Solid State Drive Forensics: Where Do We Stand?

John Vieyra[1], Mark Scanlon[2], Nhien-An Le-Khac[2]

[1]Canada Border Services Agency, Canada.
john.vieyra@ucdconnect.ie

[2]Forensics and Security Research Group, University College Dublin, Ireland.
{mark.scanlon, an.lekhac}@ucd.ie

Abstract. With Solid State Drives (SSDs) becoming more and more prevalent in personal computers, some have suggested that the playing field has changed when it comes to a forensic analysis. Inside the SSD, data movement events occur without any user input. Recent research has suggested that SSDs can no longer be managed in the same manner when performing digital forensic examinations. In performing forensics analysis of SSDs, the events that take place in the background need to be understood and documented by the forensic investigator. These behind the scene processes cannot be stopped with traditional disk write blockers and have now become an acceptable consequence when performing forensic analysis. In this paper, we aim to provide some clear guidance as to what precisely is happening in the background of SSDs during their operation and investigation and also study forensic methods to extract artefacts from SSD under different conditions in terms of volume of data, powered effect, etc. In addition, we evaluate our approach with several experiments across various use-case scenarios.

Keywords: SSD Forensics, Forensic Experiments, Data Recovery, TRIM.

1 Introduction

As the design of computers has improved with time, many manufacturers have moved from traditional Hard Disk Drives (HDDs) to Solid State Drives (SSDs). These SSDs are smaller and more compact than HDDs. They are also more robust and resilient to vibration and allow for much greater Input/Output (I/O) data transfer speeds. These drives contain no moving parts and store each bit of data in floating gate transistor rather than on a magnetic spinning platter of a HDD. Although these new types of drives have many advantages, they also have some limitations. These drives have a limited number of writes per cell, can only write in pages, and must erase a full block of pages before rewriting any single page. Depending on the type on NAND flash, some SSDs may come with a number of bad areas that need to be corrected. Because of these noted limitations, data is stored using non-traditional methods such as using error correction code, bad area management, and scrambling, etc.

On the other hand, these design changes have created much interest in the forensic community regarding how SSD data is stored and recovered from these non-traditional devices. Many papers have been written detailing how different SSDs are compared to the traditional HDDs. Some performance and forensic testing of these SSDs show disturbing results, such as nearly all the data being lost within a couple of minutes when a format command was sent [1]. This action by SSDs is "quite capable of essentially near-complete corrosion of evidence entirely under their own volition" [1]. These comments in the hands of a savvy criminal defense lawyer, could be wrongfully used to undermine the results of a competent forensics examiner. It was not the forensic examiner or the SSD itself that has caused the drive to delete its data, but the actions of the suspect entering the format command. The SSD simply reacted to the command as it has been designed to do. This is analogous to a suspect shredding documents to the point that they are irrecoverable just prior to a search warrant being executed. SSDs limitations have caused designers to implement many techniques to overcome some of these issues. Hardware manufacturers have added a system of garbage collection, wear levelling, and created TRIM to mitigate these limitations [3].

Mobile devices, e.g., smartphones and tablets, typically use NAND flash memory that interacts with the OS during the evidence recovery process. Every day, evidence is successfully entered in court retrieved from seized cellphones. Whether it is call logs or skype chats, this information is valuable in assisting the prosecution in criminal cases. However, every time a cellphone is turned on for an extraction, changes are made to the NAND Flash memory and the evidence is "walked on" during this process, i.e., in an analogous manner to walking onto a physical crime scene. Perhaps a boot loader needs to be installed on a cellphone so that the data area can be imaged, or a Factory Reset Protection Lock removed from an Android phone to allow circumvention of the device's passcode. In all cases, these methods need to slightly walk on the evidence. The premise of hashing the storage and subsequently ensuring that all images match the original hash no longer applies. The memory of a cellphone is constantly changing whenever power is applied to the controller. The process now requires that some evidence is deterministically changed for the greater good of the recovery process. However, this does not mean that evidence is added by the forensic examiner, and any changes must be kept to a minimum. Plus, once the NAND Flash memory is imaged it is now time to hash the image and confirm that all further image copies are verified with the hashing process [2].

Through the literature review for this work, we found that there is a gap in terms of forensic acquisition and analysis of SSD drives. Hence, we aim to provide some clear guidance as to what precisely is happening within the background of these SSDs and to demonstrate that most forensic Hard Disk Dive (HDD) procedures still stand when it comes to their analysis. The study we performed has been designed to recreate what would be found in any typical computer seizure. The analysis is designed to determine the following research answers:

1. Does the total amount of data stored on a SSD affect what evidence that can be recovered from the drive?
2. Does time the SSD is left powered on effect the available data?
3. What effects does TRIM have on the outcome of stored data within the drive?

4. When things do change, comparisons are performed to determine the effects of these changes.
5. How much data is left after formatting a SSD drive with both TRIM enabled and disabled?
6. Can we tell if garbage collection is happening in the background?

2 Solid State Drive Characteristics

The Flash Translation Layer (FTL) is an abstraction layer between what the Operating System (OS) sees or communicates with, and the actual way the data is stored within the SSD. The FLT makes the SSD look like a HDD to the OS. The first two things hidden behind the FTL that are important: garbage collection and wear levelling.

2.1 Garbage Collection

Garbage collection is a process that takes place in the NAND Flash memory to prepare memory cells for new data when they have previously been used to store data. NAND flash has many limitations that must be addressed by the controller chip. Some of these limitations are:

- It is unable to overwrite original data at the byte level.
- The smallest writable area is a page.
- It is made up of blocks of data containing pages.
- It has a limitation that only allows the memory to write in pages and delete in blocks. For data to be overwritten, the entire block must first be cleared. There is one exception to this, where the data to be written is a strict subset of the data. This means that the data has the exact same zero values and can write ones to zeros. It cannot write a zero to a one. If you want to write a zero to a one, then the entire block needs to be cleared and reset to accept new data.

When a page needs to be cleared you are required to clear the entire block. The problem here is that some of the adjacent pages in the same block may still have allocated data. In this case the allocated pages must be moved to a new block, and then the entire block is deleted, i.e., garbage collection. This does not solely happen when the OS tries to overwrite previous data. If the SSD supports TRIM, the OS sends a TRIM command to the SSD when it deletes data notifying the SSD that the area is no longer needed. The SSD controller then uses this information along with garbage collection to prepare new areas ahead of the time when it will be needed [4, 5].

2.2 Wear Levelling

During the regular use of a computer, many files may be put on the device's storage and remain for the entire life of that computer. Typical users would also have many files that get updated regularly. This becomes a problem for the SSD as certain areas

might be hitting the end of their usable life, while others may have just one write cycle and never been erased. The ideal scenario would be to have all the blocks fail at the exact same time. By doing this you allow the longest lifespan of the drive and the best user experience. To facilitate this behavior, the SSD controller moves data around, so that areas with low write/erase cycles are now used more frequently. This happens inside the SSD and behind the FTL and is not transparent to the OS and the end user.

There are three types of NAND flash memory and each type has a different lifespan:

- Single-Level Cell (SLC): These are high performance, enterprise grade devices. They perform up to 100,000 program/erase cycles per cell. They have lower power consumption, with faster write speeds and a much higher cost (up to three times higher than MLC) [8].
- Multi-Level Cell (MLC): This is average performance consumer grade storage. They perform up to 10,000 program/erase cycles per cell. They have higher density (two or more bits per cell), a lower endurance level than SLC, and a lower cost (up to 3 times lower than SLC). These are used in consumer goods and are not suggested for critical applications that have frequent updates of data [8].
- Three-Level Cell (TLC): These are the lower performance, lowest cost option. They perform up to 5,000 thousand program/erase cycles per cell. They have the highest density (three bits per cell), lower endurance level, and slower read/write speeds than MLC. They are a good fit for low end consumer products and are not recommended for critical applications that have frequent updates of data [6].

2.3 Data Behind the FTL

Since the data stored on SSDs is hidden behind the FTL, to directly examine it, the memory chips must be removed from the SSD circuit board. The NAND flash memory can now be directly read bypassing the controller. When we read each chip, we have no way of determining how the controller has stored the data and what it will take to put it back into a usable format. The details of the controller and the firmware are trade secrets of the manufacturer. The companies manufacturing these SSDs want to hide their technique for providing the best possible performance. To determine how SSDs store the data, visual storage representation software is used to display the data. This software also allows you to reassemble the data. Two companies that have designed products to recover this data in this way are ruSolute and ACE Labs [7].

2.4 TRIM

TRIM is a term (not an acronym) used to identify certain ATA commands that allow the OS to notify the SSD controller that data is no longer needed. The meaning of word TRIM is simply coming from the fact that the area of the drive is reduced or trimmed (made smaller). TRIM became necessary to allow the OS to tell the SSD that an area is no longer needed.

In the case of HDDs, when the OS deletes a file, the OS updates the file allocation table and marks the area as unallocated. The underlying data is not deleted from the HDD. This becomes a problem for SSDs since they need to prepare deleted areas before allowing any new data to be saved in this area. SSDs are required to write in pages (usually 512 bytes) and delete in blocks. This process works fine if the drive is not full and has available space to write its data, however when the drive starts to become full things change drastically. The drive now needs to find a page to store the data. This makes the controller work harder to try and find available space. To do this, it will start moving allocated pages from blocks with many unallocated pages to new blocks. It then clears the entire block, i.e., garbage collection. This results in a significant slowdown of the device. TRIM attempts to overcome this issue by providing a way for the OS to tell the SSD that it no longer needs certain areas. The SSD controller is now able to perform many of the functions needed to clear data well in advance of any need from the OS. These internal processes could also be done at times when the SSD is under minimal load causing the process to be hidden or masked from the user. For TRIM to be active, three things need to be present: the OS must support it, it must be supported by the SSD, and it must be supported by the file system that is used.

3 Related Work

NIST suggest that modern SSDs are subject to a process that is "self-corrosive" [2]. It suggests that these self-corrosive processes are done in the absence of computer instructions. Gubanov et al. [4] explain how the data destruction process is only triggered by the TRIM command, and the data destruction itself is carried out by the separate process of background garbage collection. The authors also explain why different outcomes are found as a result of analysis when TRIM is operating. In some cases, they found all zeros, and other times they find actual data. This is explained because some drives implement Deterministic Read After TRIM (DRAT) and Deterministic Zeroes After TRIM (DZAT). With DZAT the drive returns all-zeros immediately after the TRIM command releases a certain data block. In the case where a drive uses non-deterministic TRIM, each read command after TRIM may result in different data. In some cases, the data read can return different non-original data, not because the data has been cleaned, but because the SSD controller says that there is no valid data held in these areas. Some areas of logically corrupted data allow for recovery since the TRIM command is not issued over corrupted areas. A portion of the authors' conclusion is very important, where they note that many SSDs follow the DRAT approach, and therefore a simple format of the drive is likely to instantly render all the data inaccessible to standard read operations. Write blockers will have no effect to stop this process.

The comparison of reactions in SSD drives using different file systems and TRIM has been detailed in [8]. This is the first analysis over different file systems including NTFS, EXT4, and HFS+ that support TRIM on SSDs. This paper explains that when TRIM is enabled, the OS notifies the drive that data has been deleted from its location and then marks the location as invalid. In this paper, the authors determined that when

TRIM was enabled, the deleted data was purged and unrecoverable within minutes. This was not the case in EXT4 as the commands are sent in batches and therefor may not be sent immediately. They also found that manual TRIM being used as an anti-forensics method was not very successful and suggested the use of the ATA Secure Erase standard built into most SSDs.

Bednar and Katos presented two challenges for digital forensics of SSDs [9]. The first is that data that has been deleted is not necessarily removed from the disk because the logical structures are not necessarily mapped to physical locations on the disk. Data is required to be processed through a complex algorithm that is known only to the manufacturer. The second is that the controller purges data on its own independently of the OS. This happens whenever the drive is powered on. The authors also suggested two options to overcome these issues. One being that the memory chips could be removed from the drive and read independently. The data could be put back together for analysis. The second option would be to remove the drive controller and replace it with one that is forensically safe to recover all the data. The later would be considered almost impossible due the many different variations available as well as the difficult requirement to tamper with the evidence. The authors suggested that hash functions cannot be used to determine the integrity of SSDs as they will inevitably change over time. This will result in potentially different hash values each time they are imaged. The use of write blockers is somewhat negated as they will not stop the writes made internally to the data on the memory chips. The write blockers are attached outside of the drive on the SATA cable; therefore potentially making the data not acceptable for use in court.

Shah et al. studied the forensic potentials of SSDs from different manufacturers and determined what data will be available after deletion from the SSD. It is suggested that data can be recovered from an SSD in a similar manner as a HDD if the SSD does not have background garbage collection and TRIM has been disabled. They also indicate that data can be recovered after the SSD has been formatted. The use of TRIM only functions when the drive is connected to the SATA or NVMe primary channel. In cases where the drive is connected via a secondary SATA channel or via a USB connection, TRIM does not function, and therefore the deleted data is entirely available for recovery.

With empirical analysis, King et al. [11] tries to show how much data is retained on 15 different SSDs. The authors listed drive models with details of how much data is recovered with and without TRIM enabled. The authors found that data recovery using TRIM-enabled disks was practically impossible for large disks, showing a near zero percent of data. For the small files, these results varied with the SSD manufacturer. Some reported 25-30%, while Intel reported zero percent. They suggested that different results were caused by Intel using a proprietary controller software, while the others used software licensed from Indilinx. They also found that without TRIM (using Windows XP), they could recover almost 100% of the data. This occurred for both large and small files. An important point was that they seemed to be able to recover higher percentage of data when dealing with high usage disks. They commented that this was likely due to garbage collection struggling to prepare disk space when dealing with high usage. They found that the TRIM command has made all data not recoverable. Without TRIM enabled, allows for nearly 100% of the data being recovered. They state that “as

SSD adoption grows as well as use of Windows 7 and other TRIM-supported OS's, traditional data recovery will no longer be a viable option for Investigators."

4 Experiments and Findings

In order to answer the research questions raised in the introduction, this section describes several case studies with analysis.

4.1 Platforms

In this section, we describe the datasets and hardware we used in our experiments. The sample data needed to fill large a volume of drive space and needed to be a type of data that can be measured in free space after deletion. To do this, large text files were created with a repeating string value followed by a unique MD5 hash value. By using an alliteration multiple repeating values could be output to numerous text files thereby filling a folder to a pre-selected volume. The string and MD5 combination is then used to name the folder created to store all the data. Briefly, we have a folder named with the string and MD5 hash combination and numerous text files containing the sample data, as can be seen in Figure 1. Each of these folders will be searched for the string-MD5 combination. A count of the number of hits will be recorded for comparison, shown in Figure 2. This data will then be used to determine the percentage of recovered data in each case.

Fig. 1. View of sample of repeating data inside the one of the text files.

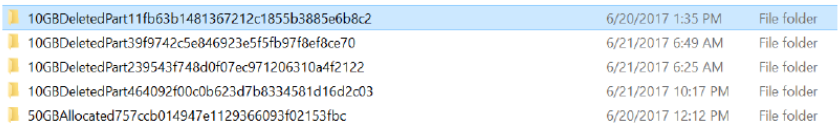

Fig. 2. Sample data used in analysis.

Twelve drives (two of each model) were used in our experiments. These were brand new and had not been used for any other purpose. Ten of the drives were 250 GB and two were 500 GB. Each drive is attached to a Z87X-UDSH Gigabyte motherboard, Intel i7 4770 CPU @ 3.5 GHz computer, on the primary SATA channel. Windows 10, 64-bit Pro edition was installed on a NTFS partition. This is done to represent a real-life scenario where a laptop or desktop computer is encountered during a typical search warrant. The odd numbered drives were confirmed to have TRIM operating by using the command line entry `fsutil` behavior query `DisableDeleteNotify`. If the response received is `DisableDeleteNotify = 0`, then TRIM is turned on. The even numbered drives had the TRIM turned off using the `fsutil` behavior set `DisableDeleteNotify = 1`, and then confirmed using `fsutil` behavior query `DisableDeleteNotify`. If TRIM is turned off correctly the response received is `DisableDeleteNotify = 1`.

4.2 Experiment 1

Through this experiment, we aim to reply the research questions 1, 2, 3 and 4 raised in Section 3.

Four sets of data folders were copied to each drive. Each set represents approximately twenty-five percent of the 250GB drive volume (in the case of the 500GB drives, two sets are copied each time). These sets contain a 50GB folder of allocated data that will be left on the drive and not deleted. The sets also contain an additional 10 GB of data that will be deleted immediately without going to the recycle bin (i.e., "shift delete" to emulate user behavior). Each of these 10 GB folders will be unique to the set. After each set is copied to the drive, the drive is imaged immediately using either a Tableau TD2 Forensic Imager, or a Tableau T35u Forensic SATA/IDE Bridge and FTK Imager software version 3.4.3.3. The drive is then left powered on idle for one hour and reimaged. After a further 8 hours of being powered on idle, it is again imaged one final time. This entire process is completed three more times, each time after adding another set of data containing an additional twenty-five percent of data.

Each image is then opened in X-ways forensics Version 19.0 SR-4 x64 software and a text search is initiated for each of the combined string and MD5 values with the option selected to count the number of hits. These values will be compared to the original data that was added and then deleted to determine how much data can be recovered. This will be completed four times in total, once after each additional amount of data is added. Comparisons will be made and percentages will be calculated for each type of data, trying to determine how much is recoverable after each addition. In cases where the image hash of the one hour and eight hour images match the zero hour hash, then only one of the images will be searched as the data will be exactly the same. In any cases where the zero hour, one hour, or eight hour hashes do not match, then the different images will be compared in X-ways forensics to try and determine what has changed. These comparisons should be able to tell if any changes are taking place inside the SSD that are visible to the OS through the FTL.

First part of this experiment consisted of adding data to fill approximately 25 percent of the drive. The analysis of this experiment showed varying amounts of data available

for recovery. These were found to be from a fraction of a percent to nearly all the data available for recovery after data deletion. What was clear from the results is that drives that had TRIM enabled on the OS, had significantly lower amounts of data available for recovery. The images taken at the 0 hour, 1 hour, and 8 hours, as can be seen in Figure 3, were all the same, i.e., their hashes matched.

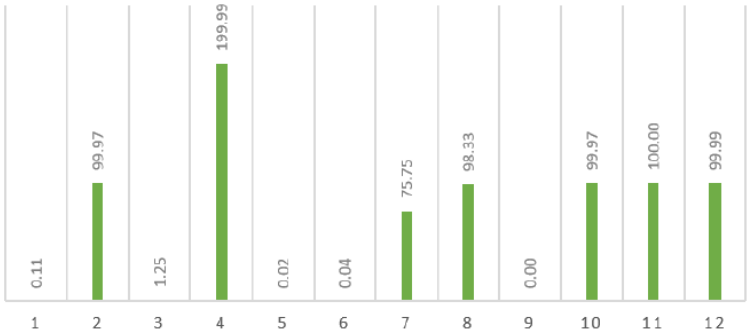

Fig. 3. Recovered data percent, first data set (even numbered drives had TRIM disabled, odd numbered drives had TRIM enabled). Note: Drives 3 and 4 were over twice the size of the others.

The second part of this experiment consisted of adding another 25 percent (50% full) of data to the drive. The analysis of this experiment again showed varying amounts of data available for recovery. These were found to be from a fraction of a percent to about 50 percent of the data was available for recovery after data deletion. What is again clear from the data is that drives that had TRIM enabled on the OS, had significantly lower amounts of data available for recovery. The images taken at the 0 hour, 1 hour, and 8 hours, as outlined in Figure 4, were again all the same, i.e., their hashes matched.

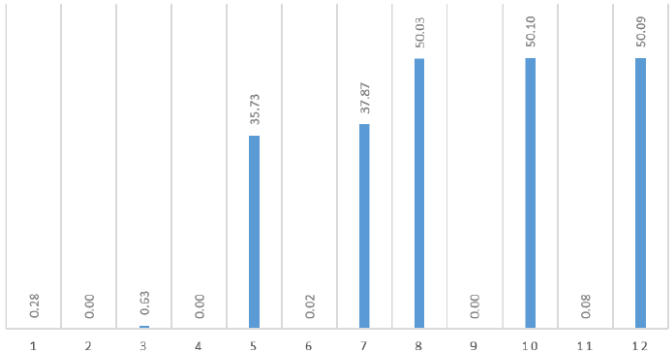

Fig. 4. Recovered data percent, second data set (even numbered drives had TRIM disabled, odd numbered drives had TRIM enabled).

The third part of this experiment consisted of adding another 25 percent (75% Full) of data to the drive. The analysis of this experiment again showed varying amounts of

data available for recovery. These were found to be from a fraction of a percent to about 65 percent of the data was available for recovery after data deletion. What is again clear from the data was that drives that had TRIM enabled on the OS, had significantly lower amounts of data available for recovery. The images taken at the 0 hour, 1 hour, and 8 hours, as shown in Figure 5, were again all the same, i.e., their hashes matched.

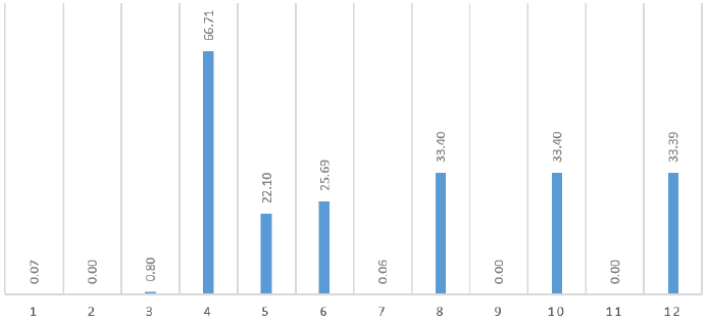

Fig. 5. Recovered data percent, third data set (even drives had TRIM disabled, odd drives had TRIM enabled).

The fourth part of this experiment consisted of adding another 25 percent (nearly 100% Full) of data to the drive. The analysis of this experiment again showed varying amounts of data available for recovery. These were found to be from a fraction of a percent to about 65 percent of the data was available for recovery after data deletion. What is again clear from the data was that drives that had TRIM enabled on the OS, had significantly lower amounts of data available for recovery. The images taken at the 0 hour, 1 hour, and 8 hours, as outlined in Figure 6, were again exact matches.

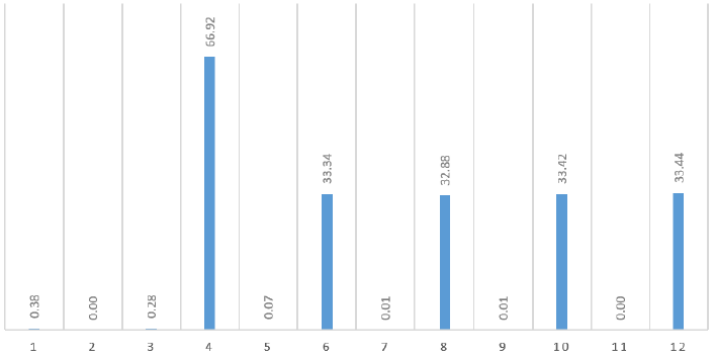

Fig. 6. Recovered data percent, fourth data set (even numbered drives had TRIM disabled, odd numbered drives had TRIM enabled).

The analysis of this data showed that data did not change over time and what seems to make the drive reduce the amount of previously deleted data from being recovered is the addition and deletion of data and not the time that the drive is powered on.

4.3 Experiment 2

In this experiment, we aim to find out "how much data is left after formatting a SSD drive with both TRIM enabled and disabled?". We are filling drives to near full capacity and formatting the drive.

The drives outlined in Experiment 1 are also used in Experiment 2. Drive 3 to 12 each have additional data added so that the free space is below 1 GB. Each of these drives is then imaged using the same process as Experiment 1. Drives 1 and 2 will not have any additional data added besides the operating system. These two drives will be used as bootable drives on the Primary SATA channel with Drives 3 to 12 being quick formatted on the secondary channel. Drive 1 is matched to Drives 3, 5, 7, 9, and 11, so that the odd drives are formatted with a TRIM enabled OS. Drive 2 is matched to Drives 4, 6, 8, 10, and 12, so that even drives are formatted with a TRIM disabled OS.

Once all the drives are formatted, they will be immediately shut down. All ten drives will be reimaged, and then again after being powered on and left idle for an additional 8 hours.

Each image is then opened in X-ways Forensics Version 19.0 SR-4 x64 software and a text search is initiated for each of the combined string and MD5 values with the option selected to count the number of hits. The amount of recovered data from images taken before the formatting will be compared to the recovered data from images taken after the formatting. From this data an analysis of how formatting the drives will affect the data recovery.

The results showed that all the data was removed and set to zeros on eight of the drives, however two drives (Drives 5 and 6) had nearly all the data available for recovery. This abnormality is difficult to understand without knowledge of the inner workings of the devices' firmware. The TRIM version containing 99.63% and the Non-TRIM version containing 82,83%, as shown in Figure 7.

	Before Format	After Format
Drive 1		
Drive 2		
Drive 3	10,000.334,198	0
Drive 4	10,000,078,637	0
Drive 5	4,444,480,004	4,428,146,536
Drive 6	4,444,492,854	3,681,448,761
Drive 7	4,444,481,343	0
Drive 8	4,444,480,371	0
Drive 9	4,444,483,675	0
Drive 10	4,444,480,004	0
Drive 11	4,444,483,170	0
Drive 12	4,444,524,993	0

Fig. 7. Amount of recovered data after a format on Drives 3 to 12 (Drive 1 and 2 were used as the base OS drives and therefore no data was gleamed from them).

4.4 Experiment 3

Experiment 3 tested if the SSD could be overwhelmed with garbage collection and if this could be seen somehow. Testing was conducted using the format command in Microsoft Windows 10 GUI file explorer. It used a new Samsung 500 Gb SSDs and measured the time to copy data to 500 GB of data to it until it reported being full. The drive would then be formatted, and another 500 GB of data would immediately be written to it. The hypothesis is that the measurement of time should significantly increase if the drive now suddenly needs to make room for the new data using garbage collection. The time of each write will be compared to see if background garbage collection will slow the data transfer after the format takes place. In this experiment no significant difference in time was found between the first and second writes. The action of background garbage collection could not be seen using this method.

4.5 Experiment 4

Experiment 4 tested if the SSD could be overwhelmed with garbage collection and if this could be seen somehow. Testing using deleting of data and rewriting.

It used a new Samsung 500 Gb SSDs and measured the time to copy data to 500 GB of data to it until it reported being full. The data on the drive was all deleted (shift delete) and another 500 GB of data would immediately be written to it. The hypothesis is that the measurement of time should significantly increase if the drive now suddenly needs to make room for the new data using garbage collection.

The time of each write will be compared to see if background garbage collection will slow the data transfer after the data is deleted. Experiment 4 was similar to Experiment 3, however instead of formatting the drive, the data was simply deleted. This was again performed on a new drive that had never been used. The fact that a similar time was required was the same as in Experiment 3.

4.6 Experiment 5

Experiment 5 was performed to see if the action of garbage collection could somehow be seen by monitoring the power requirements.

In this case, the SSD was connected to a separate power supply and attached in series to a meter that could measure the power consumption to 1/100th of a milliamp. The drive was filled with data and then deleted or formatted as in Experiments 3 and 4. The hypothesis is that the drive should draw a high-power level until garbage collection ends and then settle down to an idle value. The amount of current will be monitored to see if background garbage collection can be observed.

In all cases, after the data was deleted or the drive was formatted the drive drew between 250 and 300 milliamps initially then within 10-15 seconds settled to approximately 150 milliamps. This stayed constant and never reduced after two days. Again, this experiment was inconclusive in determining if garbage collection could somehow be observed working in the background.

4.7 Analysis

Forensic examiners have always been taught that data must be unchanging and that hashes are used to verify the integrity of all data. This was first applied to HDDs that were hashed before imaging. Once the image was complete, the new image was then hashed to confirm that it was an exact bit stream copy. The hash verified its integrity. At any point in the forensic process, the image could be rehashed to prove that it perfectly represented the original data as it was at the time of seizure.

With advancements in technology, things have begun to evolve. NAND flash memory and the use of mobile devices have caused a situation where it is now necessary to change data only slightly to be able to recover evidence. Every day criminals are convicted of crimes that depend on cellphone evidence to provide the smoking gun. But in nearly every case the extraction tools needed to recover the data must communicate with the operating phone. The phone needs to be charged, turned on, and just about ready to make a call. To have the phone operating means that it has changed data in the process. If companies like Cellebrite and Micro System Automation (XRY) could not change data, they would likely not be in business, and many crimes would go unsolved. It is the new norm to change data to allow an extraction to take place. Cellphones have an additional problem because they also use NAND flash memory that changes data behind the FTL, using garbage collection and wear levelling, as well as TRIM [2, 12].

SSDs contain NAND flash memory and behave in ways that have been compared to tampering when doing forensics on these devices [14]. It has been suggested that hashing these drives cannot confirm integrity since data changes over time [9].

Experiment 1 has found a very different conclusion. In every test of the drives being hashed found that over time the values did not change. This does not mean that the data behind the FTL was not changing, but just that what is presented to the OS does not change.

In Experiment 1, 144 images were made. These included three timed images, one immediately after the data was added and deleted, one after sitting idle for one hour, and a final completed eight hours after sitting idle. In each case, the zero hour, one hour, and eight-hour images all had their hashes match. This was opposite to what was discovered in Geier's paper. It should be noted that Geier's paper never mentioned whether they used a write blocker and that could easily explain why they saw hash changes [15].

Experiment 1 also found that data could be recovered after deletion on an SSD, however, it was not like what is expected from HDD. The amount of recovered data varied from model and manufacturer and was greatly reduced with the use of TRIM. This is opposite to what appears to be noted about TRIM by Joshi and Hubbard [14].

When data was added or deleted, it reduced the amount of recovered data from previous additions and deletions. It's changes being made to some data that may affect the recovery of other deleted data. This makes the use of forensic write blockers so important to help ensure that you recover the most evidence as well as ensuring the integrity of the original evidence. This is in opposition to some papers that comment that forensic write blockers are not suggested when imaging SSD drives [9].

Another finding in this experiment is the loss of recoverable data in SSDs where TRIM was disabled. Although much less recoverable data was available, significant

reductions in data still happened. With the deletion of files in HDDs only the File Allocation Table (FAT) is updated. In SSDs without TRIM, nothing is telling SSD to clear data with garbage collection. Some information was found that suggests that Samsung is using specialized algorithms that are capable of reading the `$BitMap` NTFS file. [2] This is a file in the NTFS file system that keeps track of the available clusters and is updated by the OS. Since the OS updates, this file with every deletion and this file is stored on the SSD; it only makes sense that the SSD controller can control the clearing of data using this file. The analysis in Experiment 1 shows that all manufacturers are using a similar design to keep up with the clearing of available data areas with or without TRIM. TRIM, however, seems to do a much better job, by leaving less recoverable data than the non-TRIM testing. It should be noted that TRIM is not needed with the advancements in newer SSD firmware. This suggestion was also mentioned in [9].

The results of this experiment are practically useful in providing support to any cases that might be challenged in court regarding the integrity of the evidence. Using a forensic write blocker ensures that any reduction of recoverable evidence was caused by the suspect and not the forensic examiner. The integrity of the evidence can be proven from this point forward by simply rehashing the image.

The analysis in Experiment 1 is more comprehensive than many other papers as it used 12 different models of drives that were imaged a total of 148 times to gather the needed evidence. These were new drives and contained no prior data, as well they were operated on the primary SATA channel with an OS installed on each drive. Experiment 1 tried to replicate a standard laptop or desktop seized during a search warrant.

An interesting finding happened with Drive 9. This drive would not boot after the first set of data was added and then imaged. After the boot failure the drive was wiped, and then the experiment was started again. In all cases, no data was recovered from the drive during first three additions of data and imaging. This was unusual to see this happen with just one drive. It is likely possible that the memory was empty when first purchased, but the wiping process added scrambled data to all areas of the memory when the wiping program used a zero's in the filling process. This would actually mean that the drive was full when the experiment took place and would likely have different results than any of the other drives. It is possible that some of the experiments in other papers that had opposing results to the ones in this paper may have used drives that had been wiped and used previously. None of the papers reviewed in this paper noted buying new unused drives. This is another reason why the results of this paper are comprehensive. This specific issue could also be the basis for further investigations regarding the issue between new drives and wiped drives.

Experiment 2 tested whether data was available for recovery after being filled with data and then formatted. The drives had data added so that they were within 1Gb of being full. The drive was imaged and then formatted. The drives were imaged immediately to create an after-format image. A subsequent image was again made after the drive sat idle and powered up for eight hours. In this case, the results were slightly mixed. Eight of the ten drives showed only the partition table and NTFS structure with the rest all being zeros. Drives 3 and 4 had nearly all the data recoverable. This shows that each of the drives have firmware that works differently. TRIM did not seem to affect any of the results in drive three and four. In every case, the eight-hour image

matched the image that was done immediately after format. This further confirmed the results found in Experiment 1 where the data did not change over time.

Experiment 3 tested if the SSD could be overwhelmed with garbage collection, and whether it could be seen somehow. It used a new Samsung 500 Gb SSDs and measured the time it took to copy 500 GB of data until it reported being full. The drive would then be formatted, and another 500 GB of data would immediately be written to it. The hypothesis is that the measurement of time on the second write should significantly increase if the drive now suddenly needs to make room for the new data using garbage collection. The first write took approximately 18 minutes using a secondary SATA channel. The second write after formatting took about the same time. In fact, it may have been slightly faster. The results of this experiment were inconclusive. It did not help determine if garbage collection was working behind the FTL, making room for data in the background.

Experiment 5 was performed to see if the action of garbage collection could somehow be seen by monitoring the power requirements. In this case, a SSD was connected to a separate power supply and attached in series to a meter that could measure the power consumption to 1/100th of a milliamp. The drive was filled with data and then deleted or formatted as in Experiments 3 and 4. The hypothesis is that the drive should draw a high-power level until garbage collection ends and the drive settle's down to an idle value. In all cases, after the data was deleted or the drive was formatted the drive drew between 250 and 300 milliamps initially then within 10-15 seconds settled to approximately 150 milliamps. This stayed constant and never reduced after two days. Again, this experiment was inconclusive in determining if garbage collection could somehow be observed happening behind the FTL.

5 Conclusion and Future work

The experiments in this paper have been successful in bringing forward an opposing view to some of the other papers [1, 15]. These experiments show that although things have changed in SSDs when compared to HDDs the way forensic examiners do forensics should stay practically the same. This comes with an understanding that NAND flash may hold deleted evidence that will not be shown to the OS. Currently, the standard practice is to image the SSD using a write blocker and the SATA/NVMe connection. In the future, forensic examiners will need to look at chip removal to bypass the FTL and be able to access all the possible deleted data. This comes with a cost where the labs will need to be able to remove the chips and correctly reconfigure the allocated data as well as recover additional deleted data. The experiments in this paper left many opportunities for further research regarding the data behind the flash translation layer and how to recover it for important cases.

References

1. B. Bell and R. Boddington, Solid State Drives: The Beginning of the End for Current Practice in Digital Forensic Recovery? , Perth: 2010
2. Guidelines on Mobile Device Forensics, from National Institute of Science and Technology, downloaded from http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-101r1.pdf on July 8, 2017
3. Sheremetov, Chip–Off Digital Forensics - Data Recovery After Deletion in Flash Memory, Myrtle Beach: Techno Security and Digital Forensics Conference, 2017
4. Y. Gubanov and O. Afonin, Recovering Evidence from SSD Drives: Understanding TRIM, Garbage Collection and Exclusions, Menlo Park, CA: Belkasoft 2014.
5. Garbage Collection in Single-Level Cell NAND Flash Memory, by Micron, https://www.micron.com/~/media/Documents/Products/Technical%2520Note/NAND%2520Flash/tn2960_garbage_collection_slc_nand.ashx+&cd=1&hl=en&ct=clnk&gl=ie&client=firefox-b
6. SLC, MLC or TLC NAND for Solid State Drives by Speed Guide.net downloaded from https://www.speedguide.net/faq/slc-mlc-or-tlc-Nand-for-solid-state-drives-406 on June 6, 2017.
7. NAND Bad Columns analysis and removal by ruSolute downloaded from http://rusolut.com/nand-bad-columns-analysis-and-removal/ on July 5, 2017
8. A. Nisbit, A Forensic Analysis and Comparison of Solid State Drive Data Retention With Trim Enabled File Systems, Auckland: Australian Digital Forensics Conference, 2013
9. P. Bednar and V. Katos, SSD: New Challenges for Digital Forensics,
10. Z. Shah, A. Mahmood, and J. Slay, Forensic Potentials of Solid State Drives, Canberra: 10th International Conference on Security and Privacy in Communication Networks, 2014
11. C. King and T. Vidas, Empirical Analysis of Solid State Disk Data Retention when used with Contemporary Operating Systems, New Orleans: The Digital Forensic Research Conference DFRWS 2011 USA, 2011
12. tn2919_nand_101 Nand Flash commands from Micron downloaded from https://www.micron.com/~/media/documents/products/.../tn2919_nand_101.pdf
13. Guidelines on Mobile Device Forensics, from National Institute of Science and Technology, downloaded from http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-101r1.pdf on July 8, 201
14. Binaya Raj Joshi and Rick Hubbard, Forensics Analysis of Solid State Drive (SSD), Omaha: Proceedings of 2016 Universal Technology Management Conference, 2016
15. F. Geier, The differences between SSD and HDD technology regarding forensic investigations, Sweden, 2015