Why SSD Drives Destroy Court Evidence, and What Can Be Done About It

Q3 2012: State of the art in SSD forensics

Yuri Gubanov yug@belkasoft.com, Oleg Afonin aoleg@voicecallcentral.com
Belkasoft Ltd. http://belkasoft.com

Abstract

Solid State drives (SSD) introduced dramatic changes to the principles of computer forensics. Forensic acquisition of computers equipped with SSD storage is very different of how we used to acquire PCs using traditional magnetic media. Instead of predictable and highly possible recovery of information the suspect attempted to destroy, we are entering the muddy waters of stochastic forensics where nothing can be assumed as a given.
Table of Contents
Why SSD Drives Destroy Court Evidence, and What Can Be Done About It .......................... 1
Q3 2012: State of the art in SSD forensics ........................................................................ 1
Table of Contents ................................................................................................................ 2
Stochastic Forensics ............................................................................................................. 2
Cannot Delete ..................................................................................................................... 3
Cannot Recover ................................................................................................................... 4
SSD Self-Corrosion ............................................................................................................... 4
TRIM: Myths and Reality .................................................................................................... 5
Exceptions ......................................................................................................................... 6
Encrypted Volumes ............................................................................................................... 7
Forensic Acquisition: The Right Way to Do ....................................................................... 7
Hardware for SSD Forensics ............................................................................................... 7
Custom Hardware: The Future of SSD Forensics? ............................................................... 8
Conclusion ............................................................................................................................ 9
About the Authors ............................................................................................................... 9
Literature .............................................................................................................................. 10

Stochastic Forensics
The way today’s SSD drives operate allows little space for positive assumptions. With SSD drives, the only thing we can assume is that an investigator can access existing information stored on the disk. Deleted files and data the suspect attempted to destroy (by e.g. formatting the disk – even in “Quick Format” mode) may be lost forever in a matter of minutes [1]. And even if the computer is powered off immediately after a destructive command has been issued (e.g. in a few minutes after the Quick Format), there is no easy way to prevent the disk from destroying the data once the power is back on. The situation is somewhat of a paradox, reminding of Schrödinger’s cat: one will never know if the cat is alive before opening the box [2].

Schrödinger’s cat, image from Wikipedia

The golden age of forensics is going to end. “Given the pace of development in SSD memory and controller technology, and the increasingly proliferation of manufacturers, drives, and firmware versions, it will probably never be possible to remove or narrow this new grey area
within the forensic and legal domain,” the scientists, from Australia's Murdoch University, wrote. “It seems possible that the golden age for forensic recovery and analysis of deleted data and deleted metadata may now be ending.” [1]

Cannot Delete
The way SSD drives are constructed imposes several design limitations. Existing types of flash memory allow for a limited number of write operations before wearing off. Modern SSD drives employ smart wear leveling techniques [3] that, instead of re-using existing blocks of memory, will write to a different block when data stored in a certain block is being modified. This in turn will leave blocks containing potentially sensitive information scattered all over the memory chip.

To further increase effective lifespan and improve wear leveling on SSD drives, many manufacturers install chips that can hold up to 25 percent more data than their advertised capacities [4]. This extra capacity is not addressable by means of the operating system, or by any other reasonable means (e.g. without using custom hardware to access the flash chips directly). This as well makes the content on SSD drives impossible to wipe as securely as required by some government and military standards via traditional means.

To mitigate this issue, some SSD manufacturers implemented an extension to the ATA ANSI specification to enable secure destruction of information stored on all flash chips [5]. The ATA Secure Erase (SE) command, when implemented correctly [4], wipes the entire contents of the drive at a hardware level.

In general, software secure wipe tools that would overwrite information stored on a hard drive with cryptographically secure random data in several passes. The problem with these software tools is their inability to address and, therefore, access the entire storage capacity of the SSD drive (including system, reserved and remapped areas).

As opposed to software-based tools, the ATA Secure Erase command instructs built-in SSD controller supporting the command to electronically erase all blocks on all flash chips of the drive. Effectively, erased SSD drives are cleaned completely, with all blocks being completely empty and available for immediate write (additional erase cycles will not be required before writing information to wiped blocks). Effectively, the SE command restores the SSD to factory defaults and write performance. When properly implemented [4] [13], the SE command will result in complete wipe of all storage regions of the SSD drive including any reserved, system and service areas.

An example of properly implemented secure erase is found in Intel self-encrypting SSD drives. According to Intel [13], "Executing a SECURE ERASE function, such as that found in the Intel® SSD Toolbox, will cause the Intel SSD 320 Series drives to generate a new internal encryption key." This will instantly render unusable all the encrypted user data stored on an Intel 320 Series SSD (and other devices supporting hardware-level full-disk encryption).
The inability to reliably recover erased information is another side of the same coin. The use of wear leveling will cause extensive use of the drive’s storage capacity, making use of previously unoccupied blocks of data at the time each write operation commences. Even repeat writes to the same file (e.g. the page file) will cause the entire content of the SSD drive to become “dirty”, leading to severe decrease in performance with write speeds being much slower than usual. This occurs because flash technology used in SSD drives requires blocks to be erased before the controller can perform a write operation on them. This property is unique to storage devices based on the flash technology, and is very different from how traditional magnetic types of media handle write requests.

As the process of erasing previously occupied blocks tends to be much slower compared to reading and writing, SSD drives full of “dirty” blocks will require significant time to write even a single block of data as no empty (erased) blocks exist. This lead SSD manufacturers to design a process performing garbage collection, erasing “dirty” blocks in background and making them available for fast write operations again.

The issue with garbage collection is that neither the drives nor their controllers know exactly which blocks are actually occupied by files or system structures of the operating system, and which blocks are no longer used and are just “dirty”. While the controller could mark blocks that were remapped to another blocks as a part of a wear leveling process, this information would only slow down the process of the drive being filled up with “dirty” blocks during normal use of the drive that typically involves creating, writing, modifying and deleting files.

In order to mitigate this issue, SSD designers developed an interface allowing the operating system (e.g. Windows, Linux, Mac OS X etc.) to inform the controller that certain blocks are no longer in use via the TRIM command [6]. This allows the internal garbage collector to electronically erase the content of these blocks, preparing them for future write operations.

Blocks of data processed by garbage collector are physically erased. Information from such blocks cannot be recovered even with the use of expensive custom hardware. Forensic researchers named this process as “self-corrosion” [7] [12].

Today’s SSDs self-destroy court evidence through the process that can be called “self corrosion”. Garbage collection running as a background process in most modern SSDs will permanently erase data marked for deletion, making it gone forever in a matter of minutes after the data has been marked for deletion. It is not possible to prevent garbage collection by moving the disk to another PC or attaching it to a write blocking device.
The only way to prevent self-corrosion is physically detaching the disk controller from flash memory chips storing the data, and then accessing the chips directly via custom hardware [see Hardware for SSD Forensics].

**TRIM: Myths and Reality**

A common misconception is that discarded blocks of an SSD drive are immediately erased. This is not usually the case. Instead, the way the TRIM command operates is considering the contents of discarded blocks as indeterminate (the "don't care" state) until the moment these blocks are physically erased by a separate background process, the garbage collector. In other words, the TRIM command does not erase the content of discarded blocks by itself. Instead, it adds them to a queue of pending blocks for being cleared by the garbage collector.

Exceptions

The “cannot recover” rule does not apply if the TRIM command has not been issued, or if TRIM is not supported by any link of the chain. If this is the case, information from SSD drives can be recovered in pretty much the same way as from a traditional hard drive [8][9].

The TRIM protocol will be disabled, or is not supported altogether, if at least one of the following conditions is met:

- **Old SSD drives**
  Older SSD drives do not support the TRIM command. For example, Intel started manufacturing TRIM-enabled SSD drives with drive lithography of 34nm (G2); their 50nm SSDs do not have TRIM support. [10]

- **Old versions of Windows**
  In Windows Vista and earlier versions, the TRIM protocol is not supported, and the TRIM command is not issued.
  **Possible exception:** TRIM-like performance can be enabled via certain third-party solutions (e.g. Intel SSD Optimizer, a part of Intel SSD Toolbox).

- **Old versions of MacOS X**
  Mac OS X started supporting the TRIM command for Apple supplied SSD drives since version 10.6.8. Older builds of Mac OS X do not support TRIM. In addition, user-installed SSD drives not supplied by Apple itself are excluded from TRIM support.

- **(Windows) File systems other than NTFS**
  At this time, only NTFS-formatted partitions receive full TRIM support in Windows. Volumes formatted with FAT, FAT32 or other file systems are excluded.

- **External drives, USB enclosures and Network Attached Storage**
  The TRIM command is fully supported over the SATA interface, including the eSATA extension, as well as SCSI via the UNMAP command. If an SSD drive is used in a USB enclosure or installed in certain types of NAS storage, the TRIM command will not be communicated via the unsupported interface.

- **PCI-Express SSDs**
  Interestingly, the TRIM command is not natively supported by any version of Windows for high-performance SSD drives occupying the PCI Express slot.
  **Possible exception:** TRIM-like performance can be enabled via certain third-party solutions (e.g. Intel SSD Optimizer, a part of Intel SSD Toolbox).

- **RAID**
  As of this writing, the TRIM command is generally not supported over RAID configurations (with few very rare exceptions) [10]. SSD drives working as part of a RAID array can be analyzed.

- **Logical corruption**
  Surprisingly, SSD drives with corrupted system areas (damaged partition tables, skewed file systems etc.) are easier to recover than healthy ones. The TRIM command is not issued over corrupted areas [11], because files are not properly deleted; they simply become invisible or inaccessible to the operating systems. Many commercially available data recovery tools (e.g. Intel® Solid-State Drive Toolbox with Intel® SSD Optimizer, OCZ SSD Toolbox) can reliably extract information from logically corrupted SSD drives.
• **Encrypted volumes**
  Somewhat counter-intuitively, information deleted from certain types of encrypted volumes (some configurations of BitLocker, TrueCrypt, PGP and other containers.) may be easier to recover as it may not be affected by the TRIM command. Files deleted from such encrypted volumes stored on an SSD drive can be recovered (unless they were specifically wiped by the user) if the investigator knows either the original password or binary decryption keys for the volume.

**Encrypted Volumes**

Encrypted volumes and SSD drives don’t play well together due to the wear leveling and performance issues described above. In many configurations, the crypto containers will encrypt the entire space on the drive, including free space. This turns every write on that disk into a re-write, which significantly slows down write performance on SSDs. The manufacturers of crypto containers recognized the issue and introduced ways (such us various configurations and advanced options) to mitigate the issue by releasing unused space back to the SSD controller, which in turn weakens overall security (as free unencrypted sectors are easy to tell).

If an encrypted volume of a fixed size is created, the default behavior is also to encrypt the entire content of a file representing the encrypted volume, which disables the effect of the TRIM command for the contents of the encrypted volume.

A dedicated research is required to investigate these options. At this time one thing is clear: in many configurations, including default ones, files deleted from encrypted volumes will not be affected by the TRIM command. Which brings us to the question of the correct acquisition of PCs with encrypted volumes.

**Forensic Acquisition: The Right Way to Do**

The right way to acquire a PC with a crypto container can be described with the following sentence: “If it’s running, don’t turn it off. If it’s off, don’t turn it on.” Indeed, the original decryption keys are cached in the computer’s memory, and can be extracted from a Live RAM dump obtained from a running computer by performing a FireWire attack. These keys can be also contained in page files and hibernation files. Tools such as Elcomsoft Forensic Disk Decryptor can extract decryption files from memory dumps and page/hibernation files, decrypting the content of encrypted volumes.

**Hardware for SSD Forensics**

At this time, most forensic researches involving the investigation of SSD drives are still performed on dedicated but still regular computers. SSD drives are either attached directly to the computer’s SATA interface or connected via a write blocking device of the same type that is also used to investigate magnetic hard drives. While write blockers do prevent user-induced modifications to the data stored on the SSD drive, they have nothing to do with the operation of the TRIM command and the disk’s internal garbage collector. It is essential to realize that an SSD drive connected via a write blocking device will continue performing background garbage collection, possibly destroying the last remnants of deleted information from the disk.
Preventing the operation of internal garbage collection is only possible by physically disconnecting the built-in controller from actual flash chips, and accessing information stored in the chips directly. At this time, this method is far from being popular as it requires special skills and custom hardware.


**Custom Hardware: The Future of SSD Forensics?**

By physically detaching the controller and using custom hardware to read information directly from the flash ships, investigators could extract traces of destroyed information that could be stored in various areas of the flash chips.

*Custom SSD recovery hardware [4]*
A group of scientists from University of California [4] designed an FPGA-based device providing direct access to flash chips of the SSD drive while bypassing the controller. The researchers estimated the cost of their prototype as $1000, while their estimate for building production units using microcontrollers instead of FPGA’s was as little as $200.

Is this the future of SSD forensics? While custom devices such as those built by Californian researchers may help forensic specialists extract some extra traces from certain SSD drives, other researchers suggest that most information is lost from an SSD drive in just a few counted minutes after the user deletes a file or issues a quick format command. The need to maintain custom hardware as well as the need of having specially trained staff for using this method will only make it justified for very few select cases.

Conclusion

SSD forensics is different. SSDs self-destroy court evidence, making it difficult to extract deleted files and destroyed information (e.g. from formatted disks) close to impossible. However, the correct acquisition technique may result in acquiring the original binary decryption keys, allowing investigators to access information stored in encrypted volumes, which may provide access to more information than available in unencrypted areas of SSD drives. In addition, numerous exceptions exist that effectively prevent mechanisms causing evidence self-corruption on SSD drives. Currently, SSD drives used in NAS devices, participating in RAID configurations, and connected as external devices via USB and FireWire are excepted from evidence self-corruption. Old versions of Windows, Mac OS and Linux do not support SSD’s garbage collection mechanisms, and are also exceptions.

The playfield is changing very fast. What’s true today may no longer apply tomorrow. We’ll keep an eye on what’s happening in the industry, releasing an updated report in a few months.

About the Authors

**Yuri Gubanov** is a renowned computer forensics expert. He is a frequent speaker at industry-known conferences such as EuroForensics, CEIC, China Forensic Conference, FT-Day, ICDDF, TechnoForensics and others. Yuri is the Founder and CEO of Belkasoft. Besides, Yuri is an author of f-interviews.com, a blog where he takes interviews with key persons in digital forensics and security domain.

You can reach Yuri Gubanov at [yug@belkasoft.com](mailto:yug@belkasoft.com) or add him to your LinkedIn network at [http://linkedin.com/in/yurigubanov](http://linkedin.com/in/yurigubanov).

**Oleg Afonin** is an independent expert and consultant in computer forensics. You can reach Oleg at [aoleg@voicecallcentral.com](mailto:aoleg@voicecallcentral.com).
Literature


