,

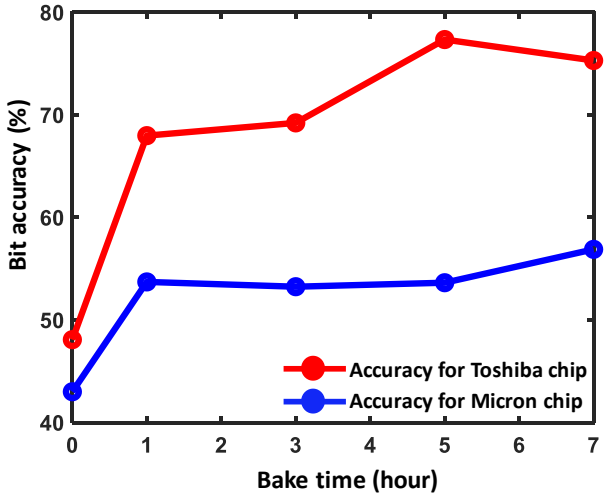

Figure 6: Bit recovery accuracy versus bake time for stored Einstein image on two different NAND flash chips.

we found that the minimum value of partial erase duration is comparable to the block erase time, and hence the NAND block gets completely erased after the partial erase operation with $t_d^e = 0s$. Thus, the data recovery process could not be successfully implemented on those chips using our experimental set-up. A faster digital interface is needed in order to implement data recovery algorithm on those chips with lower block erase time.

6.3 Effectiveness of Analog Scrubbing

In order to show the effectiveness of the proposed analog scrubbing, we perform the following step-by-step experiments. First, we write the same Einstein image on a NAND block and bake it for 3 hours at 120°C to emulate the data retention effects. We then create an all zero page using partial programming technique. The partial program duration is calculated based on memory's data retention characteristics. We then repeat the partial erasure based data retrieval process (as discussed in Section 6.2) on the scrubbed image. The results are shown in Figure 7. We find that recovered image is difficult to be identified compared to the fully scrubbed image. These results show the prospect of analog scrubbing technique for the true deletion of data. However, the accurate implementation requires careful characterization of partial program duration as a function of data retention time, which remains a topic for future investigation.

7. Related Work on NAND Sanitization Methods

In this section we provide a brief overview of the state-of-the-art sanitization methods for NAND flash memory systems. Since in-place updates are not possible in NAND flash, the standard multiple overwrite-based erasure techniques, typically used for hard drives, do not work properly for NAND storage system. Instead, following methods are typically employed by NAND controller for sanitization:

- **Block erase:** Block erasure method is a basic NAND command to remove data from all the pages of a NAND block. The method essentially removes electronic charge from the flash cells and hence physically erase the data from the NAND media. Typically, during garbage collection process this method is used to remove old invalid data once the drive is almost full [3]–[5]. Thus, this command is sparingly used by a NAND controller. However, there are certain sanitization proposals which use this command for secure deletion[14]–[16]. The major drawback of block erasure based methods is the poor performance caused by the significant valid data migration overhead[16]. The other bottleneck for the frequent block erasure is the finite endurance limit of NAND flash technology. Thus, block erasure based immediate page deletion techniques are not a practical and efficient solution for NAND storage.

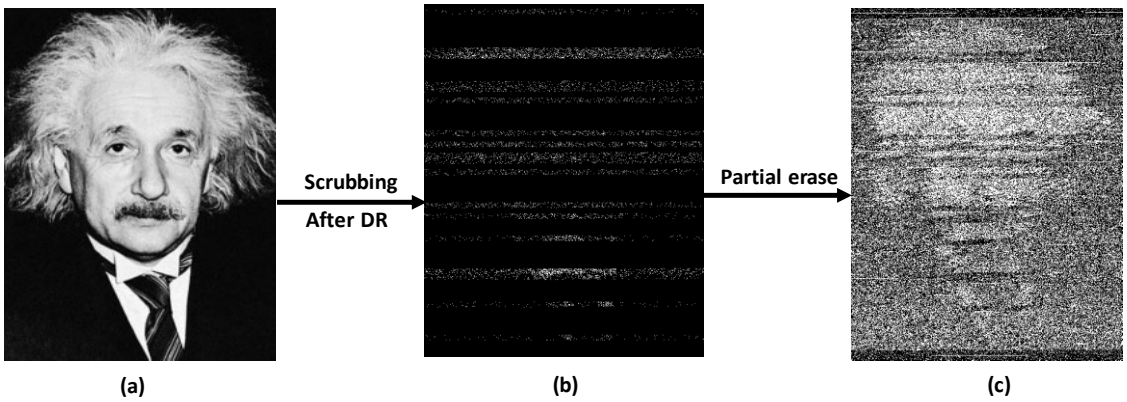

Figure 7: Analog scrubbing output (a) Stored original Einstein image (460 × 600). (b) Scrubbed image after data retention takes place. We bake the chip for 3 hours in an oven of temperature 120°C in order to accelerate the data retention effect. (c) Recovered raw image using analog scrubbing operation.

- **Logical sanitization:** Since block erasure methods suffer from poor performance, NAND storage usually performs logical sanitization by invalidating the page address of obsolete data. The page address mapping in NAND storage is handled by FTL, which performs one-to-one mapping between logical page address and the physical memory address of the flash media. Thus, for any page update operation, FTL will write the new contents to another physical page (or sector) location and update the address table map so that the new data appears at the target logical address. As a result, the old version of the data remains in the physical storage medium, which can be retrieved by the adversary.
- **Encryption based sanitization**: Several authors have recently proposed NAND sanitization methods based on an encryption technique[17]–[22]. The basic idea in this method is to encrypt the user file with an encryption key and store the encrypted data and the key in two separate NAND blocks. Secure deletion is achieved by removing the keys, which can be done efficiently as keys require smaller memory space. Even though encryption based techniques are quite fast, they suffer from the following drawbacks. First, encryption based technique carries the risk of data recovery as its implementation may have certain issues, such as random number generation (for encryption key) which can be compromised by a motivated adversary[23]. Second, encryption based sanitization requires proper sanitization of key storage block and any other derived values that might be useful in cryptanalysis. Third, several existing storage solutions and resource constrained embedded platforms may not include any encryption/decryption module and hence cannot implement this technique.
- **Scrubbing or zero-overwrite based digital sanitization:** In order to achieve page-level deletion in flash based storage, the idea of "data scrubbing" was proposed [1], [16], [24]. The key concept behind "scrubbing" based sanitization is the creation of an all-zero page (or all cells programmed), which is equivalent to the deletion of data from that page. Thus, "scrubbing" provides an alternative route to *digital sanitization* by programming all the cells in the page. However, we have shown in this paper that the scrubbed data is partially or completely recoverable due to the analog property of the programmed cells.
- **History independent erase:** Recently, several researchers have proposed secure NAND deletion methods which will not only remove data from the storage medium but also conceal deletion history from the system[25]–[27]. For example, Jia et al.[26] proposed undetectable secure deletion in flash system by using a partial scrubbing technique and removing any structural artifacts of past deletion operation from the flash system. Similarly, Chen et al. [27] designed HiFlash, a history independent flash device, which will remove all the history related artifacts in the flash layout.

8. Conclusion

In this paper, we show that digitally sanitized (zero overwrite based "scrubbing") NAND flash storage media still maintains the previously written information in the analog threshold voltage characteristics. The data retention property of the flash memory cells causes difference in the analog threshold voltage of the original zero bits in the data and the newly created zeros during scrubbing. We experimentally demonstrate that the difference in the analog threshold voltage can be exploited to recover the deleted data from a fully scrubbed all-zero page. We utilize partial erase technique to recover the deleted data and our evaluation shows more than 75% bits are recoverable depending on the specification of the NAND chip, memory cell's data retention characteristics and the nature of the image data. Finally, we describe a new method for analog sanitization of NAND memory pages using page creation time and partial program technique. Our evaluation shows promise of the proposed technique for analog sanitization and true deletion of user data from the flash media.

References

[1] M. Wei, L. M. Grupp, F. E. Spada, and S. Swanson, "Reliably Erasing Data from Flash-based Solid State Drives," in *Proceedings of the 9th USENIX Conference on File and Stroage Technologies*, Berkeley, CA, USA, 2011, pp. 8–8.

[2] Y. Cai, O. Mutlu, E. F. Haratsch, and K. Mai, "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.

[3] L. Zuolo, C. Zambelli, R. Micheloni, and P. Olivo, "Solid-State Drives: Memory Driven Design Methodologies for Optimal Performance," *Proc. IEEE*, vol. 105, no. 9, pp. 1589–1608, Sep. 2017.

[4] F. Chen, T. Zhang, and X. Zhang, "Software Support Inside and Outside Solid-State Devices for High Performance and High Efficiency," *Proc. IEEE*, vol. 105, no. 9, pp. 1650–1665, Sep. 2017.

[5] N. R. Mielke, R. E. Frickey, I. Kalastirsky, M. Quan, D. Ustinov, and V. J. Vasudevan, "Reliability of Solid-State Drives Based on NAND Flash Memory," *Proc. IEEE*, vol. 105, no. 9, pp. 1725–1750, Sep. 2017.

[6] Y. Cai, S. Ghose, E. F. Haratsch, Y. Luo, and O. Mutlu, "Error Characterization, Mitigation, and Recovery in Flash-Memory-Based Solid-State Drives," *Proc. IEEE*, vol. 105, no. 9, pp. 1666–1704, Sep. 2017.

[7] C. M. Compagnoni, A. Goda, A. S. Spinelli, P. Feeley, A. L. Lacaita, and A. Visconti, "Reviewing the Evolution of the NAND Flash Technology," *Proc. IEEE*, vol. 105, no. 9, pp. 1609–1633, Sep. 2017.

[8] L. M. Grupp *et al.*, "Characterizing flash memory: Anomalies, observations, and applications," in *2009 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2009, pp. 24–33.

[9] Y. Luo, S. Ghose, Y. Cai, E. F. Haratsch, and O. Mutlu, "Enabling Accurate and Practical Online Flash Channel Modeling for Modern MLC NAND Flash Memory," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 9, pp. 2294–2311, Sep. 2016.

[10] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Threshold voltage distribution in MLC NAND flash memory: Characterization, analysis, and modeling," in *2013 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2013, pp. 1285–1290.

[11] P. Gutmann, "Data Remanence in Semiconductor Devices," in *USENIX Security Symposium*, 2001.

[12] P. Gutmann, "Secure Deletion of Data from Magnetic and Solid-state Memory," in *Proceedings of the 6th Conference on USENIX Security Symposium, Focusing on Applications of Cryptography - Volume 6*, Berkeley, CA, USA, 1996, pp. 8–8.

[13] S. Skorobogatov, "Data Remanence in Flash Memory Devices," in *Proceedings of the 7th International Conference on Cryptographic Hardware and Embedded Systems*, Berlin, Heidelberg, 2005, pp. 339–353.

[14] J. Reardon, C. Marforio, S. Capkun, and D. Basin, "User-level Secure Deletion on Log-structured File Systems," in *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, New York, NY, USA, 2012, pp. 63–64.

[15] S. M. Diesburg *et al.*, "TrueErase: per-file secure deletion for the storage data path," in *ACSAC*, 2012.

[16] K. Sun, J. Choi, D. Lee, and S. H. Noh, "Models and Design of an Adaptive Hybrid Scheme for Secure Deletion of Data in Consumer Electronics," *IEEE Trans. Consum. Electron.*, vol. 54, 2008.

[17] J. Reardon, S. Capkun, and D. Basin, "Data Node Encrypted File System: Efficient Secure Deletion for Flash Memory," presented at the Presented as part of the 21st {USENIX} Security Symposium ({USENIX} Security 12), 2012, pp. 333–348.

[18] J. Reardon, D. Basin, and S. Capkun, "On Secure Data Deletion," *IEEE Secur. Priv.*, vol. 12, no. 3, pp. 37–44, May 2014.

[19] J. Reardon, D. Basin, and S. Capkun, "SoK: Secure Data Deletion," in *2013 IEEE Symposium on Security and Privacy*, 2013, pp. 301–315.

[20] J. Lee, J. Heo, Y. Cho, J. Hong, and S. Y. Shin, "Secure deletion for NAND flash file system," in *In ACMSymposium on Applied Computing,2008*.

[21] S. Jia, L. Xia, B. Chen, and P. Liu, "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, pp. 2217–2229.

[22] L. Yang, T. Wei, F. Zhang, and J. Ma, "SADUS: Secure data deletion in user space for mobile devices," *Comput. Secur.*, vol. 77, pp. 612–626, Aug. 2018.

[23] T. Ristenpart and S. Yilek, "When Good Randomness Goes Bad: Virtual Machine Reset Vulnerabilities and Hedging Deployed Cryptography," in *Ndss '10 (network and Distributed Security Symposium)*, 2010.

[24] W. Wang, C. Ho, Y. Chang, T. Kuo, and P. Lin, "Scrubbing-Aware Secure Deletion for 3-D NAND Flash," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 37, no. 11, pp. 2790–2801, Nov. 2018.

[25] B. Chen, S. Jia, L. Xia, and P. Liu, "Sanitizing Data is Not Enough!: Towards Sanitizing Structural Artifacts in Flash Media," in *Proceedings of the 32Nd Annual Conference on Computer Security Applications*, New York, NY, USA, 2016, pp. 496–507.

[26] S. Jia, L. Xia, B. Chen, and P. Liu, "NFPS: Adding Undetectable Secure Deletion to Flash Translation Layer," in *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, New York, NY, USA, 2016, pp. 305–315.

[27] B. Chen and R. Sion, "HiFlash: A History Independent Flash Device," *ArXiv*, vol. abs/1511.05180, 2015.