



Netherlands Forensic Institute
Ministry of Justice and Security

Technical Supplement Forensic Use of Hash Values and Associated Hash Algorithms

Table of Contents

1. Technical Supplement in General
2. Introduction
3. Hash Values
4. One-way Hash Algorithms
5. Forensic Applications
 - 5.1. Integrity Check
 - 5.2. File Identification and Classification
6. Hash Values in Practice
7. Hash Algorithms Used by DBS
 - 7.1. Current Status of Hash Algorithms
 - 7.2. Current Choice of Hash Algorithms at DBS
 - 7.3. Chance in Practice of Another File with the Same Hash Value
8. Glossary
9. Bibliography

1. Technical Supplement in General

The Netherlands Forensic Institute (NFI) conducts many types of examinations. Usually every expert witness report will be accompanied by a technical supplement. This supplement serves as an explanation of the investigation and has a purely informative character - it does not contain any case-specific information. At the end of this technical supplement, a glossary and a bibliography have been included.

2. Introduction

Various teams¹ of the Digital and Biometric Traces Division (DBS in Dutch) of the NFI receive and analyse mainly digital material. The results that these teams deliver also consist mainly of digital data. To ensure the integrity of the contents of the digital material, DBS uses *hash values*, which may also be used for the classification and identification of files.

¹ In any case, the Forensic Digital Technology team, Forensics Big Data Analytics team and the Imaging group of the Forensic Biometrics team. Hereafter where DBS is mentioned, these specific teams are intended.

Hash values are calculated over the content of the digital material by using programmes that use specifically designed arithmetic methods called *hash algorithms*.

The terms *hash* and *hash value* are interchangeable. Furthermore, the hash value of a file is sometimes referred to as its 'digital fingerprint'.²

Reports from expert witnesses of DBS often contain the hash values of received digital material as well as that of the digital results of an examination. This technical supplement discusses hash values in more detail and explains its main forensic applications. The use of hash values in practice is also discussed. Finally, the technical choices made by the DBS Division of the NFI for the uses of hash values are described.

3. Hash Values

A hash value is a condensed representation of the binary (digitized) content of digital material, but does not provide further information about the content of the material that can be *interpreted* by a person³. Where, for example, a person can see the content of two images as the same (contain the same interpretable content), the files may differ binarily (for example, when the storage size of the images differ). Henceforth in this technical statement, the content of a file refers to the binary content, unless otherwise stated.

A digital file consists of a series of zeros and ones (*bits*). The number and position of these zeros and ones determine the content of the file and therefore also the hash value of the file. By using a hash algorithm, the specific zeros and ones of a particular file are converted into a much simpler and much more condensed notation. A hash algorithm is a finite (and usually complex) set of (mathematical) instructions that generate a sequence of zeros and ones of a set length based on the contents of the original file. This generated sequence is characteristic of the contents of the original file and is referred to as the "hash value" of the original file. Although this technical supplement covers mostly (digital) files, the (forensic) use of hash

values applies to all digital material such as, for example, a full data copy or *image* of a digital data carrier.

As previously stated, one must be aware that two files can look the same to a user, but when the files differ at bit level, they will have (with an extremely high probability) different hash values.

Among other things, in order to increase readability, DBS hash values do not appear in ones and zeros but rather denoted in the so-called sixteen-digit (hexadecimal) notation. In this notation, four consecutive zeros and ones (bits) are denoted by one of the following characters: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f (see Table 1). There is no distinction made between capital and small letters.

Bits	Hexadecimal	Bits	Hexadecimal
0000	0	1000	8
0001	1	1001	9
0010	2	1010	a
0011	3	1011	b
0100	4	1100	c
0101	5	1101	d
0110	6	1110	e
0111	7	1111	f

Table 1: from bits to hexadecimal notation

To further enhance readability, the characters are often displayed grouped in sequences of four characters, with each character representing 4 bits. An example of a hash value with a length of 256 bits (64 characters) looks like the following:

```
2584 878d 8694 6001 3fab 045c 4de2 03d1  
ca8a bb44 3b6b 2169 c9a1 4021 8d2b 21bb
```

Each two characters together (therefore 8 bits), are called a byte. The length of the hash value is generally indicated in the number of bytes. The 64-character listed here corresponds to 32 bytes.

² The comparison between a hash value and a fingerprint applies only to the notion of their respective distinctive value. Fingerprints containing similar attributes are not distinctive. It is extremely likely that almost identical hash values belong to files with completely different content and are therefore very distinctive.

³ Examples of files interpretable by a person are digital text documents and images (visual) and digital audio files (auditory).

4. One-way Hash Algorithms

When using (good) hash algorithms, a change of only one bit in a file will, with extremely high probability, result in a completely different hash value than that of the file before the change. This is referred to as the *avalanche effect*. In order to illustrate this, see Table 2 below: the hash values (in this case MD5, see also section 7) calculated over almost identical bit sequences are totally different.

Bit sequences	MD5-hash values
1000000000000000	fc60 9b43 cb85 95a9 a832 cbc2 591e d83a
1000000000000001	9d26 f82a f654 8210 454c 017a 3179 c0ec
1000000010000000	688e c893 e933 aff7 2480 5d61 4e43 f68e

Table 2: MD5-hash values of almost identical bit sequences

This technical supplement is limited to a specific category of hash algorithms and associated hash values, the so-called *cryptographic* or *one-way hash algorithms*.

Wherever the term hash algorithm is used in this document, it always concerns a one-way hash algorithm. Apart from the avalanche effect, these algorithms have at least the following properties:

1. *One-way or preimage resistant*: it is practically impossible⁴ to find or create a file that corresponds to a specified hash value.
2. *Target collision resistant* or *second preimage resistant*: it is practically impossible to find or create a file with different content but with the same hash value as a specified file.
3. *Random collision resistant*: it is practically impossible to find two files that have different content but have the same hash value.

These described properties of a one-way hash algorithm are assumptions deemed reasonable until evidence to the contrary exists. At present there is no mathematical proof that true one-way hash algorithms exist at all. A one-way hash algorithm loses its assumed one-way status as soon as it is *cracked*. Cracking a

⁴ 'Practically impossible' can be read here as: 'Even when all the computing power of the world could be used simultaneously, it is still impossible'.

hash algorithm is usually performed by searching for files that have the same hash value. Subsequently a method is sought to find such files efficiently. The above properties (assumptions) no longer hold for a particular hash algorithm once such a method is discovered. If a current hash algorithm is cracked or threatened to be so, this will usually be replaced by a new hash algorithm that cannot be cracked using the same technique(s). It is therefore necessary to switch to a new hash algorithm from time to time. In section 7.1 the current status is displayed for each of the algorithms used by DBS.

5. Forensic Applications

As long as the *one-way resistant* and the *target collision resistant* properties of a one-way hash algorithm have not been cracked, the hash values that are calculated for this purpose are forensically applicable to:

1. integrity check of digital material (see section 5.1);
2. efficient identification and classification of files (see section 5.2).

From a forensic perspective, it does not matter whether or not the properties of the random collision resistant hash algorithm have been cracked. With the other two properties, one of the two hash values is given, either as a hash value or as the digital material that the hash value has to be calculated upon. In this respect, (other) digital material must be found with the same hash value. By *random collision resistant*, no hash value is given in advance: it is sufficient to find or create two different pieces of digital material with the same hash value. The latter does not occur in forensic applications, since digital material is always delivered or received, with the calculated hash value attached to it.

5.1. Integrity Check

The main purpose of integrity checks of digital material is to detect accidental changes in (copies of) said digital material. Additionally, some forms of conscious manipulation of digital material can also be detected with an integrity check.

It is rather easy to change digital data, whether intentional or otherwise. Through the use of hash values, people can let each other know with which digital material they have worked. They can also determine whether they have worked with the same material as another person. One person, such as a digital detective, reports the hash value of the digital

material to another person, such as a DBS investigator. He/she then re-calculates the hash value of (a copy of) the supplied material. The outcome is then compared to the reported hash value. If the result is not exactly the same as the previously reported hash, the file has been changed in the meantime. The hash value does not indicate how or where the file differs from the original data. If the resulting hash is the same as the previously reported hash value, then it is extremely probable that the digital material has not been changed since the reported hash value was calculated.

5.2. File Identification and Classification

In general, the use of hash values for identification and classification of files is efficient because hash values of a file are relatively small. Nowadays files are often (very) large (a few gigabytes in size whereby a gigabyte is approximately 1 billion bytes). The maximum length of a hash value currently used by DBS is only 32 bytes (64 characters). It is therefore much easier and faster to compare the hash values of files than the contents of the files themselves. In addition, it is much easier and faster to communicate hash values of files than (the contents of) the files themselves.

Hash values are also useful in identifying files. For this forensic application, a database containing hash values of known classified files is used. This method may be used, for example, when identifying files containing child pornographic images. To identify a file, its hash value is first computed and then compared against the hash values in the database. If this hash value is present in the database, then a closer examination is performed comparing the file and the database result. If this is the case, the investigator checks in which category the hash value has been placed. The database may contain hash values of files with known child pornographic images, as well as hash values of files containing known, irrelevant data. This method makes it possible to detect relevant files, and to exclude irrelevant files from further examination at an early stage. Because it is extremely improbable that through this method different files would have the same hash value, the probability of an incorrect classification is negligible (almost 0). In section 7.3, each of the hash algorithms used by DBS is mathematically explained to show why the risk of misclassification is negligible.

6. Hash Values in Practice

DBS uses hash values mainly to verify the integrity of digital material (at the start of the investigation) or to

make the integrity of digital material verifiable (when delivering results of the investigation).

Verifying the integrity involves checking whether the materials received, or copies thereof, have not incurred any defects during transport to or from DBS and during the examination itself. An example is the securing of data stored on the hard disk of a computer (from a suspect) by the police. In such a case, the police compute one or more hash values over the data, i.e. the full contents of the hard disk. A copy of the hard disk is then sent to the DBS division together with the hash values. A DBS examiner recomputes the hash values over the received copy and compares the results to the hash values computed by the police. If the hashes are the same, it is assumed that no intervening changes have occurred during the transport. When a hash value calculated by DBS differs from the supplied hash value, then something went wrong during transport. The applicant will then be informed.

Preferably a similar check is performed during the seizure of digital material. For example, while securing data from a seized hard disk by the police, one or more hash files will be calculated on this data. Once all the data is secured, hash values will be re-calculated. Both series of hash values are subsequently compared to each other. Here too it is assumed that no changes have occurred if the hashes are the same. If, at a later stage, hash values are calculated on the data present on the same hard disk, checks can also be made as to whether or not the data has been changed in the meantime.

Digital material that is a result of an investigation is handed over to the client on, for example, a DVD-recordable or a transport hard disk. In order to be able to verify the integrity of the results at a later date, the department computes a hash value of each file produced. If the result of a case consists of a small number of files, the DBS Department reports the hash values of each file. If the result consists of a large number of files, DBS stores the hash values of these files in a new file, usually called *hashes.txt*, computes the hash value of *hashes.txt*, and includes this single hash value in its report.

Figure 1 shows schematically where hash values are (or should be) used in the process of seizure and delivery of results.

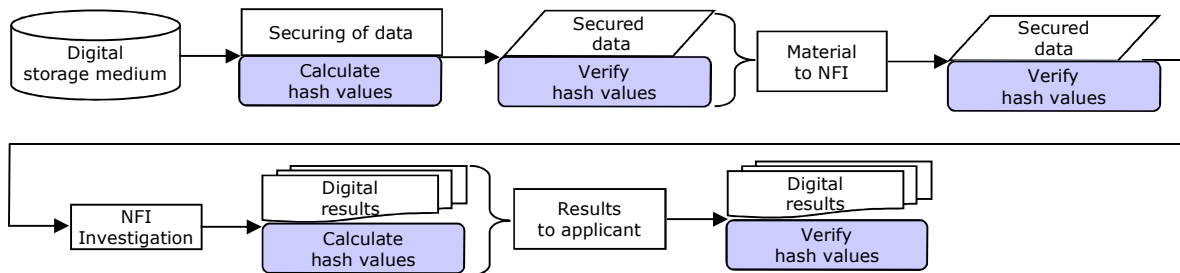


Figure 1: schematic representation of the verification of digital material by using hash values

7. Hash Algorithms Used by DBS

This section discusses the current status of the hashing algorithms in use. Based on this status, it is explained why certain algorithms are used by DBS. Subsequently, for each of these algorithms, the probability of another file with the same hash value is given.

7.1. Current Status of Hash Algorithms

Examples of conventional hash algorithms are the MD4 and MD5 ('Message Digest'), as well as the SHA ('Secure Hash Standard') series of standardised hash algorithms. The hash algorithms MD4 and SHA-0 have already been cracked. The American National Security Agency (NSA) recommended a switch over to the enhanced hash algorithm SHA-1 in 1995 because of a flaw discovered in SHA-0. Researchers discovered and published flaws in the commonly used hash algorithm MD5 in 2004. This discovery made it possible to create two files that have the same MD5 hash value. This attack on MD5 was later improved upon, making it relatively easy to create two different and meaningful files (such as legible documents) that have the same MD5 hash value. The same applies for the SHA-1 algorithm since February 2017.

It is important to note that only the collision resistance of MD5 and SHA-1 has been cracked. The forensic applications described in Section 5 are however independent of this. This means that up until now, MD5 and SHA-1 are both perfectly usable for integrity control of digital material and the identification and classification of files.

A major problem exists for some applications when an efficient method is found for creating two different files with the same hash value. In theory, such a so-called

collision attack may complicate file categorisation when a database of hash values is used to exclude non-relevant files from further examination. The creator of such a non-relevant file may also have created another, potentially incriminating file that has the same hash value. In practice, this problem can be evaded. The solution is to categorise only files of known and trusted origins as non-relevant, and thus avoid categorizing files of unknown creators as non-relevant.

For forensic applications it is much more problematic when an efficient method is found for constructing a file with the same hash value as a different given file. This is called a *second preimage attack*. For the commonly used MD5 and SHA-1 algorithms, such a method has not yet been found, but research literature does describe progress in finding such a second preimage attack for MD5 and SHA-1. A preventive switch to one of the safer SHA-2 hash algorithms is therefore recommended for forensic applications. Apart from a few exceptions, DBS switched to the SHA-2 algorithm with a length of 256 bits (also called SHA-256) in the middle of 2010.

7.2. Current Choice of Hash Algorithms at DBS

The hash algorithms used most frequently worldwide for verifying the integrity of digital data are MD5 which generates hash values with a length of 128 bits, and SHA-1 which generates hash values with a length of 160 bits. These algorithms are widely used, both within and outside the forensic community. The safety guarantees that they offer (for integrity check and file identification) are still extremely high, despite constant 'attacks' on these algorithms (for numerical definitions, see section 7.3). As mentioned earlier, DBS switched over to the SHA-256 algorithm in 2010 as a precaution. However in practice, DBS still often has to verify hash values that chain partners supply. Here, MD5 and SHA-1 hash values are still supplied (also due to the software used by the chain partners).

In the period 2010-2012, a public competition was held via the American National Institute for Standards and Technology (NIST) to choose an even better non-reversible hash algorithm (SHA-3). On the 2nd October 2012 the winning algorithm was chosen, named *Keccak*. At the time of the adaptation of this technical supplement, the use of SHA-3-Keccak is still not common in the forensic world. Since SHA-256 is still safe enough for the time being, DBS has not yet switched to SHA-3-Keccak.

7.3. Chance in Practice of Another File with the Same Hash Value

The probability that another random file has the same hash value as that of a *given* file will be explained here by means of the following example. A certain database contains hash values of files with content that has been classified as child pornography. Now suppose that in an investigation a file with the name B is found, where the hash value of B occurs in this database. What is the chance that B happens to have a different content than the child pornographic file? This is calculated here for each of the hash algorithms currently in use.

- MD5
An MD5 hash value consists of 128 bits. Assuming that every MD5 hash value can occur with equal probability, then there is a total of 2^{128} ($\approx 3,40 \times 10^{38}$) possible MD5 values. This means that the probability that any other file with different content and yet with the same MD5 hash value is equal to $\frac{1}{2^{128}} \approx 2,9 \times 10^{-39}$.
- SHA-1
A SHA-1 hash value consists of 160 bits. A similar calculation to that of the MD5 hash value indicates that the probability that any other file with different content and the same SHA-1 hash value is equal to $\frac{1}{2^{160}} \approx 6,8 \times 10^{-49}$.
- SHA-256
A SHA-256 hash value consists of 256 bits. A similar calculation to that of MD5 and SHA-1 has values indicates that the probability that any other file with different content but with the same SHA-256 hash value is equal to $\frac{1}{2^{256}} \approx 8,6 \times 10^{-78}$.

On the basis of these calculations and with the current state of technology, for each of the hash values used by DBS, it can be said that the probability that a file with content other than a known file with the same hash value is extremely low (almost zero).

By comparison: the theoretical *random match probability* of the *Next-Generation Multiplex* (NGM)

system in the Dutch population (based on 15 loci) is at least $1,0 \times 10^{-25}$. From the above chance calculations, it appears that the probability that two files with differing content by coincidence have the same MD5, SHA-1 and / or SHA-256 hash value is equal to or less than $2,9 \times 10^{-39}$.

This last probability is therefore extremely much lower than that of two different people that have by coincidence an indistinguishable DNA profile.

8. Glossary

- Algorithm
A finite series of instructions to achieve a clearly described final result from a given initial state (named after the Arabic mathematician Al-Chwarizmi). By using a computer language, an algorithm can be implemented into a computer program and thus (automatically) executed by a computer. An algorithm can be compared to a recipe: certain steps must be taken in order to achieve the (desired) final result. An extremely simple example of an (arithmetic) algorithm to calculate the number of times a number must be added to itself in order to be equal to or larger than 100, is as follows:

```

Step 1: ask for a number
Step 2: number of times = 0
Step 3: intermediate result = number
Step 4: if the intermediate result is
equal to or larger than 100, go to step 8
Step 5: increment the intermediate result
by the number
Step 6: increment the number of times by 1
Step 7: go back to step 4
Step 8: give the number of times


```

If in the first step the given number is '7', then the answer ('number of times') will be '14'.

- Bits and Bytes
A bit can have a value of '0' or '1'. A byte consists of 8 bits. Because each bit can have two values ('0' or '1'), a byte can have $2^8 = 256$ possible values. Bytes are often written in hexadecimal notation (prefixed by a 0x to distinguish it from a normal decimal number). The value of a byte therefore varies in hexadecimal between 0x00 and 0xFF.

9. Bibliography

- Rivest, R.L. *The MD4 Message Digest Algorithm*, Crypto '90 Proceedings, 1991.
- Rivest, R.L. *The MD5 Message-Digest Algorithm*, Request for Comments (RFC) 1321, Internet Activities Board, Internet Privacy Task Force, 1992.
- FIPS 180-1, *Secure hash standard*, National Institute for Standards and Technology (NIST), US Department of Commerce, Washington D.C., April 1995. Springer-Verlag, 1996.
- Menezes, A.J., Oorschot, P.C. van, Vanstone, S.A. *Handbook of Applied Cryptography*, CRC Press, 1997.
- Chabaud, F., Joux, A. *Differential Collisions in SHA-0*, Advances in Cryptology, Springer Verlag, LNCS 1462, 1998.
- Wang, X.Y., Guo, F.D., Lai, X.J., Yu, H.B. *Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD*, Rump Session of Crypto'04, E-print, 2004
- Weger, B.M.M. de, *Hash-functies onder vuur, de situatie van MD5 en SHA-1*, PvIB (Platform voor Informatie Beveiliging), number 2, pages 25-30, 2005 (in Dutch)
- Wang, X.Y., Lai, X.J., Feng, D., Chen, H., Yu, X.Y. *Cryptanalysis of the Hash Functions MD4 and RIPEMD*, Advances in Cryptology, Springer Verlag, LNCS 3494, 2005.
- Hoffman, P. Schneier, B. *Attacks on Cryptographic Hashes in Internet Protocols*, Request for Comments (RFC) 4270, Internet Engineering Task Force, 2005.
- Joux, A. *Multicollisions in iterated hash functions. Application to cascaded constructions*, DCSSI Crypto Lab, France, 2005.
- Biham, E., Chen, R., Joux, A., Carribault, P., Lemuët, C., Jalby, W. *Collisions of SHA-0 and Reduced SHA-1*, IACR paper volume 3494, pages 36-57, 2005.
- Wang, X.Y., Yu, H.B. *How to Break MD5 and Other Hash Functions*, Advances in Cryptology EuroCrypt 2005, Springer Verlag, pages 19-35, 2005.
- Aumasson, J.P., Meier W., Mende, F. *Preimage Attacks on 3-Pass HAVAL and Step-Reduced MD5*, Cryptology ePrint Archive, Report 2008/183, 2008.
- Cannière, C., Rechberger, C. *Preimages for Reduced SHA-0 and SHA-1*. Advances in Cryptology, Springer Verlag, LNCS 5157, 2008.
- Meulenbroek, A.J. *De essenties van forensisch biologisch onderzoek – Humane biologische sporen en DNA*, publisher Paris, chapter 7 (pages 157 – 176), 5th edition, 2009 (in Dutch).
- Information sheet *DNA-verwantschapsonderzoek*, version 1, NFI (in Dutch).
- Bertoni, G., Daemen, J., Peeters, M., Assche, G. van, *The Keccak reference*, version 3.0, 2011 (www.keccak.noekeon.org).
- NIST, *SHA-3 Competition (2007-2012)*, <http://csrc.nist.gov/groups/ST/hash/sha-3/index.html>.
- Clark, R.A., Morell, M.L., Stone, G.R., Sunstein, C.R., Swirre, P. *The NSA Report*, Princeton University Press, 2013.
- Stevens, M., Bursztein, E., Karpman, P., Albertini, A., Markov, Y. *The first collision for full SHA-1*, preprint 2017 (<https://shattered.io/>).



For general questions, please contact the Frontdesk,
telephone number +31 (0)70 888 68 88.

For questions concerning the content, please contact
the Digital and Biometric Traces Division, telephone
number +31 (0) 70 888 6400.

Netherlands Forensic Institute
Ministry of Justice and Security
P.O. Box 24044 | 2490 AA The Hague

Telephone number +31 (0)70 888 66 66
www.forensicinstitute.nl

January 2018