NUMBER 4 — NOVEMBER 12, 1996

RSA Laboratories' Bulletin

News and advice from RSA Laboratories

# On Recent Results for MD2, MD4 and MD5

**M.J.B. Robshaw**
*RSA Laboratories*

***Abstract.*** *Recent cryptanalytic results on the properties of three popular hash functions have raised questions about their security. This note summarizes these results, gives our assessment of their implications and offers our recommendations for product planners and developers who may be using these algorithms.*

## 1. Introduction

A *hash function* (or more accurately a *cryptographic hash function* or *message-digest algorithm*) operates on an input string of arbitrary length and generates an output string of fixed length. This output is commonly called a *hash value* or a *message digest*. While much of the motivation for the design of a hash function comes from its usefulness in optimizing the process of digitally signing some document, hash functions can be used for a wide range of purposes.

MD2 [13], MD4 [20] and MD5 [21] are hash functions that were developed by Ron Rivest at MIT for RSA Data Security. A description of these hash functions can be found in RSA Laboratories Technical Report TR-101 [22]. The widespread popularity of the MD family of hash functions is a testament to their innovative and successful design. Indeed MD4 in particular has been used as the basis for the design of many other hash functions (including MD5, SHA-1, RIPEMD) and MD5 is one of the most widely used hash functions in the world today.

Over the years, various results in the cryptanalysis of MD2, MD4 and MD5 have become available and this note is intended to summarize these results and their impact. First however we will consider the properties of hash functions. An important issue, that we will repeatedly stress, is that not all applications of a hash function rely for their security on the same property. Consequently, identifying which property of a hash function is appealed to within an implementation is very important and sometimes leads to surprising results.

*Matt Robshaw is a senior research scientist at RSA Laboratories. He can be contacted at matt@rsa.com*

## 2. Hash Function Properties

Hash functions are designed with a variety of properties in mind and three are commonly singled out in the literature [18].

First, given the hash value output by some hash function, it should be infeasible to find an input or *preimage* that will produce the given output. A slight variation on this gives a second condition which is that even when given an input and output pair for some hash function, it should remain infeasible to find a second distinct preimage that would generate the same output. This is commonly referred to as finding a *second preimage* and a hash function for which it is difficult to find either a preimage or a second preimage is sometimes called a *one-way hash function* [18].

A third condition that is sometimes required is that it be infeasible to find two inputs to the hash function that will produce the same output. This is commonly referred to as finding a *collision* for the hash function. Since there are an arbitrary number of possible input strings but only a fixed number of outputs, collisions must exist for a hash function — our objective is to ensure that it is computationally infeasible to find such examples. The term *collision-*

RSA LABORATORIES
CRYPTOGRAPHIC RESEARCH AND CONSULTATION

*resistant hash function* [18] is sometimes used to describe a hash function that possesses all three of the properties described here and it is what most people have in mind when talking about hash functions in general.

Note that while a collision-resistant hash function has many useful properties, the property that is actually appealed to within some application can vary dramatically.

For instance, a hash function is often used to reduce some message to a short string which is then signed with a digital signature algorithm. There is a well-known attack on this procedure [26] that depends on the so-called birthday paradox. To prevent this attack we require the property of collision-resistance and as a consequence hash functions that produce an output of at least 128 bits in length. It is worth mentioning that this style of attack is more applicable when the digital signature is computed directly on the digest of the message being signed. Newly proposed probabilistic signature schemes by Bellare and Rogaway [3] have some very attractive properties and it appears that the requirements we place on a hash function in such schemes might be less demanding[1].

Interestingly, digital signatures previously signed using a hash function that is no longer collision resistant are likely to remain safe from compromise. In attacking existing signatures, the task of the cryptanalyst is essentially that of finding a second preimage since the first preimage (the message signed) and the associated hash value are already fixed. This is a much harder problem than that of generally finding a collision and it might well be the case that techniques that generate collisions for a hash function offer no advantage in finding a second preimage. Thus, existing signatures might well remain free from risk of compromise even when collision-resistance is lost.

When a hash function is used for properties other than being collision-resistant, the situation with regards to the suitability of that hash function after the discovery of collisions might remain essentially unchanged. Perhaps one of the properties most commonly appealed to in a hash function is that of providing a random looking output. A function with this property has many applications and while work on collisions for the hash function clearly has implications for any assumptions about its ideal behavior, it is not clear that they will necessarily offer a substantial practical reason to move away from that hash function as a matter of urgency. Another property that might well be largely unaffected by work on collisions is that of being one-way. When a hash function is being used purely as a one-way function, perhaps to store a table of hashes of computer passwords, then work on collisions might well have little relevance.

In short there are many situations where hash functions are used for properties other than being collision-resistant and we observe that properties that are relevant to cryptographic security in one environment are not necessarily relevant in another.

## 3. Hash Function Structure

Most hash functions have a similar iterative structure which is based around what is termed a *compression function* [4, 14]. In short, the computation of the hash value for some message depends on what is called a *chaining variable*. At the start of hashing, this chaining variable has a fixed initial value which is specified as part of the algorithm. The compression function is then used to update the value of this chaining variable in a suitably complex way under the action and influence of part of the message being hashed. This process continues recursively, with the chaining variable being updated under the action of different parts of the message, until all the message (and any additional padding specified by the algorithm) has been used. The final value of the chaining variable is then output as the hash value corresponding to that message.

The hash functions MD4 and MD5 are quite similar in design and, as we have already mentioned, they were the inspiration behind the design of several more recent hash functions. MD2 was an earlier hash function with a different structure but it is still widely used if only because it is suitable for 8-bit environments (unlike MD4 and MD5 which are aimed at 32-bit architectures). In the following sections, we will review the status of the three hash functions MD2, MD4 and MD5. This note might be viewed as an update to the RSA Technical Report TR-101 [22] which provides an overview of the same hash functions prior to the very latest developments.

[1] *In particular, certain simple variants of the schemes in [3] might still securely allow the use of a hash function even though that hash function might not be fully collision-resistant. (These variants would only differ from the detailed specifications in [3] in the order in which the random number* r *and the message* M *are concatenated prior to hashing.) Further work may well reveal other advantages in varying the length, or the means of generation, of the value* r.

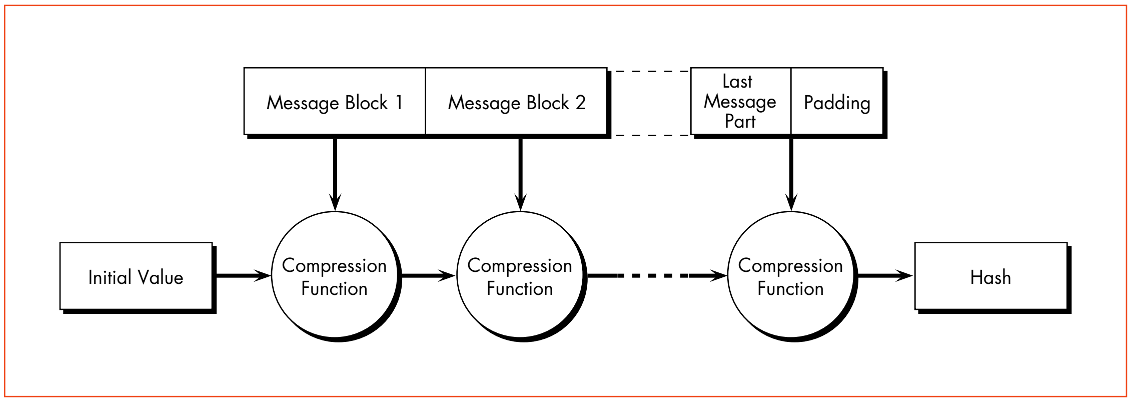

*Figure 1. The use of a compression function in an iterative hash function. The hash functions MD2, MD4, MD5 and SHA-1 essentially follow this design.*

## 4. The Status of MD2

When hashing a message with MD2 there are three different phases. The first is a padding phase whereby the message is padded to form a string that has a length in bytes which is divisible by 16. The second phase is the computation of a 16-byte checksum C which is appended to the end of the message; the checksum is a function of the message and it is computed under the action of a non-linear substitution table based on the digits of $\pi$. The resultant string is then divided into 16-byte blocks. The final block, therefore, is the 16-byte checksum. The third phase consists of repeated application of the compression function to compute new values for the chaining variable as a function of both the current value and each message block in turn. The initial value of the chaining variable is fixed as part of the algorithm specifications and the last value for the chaining variable becomes the 16-byte (128-bit) MD2 hash value.

Considerable progress has been made in the cryptanalysis of MD2 [23] and it has been shown that it is possible to find collisions for the compression function of MD2. More precisely, it is possible to find two 16-byte strings such that compression, when starting with particular values of the chaining variable (including the correct initial value), will yield the same output from the compression function. Such work would lead directly to collisions in MD2 were it not for the fact that the final computation of the MD2 hash value depends on the value of the checksum C and this is likely to be different for the two messages.

Currently, it appears to be difficult to make allowances for the existence of the checksum during cryptanalysis. Without the checksum MD2 would be broken but with it, this attempt at cryptanalysis has been thwarted at the very last hurdle. This is perhaps rather too close for comfort and caution requires that MD2 be no longer recommended for new applications where collision-resistance is required. Questions about the continuing suitability of MD2 for existing applications remain open. Certainly MD2 has not been broken and existing signatures are not at risk since it is still difficult to find preimages (input messages which yield a given target value with MD2). In addition, future signing will only become vulnerable if full collisions for MD2 can be found. Despite this, while MD2 might still be used in the short term, our recommendation would be to upgrade applications away from MD2 whenever it is practical.

## 5. The Status of MD4

MD4 was an early design for a particularly fast hash function. Successful cryptanalysis of reduced-round versions of MD4 [15, 5] soon demonstrated, however, that the design of MD4 represented an uncomfortable compromise between security and speed. As a consequence, the more conservatively designed MD5 has always been recommended for use instead of MD4.

Work by Dobbertin [7, 8] has vindicated this early concern about MD4 and it has been shown that collisions for MD4 can be found in about a minute on a typical PC. In addition, a variant of MD4 called extended-MD4 [20], which offers a 256-bit hash value instead of the usual 128-bit MD4 output, has also been seriously undermined by the work of Dobbertin.

While much of this cryptanalytic work is concerned exclusively with finding collisions, considerable insight into the behavior of MD4 has also been gained, particularly when combined with other independent cryptanalytic work. Both MD4 and extended-MD4 should not be used.

## 6. The Status of MD5

Attacking MD5 is a much more involved proposition than attacking MD4 since it is a far more complicated algorithm to analyze. The first important advance in the cryptanalysis of MD5 was the discovery of what are termed *pseudo-collisions* for the compression function of MD5 [6]. A pseudo-collision for the compression function is exemplified by fixing the value of some message block and finding two distinct values for the chaining variable that provide the same output. While the existence of pseudo-collisions is significant on an analytical level, it is of less practical importance. Recall that only a single chaining variable is used during hashing and so the behavior of two related chaining variables is not directly relevant. Instead, it would be more significant if we could identify the value of a single chaining variable for which two different message blocks produce the same output from the compression function. Such an occurrence would have obvious implications for the collision-resistant property we often desire of a hash function. If the value of the chaining variable involved were not the same as the initial value (as provided in the algorithm specifications) then such an occurrence would be termed a *collision for the compression function*. If, however, we could identify two message blocks which provide a collision when the pre-specified initial value is used, then we would have full collisions for the hash function.

At Eurocrypt '96 it was announced that collisions for the compression function of MD5 had been found [9]. In a modification to the techniques used so devastatingly on MD4, Dobbertin demonstrated that collisions for the compression function of MD5 could be found in around 10 hours on a PC. Whereas the pseudo-collisions discovered by den Boer and Bosselaers could not be extended to full collisions for MD5, no such comfort can be drawn with regard to the recent work of Dobbertin. This, of course, should not be viewed as a statement that such an extension is trivial. With the current attack, the cryptanalyst has some freedom in deriving the collision for the compression function, but once this collision has been achieved it seems to be difficult for the cryptanalyst to account for the fact that the initial value to the chaining variable is pre-specified as part of the algorithm. Clearly, to become an attack on the full MD5, some means of directing the value of the input chaining variable to the one specified in the algorithm is required.

While more, possibly very complex, analytical work is required in designing an attack for MD5, there is no telling how much time or effort this might require. Nevertheless, it would be a mistake to rely too much on the fact that current techniques only provide collisions for the compression function of MD5. Given the surprising speed with which techniques on MD4 were extended to MD5 we feel that it is only prudent to draw a cautious conclusion and to expect that collisions for the entire hash function might soon be found.

Recalling our comments in Section 2, however, we note that there may well be situations where this cryptanalytic work on MD5 has little impact. Note that existing signatures that were generated using MD5 are likely to remain safe from compromise since it seems that current techniques used to cryptanalyze MD5 do not offer any advantage in finding a second preimage. Existing signatures should not be considered as being at risk of compromise at this point. Likewise the random-looking appearance of the output from MD5 and the property of being one-way are not considered to be seriously in question. Finally, one major recent proposal for the use of MD5 has been in what is sometimes referred to as the *keyed-MD5* approach to message authentication. In particular, one proposal termed HMAC [1, 2] produces a message authentication code for some message with the help of a hash function, and for implementation efficiency, the use of MD5 in particular has been proposed. The design and properties of HMAC are such that Dobbertin's current techniques that might find collisions for either the compression function or the full hash function of MD5 seem to have little immediate impact on the security of HMAC when MD5 is used [10].

## 7. Some Alternative Hash Functions

As alternative hash functions, SHA-1, RIPEMD-128 and RIPEMD-160 can still be recommended as being secure for any application. While all three hash functions bear similarities to MD4 and MD5 in their design, the techniques of Dobbertin do not readily extend to these hash functions. Indeed, one of the design criteria for RIPEMD-128 and -160 was that this be the case.

SHA-1 is a revision [17] of the Secure Hash Algorithm (SHA) which first appeared as part of the Secure Hash Standard, FIPS 180 [16]. While the fault in the original SHA and the reasons for the particular change made in SHA-1 are not known, it can be anticipated that SHA-1 is a good hash algorithm to use.

The forerunner to both RIPEMD-128 and RIPEMD-160 was RIPEMD [19], a 128-bit hash function developed within the framework of the European

Union project RIPE (Race Integrity Primitives Evaluation). Its design was based very closely around the principles adopted by Rivest in the design of MD4, and was the subject of considerable analysis, chiefly during the writing of the RIPE report. Despite this however, the techniques recently developed by Dobbertin were first used in attacks on reduced-round versions of RIPEMD [11]. While not extending to the full hash function, the situation with regards to RIPEMD is analogous to that which existed for several years with MD4 when only attacks against a reduced-round version of MD4 were known.

As a consequence, two variants of RIPEMD were proposed [12]. RIPEMD-128 (providing a 128-bit hash value) was intended as a drop-in replacement for RIPEMD and offers additional protection against the cryptanalytic techniques of Dobbertin. However, the longer hash value provided by RIPEMD-160 (with a 160-bit hash value) is likely to be appealing in the longer term [24]. With regards to performance, while RIPEMD-128 appears to be somewhat faster than SHA-1, RIPEMD-160 is slightly slower [12]. All three hash functions are slower than MD5.

## 8. Summary and Conclusions

In summary, we list the more significant cryptanalytic results on MD2, MD4 and MD5 and describe some of their immediate consequences.

**MD2** Collisions have been demonstrated for a modified version of MD2 [23].

> *Existing signatures formed using MD2 are not at risk, but MD2 can no longer be recommended for future applications that will depend on the hash function being collision-resistant. MD2 remains suitable for use as a one-way hash function.*

**MD4** A variety of cryptanalytic results cast doubt on the complexity of the MD4 design [15, 5, 25]. Collisions have been demonstrated for MD4 [7, 8].

> *MD4 should not be used. (This merely restates a recommendation which has been present in the literature for some time, in fact, its use has not been recommended since the introduction of MD5.)*

**MD5** Both pseudo-collisions [6] and collisions [9, 10] for the compression function of MD5 have been demonstrated, though collisions for the full MD5 have not yet been achieved.

> *Existing signatures formed using MD5 are not at risk and while MD5 is still suitable for a variety of applications (namely those which rely on the one-way property of MD5 and on the random appearance of the output) as a precaution it should not be used for future applications that require the hash function to be collision-resistant.*

In this bulletin we have considered various cryptanalytic developments in the analysis of the compression functions of MD2 and MD5 and for the entire MD4 hash function.

Some of this new cryptanalytic evidence confirms suspicions voiced several years ago by Ron Rivest and makes clear that MD4 should not be used. With regards to existing applications which use MD2 and MD5, collisions for these hash functions have not yet been discovered but this advance should be expected. As a consequence applications which rely on the collision-resistance of a hash function should be upgraded away from MD2 and MD5 when practical and convenient. They can probably be safely "swapped out" in coordination with the vendor's normal product release cycle. RSA Laboratories currently recommends that in general, the hash function SHA-1 [17] be used instead but RIPEMD-160 would also be a good alternative. (It is interesting that both hash functions borrow heavily from the initial structural design of MD4.)

Occasionally performance requirements or the existence of fielded applications might make a move to SHA-1 undesirable when contrasted with the possibility of using MD5. In such cases it should be confirmed that either collision-resistance is not required within the application or that existing collision attacks do not apply in that particular environment.

It is important to note that these recent cryptanalytic efforts on both MD2 and MD5 are almost exclusively relevant to finding collisions. The most significant impact of the discovery of hash function collisions is on the suitability of that hash function as a part of the process of digitally signing some document. Even then, some signature schemes might well be more tolerant of the presence of collisions than others and so careful analysis is recommended to assess exactly what properties of the hash function are being relied upon in an application. Note that we expect the discovery of collisions to have little or no impact on the other properties that we usually associate with a hash function, such as pseudo-randomness or one-wayness.

BSAFE customers should note that while the continued suitability of MD5 for applications requiring collision-resistance is in question, the use of MD5 in other roles, for example as part of a mechanism for random number generation, is still considered safe. Indeed, MD2 and MD5 may well remain suitable for a wide range of applications.

## References

[1] M. Bellare, R. Canetti and H. Krawczyk. The HMAC construction. *CryptoBytes*, 2(1): 12-15, 1996.

[2] M. Bellare, R. Canetti and H. Krawczyk. Keying hash functions for message authentication. In *Advances in Cryptology — Crypto '96*, pages 1-15, Springer-Verlag, 1996.

[3] M. Bellare and P. Rogaway. The exact security of digital signatures — how to sign with RSA and Rabin. In *Advances in Cryptology — Eurocrypt '96*, pages 399-416, Springer-Verlag, 1996.

[4] I. Damgård. A design principle for hash functions. In *Advances in Cryptology — Crypto '89*, pages 416–427, Springer-Verlag, 1990.

[5] B. den Boer and A. Bosselaers. An attack on the last two rounds of MD4. In *Advances in Cryptology — Crypto '91*, pages 194-203, Springer-Verlag, 1992.

[6] B. den Boer and A. Bosselaers. Collisions for the compression function of MD5. In *Advances in Cryptology — Eurocrypt '93*, pages 293-304, Springer-Verlag, 1994.

[7] H. Dobbertin. Alf swindles Ann. *CryptoBytes*, 1(3): 5, 1995.

[8] H. Dobbertin. Cryptanalysis of MD4. In *Proceedings of the 3rd Workshop on Fast Software Encryption*, Cambridge, U.K., pages 53-70, Lecture Notes in Computer Science 1039, Springer-Verlag, 1996.

[9] H. Dobbertin. *Cryptanalysis of MD5 Compress*. Presented at the rump session of Eurocrypt '96, May 14, 1996.

[10] H. Dobbertin. The Status of MD5 after a Recent Attack. *CryptoBytes*, 2(2): 1-6, 1996

[11] H. Dobbertin. RIPEMD with two round compress function is not collision-free. *Journal of Cryptology*. To appear.

[12] H. Dobbertin, A. Bosselaers, and B. Preneel. RIPEMD-160: A strengthened version of RIPEMD. Final version available via ftp at `ftp.esat.kuleuven.ac.be/pub/COSIC/bosselae/ripemd/`.

[13] B.S. Kaliski Jr. *RFC 1319: The MD2 Message-Digest Algorithm*. RSA Data Security, Inc., April 1992.

[14] R.C. Merkle. One way hash functions and DES. In *Advances in Cryptology — Crypto '89*, pages 428–446, Springer-Verlag, 1990.

[15] R.C. Merkle. *Note on MD4*. 1990. Unpublished. (A description can be found in [5].)

[16] National Institute of Standards and Technology (NIST). *FIPS Publication 180: Secure Hash Standard (SHS)*. May 1993.

[17] National Institute of Standards and Technology (NIST). *FIPS Publication 180-1: Secure Hash Standard.* April 1994. Available from `http://csrc.ncsl.nist.gov/fips/`.

[18] B. Preneel. *Analysis and Design of Cryptographic Hash Functions*. Ph.D. Thesis, Katholieke Universiteit, Leuven, 1993.

[19] *RIPE Integrity Primitives*. Final report of RACE Integrity Primitives Evaluation (R1040), Part III: Recommended Integrity Primitives, Chapter 3: RIPEMD, June 1992, pages 67-109.

[20] R.L. Rivest. The MD4 message digest algorithm. In *Advances in Cryptology — Crypto '90*, pages 303-311, Springer-Verlag, 1991.

[21] R.L. Rivest. *RFC 1321: The MD5 Message-Digest Algorithm*. M.I.T. Laboratory for Computer Science and RSA Data Security, Inc., April 1992.

[22] M.J.B. Robshaw. *MD2, MD4, MD5, SHA and other Hash Functions*. RSA Laboratories Technical Report TR-101, version 4.0, July 24, 1995.

[23] N. Rogier and P. Chauvaud. The compression function of MD2 is not collision free. Presented at *Selected Areas in Cryptography '95*, Carleton University, Ottawa, Canada. May 18-19, 1995.

[24] P. van Oorschot and M. Wiener. Parallel collision search with application to hash functions and discrete logarithms. In *Proceedings of 2nd ACM Conference on Computer and Communication Security*, pages 210-218, ACM Press, 1994.

[25] S. Vaudenay. On the need for multipermutations: Cryptanalysis of MD4 and SAFER. In *Proceedings of the 2nd Workshop on Fast Software Encryption*, Leuven, Belgium, pages 286-297, Lecture Notes in Computer Science 1008, Springer-Verlag, 1995.

[26] G. Yuval. How to swindle Rabin. *Cryptologia*, July 1979.

For more information on this and other recent security developments, contact RSA Laboratories at one of the addresses below.

**RSA Laboratories**
100 Marine Parkway, Suite 500
Redwood City, CA 94065 USA
415/595-7703
415/595-4126 (fax)
*rsa-labs@rsa.com*
*http://www.rsa.com/rsalabs/*