MD5 HASHING IN DIGITAL FORENSICS

James Buncle

University of Birmingham
ug67jxb@cs.bham.ac.uk

ABSTRACT

This report gives an overview of the MD5 cryptographic hash function and its key uses in digital forensics. It also introduces Fuzzy Hashing for identifying similar files and its potential uses in digital forensics.
Background research has been completed and various papers have been studied in detail, including both those specific to MD5 in digital forensics and those that are related to it. Summaries have been produced for the key current practises and the reasoning behind their use. Suggestions of current and future implications and usages for MD5 have also been made, and how Fuzzy Hashing is introduced as a result of some of these issues.

Keywords: Digital Forensics, MD5, Fuzzy Hashing

1. INTRODUCTION

Message-Digest Algorithm 5 (MD5) Hashing is the process of transforming a stream of data into a shorter cryptographic hash key, which is calculated using the MD5 hash function. The data stream can be of any length and is normally referred to as the 'message'. The hash key produced is known as the message digest or just 'digest', and is of a fixed length. The produced digest is considered as the 'digital fingerprint' of its data stream.[5][7][11]

More specifically the MD5 digest is 128 bits long and is expressed as a 32 digit hexadecimal number, therefore there are 2^{128} ($3.4x10^{38}$) possible unique MD5 hashes[3][8].

Illustration 1: MD5 Function Process

The main quality of the MD5 hash function is that it is relatively collision resistant, meaning that it is very unlikely for two different messages to have the same MD5 digest. [3][11]
As with all hash functions, the same file will always produce the same hash (as long as it has not been modified), and it is very difficult to reverse and produce the original file from a given hash.

The MD5 hash function has a wide range of applications, such as for storing passwords in computer security. It can also be used for verifying data integrity, where a precomputed MD5 of an original file is used to validate a copy. Similar techniques are also used in digital forensics.

2. MD5 AND DISK IMAGING

One of the applications of MD5 Hashing in digital forensics is for error detection during the duplication of disk drives, files and any other stream of data[6]. The following method is used in digital forensics to ensure data integrity and to verify that a copy of evidence is digitally identical to the original:

1. The original data stream is hashed using the MD5 hash function, producing a unique MD5 hash for that specific stream of bytes
2. The data stream is then copied, creating a supposedly bitwise image
3. The image created is then also hashed
4. The original data streams hash is then compared to the images hash

This is a very reliable method for verification as a single bit difference in the image will produce a completely different hash to the input stream [13]. Therefore a match will strongly suggest that the image is an exact bitwise copy of the original. It is very unlikely that any errors would still produce an identical hash.[21][1][2]

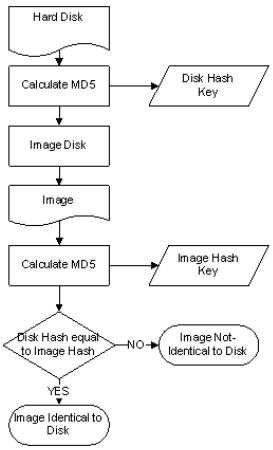

Illustration 2: Use of MD5 when Disk Imaging

The original drives hash can also be used further to ensure that no modifications to the device or image have been made throughout the forensic investigation. The device can be re-hashed after the examination has been completed, and compared to the original. If there is a match this would confirm that no changes had been made.

These are relatively simple and straightforward techniques, however if the image produces a different hash, which would indicate an error during the imaging process, the whole drive will be need to be re-imaged. An improvement on this would be to copy a disk sector by sector and verify these sections independently, for example by using block-based or piecewise hashing. This way, if an error is produced, it will be for a specific sector, and then only the data integrity of that part would be threatened.

A further improvement would be to first rehash the device and the image if a difference between hashes is found. The new hashes would then be compared again in case the error was thrown during the hashing process, reducing any unnecessary recopying of the disk.

3. MD5 FOR FILE SYSTEM ANALYSIS

Another application for MD5 hashing is for the reliable identification of files and data reduction during file system analysis [14][6].

Due to the increasing capacity of digital electronic storage devices, there can be a vast amount of data to analyse during a forensic examination. To examine every file on a hard drive would be a lengthy and time consuming task. Therefore MD5 can be used for both reducing the amount of data needing to be examined, and also identifying and flagging files for inspection automatically.

As the MD5 hash function turns files of any size into single 128bit hash keys, it can be used to reduce the files on a disk to a set of fixed length strings that uniquely identify them. This results in a much smaller representation of a disks file system, therefore making it significantly quicker to examine.

The generated set of hash keys can then be used for the identification and therefore omission of irrelevant files. Each hash can be compared to a set of precomputed hashes for known files, such as Operating System files. When a match is found the hash is omitted, further reducing the amount of data left to examine. When all irrelevant files are removed, a much smaller set of hashes will remain. Due to MD5s collision resistance, this is a reliable and fast method for data reduction.

The same technique used for omitting files can also then be used for the identification of incriminating files. Instead of comparing the hashes to a set of irrelevant file hashes, they are compared to a set of known illegal or incriminating files. The files identified as incriminating can then be flagged for further inspection. Normally, the checking for both irrelevant and incriminating files would occur during the same process.[6][9]

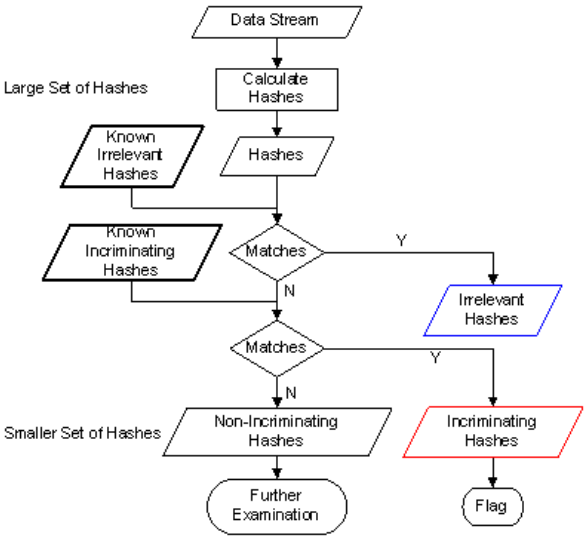

Illustration 3: MD5 for File System Analysis

MD5 is not always appropriate, for example during a process known as file carving where files are examined without using the files meta-data[19].

Theoretically, in the future this technique may become unreliable as increasingly more files can be found on a single storage device. If it comes to a

point where there are more files on a drive than there are possible hashes, for example if the drive has $2^{128}+1$ unique files, then collisions will begin to occur, as there will be at least two files with the same hash. This would result in the possibility of incriminating files being accidentally omitted and vice versa. However, this would only have a considerable effect if there are significantly more than 2^{128} files.

Another problem, although currently not possible[3], may be the development of programs that will cause a given file to have the same hash as another even though they are different, for example by adding certain data to the end of the file. This could then be used by a suspect to 'disguise' an incriminating file as one that would normally be discarded, such as a system file. A solution to this could be the use of cryptographic salting, where data is inserted into the message before hashing. Due to the nature of MD5 having a different hash from small changes, applying the same salt function to the files being compared, will cause a 'disguised' file to have a different hash, but identical files will still have the same.

4. FUZZY HASHING

One of the issues with using MD5 hashing to identify incriminating files on a suspects device is that the suspect is able to avoid a file being detected by changing or adding a single bit in the file, this will have a cascading effect which would cause the file to produce a completely different hash[3]. One solution to this is to compare files for similarities, instead of just whether or not they are digitally identical[6].

Finding the similarity between files has been studied for decades[6]. However, more recently a new methodology has been introduced to overcome this issue, and is used for the identification of homologous (similar) files.
This method is known as Context Triggered Piecewise (CTP) Hashing, or “Fuzzy Hashing”[4] [17], and is a combination of the Rolling hash and Piecewise hashing[12].

Piecewise hashing (also known as Block-based hashing) involves dividing a data stream into chunks and hashing each part separately [12]. Then, to discover the similarity of two files, the data streams hashes can be compared with each other and a count of matches taken. [13][14][15]

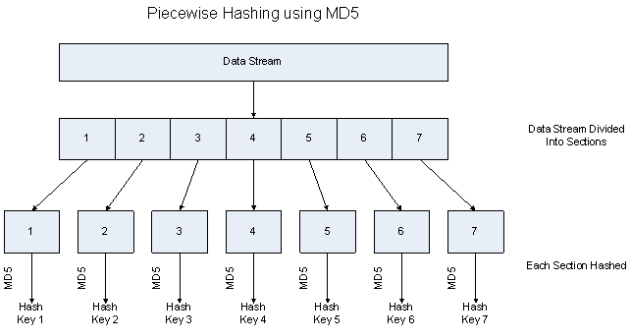

Illustration 4: Piecewise Hashing

However, the problem with this is that it will only identify files that have had sections replaced by data of equal length and not where data has been inserted or deleted. This would cause a shift in the data stream, producing different hashes. The quality of being able to cope with the shifting problem is known as Alignment robustness. [17]

To overcome this issue, Fuzzy Hashing uses another hash to identify where each section for the piecewise hash begins and ends, namely the rolling hash[19][5]. This is done by setting reset points or trigger values in a file. A simpler example of this would be to set a specific character as a trigger value when comparing strings. Once the trigger value has been set, the data before and after each value is hashed, resulting in a set of hashes. This should ensure that the 'same' sections in homologous files are hashed. [14]
The other option would be to produce a set of possible hashes created by calculating the hashes at various shifts, however this would consume resources unnecessarily.
The CTP hash consists of combining the individual hashes produced and separating them by a semicolon. [13]
The hash can then be used to compare and rank a set of files based on the similarity to another, by taking a frequency count of the number of matches in the hashes.
The files with a relatively high similarity can then be flagged for further examination. [13][17][19]

This technique could be improved by recursively checking files over a certain similarity with a more thorough CTP hash function, for example by rehashing a flagged file using a higher number of trigger values that section the file for hashing. This would increase the number of hashes available to compare and check for similarity, if the file continues to show high similarity it can remain flagged for inspection, or otherwise omitted.

As Fuzzy Hashing sections the data input when the function is applied, it could also be used on a drive as a whole, instead of individual files. This way sections on the drive can be highlighted without having to hash individual files. This would provide a single but long CTP hash to use for comparison. However, this would only be effective if the hash function was detailed enough. Also, if used alone, would not reduce the amount of non-flagged data that remains to be examined manually.

5. CONCLUSION

The MD5 hash function will always be able to play a part in digital forensics, although not necessarily in the same way. It can always be used for ensuring data integrity when copying disks, because the security of the function has no implications in this method. It is likely that in the future the MD5 function will be used in a piecewise manner or in Fuzzy Hashing for the identification of incriminating files.

As with most hashes, the MD5 hash function may eventually become 'broken', and its reliability reduced as a result. However there are techniques that can be used to overcome this, such as salting which was mentioned earlier.

MD5 may be replaced in the future by a more improved version, such as one that is optimised and can perform faster and more efficiently, however there is little need for this as the hardware implementing the function is more likely to be improved first.

REFERENCES

[1] Mamoun Alazab, Sitalakshmi Venkatraman, Paul Watters, "*Effective Digital Forensic Analysis of the NTFS Disk Image*", University of Ballarat, 2009 (http://www.ubicc.org/files/pdf/3_371.pdf)

[2] Ming Hu; Yan Wang, "*MD5-Based Error Detection*" PACCS '09. Pacific-Asia Conference on , pp.187-190, 16-17 May 2009, (http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5232318&isnumber=5231959)

[3]AccessData Corp, "*MD5 Collisions: The Effect on Computer Forensics*", April 2006, (http://www.accessdata.com/media/en_us/print/papers/wp.md5_collisions.en_us.pdf)

[4] Creative Commons, "*Context Triggered Piecewise Hashing*", 2007, (http://www.forensicswiki.org/wiki/Context_Triggered_Piecewise_Hashing)

[5] Microsoft Corporation, "*Remote Differential Compression Algorithm Specification*", 2009 (http://msdn.microsoft.com/en-us/library/dd357722%28PROT.13%29.aspx)

[6] Vassil Roussev, Yixin Chen, Timothy Bourg, Golden G. Richard III, "*md5bloom: Forensic filesystem hashing revisited*, *Digital Investigation*", Volume 3, Supplement 1, The Proceedings of the 6th Annual Digital Forensic Research Workshop (DFRWS '06), September 2006, Pages 82-90, ISSN 1742-2876, DOI: 10.1016/j.diin.2006.06.012.

(http://www.sciencedirect.com/science/article/B7CW4-4KCXJJ6-5/2/70726ca488e87b615f649db7ee3a15e5)

[7] Jarvinen, K.; Tommiska, M.; Skytta, J., "*Hardware Implementation Analysis of the MD5 Hash Algorithm*" System Sciences, 2005. HICSS '05. Proceedings of the 38th Annual Hawaii International Conference on , pp. 298a-298a, 03-06 Jan. 2005 (http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1385853&isnumber=30166)

[8] "MD5", (http://en.wikipedia.org/wiki/Md5)

[9] Shira Danker Rick Ayers Richard P. Mislan, "*Hashing Techniques for Mobile Device Forensics*" ALL SCALE DIGITAL DEVICE FORENSICS JOURNAL, VOL. 3, NO. 1, JUNE 2009 ISSN# 1941-6164 1 (http://www.ssddfj.org/papers/SSDDFJ_V3_1_Dankner_Ayers_Mislan.pdf)

[10]William Josephson, "Hashing and Fingerprinting", Princeton University, 2005, (http://www.cs.princeton.edu/courses/archive/spr05/cos598E/bib/William.pdf)

[11] Singh, M.; Garg, D., "*Choosing Best Hashing Strategies and Hash Functions*", Advance Computing Conference, 2009. IACC 2009. IEEE International , pp.50-55, 6-7 March 2009

[12] Long Chen, Guoyin Wang, "*An Efficient Piecewise Hashing Method for Computer Forensics*", International Workshop on Knowledge Discovery and Data Mining, pp. 635-638, First International Workshop on Knowledge Discovery and Data Mining (WKDD 2008), 2008.

[13] Dustin Hurlbut, "*Fuzzy Hashing for Digital Forensic Investigators*", January 2009, (http://www.accessdata.com/downloads/media/Fuzzy_Hashing_for_Investigators.pdf)

[14] Jesse Kornblum, "*Identifying almost identical files using context triggered piecewise hashing*", Digital Investigation, Volume 3, Supplement 1, The Proceedings of the 6th Annual Digital Forensic Research Workshop (DFRWS '06), September 2006, Pages 91-97, ISSN 1742-2876, DOI: 10.1016/j.diin.2006.06.015.

(http://www.sciencedirect.com/science/article/B7CW4-4KCPVBY-8/2/2c551a44e599be80e791d28d74f458c9)

[15] Vassil Roussev, Golden G. Richard III, Lodovico Marziale, "*Multi-resolution similarity hashing*, *Digital Investigation*", Volume 4, Supplement 1, September 2007, Pages 105-113, ISSN 1742-2876, DOI: 10.1016/j.diin.2007.06.011.

(http://www.sciencedirect.com/science/article/B7CW4-4P06CJD-7/2/17011376c1aee1cef8c99e728e67494f)

[16]Creative Commons, "*Piecewise Hashing*", 2007, (http://www.forensicswiki.org/wiki/Piecewise_hashing)

[17] DigitalNinja,"'*Fuzzy Clarity' Using Fuzzy Hashing Techniques to Identify Malicious Code*", 2007, (http://digitalninjitsu.com/downloads/Fuzzy_Clarity_rev1.pdf)

[18] Andrew Tridgel, "*Spamsum Readme*", October 2009, (http://samba.org/ftp/unpacked/junkcode/spamsum/README)

[19] Jesse Kornblum, "*Fuzzy Hashing*", ManTech SMA, (http://jessekornblum.com/presentations/htcia06.pdf)

[20] Xiaoyun Wang and Hongbo Yu, "*How to Break MD5 and Other Hash Functions*" Shandong University

[21] Wick, C.; Avramov-Zamurovic, S.; Lyle, J., "*Hard disk interface used in computer forensic science*", Instrumentation and Measurement Technology Conference, *2004. IMTC 04.* Proceedings of the 21st IEEE , vol.3, pp. 1780-1783 Vol.3, 18-20 May 2004 (http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1351427&isnumber=29628)

[22] Eric Thompson, "*MD5 collisions and the impact on computer forensics*", Digital Investigation, Volume 2, Issue 1, February 2005, Pages 36-40, ISSN 1742-2876, DOI: 10.1016/j.diin.2005.01.004.

(http://www.sciencedirect.com/science/article/B7CW4-4FM544N-1/2/ef0d68006096a3f12959a02436506beb)

[25] Polytechnic University, "Digital Forensics", (http://isis.poly.edu/courses/cs996-forensics/Lectures/forensics_module11.ppt)

[26] Digital Forensics BJ Gleason

[27] Wilsdon, T.; Slay, J., "*Digital forensics: exploring validation, verification & certification*", Systematic Approaches to Digital Forensic Engineering, *2005. First International Workshop on*, vol.,no., pp. 48-55, 7-9 Nov. 2005 (http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1592521&isnumber=33521)

[28] J. Black, M. Cochran, T. Highland "A Study of the MD5 Attacks: Insights and Improvments" March 2006, (http://www.cs.colorado.edu/~jrblack/papers/md5e-full.pdf)