Reprint of pp.55-70 in Advances in Cryptology-EUROCRYPT'92 Proceedings, LNCS 658, Springer-Verlag, 1993.

# Hash Functions Based on Block Ciphers

Xuejia Lai and James L. Massey

Signal and Information Processing Laboratory
Swiss Federal Institute of Technology
CH–8092 Zürich, Switzerland

**Abstract.** Iterated hash functions based on block ciphers are treated. Five attacks on an iterated hash function and on its round function are formulated. The wisdom of strengthening such hash functions by constraining the last block of the message to be hashed is stressed. Schemes for constructing $m$-bit and $2m$-bit hash round functions from $m$-bit block ciphers are studied. A principle is formalized for evaluating the strength of hash round functions, viz., that applying computationally simple (in both directions) invertible transformations to the input and output of a hash round function yields a new hash round function with the same security. By applying this principle, four attacks on three previously proposed $2m$-bit hash round functions are formulated. Finally, three new hash round functions based on an $m$-bit block cipher with a $2m$-bit key are proposed.

## 1 Introduction

This paper is intended to provide a rather rounded treatment of hash functions that are obtained by iterating a round function. Section 2 examines the possible attacks on such iterated hash functions, considers relations between the security of an iterated hash function and the security of its hash round function, and points out the wisdom of strengthening the hash function by constraining the last block of the message to be hashed.

In Section 3, we consider hash round functions constructed from secret-key block ciphers. In particular, we consider the problems of constructing $m$-bit hash round functions and $2m$-bit hash round functions from $m$-bit block ciphers. A principle is formalized for evaluating the strength of hash round functions, viz., that applying computationally simple (in both directions) invertible transformations to the input and output of a hash round function yields a new hash round function with the same security. To demonstrate this principle, we present four attacks on three previously proposed $2m$-bit hash round functions. Finally, three new hash round functions based on an $m$-bit block cipher with a $2m$-bit key are proposed.

## 2 Iterated hash functions and attacks

A *hash function* is an easily implementable mapping from the set of all binary sequences of some specified minimum length or greater to the set of binary sequences of some fixed length. In cryptographic applications, hash functions are used within digital signature schemes and within schemes to provide data integrity (e.g., to detect modification of a message).

An *iterated hash function* is a hash function $\mathrm{Hash}(\cdot)$ determined by an easily computable function $h(\cdot,\cdot)$ from two binary sequences of respective lengths $m$ and $l$ to a binary sequence of length $m$ in the manner that the message $M = (M_1, M_2, ..., M_n)$, where $M_i$ is of length $l$, is hashed to the *hash value* $H = H_n$ of length $m$ by computing recursively

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

where $H_0$ is a specified *initial value*. We will write $H = \mathrm{Hash}(H_0, M)$ to show explicitly the dependence on $H_0$. The function $h$ will be called the *hash round function*. Such a recursive construction of hash functions has been called the "meta-method" by Merkle [13], see also [4, 15]. For message data whose total length in bits is not a multiple of $l$, one can apply deterministic "padding" [7, 13] to the message to be hashed by (1) to increase the total length to a multiple of $l$.

For iterated hash functions, we distinguish the following five attacks:

1. **Target attack:** Given $H_0$ and $M$, find $M'$ such that $M' \neq M$ but $\mathrm{Hash}(H_0, M') = \mathrm{Hash}(H_0, M)$.

2. **Free-start target attack:** Given $H_0$ and $M$, find $H_0'$ and $M'$ such that $(H_0', M') \neq (H_0, M)$ but $\mathrm{Hash}(H_0', M') = \mathrm{Hash}(H_0, M)$.

3. **Collision attack:** Given $H_0$, find $M$ and $M'$ such that $M' \neq M$ but $\mathrm{Hash}(H_0, M') = \mathrm{Hash}(H_0, M)$.

4. **Semi-free-start collision attack:** Find $H_0$, $M$ and $M'$ such that $M' \neq M$ but $\mathrm{Hash}(H_0, M') = \mathrm{Hash}(H_0, M)$.

5. **Free-start collision attack:** Find $H_0$, $H_0'$, $M$ and $M'$ such that $(H_0', M') \neq (H_0, M)$ but $\mathrm{Hash}(H_0', M') = \mathrm{Hash}(H_0, M)$.

**Remark.** In applications where $H_0$ is specified and fixed, attacks 2, 4 and 5 are not "real attacks". This is because the initial value $H_0$ is then an integral part of the hash function so that a hash value computed from a different initial value will not be accepted. However, if the sender is free to choose and/or to change $H_0$, attacks 2, 4 and 5 can be real attacks, depending on the manner in which the hash function is used. Note that the free-start and semi-free-start attacks are never harder than the attacks where $H_0$ is specified in advance.

For an $m$-bit hash function, brute-force target attacks, in which one randomly chooses an $M'$ until one hits the "target" $H = \mathrm{Hash}(H_0, M)$, require about $2^m$ computations of hash values. It follows from the usual "birthday argument" that brute-force collision attacks require about $2^{m/2}$ computations of hash values. In particular, for hash round functions with $l \geq m$ so that all $2^m$ hash values can be reached with one-block messages, brute-force target 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 $h$. We will say that the computational security of the hash function is *ideal* when there is no attack substantially better than brute force.

In the following discussion, we consider some relations between the security of an iterated hash function and the strength of its hash round function. By an *attack on the hash round function* we mean an attack in which all the involved messages contain only *one* block. For example, a target attack on the round function $h$ reads: given $H_0$ and $M_1$, find $M_1'$ such that $M_1' \neq M_1$ but $h(H_0, M_1') = h(H_0, M_1)$. Once a target attack on the round function yields $M_1'$, then, by "attaching" the message blocks $M_2, ..., M_n$ of the given message to $M_1'$, one obtains success in a target attack on the iterated hash function. Similar arguments hold also for other types of attacks.

**Proposition 1** *For an iterated hash function, any attack on its round function implies an attack of the same type on the iterated hash function with the same computational complexity.*

It should be noted that the converse of the statement of Theorem 1 is not true in general. There may be attacks on the iterated hash function that are easier than attacks on the round function alone, as the following three examples show.

**Example 1 (Long message attack.)** *For an $m$-bit iterated hash function, given an $n$-block message $M = (M_1, M_2, ..., M_n)$, there is a target attack which takes about*

$$C = \begin{cases} \frac{2^m}{n} + n & \text{for } n \leq 2^{m/2} \\ 2 \times 2^{m/2} & \text{for } n > 2^{m/2} \end{cases}$$

*computations of the round function. [Essentially the above result for $n \leq 2^{m/2}$ is due to Winternitz [23].]*

**Proof.** First we consider the case $n \leq 2^{m/2}$. For the given $M$, we compute $H_i = h(H_{i-1}, M_i)$ for $i = 1, .., n$ and store these values. Then we compute $H^* = h(H_0, M_1^*)$ repetitively with randomly chosen $M_1^*$. After computing $\frac{2^m}{n}$ values for $H^*$, the probability that $H^* = H_i$ for some $i, 1 \leq i \leq n$, is

$$1 - [(1 - 2^{-m})^n]^{\frac{2^m}{n}} = 1 - (1 - 2^{-m})^{2^m} \approx 1 - e^{-1} \approx 0.63,$$

which shows that fewer than $\frac{2^m}{n}$ computations of round function will usually suffice. The message $M' = (M_1^*, M_{i+1}, \ldots, M_n)$ hashes to the same value $H$ as the message $M$, and total number of computations of the round function is about $\frac{2^m}{n} + n$. The probability that $M' = M$ is negligible.

For $n > 2^{m/2}$, we compute and store only $H_1, H_2, \ldots, H_{2^{m/2}}$. Then $2^{m/2}$ random choices of $M_1^*$ will yield a "match" of some $H^*$ with some $H_i$, $1 \leq i \leq 2^{m/2}$, with probability about 0.63. □

For an iterated hash function, one can always do the following "trivial" free-start attacks.

**Example 2 (Trivial free-start attacks.)** *Consider a message $M = (M_1, M_2)$ that hashes to $H$ with initial value $H_0$. Then, for the initial value $H_1 = h(H_0, M_1)$, the "truncated" message $M' = M_2$ hashes also to the value $H = h(H_1, M_2)$. That is, a free-start target attack can always be done if the message contain more than one block. Similarly, one can do a trivial free-start collision attack.*

The following attack using a "fixed-point" of the hash round function was proposed in [16].

**Example 3 (A trivial semi-free-start collision attack based on a 'fixed point'.)** *If the hash round function $h$ has a recognizable "fixed point", i.e., if one can somehow find $(H, M)$ such that $H = h(H, M)$, then there is a trivial semi-free-start collision attack since, starting with the initial value $H_0 = H$, the "different" messages $M = M$ and $M' = (M, M)$ both hash to the same value $H$.*

Note that in the trivial free-start and semi-free-start attacks and in the "long-message" attack described in the above three examples, one breaks the iterated hash function without breaking its round function. Such attacks are based on the fact that, for an iterated hash function of the form (1), the attacker can take advantage of the fact that a falsified message can have a *length different* from that of the given genuine message. This problem can be overcome by the following strengthening of iterated hash functions, which was proposed independently by Merkle[13] and by Damgaard[4]:

**Merkle-Damgaard Strengthening (MD-strengthening)** *For the iterated hash function, specify that the last block $M_n$ of the "message" $M = (M_1, M_2, ..., M_n)$ to be hashed must represent the length of the "true message" in bits, i.e., the length of the unpadded portion of the first $n - 1$ blocks.*

Using arguments similar to those in [4, 13, 17], one can show that:

**Proposition 2** *Against a free-start (target or collision) attack, an iterated hash function with MD-strengthening, $Hash_{\mathrm{MD}}$, has roughly the same computational security as its hash round function.*

In the previous discussions we have considered the security of an iterated hash function and the security of its round function against an attack of the *same* type. Now we consider how to relate "non-real" free-start target attacks to "real" target attacks. The following result shows that, for an iterated hash function, when a "random inverse" of the hash round function can be found with less than the ideal maximum of about $2^m$ computations, then there always exists a target attack on the hash function that is better than the brute-force target attack.

**Proposition 3 ( A meet-in-the-middle target attack by "working backwards".)** *Let $Hash_{\mathrm{MD}}$ be an $m$-bit iterated hash function with MD-strengthening* *and with round function $h$. If, for most $H$ in the range of $h$, it takes about $2^s$ computations of $h$ to find a new solution $(H', M')$ of $H = h(H', M')$ for which $H'$ appears to be essentially randomly chosen and if the unconstrained portion of messages contains at least two blocks, i.e., $n - 1 \geq 2$, then there exists a target attack on $Hash_{\mathrm{MD}}$ that takes about $2 \times 2^{\frac{m+s}{2}}$ computations of $h$.*

**Proof.** For given $M$ and $H_0$, let the results of the first two iterations be

$$H_1 = h(H_0, M_1), \qquad H_2 = h(H_1, M_2).$$

We show how to find two message blocks $(M_1', M_2')$ that hash to $H_2$ by a "meet-in-the-middle" attack. Then replacing the first two blocks $(M_1, M_2)$ in the given message $M$ by $(M_1', M_2')$, we obtain a message $M'$ of the same length as, but different from, $M$ that hashes to the same $H$.

First, we compute $G_1 = h(H_0, M_1')$ for $2^{\frac{m+s}{2}}$ randomly chosen $M_1'$'s; then we find $2^{\frac{m-s}{2}}$ pairs $(G_1', M_2')$ such that $H_2 = h(G_1', M_2')$ and $G_1'$ appears essentially randomly chosen. The attack succeeds if some $G_1$ and some $G_1'$ take on the same value. Thus, the attack succeeds with probability

$$1 - [(1 - 2^{-m})^{2^{\frac{m+s}{2}}}]^{2^{\frac{m-s}{2}}} = 1 - (1 - 2^{-m})^{2^m} \approx 1 - e^{-1} \approx 0.63,$$

as follows from the facts that the probability of choosing $M_1'$ so that $G_1$ will not equal $G_1'$ is $1 - 2^{-m}$, that there are $2^{\frac{m+s}{2}}$ independent chances to choose $M_1'$ so that $G_1$ will "miss" a particular $G_1'$, and there are $2^{\frac{m-s}{2}}$ independently chosen values of $G_1'$ to miss. Both the "forwards" computation for computing values of $G_1$ and the "backwards" computation for computing values of $G_1'$ take $2^{\frac{m+s}{2}}$ computations of the round function $h$. □

The method used in the above proof of attacking an iterated hash function by "working backward" [1, 22] has been used to attack several proposed iterated hash functions [15, 22]. The above result shows that if the hash round function does not have ideal computational security against a free-start target attack, then the iterated hash function cannot achieve ideal computational security against a target attack. Proposition 2, together with the argument used to prove Proposition 3, implies:

**Proposition 4** *Suppose that the unconstrained portion of messages must contain at least two blocks, i.e., $n-1 \geq 2$. Then an iterated hash function with MD-strengthening, $Hash_{\mathrm{MD}}(\cdot)$, has ideal computational security against a target attack if and only if its hash round function $h(\cdot, \cdot)$ has ideal computational security against a free-start target attack.*

**Proof.** Suppose the round function $h$ has ideal computational security against a free-start target attack. Then Proposition 2 shows that $\mathrm{Hash}_{\mathrm{MD}}(\cdot)$ has the same ideal security against a free-start target attack. But a target attack without free start is no easier than a free-start target attack so that $\mathrm{Hash}_{\mathrm{MD}}(\cdot)$ also has ideal computational security against a target attack.

Conversely, if for an $m$-bit hash round function $h$, a free-start target attack takes less than $2^m$ computations, then Proposition 3 implies a target attack on $\text{Hash}_{\text{MD}}$ with less than $2^m$ computations. □

From the above two propositions, we see that MD-strengthening creates secure iterated hash functions from secure round functions. In particular, the trivial free-start and semi-free-start attacks and the long-message target attack in the above examples *cannot* be used to attack an iterated hash function with MD-strengthening. Such considerations suggest an obvious implementation principle for iterated hash functions, viz., that *iterated hash functions should be used only with MD-strengthening.* In the following discussion, whenever the security of an iterated hash function is considered, we always mean the security of the hash function with MD-strengthening.

Because of Proposition 4 and Proposition 2 and because one generally desires that the hash function be strong enough to provide protection against free-start attacks, the problem of constructing secure hash functions reduces to the problem of constructing hash round functions that are secure against free-start attacks, which will be considered in the next section.

# 3 Hash round functions based on block ciphers

In the following discussion, we consider schemes for constructing hash round functions from a block cipher. In what follows, we write $Y = E_Z(X)$, for an $m$-bit block cipher $E$ with $k$-bit key, to mean that the $m$-bit ciphertext $Y$ is computed from the $m$-bit plaintext $X$ and $k$-bit key Z. Based on the discussion in the last section, we consider only attacks on the hash round function or equivalently, attacks on the iterated hash function with MD-strengthening.

## 3.1 Some $m$-bit hash round functions

**Davies-Meyer (DM) scheme:** The DM-scheme was proposed independently by Davies and by Meyer, cf. [5, 11, 22]. This scheme can be used with any block cipher. The message block $M_i$ that is hashed in each step of this scheme has length $l$ equal to the key length $k$ of the block cipher, i.e., $l = k$. The hash round function is given by

$$h(H_{i-1}, M_i) = E_{M_i}(H_{i-1}) \oplus H_{i-1} \qquad (2)$$

and is illustrated in Fig.1 where here and hereafter $\oplus$ denotes bit-by-bit modulo-two addition.

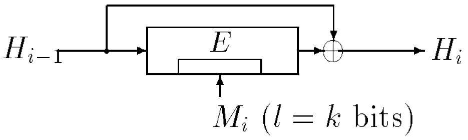

Figure 1: The hash round function of the DM-scheme. The small box indicates the key input to the block cipher.

The DM-scheme with MD-strengthening is generally considered to be secure in the sense that, if the block cipher has no known weakness, then no attack better than the brute-force attacks is known, i.e., the free-start target attack on $h$ takes about $2^m$ computations and the free-start collision attack on $h$ takes about $2^{m/2}$ computations. In particular, with MD-strengthening, none of the attacks mentioned in the three examples of the last section can be effectively used against an iterated hash function based on the DM-scheme. The DM-scheme is currently under consideration as an ISO standard [7].

**A proposed $m$-bit hash round function using a block cipher with $m$-bit block and $2m$-bit key:** This method is based on a block cipher with block-length $m$ and key-length $k = 2m$. For example, one could use the block cipher PES [8] or its improved version IPES [9]. For such a cipher with $k = 2m$, we will write $Y = E_{Z_a,Z_b}(X)$ to mean that the $m$-bit ciphertext is computed from the $m$-bit plaintext $X$ and two $m$-bit subkeys $Z_a$ and $Z_b$. The proposed hash round function is given by

$$h(H_{i-1}, M_i) = E_{H_{i-1},M_i}(H_{i-1})$$

and is illustrated in Fig. 2. We have been unable to find an attack on this hash function better than the brute force attack when the underlying block cipher has no known weakness.

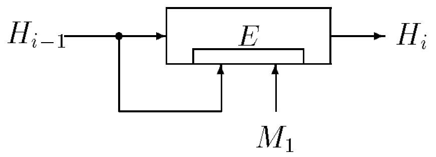

Figure 2: A proposed $m$-bit hash function based on an $m$-bit block cipher with a 2m-bit key.

## 3.2 Construction of $2m$-bit hash round functions

When the block length $m$ of a block cipher is 64 (which is the case for many practical block ciphers), one can obtain a 64-bit iterated hash function by using the DM-scheme. The "brute-force" collision attack on any 64-bit hash function has complexity about $2^{32}$, which is certainly too small in many applications. Thus, several efforts [2, 13, 14, 18, 20] have been made to construct a $2m$-bit hash function based on an $m$-bit block cipher by modifying the (apparently secure) DM-scheme. This will be considered in the following sections.

## 3.3 A principle for evaluating hash round functions and four attacks on three $2m$-bit hash round functions

In this section, we point out an obvious (once the 5 attacks have been formulated) but useful principle for evaluating the security of a hash round function, viz. that *applying any simple (in both directions) invertible transformations to the input and to the output of the hash round function yields a new hash round function with the same security as the original one.* [A similar principle has been used by Meier and Staffelbach in [12] to classify nonlinearity criteria for cryptographic functions]. For example, for a block cipher with block length equal to key length, it follows from this principle that the hash round function (2) of the DM-scheme has the same security as the following hash round function proposed in [11]

$$h(H_{i-1}, M_i) = E_{H_{i-1}}(M_i) \bigoplus M_i,$$

since this hash round function differs from that in (2) only by a "swapping" of the input blocks $H_{i-1}$ and $M_i$.

To demonstrate this principle, we present four "meet-in-middle" attacks on three $2m$-bit hash round functions based on an $m$-bit block cipher with an $m$-bit key. The basic purpose of these three schemes is to construct a $2m$-bit hash function based on an $m$-bit block cipher by modifying the (apparently secure) DM-scheme (2). We now show that these $2m$-bit hash round functions are in fact weaker than the $m$-bit hash round function of the DM-scheme. More precisely, for each scheme, we present a free-start target attack that takes only about $2^{m/2}$ (instead of the ideal maximum $2^{2m}$) computations of the round function. [Recall that the free-start target attack on the $m$-bit hash round function in the DM-scheme has complexity $2^m$.]

### 3.3.1 The Preneel-Bosselaers-Govaerts-Vandewalle (PBGV) scheme.

The PBGV scheme was proposed in [18]. In this scheme, which uses an $m$-bit block cipher with an $m$-bit key, a $2m$-bit hash value $H = (H_n, G_n)$ is computed from a $2mn$-bit message $(L_1, N_1, L_2, N_2, ..., L_n, N_n)$ and a $2m$-bit initial value $(H_0, G_0)$. In each round, two new $m$-bit values $H_i$ and $G_i$ are computed from the two previous $m$-bit values $H_{i-1}$ and $G_{i-1}$ and from the two $m$-bit message blocks $L_i$ and $N_i$ as follows:

$$\begin{aligned} H_i &= E_{L_i \bigoplus N_i}(H_{i-1} \oplus G_{i-1}) \oplus L_i \oplus H_{i-1} \oplus G_{i-1} \\ G_i &= E_{L_i \bigoplus H_{i-1}}(N_i \oplus G_{i-1}) \oplus N_i \oplus H_{i-1} \oplus G_{i-1} \end{aligned} \tag{3}$$

for $i = 1, 2, \ldots, n$.

The round function for the PBGV-scheme produces the output pair $(h, g)$ from the inputs $(h_0, g_0, l, n)$ in the manner

$$\begin{aligned} h &= E_{l \bigoplus n}(h_0 \oplus g_0) \oplus l \oplus h_0 \oplus g_0 \\ g &= E_{l \bigoplus h_0}(n \oplus g_0) \oplus n \oplus h_0 \oplus g_0. \end{aligned} \tag{4}$$

By applying the simple and simply inverted transformations

$$(h, g) \longrightarrow (h, f) = (h, h \oplus g) \tag{5}$$

on the output and

$$(h_0, g_0, l, n) \longrightarrow (h_0', g_0', l', n') = (h_0 \oplus g_0, g_0 \oplus n, l \oplus n, n), \tag{6}$$

on the input, we obtain the round function illustrated in Fig.3 that computes $(h, f)$ from the input $(h'_0, g'_0, l', n')$ in the manner

$$\begin{aligned} h &= E_{l'}(h'_0) \oplus l' \oplus n' \oplus h'_0 \\ f &= E_{l' \oplus h'_0 \oplus g'_0}(g'_0) \oplus E_{l'}(h'_0) \oplus l'. \end{aligned} \tag{7}$$

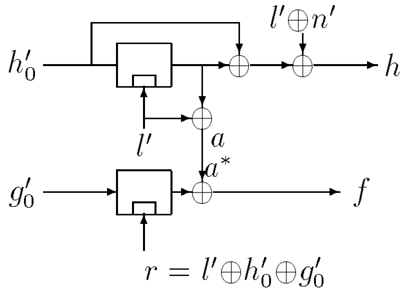

Figure 3: The transformed function used to attack the PBGV round function.

Because the transformations (5) and (6) are both easy to compute and easy to invert, it follows from our principle that an attack on the round function (7) has the same complexity as an attack on the round function (4).

**A free-start target attack on the PBGV round function with complexity about** $2^{m/2}$**:** In this attack, we show how to find a "random inverse" of (7), i.e., we show how, for given $(h, f)$, to find $(h'_0, g'_0, l', n')$ satisfying (4) for which $(h'_0, g'_0)$ appears randomly chosen.

1. Choose an arbitrary constant $c_0$.
2. For the given $h$, compute $a = E_{l'}(h'_0) \oplus l'$ for $2^{m/2}$ randomly chosen values of $h'_0$ and corresponding $l'$ such that $h'_0 \oplus l' = c_0$.
3. For the given $f$, compute $a^* = E_r(g'_0) \oplus f$ for $2^{m/2}$ randomly chosen values of $g'_0$ and corresponding $r$ such that $g'_0 \oplus r = c_0$.

The probability that some $a$ and some $a^*$ take on the same value is about 0.63. For such $(g'_0, r, a = a^*, h'_0, l')$, we obtain a solution $(h'_0, g'_0, l', n')$ for (7) by computing $n' = a \oplus l' \oplus h'_0 \oplus l' \oplus h$. □

[A recent result of Preneel [19] gives a free-start target attack on the PBGV round function that requires only the computation of one decryption with the block cipher.]

**A target attack on the PBGV round function with complexity about** $2^m$**:** In this attack, we find, for the given $(h_0, g_0)$ and $(h, g)$, a message block $(l, n)$ satisfying (4). We will use the notation of Fig.3.

From (5) and (6), we see that $(h, f)$ and $h'_0$ are determined by the given $(h_0, g_0)$ and $(h, g)$. We randomly choose $l'$, then compute

$$a = E_{l'}(h'_0) \oplus l',$$

$$n' = a \oplus h'_0 \oplus h,$$

$$r = l' \oplus h_0' \oplus g_0' = l' \oplus h_0' \oplus g_0 \oplus n'$$

and

$$g_0' = D_r(a \oplus f),$$

where $D_z(y)$ denotes the result of deciphering $y$ with key $z$.

After $2^m$ such computations, $g_0' \oplus n'$ will take on the given value $g_0$ with probability 0.63. Then using (5) and (6), we obtain a solution $(l, n)$ for (4). □

### 3.3.2 The first Quisquater-Girault (QG-I) scheme.

The QG-I scheme was proposed in the Abstracts from Eurocrypt'89 [20]. It also appeared in a draft ISO standard [6], see also [15]. However, this scheme was dropped from the recent version of the draft ISO standard CD10118 [7]. [In unpublished work, Coppersmith pointed out to its inventors some weakness of this scheme [21]. In the subsequent Proceedings paper [21], a "weaker" round function was used, but with additional functional strengthening.] Similarly to the PBGV-scheme discussed above, the QG-I scheme is based on an $m$-bit block cipher with an $m$-bit key. A $2m$-bit hash value $(H_n, G_n)$ is computed from a $2mn$-bit message $(L_1, N_1, L_2, N_2, ..., L_n, N_n)$ and a $2m$-bit initial value $(H_0, G_0)$. In each round, two new $m$-bit values $H_i$ and $G_i$ are computed from the two previous $m$-bit values $H_{i-1}$ and $G_{i-1}$ and from the two $m$-bit message blocks $L_i$ and $N_i$ as follows:

$$\begin{array}{rcl} W_i & = & E_{L_i}(G_{i-1} \oplus N_i) \oplus N_i \oplus H_{i-1} \\ H_i & = & W_i \oplus G_{i-1} \\ G_i & = & E_{N_i}(W_i \oplus L_i) \oplus H_{i-1} \oplus G_{i-1} \oplus L_i \end{array} \qquad (8)$$

for $i = 1, 2, \ldots, n$.

The round function of the QG-I scheme produces the output pair $(h, g)$ from the input $(h_0, g_0, l, n)$ in the manner

$$\begin{array}{rcl} h & = & E_l(g_0 \oplus n) \oplus n \oplus h_0 \oplus g_0 \\ g & = & E_n\left(E_l(g_0 \oplus n) \oplus n \oplus h_0 \oplus l\right) \oplus h_0 \oplus g_0 \oplus l. \end{array} \qquad (9)$$

We will consider the pair $(h, f) = (h, h \oplus g)$ illustrated in Fig.4 and defined by

$$\begin{array}{rcl} h & = & E_l(g_0 \oplus n) \oplus n \oplus h_0 \oplus g_0 \\ f = h \oplus g & = & E_n\left(E_l(g_0 \oplus n) \oplus n \oplus h_0 \oplus l\right) \oplus E_l(g_0 \oplus n) \oplus l \oplus n. \end{array} \qquad (10)$$

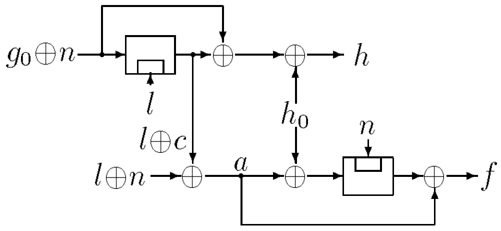

Figure 4: The pair $(h, f)$ used in the attack on the QG-I scheme.

**A free-start target attack on the QG-I scheme with complexity about $2^{m/2}$:** In the following we show that, for any given $(h, f)$, one can find, in about $2^{m/2}$ decrypting computations for the block cipher, a solution $(h_0, g_0, l, n)$ satisfying (10) by a "meet-in-the-middle" attack.

We will use the notation shown in Fig.4. Let $c$ be a fixed $m$-tuple.

1. Randomly choose values for $a$ and choose $n$ such that $a \oplus n = c$. Then, for the given value of $f$, compute $h'_0 = a \oplus D_n(a \oplus f)$. Repeat this process $2^{m/2}$ times to obtain $2^{m/2}$ values for $(h'_0, n)$ with randomly chosen values for $h'_0$.
2. Randomly choose $l$ and compute $h^*_0 = h \oplus (l \oplus c) \oplus D_l(l \oplus c)$. In $2^{m/2}$ computations, one obtains $2^{m/2}$ values for $(h^*_0, l)$ with randomly chosen values for $h^*_0$.

Note that both $h'_0$ and $h^*_0$ are $m$-bit blocks so that some $h'_0$ and some $h^*_0$ obtained as above will take on the same value with probability about 0.63. Thus, we can find $(h'_0, h^*_0, l, n)$ such that $h'_0 = h^*_0$. (Note that the constraint that $l \oplus c \oplus l \oplus n = a$ is automatically satisfied.) From the obtained $(l, n)$, compute $g_0 = D_l(l \oplus c) \oplus n$. Then the resulting $(h_0, g_0, l, n)$ is the desired solution. □

### 3.3.3 The LOKI Double Block Hash (DBH) function.

The block cipher LOKI, proposed in [2], is a DES-like 64-bit block cipher with a 64-bit key. In [2], a 128-bit iterated Double Block Hash (DBH) function based on the cipher LOKI was proposed, but this scheme can in fact be used for any $m$-bit block cipher with an $m$-bit key. In LOKI DBH, a $2m$-bit hash value $(H_n, G_n)$ is computed from a $2mn$-bit message $(L_1, N_1, L_2, N_2, ..., L_n, N_n)$ and a $2m$-bit initial value $(H_0, G_0)$. In each round, two new $m$-bit values $H_i$ and $G_i$ are computed from the two previous $m$-bit values $H_{i-1}$ and $G_{i-1}$ and from the two current $m$-bit message blocks $L_i$ and $N_i$ as follows:

$$\begin{array}{rcl} W_i & = & E_{L_i \oplus G_{i-1}}(G_{i-1} \oplus N_i) \oplus N_i \oplus H_{i-1} \\ H_i & = & W_i \oplus G_{i-1} \\ G_i & = & E_{N_i \oplus H_{i-1}}(W_i \oplus L_i) \oplus H_{i-1} \oplus G_{i-1} \oplus L_i \end{array} \qquad (11)$$

for $i = 1, 2, \ldots, n$.

The LOKI DBH round function was derived from the hash round function of the QG-I scheme (8) by the bitwise addition modulo 2 of the previous hash value blocks ($H_{i-1}$ and $G_{i-1}$) to the current message blocks ($L_i$ and $N_i$) to obtain the key inputs for the two LOKI encryptions. This was done to avoid some attacks derived from the 'weak key' of the underlying cipher. By applying our security evaluation principle, we obtain the following free-start target attack on the LOKI DBH round function that has complexity only about $2^{m/2}$.

The round function for the LOKI DBH produces the output pair $(h, g)$ from the input $(h_0, g_0, l, n)$ in the manner

$$\begin{array}{rcl} h & = & E_{l \oplus g_0}(g_0 \oplus n) \oplus n \oplus h_0 \oplus g_0 \\ g & = & E_{n \oplus h_0}\left(E_{l \oplus g_0}(g_0 \oplus n) \oplus n \oplus h_0 \oplus l\right) \oplus h_0 \oplus g_0 \oplus l. \end{array} \qquad (12)$$

By applying the transformation

$$(h, f) = (h, h \oplus g) \tag{13}$$

on the LOKI DBH output pair $(h, g)$ and applying the transformation

$$(h_0, g_0, l', n') = (h_0, g_0, l \oplus g_0, n \oplus g_0) \tag{14}$$

on the LOKI DBH inputs $(h_0, g_0, l, n)$, we obtain the function illustrated in Fig.5 that computes $(h, f)$ from the inputs $(h_0, g_0, l', n')$ in the manner

$$\begin{array}{rcl} h &=& E_{l'}(n') \oplus n' \oplus h_0 \\ f &=& E_{n' \oplus h_0 \oplus g_0}(h \oplus l') \oplus h \oplus l' \oplus h_0. \end{array} \tag{15}$$

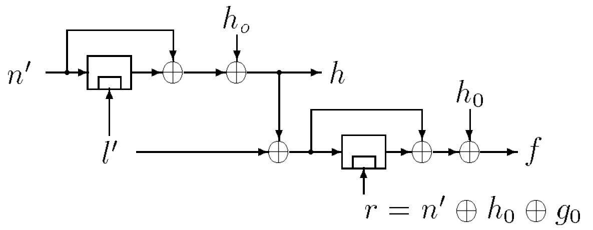

Figure 5: The new function used to attack the LOKI DBH round function.

**A free-start target attack on the LOKI DBH with complexity about $2^{m/2}$:** In the following, we show that, for any given $(h, f)$, one can find, in about $2 \times 2^{m/2}$ encrypting computations for the block cipher, a solution for $(h_0, g_0, l, n)$ satisfying (10) by a "meet-in-the-middle" attack.

Because the transformations (13) and (14) are both easy to compute and easy to invert, it follows from our principle that finding a solution $(h_0, g_0, l, n)$ of (12) for a given $(h, g)$ is computationally the same as finding a solution $(h_0, g_0, l', n')$ of (15) for a given $(h, f)$. This can be done in about $2 \times 2^{m/2}$ encryptions as we now show.

1. Choose an arbitrary value for $l'$.
2. For the given $h$ and the chosen $l'$, compute $h_0 = h \oplus n' \oplus E_{l'}(n')$ for $2^{m/2}$ randomly chosen values of $n'$.
3. For the given $h, f$ and the chosen $l'$, compute $h_0^* = E_r(h \oplus l') \oplus h \oplus l' \oplus f$ for $2^{m/2}$ randomly chosen values of $r$ $(= n \oplus h_0 \oplus g_0)$.

The probability that some $h_0$ and some $h_0^*$ take on the same value is about 0.63. For $h_0 = h_0^*$, by computing $g_0 = r \oplus n' \oplus h_0$, we obtain a solution $(h_0, g_0, l', n')$ for (15). □

**Remark.** We have given three free-start target attacks on three hash round functions in this section. The "real" target attacks (with specified initial value) will usually be more difficult. For example, when $m$ is 64 bits, a target attack on the 128-bit hash function LOKI DBH obtained by combining the above attack with the attack used in the proof of Theorem 3 will take about $2^{\frac{128-32}{2}} = 2^{80}$ computations. A similar conclusion holds also for the QG-I scheme hash function.

## 3.4 Complexity of known attacks on $2m$-bit hash functions

We consider here some known 128-bit iterated hash functions based on two uses of an $m = 64$-bit block cipher with key-length $k = 64$ or $k = 56$ in each round. All these schemes can be considered as slight modifications of the 64-bit DM-scheme hash round function. The complexities of known attacks on these hash functions are listed in Table 1. We assume that all the iterated hash functions are used with MD-strengthening and that the underlying block cipher has no known weakness (such as weak keys).

| $h(\cdot,\cdot)$ | PBGV | GQ-I | LOKI-DBH | Merkle$_{\leadsto 12}$ | M-S$_{\leadsto 13}$ | ideal |
|---|---|---|---|---|---|---|
| $(m,k)_{\leadsto 1}$ | (64,64) | (64,64) | (64,64) | (64,56) | (64,56) | (64,k) |
| target | $2^{64}_{\leadsto 2}$ | $2^{80}_{\leadsto 5}$ | $2^{80}_{\leadsto 9}$ | $2^{112}$ | $2^{81}_{\leadsto 14}$ | $2^{128}$ |
| f-s target | $o(1)_{\leadsto 3}$ | $2^{32}_{\leadsto 6}$ | $2^{32}_{\leadsto 10}$ | $2^{112}$ | $2^{54}_{\leadsto 15}$ | $2^{128}$ |
| collision | $2^{32}_{\leadsto 3}$ | $2^{64}$ | $2^{64}$ | $2^{56}$ | $2^{54}$ | $2^{64}$ |
| semi-f-s col. | $2^{32}_{\leadsto 3}$ | $2^{32}_{\leadsto 7}$ | $2^{64}$ | $2^{56}$ | $2^{54}$ | $2^{64}$ |
| f-s coll. | $o(1)_{\leadsto 4}$ | $o(1)_{\leadsto 8}$ | $2^{32}_{\leadsto 11}$ | $2^{56}$ | $2^{27}_{\leadsto 16}$ | $2^{64}$ |
| leng$(M_i)$ | 128 | 128 | 128 | 7 | 64 | $l_{\leadsto 17}$ |

$\leadsto$ 1: $m$: block-length, $k$: key-length of the underlying cipher;

$\leadsto$ 2: see last section;

$\leadsto$ 3: recent results of Preneel [19];

$\leadsto$ 4: a free-start collision attack is no harder than a free-start target attack;

$\leadsto$ 5: from the free-start target attack$_{\leadsto 6}$ and Proposition 3;

$\leadsto$ 6: see last section;

$\leadsto$ 7,8: see [16];

$\leadsto$ 9,10: same as $\leadsto$ 5,6;

$\leadsto$ 11: same as $\leadsto$ 4;

$\leadsto$ 12: Merkle's scheme [13]: hash-code is of length 112 bits; this scheme appears to have ideal security; however, each round can 'digest' only 7 bits of message;

$\leadsto$ 13: Meyer-Schilling's scheme [14]: 128-bit hash code, but round output has length 108 bits;

$\leadsto$ 14,15: each round output (two blocks) has length 108 bits; a free-start target attack on one (54-bit) block takes about $2^{54}$ computations; then use Proposition 3; see also [14];

$\leadsto$ 16: collision is achieved on one (54-bit) block.

$\leadsto$ 17: see next section.

Table 1: Complexity of known attacks on some hash round functions.

## 3.5 Proposed schemes for block ciphers with $k = 2m$

The study of previously proposed hashing schemes (see Table 1) suggests that it is difficult, if not impossible, to build a $2m$-bit hash round function with ideal computational security that can "digest" in each round at least $m$ bits of message by two uses of an $m$-bit block cipher with an $m$-bit key. However, if an $m$-bit block cipher with a $2m$-bit key is available, then there are more possibilities to construct a possibly secure $2m$-bit hash round function. In the following, we propose two $2m$-bit hash round functions that use an $m$-bit block cipher with a $2m$-bit key and that appear to be secure.

**Tandem DM:** We refer to our first proposed $2m$-bit hash function as the *Tandem* DM scheme because it is based on cascading two DM-schemes as in (2). The round function of the Tandem DM scheme is shown in Fig.6. In each iteration, two new $m$-bit values $H_i$ and $G_i$ are computed from the two previous $m$-bit values $H_{i-1}$ and $G_{i-1}$ and from an $m$-bit message block $M_i$ as follows:

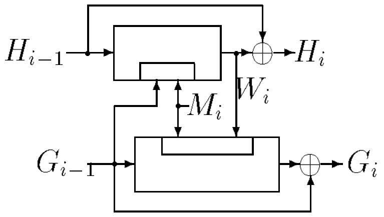

Figure 6: The Tandem DM $2m$-bit hash round function based on an $m$-bit block cipher with a $2m$-bit key.

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

**Abreast DM** We next propose the *Abreast* DM scheme in which two DM-schemes are used side-by-side. The hash round function is illustrated in Fig.7. In each round, two new $m$-bit values $(H_i, G_i)$ are computed from the two previous $m$-bit values $(H_{i-1}, G_{i-1})$ and from an $m$-bit message block $M_i$ 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 $\overline{G}$ denotes the bit-by-bit complement of $G$.

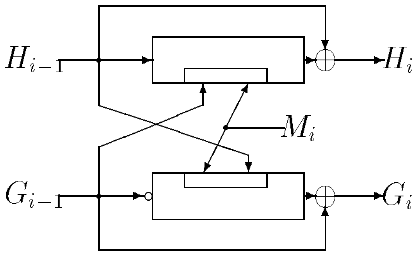

Figure 7: The Abreast DM $2m$-bit hash round function based on an $m$-bit block cipher with a $2m$-bit key. The circle indicates that the input to the lower encrypter is bitwise complemented.

**Remarks:** 1. The Tandem DM and the Abreast DM schemes were constructed on the following consideration. The round function $h$ consists of two subfunctions $h_1$ and $h_2$:

$$(H_i, G_i) = h(H_{i-1}, G_{i-1}, M_i) = [h_1(H_{i-1}, G_{i-1}, M_i), h_2(H_{i-1}, G_{i-1}, M_i)],$$

both of which have the same inputs. Thus, to attack $h$ (in a free-start target or free-start collision attack) implies that one must attack both $h_1$ and $h_2$ simultaneously. If the subfunctions $h_1$ and $h_2$ are so 'different' that an attack on one subfunction provides no help in attacking the other subfunction and if both $h_1$ and $h_2$ are equivalent (in the sense of security) to the apparently secure DM-scheme, then we can expect that an attack on $h$ will have complexity equal to the product of the complexities of the attacks on $h_1$ and on $h_2$. In the proposed Tandem DM and Abreast DM schemes, the subfunctions $h_1$ and $h_2$ are chosen to be as "different" as possible.

2. The Abreast DM scheme gives a $2m$-bit hash function that is at least as strong as the $m$-bit DM-scheme. [This is true also for the Meyer-Schilling scheme [7, 14].]

3. Our investigations to this point have shown no weakness in either of these two new proposed $2m$-bit hash round functions, i.e., we have been unable to find any attacks better than brute-force attacks when the underlying cipher is assumed to have no weakness. We should point out, however, that our Tandem DM and Abreast DM schemes use two $m$-bit block encryptions for each block of $m$ message bits in order to compute a final hash value of length $2m$ bits.

## Acknowledgements

The authors are grateful to Prof. F. Piper, Dr. R. Rueppel and the anonymous referee for informing them about recent developments related to ISO standard CD10118. The authors would like in particular to thank B. Preneel for his many useful comments on this paper. This research was supported by the Swiss Commission for the Advancement of Scientific Research, Research Grant KWF 2146.1.

# References

[1] S. G. Akl, "On the Security of Compressed Encodings", Advances in Cryptology-CRYPTO'83, Proceedings, pp. 209-230, Plenum Press, New York, 1984.

[2] L. Brown, J. Pieprzyk and J. Seberry, "LOKI – A Cryptographic Primitive for Authentication and Secrecy Applications", Advances in Cryptology – AUSCRYPT'90, Proceedings, LNCS 453, pp. 229-236, Springer-Verlag, 1990.

[3] *Data Encryption Standard*, FIPS PUB 46, National Tech. Info. Service, Springfield, VA, 1977.

[4] I. B. Damgaard, "A Design Principle for Hash Functions", Advances in Cryptology-CRYPTO'89, LNCS 435, pp. 416-427, Springer-Verlag, 1990.

[5] R. W. Davies and W. L. Price, "Digital Signature – an Update", Proc. International Conference on Computer Communications, Sydney, Oct 1984, Elsevier, North-Holland, pp. 843-847, 1985.

[6] I.S.O. DP 10118, *Hash-functions for Digital Signatures*, I.S.O., April 1989.

[7] ISO/IEC CD 10118, *Information technology – Security techniques – Hash-functions*, I.S.O., 1991.

[8] X. Lai and J. L. Massey, "A Proposal for a New Block Encryption Standard", Advances in Cryptology-EUROCRYPT'90, Proceedings, LNCS 473, pp. 389-404, Springer-Verlag, Berlin, 1991.

[9] X. Lai, J. L. Massey and S. Murphy, "Markov Ciphers and Differential Cryptanalysis", Advances in Cryptology-EUROCRYPT'91, Proceedings, LNCS 547, pp. 17-38, Springer-Verlag, Berlin, 1991.

[10] S. M. Matyas, "Key Processing with Control Vectors", Journal of Cryptology, Vol. 3, No. 2, pp. 113–136, 1991.

[11] S. M. Matyas, C. H. Meyer and J. Oseas, "Generating Strong One-way Functions with Cryptographic Algorithm", IBM Technical Disclosure Bulletin, Vol. 27, No. 10A, pp. 5658-5659, March 1985.

[12] W. Meier, O. Staffelbach, " Nonlinearity Criteria for Cryptographic Functions", Advances in Cryptology - EUROCRYPT'89, Proceedings, LNCS 434, pp. 549-562, Springer-Verlag, 1990.

[13] R. C. Merkle, "One Way Hash Functions and DES", Advances in Cryptology-CRYPTO'89, Proceedings, LNCS 435, pp. 428-446, Springer-Verlag, 1990.

[14] C. H. Meyer and M. Schilling, "Secure Program Code with Modification Detection Code", Proceedings of SECURICOM 88, pp. 111-130, SEDEP.8, Rue de la Michodies, 75002, Paris, France.

[15] C. J. Mitchell, F. Piper and P. Wild, "Digital Signatures", *Contemporary Cryptology* (Ed. G. Simmons), pp. 325-378, IEEE Press, 1991.

[16] S Miyaguchi, K. Ohta and M. Iwata, "Confirmation that Some Hash Functions Are Not Collision Free", Advances in Cryptology-EUROCRYPT'90, Proceedings, LNCS 473, pp. 326-343, Springer-Verlag, Berlin, 1991.

[17] M. Naor and M. Yung, "Universal One-way Hash Functions and Their Cryptographic Applications", Proc. 21 Annual ACM Symposium on Theory of Computing, Seattle, Washington, May 15-17, 1989, pp. 33-43.

[18] B. Preneel, A. Bosselaers, R. Govaerts and J. Vandewalle, "Collision-free Hashfunctions Based on Blockcipher Algorithms." Proceedings of 1989 International Carnahan Conference on Security Technology, pp. 203-210.

[19] Private communication, B. Preneel to X. Lai, June 1992.

[20] J. J. Quisquater and M. Girault, "2n-bit Hash Functions Using n-bit Symmetric Block Cipher Algorithms", Abstracts of EUROCRYPT'89.

[21] J. J. Quisquater and M. Girault, "2n-bit Hash Functions Using n-bit Symmetric Block Cipher Algorithms", Advances in Cryptology-EUROCRYPT'89, Proceedings, LNCS 434, pp. 102-109, Springer-Verlag, Berlin, 1990.

[22] R. S. Winternitz, "Producing One-Way Hash Function from DES", Advances in Cryptology-CRYPTO'83, Proceedings, pp. 203-207, Plenum Press, New York, 1984.

[23] R. S. Winternitz, "A Secure One-way Hash Function Built from DES", Proc. 1984 IEEE Symposium on Security and Privacy, Oakland, 1984, pp. 88-90.