Fast and Secure Hashing Based on Codes

Lars Knudsen and Bart Preneel*

Katholieke Universiteit Leuven, Dept. Electrical Engineering-ESAT,
Kardinaal Mercierlaan 94, B-3001 Heverlee, Belgium
{lars.knudsen,bart.preneel}@esat.kuleuven.ac.be

Abstract. This paper considers hash functions based on block ciphers. It presents a new attack on the compression function of the 128-bit hash function MDC-4 using DES with a complexity far less that one would expect, and proposes new constructions of fast and secure compression functions based on error-correcting codes and m-bit block ciphers with an m-bit key. This leads to simple and practical hash function constructions based on block ciphers such as DES, where the key size is slightly smaller than the block size, IDEA, where the key size is twice the block size and to MD4-like hash functions. Under reasonable assumptions about the underlying block cipher, we obtain collision resistant compression functions. Finally we provide examples of hashing constructions based on both DES and IDEA more efficient than previous proposals and discuss applications of our approach for MD4-like hash functions.

1 Introduction

Cryptographic *hash functions* map a string of arbitrary size to a short string of fixed length, typically 128 or 160 bits. Hash functions are used in cryptographic applications such as digital signatures, password protection schemes, and conventional message authentication. For these applications one requires that it is hard to find an input corresponding to a given hash result (preimage resistance) or a second input with the same hash result as a given input (2nd preimage resistance). Moreover, many applications also require that it is hard to find two inputs with the same hash result (collision resistance). For the remainder of this paper we consider hash functions satisfying these three properties.

All important hash functions are *iterated hash functions* based on an easily computable compression function $h(\cdot,\cdot)$ from two binary sequences of respective lengths m and l to a binary sequence of length m. The message M is split into blocks M_i of l bits, $M = (M_1, M_2, \ldots, M_n)$. If the length of M is not a multiple of l, M is padded using an unambiguous padding rule. The *hash result* $\mathrm{Hash}(IV, M) = H = H_t$ is obtained by computing iteratively

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

* F.W.O. postdoctoral researcher, sponsored by the Fund for Scientific Research, Flanders (Belgium).

where $H_0 = IV$ is a specified *initial value*. Collision attacks, 2nd preimage attacks, and preimage attacks can be applied to both the compression function and the hash function. For the remainder of this paper we make no distinction between the latter two attacks and refer to both of them as preimage attacks.

It is possible to relate the security of $\mathrm{Hash}(\cdot)$ to that of $h(\cdot,\cdot)$ in several models [2, 12, 15, 17]. To do this, one needs to append an additional block at the end of the input string containing its length, known as MD-strengthening (after Merkle [15] and Damgård [2]), leading to the following result.

Theorem 1. *Let* $\mathrm{Hash}(\cdot)$ *be an iterated hash function with MD-strengthening. Then preimage and collision attacks on* $\mathrm{Hash}(\cdot,\cdot)$ *(where an attacker can choose IV freely) have roughly the same complexity as the corresponding attacks on* $h(\cdot,\cdot)$.

Theorem 1 gives a lower bound on the security of $\mathrm{Hash}(IV,\cdot)$. Most practical hash functions do not have a collision resistant compression function and do not treat the two inputs of the compression function in the same way; exceptions are the DES based hash functions of Merkle [15] and those of the authors [11]. As an example, collisions for the compression function of MD5 have been presented in [3, 6].

We consider hash functions based on block ciphers with block length m bits and key length k bits. For convenience we will assume that k is an integer multiple of m. Such a block cipher defines, for each k-bit key, a permutation on m-bit strings. The advantage of constructing hash functions based on block ciphers is the minimization of the design and implementation effort. Moreover, the security of the block cipher can, to a certain extent, be transferred to hash functions. One difference is that the block cipher has to satisfy certain properties even if the key is known (see for example [19]). The main disadvantage of the block cipher approach is that customized hash functions are likely to be more efficient. However, Dobbertin's attacks [4, 6] on specific hash functions such as MD4 [21] and MD5 [22] have shown that this efficiency can come at a high cost, which makes the block cipher constructions more attractive.

We define the *hash rate* of a hash function based on an m-bit block cipher 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 an attacker to succeed with a high probability.

The first secure constructions for a hash function based on a block cipher were the scheme by Matyas, Meyer, and Oseas [14], which has been specified in ISO/IEC 10118 [9], and its dual, widely known as the Davies-Meyer scheme.[2] Both schemes are *single block length hash functions* giving a hash result of only m bits with hash rate 1. A classification of these schemes has been presented by Preneel et al. in [18]. Using a birthday attack collisions for such a scheme can be found in about $2^{m/2}$ operations and a preimage in about 2^m operations. However, since most block ciphers have a block length of $m = 64$ bits, collisions can be found in only 2^{32} operations and hash functions with a larger hash result

[2] The real inventors are probably S.M. Matyas and C.H. Meyer.

are needed. The aim of *double block length* hash functions is first of all to achieve a security level against collision attacks of at least 2^m encryptions. For all constructions with rate 1 of a large class the authors have shown that collisions (for H) can be found in no more than $2^{3m/4}$ operations [11]. Using DES the best known constructions are MDC-2 of rate 1/2 [1, 9], and MDC-4 of rate 1/4 [1]; for both a collision attack is believed to require at least 2^m operations. Another important class of constructions are those of Merkle [15]. In [12], Lai and Massey have proposed two constructions for hash functions based on their block cipher IDEA (with $m = 64$ and $k = 128$): Abreast-DM and Tandem-DM have hash rate 1/2 and a claimed security level against collision attacks of 2^m operations.

However, for all these constructions, except for those of Merkle, there is no proof of security and for MDC-2 and MDC-4 collision for the compression function can be found faster than by a brute force attack. Moreover, for block ciphers with $m = 64$ (e.g., DES and IDEA), the double block constructions do not provide an acceptable security level against parallel brute force collision attacks: it follows from [20, 23] that a security level of at least $2^{75} \ldots 2^{80}$ encryptions is required, which is not offered by any of the current proposals. In [11] we developed a new framework for constructing hash functions based on m-bit block ciphers with m-bit keys using linear codes over $GF(2^2)$. Constructions were shown for which finding a collision requires at least $2^{qm/2}$ encryptions (with $q \geq 2$), and finding a preimage requires at least 2^{qm} encryptions at the cost of more internal memory. For $q = 2$, the constructions are more efficient than existing proposals with the same security level.

In this paper we first show that collisions for the compression function of MDC-4 can be found in time $2^{3m/4}$. For DES some key bits have to be fixed to avoid weak keys and the complementation property, hence the complexity of our attack is only 2^{41}, which is quite feasible today. Since the hash rate of MDC-4 is 1/4, i.e., only one block is processed per four encryptions, one should expect a higher level of security for the compression function. Second we generalize the approach of [11]. We propose a method for constructing hash functions based on block ciphers with larger keys and working on smaller blocks. More precisely, we consider m-bit block ciphers with tm-bit keys, $t \geq 1$, using linear codes over $GF(2^s)$, $s \geq 2$. Our constructions result in more efficient hash functions than those of [11]. Using DES and IDEA we show that it is possible to obtain hash rates close to 1 respectively 2 with a high security level against collision attacks.

In the following $E_K(X)$ and $D_K(Y)$ denote the encryption and decryption of the m-bit plaintext X respectively ciphertext Y under the tm-bit key K. It will be assumed that the block cipher has no weaknesses, i.e., that in attacks on the hash functions based on the block cipher, no short-cut attacks on the block cipher will help an attacker.

The remainder of this paper is organized as follows. In §2 we analyze the compression function of MDC-4. Our new constructions are presented in §3, and in §4 we provide examples of efficient constructions using DES, IDEA, and the MD4-like functions, hereafter denoted the *MDx family*.

2 The Compression Function of MDC-4

MDC-2 and MDC-4 [1] are hash functions based on a block cipher; they are also known as the Meyer-Schilling hash functions after the authors of the first paper describing these schemes [16]. MDC-2 is included in ISO/IEC 10118 Part 2, standardizing a hash function based on a block cipher [9]. MDC-2 can be described as follows, where $h(X,Y) = E_X(Y) \oplus Y$ and $E_K(\cdot)$ denotes encryption with an m-bit block cipher with key K:

$$
\begin{aligned}
T1_i &= h(u(H1_{i-1}), X_i) = LT1_i \parallel RT1_i \\
T2_i &= h(v(H2_{i-1}), X_i) = LT2_i \parallel RT2_i \\
H1_i &= LT1_i \parallel RT2_i \\
H2_i &= LT2_i \parallel RT1_i \,.
\end{aligned}
$$

The variables $H1_0$ and $H2_0$ are initialized with the values IV_1 and IV_2 respectively, and the hash result is equal to the concatenation of $H1_t$ and $H2_t$. The rate of this scheme is equal to 1/2. The ISO/IEC standard does not specify any particular block cipher; it also requires the specification of two mappings u, v from the ciphertext space to the key space such that $u(IV_1) \neq v(IV_2)$. For DES, u and v omit the parity bits and fix the second and third bit to 10 and 01 respectively, to preclude attacks based on (semi-)weak keys. The compression function of MDC-2 is certainly not collision resistant: for any fixed choice of X_i, one can find collisions for both $H1_i$ and $H2_i$ independently with a simple birthday attack requiring about $2^{m/2}$ operations.

One iteration of MDC-4 [1] is defined as a concatenation of two MDC-2 steps, where the plaintexts in the second step are equal to $H2_{i-1}$ and $H1_{i-1}$:

$$
\begin{aligned}
T1_i &= h(u(H1_{i-1}), X_i) = LT1_i \parallel RT1_i \\
T2_i &= h(v(H2_{i-1}), X_i) = LT2_i \parallel RT2_i \\
U1_i &= LT1_i \parallel RT2_i \\
U2_i &= LT2_i \parallel RT1_i \\
V1_i &= h(u(U1_i), H2_{i-1}) = LV1_i \parallel RV1_i \\
V2_i &= h(v(U2_i), H1_{i-1}) = LV2_i \parallel RV2_i \\
H1_i &= LV1_i \parallel RV2_i \\
H2_i &= LV2_i \parallel RV1_i \,.
\end{aligned}
$$

The rate of MDC-4 is equal to 1/4. It is clear that the "exchange" of $H1_{i-1}$ and $H2_{i-1}$ in the second step does not improve the algorithm: after the exchange of right halves, $U1_i$ and $U2_i$ are symmetric with respect to $H1_{i-1}$ and $H2_{i-1}$.

Finding a collision for the compression function is harder than in the case of MDC-2. On the other hand, collisions for the compression function of MDC-2 with different values of X_i and with the same value of $(H1_{i-1}, H2_{i-1})$ will also yield collisions for MDC-4. but generally this property does not hold for other

collisions for the basic function like pseudo-collisions, i.e., collisions where all the inputs to the compression function are varied.

In the following we will show a collision attack on the compression function with a complexity smaller than a brute-force attack:

1. Choose a random value of X_i and of $H2_{i-1}$ (with $i = 1$ this can be the specified initial value).
2. Calculate $T1_i$ for $2^{3m/4}$ values of $H1_{i-1}$. One expects to find an $m/2$-bit string S such that for a set of $2^{m/4}$ values the relevant bits of LT_i are equal to S (in fact for 50% of the strings S there will be $2^{m/4}$ cases or more).
3. Compute $V2_i$ for the $2^{m/4}$ inputs in this set. The probability of obtaining a collision for $V2_i$ is equal to $\left(2^{m/4}\right)^2 / 2^{m+1} = 2^{-m/2+1}$.
4. If no match is obtained, one chooses a new value for S; one can avoid recomputing $T1_i$ if one stores $2^{m/4} \cdot 2^{m/2} = 2^{3m/4}$ m-bit quantities.

The expected effort to find a collision for the compression function of MDC-4 is $2^{3m/4}$ encryptions and the storage of $2^{3m/4}$ m-bit quantities. The keys of MDC-4 using DES are effectively only 54 bits long [1]. In that case the collision attack requires about 2^{41} DES encryptions and a storage of 2^{30} 5 54-bit quantities (about 10 Gigabyte).

3 Constructions with Linear Codes

In this section we present new constructions for collision resistant hash functions based on an m-bit block cipher and linear codes. These constructions extend a simple hash mode which is believed to be secure to a multiple hash mode. As the simple hash mode we will use the Davies-Meyer hash function:

$$h_i(M_i, H_{i-1}) = E_{M_i}(H_{i-1}) \oplus H_{i-1}. \tag{2}$$

Any of the 12 secure single block length hash functions described in [18] can be used. The advantage of using the Davies-Meyer hash function is that it is defined for block ciphers with different block and key sizes. The following assumption is standard in cryptography today.

Assumption 1 *Let $E_K(.)$ be an m-bit block cipher with a tm-bit key K for an integer $t > 0$. Define the compression function h to be the Davies-Meyer function (2). Then finding collisions for h requires about $2^{m/2}$ encryptions (of an m-bit block), and finding a preimage for h requires about 2^m encryptions.*

Definition 2 (Multiple Davies-Meyer). Let $E_K(.)$ be an m-bit block cipher with a tm-bit key K for integer $t > 0$. Let $h_1, h_2, \ldots h_n$ be different instantiations of the Davies-Meyer function, that is, $h_i(X_i, Y_i) = E_{X_i}(Y_i) \oplus Y_i$, obtained by fixing $\lceil \log_2 n \rceil$ key (or plaintext) bits to different values, where the X_i's are tm-bit string and the Y_i's are m-bit strings. The compression function of a multiple Davies-Meyer scheme takes r m-bit input blocks, which are expanded by an affine mapping to the n pairs (X_i, Y_i). The output is the concatenation of the

outputs of the function $h_1, \ldots, h_n$. The output of the compression function shall depend on all r input blocks, in other words, the matrix of the affine mapping has full rank.

A subfunction $h_i(X_i, Y_i)$ is called *active*, if in a collision or preimage attack the input blocks forming (X_i, Y_i) are different. Two subfunctions $h_i(X_i, Y_i)$ and $h_j(X_j, Y_j)$ can be attacked *independently*, if one can vary one (or more) input blocks to the compression function, such that the arguments (X_i, Y_i) of h_i vary, while the arguments (X_j, Y_j) of h_j do not.

We make the following assumption, which is a slightly improved version of the assumption in [11].

Assumption 2 *Assume that a collision or preimage for the compression function of a multiple Davies-Meyer scheme has been found, that is, simultaneously for h_1, h_2, $\ldots h_n$. Let N be the number of active subfunctions and let $N - v$ be the maximum number of the N subfunctions that can be attacked independently. Then it is assumed that obtaining this collision or preimage must have required at least $2^{vm/2}$ respectively 2^{vm} encryptions.*

Note that in an attempt to find collisions or preimages for a multiple Davies-Meyer scheme it will always be possible to fix some input blocks and thereby fix the outputs. Let N denote the number of active subfunctions. What Assumption 2 says is, that if $N - v$ of these N functions can be attacked independently (separately), then there exists no better attack than a brute force attack on the remaining v subfunctions. Note that in the overall complexity of the collision (or the preimage) attack we do not consider the complexity of the attack on the $N - v$ functions, which makes our assumption strong and plausible.

3.1 The new constructions

The following theorem shows how hash functions based on block ciphers can be constructed using non-binary linear error correcting codes. It extends the main theorem of [11].

Theorem 3. *If there exists an $[n, k, d]$ code over $GF(2^{t+1})$ of length n, dimension k, and minimum distance d, with $(t+1)k > n$, for $t \geq 1$ and $m \gg \log_2 n$, then there exists a parallel hash function based on an m-bit block cipher with a tm-bit key for which finding a collision for the compression function requires at least $2^{(d-1)m/2}$ encryptions and finding a preimage requires at least $2^{(d-1)m}$ encryptions provided that Assumption 2 holds. The hash function has an internal memory of $n \cdot m$ bits, and a rate of $(t+1)\frac{k}{n} - 1$.*

Proof: The compression function consists of n different functions h_i with $1 \leq i \leq n$, see Definition 2. The input to the compression function consists of $(t+1)k$ m-bit blocks: the n variables H_{i-1}^1 through H_{i-1}^n (the output of the n functions of the previous iteration) and r message blocks M_i^1 through M_i^r, with $r = (t+1)k - n > 0$. In the following, every individual bit of these m-bit

blocks is treated in the same way. The bits of $(t+1)$ consecutive input blocks are concatenated yielding k elements of $GF(2^{t+1})$. These elements are encoded using the $[n,k,d]$ code, resulting in n elements of $GF(2^{t+1})$. Each of these elements represents the $(t+1)$-bit inputs to one of the n functions, that is, one bit represents the plaintext block input and the remaining t bits represent the key input to the block cipher. The individual input bits are obtained by representing the elements of $GF(2^{t+1})$ as a vector space over $GF(2)$. This construction guarantees that the conditions for Assumption 2 are satisfied for the value $v = d-1$. It follows from the minimum distance of the code that at least d subfunctions are active in a collision. Also, k is the maximum number of subfunctions, which can be attacked independently. But since $n-k \geq d-1$ by the Singleton bound [13, p. 33] the result follows. ∎

The existence of efficient constructions for $d=3$ and $d=4$ follows from the existence of perfect Hamming codes over GF(2^t) (see e.g., [13]) with parameters $n = (q^s-1)/(q-1)$, $k = n-s$, $d=3$ for a prime power q, and from the existence of triply extended MDS (maximum distance separable) codes [13, Ch. 11, Th. 10] with parameters $n = q^s+2$, $k = q^s-1$, $d=4$ for an even prime power q.

Corollary 4. *Provided that Assumption 2 holds, there exist parallel hash functions based on an m-bit block cipher with a tm-bit key of rate close to t for which finding a collision (a preimage) takes at least* 2^m *(*2^{2m}*) operations respectively at least* $2^{3m/2}$ *(*2^{3m}*) operations.*

Proof: From Theorem 3 it follows that there exist hash functions with rates $(t+1)k/n-1$. But since for Hamming codes $n/k \to 1$ for large values of n, the result follows. ∎

This result implies that at the cost of a larger internal memory, using DES one can obtain hash functions of rate 1, and using IDEA one can obtain hash functions of rate 2. For comparison MDC-2 and MDC-4 have rates 1/2 respectively 1/4, and Abreast-DM and Tandem-DM developed for IDEA [12] have rates 1/2.

In [11] we gave an example of a hash function based on an m-bit block cipher with an m-bit key using the code $[8,5,3]$ over $GF(2^2)$. The code is obtained by shortening the Hamming code $[21,18,3]$. The hash function has rate 1/4 and an internal memory of $8 \cdot m$ bits. In the following we show that one can improve this result.

The idea is to divide the m-bit words into smaller blocks and to use codes over bigger fields. As an example, assume we have a block cipher with m-bit blocks and m-bit keys for even m. In Theorem 3 codes over $GF(2^2)$ are used, where the two bits of the code words represent the plaintext respectively the key inputs to the block ciphers. An alternative method is to divide all m-bit blocks into blocks of $m/2$ bits and use codes over $GF(2^4)$. The advantage of this approach, which will be illustrated later, is that better bounds exist for such codes. This leads to the following generalization.

Theorem 5. *Let m be a multiple of b, i.e., $m = b \cdot m_b$. If there exists an $[n, k, d]$ code over $GF(2^{b(t+1)})$ of length n, dimension k, and minimum distance d, with $(t+1)k > n$, and $m \gg \log_2 n$, then there exists a parallel hash function based on an m-bit block cipher with a tm-bit key for which finding a collision for the compression function requires at least $2^{(d-1)m/2}$ encryptions provided that Assumption 2 holds. The hash function has an internal memory of $n \cdot m$ bits, has a rate of $(t+1)\frac{k}{n} - 1$ and works on m_b-bit blocks.*

Proof: Similar to that of Theorem 3. ∎

As an example, we can construct a hash function based on an m-bit block cipher with an m-bit key by using the code $[8, 6, 3]$ over $GF(2^4)$, which is obtained by shortening the Hamming code $[17, 15, 3]$. The hash function has rate $1/2$ and an internal memory of $8 \cdot m$ bits, and is thus twice as fast as the example using the code $[8, 5, 3]$ mentioned above. With $m = 64$ this construction operates on 32-bit words. One can extend this approach to construct hash functions with codes over $GF(2^8)$, i.e., operating on 16-bit words, for example by shortening the $[257, 255, 3]$ Hamming code ($s = 2$). However, since a $[17, 15, 3]$ over $GF(2^4)$ exists, the construction over $GF(2^8)$ is only more efficient for $n > 17$. Using an $[n, k, d]$-code requires $n \cdot m$ bits of internal memory, which make the constructions less attractive for larger n. Moreover, the codes obtained from splitting the blocks into smaller words result in a more complex implementation.

Notes:

1. Apart from the simple security proof and the relatively high rates, the schemes have the advantage that the n encryptions can be carried out in parallel.
2. The disadvantages of the schemes are the increased amount of internal memory and the cost of the code implementation (mainly some exclusive ors).
3. For the preimage attack, the security bounds assume that the entropy of the unknown part of the input is at least $(d-1)m$ bits.

As for (3), it is clear that if $D < (d-1)m$ bits are unknown to the attacker, a brute force preimage attack can be done in about 2^D encryptions.

3.2 Output transformation

The constructions presented above have the following problems:

1. since every output bit does not depend on all input bits of the compression function, it is relatively easy to find many inputs for which several output blocks of the compression function are equal,
2. the number of output blocks is typically much larger than the security level suggests.

The solution is to apply an output transformation to the outputs of the compression function. This transformation can be slow, since it has to be applied only once. Therefore there are many possible constructions.

First we present an approach that does not affect the provable security of the compression functions. One encrypts the n output blocks of the compression function using the block cipher with a fixed key, such that all output blocks of the encryption depend on all input blocks in a complicated way. Subsequently, the n blocks concatenated with the n encrypted blocks are hashed using the following construction. Denote with $n_{\min}$ the smallest possible value of n for a given value of d, such that Theorem 5 holds. Compress the $2n$ blocks to $n_{\min}$ blocks using the new construction with $n_{\min}$ parallel blocks (if $n_{\min} < n$, this hash function will have a lower rate than the original one). This approach solves the first problem and partly the second problem. However, if a further reduction to less than $n_{\min}$ blocks is required, other approaches are necessary.

We present next a generic approach for all values of n, which can be used instead of or in conjunction with the first approach. First one constructs from the m-bit block cipher a large, strong block cipher with block length $n \cdot m$ bits. This block cipher can be slow, since it is applied only once. Subsequently, the n ($n_{\min}$) blocks from the compression function are input to a Davies-Meyer construction where the block cipher key is randomly chosen and fixed (and part of the hash function description). Under Assumption 1 this is a secure hash function. The output can be truncated to any s blocks, where $s \geq d - 1$.

We give here an example of such a construction. One iteration consists of the following two steps. First, permute the blocks, such that block i becomes block $i+1$ and the last block becomes the first block. Second, encrypt the input blocks using the small block cipher in CBC mode. More precisely, denote the output of the compression function with $H^1, \ldots, H^n$. Let K_i for $i = 1, \ldots, n$ be n randomly chosen and fixed keys, and let $C^i = H^i$ for $i = 1, \ldots, n$. Repeat the following procedure r times:

$$
\begin{aligned}
&C^0 = C^n \\
&C^i = C^{i-1} \text{ for } i = 1, \ldots, n, \\
&C^i = E_{K_i}(C^i \oplus C^{i-1}) \text{ for } i = 1, \ldots, n
\end{aligned}
$$

(Here we use the same n keys in every iteration. Alternatively, different keys can be used in all rounds. Also the block permutation in the first iteration can be omitted.) After one iteration the last block will depend on all input blocks and in general, block i depends on i input blocks. After two iterations all blocks depend on all input blocks. However, two rounds are insufficient to make a strong block cipher. Therefore we recommend that this block cipher is used with at least $r = n$ rounds. Finally, the result of the output transformation is defined as the concatenation of the blocks $H^i \oplus C^i$ for $i = 1, \ldots, n$. If the output is truncated we recommend that the final output is formed from the blocks with the highest indices. With r rounds the output transformation requires r^2 encryptions of the small block cipher.

In the next section we give some practical examples.

4 Some Practical Examples

This section contains some examples of new constructions for different parameters of the underlying block cipher. We will use the notation (m, k) for an m-bit block cipher with a k-bit key. In the examples to come we use Hamming codes with $d = 3$ and MDS codes with $d = 4$. The existence of the latter codes follows from [13, Ch. 11, Th. 10].

4.1 Using an (m, m)-block cipher

Table 1. Rates and complexities of previous proposals for (m, m)-block ciphers.

Scheme	Rate	Collision h	Collision H	Reference
MDC-2	1/2	$2^{m/2}$	2^m	[1]
MDC-4	1/4	$2^{3m/4}$	2^m	[1]
Merkle	0.27	2^m	2^m	[15]

Table 2. Comparison of constructions based on codes over $GF(2^2)$ and over $GF(2^4)$ for (m, m)-block ciphers.

$GF(2^2)$		$GF(2^4)$		Collision
Code	Rate	Code	Rate	
$[5, 3, 3]$	1/5	$[6, 4, 3]$	1/4	$\geq 2^m$
$[8, 5, 3]$	1/4	$[8, 6, 3]$	1/2	$\geq 2^m$
$[12, 9, 3]$	1/2	$[12, 10, 3]$	2/3	$\geq 2^m$
$[9, 5, 4]$	1/9	$[9, 6, 4]$	1/3	$\geq 2^{3m/2}$
$[16, 12, 4]$	1/2	$[16, 13, 4]$	5/8	$\geq 2^{3m/2}$

Note that the 2^m complexities of MDC-2 and MDC-4 are the best *known* attacks against the hash function, while against Merkle's scheme and the schemes of Table 2 the 2^m complexity is against the compression functions and is a lower bound for the complexity of an attack on the hash function.

In the following we show an implementation of the construction using the code $[9, 6, 4]$. We define $GF(2^4)$ as the extension field $GF(2)[x]/(x^4 + x + 1)$. There are many generator matrices for a $[9, 6, 4]$ linear code over GF(2^4). We have searched for a generator matrix which leads to a simple and efficient compression function, as explained below. The generator matrix has the following form:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & \alpha & \beta \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & \beta & \alpha \\ 0 & 0 & 0 & 1 & 0 & 0 & \alpha & 1 & \beta \\ 0 & 0 & 0 & 0 & 1 & 0 & \alpha & \beta & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & \beta & 1 & \alpha \end{bmatrix} \tag{3}$$

Here 0 and 1 are the additive and multiplicative neutral elements in $GF(2^4)$ and $\alpha = x$, and $\beta = x^3+x^2+1$. The motivation for the choice of the generator matrix is as follows. In an implementation of the compression functions the elements of $GF(2^4)$ are represented as elements of a vector space over $GF(2)$. Clearly, multiplications with 0 and 1 are the easiest to implement. A closer analysis shows that multiplication with α and β in the above example can be implemented with one respectively two exclusive-ors. An exhaustive search for the matrix with the easiest implementation is clearly not feasible, but by restricting ourselves to using the elements 0, 1, α, and β we obtain a solution close to the optimal one.

Let $f_i(X,Y)$ be different instantiations of the function $E_X(Y) \oplus Y$, let X_L and X_R denote the leftmost respectively rightmost $m/2$ bits of X and let $\|$ denote concatenation of $m/2$ bit blocks. Furthermore, let $G^1, \ldots, G^9$ be the 9 input blocks coming from the compression function in the previous iteration and let M^1, M^2, M^3 be the 3 message block inputs. This results in the following compression function:

$$
\begin{aligned}
H^1 &= f_1(G^1, G^2) \\
H^2 &= f_2(G^3, G^4) \\
H^3 &= f_3(G^5, G^6) \\
H^4 &= f_4(G^7, G^8) \\
H^5 &= f_5(G^9, M^1) \\
H^6 &= f_6(M^2, M^3) \\
H^7 &= f_7(G^1 \oplus G^3 \oplus G^5 \oplus (G^7_R \| G^8_L) \oplus (G^9_R \| M^1_L) \oplus (M^3_{LR} \| M^3_R), \\
&\quad G^2 \oplus G^4 \oplus G^6 \oplus (G^7_L \oplus G^8_R \| G^7_L) \oplus (G^9_L \oplus M^1_R \| G^9_L) \oplus (M^2_L \| M^2_R \oplus M^3_{LR})) \\
H^8 &= f_8(G^1 \oplus G^7 \oplus M^2 \oplus (G^3_R \| G^4_L) \oplus (G^6_{LR} \| G^6_R) \oplus (M^1_{LR} \| M^1_R), \\
&\quad G^2 \oplus G^8 \oplus M^3 \oplus (G^4_R \oplus G^3_L \| G^3_L) \oplus \mathcal{G}^5 \oplus (G^9_L \| G^9_R \oplus M^1_{LR})) \\
H^9 &= f_9(G^1 \oplus G^9 \oplus (G^4_{LR} \| G^4_R) \oplus (G^5_R \| G^6_L) \oplus (G^8_{LR} \| G^8_R) \oplus (M^2_R \| M^3_L), \\
&\quad G^2 \oplus M^1 \oplus \mathcal{G}^3 \oplus (G^6_R \oplus G^5_L \| G^5_L) \oplus \mathcal{G}^7 \oplus (M^2_L \oplus M^3_R \| M^2_L)),
\end{aligned}
$$

where $\mathcal{G}^t = (G^t_L \| G^t_R \oplus G^{t+1}_L \oplus G^{t+1}_R)$ and $X_{LR} = X_L \oplus X_R$.

As an output transformation we suggest to first hash the 9 blocks to 7 blocks via the compression function using the code $[7,4,4]$ ($n_{\min} = 7$ for $d = 4$) and then to hash the 7 blocks to 3 blocks using the output transformation with 7 rounds described in the previous section.

4.2 Using an $(m, 2m)$-block cipher

The only known hash functions based on an $(m, 2m)$-block cipher with a $2m$-bit hash result are the Abreast-DM and the Tandem-DM from [12]. The compression functions of both hash functions process an m-bit block using two encryptions, i.e., with rate 1/2. Let H_i and G_i denote the intermediate hash values. The Abreast-DM is defined as follows

$$
\begin{aligned}
H_i &= H_{i-1} \oplus E_{G_{i-1}, M_i}(H_{i-1}) \\
G_i &= G_{i-1} \oplus E_{M_i, H_{i-1}}(\overline{G}_{i-1}),
\end{aligned}
$$

where $\overline{G}$ is the bitwise complemented value of G. The Tandem-DM is defined as follows

$$W_i = E_{G_{i-1},M_i}(H_{i-1})$$
$$H_i = W_i \oplus H_{i-1}$$
$$G_i = G_{i-1} \oplus E_{M_i,W_i}(G_{i-1}).$$

Table 3 lists the rates and complexities of the best known attacks on the two constructions. However, as already indicated before, there exist more efficient

Table 3. Rates and complexities of previous proposal for $(m, 2m)$-block ciphers [12].

Scheme	Rate	Collision h	Collision H
Abreast-DM	1/2	2^m	2^m
Tandem-DM	1/2	2^m	2^m

constructions with a higher security level. Table 4 lists the rates and complexities of such constructions. As before, it is possible divide the m-bit blocks into smaller subblocks. E.g., the blocks can be divided into halves and expanded with a code over $GF(2^6)$, such as $[65, 63, 3]$.

Table 4. Rates and complexities of our proposals for $(m, 2m)$-block ciphers using codes over $GF(2^3)$.

Code	Rate	Collision
$[4, 2, 3]$	1/2	$\geq 2^m$
$[6, 4, 3]$	1	$\geq 2^m$
$[9, 7, 3]$	4/3	$\geq 2^m$
$[5, 2, 4]$	1/5	$\geq 2^{3m/2}$
$[7, 4, 4]$	5/7	$\geq 2^{3m/2}$
$[10, 7, 4]$	11/10	$\geq 2^{3m/2}$

4.3 The MDx family

Both MD4 [21] and MD5 [22] can be viewed as a Davies-Meyer construction with an underlying m-bit block cipher with a $4m$-bit "key." From this perspective, both constructions have rate 4.

However, Dobbertin's attacks [4, 6] on MD4 and MD5 show that the compression functions are not collision resistant, and his attack on the extended MD4 [5] shows that for MD4 even two dependent runs of the compression function are

not collision resistant. However, it seems unlikely that Dobbertin's attacks extend to compression functions consisting of two or more instantiations of MD5. We can apply our methods to construct parallel MD5 hash functions based on linear codes over $GF(2^5)$. In Table 5 we list possible constructions.

Table 5. Rates and complexities of our proposals for the MDx family using codes over $GF(2^5)$.

Scheme	Rate	Scheme	Rate	Scheme	Rate
MD4	4	[5, 3, 3]	2	[5, 2, 4]	1
MD5	4	[10, 8, 3]	3	[10, 7, 4]	2.5
		[20, 18, 3]	3.5	[20, 17, 4]	3.25

Since the assumption for our constructions, i.e., that the basic components are secure, does not hold for MD4 and MD5, we do not specify explicit bounds for the complexities of collision attacks on the compression functions. However, we conjecture that for the constructions using MD5 and codes of minimum distance 4, a collision attack is infeasible. The attack requires a simultaneous collision for at least 3 different instantiations with dependent inputs.

5 Conclusion

We have demonstrated that finding collisions of the compression function of MDC-4 takes 2^{41} operations when used with DES. This casts some doubts on the security of MDC-4. We have presented a new method for construction of hash functions based on block ciphers such as DES which is faster and more secure than existing proposals. Also, our method extends to block ciphers such as IDEA where the block size and key size are different. For large values of the internal memory, constructions using IDEA exist with rates close to two, which is a factor of four faster than existing proposals. Finally, we discussed the applications of our result to the MDx family. We show constructions using MD5, almost as fast as MD5, but (conjectured) much more secure than MD5 itself.

References

1. B.O. Brachtl, D. Coppersmith, M.M. Hyden, S.M. Matyas, C.H. Meyer, J. Oseas, S. Pilpel, 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. I.B. Damgård, "A design principle for hash functions," *Advances in Cryptology, Proc. Crypto'89, LNCS 435*, G. Brassard, Ed., Springer-Verlag, 1990, pp. 416–427.
3. B. den Boer, A. Bosselaers, "Collisions for the compression function of MD5," *Advances in Cryptology, Proc. Eurocrypt'93, LNCS 765*, T. Helleseth, Ed., Springer-Verlag, 1994, pp. 293–304.
4. H. Dobbertin, "Cryptanalysis of MD4," *Fast Software Encryption, LNCS 1039*, D. Gollmann, Ed., Springer-Verlag, 1996, pp. 53–69.

5. H. Dobbertin, "Extended MD4 Compress is not Collision-free," Unpublished, October 1995.
6. H. Dobbertin, "Cryptanalysis of MD5 compress," *Presented at the rump session of Eurocrypt'96*, May 1996.
7. 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.
8. W. Hohl, X. Lai, T. Meier, C. Waldvogel, "Security of iterated hash functions based on block ciphers." *Advances in Cryptology, Proc. Crypto'93, LNCS 773*, D. Stinson, Ed., Springer-Verlag, 1994, pp. 379–390.
9. ISO/IEC 10118, *"Information technology – Security techniques – Hash-functions, Part 1: General and Part 2: Hash-functions using an n-bit block cipher algorithm,"* 1994.
10. L.R. Knudsen, X. Lai, "New attacks on all double block length hash functions of hash rate 1, including the parallel-DM," *Advances in Cryptology, Proc. Eurocrypt'94, LNCS 959*, A. De Santis, Ed., Springer-Verlag, 1995, pp. 410–418.
11. L.R. Knudsen, B. Preneel, "Hash functions based on block ciphers and quaternary codes," *Advances in Cryptology, Proc. Asiacrypt'96, LNCS 1163*, K. Kim, T. Matsumoto, Eds., Springer-Verlag, 1996, pp. 77–90.
12. X. Lai, *"On the Design and Security of Block Ciphers,"* ETH Series in Information Processing, Vol. 1, J.L. Massey, Ed., Hartung-Gorre Verlag, Konstanz, 1992.
13. F.J. MacWilliams, N.J.A. Sloane, *"The Theory of Error-Correcting Codes,"* North-Holland Publishing Company, Amsterdam, 1978.
14. S.M. Matyas, C.H. Meyer, J. Oseas, "Generating strong one-way functions with cryptographic algorithm," *IBM Techn. Disclosure Bull.*, Vol. 27, No. 10A, 1985, pp. 5658–5659.
15. R. Merkle, "One way hash functions and DES," *Advances in Cryptology, Proc. Crypto'89, LNCS 435*, G. Brassard, Ed., Springer-Verlag, 1990, pp. 428–446.
16. C.H. Meyer, M. Schilling, "Secure program load with Manipulation Detection Code," *Proc. Securicom 1988*, pp. 111–130.
17. M. Naor, M. Yung, "Universal one-way hash functions and their cryptographic applications," *Proc. 21st ACM Symposium on the Theory of Computing*, ACM, 1989, pp. 387–394.
18. B. Preneel, R. Govaerts, J. Vandewalle, "Hash functions based on block ciphers: a synthetic approach," *Advances in Cryptology, Proc. Crypto'93, LNCS 773*, D. Stinson, Ed., Springer-Verlag, 1994, pp. 368–378.
19. V. Rijmen, B. Preneel, "Improved characteristics for differential cryptanalysis of hash functions based on block ciphers," *Fast Software Encryption, LNCS 1008*, B. Preneel, Ed., Springer-Verlag, 1995, pp. 242–248.
20. J.-J. Quisquater, J.-P. Delescaille, "How easy is collision search? Application to DES," *Advances in Cryptology, Proc. Eurocrypt'89, LNCS 434*, J.-J. Quisquater, J. Vandewalle, Eds., Springer-Verlag, 1990, pp. 429–434.
21. R.L. Rivest, "The MD4 message digest algorithm," *Advances in Cryptology, Proc. Crypto'90, LNCS 537*. S. Vanstone, Ed., Springer-Verlag, 1991, pp. 303–311.
22. R.L. Rivest, "The MD5 message-digest algorithm," *Request for Comments (RFC) 1321*, Internet Activities Board, Internet Privacy Task Force, April 1992.
23. P.C. van Oorschot, M.J. Wiener, "Parallel collision search with application to hash functions and discrete logarithms," *Proc. 2nd ACM Conference on Computer and Communications Security*, ACM, 1994, pp. 210–218.