THE IMPACT OF MD5 FILE HASH COLLISIONS ON DIGITAL FORENSIC IMAGING

Gary C. Kessler
Embry-Riddle Aeronautical University
Daytona Beach, Florida
386-226-7947
gary.kessler@erau.edu

ABSTRACT

The Message Digest 5 (MD5) hash is commonly used as for integrity verification in the forensic imaging process. The ability to force MD5 hash collisions has been a reality for more than a decade, although there is a general consensus that hash collisions are of minimal impact to the practice of computer forensics. This paper describes an experiment to determine the results of imaging two disks that are identical except for one file, the two versions of which have different content but otherwise occupy the same byte positions on the disk, are the same size, and have the same hash value.

Keywords: MD5 hash collisions, forensic imaging, computer forensics, digital forensics

1. INTRODUCTION

The use of hash functions is widely used in the practice of digital forensics to ensure the integrity of files and the accuracy of forensic imaging. The Message Digest 5 (MD5) hash algorithm remains as one of the most commonly used hashes in digital forensics (Casey, 2011; Maras, 2015; Nelson, Phillips, & Steuart, 2015).

Hash collisions -- i.e., the occurrence where two files with different content have the same hash value -- have been identified in several well-known hash algorithms, in particular MD5 (McHugh, 2014; Wang, Feng, Lai, & Yu, 2004; Wang & Yu, 2005). Hashes are used for a variety of applications, including digital signature verification, computer forensic image verification, user identification and authentication, identifying known good or bad files in a hashset, and secure message exchange. The significance and meaning of a third-party being able to force hash collisions is different for these different applications; while forcing a hash collision in an authentication application could be quite serious, the impact might be less damaging when identifying files in a hashset (AccessData, 2006; Lewis, 2008; Thompson, 2005). Nevertheless, the use of hashing is so ingrained in digital forensics training and practice that the impact of such collisions in validating an evidentiary copy continues to be discussed at conferences and training sessions.

This paper will address the impact of MD5 hash collisions on validating the results of the computer forensics imaging process. Section 2 will identify the specific problem of hash collisions as it applies to imaging, followed by a restatement of the problem as a research question in Section 3. Section 4 will describe an experimental framework with which to test the research hypothesis, followed by test results in Section 5. Section 6 will offer some conclusions.

2. PROBLEM STATEMENT

The MD5 algorithm is described by Rivest (1992) and its use to validate forensic images is described in almost all computer forensics textbooks, including Casey (2011), Maras (2015), and Nelson et al. (2015). AccessData (2006) and Thompson (2005), among others, have suggested that MD5 hash collisions have minimal impact on the results of computer forensics examinations and, in practice, can be ignored. However, training in the computer forensics field for more than a quarter century has emphasized the importance of hashes as *the* key to proving the integrity of a digital forensic copy -- i.e., an image -- almost to the exclusion of the efficacy of training, experience, and the forensic imaging tools (Cohen, 2013).

One nightmare scenario for law enforcement, as a possible result of MD5 (or other) hash collisions, is this: A prosecutor introduces a set of N images of child sexual assault as evidence at trial, complete with the MD5 file hashes. The defense counters by producing a set of N images of the defendant on a dive boat in Aruba, complete with the same set of MD5 hashes. If this situation was possible, it can always be resolved, presumably, by viewing the original images on the evidentiary drive. However, the FUD (fear, uncertainty, and doubt) Factor has already been seeded and a good argument might then be made that could cause a jury -- or jurist -- to doubt the veracity and integrity of even the original evidence because the next obvious question is: If the first set of (innocent) images has the same hash values as the second set of (damning) images, could not the second set of images have been placed on the evidentiary disk by an over-zealous prosecutor or investigator?

It is well known that MD5 hash collisions exist, although they have largely been forced to occur in the laboratory (Burr, 2006; Gutman, Naccache, & Palmer, 2005; McHugh, 2014; Wang, Feng, Lai, & Yu, 2004; Wang & Yu, 2005). No one has yet reported hash collisions occurring in "nature;" that is, there are no reports of finding two different files on a given disk drive having the same MD5 hash. This is not surprising, given that there are 2^{128} (or ~10^{43}) possible MD5 hash values.

In digital forensics, we computer hash values not only on the individual files but also the entire disk that is being imaged. If we have two files, A and B, that have the same hash but are of different sizes, it is clear that the image hash will be different because there will be changes not only in the file content but also in other parts of the disk, such as allocated or unallocated space. Indeed, the file system metadata -- e.g., the file size in the directory entry as well as File Allocation Table (FAT), $Bitmap, or inode entries -- will also be different if the file sizes differ.

The impact is less obvious if files A and B are the same size because all of the file system metadata might be unchanged. Thus, is the nightmare scenario suggested above actually possible? This could theoretically only occur if one believes that the disk image hash remains the same if all of the files on the disk have the same hash. The experiment described in this paper addresses this question.

3. RESEARCH QUESTION

The scenario mentioned in Section 2 can be described as follows: Suppose one has two files, A and B, that have different content but are the same size and have the same MD5 hash value. What is the effect on the hash value of two disk images that differ only in that one disk contains File A and the other disk contains File B (where Files A and B occupy the same location on the two disk images)?

The research question is to test the following null hypothesis (H_0) as follows:

- The resultant two disk images will have the same hash value.

The alternative hypothesis (H_1) is as follows:

- The resultant two disk images will have different hash values.

4. EXPERIMENTAL SETUP

To address the research questions, two files were needed that were the same size, had the same MD5 hash, and had different content. Selinger (2011) provides such a pair of 128-byte files, called *hash1.bin* and *hash2.bin*, below:

```
hash1.bin
00000000: d131dd02c5e6eec4693d9a0698aff95c
00000010: 2fcab58712467eab4004583eb8fb7f89
00000020: 55ad340609f4b30283e488832571415a
00000030: 085125e8f7cdc99fd91dbdf280373c5b
00000040: d8823e3156348f5bae6dacd436c919c6
00000050: dd53e2b487da03fd02396306d248cda0
00000060: e99f33420f577ee8ce54b67080a80d1e
00000070: c69821bcb6a8839396f9652b6ff72a70

hash2.bin
00000000: d131dd02c5e6eec4693d9a0698aff95c
00000010: 2fcab50712467eab4004583eb8fb7f89
00000020: 55ad340609f4b30283e4888325f1415a
00000030: 085125e8f7cdc99fd91dbd7280373c5b
00000040: d8823e3156348f5bae6dacd436c919c6
00000050: dd53e23487da03fd02396306d248cda0
00000060: e99f33420f577ee8ce54b67080280d1e
00000070: c69821bcb6a8839396f965ab6ff72a70
```

The contents of the two files differ only by six bits, shown above in the six **bolded** nibbles. This is confirmed when executing the *fc* (file compare) command against the two files:

```
Comparing files hash1.bin and hash2.bin
00000013: 87 07         10000111 00000111
0000002D: 71 F1         01110001 11110001
0000003B: F2 72         11110010 01110010
00000053: B4 34         10110100 00110100
0000006D: A8 28         10101000 00101000
0000007B: 2B AB         00101011 10101011
```

While the two files have the same 128-bit MD5 hash, it is worth noting that their 160-bit Secure Hash Algorithm (SHA-1) values differ (Eastlake & Jones, 2001). This confirms that the contents of the two files are actually different and that there is a bona fide MD5 hash collision:

```
File: hash1.bin
MD5  9054025255FB1A26E4BC422AEF54EB4
SHA..A34473CF767C6108A5751A20971F1FDFBA97690A

File: hash2.bin
MD5  79054025255FB1A26E4BC422AEF54EB4
SHA  4283DD2D70AF1AD3C2D5FDC917330BF502035658
```

A 32 MB thumb drive was used as the test media. Using Windows 7, the thumb drive was formatted using the **format e: /v:HASHTEST /p:1** command. This initialized a FAT16 partition where the data area was overwritten with zeroes. The contents of the thumb drive were verified using the WinHex (v18.6) hex editor. Finally, a set of seven files were copied -- six arbitrary files plus *hash1.bin* -- to the thumb drive. The file list and hash values were:

```
File: 100_0230.JPG
MD5  097D23B541E4F58F03C57D410C3E3AD5
SHA  EB916AF75CB5B5BB145F7C11DF17FEC2B04B4395

File: Charts_Navigation.pdf
MD5  4942439FA574809EEAFFF72989FE4276
SHA  6DF61583B57FE4832AD5929E14AFA10638836FA9

File: diveboat.jpg
MD5  91700649FD62204C3675A045142424E8
SHA  B043E115E14C9EA3870D208526EEF300D4F4CCEC

File: hash1.bin
MD5  79054025255FB1A26E4BC422AEF54EB4
SHA  A34473CF767C6108A5751A20971F1FDFBA97690A

File: IMG_1425.JPG
MD5  CB8FE970560AA6184ED1BC2EEC887681
SHA  8A37616C53CD53B1281B32889A07E29EAC99B09B

File: in_5615551872.flv
MD5  27DE3209E3B68414A7429E4104C22185
SHA  40E6AD48C728C4FF916E354B962FBA4B5C7C77A6

File: PICT0131_GCK.JPG
MD5  A9ABC3E926F93A03D4844323B21C513D
SHA  C7FD4F3B8F743BF6202E6C57CC621A0EE6F5C6B5
```

5. TESTS AND RESULTS

Four tests were conducted on the media described above. The results described in this section are summarized in Table 1.

In Test #1, the thumb drive was imaged using FTK Imager (v3.1.3.2). The purpose of this test was merely to prepare a baseline disk image and set of hash values. The image verification MD5 hash of the thumb drive was **d1fdd4a0019fbedcd4459b51633ad9b8** and the complete FTK Imager report can be found in Appendix 1. The image was examined with FTK (v1.81.6) and the file listing showed the expected MD5 and SHA-1 hash values for the *hash1.bin* file (as shown in Section 4).

For Test #2, the thumb drive was mounted with WinHex and the contents of *hash1.bin* were copied over the location where *hash1.bin* resided on the thumb drive (128 bytes starting at offset 0x6149). The purpose of this test was to confirm that overwriting data in this way was possible and reliable. Note that it was not necessary to change anything else on the thumb drive since the two files were the same size; no changes were necessary to the FAT table entries or to the directory name, address, or file size. The thumb drive was then re-imaged. The image verification MD5 hash was **d1fdd4a0019fbedcd4459b51633ad9b8** -- the same as in Test #1. This result confirms that overwriting data in this way is an adequate process and changes nothing else on the drive. A portion of the FTK Imager report can be found in Appendix 2. The FTK file listing showed the expected MD5 and SHA-1 hash values for the *hash1.bin* file.

For Test #3, the thumb drive was mounted in WinHex and the contents of *hash2.bin* were copied over the location where *hash1.bin* resided on the thumb drive. This test was really the crux of the hypothesis experiment since *hash2.bin* is the "hash-equivalent, content-different" file to *hash1.bin*. The thumb drive was re-imaged, yielding an image verification MD5 hash of **8045e3c1d5a44eeb5297447b85ecada4** -- different than Tests #1 and #2. A portion of the FTK Imager report can be found in Appendix 3. The FTK file listing showed the expected MD5 and SHA-1 hash values for the *hash2.bin* file.

For Test #4, the thumb drive was mounted with WinHex and the contents of *hash1.bin* were copied back over the location where *hash2.bin* now resided on the thumb drive. The purpose of this test was to restore the drive to its original state and confirm that Test #3 changed nothing more than the 128 bytes where the test data resided. The fourth image verification MD5 hash was **d1fdd4a0019fbedcd4459b51633ad9b8** -- the same as Tests #1 and #2. This result confirms that Test #4 had restored the disk to its initial state and that Test #3 changed nothing more than the file data. A portion of the FTK Imager report can be found in Appendix 4. The FTK file listing showed the expected MD5 and SHA-1 hash values for the *hash1.bin* file.

6. CONCLUSIONS

The image verification MD5 hashes in Tests #1, #2, and #4 -- images that each held the *hash1.bin* content -- had the same value, whereas the image verification MD5 hash value in Test #3 -- when the image held the *hash2.bin* content -- was different from the other tests. The fact that Tests #1, #2, and #4 had the same hash proved that the test process worked as desired; the fact that Test #3 had a different result shows that the hash value of the imaged drive depends upon the actual bit content of the entire drive. Since the hash values of the two images are not the same, the null hypothesis (H_0) is disproven and the alternate hypothesis (H_1) is proven.

If the hash value of the disk were a function of the hashes of the individual components of the disk's contents, then one would expect to find the disk image unchanged when the files were substituted, meaning that the "nightmare scenario" could be realized. If the hash of the disk, however, were just based upon the bits on the disk, the two image hashes would be different when the files were exchanged, meaning that the scenario could not actually be perpetrated in this way.

Disproving the null hypothesis, then, is the expected result because the hash value of a disk image is supposed to be based upon the bit contents of the disk rather than the hashes of the individual files -- including file system structures and unallocated space -- that compose the disk contents. Thus, even if all of the file hashes on two disks are the same, the disk image hashes will be different if the contents of the files are different. Given this result, the scenario described in Section 2 cannot be realized.

It is hoped that this result will lay the concern about file hash collisions to rest as they apply to digital forensic imaging. As long as both individual files and the entire image are hashed, the theoretical occurrence of individual file collisions is not a factor in confirming the evidentiary integrity of a forensic copy.

As noted above, the SHA-1 hash values are different for the *hash1.bin* and *hash2.bin* files, although SHA-1 collisions are also theoretically possible (Stevens, Karpman, & Peyrin, 2015; Stevens et al., 2017). Since the MD5 and SHA-1 algorithms are different, the manipulation that can create an MD5 collision cannot create a SHA-1 collision and, to date, no one has yet shown a practical method with which to cause both an MD5 and SHA-1 collision in the same file. The results of the experiment reported in this paper, however, suggests that it would not matter since a file hash collision will still result in different image file hashes.

Table 1.
Summary of the four tests and the results.

Description of Test	**Image MD5 Hash Value**
#1 - Drive with *hash1.bin* file at bytes 0x6149-0x61C8	d1fdd4a0019fbedcd4459b51633ad9b8
#2 - Overwrite bytes 0x6149-0x61C8 with *hash1.bin*	d1fdd4a0019fbedcd4459b51633ad9b8
#3 - Overwrite bytes 0x6149-0x61C8 with *hash2.bin*	8045e3c1d5a44eeb5297447b85ecada4
#4 - Overwrite bytes 0x6149-0x61C8 with *hash1.bin*	d1fdd4a0019fbedcd4459b51633ad9b8

NOTE

All FTK Imager reports, FTK reports, and ancillary files are available for examination at http://www.garykessler.net/gck/hash_test.zip.

AUTHOR BIOGRAPHY

Gary C. Kessler, Ph.D., is a professor of cybersecurity and chair of the Security Studies & International Affairs Department at Embry-Riddle Aeronautical University in Daytona Beach, Florida. He is a Certified Computer Examiner (CCE), Certified Cyber Forensics Professional (CCFP), and Certified Information Systems Security Professional (CISSP), and a member of the Hawaii and North Florida Internet Crimes Against Children (ICAC) Task Force. Additional information can be found at http://www.garykessler.net.

REFERENCES

AccessData. (2006, April). *MD5 Collisions: The Effect on Computer Forensics.* AccessData White Paper. Retrieved from https://ad-pdf.s3.amazonaws.com/papers/wp.MD5_Collisions.en_us.pdf

Burr, W. (2006, March/April). Cryptographic hash standards: Where do we go from here? *IEEE Security & Privacy*, *4*(2), 88-91. Retrieved from http://www.csee.wvu.edu/%7Ekaterina/Teaching/CS-465-Spring-2007/HashStandards.pdf

Casey, E. (2011). *Digital Evidence and Computer Crime*, 3rd ed. Amsterdam: Elsevier.

Cohen, F. (2013). *Digital Forensic Evidence Examination*, 5th ed. Livermore (CA): Fred Cohen & Associates. Retrieved from http://all.net/books/2013-DFE-Examination.pdf

Eastlake, D., 3rd, & Jones, P. (2001, September). US Secure Hash Algorithm 1 (SHA1). Requests for Comments (RFC) 3174. Retrieved from https://www.rfc-editor.org/rfc/rfc3174.txt

Gutman, P., Naccache, D., & Palmer, C.C. (2005, May/June). When hashes collide. *IEEE Security & Privacy*, *3*(3), 68-71. Retrieved from https://researchspace.auckland.ac.nz/bitstream/handle /2292/269/269.pdf

Lewis, D.L. (2008, December 1). The Hash Algorithm Dilemma -- Hash Value Collisions. *Forensic Magazine*. Retrieved from http://www.forensicmag.com/article/2008/12/hash-algorithm-dilemma%E2%80%93hash-value-collisions

Maras, M. H. (2015). *Computer Forensics: Cybercriminals, Laws, and Evidence*, 2nd ed. Burlington, MA: Jones & Bartlett Learning.

McHugh, N. (2014, October 31). How I created two images with the same MD5 hash. Retrieved from http://natmchugh.blogspot.com/2014/10/how-i-created-two-images-with-same-md5.html

Nelson, B., Phillips, A., & Steuart, C. (2015). *Guide to Computer Forensics and Investigations*, 5th ed. Boston: Course Technology.

Rivest, R. (1992, April). The MD5 Message Digest Algorithm. Request for Comments (RFC) 1321. Retrieved from https://www.rfc-editor.org/rfc/rfc1321.txt

Selinger, P. (2011, October 11). MD5 Collision Demo. Retrieved from http://www.mathstat.dal.ca /~selinger/md5collision/

Stevens, M., Bursztein, E., Karpman, P., Albertini, A., Markov, Y., Bianco, A.P., & Biasse, C. (2017, February 23). Announcing the first SHA1 collision. *Google Security Blog.* Retrieved from https://security.googleblog.com/2017/02/ announcing-first-sha1-collision.html

Stevens, M., Karpman, P., & Peyrin, T. (2015, October 8). Freestart collision on full SHA-1. Cryptology ePrint Archive, Report 2015/967. Retrieved from https://eprint.iacr.org/2015/967.pdf

Thompson, E. (2005, February). MD5 collisions and the impact on computer forensics. *Digital Investigation*, *2*(1), 36-40.

Wang, X., Feng, D., Lai, X., & Yu, H. (2004, August). Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD. Retrieved from http://eprint.iacr.org/2004/199.pdf

Wang, X.Y. & Yu, H.B. (2005, May). How to break MD5 and other hash functions. *Advances in Cryptology–Eurocrypt'05*, pp. 19-35. Retrieved from *merlot.usc.edu/csac-f06/papers /Wang05a.pdf*

APPENDICES

Appendix 1: FTK Imager report for Test #1

```
Created By AccessData® FTK® Imager 3.1.3.2

Case Information:
Acquired using: ADI3.1.3.2
Case Number: Hash Test
Evidence Number: 1
Unique Description:
Examiner: GCK
Notes: hash1.bin

--------------------------------------------------------------

Information for C:\Users\gck\Desktop\hash_test\Test1:

Physical Evidentiary Item (Source) Information:
[Device Info]
 Source Type: Physical
[Drive Geometry]
 Cylinders: 3
 Tracks per Cylinder: 255
 Sectors per Track: 63
 Bytes per Sector: 512
 Sector Count: 62,719
[Physical Drive Information]
 Drive Model: SanDisk Cruzer Mini USB Device
 Drive Serial Number:
 Drive Interface Type: USB
 Removable drive: True
 Source data size: 30 MB
 Sector count:    62719
[Computed Hashes]
 MD5 checksum:    d1fdd4a0019fbedcd4459b51633ad9b8
 SHA1 checksum:   169d0f1972364d65760f17fc49838cc27ba378f1

Image Information:
 Acquisition started:   Mon Jun 20 19:37:47 2016
 Acquisition finished:  Mon Jun 20 19:37:52 2016
 Segment list:
  C:\Users\gck\Desktop\hash_test\Test1.E01

Image Verification Results:
 Verification started:  Mon Jun 20 19:37:52 2016
 Verification finished: Mon Jun 20 19:37:52 2016
 MD5 checksum:    d1fdd4a0019fbedcd4459b51633ad9b8 : verified
 SHA1 checksum:   169d0f1972364d65760f17fc49838cc27ba378f1 : verified
```

Appendix 2: FTK Imager report (partial) for Test #2

```
Created By AccessData® FTK® Imager 3.1.3.2

Case Number: Hash Test 2
Evidence Number: 2
Examiner: GCK
Notes: hash1.bin (overwritten)

--------------------------------------------------------------

Information for C:\Users\gck\Desktop\hash_test\Test2:

[Computed Hashes]
 MD5 checksum:    d1fdd4a0019fbedcd4459b51633ad9b8
 SHA1 checksum:   169d0f1972364d65760f17fc49838cc27ba378f1

Image Information:
 Acquisition started:   Mon Jun 20 19:40:24 2016
 Acquisition finished:  Mon Jun 20 19:40:29 2016
 Segment list:
  C:\Users\gck\Desktop\hash_test\Test2.E01

Image Verification Results:
 Verification started:  Mon Jun 20 19:40:29 2016
 Verification finished: Mon Jun 20 19:40:29 2016
 MD5 checksum:    d1fdd4a0019fbedcd4459b51633ad9b8 : verified
 SHA1 checksum:   169d0f1972364d65760f17fc49838cc27ba378f1 : verified
```

Appendix 3: FTK Imager report (partial) for Test #3

```
Created By AccessData® FTK® Imager 3.1.3.2

Case Number: Hash Test
Evidence Number: 3
Examiner: GCK
Notes: hash2.bin overwrite

--------------------------------------------------------------

Information for C:\Users\gck\Desktop\hash_test\Test3:

[Computed Hashes]
 MD5 checksum:    8045e3c1d5a44eeb5297447b85ecada4
 SHA1 checksum:   177774eefa63b5e67c04a2e9d2d875e2353400df

Image Information:
 Acquisition started:   Mon Jun 20 19:43:14 2016
 Acquisition finished:  Mon Jun 20 19:43:18 2016
 Segment list:
  C:\Users\gck\Desktop\hash_test\Test3.E01

Image Verification Results:
 Verification started:  Mon Jun 20 19:43:18 2016
 Verification finished: Mon Jun 20 19:43:18 2016
 MD5 checksum:    8045e3c1d5a44eeb5297447b85ecada4 : verified
 SHA1 checksum:   177774eefa63b5e67c04a2e9d2d875e2353400df : verified
```

Appendix 4: FTK Imager report (partial) for Test #4

```
Created By AccessData® FTK® Imager 3.1.3.2

Case Number: Hash Test
Evidence Number: 4
Examiner: GCK
Notes: hash1.bin re-written

--------------------------------------------------------------

Information for C:\Users\gck\Desktop\hash_test\Test4:

[Computed Hashes]
 MD5 checksum:    d1fdd4a0019fbedcd4459b51633ad9b8
 SHA1 checksum:   169d0f1972364d65760f17fc49838cc27ba378f1

Image Information:
 Acquisition started:   Mon Jun 20 19:45:52 2016
 Acquisition finished:  Mon Jun 20 19:45:57 2016
 Segment list:
  C:\Users\gck\Desktop\hash_test\Test4.E01

Image Verification Results:
 Verification started:  Mon Jun 20 19:45:57 2016
 Verification finished: Mon Jun 20 19:45:57 2016
```

```
MD5 checksum:    d1fdd4a0019fbedcd4459b51633ad9b8 : verified
SHA1 checksum:   169d0f1972364d65760f17fc49838cc27ba378f1 : verified
```