Indifferentiability of Single-Block-Length and Rate-1 Compression Functions

Hidenori Kuwakado Masakatu Morii

Faculty of Engineering
Kobe University
1-1 Rokkodai-cho Nada-ku Kobe
657-8501, Japan

December 26, 2006

Abstract

The security notion of indifferentiability was proposed by Maurer, Renner, and Holenstein in 2004. In 2005, Coron, Dodis, Malinaud, and Puniya discussed the indifferentiability of hash functions. They have showed that the Merkle-Damgård construction is not secure in the sense of indifferentiability. In this paper, we analyze the security of single-block-length and rate-1 compression functions in the sense of indifferentiability. We formally show that all single-block-length and rate-1 compression functions, which include the Davies-Meyer compression function, are insecure. Furthermore, we show how to construct a secure single-block-length and rate-1 compression function in the sense of indifferentiability. This does not contradict our result above.

1 Introduction

Most of the hash functions are designed as iterative processes which hash an arbitrary-length message by processing successive fixed-size blocks of the message. A function for processing the fixed-size blocks is called a compression function. The Merkle-Damgård construction is widely used to construct a hash function by the iteration of the compression function [4][10]. To improve the security, other constructions have been proposed recently (e.g., [3][7][14]).

In all constructions, the security of the underlying compression function is closely related with that of the constructed hash function. Compression functions are usually designed using secure block ciphers. For example, the Davies-Meyer compression function, the Matyas-Meyer-Oseas compression function, and the Miyaguchi-Preneel compression function, which are the most popular compression functions, are based on block ciphers [9]. Compression functions can be classified as the hash length and the number of calls. Let n be the length of an output of the underlying block cipher, and let ℓ denote the length of an output of the compression function.

Hash length: If $\ell/n = 1$, then the compression function is called a single-block-length compression function. If $\ell/n = 2$, then it is called a double-block-length compression function.

Number of calls: Let σ be the number of calling the underlying block cipher to produce one output of the compression function. A rate r of the compression function is defined as $r = \ell/(n\sigma)$, which represents the efficiency of the compression function. A compression function with a rate of r is called a rate-r compression function.

The security of compression functions has been discussed in the sense of preimage resistance, second-preimage resistance, and collision resistance. In particular, the security of collision resistance is theoretically and practically important. From the theoretical viewpoint, the complexity of collision resistance is not larger than $O(2^{n/2})$ where n is the output length. This fact is often called the birthday bound. From the practical viewpoint, drawbacks of actual hash functions have been firstly found in terms of collision resistance [5][15][16]. In addition to the three security notions above, the notion of indifferentiability has been introduced to the security of hash functions and compression functions recently [3][6]. The notion of indifferentiability is stronger than that of collision resistance. To design more secure hash functions or more secure compression functions, it is necessary to discuss the security in the sense of indifferentiability.

In this paper, we restrict our attention to single-block-length and rate-1 compression functions (SBL-1 compression functions). The class of SBL-1 compression functions is practically important. For example, the popular compression functions above are included in this class. The efficiency of SBL-1 compression functions is optimal because, by definitions, the rate of any single-block-length compression function is not larger than 1.

The security of SBL-1 compression functions has been studied in detail. Preneel, Govaerts, and Vandewalle [13] have discussed the security against several attacks, and concluded that 12 compression functions are secure against the attacks. We note that 64 possible SBL-1 compression functions exist. Black, Rogaway, and Shrimpton [2] have investigated the provable security of the 64 possible SBL-1 compression functions. They concluded that the 12 compression functions of Preneel et al. is optimally collision resistant.[1] Black, Cochran, and Shrimpton [1] have shown that no rate-1 compression function can produce a provable collision-resistant hash function if the key space is a small fixed set. Their result is practically significant because the practical hash speed would be improved by the precomputation of the key schedule algorithm of the block cipher if the key space were small. Accordingly, SBL-1 compression functions must be designed in such a way that the key space is fully used.

The notion of indifferentiability has been introduced by Maurer, Renner, and Holenstein [8] to discuss the extended indistinguishability of systems. Coron, Dodis, Malinaud, and Puniya [3] have applied the notion of indifferentiability to the construction of hash functions from underlying compression functions. They showed that the Merkle-Damgård construction is insecure in the sense of indifferentiability, and they also proposed some methods for enhancing the security of the Merkle-Damgård construction. In this paper, we discuss the security of SBL-1 compression functions in terms of indifferentiability, not hash functions. Indeed, Coron et al. [3] briefly discussed the indifferentiability only of the Davies-Meyer compression function. Hirose [6] discussed the indifferentiability of double-block-length compression functions with the special-form input. This paper formally discusses the indifferentiability of the 12 SBL-1 compression functions, which are secure in the sense of collision resistance [2][13]. Furthermore, this paper proposes a new SBL-1 compression function that is secure in the sense of indifferentiability.

The organization and the contribution of this paper are as follows. In Sect. 2, we describe notations and definitions about indifferentiability. The definition of indifferentiability is the same as that of previous papers [3][8], but we introduce an experiment to evaluate the advantage of a distinguisher.

In Sect. 3, we discuss the indifferentiability of the Davies-Meyer compression function. When the distinguisher can have access only to the encryption oracle, the Davies-Meyer compression function is not so insecure. However, when the distinguisher can have access to the encryption oracle and the decryption oracle, the Davies-Meyer compression function is insecure. This drawback was briefly pointed out by Coron et al. [3], but they did not analyze the security formally. We give the formal analysis of the Davies-Meyer compression function in terms of indifferentiability.

In Sect. 4, we show that the negative result of the Davies-Meyer compression function is applicable to other SBL-1 compression functions. As a result, all the 64 SBL-1 compression functions have the security drawback in terms of preimage resistance, second-preimage resistance, collision resistance, or indifferentiability.

In Sect. 5, we propose a new SBL-1 compression function, which is called a block-cipher-selection compression function (a BCS compression function). The feature of the BCS compression function is that one block cipher is selectively used among many block ciphers to produce an output for a given

[1]They also showed that 8 compression functions are not collision resistant, but hash functions based on the 8 compression functions are collision resistant. This is not the Merkle-Damgård paradigm. The other 44 compression functions are not suitable for hash functions.

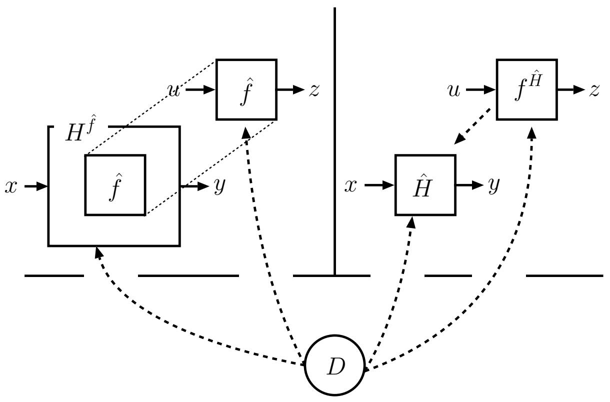

Figure 1: The indifferentiability between $H^{\hat{f}}$ and $\hat{H}$.

input. We prove that the BCS compression function is secure in the sense of indifferentiability. We explain why this result does not contradict the result of Sect. 4. In Sect. 6, we summarize results of this paper and problems in the future.

2 Notations and Definitions

We will write $a \leftarrow b$ to mean that a is to be set to the result of evaluating expression b, and write $a \overset{\$}{\leftarrow} \mathcal{A}$ to mean that a is uniformly chosen at random from a finite set $\mathcal{A}$. Let κ, n be positive integers. A block cipher is a function e: $\{0,1\}^\kappa \times \{0,1\}^n \rightarrow \{0,1\}^n$ where, for each $w \in \{0,1\}^\kappa$, $e(w,\cdot)$ is a permutation on $\{0,1\}^n$. For the block cipher e, the inverse (the decryption function) is denoted by d, i.e., $d(w,z)$ is the string x such that $z = e(w,x)$. Let $\mathcal{E}_{\kappa,n}$ be the set of all block ciphers e: $\{0,1\}^\kappa \times \{0,1\}^n \rightarrow \{0,1\}^n$. A block cipher $\hat{e}$ is said to be an ideal block cipher if $\hat{e}$ is uniformly selected at random from $\mathcal{E}_{\kappa,n}$, i.e., if $\hat{e} \overset{\$}{\leftarrow} \mathcal{E}_{\kappa,n}$. Let $\mathcal{H}_{m,n}$ be the set of all functions from $\{0,1\}^m$ to $\{0,1\}^n$. A function $\hat{H}$ is said to be a random function if $\hat{H}$ is randomly selected from $\mathcal{H}_{m,n}$, i.e., if $\hat{H} \overset{\$}{\leftarrow} \mathcal{H}_{m,n}$. By definitions, it holds that $\mathcal{E}_{\kappa,n} \subset \mathcal{H}_{\kappa+n,n}$.

In this paper, we discuss the construction of a single-block-length and rate-1 compression function H based on the block cipher $e \in \mathcal{E}_{n,n}$. That is, H is a function in $\mathcal{H}_{2n,n}$ and requires one invocation of e to produce an output for a given input. Our goal is to construct the compression function H that is indistinguishable from the random function $\hat{H} \in \mathcal{H}_{2n,n}$. In previous works, the security of the compression function based on the block cipher has been studied in terms of preimage resistance, second-preimage resistance, and collision resistance. Since the random function $\hat{H}$ naturally satisfies these security requirements with desirable level, our goal is reasonable. Our idea is similar to that of Coron et al. [3] They considered that the random function was ideal of the hash function. Accordingly, they studied how to construct a hash function $\{0,1\}^* \rightarrow \{0,1\}^n$ that is indistinguishable from a random function $\{0,1\}^* \rightarrow \{0,1\}^n$.

To measure the indifferentiability between two functions, the following definition has been introduced in [3][8] (Fig. 1).

Definition 1 *A function $H^{\hat{f}}$ with oracle access to an ideal primitive $\hat{f}$ is said to be (t_D, t_f, q, ϵ) indifferentiable from an ideal primitive $\hat{H}$ if there exists a simulator $f^{\hat{H}}$ such that the following equation holds for any distinguisher D.*

$$\left| \Pr\left[D^{H^{\hat{f}},\hat{f}} = 1\right] - \Pr\left[D^{\hat{H},f^{\hat{H}}} = 1\right] \right| < \epsilon$$

The distinguisher has access to not only $H^{\hat{f}}$ but also $\hat{f}$ (similarly, not only $\hat{H}$ but also $f^{\hat{H}}$). The simulator $f^{\hat{H}}$ has oracle access to $\hat{H}$ and runs in time at most t_f. The distinguisher runs in time at

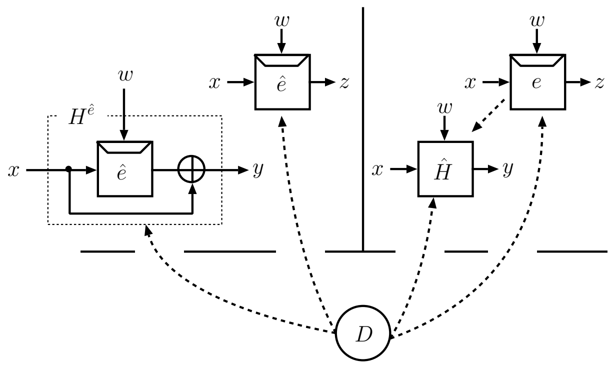

Figure 2: The indifferentiability of the Davies-Meyer compression function.

most t_D and makes at most q queries. If ϵ is a negligible function of the security parameter k, then $H^{\hat{f}}$ is said to be indifferentiable from $\hat{H}$.

When we focus on the probability ϵ, let us consider the following experiment to evaluate it.

step 1: Choose a bit $\xi \in \{0,1\}$ at random, then determine $(\overline{H}, \overline{f})$ as follows:

$$(\overline{H}, \overline{f}) = \begin{cases} (H^{\hat{f}}, \hat{f}) & \text{if } \xi = 0, \\ (\hat{H}, f) & \text{otherwise.} \end{cases}$$

The distinguisher does not know the value of ξ.

step 2: After the distinguisher has access to $(\overline{H}, \overline{f})$, the distinguisher guesses the value of ξ, denoted by ξ_D.

Using the above experiment, we define the advantage of the distinguisher D with at most q queries as follows:

$$\mathbf{Adv}_H(D, q) = \Pr[\xi = \xi_D] - \frac{1}{2},$$

where $\Pr[\xi = \xi_D]$ is the probability that D can correctly guess the value of ξ with at most q queries. We note that if D determines ξ_D at random, then $\Pr[\xi = \xi_D]$ is equal to 1/2. Thus, it can be assumed that $\Pr[\xi = \xi_D]$ is not smaller than 1/2. We use the value of $\mathbf{Adv}_H(D, q)$ as the value of ϵ of Definition 1. We define the achievable advantage with q queries as

$$\mathbf{Adv}_H(q) = \max_D \{\mathbf{Adv}_H(D, q)\}.$$

In this paper, we denote by a symbol without '$\hat{}$' an actual algorithm, and denote by a symbol with '$\hat{}$' an ideal function. In addition, we denote by a symbol with '$\overline{}$' either the actual algorithm or the ideal function. For example, when we argue a block cipher, g is an actual block cipher, $\hat{g}$ is an ideal block cipher, and $\overline{g}$ is either the actual block cipher g or the ideal block cipher $\hat{g}$.

3 Davies-Meyer Compression Function

There are 12 compression functions that are secure in the sense of collision resistance [2]. The Davies-Meyer compression function, which is widely used in actual hash algorithms such as SHA-2 [12], is one of the 12 compression functions. In this section we discuss the indifferentiability of the Davies-Meyer compression function (Fig. 2). The Davies-Meyer compression function H is defined as

$$H(w, x) = \hat{e}(w, x) \oplus x, \tag{1}$$

where $\hat{e}(w, x)$ is the ideal block cipher in $\mathcal{E}_{n,n}$.

3.1 Only Encryption Oracle

We discuss the indifferentiability between the Davies-Meyer compression function and the random function when the distinguisher is allowed to have access to only $\overline{e}$. One must construct a simulator $e^{\hat{H}}$ such that interacting $(H^{\hat{e}}, \hat{e})$ is indistinguishable from interacting $(\hat{H}, e^{\hat{H}})$. We assume that the distinguisher does not make the same query. Our simulator $e^{\hat{H}}$ is defined as follows:

step 1: Initially, $i \leftarrow 0$ and $e^{\hat{H}}(w, x)$ is undefined for any $(w, x) \in \{0,1\}^n \times \{0,1\}^n$.

step 2: $e^{\hat{H}}$ receives a query (w, x) from the distinguisher. $i \leftarrow i+1$, $w_i \leftarrow w$, $x_i \leftarrow x$, $y_i \leftarrow \hat{H}(w_i, x_i)$, $z_i \leftarrow y_i \oplus x_i$.

step 3: $e^{\hat{H}}(w_i, x_i) \leftarrow z_i$ and $e^{\hat{H}}$ returns $e^{\hat{H}}(w_i, x_i)$.

The above simulator $e^{\hat{H}}$ may fail to emulate the ideal block cipher $\hat{e}$. Specifically, if $e^{\hat{H}}(w_i, x_i) = e^{\hat{H}}(w_\iota, x_\iota) \wedge w_i = w_\iota$ for $\exists \iota \in \{1, 2, \ldots, i-1\}$, then $e^{\hat{H}}$ fails to emulate $\hat{e}$ because $e^{\hat{H}}_{w_i}$ is no longer a permutation on $\{0,1\}^n$. Let C_i be the event that there exists such an $\iota \in \{1, 2, \ldots, i-1\}$ at the i-the query and there does not exist such an ι at previous queries. As long as C_i does not occur, $e^{\hat{H}}$ can perfectly emulate $\hat{e}$. Since the probability $\Pr[\mathsf{C}_i]$ is not larger than $(i-1)/2^n$, the probability $\Pr[\mathsf{fail}]$ that $e^{\hat{H}}$ fails to emulate $\hat{e}$ is given as follows:

$$\begin{aligned} \Pr[\mathsf{fail}] &= \Pr[\mathsf{C}_1 \vee \mathsf{C}_2 \vee \ldots \vee \mathsf{C}_q] \\ &\le \sum_{i=1}^{q} \Pr[\mathsf{C}_i] \\ &\le \sum_{i=1}^{q} \frac{i-1}{2^n} = \frac{q(q-1)}{2^{n+1}} \end{aligned}$$

Since the following equation holds for any distinguisher D,

$$\begin{aligned} \Pr[\xi = \xi_D] &= \Pr[\xi = \xi_D | \neg\mathsf{fail}] \Pr[\neg\mathsf{fail}] + \Pr[\xi = \xi_D | \mathsf{fail}] \Pr[\mathsf{fail}] \\ &\le \frac{1}{2} + \frac{1}{2}\Pr[\mathsf{fail}] \end{aligned}$$

we have

$$\begin{aligned} \mathbf{Adv}_H(q) &\le \frac{1}{2}\Pr[\mathsf{fail}] \\ &\le \frac{q(q-1)}{2^{n+2}}. \end{aligned} \quad (2)$$

By demonstrating a distinguisher that achieves advantage close to the above upper bound, we show that the bound is tight. Our distinguisher D is defined as follows:

step 1: D randomly chooses w_0 from $\{0,1\}^n$.

step 2: For $i = 1, 2, \ldots, q$, D performs the following. D sends (w_0, i) to $\overline{H}$, and receives $y_i = \overline{H}(w_0, i)$. D computes $z_i = y_i \oplus i$

step 3: If there exists a pair of (ι, λ) such that $z_\iota = z_\lambda$ in $\{z_1, z_2, \ldots, z_q\}$, then D outputs $\xi_D = 1$, otherwise D outputs $\xi_D = 0$.

We note that if $\overline{H} = H^{\hat{e}}$, i.e., $\xi = 0$, then there is no such a pair. We evaluate the probability $\Pr[\mathsf{col}]$ that there is such a pair as follows:

$$\begin{aligned} \Pr[\mathsf{col}] &= 1 - \prod_{i=1}^{q-1} \left(1 - \frac{i}{2^n}\right) \\ &\ge 1 - \prod_{i=1}^{q-1} \exp\left(-\frac{i}{2^n}\right) \\ &\ge 1 - \exp\left(-\frac{q(q-1)}{2^{n+1}}\right), \end{aligned}$$

where we used the fact that $1 - x \le \exp(-x)$ for $0 \le x \le 1$. Since the following equation holds,

$$\begin{aligned}\Pr[\xi = \xi_D] &= \Pr[\xi = \xi_D | \neg\mathsf{col}] \Pr[\neg\mathsf{col}] + \Pr[\xi = \xi_D | \mathsf{col}] \Pr[\mathsf{col}] \\ &= \frac{1}{2} + \frac{1}{2}\Pr[\mathsf{col}],\end{aligned}$$

we have

$$\begin{aligned}\mathbf{Adv}_H(D, q) &= \Pr[\xi = \xi_D] - \frac{1}{2} \\ &= \frac{1}{2}\Pr[\mathsf{col}] \\ &\ge \frac{1}{2} - \frac{1}{2}\exp\left(-\frac{q(q-1)}{2^{n+1}}\right). \qquad (3)\end{aligned}$$

Using the fact that $\exp(-x)$ gives a close approximation to $1 - x$ when x is close to 0, we observe that the lower bound of Eq. (3) is approximately equal to the upper bound of Eq. (2) if q is sufficiently smaller than 2^n. We notice that the above distinguisher does not have access to $\overline{e}$. The availability of $\overline{e}$ is not effective for this distinguisher.

3.2 Encryption and Decryption Oracles

We discuss the indifferentiability between the Davies-Meyer compression function and the random function when the distinguisher is allowed to have access to both of the encryption oracle $\overline{e}$ and the decryption oracle $\overline{d}$. Although one should construct a simulator $(e^{\hat{H}}, d^{\hat{H}})$ such that interacting $(H^{\hat{e}}, (\hat{e}, \hat{d}))$ is indistinguishable from interacting $(\hat{H}, (e^{\hat{H}}, d^{\hat{H}}))$, it is impossible to construct such a simulator. This drawback was briefly pointed out by Coron at el. [3], but they did not discuss the differentiability quantitatively. In this section, we analyze the differentiability by demonstrating a distinguisher.

Let us consider the distinguisher D defined below.

step 1: For $i = 1, 2, \ldots, q$, D performs the following. D sends a query $(i, 0)$ to $\overline{d}$, and receives $x_i = \overline{d}(i, 0)$. D sends a query (i, x_i) to $\overline{H}$, and receives $y_i = \overline{H}(i, x_i)$.

step 2: If $x_j = y_j$ for $\forall j \in \{1, 2, \ldots, q\}$, then D outputs $\xi_D = 0$, otherwise D outputs $\xi_D = 1$.

If $\overline{H} = H^{\hat{e}}$, i.e., $\xi = 0$, then it holds that $x_j = y_j$ for $\forall j \in \{1, 2, \ldots, q\}$ from Eq. (1). If $\overline{H} = \hat{H}$, i.e., $\xi = 1$, then there may exist ι such that $x_\iota \ne y_\iota$ for $\exists \iota \in \{1, 2, \ldots, q\}$. When $\overline{H} = \hat{H}$, the probability $\Pr[\mathsf{neq}]$ that there exists such an ι is given by

$$\begin{aligned}\Pr[\mathsf{neq}] &= 1 - \left(1 - \left(1 - \frac{1}{2^n}\right)^{2^n}\right)^q \\ &\ge 1 - (1 - \exp(-1))^q \\ &\ge 1 - \left(\frac{64}{100}\right)^q.\end{aligned}$$

Hence, if $\overline{H} = \hat{H}$, then the probability that $\xi_D = 1$ exponentially tends to 1 as q increases. For example, when $q = 10$, we have $\Pr[\mathsf{neq}] \ge 0.98$. Using the following equation,

$$\begin{aligned}\Pr[\xi = \xi_D] &= \Pr[\xi = \xi_D | \neg\mathsf{neq}] \Pr[\neg\mathsf{neq}] + \Pr[\xi = \xi_D | \mathsf{neq}] \Pr[\mathsf{neq}] \\ &= \frac{1}{2} + \frac{1}{2}\Pr[\mathsf{neq}]\end{aligned}$$

we obtain the advantage of the distinguisher D as follows:

$$\begin{aligned}\mathbf{Adv}_H(D, q) &= \frac{1}{2}\Pr[\mathsf{neq}] \\ &\ge \frac{1}{2}\left(1 - \left(\frac{64}{100}\right)^q\right) \qquad (4)\end{aligned}$$

In the above discussion, we did not say anything about the algorithm of the simulator $d^{\hat{H}}$. Because, even if $d^{\hat{H}}$ is intelligently constructed, the advantage cannot be decreased. To decrease the advantage, $d^{\hat{H}}$ must find x such that $x = \hat{H}(w, x)$ for a given w. Since $\hat{H}$ is the random function, the probability that there exists such an x is given by

$$\begin{aligned}\Pr\left[x \text{ s.t. } x = \hat{H}(w,x)\right] &= 1 - \left(1 - \frac{1}{2^n}\right)^{2^n} \\ &= 1 - \frac{1}{\exp(1)} \quad (\text{as } 2^n \to \infty) \\ &\approx \frac{64}{100}.\end{aligned}$$

Conversely, there does not exist such an x with a probability of approximately 36/100. Since this probability is based only on the randomness of $\hat{H}$, $d^{\hat{H}}$ cannot change it even if $d^{\hat{H}}$ has the unlimited computational resources. In the case that there is no such an x, whatever $d^{\hat{H}}$ may return as x_j, the distinguisher can obtain (x_j, y_j) such that $x_j \neq y_j$.

4 Other Collision-Resistant Compression Functions

In Sect. 3, we discussed the (in)differentiability of the Davies-Meyer compression function. In this section, we discuss the indifferentiability of other 11 SBL-1 compression functions that are secure in the sense of collision resistance [2]. As described below, we will find that the 11 compression functions are not better than the Davies-Meyer compression function in terms of the indifferentiability.

We classify the 12 SBL-1 compression functions including the Davies-Meyer compression function as the feedforward. Denoting by $\hat{e}$ the ideal block cipher in $\mathcal{E}_{n,n}$, we have the following six sets.

$$\begin{array}{ll}\text{Type-1:} & H_1^{\hat{e}}(w,x) = \hat{e}(w,x) \oplus x \\ \text{Type-2:} & H_2^{\hat{e}}(w,x) = \hat{e}(w,x) \oplus x \oplus w \\ \text{Type-3:} & H_3^{\hat{e}}(w,x) = \hat{e}(w,w \oplus x) \oplus x \\ \text{Type-4:} & H_4^{\hat{e}}(w,x) = \hat{e}(w,w \oplus x) \oplus x \oplus w \\ \text{Type-5:} & H_5^{\hat{e}}(w,x) = \hat{e}(w \oplus x,x) \oplus x \\ \text{Type-6:} & H_6^{\hat{e}}(w,x) = \hat{e}(w \oplus x,x) \oplus w\end{array}$$

For example, the type-1 set includes the Davies-Meyer compression function and the Matyas-Meyer-Oseas compression function. In the case of the Davies-Meyer compression function, w is a message block and x is an output of the previous compression function. In the case of the Matyas-Meyer-Oseas compression function, w is an output of the previous compression function and x is a message block. The type-2 set includes the Miyaguchi-Preneel compression function, which takes w as an output of the previous compression function and x as a message block.

The discussion in Sect. 3 does not depend on the use of parameters, i.e., which a parameter is a message block. Hence, all type-1 functions have the same security as the Davies-Meyer compression function in the sense of indifferentiability.

4.1 Only Encryption Oracle

Suppose that the distinguisher is allowed to have access to only $\overline{e}$. We consider the construction of a simulator $e^{\hat{H}_i}$ where $i \in \{2, 3, \ldots, 6\}$ that is similar to the simulator of Sect. 3.1. For a given query (ω, χ) to $e^{\hat{H}_i}$, the query (w, x) to $\hat{H}_i$ is uniquely determined. Like the simulator of Sect. 3.1, the output z_i is computed from $\hat{H}_i(w, x)$ and (ω, χ). Although the computation of z_i depends on $\hat{H}_i$, it can be done by the bitwise exclusive OR. When $e^{\hat{H}_i}(\omega, \cdot)$ is no longer the permutation for $\exists\omega$, $e^{\hat{H}_i}(\omega, \cdot)$ fails to emulate $\hat{e}(\omega, \cdot)$. Hence, the advantage of the distinguisher is given by Eq. (2).

4.2 Encryption and Decryption Oracles

Suppose that the distinguisher is allowed to have access to $\overline{e}$ and $\overline{d}$. We consider a distinguisher that is a similar to that of Sect. 3.2. Namely, the distinguisher first asks $(\omega, 0)$ to $\overline{d}$, and then asks the query (w, x), which is based on the response of $\overline{d}$, to $\overline{H}_i$. When the output of $\hat{e}$ is 0, the output of $H_i^{\hat{e}}$ is computed by a bitwise function of (w, x). On the other hand, the output of $\hat{H}$ is not probably given by the bitwise function of (w, x). Therefore, there exists the distinguisher that can achieve the advantage of Eq. (4).

5 Block-Cipher-Selection Compression Function

From results of [2][13] and results of previous sections, we conclude that all the 64 SBL-1 compression functions have the security drawback in terms of preimage resistance, second-preimage resistance, collision resistance, or indifferentiability. However, we propose a new SBL-1 compression function that is secure in the sense of indifferentiability.

5.1 Construction

Let $u = 2^n - 1$. Let $\hat{e}_i$ $(i = 0, 1, \dots, u)$ be ideal block ciphers in $\mathcal{E}_{n,n}$ where each $\hat{e}_i$ is selected independently of each other, i.e., $\hat{e}_i \overset{\$}{\leftarrow} \mathcal{E}_{n,n}$ for $\forall i \in \{0, 1, \dots, u\}$. We define a compression function $H^{\hat{e}_0, \hat{e}_1, \dots, e_u}$ as follows:

$$H^{\hat{e}_0,\hat{e}_1,\dots,e_u}(w, x) = \begin{cases} \hat{e}_0(w, x) & \text{if } x = 0, \\ \hat{e}_1(w, x) & \text{if } x = 1, \\ \dots \\ \hat{e}_u(w, x) & \text{if } x = u, \end{cases} \tag{5}$$

where $x \in \{0, 1\}^n$ is considered as the binary representation of an integer in $\{0, 1, \dots, u\}$. For simplification, we denote by $\{\hat{e}\}_u$ the set of $\hat{e}_0, \hat{e}_1, \dots,$ and $\hat{e}_u$. Using this notation, we rewrite Eq. (5) as

$$H^{\{\hat{e}\}_u}(w, x) = \hat{e}_x(w, x).$$

Since $H^{\{\hat{e}\}_u}$ is a function in $\mathcal{H}_{2n,n}$ and requires one invocation of $\hat{e}_i$ to produce an output for a given input, $H^{\{\hat{e}\}_u}$ is an SBL-1 compression function. We call this compression function a block-cipher-selection compression function (a BCS compression function).

5.2 Security Analysis

We now discuss the indifferentiability between the BCS compression function $H^{\{\hat{e}\}_u}$ and the random function $\hat{H}$. As described below, unlike the 64 SBL-1 compression functions, the BCS compression function is completely indifferentiable from the random function. That is, even if the distinguisher has infinitely computational resources, the probability that the distinguisher can distinguish the BCS compression function from the random function is equal to 0. Hence, the BCS compression is optimal in the sense of indifferentiability. This fact is due to two properties of the BCS compression function.

- For a fixed i, $\hat{e}_i(\cdot, i)$ is the random function in $\mathcal{H}_{n,n}$, not a permutation on $\{0, 1\}^n$.
- For a fixed i, almost all mappings $\hat{e}_i(w, x)$ are not used for the computation of $H^{\hat{e}_i}$, i.e., only mappings $\hat{e}_i(w, i)$ are used.

We assume that the distinguisher does not make the same query. Let (e, i, w, x) be a query to $\overline{e}_i$ for requiring the computation of $\overline{e}_i(w, x)$ where e is a special symbol to denote a used oracle. Similarly, let (d, i, w, y) be a query to $\overline{d}_i$ for requiring $\overline{d}_i(w, y)$.

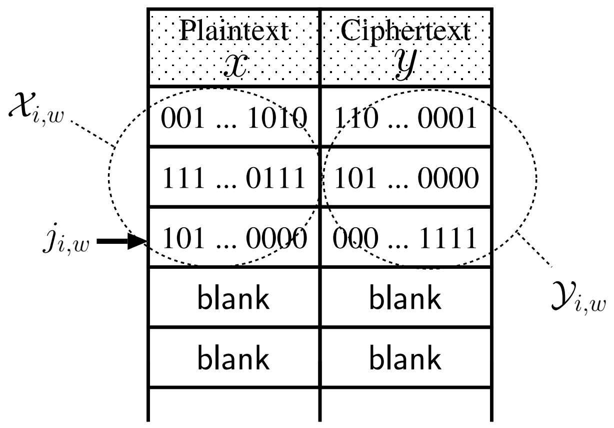

Figure 3: The plaintext-ciphertext table $e_i(w)$.

5.2.1 Only Encryption Oracle

Suppose that only $\{\overline{e}\}_u$ is available to the distinguisher. We discuss the indifferentiability between $H^{\{\hat{e}\}_u}$ and $\hat{H}$.

Theorem 1 *Suppose that only $\{\overline{e}\}_u$ is available to the distinguisher. The BCS compression function of Eq. (5) is $(\infty, t, q, 0)$ indifferentiable from the random function, where $t \leq \min(2^{2n}, q)$ and $q \leq 2^{2n}(2^n + 1)$. The space complexity of the simulator is $O(q)$.*

One must construct a simulator $\{e^{\hat{H}}\}_u$ such that interacting $(H^{\{\hat{e}\}_u}, \{\hat{e}\}_u)$ is indifferentiable from interacting $(\hat{H}, \{e^{\hat{H}}\}_u)$. Our simulator $\{e^{\hat{H}}\}_u$ is defined as follows. Our simulator has plaintext-ciphertext tables of $e_i^{\hat{H}}(w, \cdot)$ for $\forall i \in \{0, 1, \ldots, u\}$ and $\forall w \in \{0,1\}^n$ as shown in Fig. 3. Each table is denoted by $e_i(w)$. Let $\mathcal{X}_{i,w}$ be the set of plaintexts x appeared in $e_i(w)$, and let $\mathcal{Y}_{i,w}$ be the set of ciphertexts y appeared in $e_i(w)$.

step 1: Initially, $j_{i,w} \leftarrow 0$ for $\forall i \in \{0, 1, \ldots, u\}$ and $\forall w \in \{0,1\}^n$. All tables are initialized with a special symbol blank, which means that the entry is not yet used.

step 2: The simulator receives a query (e, i, w, x) from the distinguisher. $j_{i,w} \leftarrow j_{i,w}+1$ and $x_{j_{i,w}} \leftarrow x$. After $v \leftarrow \hat{H}(w, i)$, the simulator determines $y_{j_{i,w}}$ as follows:

$$\begin{cases} y_{j_{i,w}} \leftarrow v & \text{if } i = x, \\ y_{j_{i,w}} \overset{\$}{\leftarrow} \{0,1\}^n - \mathcal{Y}_{i,w} - \{v\} & \text{otherwise.} \end{cases}$$

The simulator replaces (blank, blank) in the table $e_i(w)$ with $(x_{j_{i,w}}, y_{j_{i,w}})$ and returns $y_{j_{i,w}}$.

Unlike the simulator of Sect. 3.1, $\{e^{\hat{H}}\}_u$ can perfectly emulate $\{\hat{e}\}_u$. In step 2, the simulator updates the table $e_i(w)$, keeping a one-to-one and random mapping. When x takes a string in $\{0,1\}^n$ for a fixed i and a fixed w, the distribution of $e_i^{\hat{H}}(w, x)$ is identical to that of $\hat{e}_i(w, x)$. Hence, we have $\mathbf{Adv}_H(q) = 0$, which is better than that of the Davies-Meyer compression function.

The time complexity of the simulator depends on the number of queries to $\hat{H}$. Keeping $\hat{H}(w, i)$, the simulator makes at most $\min(q, 2^{2n})$ queries to $\hat{H}$. The size of tables is $O(q)$. The distinguisher can make at most 2^{3n} queries to $\{\overline{e}\}_u$ and at most 2^{2n} queries to $\overline{H}$. Hence, the number q of queries is not larger than $2^{2n}(2^n + 1)$.

5.2.2 Encryption and Decryption Oracles

Suppose that both of $\{\overline{e}\}_u$ and $\{\overline{d}\}_u$ are available to the distinguisher. We now discuss the indifferentiability between $H^{\{\hat{e}\}_u}$ and $\hat{H}$.

Theorem 2 *Suppose that both of $\{\overline{e}\}_u$ and $\{\overline{d}\}_u$ are available to the distinguisher. The BCS compression function of Eq. (5) is $(\infty, t, q, 0)$ indifferentiable from $\hat{H}$ where $t \leq \min(2^{2n}, q)$ and $q \leq 2^{2n}(2^{n+1}+1)$. The space complexity of the simulator is $O(q)$.*

One must construct a simulator $(\{e^{\hat{H}}\}_u, \{d^{\hat{H}}\}_u)$ such that interacting $(H^{\{\hat{e}\}_u}, (\{\hat{e}\}_u, \{\hat{d}\}_u))$ is indifferentiable from interacting $(\hat{H}, (\{e^{\hat{H}}\}_u, \{d^{\hat{H}}\}_u))$. Our simulator $(\{e^{\hat{H}}\}_u, \{d^{\hat{H}}\}_u)$ is defined as follows. Our simulator has plaintext-ciphertext tables of $e_i^{\hat{H}}(w, \cdot)$ for $\forall i \in \{0, 1, \ldots, u\}$ and $\forall w \in \{0,1\}^n$, as well as the simulator of Sect. 5.2.1 (Fig. 3). Suppose that the table $e_i(w)$ is shared by $e_i^{\hat{H}}$ and $d_i^{\hat{H}}$.

step 1: Initially, $j_{i,w} \leftarrow 0$ for $\forall i \in \{0, 1, \ldots, u\}$ and $\forall w \in \{0,1\}^n$. All tables are initialized with blank.

step 2: When the simulator received a query (e, i, w, x), the simulator checks whether $x \in \mathcal{X}_{i,w}$ or not.

step 2.1: If x is equal to $x_\iota \in \mathcal{X}_{i,w}$, then the simulator returns y_ι.

step 2.2: If x does not exist in $\mathcal{X}_{i,w}$, then $j_{i,w} \leftarrow j_{i,w} + 1$ and $x_{j_{i,w}} \leftarrow x$. After $v \leftarrow \hat{H}(w, i)$, the simulator determines y_{j_i} as follows:

$$\begin{cases} y_{j_{i,w}} \leftarrow v & \text{if } i = x, \\ y_{j_{i,w}} \overset{\$}{\leftarrow} \{0,1\}^n - \mathcal{Y}_{i,w} - \{v\} & \text{otherwise.} \end{cases}$$

The simulator replaces $(\mathsf{blank}, \mathsf{blank})$ in the table $e_i(w)$ with $(x_{j_{i,w}}, y_{j_{i,w}})$ and returns $y_{j_{i,w}}$.

step 3: When the simulator received a query (d, i, w, y), the simulator checks whether $y \in \mathcal{Y}_{i,w}$ or not.

step 3.1: If y is equal to $y_\iota \in \mathcal{Y}_{i,w}$, then the simulator returns x_ι.

step 3.2: If y does not exist in $\mathcal{Y}_{i,w}$, then $j_{i,w} \leftarrow j_{i,w} + 1$ and $y_{j_{i,w}} \leftarrow y$. After $v \leftarrow \hat{H}(w, i)$, the simulator determines x_{j_i} as follows:

$$\begin{cases} x_{j_{i,w}} \leftarrow i & \text{if } y = v, \\ x_{j_{i,w}} \overset{\$}{\leftarrow} \{0,1\}^n - \mathcal{X}_{i,w} - \{i\} & \text{otherwise.} \end{cases}$$

The simulator replaces $(\mathsf{blank}, \mathsf{blank})$ in the table $e_i(w)$ with $(x_{j_{i,w}}, y_{j_{i,w}})$ and returns $x_{j_{i,w}}$.

This simulator can perfectly emulate $(\{\hat{e}\}_u, \{\hat{d}\}_u)$. For $\forall e_i(w)$, $x \in \mathcal{X}_{i,w}$ corresponds to unique $y \in \mathcal{Y}_{i,w}$, i.e., there is no $x_\iota, x_\lambda \in \mathcal{X}_{i,w}$ such that $e_i^{\hat{H}}(w, x_\iota) = e_i^{\hat{H}}(w, x_\lambda)$ and there is no $y_\iota, y_\lambda \in \mathcal{Y}_{i,w}$ such that $d_i^{\hat{H}}(w, y_\iota) = d_i^{\hat{H}}(w, y_\lambda)$. Furthermore, y_{j_i} in step 2.2 (or x_{j_i} in step 3.2) is randomly selected from $\{0,1\}^n$ avoiding the collision. This behavior is identical to that of the ideal block cipher $\hat{e}_i(w, \cdot)$. Hence, we have $\mathbf{Adv}_H(q) = 0$, which is substantially better than that of the Davies-Meyer compression function.

The time complexity of the simulator depends on the number of queries to $\hat{H}$. Keeping $\hat{H}(w, i)$, the simulator makes at most $\min(q, 2^{2n})$ queries to $\hat{H}$. The size of tables is $O(q)$. The distinguisher can make at most 2^{3n+1} queries to $(\{\overline{e}\}_u, \{\overline{d}\}_u)$ and at most 2^{2n} queries to $\overline{H}$. Hence, the number q of queries is not larger than $2^{2n}(2^{n+1}+1)$.

5.3 Implementation

When the compression function based on the block cipher is implemented, the ideal block cipher is probably replaced with an actual block cipher such as AES [11]. Since usual SBL-1 compression functions require only one block cipher, its implementation is easy. In contrast, it seems impractical to implement the BCS compression function because it requires 2^n actual block ciphers.

However, we have a possible way to implement the BCS compression. For example, the 256-bit key is available to AES, i.e., $\mathrm{AES}^{256} : \{0,1\}^{256} \times \{0,1\}^{128} \to \{0,1\}^{128}$. For $w, x \in \{0,1\}^{128}$, $e_i(w, x)$ is implemented as $\mathrm{AES}^{256}(i \parallel w, x)$ where $\parallel$ denotes the concatenation operator on strings. Using this way, we can obtain 2^{128} block ciphers in $\mathcal{E}_{128,128}$. Since a long-length key is usually available to actual block ciphers, this way is applicable to many actual block ciphers. We notice that the discussion is separately needed to make a formal determination whether the set of block ciphers obtained by this way can be considered as the set of ideal block ciphers.

6 Concluding Remarks

In this paper, we have shown that the 12 single-block-length and rate-1 (SBL-1) compression functions, which are secure in the sense of collision resistance, are not secure in the sense of indifferentiability. Notice that the Davies-Meyer compression function is one of the 12 SBL-1 compression functions. From results of [2][13] and those of this paper, we concluded that all the 64 SBL-1 compression functions have the security drawback in terms of preimage resistance, second-preimage resistance, collision resistance, or indifferentiability.

We have proposed the block-cipher-selection (BCS) compression function, which is secure in terms of indifferentiability. Although the BCS compression function is formally the SBL-1 compression function, the negative result above is inapplicable to the BCS compression function. Unlike the 64 SBL-1 compression functions, the BCS compression function selectively uses one block cipher among many block ciphers. We have proved that if the simulator has reasonable computational resources, then the BCS compression function can perfectly emulate the random function, that is, the BCS compression function is optimal in the sense of indifferentiability. We have also proposed the implementation of the BCS compression function using actual block ciphers. However, the security of the implementation must be analyzed in future.

References

[1] J. Black, M. Cochran, and T. Shrimpton, "On the impossibility of highly-efficient blockcipher-based hash functions," Advances in Cryptology - EUROCRYPT 2005, Lecture Notes in Computer Science, vol. 3494, pp. 526–541, 2005.

[2] J. Black, P. Rogaway, and T. Shrimpton, "Black-box analysis of the block-cipher-based hash-function constructions from PGV," Advances in Cryptology - CRYPTO 2002, Lecture Notes in Computer Science, vol. 2442, pp. 320–335, 2002.

[3] J. S. Coron, Y. Dodis, C. Malinaud, and P. Puniya, "Merkle-Damgård revisited: How to construct a hash function," Advances in Cryptology - CRYPTO 2005, Lecture Notes in Computer Science, vol. 3621, pp. 430–448, 2005.

[4] I. B. Damgård, "Collision free hash functions and public key signature schemas," Advances in Cryptology - EUROCRYPT '87, Lecture Notes in Computer Science, vol. 304, pp. 203–216, 1988.

[5] H. Dobbertin, "Cryptanalysis of MD4," Fast Software Encryption, Lecture Notes in Computer Science, vol. 1039, pp. 53–69, 1996.

[6] S. Hirose, "How to construct double-block-length hash functions," The Second Cryptographic Hash Workshop, August 2006. `http://www.csrc.nist.gov/pki/HashWorkshop/2006/Papers/HIROSE_article.pdf`.

[7] S. Lucks, "A failure-friendly design principle for hash functions," Advances in Cryptology - ASIACRYPT 2005, Lecture Notes in Computer Science, vol. 3788, pp. 474–494, 2005.

[8] U. Maurer, R. Renner, and C. Holenstein, "Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology," First Theory of Cryptography Conference, TCC 2004, Lecture Notes in Computer Science, vol. 2951, pp. 21–39, 2004.

[9] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, HANDBOOK of APPLIED CRYPTOGRAPHY, CRC Press, 1996.

[10] R. C. Merkle, "One way hash functions and DES," Advances in Cryptology - CRYPTO '89, Lecture Notes in Computer Science, vol. 435, pp. 428–446, 1990.

[11] National Institute of Standards and Technology, "Advanced encryption standard (AES)," Federal Information Processing Standards Publication 197, 2001. `http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf`.

[12] National Institute of Standards and Technology, "Secure hash standard," Federal Information Processing Standards Publication 180-2, August 2002. `http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf`.

[13] B. Preneel, R. Govaerts, and J. Vandewalle, "Hash functions based on block ciphers: a synthetic approach," Advances in Cryptology - CRYPTO '93, Lecture Notes in Computer Science, vol. 773, pp. 368–378, 1993.

[14] P. Sarkar and P. J. Schellenberg, "A parallel algorithm for extending cryptographic hash functions," Progress in Cryptology – INDOCRYPT 2001, Lecture Notes in Computer Science, vol. 2247, pp. 40–49, 2001.

[15] X. Wang, X. Lai, D. Feng, H. Chen, and X. Yu, "Cryptanalysis of the hash functions MD4 and RIPEMD," Advances in Cryptology - EUROCRYPT 2005, Lecture Notes in Computer Science, vol. 3494, pp. 1–18, 2005.

[16] X. Wang and H. Yu, "How to break MD5 and other hash functions," Advances in Cryptology - EUROCRYPT 2005, Lecture Notes in Computer Science, vol. 3494, pp. 19–35, 2005.