Hash Functions Based on Block Ciphers and Quaternary 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. We consider constructions for cryptographic hash functions based on m-bit block ciphers. First we present a new attack on the LOKI-DBH mode: the attack finds collisions in $2^{3m/4}$ encryptions, which should be compared to 2^m encryptions for a brute force attack. This attack breaks the last remaining subclass in a wide class of efficient hash functions which have been proposed in the literature. We then analyze hash functions based on a collision resistant compression function for which finding a collision requires at least 2^m encryptions, providing a lower bound of the complexity of collisions of the hash function. A new class of constructions is proposed, based on error correcting codes over GF(2^2) and a proof of security is given, which relates their security to that of single block hash functions. For example, a compression function is presented which requires about 4 encryptions to hash an m-bit block, and for which finding a collision requires at least 2^m encryptions. This scheme has the same hash rate as MDC-4, but better security against collision attacks. Our method can be used to construct compression functions with even higher levels of security at the cost of more internal memory.

1 Introduction

Hash functions are functions which map a string of arbitrary size to a short string of fixed length (typically 128...160 bits). Hash functions which satisfy security properties (collision resistance, second preimage resistance, and preimage resistance) are widely used in cryptographic applications such as digital signatures, password protection schemes, and conventional message authentication.

We consider 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 permutation on m-bit strings. The main argument to construct hash functions based on block ciphers is the minimization of the design and implementation effort. Additionally, the trust in existing block ciphers can be transferred to hash functions. These arguments are historically very important (DES [10] was

* The work in this paper was initiated while the authors were visiting the Isaac Newton Institute, Cambridge, U.K., February 1996

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

seen as a main building block for practical cryptography), and have gained more importance due to the recent cryptanalytical successes achieved by H. Dobbertin [7, 8] on custom designed hash functions such as MD4 [26] and MD5 [27].

The main disadvantage of this approach is that specific hash functions are likely to be more efficient. One also has to take into account that legal restrictions on the use and import or export of hash functions might be weaker than those applying to encryption algorithms. Finally, block ciphers may exhibit some weaknesses that are only important if they are used in a hashing mode. The (semi)-weak keys of DES and the corresponding fixed points [20], and the key schedule weakness of SAFER [13] are good illustrations of this fact.

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.

After the publication of several weak proposals in the late 1970s, the first secure constructions for a hash function based on a block cipher were the scheme by Matyas, Meyer, and Oseas [18] and its dual, which is widely known as the Davies-Meyer scheme[1]. Both schemes have rate 1, and give a hash result of only m bits (which explains the name *single block length hash functions*). Finding a collision for such a scheme using the birthday paradox [30] requires about $2^{m/2}$ encryptions, while finding a (second) preimage requires about 2^m encryptions. Later it was shown that there exist essentially two secure single block length hash functions in this model, and that 12 different schemes can be obtained by applying a linear transformation to the inputs [24]. One of these schemes has been specified in ISO/IEC 10118 [12].

Since most block ciphers have a block length of $m = 64$ bits, the need for *double block length* hash functions became apparent. The goal of these constructions is to achieve a security level against brute force collision attacks of at least 2^m encryptions. Three main classes of hash functions have been proposed:

- Hash functions with rate 1 and with an internal memory of $2m$ bits; most of these constructions have been broken (see for example [14]), a notable exception being the LOKI-DBH mode proposed at Auscrypt'90 [4]. In this paper it will be shown that a collision attack for this hash function requires only $2^{3m/4}$ encryptions; the attack is more general and shows that there are no hash functions in this class for which finding a collision requires more than $2^{3m/4}$ operations.
- MDC-2 (rate 1/2) and MDC-4 (rate 1/4) [2]; their security to brute force collision attacks is about 2^m encryptions, and to preimage attacks about $2^{3m/2}$ respectively 2^{2m} encryptions. For the important practical case of DES, $m = 64$, but the key is only 56 bits long; a collision search requires then only 2^{55} encryptions for both schemes, and a preimage can be obtained after 2^{83} respectively 2^{109} encryptions [22]. A further generalization of MDC-2 has been included in ISO/IEC 10118 [12].
- Merkle's schemes, the best of which has rate 0.276 (for a 64-bit block cipher with a 56-bit key) [19]. A security proof is given which assumes that the

[1] This scheme was apparently known to Meyer and Matyas ca 1979.

underlying single block length hash function is secure. If DES is used, the rate becomes about 0.266, and finding a collision requires at least 2^{56} operations [19, 22]. A practical disadvantage are the inconvenient block sizes.

In summary, the known constructions do not provide an acceptable security level against parallel brute force collision attacks: it follows from [25, 28] 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. Moreover, a similar or higher security level for preimage attacks would be desirable as well.

In this paper we try to resolve this problem by proposing a new efficient construction for a hash function based on a block cipher 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. For $q = 2$, the constructions are more efficient than existing proposals with the same security level. The new constructions result in a parallelizable hash function, the security of which can be related to that of the well known single block length hash functions. The basic ingredients of the scheme will be the use of a larger internal memory and a quaternary error correcting code.

The remainder of this paper is organized as follows. In §2 we review a general model and some properties of hash functions. §3 gives the new attack on the LOKI-DBH mode. In §4 it is shown that extending the existing schemes to triple and quadruple solutions does not result in an improved security level. Our new construction for a hash function based on block ciphers is presented in §5, and §6 contains the conclusions.

2 Hash Functions

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. Almost all hash functions described in the literature are *iterated hash functions*, i.e., they are 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. The message M is split into blocks M_i of l bits or $M = (M_1, M_2, \ldots, M_n)$. If the length of M is not a multiple of l, M is padded using an unambiguous padding rule. The value H_n of length m is obtained by computing iteratively

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

where H_0 is a specified *initial value* denoted with IV. The function h is called the *compression function* or the hash round function. The *hash result* is then obtained by applying an output transformation g to H_n, or $\mathrm{Hash}(IV, M) = H = g(H_t)$. Note that the output transformation is often the identity function.

The theoretical work on the security of hash functions has concentrated on the reduction of the security of $\mathrm{Hash}(\cdot)$ to that of $h(\cdot, \cdot)$ [5, 16, 19, 21]. For these reductions to work in practice, we need to append an additional block at the end

of the input string which contains its length. This operation, proposed independently by R. Merkle [19] and I. Damgård [5] is known as MD-strengthening. If MD-strengthening is used, one can prove the following connection between the security of a hash function and of its compression function.

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

Note that in practical applications, the IV of a hash function is fixed in the specifications. This might lead to a higher security level; Theorem 1 gives a lower bound on the security of Hash$(IV,\cdot)$.

The use of a secure compression function $h(\cdot,\cdot)$ allows for the replacement of the iterative construction by a parallelizable tree construction [5].

In view of the above, it is remarkable that most practical hash functions do not have a collision resistant compression function (the most important exceptions are the DES based hash functions of R. Merkle [19]). For the first class of hash functions (as discussed in §1) it was shown that collisions for the compression function require at most $2^{m/2}$ encryptions [11]; the same result was proved for a large class of related hash functions of rate $1/2$. Collisions for the compression function of MDC-2 and MDC-4 can be found in time respectively 2^{28} and 2^{41} [22]. Collisions for the compression function of MD5 have been presented in [6, 8].

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. It will be assumed that the block cipher has no weaknesses, i.e., for every key it can be modeled as a random permutation (see for example [19]).

3 Attacks on LOKI-DBH

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

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

For a scheme of hash rate 1, 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*.

In [14] L. Knudsen and X. Lai present attacks on a large number of double block length hash functions of hash rate 1 for fixed IV. However, for one class of functions their (second) preimage attacks required a huge table of 2^m $2m$-bit values; no collision attacks were reported for these hash functions. In the following we provide the last piece of the puzzle by presenting attacks on all hash functions of hash rate 1 defined by (2) which are much faster than brute

force attacks and which require only small memory. In particular these attacks break the LOKI-DBH hash function [4].

We consider double block length hash functions for which H_i^1 (or H_i^2) can be written as

$$H_i^1 = E_A(B) \oplus C \qquad \text{with} \qquad \begin{bmatrix} A \\ B \\ C \end{bmatrix} = L \cdot \begin{bmatrix} H_{i-1}^1 \\ H_{i-1}^2 \\ M_i^1 \\ M_i^2 \end{bmatrix} \tag{3}$$

where L is a binary 3×4 matrix. This includes all double block length hash functions of hash rate 1 (serial or parallel), as defined in (2), but also more complex schemes. Also, we will rewrite A and B of (3) as follows

$$\begin{bmatrix} A \\ B \end{bmatrix} = N_1 \cdot \begin{bmatrix} H_{i-1}^1 \\ H_{i-1}^2 \end{bmatrix} \oplus N_2 \cdot \begin{bmatrix} M_i^1 \\ M_i^2 \end{bmatrix} \tag{4}$$

where N_1 and N_2 are 2×2 binary submatrices of L.

Theorem 2. *For the double block length hash functions of hash rate 1, whose compression functions have the form of (2), the complexities of second preimage and preimage attacks are upper bounded by about* 4×2^m. *The complexity of a collision attack is upper bounded by about* $3 \times 2^{3m/4}$. *For all but two classes of hash functions, the complexity of the collision attack is at most* $4 \times 2^{m/2}$.

Proof: Consider the general form (2). In the following we will attack the compression function H_i^1 and use the notation of (3). The proofs for $Rank(L) < 3$ and for $Rank(L) = 3, Rank(N_2) = 1$ were given in [14]. In the following let $\text{Rank}(L) = 3$ and $\text{Rank}(N_2) = 2$.
Since $\text{Rank}(L) = 3$, A, B, and C are linearly independent, and H_i^1 can be written as

$$\begin{aligned} H_i^1 &= E_A(B) \oplus C^0 \\ &= E_A(B) \oplus A \oplus C^1 \\ &= E_A(B) \oplus B \oplus C^2 \\ &= E_A(B) \oplus A \oplus B \oplus C^3 \end{aligned}$$

with C_0, C_1, C_2, and C_3 different from zero. $\text{Rank}(N_2) = 2$ implies that one of the four variables C^0, C^1, C^2 or C^3 does not contain any of the message variables M_i^1, M_i^2 or $M_i^1 \oplus M_i^2$. Let C^j denote that variable.
The cases $C^j = C^0$ and $C^j = C^1$ were discussed in [14]. If $C^j = C^2$, we assume without loss of generality that $C^2 = H_{t-1}^1$. We present meet-in-the-middle attacks which are faster than brute-force attacks and distinguish between the (second) preimage attack and the collision attack.

The (second) preimage attack:

1. Backward step: choose 2^m values for (A, B) and compute

$$H'^1_{t-1} = C^2 = E_A(B) \oplus B \oplus H_t^1 \qquad \text{for the given value of } H_t^1.$$

2. Forward step: choose 2^m values for $({M''}^1_{t-1}, {M''}^2_{t-1})$ and compute $({H''}^1_{t-1}, {H''}^2_{t-1})$ from (H^1_{t-2}, H^2_{t-2}).

Find matches ${H''}^1_{t-1} = {H'}^1_{t-1}$. For every match use equation (4) to find the values of (M^1_t, M^2_t) from (A, B) and $({H''}^1_{t-1}, {H''}^2_{t-1})$ (note $\mathrm{Rank}(N_2) = 2$). Finally compute the corresponding value of H^2_t. The quantities in the meet-in-middle attack are m bits long, so this gives us about $(2^m \times 2^m)/2^m = 2^m$ values of $(H^1_{t-1}, H^2_{t-1}, M^1_t, M^2_t)$ all hitting the same value of H^1_t. Thus, we will find messages hitting H^2_t as well with probability about 63%; the total number of operations is about 4×2^m.

The collision attack:

1. Backward step: choose $2^{3m/4}$ values for A and B and compute

$$ {H'}^1_{t-1} = C^2 = E_A(B) \oplus B \oplus H^1_t \qquad \text{for the given value of } H^1_t. $$

2. Forward step: choose $2^{3m/4}$ values for $({M''}^1_{t-1}, {M''}^2_{t-1})$ and compute $({H''}^1_{t-1}, {H''}^2_{t-1})$ from (H^1_{t-2}, H^2_{t-2}).

Find matches ${H''}^1_{t-1} = {H'}^1_{t-1}$. For every match compute the values of (M^1_i, M^2_i) from (A, B) and $({H''}^1_{t-1}, {H''}^2_{t-1})$ using equation (4) and the corresponding value of H^2_t. Since the quantities in the meet-in-middle attack are m bits long, this gives us about $(2^{3m/4} \times 2^{3m/4})/2^m = 2^{m/2}$ values of $(H^1_{t-1}, H^2_{t-1}, M^1_t, M^2_t)$ all hitting the same value of H^1_t. Thus, by the birthday paradox we will find among these a match for H^2_t with probability $1 - \exp\left((2^{m/2})^2)/2^{m+1}\right) \approx 39\%$ [9]. The total number of operations is equal to $3 \times 2^{3m/4}$.
If $C^j = C^3$, we choose random values for A and B and compute

$$ C^3 = E_A(B) \oplus A \oplus B \oplus H^1_t \; . $$

Then we proceed similarly as in the previous case. The LOKI-DBH hash function [4] is an instance of this class of hash functions. ■

4 Triple and Quadruple Constructions

When the compression function consists of several subfunctions $(H^1_i, \ldots, H^n_i$ as above) care has to be taken. As illustrated by the attacks presented in the previous section, compression functions for which it is possible to invert (parts of) the outputs are vulnerable to a meet-in-the-middle attack. Therefore we will require that each of the subfunctions H^j_i have the form of a secure single block length hash function [24].

Moreover, since most block ciphers have a block length of 64 bits, a brute-force collision attack on a double block length hash function, as defined by (2), will never take more than 2^{64} operations. Recent work by van Oorschot and Wiener [28] shows how parallelization techniques can be used to find collisions

for such hash functions in reasonable time and at a reasonable cost (less than a month with 10 mio. US$).

This leads us to specify the requirements for a hash function based on an m-bit block cipher with compression function consisting of subfunctions $H_i^1, \ldots, H_i^n$ as follows.

1. Each function H_i^j is preimage resistant, i.e., given $y = H_i^j(X)$ about 2^m encryptions are required to find an X' such that $H_i^j(X') = Y$.
2. A collision attack on the compression function takes at least 2^m operations.

The first requirement above implies that the subfunctions have the form of a secure single block length hash function [24]. It may help the reader at this point to think of the subfunction h_i as the compression function of the Davies-Meyer hash function:

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

The second requirement above implies that the number of subfunctions must be at least three, i.e., $n \geq 3$.

The compression function of a triple block length hash function based on a block cipher involves three functions H_i^1, H_i^2, H_i^3. One iteration will involve three chaining variables $H_{i-1}^1, H_{i-1}^2, H_{i-1}^3$ and at least one message variable M_i. It is clear that we cannot meet both our requirements above. Indeed, it will be possible to make the inputs to one of the subfunctions (e.g., H_i^1) constant by an appropriate choice of the input variables; subsequently the remaining freedom in the variables can be used for a brute force collision search on H_i^2 and H_i^3. This freedom corresponds to two variables.

A similar result holds for quadruple block length hash functions. One iteration will involve at least five variables. One can make the inputs to two of the subfunctions constant by an appropriate choice of the input variables; the remaining freedom, corresponding to one variable, can be used for a brute force collision search on the other two subfunctions.

In the next section we show that hash functions exist meeting our two requirements for schemes with $n > 4$.

5 A New General Construction

In this section we present a new general construction for a collision resistant hash function based on an m-bit block cipher, that is, where collisions will take more than 2^m operations. The construction extends a simple hash mode which is believed to be secure to a multiple hash mode. The security of the new construction can be proved based on the following two assumptions:

- the simple hash mode is secure;
- no shortcuts can be found to break independent instances of the simple hash mode.

First we make the assumptions more precise, and subsequently we present a new construction and prove that its security can be derived from the assumptions. Then extensions of the construction are discussed and a complete example is provided.

5.1 Security assumptions

We now state the two assumptions required to prove the security of the new construction.

Assumption 1 (Davies-Meyer) *Let $E_K(.)$ be an m-bit block cipher with an m-bit key K. Define the compression function h to be the Davies-Meyer function (5). Then finding collisions for h requires about $2^{m/2}$ encryptions (of an m-bit block), and finding a (second) preimage for h requires about 2^m encryptions.*

This assumption is motivated by fact that the Davies-Meyer scheme (and 11 variants of this scheme, see [24]) seems to be secure; a similar assumption has been used and heuristically motivated by R. Merkle in [19].

Assumption 2 (Multiple Davies-Meyer) *Let f_1, f_2, $\ldots f_n$, be different instantiations of the Davies-Meyer function, that is, $f_i(X_i, Y_i) = E_{X_i}(Y_i) \oplus Y_i$ (these can be obtained by fixing $\lceil \log_2 n \rceil$ key bits to different values). Consider a compression function with $2k$ m-bit input blocks, Z_i, which are expanded by an affine mapping to the n pairs (X_i, Y_i)[2]. Assume that the output of the compression function depends on all $2k$ blocks, and, more precisely, the matrix of the affine expansion mapping has rank $2k$.*

Assume a simultaneous collision for f_1, f_2, $\ldots f_n$, has been found, i.e., two different sets of input values $\{z_i\}$ and $\{z'_i\}$ yielding equal outputs for all n functions. Let $\{f_j\}$ be the set of N functions, $N \leq n$, which depend on the input blocks Z_i and Z'_i for which $z_i \neq z'_i$. Let the rank of the submatrix of the functions $\{f_j\}$ be $N - v$. Then it is assumed that finding simultaneous collisions for these N functions f_i requires at least $2^{vm/2}$ encryptions, and finding a simultaneous (second) preimage for the N functions requires at least 2^{vm} encryptions.

This assumption can be justified with an appropriate assumption about the underlying block cipher of a multiple Davies-Meyer compression function. More precisely, consider 2 different subfunctions f_i and f_j and assume that the inputs to one subfunction f_i are dependent on the inputs to another subfunction f_j. Then if the block cipher is secure, we assume that two inputs colliding for f_i will collide also for f_j with probability 2^{-m}. In other words, our assumption is that, if the block cipher is secure, finding collisions simultaneously for f_i and f_j for dependent inputs will not be easier than a brute-force attack.

Consider a multiple Davies-Meyer compression function with n subfunctions and assume that a simultaneous collision have been found as discussed in Assumption 2. Since the rank of the submatrix of the N functions $\{f_j\}$ is $N - v$,

[2] The choice of an even number of input blocks is not mandatory, but this choice will become clear later.

it will be possible to find collisions for $N - v$ of these functions independently of each other. Our assumption then says that finding simultaneous collisions also for the remaining v functions requires at least $2^{vm/2}$ encryptions, and finding a simultaneous (second) preimage requires at least 2^{vm} encryptions. Thus, the best an attacker can hope for is a brute force attack on the v remaining functions.

5.2 The new construction

The following theorem describes the new construction and gives a proof for its security.

Theorem 3. *If there exists a quaternary $[n, k, d]$ code of length n, dimension k, and minimum distance d, with $2k > n$, and $n = O(m)$, then there exists a parallel hash function based on an m-bit block cipher for whose compression function finding a collision requires at least $2^{(d-1)m/2}$ encryptions and finding a second 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 $(2k/n) - 1$.*

Proof: The proof is constructive. The compression function consists of n functions f_i with $1 \leq i \leq n$ (see Assumption 2) which have been made different by fixing $\lceil \log_2 n \rceil$ key bits to different values. The input to the compression function consists of $2k$ m-bit blocks: the n chaining variables H^1 through H^n (the output of the n functions of the previous iteration) and r message blocks M^1 through M^r, with $r = 2k - n > 0$. In the following, every individual bit of these m-bit blocks is treated in the same way. The bits of two consecutive input blocks are pairwise combined yielding k elements of $GF(2^2)$. These elements are encoded using a quaternary $[n, k, d]$ code, resulting in n elements of $GF(2^2)$. Each of these elements represents the 2-bit inputs to one of the n functions. The individual input bits are obtained by representing the elements of $GF(2^2)$ as a vector space over $GF(2)$. This construction guarantees that the conditions for Assumption 2 are satisfied for the value $v = d - 1$. By a linear transformation of the $2k$ inputs it is possible to obtain a compression function, where the inputs to the first k subfunctions are independent of each other. The inputs to the last $n - k$ subfunctions will depend on the inputs to the first k subfunctions. The minimum distance of the quaternary code ensures that the inputs to at least $d-1$ of the last $n - k$ functions will depend on the inputs to the first k functions. Since $n - k \geq d - 1$ [17] the result follows immediately. ∎

The security bound will be slightly smaller than indicated, since a number of key bits have to be fixed. However, $n = O(m)$, and thus the resulting reduction of the security level will be negligible. The above bound is a lower bound and the existence of attacks meeting this bound is not guaranteed. Later we give a concrete proposal of a compression function constructed as above, for which the complexity of the best known attacks are even higher than the proven lower bound.

The existence of efficient constructions for $d = 3$ follows from the existence of perfect Hamming codes over GF(2^2) (see for example [17, 179–180] or [3]).

Theorem 4. *Let q be a prime power. The (perfect) Hamming codes over GF(q) have the following parameters:*

$$\left[n = \frac{q^s - 1}{q - 1}, k = n - s, d = 3 \right] .$$

The resulting codes can be shortened without decreasing the minimum distance, which implies the following result:

Corollary 5. *There are parallel hash functions based on an m-bit block cipher for which for the compression function finding a collision requires at least 2^m encryptions and finding a second preimage requires at least 2^{2m} encryptions provided that Assumption 2 holds. The following rates can be obtained in function of the internal memory: $n = 5$, rate $1/5$, $n = 8$, rate $1/4$, $n = 10$, rate $2/5$, $n = 12$, rate $1/2$, and $n = 21$, rate $5/7$ ($s = 3$). For large values of n, the rate is about*

$$1 - \frac{\log_2(3n + 1)}{n}$$

which is very close to 1.

To obtain an even higher level of security, one can use other codes to construct schemes for $d = 4$, e.g. using the codes $[9, 5, 4]$, $[12, 8, 4]$, $[16, 12, 4]$, see [3].

Corollary 6. *There are parallel hash functions based on an m-bit block cipher for which for the compression function finding a collision requires at least $2^{3m/2}$ encryptions and finding a second preimage requires at least 2^{3m} encryptions provided that Assumption 2 holds. The following rates can be obtained in function of the internal memory: $n = 9$, rate $1/9$, $n = 12$, rate $1/3$, $n = 16$, rate $1/2$.*

Notes:

1. Apart from the simple security proof and the relatively high rates, the schemes have the advantage that n encryptions can be carried out in parallel. The disadvantage of the schemes is the increased amount of internal memory.
2. For the (second) preimage attack, the security bounds assume that the entropy of the unknown part of the input is at least $(d - 1)m$ bits.
3. Theorem 3 only considers the strength against attacks on the compression function. The strength of the hash function itself could be improved by dividing the output of a single f_i function over all the other functions. However, proving a stronger security bound is probably quite difficult.

5.3 Extensions

The construction has the property that the size of the hash result is larger than the security level of the hash function would suggest. This could be a problem in applications where 'near collisions' are not acceptable [1] (e.g., it is feasible to find two hash values which have $n-d+1$ colliding blocks). In that case, an output transformation can be defined which compresses the result of the final iteration.

We present here two approaches, which do not affect the provable security (both techniques can be combined; an example is given in § 5.4 below).

Denote with $n_{\min}$ the smallest possible value of n for a given value of d, such that Theorem 3 holds (e.g., for $d = 3$, $n_{\min} = 5$, and $d = 4$, $n_{\min} = 9$). If $n_{\min} < n$, compress the n blocks to $n_{\min}$ blocks using the new construction with $n_{\min}$ parallel blocks (since $n_{\min} < n$, this hash function will have a lower rate than the original one).

If a reduction is required to less than $n_{\min}$ blocks, one can use a number of parallel and independent instances of Merkle's hash function based on a block cipher [19]. This will be rather slow compared to the new construction, but this is not so important since Merkle's function is only required for the output transformation.

5.4 A practical example

We present the details of a concrete proposal based on an $[8, 5, 3]$ quaternary code, which is obtained by shortening the $[21, 18, 3]$ Hamming code. The hash function uses $n = 8$ parallel encryptions, and hashes $2 \cdot 5 - 8 = 2$ 64-bit message blocks, resulting in a rate of $1/4$. Using a 64-bit block cipher with a 64-bit key, finding a collision for the compression function requires at least 2^{64} encryptions, and finding a (second) preimage requires at least 2^{128} encryptions. Later we describe the best attacks we have found, which requires even more encryptions.

The generator matrix of the $[8, 5, 3]$ Hamming code over $\mathrm{GF}(2^2)$ has the following form:

$$\begin{bmatrix} 1\,0\,0\,0\,0\,0\,1\;1 \\ 0\,1\,0\,0\,0\,1\,0\;\alpha \\ 0\,0\,1\,0\,0\,1\,0\;\beta \\ 0\,0\,0\,1\,0\,1\,0\;1 \\ 0\,0\,0\,0\,1\,1\,1\;0 \end{bmatrix} \tag{6}$$

here $0 = [00]$, $1 = [01]$, $\alpha = [10]$, and $\beta = [11]$. The order of the chaining variables is chosen to be $H_{i-1}^1, M_i^1, H_{i-1}^3, H_{i-1}^4, H_{i-1}^5, H_{i-1}^6, H_{i-1}^7, H_{i-1}^8, H_{i-1}^2, M_i^2$, but can be chosen arbitrarily (the motivation for this particular choice is indicated below). This results in the following compression function, where $h(X, Y)$ denotes $E_X(Y) \oplus Y$:

$$\begin{aligned}
H_i^1 &= f_1(H_{i-1}^1, M_i^1) \\
H_i^2 &= f_2(H_{i-1}^3, H_{i-1}^4) \\
H_i^3 &= f_3(H_{i-1}^5, H_{i-1}^6) \\
H_i^4 &= f_4(H_{i-1}^7, H_{i-1}^8) \\
H_i^5 &= f_5(H_{i-1}^2, M_i^2) \\
H_i^6 &= f_6(H_{i-1}^3 \oplus H_{i-1}^5 \oplus H_{i-1}^7 \oplus H_{i-1}^2, H_{i-1}^4 \oplus H_{i-1}^6 \oplus H_{i-1}^8 \oplus M_i^2) \\
H_i^7 &= f_7(H_{i-1}^1 \oplus H_{i-1}^2, M_i^1 \oplus M_i^2) \\
H_i^8 &= f_8(H_{i-1}^1 \oplus H_{i-1}^3 \oplus H_{i-1}^4 \oplus H_{i-1}^6 \oplus H_{i-1}^7, M_i^1 \oplus H_{i-1}^3 \oplus H_{i-1}^5 \oplus H_{i-1}^6 \oplus H_{i-1}^8)
\end{aligned}$$

In addition, the 8 functions are made different by fixing the values of 3 key bits to the block cipher.

Although we have proved a lower bound on the complexity of attacks on the above compression function we have not succeeded in finding such attacks. Indeed the best collision attack we know is the following. Fix the inputs to H_i^1 and H_i^5. This yields constant values also for H_i^7. Find independently $2^{m/3}$-fold collisions for each of H_i^2, H_i^3, and H_i^4. This will require about $3 \times 2^{4m/3}$ encryptions of the block cipher [23]. These three sets are combined to a set of 2^m simultaneous collisions for H_i^2, H_i^3, and H_i^4. With a high probability one of these collisions yield a collision also for H_i^6 and H_i^8. Therefore the best known collision attack requires about $3 \times 2^{4m/3}$ encryptions of the block cipher. We need to fix three bits of the key, which gives a security level for collision attacks of about 2^{81} for $m = 64$.

In the case of DES [10], one needs to fix an additional 2 key bits in order to avoid the complementation property, and the (semi-)weak keys and the associated (anti-)fixed points (bits 2 and 3 of the key are fixed to 01 [2]). Since DES has a 56-bit key, the effective key size will be reduced to 51 bits, resulting in a security level of $2^{4 \cdot 51/3} = 2^{68}$. By fixing two key bits and three plaintext bits, this could be improved to 2^{72}, which seems to be sufficient to preclude brute force collision search at the cost of a more complex implementation. The order of the chaining variables above was chosen to ensure that the message variables do not occur as key input to the block cipher. The most important reason for this is to obtain a hash rate of 1/4 since the key size is less than the block size in DES; moreover, this avoids giving an attacker the advantage in 'real' attacks on the hash function with a fixed value of IV (compared to attacks on the compression function) of being able to choose the keys.

A reduction of the eight 64-bit blocks to three 64-bit blocks using two parallel instantiations of Merkle's construction (resulting in a rate of about $0.25 \times 0.5 = 1/8$) requires about 64 DES encryptions.

Note that the 2^m complexity of MDC-2 is the best known attack against the hash function, while against Merkle's scheme and our schemes the 2^m complexity is against the compression functions and is a lower bound for the complexity of an attack on the hash function. The complexity of the best known attack against the compression function of MDC-2 is only $2^{m/2}$.

6 Conclusion

We have demonstrated that designing efficient hash functions with minimal internal memory based on an m-bit block cipher is a difficult problem. Moreover, none of the previous proposals provides a protection better than 2^m against brute force collision search. By changing the model slightly (i.e., by increasing the size of the internal memory, and by introducing an output transformation), it is possible to obtain a compression function and thus a hash function for which the security can be proved based on a plausible assumption. The proposed construction achieve the same efficiency as the best current proposals (namely rate

1/4, or 4 encryptions to hash a single block) but a higher security level, or a higher efficiency for the same security level. For large value of the internal memory, rates close to one can be obtained. Also, the constructions allows for a high degree of parallelism, which will yield even more efficient implementations.

Acknowledgments

The authors wish to thank Tor Helleseth and Torleiv Kløve for helpful discussions of useful codes.

References

1. R. Anderson, "The classification of hash functions," *Codes and Cyphers: Cryptography and Coding IV*, P.G. Farrell, Ed., Institute of Mathematics & Its Applications (IMA), 1995, pp. 83–93.
2. 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.
3. A.E. Brouwer, "Linear code bound," `http://www.win.tue.nl/win/math/dw/voorlincod.html`.
4. L. Brown, J. Pieprzyk, J. Seberry, "LOKI – a cryptographic primitive for authentication and secrecy applications," *Advances in Cryptology, Proc. Auscrypt'90, LNCS 453*, J. Seberry, J. Pieprzyk, Eds., Springer-Verlag, 1990, pp. 229–236.
5. 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.
6. 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.
7. H. Dobbertin, "Cryptanalysis of MD4," *Fast Software Encryption, LNCS 1039*, D. Gollmann, Ed., Springer-Verlag, 1996, pp. 53–69.
8. H. Dobbertin, "Cryptanalysis of MD5 compress," Presented at the rump session of Eurocrypt'96, May 1996.
9. W. Feller, *"An Introduction to Probability Theory and Its Applications, Vol. 1,"* Wiley & Sons, 1968.
10. 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.
11. 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.
12. ISO/IEC 10118, *"Information technology – Security techniques – Hash-functions, Part 1: General and Part 2: Hash-functions using an n-bit block cipher algorithm,"* IS 10118, 1994.
13. L.R. Knudsen, "A Key-schedule Weakness in SAFER K-64," *Advances in Cryptology, Proc. Crypto'94, LNCS 839*, Y. Desmedt, Ed., Springer-Verlag, 1994, pp. 274–286.

14. 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.
15. L.R. Knudsen, X. Lai, B. Preneel, "Attacks on Fast Double Block Length Hash Functions". Submitted to the Journal of Cryptology.
16. 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.
17. F.J. MacWilliams, N.J.A. Sloane, *"The Theory of Error-Correcting Codes,"* North-Holland Publishing Company, Amsterdam, 1978.
18. 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.
19. 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.
20. J.H. Moore, G.J. Simmons, "Cycle structure of the DES for keys having palindromic (or antipalindromic) sequences of round keys," *IEEE Trans. on Software Engineering*, Vol. SE–13, No. 2, 1987, pp. 262–273.
21. 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.
22. B. Preneel, "Analysis and design of cryptographic hash functions," *Doctoral Dissertation*, Katholieke Universiteit Leuven, 1993.
23. B. Preneel, R. Govaerts, J. Vandewalle, "On the power of memory in the design of collision resistant hash functions," *Advances in Cryptology, Proc. Auscrypt'92, LNCS 718*, J. Seberry, Y. Zheng, Eds., Springer-Verlag, 1993, pp. 105–121.
24. 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.
25. 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.
26. 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.
27. R.L. Rivest, "The MD5 message-digest algorithm," *Request for Comments (RFC) 1321*, Internet Activities Board, Internet Privacy Task Force, April 1992.
28. 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.
29. M.J. Wiener, "Efficient DES key search," *Technical Report TR-244*, School of Computer Science, Carleton University, Ottawa, Canada, May 1994. Presented at the rump session of Crypto'93.
30. G. Yuval, "How to swindle Rabin," *Cryptologia*, Vol. 3, No. 3, 1979, pp. 187–189.

This article was processed using the LaTeX macro package with LLNCS style