J. Cryptology (1998) 11: 59–72

Journal of
CRYPTOLOGY

Attacks on Fast Double Block Length Hash Functions*

Lars R. Knudsen
Katholieke Universiteit Leuven, Kardinaal Mercierlaan 94,
B-3001 Heverlee, Belgium

Xuejia Lai
R^3 Security Engineering, Aathal, Switzerland

Bart Preneel
Katholieke Universiteit Leuven, Kardinaal Mercierlaan 94,
B-3001 Heverlee, Belgium

Communicated by Tom Berson

Received 1 March 1996 and revised 16 December 1996

Abstract. The security of hash functions based on a block cipher with a block length of m bits and a key length of k bits, where $k \leq m$, is considered. New attacks are presented on a large class of iterated hash functions with a $2m$-bit hash result which processes in each iteration two message blocks using two encryptions. In particular, the attacks break three proposed schemes: Parallel-DM, the PBGV hash function, and the LOKI DBH mode.

Key words. Cryptanalysis, Cryptographic hash functions, Block ciphers, Double block length hash functions, Birthday attacks.

1. Introduction

A *hash function* is an easily implementable mapping from the set of all binary sequences to the set of binary sequences of some fixed length. An *iterated hash function* is a hash function Hash$(\cdot)$ based on an easily computable function $h(\cdot, \cdot)$ from two binary sequences of respective lengths m and l to a binary sequence of length m. If the length of the input string is not a multiple of l, the input string is padded using an unambiguous padding rule. The padded input M is then split into l-bit blocks M_i, or $M = (M_1, M_2, \ldots, M_n)$. The *hash result* $H = H_n$ of length m is obtained by computing

* Part of the results in this paper were presented at Eurocrypt '94. Lars R. Knudsen is a postdoctoral researcher, sponsored by the Danish Technical Science Foundation. Bart Preneel is an N.F.W.O. postdoctoral researcher, sponsored by the National Fund for Scientific Research (Belgium).

iteratively

$$H_i = h(H_{i-1}, M_i), \qquad i = 1, 2, \ldots, n, \tag{1}$$

where H_0 is a specified *initial value*. The function h is called the *hash round function* or the compression function. Hash functions which satisfy some security properties are widely used in cryptographic applications such as digital signatures, password protection schemes, and conventional message authentication.

For iterated hash functions, seven attacks can be distinguished:

1. *Preimage attack*: Given H_0 and $\text{Hash}(H_0, M)$ find M' such that $\text{Hash}(H_0, M') = \text{Hash}(H_0, M)$.
2. *2nd preimage attack*: Given H_0 and M, find M' such that $M' \neq M$ but $\text{Hash}(H_0, M') = \text{Hash}(H_0, M)$.
3. *Free-start preimage attack*: Given H_0 and $\text{Hash}(H_0, M)$, find H_0' and M' such that $\text{Hash}(H_0', M') = \text{Hash}(H_0, M)$.
4. *Free-start 2nd preimage attack*: Given H_0 and M, find H_0' and M' such that $(H_0', M') \neq (H_0, M)$ but $\text{Hash}(H_0', M') = \text{Hash}(H_0, M)$.
5. *Collision attack*: Given H_0, find M and M' such that $M' \neq M$ but $\text{Hash}(H_0, M') = \text{Hash}(H_0, M)$.
6. *Semi-free-start collision attack*: Find H_0, M, and M' such that $M' \neq M$ but $\text{Hash}(H_0, M') = \text{Hash}(H_0, M)$.
7. *Free-start collision attack*: Find H_0, H_0', M, and M' such that $(H_0', M') \neq (H_0, M)$ but $\text{Hash}(H_0', M') = \text{Hash}(H_0, M)$.

This list of attacks is from [11]. Similar definitions appear in [12] and [14]. It depends on the application which of these attacks on hash functions are relevant; if an attack applies, one has to impose that it is computationally infeasible for the hash function in question. For example, for password protection, a preimage attack is the only possible attack provided H_0 is included in the definition of the hash function. In most applications H_0 is specified and fixed, and the only feasible attacks are attacks 1, 2, and 5; attacks 3, 4, 6, and 7 cannot be used since a hash result computed from a different initial value will not be accepted. However, these attacks correspond to attacks on the hash round function, and as such have some value as certificational attacks. However, if the sender is free to choose and/or to change H_0, they can be realistic attacks. Note that the free-start and semi-free-start attacks are never harder than the attacks where H_0 is specified in advance. Also, a collision attack cannot be harder than a 2nd preimage attack.

For an m-bit hash function, brute-force preimage (and 2nd preimage) attacks, in which an M' is randomly chosen until one hits a given $H = \text{Hash}(H_0, M)$, require about 2^m evaluations of $\text{Hash}(\cdot)$. It follows from the usual "birthday argument" that brute-force collision attacks require about $2^{m/2}$ evaluations of $\text{Hash}(\cdot)$. In most cases preimage attacks require about 2^m computations of the round function h while brute-force collision attacks require about $2^{m/2}$ computations of the round function h. Using the method of "distinguished points" [17], [19], the collision attacks can be implemented with little memory. This collision search technique can be parallelized efficiently; the reader is referred to [19] for more details.

To avoid some trivial attacks [12], the following strengthening of iterated hash functions was proposed independently by Damgård [3] and Merkle [13].

Definition 1 (MD-Strengthening (Merkle–Damgård Strengthening)). Consider an iterated hash function Hash$(\cdot)$ and a message to be hashed $M = (M_1, M_2, \ldots, M_n)$. Specify an extra message block M_{n+1} containing the length of M (before padding) in bits. The hash result of M is defined as Hash(M'), where $M' = (M_1, M_2, \ldots, M_n, M_{n+1})$.

The attacks presented in this paper produce messages of equal length, and are therefore independent of MD-strengthening.

The following connection between a hash function and its hash round function can be proved [3], [12], [13].

Theorem 1. *Free-start collision and free-start preimage attacks against an iterated hash function with MD-strengthening have roughly the same complexities as free-start collision and free-start preimage attacks against the hash round function.*

This paper considers iterated hash functions based on block ciphers with block length and key length both equal to m bits. Such a block cipher defines, for each m-bit key, a reversible mapping from the set of all m-bit plaintexts onto the set of all m-bit ciphertexts. In the following $E_Z(X)$ denotes the encryption of the m-bit plaintext X under the m-bit key Z, and $D_Z(Y)$ denotes the decryption of the m-bit ciphertext Y under the m-bit key Z. Note that it is possible to extend the attacks to the case where the key length is less than the block length, such as DES [5], which has a 64-bit block length and a 56-bit key. It is assumed that the block cipher has no weaknesses, i.e., for every key it can be modeled as a random permutation (see, for example, [13]).

The *hash rate* of an iterated hash function (or, equivalently, of a round function) is defined as the number of m-bit message blocks processed per encryption or decryption. The *complexity* of an attack is the total number of operations, i.e., encryptions or decryptions, required for the attack to succeed with a high probability.

The following results from probability theory are used [4], [7]:

Lemma 1. *When drawing a sample of size r from a set of N elements with replacements, where $r, N \to \infty$ and $r/N \to z$, the probability that a given element is drawn converges to*

$$1 - \exp(-z). \tag{2}$$

Lemma 2. *When drawing a sample of size r from a set of N elements with replacements, where $r, N \to \infty$ and $r^2/(2N) \to \lambda$, the distribution of the number of coincidences converges to a Poisson distribution with expected value λ or*

$$\Pr(\#\ \textit{coincidences} = c) = e^{-\lambda} \cdot \frac{\lambda^c}{c!}, \qquad c \geq 0. \tag{3}$$

The probability that there is at least one coincidence is given by

$$1 - \exp(-\lambda). \tag{4}$$

An identical result holds when two samples of sizes r and s are drawn from a set of N elements with replacements, where $r, s, N \to \infty$ and $rs/N \to \lambda$.

The remainder of this paper is organized as follows. In Section 2 *single block length hash functions* are reviewed, i.e., hash functions based on block ciphers with an m-bit hash result. In Section 3 *double block length hash functions* are considered, i.e., hash functions with a $2m$-bit hash result. Section 4 presents a new attack, the "solving one-half" attack. Section 5 contains the main result of the paper: it describes attacks on all double block length hash functions of hash rate 1. In the last section the conclusions are given as well as some open problems.

2. Single Block Length Hash Functions

In [15] it was shown that there exist basically two single block length hash functions believed to be secure, namely, the scheme known as the Davies–Meyer scheme:

$$H_i = E_{M_i}(H_{i-1}) \oplus H_{i-1}, \tag{5}$$

and the scheme proposed independently by Miyaguchi [10] and Preneel [14]:

$$H_i = E_{M_i}(H_{i-1}) \oplus H_{i-1} \oplus M_i. \tag{6}$$

All other secure single block length hash functions can be transformed into either (5) or (6) by linear transformations of the inputs M_i and H_{i-1} [15]. For example, the single block length hash function of the International Standard ISO/IEC 10118-2 [9] is obtained by interchanging M_i and H_i in the first scheme. The schemes (5) and (6) are believed to be secure in the sense that the complexities of free-start collision and preimage attacks are $2^{m/2}$ and 2^m, respectively.

Since most block ciphers have a block length of 64 bits, the complexity of a brute-force collision attack is only about 2^{32}. A second consideration is that 2^{64} off-line operations for a preimage attack are becoming more and more realistic [20]. In this context it is important to note that "Moore's law" states that the cost of computation reduces by a factor of four every 3 years.

3. Double Block Length Hash Functions

The previous section motivates the attempts which have been made to construct hash round functions based on two parallel or consecutive runs of a block cipher, yielding a $2m$-bit hash result. Natural requirements for these hash functions are that the complexity of a preimage attack is higher than 2^m and more importantly that the complexity of a collision attack is substantially higher than $2^{m/2}$.

One such scheme is MDC-2, which was developed by Brachtl et al. [1] for use in combination with DES; its generalization to an arbitrary block cipher is included in ISO/IEC 10118-2 [9]:

$$\begin{cases} T_i^1 = E_{H_{i-1}^1}(M_i) \oplus M_i = LT_i^1 \parallel RT_i^1, \\ T_i^2 = E_{H_{i-1}^2}(M_i) \oplus M_i = LT_i^2 \parallel RT_i^2, \\ H_i^1 = LT_i^1 \parallel RT_i^2, \\ H_i^2 = LT_i^2 \parallel RT_i^1. \end{cases} \tag{7}$$

It is believed that the complexities for preimage and collision attacks on MDC-2 are about $2^{3m/2}$ and 2^m, respectively. For DES, with a block size of 64 bits, these attacks require about 2^{81} and 2^{54} operations, since the effective key size in this construction is only 54 bits (two of the 56 key bits are fixed to make the two encryption functions different and to avoid weak DES-keys).

The hash rate of MDC-2 is only $\frac{1}{2}$, i.e., the hash function takes two encryptions per message block, which implies that MDC-2 is at least twice as slow as the underlying block cipher. In addition, each hash round function requires two key schedulings. Several attempts have been made to construct fast double block length hash functions with hash rate 1 [2], [8], [14], [16].

Consider the following general form of a double block length hash function:

$$\begin{cases} H_i^1 = E_A(B) \oplus C, \\ H_i^2 = E_R(S) \oplus T. \end{cases} \tag{8}$$

For a hash rate $\frac{1}{2}$ scheme, A, B, and C are binary linear combinations of the m-bit vectors H_{i-1}^1, H_{i-1}^2, and M_i, and R, S, and T are binary linear combinations of the vectors H_{i-1}^1, H_{i-1}^2, M_i, and H_i^1. For a hash rate 1 scheme, the message M is divided into blocks (M_i^1, M_i^2) of $l = 2m$ bits (each M_i^j of m bits), i.e., $M = (M_1^1, M_1^2, \ldots, M_n^1, M_n^2)$. A, B, and C are binary linear combinations of the m-bit vectors H_{i-1}^1, H_{i-1}^2, M_i^1, and M_i^2, and R, S, and T are binary linear combinations of the vectors H_{i-1}^1, H_{i-1}^2, M_i^1, M_i^2, and H_i^1. If H_i^1 and H_i^2 can be computed independently, the hash function is called *parallel*; if H_i^2 depends on H_i^1, the hash function is called *serial*.

This paper presents attacks on all hash functions of hash rate 1 defined by (8). In particular our attacks break the following three proposed schemes.

Parallel-DM [8] (see also Fig. 1)

$$\begin{cases} H_i^1 = E_{M_i^1 \oplus M_i^2}(H_{i-1}^1 \oplus M_i^1) \oplus H_{i-1}^1 \oplus M_i^1, \\ H_i^2 = E_{M_i^1}(H_{i-1}^2 \oplus M_i^2) \oplus H_{i-1}^2 \oplus M_i^2, \end{cases} \tag{9}$$

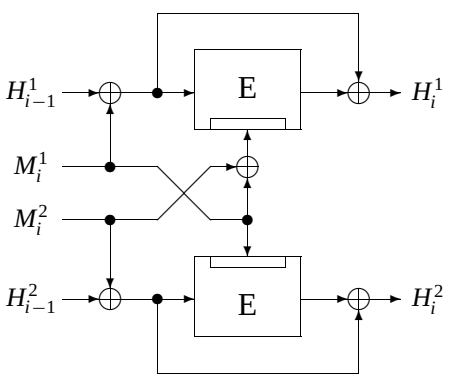

Fig. 1. The $2m$-bit round function of the Parallel-DM scheme.

The PBGV hash function [16]

$$\begin{cases} H_i^1 = E_{M_i^1 \oplus M_i^2}(H_{i-1}^1 \oplus H_{i-1}^2) \oplus M_i^1 \oplus H_{i-1}^1 \oplus H_{i-1}^2, \\ H_i^2 = E_{M_i^1 \oplus H_{i-1}^1}(M_i^2 \oplus H_{i-1}^2) \oplus M_i^2 \oplus H_{i-1}^1 \oplus H_{i-1}^2. \end{cases} \tag{10}$$

The LOKI DBH mode [2]

$$\begin{cases} H_i^1 = E_{H_{i-1}^2 \oplus M_i^2}(H_{i-1}^2 \oplus M_i^1) \oplus H_{i-1}^1 \oplus H_{i-1}^2 \oplus M_i^1, \\ H_i^2 = E_{H_{i-1}^1 \oplus M_i^1}(H_{i-1}^2 \oplus M_i^2 \oplus H_i^1) \oplus H_{i-1}^1 \oplus H_{i-1}^2 \oplus M_i^2. \end{cases} \tag{11}$$

The LOKI DBH mode is a variant of a scheme presented by Quisquater and Girault at Eurocrypt '89.

The following free-start attacks on double block length hash functions were given in [8]:

Theorem 2. *For the* $2m$*-bit iterated hash function with hash rate* $\frac{1}{2}$ *or* 1 *whose* $2m$*-bit round function is of type* (8)*, there exist a free-start preimage attack and a free-start collision attack with complexities about* 2×2^m *and* $2 \times 2^{m/2}$*, respectively.*

Also in [8], the following approach was suggested to the design of double block length hash functions: hash functions meeting these upper bounds for the free-start attacks are said to be *optimum* against a free-start attack. For such a hash function *optimum* security against free-start attacks can be obtained if the scheme is equivalent to either two runs of (5) or two runs of (6) by a simple invertible transformation of the inputs M_i^1, M_i^2, H_{i-1}^1, and H_{i-1}^2. The designer hopes that for the complete hash function, with a fixed initial value, the complexity of collision and preimage attacks are higher than the proven lower bounds for the free-start attacks. One example of such a construction is Parallel-DM (9).

The attacks presented in the remainder of this paper are attacks with a given, fixed initial value.

4. The Solving One-Half Attacks

In this section a general class of attacks is proposed which exploits the fact that one of the equations of (8) can be solved for the message blocks. First attacks on the parallel version are presented.

Theorem 3. *Consider a double block length hash function with round function of the form* (12)*, where each* h^i *contains one encryption*:

$$\begin{cases} H_i^1 = h^1(H_{i-1}^1, H_{i-1}^2, M_i^1, M_i^2), \\ H_i^2 = h^2(H_{i-1}^1, H_{i-1}^2, M_i^1, M_i^2). \end{cases} \tag{12}$$

If T *operations are required to find one pair of* (M_i^1, M_i^2) *for any given value of* (H_{i-1}^1, H_{i-1}^2)*, such that the resulting* 4*-tuple* $(H_{i-1}^1, H_{i-1}^2, M_i^1, M_i^2)$ *yields the fixed value*

for H_i^1 (or H_i^2 or $H_i^1 \oplus H_i^2$), there exist 2nd preimage and preimage attacks on the hash function with complexities about $(T+3) \times 2^m$; and there exists a collision attack on the hash function with complexity about $(T+3) \times 2^{m/2}$.

Proof. The attacks start by choosing arbitrary message blocks (M_i^1, M_i^2), for $i = 1, \ldots, n-2$, and by computing the values (H_{n-2}^1, H_{n-2}^2) forward from the given initial values (H_0^1, H_0^2). Then one searches for the four message blocks (M_{n-1}^1, M_{n-1}^2) and (M_n^1, M_n^2) such that the hash result is hit (in the case of a (2nd) preimage attack) or for a pair of four correcting blocks which yield a collision. The initial $2 \cdot (n-2)$ operations can be ignored in the complexity measurements, if $n \ll 2^m$.

The (2nd) preimage attack. Let (H_n^1, H_n^2) be the hash result of a message M', where $n \geq 2$. For a preimage attack, a message M yielding this hash result has to be found, and for the 2nd preimage attack M' is given and a message $M \neq M'$ yielding this hash result needs to be found. The attacker proceeds as follows:

1. Compute forward the pair (H_{n-1}^1, H_{n-1}^2) from the given values (H_{n-2}^1, H_{n-2}^2) and a pair of message blocks (M_{n-1}^1, M_{n-1}^2).
2. Find the pair (M_n^1, M_n^2) from the pair (H_{n-1}^1, H_{n-1}^2) obtained above so that the 4-tuple $(H_{n-1}^1, H_{n-1}^2, M_n^1, M_n^2)$ yields the given value of the hash result H_n^1.
3. Compute the value for H_n^2 from the 4-tuple $(H_{n-1}^1, H_{n-1}^2, M_n^1, M_n^2)$.

Repeat the above procedure 2^m times for different choices of (M_{n-1}^1, M_{n-1}^2). Since H_n^2 is m bits long, the probability of hitting the right value of H_n^2 is equal to 0.63, according to Lemma 1 with $z = 1$. Step 1 takes two operations, step 2 takes T operations, and step 3 takes one operation, in total $T+3$ operations. Note that if the mapping between M_{n-1}^1, M_{n-1}^2, or $M_{n-1}^1 \oplus M_{n-1}^2$ and H_n^2 is bijective, the choice of the message blocks in step 1 can be optimized such that the probability of success is equal to 1.

The collision attack. Two different sets of messages blocks $(M_{n-1}^1, M_{n-1}^2, M_n^1, M_n^2)$ and $(M'^1_{n-1}, M'^2_{n-1}, M'^1_n, M'^2_n)$ which produce the same hash result (H_n^1, H_n^2) have to be found. Choose a value for H_n^1 and proceed in the same way as in the preimage attack, i.e., perform steps 1–3 above. Repeat this procedure $2^{m/2}$ times. Since H_n^2 is m bits long, the probability of finding two different messages with equal values of H_n^2 is equal to 0.39, where Lemma 2 has been applied with $\lambda = \frac{1}{2}$. Note that in this case the message blocks have to be chosen such that the mapping between M_{n-1}^1, M_{n-1}^2, or $M_{n-1}^1 \oplus M_{n-1}^2$ and H_n^2 is not bijective. □

If the collision attack fails, the number of operations can be doubled; it follows from Lemma 2 that the success probability will increase to about 0.86 ($\lambda = 2$). Also observe that the collision attack can be extended to the case where the message blocks used in the first $n-2$ iterations are different for the colliding pair.

Proposition 1. *The attacks described in Theorem 3 can be implemented with only small memory.*

Proof. First note that the preimage attack requires only very little memory. The values of H_n^2 computed in step 3 can be immediately tested against the given value of H_n^2. For the collision attack $2^{m/2}$ values of H_n^2 can be collected and a match between them found. However, the method of [17] and [19] can be used to reduce the required memory. From there it follows that a nonbijective function f has to be defined. The values of H_n^2 computed in step 3 of the attack are images of a function F on input random values (M_{n-1}^1, M_{n-1}^2). Choose random values of $(M_{n-1}^1, M_{n-1}^2) = (r_1, r_2)$, and define $H_n^2(1) = f(r_1)$ and $H_n^2(j) = f(H_n^2(j-1))$, where $f(a) = F(a, r_2)$. In the rare cases where f defines a bijection, the attack will fail. However, there are many ways to construct f from F, and for a particular hash function it is easy to find such a nonbijective mapping. □

Theorem 3 can be extended to serial double block length hash functions as follows.

Theorem 4. *Consider a double block length hash function of hash rate* 1 *with round function of the form* (13), *where each* h^i *contains one encryption*:

$$\begin{cases} H_i^1 = h^1(H_{i-1}^1, H_{i-1}^2, M_i^1, M_i^2), \\ H_i^2 = h^2(H_{i-1}^1, H_{i-1}^2, M_i^1, M_i^2, H_i^1). \end{cases} \tag{13}$$

If T *operations are required to find one pair of* (M_i^1, M_i^2) *for any given value of* (H_{i-1}^1, H_{i-1}^2), *such that the resulting* 4*-tuple* $(H_{i-1}^1, H_{i-1}^2, M_i^1, M_i^2)$ *yields the fixed value for* H_i^1, *the complexities of* 2*nd preimage and preimage attacks on the hash function are about* $(T+3) \times 2^m$; *and the complexity of a collision attack on the hash function is about* $(T+3) \times 2^{m/2}$.

Note that in the collision attacks above (and in the rest of this paper) one of the two values of the hash result (H_n^1, H_n^2) can be chosen, while the second one will be a random value. That is, two messages with hash results (H_n^1, X) for given H_n^1 and random X can be found. Finding such a collision by combining a brute-force preimage attack and a birthday attack requires about $2^{3m/2}$ operations, compared with 2^m operations for a random collision. Thus, the collision attacks presented are more powerful than the standard ones.

5. Attacks on All Hash Rate 1 Schemes

In this section it is shown that for any double block length hash function of type (8) with hash rate 1, preimages and collisions can be found in time much less than by brute force. In particular this holds for the Parallel-DM (9), the PBGV hash function (10), and the LOKI DBH hash function (11).

The results apply to double block length hash functions for which H_i^1 (or H_i^2) can be written as

$$H_i^1 = E_A(B) \oplus C \qquad \text{with} \quad \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \mathcal{L} \cdot \begin{bmatrix} H_{i-1}^1 \\ H_{i-1}^2 \\ M_i^1 \\ M_i^2 \end{bmatrix}. \tag{14}$$

Here the vector elements are m-bit strings and $\mathcal{L}$ is a 3×4 block matrix with the $(m \times m)$ identity or zero matrix as blocks. The binary 3×4 matrix L is now defined as follows: when $\mathcal{L}_{ij}$ is an identity (resp. zero) matrix, L_{ij} is one (resp. zero). The rank of L denotes the formal rank of $\mathcal{L}$. Also, L_1 and L_2 denote the submatrices of L consisting of L_{ij} for $j = 1, 2$ and for $j = 3, 4$, respectively.

This description includes all double block length hash functions of hash rate 1 (serial or parallel), as defined in (8), but also more complex schemes. As an example, for LOKI DBH (11) one gets

$$L = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \end{bmatrix}.$$

Also, A and B of (14) will be rewritten as follows:

$$\begin{bmatrix} A \\ B \end{bmatrix} = \mathcal{N}_1 \cdot \begin{bmatrix} H_{i-1}^1 \\ H_{i-1}^2 \end{bmatrix} \oplus \mathcal{N}_2 \cdot \begin{bmatrix} M_i^1 \\ M_i^2 \end{bmatrix}, \tag{15}$$

where $\mathcal{N}_1$ and $\mathcal{N}_2$ are 2×2 binary submatrices of $\mathcal{L}$. Similarly, N_1 and N_2 are the binary matrices derived from $\mathcal{N}_1$ and $\mathcal{N}_2$, respectively.

Before stating the main result of the paper, a simple lemma is proved.

Lemma 3. *If the rank of L_2 is less than two, then there exist collision, 2nd preimage, and preimage attacks on the hash function with complexities $3 \times 2^{m/2}$, 3×2^m, and 4×2^m, respectively.*

Proof. If the rank of L_2 is at most one, H_i^1 depends on a subspace of $\langle M_i^1, M_i^2 \rangle$ of dimension at most one. Consider first the collision and (2nd) preimage attacks. It follows that an attacker has at least one degree of freedom to find 2^m values of M_i^1, M_i^2, or $M_i^1 \oplus M_i^2$ yielding the given value of the hash result H_n^1, thus Theorem 4 holds with $T = 0$. For the preimage attack an attacker would first have to perform a brute-force attack of complexity 2^m to find message blocks that hash to H_n^1. Subsequently, Theorem 4 holds with $T = 0$. □

Theorem 5. *For the double block length hash functions of hash rate 1, for which one of the round functions has the form of (8), there exist 2nd preimage and preimage attacks with complexities of about 4×2^m. Furthermore, there exists a collision attack with complexity of about $3 \times 2^{3m/4}$. For all but two classes of hash functions, there exists a collision attack with complexity of about $4 \times 2^{m/2}$.*

Proof. Consider the general form (8). In the following the hash round function H_i^1 will be attacked and the notation of (14) will be used. The proof is divided into three cases depending on the rank of L, denoted by $\mathrm{Rank}(L)$.

- $\mathrm{Rank}(L) = 1$

This implies that $\mathrm{Rank}(L_2) \leq 1$ and the result follows from Lemma 3.

- $\mathrm{Rank}(L) = 2$

It follows that one of the rows of L can be expressed as a linear combination of the

other two. Assume first that the third row can be expressed as a linear combination of the first two. From (15) it follows that $\text{Rank}(N_2) = \text{Rank}(L_2)$. If $\text{Rank}(N_2) = 1$ the result follows from Lemma 3. If $\text{Rank}(N_2) = 2$, then N_2 is invertible and from (15) one obtains

$$\begin{bmatrix} M_i^1 \\ M_i^2 \end{bmatrix} = N_2^{-1} \cdot \left[N_1 \cdot \begin{bmatrix} H_{i-1}^1 \\ H_{i-1}^2 \end{bmatrix} \oplus \begin{bmatrix} A \\ B \end{bmatrix} \right]. \tag{16}$$

For the 2nd preimage attack and the collision attack proceed as follows. For the given hash result H_i^1, let (a, b) be the values of (A, B) used in the computation of H_i^1. Using (16) with (a, b) for (A, B), one finds (M_i^1, M_i^2) for any values (H_{i-1}^1, H_{i-1}^2) and H_i^1. Thus, Theorem 4 holds with $T \simeq 0$ (it is assumed that the time for the additions is negligible), yielding a 2nd preimage in 3×2^m operations, and a collision in $3 \times 2^{m/2}$ operations.

For the preimage attack start with a brute-force attack on H_i^1, by trying 2^m values of (M_i^1, M_i^2). This requires 2^m operations and will have a success probability of 0.63 (or 1 if a bijective mapping can be found), according to Lemma 1. Continue as in the 2nd preimage attack. The overall complexity of the preimage attack is 4×2^m with a probability of success at least $(0.63)^2 \simeq 0.39$.

This completes the proof for the case where the third row of L can be expressed as a linear combination of the first two. Consider next the case where the second row can be expressed as a linear combination of the first and third rows. Express A and C in a similar way as A and B in (15) and define the submatrices N_1' and N_2' accordingly. It follows that the above proof holds also in this case and in the other cases where $\text{Rank}(L) = 2$.

The Parallel-DM (9) is an instance of this class of hash functions.

- $\text{Rank}(L) = 3$

Two cases are distinguished:

$\text{Rank}(N_2) \leq 1$. If $\text{Rank}(N_2) = 0$, then $\text{Rank}(L_2) = 1$ and the result follows from Lemma 3. If $\text{Rank}(N_2) = 1$, then A and B depend on a one-dimensional subspace of $\langle M_1, M_2 \rangle$. Let M_{AB} represent this subspace. Since it can be assumed that $\text{Rank}(L_2) = 2$ (if $\text{Rank}(L_2) < 2$, Lemma 3 applies again), C depends on a one-dimensional subspace of $\langle M_1, M_2 \rangle$ represented by M_C, where $M_C \neq M_{AB}$.

For any given (H_{i-1}^1, H_{i-1}^2), choose a value of M_{AB} and compute the corresponding values (a, b) of (A, B) and then of $z = E_a(b)$. Use the degree of freedom of M_C to calculate the value c of C such that H_i^1 is hit, i.e., such that $c \oplus z = H_i^1$. Now Theorem 4 holds with $T \simeq 1$.

The PBGV hash function (10) is an instance of this class of hash functions.

$\text{Rank}(N_2) = 2$. Since $\text{Rank}(L) = 3$, none of the rows of L can be written as a linear combination of two other rows. Suppose $\{\mathbf{a}, \mathbf{b}, \mathbf{c}\}$ is the standard basis for the image space of L. Then $\{\mathbf{a}, \mathbf{b}, \mathbf{c} \oplus \lambda\mathbf{a} \oplus \gamma\mathbf{b}\}$ is a basis for the image space for any $\lambda, \gamma = 0, 1$. Therefore λ, γ that yield a basis $\{\mathbf{a}, \mathbf{b}, \mathbf{c}'\}$ can be found (by elementary row operations) such that

$$\mathcal{L}' = \begin{pmatrix} N_1 & N_2 \\ * & 0 \end{pmatrix}.$$

Clearly, any C' is independent of M_i^1, M_i^2.

If $(\lambda, \gamma) = (0, 0)$, then $H_i^1 = E_A(B) \oplus C'$. Here the solving one-half attack of Theorem 4 applies with $T \simeq 1$. Indeed, first obtain the value c' of C' from the given values of (H_{i-1}^1, H_{i-1}^2). Then choose a value a of A, and compute the value b of B as $b = D_a(c' \oplus H_n^1)$. Now (16) can be used to find the corresponding values of (M_i^1, M_i^2); this is possible since $\text{Rank}(N_2) = 2$.

If $(\lambda, \gamma) = (1, 0)$, then $H_i^1 = E_A(B) \oplus A \oplus C'$. Similarly, first find the value c' of C', then choose a value a of A, and compute the value b of B from $b = D_a(c' \oplus H_n^1 \oplus a)$. Again Theorem 4 holds with $T \simeq 1$.

If $(\lambda, \gamma) = (0, 1)$, then $H_i^1 = E_A(B) \oplus B \oplus C'$. It is assumed without loss of generality that $C' = H_{n-1}^1$. For this class of hash functions, Theorem 4 does not apply. However, in the following meet-in-the-middle attacks are described which are faster than brute-force attacks. First the (2nd) preimage is given, and then the collision attack.

The attacks start by choosing arbitrary message blocks (M_i^1, M_i^2), for $i = 1, \ldots, n-2$, and by computing the values (H_{n-2}^1, H_{n-2}^2) forward from the given initial values (H_0^1, H_0^2). Then one searches for the four message blocks (M_{n-1}^1, M_{n-1}^2) and (M_n^1, M_n^2) such that the hash result is hit (in the case of a (2nd) preimage attack) or for a pair of four correcting blocks which yield a collision. The initial $2 \cdot (n-2)$ operations can be ignored in the complexity measurements if $n \ll 2^m$.

The (2*nd*) *preimage attack*:

1. Backward step: choose 2^m values (a, b) of (A, B) and compute
$$H_{n-1}'^1 = c' = E_a(b) \oplus b \oplus H_n^1$$
for the given value of H_n^1.
2. Forward step: choose 2^m values for $(M_{n-1}''^1, M_{n-1}''^2)$ and compute $(H_{n-1}''^1, H_{n-1}''^2)$ from (H_{n-2}^1, H_{n-2}^2).

Find matches $H_{n-1}''^1 = H_{n-1}'^1$. For every match use (16) to find the values of (M_n^1, M_n^2) from (a, b) and $(H_{n-1}''^1, H_{n-1}''^2)$ (note $\text{Rank}(N_2) = 2$). Finally compute the corresponding value of H_n^2. The quantities in the meet-in-middle attack are m bits long, so this gives about

$$\frac{2^m \times 2^m}{2^m} = 2^m$$

values of $(H_{n-1}^1, H_{n-1}^2, M_n^1, M_n^2)$ all hitting the same value of H_n^1. Thus, according to Lemma 1 messages hitting H_n^2 as well will be found with probability about 0.63; the total number of operations is about 4×2^m.

The collision attack:

1. Backward step: choose $2^{3m/4}$ values (a, b) for A and B and compute
$$H_{n-1}'^1 = c' = E_a(b) \oplus b \oplus H_n^1$$
for the given value of H_n^1.
2. Forward step: choose $2^{3m/4}$ values for $(M_{n-1}''^1, M_{n-1}''^2)$ and compute $(H_{n-1}''^1, H_{n-1}''^2)$ from (H_{n-2}^1, H_{n-2}^2).

Find matches $H''^1_{n-1} = H'^1_{n-1}$. For every match compute the values of (M^1_i, M^2_i) from (a, b) and $(H''^1_{n-1}, H''^2_{n-1})$ using (16) and the corresponding value of H^2_n. Since the quantities in the meet-in-middle attack are m bits long, this gives about

$$\frac{2^{3m/4} \times 2^{3m/4}}{2^m} = 2^{m/2}$$

values of $(H^1_{n-1}, H^2_{n-1}, M^1_n, M^2_n)$ all hitting the same value of H^1_n. Thus, according to Lemma 2 with $\lambda = \frac{1}{2}$, among these a match for H^2_n will be found with probability 0.39. The total number of operations is equal to $3 \times 2^{3m/4}$.

If $(\lambda, \gamma) = (1, 1)$, then $H^1_i = E_A(B) \oplus B \oplus C'$. Choose random values (a, b) of A and B and compute

$$c' = E_a(b) \oplus a \oplus b \oplus H^1_n.$$

Then proceed similarly as in the previous case.

The LOKI DBH hash function (11) is an instance of this class of hash functions. □

The following result can be deduced from the proof of Theorem 5:

Corollary 1. *For the* $2m$*-bit iterated hash function with hash rate* 1*, where* (*at least*) *one of* H^1_i *and* H^2_i *in the hash round function has the form of a* (*secure*) *single block length hash function, there exist* $2n$*d preimage attacks with complexities of about* 3×2^m*, preimage attacks with complexities of about* 4×2^m*, and collision attacks with complexities of about* $3 \times 2^{m/2}$.

Proof. Since all single block length hash functions have $\text{Rank}(L) \leq 2$, the result follows directly from the proof of Theorem 5. □

Corollary 1 implies that the method of [8] to construct schemes which are optimum against free-start attacks, such as the Parallel-DM (9), has a serious shortcoming: it will always result in schemes vulnerable to attacks with complexities similar to brute-force attacks on single block length hash functions.

The attacks presented in this paper can be applied also to schemes where the key length is less than the block length ($k < m$), such as DES [5]. Clearly, Theorem 4 also applies to such schemes, as nothing is assumed about the block and key length of the block cipher. Theorem 5 holds as well in that case. The hash round functions for such hash functions are of the form $H^1_i = E_{\tilde{A}}(B) \oplus C$, where $\tilde{A}$ is a vector, obtained from selecting k bits of A, and where A, B, and C are defined as in (14). It is easy to extend the above proof of Theorem 5 to take this into account. Also note that block ciphers for which $k > m$ (but $k \approx m$) yield more complex schemes, but with no significant improved security.

Proposition 2. *The attacks described in Theorem* 5 *can be implemented with only small memory.*

Proof. The cases where Theorem 4 was used were already proved in Proposition 1. It suffices to show the case where $\mathrm{Rank}(L) = 3$, $\mathrm{Rank}(N_2) = 2$, and $(\lambda, \gamma) = (0, 1)$. The other cases are treated similarly. A variant of the method of [17] and [19] can be used. The values of c', computed in step 1 of the attack, are images of a function F, i.e., $F(a, b)$, and the values of H''^1_{n-1}, computed in step 2 are images of a function G, i.e., $G(M''^1_{n-1}, M''^2_{n-1})$. Fix b, M''^1_{n-1}, and M''^2_{n-1} and let f and g be the functions obtained from F and G, respectively, with one of the two arguments fixed. Choose a random value r and define

$$\begin{aligned} c'(1) &= f(r), \\ H''^1_{n-1}(1) &= g(c'(1)), \\ c'(k) &= f(H''^1_{n-1}(k-1)), \\ H''^1_{n-1}(k) &= g(c'(k)). \end{aligned}$$

Whenever there is a match $f(j) = g(j')$ the values of H^1_{n-1}, H^2_{n-1}, M^1_n, and M^2_n can be computed such that the same given value of H^1_n is hit. For the preimage attack, immediately check whether the given value of H^2_n is hit as well. For the collision attack these values are stored until a sufficient number have been collected to find a match in the second chain as well. Alternatively, the methods of distinguished points can be applied one more time to these values. □

6. Conclusion

Attacks have been described on double block length hash functions, whose hash round functions use two encryptions of a block cipher to hash two message blocks. The main result is that for all double block length hash functions of this type, there exist preimage attacks and collision attacks with complexities much less than for brute-force attacks. Therefore, it is not possible to obtain a double block length hash function, as defined in this paper, which is at least as secure and faster than MDC-2. The attacks in this paper have been described for block ciphers with key size equal to the block length. They can be extended to the case where the key size is slightly larger or smaller than the block size such as DES. It is left as an open question whether it is possible to improve MDC-2 by considering either three encryptions in a double block length hash mode, or by constructing triple (or n-fold) block length hash functions with a hash rate greater than $\frac{1}{2}$.

Acknowledgments

We would like to thank the anonymous referees of this paper for constructive comments that improved the presentation of the results.

References

[1] B.O. Brachtl, D. Coppersmith, M.M. Hyden, S.M. Matyas, C.H. Meyer, J. Oseas, S. Pilpel, and M. Schilling. *Data Authentication Using Modification Detection Codes Based on a Public One Way Encryption Function*. U.S. Patent Number 4,908,861, March 13, 1990.

[2] L. Brown, J. Pieprzyk, and J. Seberry. LOKI—a cryptographic primitive for authentication and secrecy applications. In J. Seberry and J. Pieprzyk, editors, *Advances in Cryptology—Proc. AusCrypt* '90, LNCS 453, pages 229–236. Springer-Verlag, Berlin, 1990.

[3] I.B. Damgård. A design principle for hash functions. In G. Brassard, editor, *Advances in Cryptology—Proc. Crypto* '89, LNCS 435, pages 416–427. Springer-Verlag, Berlin, 1990.

[4] W. Feller. *An Introduction to Probability Theory and Its Applications*, Vol. 1. Wiley, New York, 1968.

[5] FIPS 46. *Data Encryption Standard*. Federal Information Processing Standard (FIPS), Publication 46, National Bureau of Standards, U.S. Department of Commerce, Washington, D.C., January 1977.

[6] P. Flajolet and A.M. Odlyzko. Random mapping statistics. In J.-J. Quisquater and J. Vandewalle, editors, *Advances in Cryptology—Eurocrypt* '89, LNCS 434, pages 329–354. Springer-Verlag, Berlin, 1990.

[7] M. Girault, R. Cohen, and M. Campana. A generalized birthday attack. In C.G. Günther, editor, *Advances in Cryptology—Eurocrypt'*88, *LNCS* 330, pages 129–156. Springer-Verlag, Berlin, 1988.

[8] W. Hohl, X. Lai, T. Meier, and C. Waldvogel. Security of iterated hash function based on block ciphers. In D.R. Stinson, editor, *Advances in Cryptology—Proc. Crypto* '93, LNCS 773, pages 379–390. Springer-Verlag, Berlin, 1993.

[9] ISO-10118. Information technology—Security techniques—Hash-functions, part 1: General and part 2: Hash-functions using an n-bit block cipher algorithm. ISO/IEC, 1994.

[10] ISO/IEC JTC1 SC27/WG2/N98. Hash functions using a pseudorandom algorithm. Japanese contribution, 1991.

[11] L.R. Knudsen. Block Ciphers—Analysis, Design and Applications. Ph.D. thesis, Aarhus University, 1994.

[12] X. Lai. *On the Design and Security of Block Ciphers*. ETH Series in Information Processing (J.L. Massey, editor), Vol. 1. Hartung-Gorre Verlag, Konstanz, 1992.

[13] R. Merkle. One way hash functions and DES. In G. Brassard, editor, *Advances in Cryptology—Crypto* '89, LNCS 435, pages 428–446. Springer-Verlag, Berlin, 1990.

[14] B. Preneel. Analysis and Design of Cryptographic Hash Functions. Ph.D. thesis, Katholieke Universiteit Leuven, January 1993.

[15] B. Preneel. Hash functions based on block ciphers: A synthetic approach. In D.R. Stinson, editor, *Advances in Cryptology—Proc. Crypto* '93, LNCS 773, pages 368–378. Springer-Verlag, Berlin, 1993.

[16] B. Preneel, A. Bosselaers, R. Govaerts, and J. Vandewalle. Collision-free hashfunctions based on block-cipher algorithms. In *Proceedings of* 1989 *International Carnahan Conference on Security Technology*, pages 203–210, 1989.

[17] J.-J. Quisquater and J.-P. Delescaille. How easy is collision search. Applications to DES. In J.-J. Quisquater and J. Vandewalle, editors, *Advances in Cryptology—Eurocrypt* '89, LNCS 434, pages 429–433. Springer-Verlag, Berlin, 1990.

[18] R. Sedgewick, T.G. Szymanski, and A.C. Yao. The complexity of finding cycles in periodic functions. *SIAM Journal of Computing*, 11:376–390, 1982.

[19] P.C. van Oorschot and M.J. Wiener. Parallel collision search with cryptanalytic applications. *Journal of Cryptology*. To appear.

[20] M.J. Wiener. Efficient DES Key Search. Technical Report TR-244, School of Computer Science, Carleton University, Ottawa, Ontario. May 1994. Presented at the rump session of Crypto '93.