Blockcipher-based Double-length Hash Functions for Pseudorandom Oracles

Yusuke Naito

Mitsubishi Electoric Corporation

Abstract. The notion of PRO (pseudorandom oracle) is an important security notion of hash functions because a PRO hash function inherits all properties of a random oracle up to the PRO bound (e.g., security against generic attacks, collision resistant security, preimage resistant security and so on). In this paper, we propose a new block cipher-based double-length hash function for PROs. Our hash function uses a single block cipher, which encrypts an n-bit string using a $2n$-bit key, and maps an input of arbitrary length to a $2n$-bit output. Since many block ciphers supports a $2n$-bit key (e.g. AES supports a 256-bit key), the assumption to use the $2n$-bit key length block cipher is acceptable. We prove that our hash function is PRO up to $\mathcal{O}(2^n)$ query complexity as long as the block cipher is an ideal cipher. To our knowledge, this is the first time double-length hash function based on a single (practical size) block cipher with the birthday type PRO security.

Keywords. Double-length hash function, pseudorandom oracle, ideal cipher model.

1 Introduction

The block cipher-based design method (e.g. [18, 20, 25]) is the most popular method for constructing a cryptographic hash function. A hash function is designed by the following two steps: (1) designing a block cipher and (2) designing a mode of operation. MD-family [26, 27], SHA-family [22] and SHA-3 candidates follow the design method. Another design method is to utilize a practical block cipher such as AES. Such hash functions are useful when used on size restricted devices such as RFID tags or smart cards: when implementing both a hash function and a block cipher, one has only to implement a block cipher. However, the output length of most practical encryption functions is far too short for a collision resistant hash function, e.g., 128-bits for AES. Thus the design of CR-DLHFs (collision resistant double length hash functions) is an interesting topic. The core of the design of CR-DLHFs is to design a DLCF (double-length compression functions) which maps an input of fixed length (more than $2n$-bits) to an output of $2n$-bit length when using an n-bit output length block cipher. When designing a DLCF, the security is proven in the ideal cipher model [8, 13, 16, 10, 23], e.g., Hirose's compression function, Tandem-DM, Abreast-DM and generalized DLCF ([13, 15, 23]). Then hash functions combined a domain extension (e.g. strengthened Merkle-Damgård (SMD) [5, 21]), which preserves CR security, with these compression functions yield CR-DLHFs in the ideal cipher model.

The notion of PRO (Pseudorandom Oracle), which is one application of indifferentiability introduced by Maurer et al. [19], is an important security notion for hash functions. Therefore, when designing a DLHF, it is important to consider the PRO security. Roughly speaking, if a hash function is PRO, we can use it as a random oracle. Namely, such hash functions inherit all properties of a random oracle up to the PRO security bound. Therefore, the PRO security guarantees security against generic attacks, e.g. the length extension attack, which is a requirement for SHA-3 candidates [11], and if a hash function has the birthday security for PRO, it is CR because a random oracle is.

Hereafter a block cipher which encrypts an n-bit string using a k-bit key is denoted by (k,n)-BC. Gong et al. [12] proved that the prefix-free Merkle-Damgård using the PBGV compression function [24] is PRO up to $\mathcal{O}(2^{n/2})$ query complexity as long as the ($2n$,n)-BC is an ideal cipher. When $n = 128$, the PRO security is not enough because the query complexity is $\mathcal{O}(2^{64})$. Chang et al. [2] and Hirose

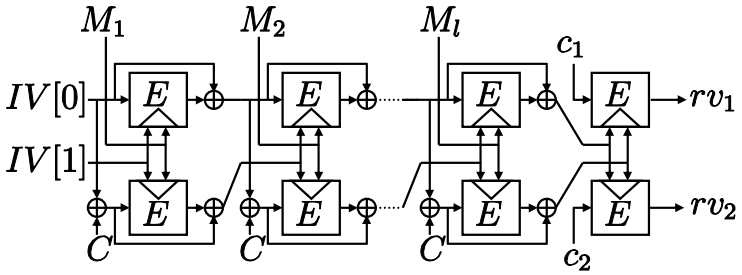

Fig. 1. Our DLHF using Hirose's compression function

et al. [14] proposed $2n$-bit output length DLHFs using a compression function $h : \{0,1\}^d \to \{0,1\}^n$ where $d > 2n$. Their proposals are PROs up to $\mathcal{O}(2^n)$ query complexity as long as h is a random oracle. Since an ideal cipher where the plain text element is fixed by a constant is a random oracle, these hash functions can be modified to block cipher-based schemes which use a (d,n)-BC. However, practical block ciphers (such as AES) don't support d-bit key where $d > 2n$. So we rise the following question:

Can we construct a DLHF from a "practical size"[1] block cipher with "birthday security" for PROs?

In this paper, we propose a DLHF using a single ($2n$,n)-BC with the $\mathcal{O}(2^n)$ PRO security[2] in the ideal cipher model. Since many block ciphers support $2n$-bit key length, e.g. AES supports 256-bit key length, and the existing DLCFs (e.g. Hirose's compression function, Tandem-DM, Abreast-DM, and generalized DLCF) use a ($2n$,n)-BC, the assumption to use a ($2n$,n)-BC is acceptable. To our knowledge, our hash function is the first time DLHF based on a practical size block cipher with the birthday-type PRO security. Moreover, since our hash function uses only a *single* block cipher, it can be utilized in size restricted devices.

Our DLHF. Our construction, which uses Hirose's compression function, Tandem-DM or Abreast-DM, iterates the compression function and uses a new finalization function f at the last iteration which calls a $(2n, n)$-BC twice. Therefore, the complexity of our hash function is the number of $(2n, n)$-BC calls of the SMD hash function plus two $(2n, n)$-BC calls. Let $\mathsf{BC}_{2n,n} = (E, D)$ be a ($2n$,n)-BC where E is an encryption function and D is a decryption function. Let $\mathrm{DLCF}^{\mathsf{BC}_{2n,n}}$ be a double-length compression function which is Hirose's compression function, Tandem-DM, or Abreast-DM. Let $\mathrm{SMD}^{\mathrm{DLCF}^{\mathsf{BC}_{2n,n}}} : \{0,1\}^* \to \{0,1\}^{2n}$ be the SMD hash function using the compression function $\mathrm{DLCF}^{\mathsf{BC}_{2n,n}}$. Our DLHF is defined as follows:

$$F^{\mathsf{BC}_{2n,n}}(M) = f^{\mathsf{BC}_{2n,n}}(\mathrm{SMD}^{\mathrm{DLCF}^{\mathsf{BC}_{2n,n}}}(M))$$

where $f^{\mathsf{BC}_{2n,n}}(x) = E(x, c_1)||E(x, c_2)$ and c_1 and c_2 are n-bit constant values. Note that the first element of the encryption function is the key element and the second element is the plain text element. The DLHF using Hirose's compression function is illustrated in Fig. 1 where $rv_1||rv_2$ is the output of the hash function. Note that in this figure we omit the suffix free padding function. We use the DLHF $\mathrm{SMD}^{\mathrm{DLCF}^{\mathsf{BC}_{2n,n}}}$ to compress an arbitrary length input into an fixed input length value. Since SMD hash functions are not PROs [4], to guarantee the PRO property, the final function $f^{\mathsf{BC}_{2n,n}}$ is

[1] "Practical size" is the size supported by practical block ciphers.

[2] The $\mathcal{O}(2^n)$ PRO security means that the number of queries is at least $\mathcal{O}(2^n)$ to differentiate the DLHF from a random oracle with probability of 1/2.

added to $\mathrm{SMD}^{\mathrm{DLCF}^{\mathsf{BC}_{2n,n}}}$. The use of constant values c_1 and c_2 of the final function is inspired by the design technique of EMD proposed by Bellare and Ristenpart [1]. This realizes two facts: (1) our hash function uses only a single block cipher and (2) the final function behaves like a random oracle. So we can treat our hash function as a NMAC hash function. Thus we can use the PrA design framework [6] when proving the PRO security of our hash function.

Proof Methodology. In the traditional PRO proof technique (e.g. [4, 1]), one has to consider the difference between a random oracle and $F^{\mathsf{BC}_{2n,n}}$ at the same time. Since a random oracle is a monolithic function while a hash function is not, there is a big structural gap between them. Therefore, the PRO proofs are very complex. Instead, we prove the PRO security of $F^{\mathsf{BC}_{2n,n}}$ by using three techniques to simplify the PRO proof: the PrA (Preimage Aware) design framework of Dodis et al. [6], PRO for a small function [4], and *indifferentiability from a hash function*. The first two techniques are existing techniques and the last technique is a new application of the indifferentiability framework [19]. Using the first two techniques, we construct a block cipher-based DLHF using two block ciphers. Then using indifferentiability from a hash function, we reduce the number of underlying block ciphers. We prove the PRO security by the following three steps.

- **Step 1.** Let $\mathsf{BC}^1_{2n,n} = (E1, D1)$ be a $(2n,n)$-BC. In this step, we prove that Hirose's compression function, Tandem-DM, and Abreast-DM are PrA up to $\mathcal{O}(2^n)$ query complexity as long as the block cipher is an ideal cipher. Then the PrA design framework guarantees that the following NMAC hash function is PRO up to $\mathcal{O}(2^n)$ query complexity as long as the block cipher is an ideal cipher and $g : \{0,1\}^{2n} \to \{0,1\}^{2n}$ is a random oracle.

$$F_1^{\mathsf{BC}^1_{2n,n}}(M) = g(\mathrm{SMD}^{\mathrm{DLCF}^{\mathsf{BC}^1_{2n,n}}}(M))$$

- **Step 2.** Let $\mathsf{BC}^2_{2n,n} = (E2, D2)$ and $\mathsf{BC}^3_{2n,n} = (E3, D3)$ be different $(2n,n)$-BCs. In this step, we prove that the function $f^{\mathsf{BC}^3_{2n,n}}$ is PRO up to $\mathcal{O}(2^n)$ query complexity as long as the block cipher is an ideal cipher where c_1 and c_2 are n-bit different values. Therefore, $f^{\mathsf{BC}^3_{2n,n}}$ can be used instead of g and thus the following hash function is PRO up to $\mathcal{O}(2^n)$ query complexity as long as the block ciphers are ideal ciphers.

$$F_2^{\mathsf{BC}^2_{2n,n},\mathsf{BC}^3_{2n,n}}(M) = f^{\mathsf{BC}^3_{2n,n}}(\mathrm{SMD}^{\mathrm{DLCF}^{\mathsf{BC}^2_{2n,n}}}(M))$$

- **Step 3.** This is the final step. In this step, we use indifferentiability from a hash function. So we prove that $F^{\mathsf{BC}_{2n,n}}$ is indifferentiable from $F_2^{\mathsf{BC}^2_{2n,n},\mathsf{BC}^3_{2n,n}}$ up to $\mathcal{O}(2^n)$ query complexity as long as the block ciphers are ideal ciphers. Therefore, $F^{\mathsf{BC}_{2n,n}}$ can be used instead of $F_2^{\mathsf{BC}^2_{2n,n},\mathsf{BC}^3_{2n,n}}$, namely, $F^{\mathsf{BC}_{2n,n}}$ is PRO up to $\mathcal{O}(2^n)$ query complexity as long as the block cipher is an ideal cipher.

By using our proof, one does not need to consider the "big" gap between the hash function and a random oracle from the following reasons. Therefore, our proof is easier than the traditional proofs.

- Step 1: PrA is the weaker notion than PRO and the DLCFs are the simpler and smaller functions than $F^{\mathsf{BC}_{2n,n}}$. Therefore, the PrA proof for the DLCFs is easier than the traditional PRO proof for $F^{\mathsf{BC}_{2n,n}}$.
- Step 2: In this step, we have only to consider the structure of $f^{\mathsf{BC}^3_{2n,n}}$. Since $f^{\mathsf{BC}^3_{2n,n}}$ is the simpler and smaller function than $F^{\mathsf{BC}_{2n,n}}$, the proof in Step 2 is easier than the traditional PRO security proof for $F^{\mathsf{BC}_{2n,n}}$.
- Step 3: The difference between $F^{\mathsf{BC}_{2n,n}}$ and $F_2^{\mathsf{BC}^2_{2n,n},\mathsf{BC}^3_{2n,n}}$ is the number of block ciphers. Namely, $F^{\mathsf{BC}_{2n,n}} = F_2^{\mathsf{BC}_{2n,n},\mathsf{BC}_{2n,n}}$ and the structure of $F^{\mathsf{BC}_{2n,n}}$ is the same as that of $F_2^{\mathsf{BC}^2_{2n,n},\mathsf{BC}^3_{2n,n}}$ except for the number of a block cipher. On the other hand, since a random oracle is a monolithic random function while $F^{\mathsf{BC}_{2n,n}}$ is not, the structure of $F^{\mathsf{BC}_{2n,n}}$is dissimilar from that of a random oracle. Therefore, the proof in this step is easier than the traditional PRO security proof for $F^{\mathsf{BC}_{2n,n}}$.

In our proof, indifferentiability from a hash function is used to reduce the number of underlying block ciphers and contributes to simplifying our proof. It may be useful for reducing the number of primitives and simplifying a proof. We discuss the application of indifferentiability from a hash function to other hash functions in Section 4.

Related Works. The PRO security proof of the EMD hash function using a random oracle compression function [1] uses a similar technique to indifferentiability from a hash function. They proved that (1) the NMAC hash function using random oracle compression functions is PRO and (2) the difference in the behaviors of adversaries A and B is negligible where A interacts with the NMAC hash function and B interacts with the EMD hash function. By combining the first step with the second step, they concluded that the EMD hash function is PRO. The second step of the proof is similar to indifferentiability from a hash function. However A is different from B and the interfaces of these adversaries are different. On the other hand, when we consider that H_2 is indifferentiable from H_1, an adversary of H_2 is the same as that of H_1 and the interface of the H_2 adversary is the same as that of the H_1 adversary due to the indifferentiability framework [19]. Therefore, proofs using indifferentiability from a hash function are easier than those using the technique of [1].

2 Preliminaris

Notation. For two values x, y, $x||y$ is the concatenated value of x and y. $x \leftarrow y$ means assigning y to x. $\oplus$ is bitwise exclusive or. $|x|$ is the bit length of x. For a set (list) $\mathcal{T}$ and an element W, $\mathcal{T} \leftarrow W$ means to insert W into $\mathcal{T}$ and $\mathcal{T} \xleftarrow{\cup} W$ means $\mathcal{T} \leftarrow \mathcal{T} \cup \{W\}$. For some d-bit value x, $x[0]$ is the first n bit value and $x[1]$ is the last $d-n$-bit value. $\mathsf{BC}_{d,n} = (E, D)$ be a block cipher where $E : \{0,1\}^d \times \{0,1\}^n \rightarrow \{0,1\}^n$ is an encryption function, $D : \{0,1\}^d \times \{0,1\}^n \rightarrow \{0,1\}^n$ is a decryption function, the key size is d bits and the cipher text size is n bits. $\mathcal{C}_{d,n} = (E_I, D_I)$ be a ideal cipher where $E_I : \{0,1\}^d \times \{0,1\}^n \rightarrow \{0,1\}^n$ is an encryption oracle, $D_I : \{0,1\}^d \times \{0,1\}^n \rightarrow \{0,1\}^n$ is a decryption oracle, the key size is d-bits and the cipher text size is n-bits. $\mathcal{F}_{m,n} : \{0,1\}^m \rightarrow \{0,1\}^n$ is a random oracle. An arbitrary input length random oracle is denoted by $\mathcal{F}_n : \{0,1\}^* \rightarrow \{0,1\}^n$. For any algorithm A, we write $\mathsf{Time}(A)$ to mean the sum of its description length and the worst-case number of steps.

Merkle-Damgård. Let $h : \{0,1\}^{2n} \times \{0,1\}^d \rightarrow \{0,1\}^{2n}$ be a compression function using a primitive P (more strictly h^P) and $\mathsf{pad} : \{0,1\}^* \rightarrow (\{0,1\}^d)^*$ be a padding function. Merkle-Damgård hash function MD^h is described as follows where IV is a $2n$-bit initial value.

$\underline{\mathrm{MD}^h(M)}$
$z \leftarrow IV$;
Break $\mathsf{pad}(M)$ into d-bit blocks, $\mathsf{pad}(N) = M_1||\cdots||M_l$;
for $i = 1, \ldots, l$ do $z \leftarrow h(z, M_i)$;
Ret z;

We denote MD^h, when padding pad is a suffix-free padding sfpad, by SMD^h, called strengthened Merkle-Damgård. We assume that it is easy to strip padding, namely that there exists an efficiently computable function $\mathsf{unpad} : (\{0,1\}^d)^* \rightarrow \{0,1\}^* \cup \{\perp\}$ such that $x = \mathsf{unpad}(\mathsf{pad}(x))$ for all $x \in \{0,1\}^*$. Inputs to unpad that are not valid outputs of pad are mapped to $\perp$ by unpad.

Indifferentiability. Let H_1^P be some cryptographic scheme (e.g. a hash function) that utilizes an ideal primitive P. Let H_2^Q be another cryptographic scheme that utilizes an ideal primitive Q. We say

that H_1^P is indifferentiable from H_2^Q if there exists an effcient simulator S that simulates P such that for any distinguisher A outputting a bit it is the case that

$$\mathsf{Adv}^{\mathsf{indif}}_{H^P,H^Q,S}(A) = Pr[A^{H_1^P,P} \Rightarrow 1] - Pr[A^{H_2^Q,S} \Rightarrow 1]$$

is small where the probabilities are taken over the coins used the experiments. S can make queries to Q. The S's task is to simulate P such that relations among responses of (H_1^P, P) hold in responses of (H_2^Q, S) as well. If the bound is small, we can use H_1^P instead of H_2^Q up to the indifferentiability bound. Namely, if cryptosystem $\mathcal{X}$ using H_2^Q is secure, $\mathcal{X}$ using H_1^P is also secure.

When considering the PRO (Pseudorandom Oracle) security for a hash function $H_1^P : \{0,1\}^s \to \{0,1\}^t$, we replace the above primitive H_2^Q with a random oracle $\mathcal{F}_{s,t}$. That is, $H_2^Q = Q = \mathcal{F}_{s,t}$. We write the pro-advantage by $\mathsf{Adv}^{\mathsf{pro}}_{H_1^P,S} = \mathsf{Adv}^{\mathsf{indif}}_{H_1^P,\mathcal{F}_{s,t},S}$. Thus

$$\mathsf{Adv}^{\mathsf{pro}}_{H_1^P,S}(A) = Pr[A^{H_1^P,P} \Rightarrow 1] - Pr[A^{\mathcal{F}_{s,t},S} \Rightarrow 1]$$

where S can makes queries to $\mathcal{F}_{s,t}$.

Preimage Awareness. The notion of preimage awareness is useful for PRO security proofs of NMAC hash functions. We only explain the definition of preimage awareness. Please see Section 3 of [7] for the spirit of the notion. Let F^P be a hash function using an ideal primitive P. The preimage awareness of F^P is estimated by the following experiment.

$\underline{\mathsf{Exp}^{\mathsf{pra}}_{F^P,P,\mathcal{E},A}}$
$x \xleftarrow{\$} A^{\mathsf{P},\mathsf{Ex}}$;
$z \leftarrow F^P(x)$;
Ret $(x \neq \mathsf{V}[z] \wedge \mathsf{Q}[z] = 1)$;

$\underline{\textbf{oracle } \mathsf{P}(m)}$
$c \leftarrow P(m)$;
$\alpha \xleftarrow{\cup} (m, c)$;
Ret c;

$\underline{\textbf{oracle } \mathsf{Ex}(z)}$
$\mathsf{Q}[z] \leftarrow 1$;
$\mathsf{V}[z] \leftarrow \mathcal{E}(z, \alpha)$;
Ret $\mathsf{V}[z]$;

Here an adversary A is provided two oracles P and Ex. The oracle P provides access to the ideal primitive P and records a query histry α. The extraction oracle Ex provides an interface to an extractor $\mathcal{E}$, which is a deterministic algorithm that uses z and the query history α of P, and returns either $\perp$ or an element x' such that $F^P(x') = z$. If x' can be constructed from α, it returns x' and otherwise returns $\perp$. In this experiment, the (initially everywhere $\perp$) array Q and the (initially empty) array V are used. When z is queried to Ex, $\mathsf{Q}[z] \leftarrow 1$ and then the output of $\mathcal{E}(z, \alpha)$ is assigned to $\mathsf{V}[z]$. For the hash function F^P, the adversary A, and the extractor $\mathcal{E}$, we define the advantage relation

$$\mathsf{Adv}^{\mathsf{pra}}_{F^P,P,\mathcal{E}} = Pr[\mathsf{Exp}^{\mathsf{pra}}_{F^P,P,\mathcal{E},A} \Rightarrow \mathsf{true}]$$

where the probabilities are over the coins used in running the experiments. When there exists an efficient extractor $\mathcal{E}$ such that for any adversary A the above advantage is small, we say that F^P is preimage aware (PrA).

The pra-advantage can be evaluated from the cr-advantage (collision resistance advantage) and the 1-wpra (1-weak PrA) advantage [7]. The 1-WPrA experiment is described as follows.

$\underline{\mathsf{Exp}^{\mathsf{1wpra}}_{F^P,P,\mathcal{E}^+,A}}$
$x \xleftarrow{\$} A^{\mathsf{P},\mathsf{Ex}^+}$;
$z \leftarrow F^P(x)$;
Ret $(x \notin L \wedge \mathsf{Q}[z] = 1)$;

$\underline{\textbf{oracle } \mathsf{P}(m)}$
$c \leftarrow P(m)$;
$\alpha \xleftarrow{\cup} (m, c)$;
Ret c;

$\underline{\textbf{oracle } \mathsf{Ex}^+(z)}$
$\mathsf{Q}[z] \leftarrow 1$;
$L \leftarrow \mathcal{E}^+(z, \alpha)$;
Ret L;

The difference between the 1-WPrA experiment and the PrA experiment is the extraction oracle. In the 1-WPrA experiment, a multi-point extractor oracle Ex^+ is used. Ex^+ provides an interface to a multi-point extractor $\mathcal{E}^+$, which is a deterministic algorithm thatuses z and α, and returns either $\perp$ or a set of an element in the domain of F^P. The output (set) of $\mathcal{E}^+$ is stored in list L. Thus, if $L \neq \{\perp\}$,

for any $x' \in L$ $F^P(x') = z$. In this experiment, an adversary A can make only a single query to Ex^+. For a hash function F^P, an adversary A, and a multi-point extractor $\mathcal{E}^+$, we define the advantage relation

$$\mathsf{Adv}^{\mathsf{1wpra}}_{F^P,P,\mathcal{E}} = Pr[\mathsf{Exp}^{\mathsf{1wpra}}_{F^P,P,\mathcal{E}^+,A} \Rightarrow \mathsf{true}]$$

where the probabilities are over the coins used in running the experiments. When there exists an efficient multi-point extractor $\mathcal{E}^+$ such that the above advantage is small for any adversary A, we say that F^P is 1-WPrA.

The definition of the cr-advantage as follows. Let A be an adversary that outputs a pair of values x and x'. To hash function F^P using primitive P and adversary A we associate the advantage relation

$$\mathsf{Adv}^{\mathsf{cr}}_{F^P,P}(A) = Pr[(x,x') \xleftarrow{\$} A^P : F^P(x) = F^P(x') \wedge x \neq x']$$

where the probability is over the coins used by A and primitive P.

Then the pra-advantage can be evaluated as follows.

Lemma 1 (Lemmas 3.3 and 3.4 of [7]). *Let $\mathcal{E}^+$ be an arbitrary multi-point extractor. There exists an extractor $\mathcal{E}$ such that for any pra-advarsary A^{pra} making q_e extraction queries and q_P primitive queries there exists 1-wpra adversary A^{1wpra} and cr-adversary A^{cr} such that*

$$\mathsf{Adv}^{\mathsf{pra}}_{F^P,P,\mathcal{E}}(A^{\mathsf{pra}}) \leq q_e \cdot \mathsf{Adv}^{\mathsf{1wpra}}_{F^P,P,\mathcal{E}^+}(A^{\mathsf{1wpra}}) + \mathsf{Adv}^{\mathsf{cr}}_{F^P,P}(A^{\mathsf{cr}}).$$

A^{1wpra} runs in time at most $\mathcal{O}(q_e\mathsf{Time}(\mathcal{E}^+))$ and makes the same number of P queries as A^{pra}. A^{cr} asks q_P queries and run in time $\mathcal{O}(q_e \cdot \mathsf{Time}(\mathcal{E}^+))$. $\mathcal{E}$ runs in the same time as $\mathcal{E}^+$. ♦

NMAC Hash Function. Let $g : \{0,1\}^n \to \{0,1\}^n$ be a function and $H^P : \{0,1\}^* \to \{0,1\}^n$ be a hash function using primitive P such that g is not used in H^P. The NMAC hash function is defined as follows.

$$\mathsf{NMAC}^{H^P,g}(M) = g(H^P(M)).$$

Dodis et al. [7] proved that the PRO security of $\mathsf{NMAC}^{H^P,g}$ can be reduced into the PrA security of H^P.

Lemma 2 (Theorem 4.1 of [7]). *Let P be an ideal primitive, g be a random oracle and $\mathcal{E}$ be any extractor for H^P. Then there exists a simulator $S = (S_P, S_g)$ such that for any PRO adversary A making at most q_F, q_P, q_g queries to its three oracles $(\mathcal{O}_F, \mathcal{O}_P, \mathcal{O}_g)$ where $(\mathcal{O}_F, \mathcal{O}_P, \mathcal{O}_g) = (\mathsf{NMAC}^{H^P,g}, P, g)$ or $(\mathcal{O}_F, \mathcal{O}_P, \mathcal{O}_g) = (\mathcal{F}_n, S_P, S_g)$, there exists a PrA adversary B such that*

$$\mathsf{Adv}^{\mathsf{pro}}_{\mathsf{NMAC}^{H^P,g},S}(A) \leq \mathsf{Adv}^{\mathsf{pra}}_{H^P,P,\mathcal{E}}(B).$$

S runs in time $\mathcal{O}(q_P + q_g \cdot \mathsf{Time}(\mathcal{E}))$. Let l be the length of the longest query made by A to $\mathcal{O}_H$. B runs in time $\mathcal{O}(\mathsf{Time}(A) + q_F t_H + q_P + q_g)$, makes $q_P + q_H q_F$ queries, q_g extraction queries, and outputs a preimage of length at most l where for any input M to H^P the output of $H^P(M)$ can be calculated within at most t_H times and q_H queries to P. ♦

Dodis et al. proved that the SMD construction preserves the PrA security as follows. Therefore, the PRO security of the NMAC hash function using the SMD hash function can be reduced into the PrA security of the compression function.

Lemma 3 (Theorem 4.2 of [7]). *Let h^P be a compression function using an ideal primitive P. Let $\mathcal{E}_h$ be an arbitrary extractor for h^P There exists an extractor $\mathcal{E}_H$ for SMD^{h^P} such that for any*

adversary A_H making at most q_P primitive queries and q_e extraction queries and outputting a message at most l blocks there exists an adversary A_h such that

$$\mathsf{Adv}^{\mathsf{pra}}_{\mathrm{SMD}^{h^P},P,\mathcal{E}_H}(A_H) \leq \mathsf{Adv}^{\mathsf{pra}}_{h^P,P,\mathcal{E}_h}(A_h)$$

$\mathcal{E}_H$ runs in time at most $l(\mathsf{Time}(\mathcal{E}_h) + \mathsf{Time}(\mathsf{unpad}))$. A_h runs in time at most $\mathcal{O}(\mathsf{Time}(A_H) + q_e l)$, makes at most $lq_H + q_P$ ideal primitive queries, and makes at most $q_e l$ extraction queries where q_H is the maximum number of P queries to calculate $\mathrm{SMD}^{h^P}(x)$ for any input x. ♦

3 Block Cipher-based Double-length Hash Functions for PROs

In this section, we propose double-length hash functions using a single block cipher with the $\mathcal{O}(2^n)$ PRO security. We prove the PRO security by using the three technique, the PrA design framework, PRO for a small function and *indifferentiability from a hash function*.

3.1 Primage Aware Double-length Hash Functions

In this subsection, we prove that Hirose's compression function [13] is PrA up to $\mathcal{O}(2^n)$ query complexity as long as the block cipher is an ideal cipher. Similarly, we can prove that Abreast-DM and Tandem-DM [15] are PrA. The PrA proofs for Abreast-DM and Tandem-DM are given in Appendix A and B. Let $\mathsf{BC}^1_{2n,n} = (E1, D1)$ be a block cipher. Let $\mathrm{DLCF}^{\mathsf{BC}^1_{2n,n}}$ be Hirose's compression function, Tandem-DM, or Abreast-DM using the block cipher $\mathsf{BC}^1_{2n,n}$. Let $g : \{0,1\}^{2n} \to \{0,1\}^{2n}$ be a function. Then, Lemmas 2 and 3 guarantee that the following hash function is PRO up to $\mathcal{O}(2^n)$ query complexity as long as the block cipher is an ideal cipher and g is a random oracle.

$$F_1^{\mathsf{BC}_{2n,n},g}(M) = \mathsf{NMAC}^{\mathrm{SMD}^{\mathrm{DLCF}^{\mathsf{BC}^1_{2n,n}}},g}(M) = g(\mathrm{SMD}^{\mathrm{DLCF}^{\mathsf{BC}^1_{2n,n}}}(M))$$

We call the function $\mathrm{SMD}^{\mathrm{DLCF}^{\mathsf{BC}^1_{2n,n}}}$ "inner hash function" and the function g "final function".

Hirose's compression function incorporates two Davies-Meyer (DM) single block length compression functions which are used side-by-side. The compression function is given in Definition 1.

Definition 1. *Let $\mathsf{BC}^1_{2n,n} = (E1, D1)$ be a block cipher. Let $\mathsf{CF}^{\mathsf{Hirose}}[\mathsf{BC}^1_{2n,n}] : \{0,1\}^{2n} \times \{0,1\}^n \to \{0,1\}^{2n}$ be a compression function such that $(G_i, H_i) = \mathsf{CF}^{\mathsf{Hirose}}[\mathsf{BC}^1_{2n,n}](G_{i-1}||H_{i-1}, M_i)$ where $G_i, H_i, G_{i-1}, H_{i-1} \in \{0,1\}^n$ and $M_i \in \{0,1\}^n$. (G_i, H_i) is calculated as follows:*

$$G_i = G_{i-1} \oplus E1(H_{i-1}||M_i, G_{i-1}) \tag{1}$$
$$H_i = C \oplus G_{i-1} \oplus E1(H_{i-1}||M_i, G_{i-1} \oplus C,) \tag{2}$$

We call the procedure 1 "first block" and the procedure 2 "second block". ♦

Theorem 1 (Hirose's Compression Function is PrA). *Let $\mathcal{C}^1_{2n,n} = (E1_I, D1_I)$ be an ideal cipher. There exists an extractor $\mathcal{E}$ such that for any adversary A making at most q_P queries to $\mathcal{C}_{2n,n}$ and q_e extraction queries we have*

$$\mathsf{Adv}^{\mathsf{pra}}_{\mathsf{CF}^{\mathsf{Hirose}}[\mathcal{C}^1_{2n,n}],\mathcal{C}_{2n,n},\mathcal{E}}(A) \leq \frac{2q_P^2}{(2^n - 2q_P)^2} + \frac{2q_P}{2^n - 2q_P} + \frac{2q_P q_e}{(2^n - q_P)^2}$$

where $\mathcal{E}$ runs in time at most $\mathcal{O}(q_e q_P)$. ♦

Proof. We prove that Hirose's compression function is 1-WPrA, and then Lemma 1 gives the final bound. We note that Theorem 3 of [10] upperbounds the cr-advantage of A by $2q_P^2/(2^n - 2q_P)^2 + 2q_P/(2^n - 2q_P)$, yielding the first two terms.

We define the multi-point extractor to utilize the preimage resistant bound, proven in [10], of Hirose's compression function as follows.

algorithm $\mathcal{E}^+(z,\alpha)$
Let L be an empty list;
Parse $(k_1,x_1,y_1),\ldots,(k_i,x_i,y_i) \leftarrow \alpha$; $//E1(k_j,x_j)=y_j$
For $j=1$ to i do
  If $z[0]=x_j\oplus y_j$ then
    $y \leftarrow E1_I(k_j, x_j\oplus C)$;
    If $z[1]=C\oplus x_j\oplus y$ then $L \xleftarrow{\cup} (x_j||k[0],k[1])$;
  If $z[1]=x_j\oplus y_j$ then
    $y \leftarrow E1_I(k_j, x_j\oplus C)$;
    If $z[0]=C\oplus x_j\oplus y$ then $L \xleftarrow{\cup} ((x_j\oplus C)||k[0],k[1])$;
If L is not an empty list then return L otherwise return $\perp$;

If an input-output triple of the first block is defined, automatically the input of the second block is defined, and vice versa, from the definition of the compression function. For a query (z,α) to $\mathcal{E}^+$, when there is an input-output triple (k,x,y) such that $x\oplus y=z[0]$, the multi-point extractor $\mathcal{E}^+$ checks whether the output of the second block is equal to $z[1]$ or not and if this holds the multi-point extractor stores it in the return list L, and vice versa. Therefore, A must find a preimage of z to win the 1-WPrA experiment. Thus one can straightforwardly adapt the preimage resistant advantage of the compression function (described in Theorem 5 of [10]). The advantage is at most $2q_P/(2^n-q_P)^2$. □

Remark 1. Using the above technique, we can obtain PrA security bounds of other DLHFs from bounds of collision resistant security and preimage resistant security (e.g. schemes in [8, 10, 23]).

3.2 Double-length Hash Function Using Two Block Ciphers

Let $\mathsf{BC}^2_{2n,n}=(E2,D2)$ and $\mathsf{BC}^3_{2n,n}=(E3,D3)$ be block ciphers. In this subsection, we propose a function $f^{\mathsf{BC}^3_{2n,n}}:\{0,1\}^{2n}\to\{0,1\}^{2n}$ that uses the block cipher $\mathsf{BC}^3_{2n,n}$ and prove that $f^{\mathsf{BC}^3_{2n,n}}$ is PRO up to $\mathcal{O}(2^n)$ query complexity as long as the block cipher is an ideal cipher. Since $F_1^{\mathsf{BC}^1_{2n,n},g}$ is PRO up to $\mathcal{O}(2^n)$ query complexity as long as the block cipher is an ideal cipher and g is a random oracle, the following hash function, which uses f instead of g, is PRO up to $\mathcal{O}(2^n)$ query complexity as long as the block ciphers are ideal ciphers.

$$F_2^{\mathsf{BC}^2_{2n,n},\mathsf{BC}^3_{2n,n}}(M)=f^{\mathsf{BC}^3_{2n,n}}(\mathrm{SMD}^{\mathrm{DLCF}^{\mathsf{BC}^2_{2n,n}}}(M))$$

where $\mathrm{DLCF}^{\mathsf{BC}^2_{2n,n}}$ is the double-length compression function, Hirose's compression function, Tandem-DM or Abreast-DM, using the block cipher $\mathsf{BC}^2_{2n,n}$.

We define the function $f^{\mathsf{BC}^3_{2n,n}}:\{0,1\}^{2n}\to\{0,1\}^{2n}$ as follows.

Definition 2. *Let $\mathsf{BC}^3_{2n,n}=(E3,D3)$ be a block cipher. The function $f^{\mathsf{BC}^3_{2n,n}}$ is built as*

$$f^{\mathsf{BC}^3_{2n,n}}(x)=E3(x,c_1)||E3(x,c_2)$$

where c_1 and c_2 are n-bit constant values such that $c_1\neq c_2$. ♦

Theorem 2 ($f^{\mathcal{C}^3_{2n,n}}$ is PRO). *Let $\mathcal{C}^3_{2n,n}=(E3_I,D3_I)$ be an ideal cipher. Let $g=\mathcal{F}_{2n,2n}$. There exists a simulator $S=(S_E,S_D)$ such that for any distinguisher A making at most q_f, q_E and q_D queries to oracles $(\mathcal{O}_f,\mathcal{O}_E,\mathcal{O}_D)$ where $(\mathcal{O}_f,\mathcal{O}_E,\mathcal{O}_D)=(f^{\mathcal{C}^3_{2n,n}},E3_I,D3_I)$ or $(\mathcal{O}_f,\mathcal{O}_E,\mathcal{O}_D)=(g,S_E,S_D)$, we have*

$$\mathsf{Adv}^{\mathsf{pro}}_{f^{\mathcal{C}^3_{2n,n}},S}(A)\le\frac{q_f+q_E+q_D}{2^n}$$

where S works in time $\mathcal{O}(\mathsf{Time}(A)+q_E+q_D)$ and makes at most queries q_E+q_D. ♦

Intuition. Consider the fixed plain text encryption oracle of the ideal cipher $f_c(x) = E3_I(x, c)$ where c is an n-bit constant value. Since the second input is fixed by c, the output of $f_c(x)$ is chosen from the uniformly distribution on $\{0,1\}^n$. Therefore, when considering two functions f_{c_1} and f_{c_2} where $c_1 \neq c_2$, the output y_1 of $f_{c_1}(x)$ is chosen from the uniformly distribution on $\{0,1\}^n$ and the output y_2 of $f_{c_2}(x)$ is chosen from the uniform distribution on $\{0,1\}^n \backslash \{y_1\}$. Namely, the first n-bit value of $f^{\mathcal{C}^3_{2n,n}}(x)$ is chosen uniformly from $\{0,1\}^n$ and the last n-bit value is chosen from the uniform distribution on $\{0,1\}^n \backslash \{y_1\}$. Since g is a random oracle, if for any query to $g(x)$ the first n-bit value of the output is not equal to the last n-bit value, $f^{\mathcal{C}^3_{2n,n}}$ is equal to g, namely, in this case no A can differentiate $f^{\mathcal{C}^3_{2n,n}}$ from g. Since g is a random oracle, the probability that A can differentiate $f^{\mathcal{C}^3_{2n,n}}$ from g is negligible. ♦

Proof. We define $S = (S_E, S_D)$ such that it simulates $\mathcal{C}^3_{2n,n} = (E3_I, D3_I)$ and the relation among responses of $(f^{\mathcal{C}^3_{2n,n}}, E3_I, D3_I)$ holds in responses of (g, S_E, S_D) as well. Since $f^{\mathcal{C}^3_{2n,n}}(k) = E3_I(k, c_1)||E3_I(k, c_2)$, we define S such that $g(k) = S_E(k, c_1)||S_E(k, c_2)$. Namely, the output of $S_E(k, c_1)$ is $y[0]$, the output of $S_E(k, c_2)$ is $y[1]$, the output of $S_D(k, y[0])$ is c_1, and the output of $S_D(k, y[1])$ is c_2 where $y = g(k)$. Since S simulates the ideal cipher, if $x \neq c_1$ and $x \neq c_2$, the output $S_E(k, x)$ is chosen from the uniform distribution on the set of $\{0,1\}^n$ except for previous responses, and if a query to $S_D(k, y)$ does not correspond with c_1 and c_2, the output is chosen from the uniform distribution on the set of $\{0,1\}^n$ except for previous responses. Thus we define S as follows.

simulator $S_E(k, x)$	**simulator** $S_D(k, y)$
01 If $\mathsf{E}[k, x] \neq \perp$ then ret $\mathsf{E}[k, x]$;	11 If $\mathsf{D}[k, y] \neq \perp$ then ret $\mathsf{D}[k, y]$;
02 $y^* \leftarrow g(k)$;	12 $y^* \leftarrow g(k)$;
03 If $x = c_1$, $y \leftarrow y^*[0]$;	13 If $y = y^*[0]$, $x \leftarrow c_1$;
04 Else if $x = c_2$, $y \leftarrow y^*[1]$;	14 Else if $y = y^*[1]$, $x \leftarrow c_2$;
05 Else $y \xleftarrow{\$} \{0,1\}^n \backslash (\{y^*[0], y^*[1]\} \cup \mathcal{T}_E[k])$;	15 Else $x \xleftarrow{\$} \{0,1\}^n \backslash (\{c_1, c_2\} \cup \mathcal{T}_D[k])$;
06 $\mathsf{E}[k, x] \leftarrow y$;	16 $\mathsf{E}[k, x] \leftarrow y$;
07 $\mathsf{D}[k, y] \leftarrow x$;	17 $\mathsf{D}[k, y] \leftarrow x$;
08 Ret y;	18 Ret x;

S has (initially everywhere $\perp$) arrays E, D and (initially empty) tables $\mathcal{T}_E, \mathcal{T}_D$. When $y = S_E(k, x)$, y is stored in $\mathsf{E}[k, x]$ and x is stored in $\mathsf{D}[k, y]$. If $\mathsf{E}[k, x]$ is defined, $\mathsf{E}[k, x]$ is stored in $\mathcal{T}_E[k]$, and if $\mathsf{D}[k, y]$ is defined, $\mathsf{D}[k, y]$ is stored in $\mathcal{T}_D[k]$.

We give the proof via a game-playing argument by the following game sequences Game G0, Game G1, Game G2 and Game G3. In each game, A can make queries to three oracles $(\mathcal{O}_f, \mathcal{O}_E, \mathcal{O}_D)$.

- **Game G0:** This is the $\mathcal{C}^3_{2n,n}$ scenario, that is, $(\mathcal{O}_f, \mathcal{O}_E, \mathcal{O}_D) = (f^{\mathcal{C}^3_{2n,n}}, E3_I, D3_I)$.
- **Game G1:** In this game, we replace $(\mathcal{O}_E, \mathcal{O}_D)$ from $\mathcal{C}^3_{2n,n}$ to S. Thus $(\mathcal{O}_f, \mathcal{O}_E, \mathcal{O}_D) = (f^S, S_E, S_D)$. Note that $f^S(k) = S_E(k, c_1)||S_E(k, c_2)$
- **Game G2:** In this game, we modify $\mathcal{O}_f$ from f^S to a new function f_1^S. f_1^S is that for query k it calculates $f^S(k)$ and returns the output of $g(k)$. Thus $(\mathcal{O}_f, \mathcal{O}_E, \mathcal{O}_D) = (f_1^S, S_E, S_D)$.
- **Game G3:** This is the final game. This game is the g scenario. That is, $(\mathcal{O}_f, \mathcal{O}_E, \mathcal{O}_D) = (g, S_E, S_D)$.

Let Gj be the event that in Game Gj distinguisher A outputs 1.

G0→G1: Since $\mathcal{O}_E$ and $\mathcal{O}_D$ are modified from $\mathcal{C}^3_{2n,n}$ to S, we show that S behaves like an ideal cipher. For a key k, first the output $S_E(k, c_1)$ $(= y^*[0])$ is chosen from the uniform distribution on $\{0,1\}^n$ (due to g) and second the output of $S_E(k, c_2)$ $(= y^*[1])$ is chosen from the uniform distribution on $\{0,1\}^n$ (due to g). The output of $S_E(k, x)$ such that $x \neq c_1$ and $x \neq c_2$ is chosen from the uniform distribution on $\{0,1\}^n \backslash (\{y^*[0], y^*[1]\} \cup \mathcal{T}_E[k])$ after the outputs of $S_E(k, c_1)$ and $S_E(k, c_2)$ are defined. Thus, if the output of $S_E(k, c_2)$ $(= y^*[1])$ does not collide with that of $S_E(k, c_1)$ $(= y^*[0])$, S_E behaves

like an encryption oracle of an ideal cipher. Similarly, if $y^*[1]$ does not collide with $y^*[0]$, S_D behaves like a decryption oracle. Thus the A's view in Game G1 is identical with that in Game G0 if for any query to $g(x)$ $y[1] \neq y[0]$ where $y = g(x)$. Since the number of g queries is at most $q_f + q_E + q_D$, we have that $|Pr[G1] - Pr[G0]| \leq \frac{q_f+q_E+q_D}{2^n}$.

G1→G2: In Game G1 the output of $\mathcal{O}_f(k)$ is defined by $f^S(M)$, while in Game 2 the output of $\mathcal{O}_f(k)$ is defined by $g(k)$. Since in Game G1 $\mathcal{O}_f(k)$ $(= S_E(k, c_1)||S_E(k, c_2))$ is defined by $g(k)$ due to Lines 02-04 and 12-14, the A's view in Game G2 is identical with that in Game G1 and we have that $Pr[G1] = Pr[G2]$.

G2→G3: This is the final game transformation. In both games, the output of $\mathcal{O}_f(k)$ is defined by $g(k)$. However, in Game G2, for any query to $\mathcal{O}_f(k)$ it makes queries to $S_E(k, c_1)$ and $S_E(k, c_2)$ and then S_E returns $y^*[0]$ and $y^*[1]$ respectively where y^* is the output value of $g(k)$, while in Game G3 $\mathcal{O}_f$ does not makes these S_E queries. Since the output of $S_E(k, c_1)$ is always defined by $y^*[0]$, the output of $S_E(k, c_2)$ is always defined by $y^*[1]$ and in Game G2 A cannot see queries from $\mathcal{O}_f$ to S, the modification in the game transform does not change the A's view. Therefore, $Pr[G2] = Pr[G3]$.

Consequently, the above discussion guarantees the claim of Theorem 2. □

3.3 Double-length Hash Function Using Single Block Cipher

Let $\mathsf{BC}_{2n,n} = (E, D)$, $\mathsf{BC}^2_{2n,n} = (E2, D2)$ and $\mathsf{BC}^3_{2n,n} = (E3, D3)$ be block ciphers. In this subsection, we prove that the following hash function is PRO up to $\mathcal{O}(2^n)$ query complexity as long as the block cipher is an ideal cipher.

$$F^{\mathsf{BC}_{2n,n}}(M) = f^{\mathsf{BC}_{2n,n}}(\mathrm{SMD}^{\mathrm{DLCF}^{\mathsf{BC}_{2n,n}}}(M))$$

where $\mathrm{DLCF}^{\mathsf{BC}_{2n,n}}$ is Hirose's compression function, Tandem-DM, or Abreast-DM using the block cipher $\mathsf{BC}_{2n,n}$. To prove the PRO security, we consider a new application of indifferentiability [19]. We call the new application *indifferentiability from a hash function*. We apply this to $F^{\mathsf{BC}_{2n,n}}$ and $F_2^{\mathsf{BC}^2_{2n,n},\mathsf{BC}^3_{2n,n}}$ and prove that $F^{\mathsf{BC}_{2n,n}}$ is indifferentiable from $F_2^{\mathsf{BC}^2_{2n,n},\mathsf{BC}^3_{2n,n}}$ up to $\mathcal{O}(2^n)$ query complexity as long as the block ciphers are ideal ciphers. Then, since $F_2^{\mathsf{BC}^2_{2n,n},\mathsf{BC}^3_{2n,n}}$ is PRO up to $\mathcal{O}(2^n)$ query complexity as long as the block ciphers are ideal ciphers, the indifferentiability framework guarantees that $F^{\mathsf{BC}_{2n,n}}$ is PRO up to $\mathcal{O}(2^n)$ query complexity as long as the block cipher is an ideal cipher.

In the following proof, we only consider the case of Hirose's compression function. Using the same proof, we can prove the cases of Tandem-DM and Abreast-DM. So we omit these proofs. Note that when using Hirose's compression function, we use the constant values c_1 and c_2 of the final function f such that c_1 and c_2 are not equal to $C \oplus IV[0]$ where IV is the initial value of $\mathrm{SMD}^{\mathrm{DLCF}^{\mathsf{BC}_{2n,n}}}$ and C is the constant value used in Hirose's compression function.

Theorem 3. *Let $\mathcal{C}_{2n,n} = (E_I, D_I)$ be an ideal cipher. Let $\mathcal{C}^2_{2n,n} = (E2_I, D2_I)$ and $\mathcal{C}^3_{2n,n} = (E3_I, D3_I)$ be different ideal ciphers. There exists a simulator $S = (S_E, S_D)$ such that for any distinguisher A making at most q_F, q_E and q_D queries to its oracles $(\mathcal{O}_F, \mathcal{O}_E, \mathcal{O}_D)$ where $(\mathcal{O}_F, \mathcal{O}_E, \mathcal{O}_D) = (F^{\mathcal{C}_{2n,n}}, E_I, D_I)$ or $(\mathcal{O}_F, \mathcal{O}_E, \mathcal{O}_D) = (F_2^{\mathcal{C}^2_{2n,n},\mathcal{C}^3_{2n,n}}, S_E, S_D)$, we have*

$$\mathsf{Adv}^{\mathsf{indif}}_{F,F_2,S}(A) \leq \frac{8(2lq_F + q_E + q_D)}{2^n - (2lq_F + q_E + q_D)}$$

where S works in time $\mathcal{O}(\mathsf{Time}(A) + q_E + q_D)$ and makes at most ideal cipher queries $q_E + q_D$. l is the maximum number of n-bit blocks of a query to $\mathcal{O}_F$. ♦

Intuition. The difference between $F^{\mathcal{C}_{2n,n}}$ and $F_2^{\mathcal{C}^2_{2n,n},\mathcal{C}^3_{2n,n}}$ is the number of underlying ideal ciphers. $F^{\mathcal{C}_{2n,n}}$ uses the single ideal cipher, while $F_2^{\mathcal{C}^2_{2n,n},\mathcal{C}^3_{2n,n}}$ use two ideal ciphers. In $F_2^{\mathcal{C}^2_{2n,n},\mathcal{C}^3_{2n,n}}$, $\mathcal{C}^2_{2n,n}$ is used in the inner hash function and $\mathcal{C}^3_{2n,n}$ is used in the final function. Therefore, if in $F^{\mathcal{C}_{2n,n}}$ input-output triples of $\mathcal{C}_{2n,n}$ used in the inner hash function are not used in the final function, then we can see that in $F^{\mathcal{C}_{2n,n}}$ the ideal cipher used in the inner hash function is different from that used in the final function. Therefore, since in $F^{\mathcal{C}_{2n,n}}$ the second input of the encryption function used in the final function is c_1 or c_2, if no second input of the encryption function used in the inner hash function is equal to c_1 or c_2, no A can differentiate $(F^{\mathcal{C}_{2n,n}}, E_I, D_I)$ from $(F_2^{\mathcal{C}^2_{2n,n},\mathcal{C}^3_{2n,n}}, S_E, S_D)$. Since second inputs of the encryption function used in the inner hash function are determined by chaining values which are determined by outputs of the ideal cipher, the probability that some second input of the encryption function in the inner hash function is equal to c_1 or c_2 is negligible. ♦

Proof. We define a simulator $S = (S_E, S_D)$ such that it simulates the ideal cipher $\mathcal{C}_{2n,n} = (E_I, D_I)$ and the relation among responses of $(F_2^{\mathcal{C}^2_{2n,n},\mathcal{C}^3_{2n,n}}, S_E, S_D)$ holds in responses of $(F^{\mathcal{C}_{2n,n}}, E_I, D_I)$ as well, namely, $F^S(M) = F_2^{\mathcal{C}^2_{2n,n},\mathcal{C}^3_{2n,n}}(M)$. Since $E2_I$ is used in inner calculations and $E3_I$ is used in final calculations, if for a query to $S_E(k,x)$ (k,x) is used in the final calculations, it returns the output of $E3_I(k,x)$, and otherwise it returns the output of $E1_I(k,x)$. Since in the final calculation the second input x of a S_E query is c_1 or c_2, we define S such that if $x = c_1$ or $x = c_2$, $S_E(k,x)$ is defined by $E3_I(k,x)$, and otherwise defined by $E2_I(k,x)$.

simulator $S_E(k,x)$	**simulator** $S_D(k,y)$
01 If $\mathsf{E}[k,x] \neq \perp$ then ret $\mathsf{E}[k,x]$;	11 If $\mathsf{D}[k,y] \neq \perp$ then ret $\mathsf{D}[k,y]$;
02 $y \leftarrow E3_I(k,x)$;	12 $x \leftarrow D3_I(k,y)$;
03 If $x \neq c_1$ and $x \neq c_2$, $y \leftarrow E2_I(k,x)$;	13 If $x \neq c_1$ and $x \neq c_2$, $x \leftarrow D2_I(k,y)$;
04 $\mathsf{E}[k,x] \leftarrow y$;	14 $\mathsf{E}[k,x] \leftarrow y$;
05 $\mathsf{D}[k,y] \leftarrow x$;	15 $\mathsf{D}[k,y] \leftarrow x$;
06 Ret y;	16 Ret x;

The simulator has (initially everywhere $\perp$) arrays E and D. For an input-output triple (k,x,y) of S_E and S_D, y is stored in $\mathsf{E}[k,x]$ and x is stored in $\mathsf{D}[k,y]$ where $y = S_E(k,x)$ and $x = S_D(k,y)$.

We give the proof via a game-playing argument by using the following game sequences $G0, G1, G2$, and $G3$. In each game, A can make queries to three oracles $(\mathcal{O}_F, \mathcal{O}_E, \mathcal{O}_D)$.

- **Game G0:** This is the $\mathcal{C}_{2n,n}$ scenario. Thus $(\mathcal{O}_F, \mathcal{O}_E, \mathcal{O}_D) = (F^{\mathcal{C}_{2n,n}}, E_I, D_I)$.
- **Game G1:** In this game, $(\mathcal{O}_E, \mathcal{O}_D)$ is modified from $\mathcal{C}_{2n,n}$ to S. Thus $(\mathcal{O}_F, \mathcal{O}_E, \mathcal{O}_D) = (F^S, S_E, S_D)$.
- **Game G2** In this game, the ideal cipher used in the final function is modified from S to $\mathcal{C}^3_{2n,n}$. Namely, $\mathcal{O}_F = F_2^{S,\mathcal{C}^3_{2n,n}}$ and thus $(\mathcal{O}_F, \mathcal{O}_E, \mathcal{O}_D) = (F_2^{\mathcal{C}^2_{2n,n},S}, S_E, S_D)$.
- **Game G3** This is the final game. In this game, the ideal cipher used in the inner hash function is modified from S to $\mathcal{C}^2_{2n,n}$. Namely, $\mathcal{O}_F = F_2^{\mathcal{C}^2_{2n,n},\mathcal{C}^3_{2n,n}}$ and thus $(\mathcal{O}_F, \mathcal{O}_E, \mathcal{O}_D) = (F_2^{\mathcal{C}^2_{2n,n},\mathcal{C}^3_{2n,n}}, S_E, S_D)$.

Let Gj be the event that in Game Gj A outputs 1.

G0→G1: The underlying block cipher used in both games is different. Therefore, if S behaves like an ideal cipher, the A's view in Game G1 is equal to that in Game G0. If any output of $E2_I(k,\cdot)$ does not collide with $E3_I(k,c_1)$ and $E3_I(k,c_2)$ ($\neg Event1$) and the output of $D2_I(k,y)$ does not collide with c_1 and c_2 ($\neg Event2$), S is an ideal cipher because a fixed key ideal cipher is a random permutation.

- $Event1$: The output of $S_E(k,x)$ (defined by $E2_I(k,x)$) such that $x \neq c_1$ and $x \neq c_2$ is equal to y_1 or y_2 where $y_1 = E3_I(k,c_1)$ and $y_2 = E3_I(k,c_2)$.
- $Event2$: The output of $S_D(k,y)$ (defined by $D2_I(k,y)$) such that $y \neq E3_I(k,c_1)$ and $y \neq E3_I(k,c_1)$ is equal to c_1 or c_2.

Therefore,

$$|Pr[G1] - Pr[G0]| \leq Pr[Event1] + Pr[Event2].$$

First we evaluate $Pr[Event1]$. Note that since no A can make a query to $D2_I$, she does not know the input (k, x) of $E2_I$ such that $E2_I(k, x)$ is equal to $E3_I(k, c_1)$ or $E3_I(k, c_2)$ before $Event1$ occurs. Therefore, the probability $Pr[Event1]$ is determined by randomness of outputs of $E2_I$. The number of $E2_I$ and $D2_I$ queries is at most $2lq_F + q_E + q_D$, the number of queries to S_E is at most $2lq_F + q_E$ where the second inputs are not equal to c_1 and c_2. Thus,

$$Pr[Event1] \leq (2lq_F + q_E) \times \frac{2}{2^n - (2lq_F + q_E + q_D)}.$$

Second we evaluate $Pr[Event2]$. Note that since no A can make a query to $E2_I$, she does not know the input (k, y) of $D2_I$ before $Event2$ occurs such that $D2_I(k, y) = c_1$ or c_2. Therefore, the probability $Pr[Event2]$ is determined by randomness of outputs of $D2$. Since the number of S_D queries is at most q_D,

$$Pr[Event2] \leq q_D \times \frac{2}{2^n - (2lq_F + q_E + q_D)}.$$

Consequently, we have

$$|Pr[G1] - Pr[G0]| \leq \frac{2(2lq_F + q_E + q_D)}{2^n - (2lq_F + q_E + q_D)}.$$

G1→G2: The difference between both games is an ideal cipher of the final function: in Game G1 S is used while in Game G2 $\mathcal{C}^3_{2n,n}$. Therefore, in the final function in Game G2 $\mathcal{O}_F$ does not make a S_E query. We assume that in both games $Event1$ and $Event2$, which are defined in the previous game transform, do not occur. Then, for a value k, $S(k, \cdot)$ is a random permutation. This means that the output of any query to $S_E(k, x)$ is not doubly defined, namely, if $x = c_1$ or $x = c_2$, the output $S_E(k, x)$ is always defined by $E3_I(k, x)$. Therefore, if in both games $Event1$ and $Event2$ does not occur, the modification of $\mathcal{O}_F$ in the game transform does not change the A's view. In both games, the number of queries to $E2$ and $D2$ is at most $2lq_F + q_E + q_D$, the number of queries to S_E is at most $2lq_F + q_E$ where the second inputs are not equal to c_1 and c_2, and the number of queries to S_D is at most q_D. Therefore, the probability $Pr[Event1] + Pr[Event2]$ is the same as that in previous game transform and we have that

$$|Pr[G2] - Pr[G1]| \leq \frac{2(2lq_F + q_E + q_D)}{2^n - (2lq_F + q_E + q_D)}.$$

G2→G3: The following figure illustrates Game G2 and Game G3. When boxed statements are included, the following figure illustrates Game G2. When boxed statements are removed, the following figure illustrates Game G3.

$\underline{\mathcal{O}_E(k,x)}$
01 If $\mathsf{E}[k,x] \neq \perp$, ret $\mathsf{E}[k,x]$;
02 $y \leftarrow E3_I(k,x)$;
03 If $x \neq c_1$ and $x \neq c_2$, $y \leftarrow E2_I(k,x)$;
04 $\mathsf{E}[k,x] \leftarrow y$;
05 $\mathsf{D}[k,y] \leftarrow x$;
06 Ret y;

$\underline{\mathcal{O}_D(k,y)}$
11 If $\mathsf{D}[k,y] \neq \perp$ then ret $\mathsf{D}[k,y]$;
12 $x \leftarrow D3_I(k,y)$;
13 If $x \neq c_1$ and $x \neq c_2$, $x \leftarrow D2_I(k,y)$;
14 $\mathsf{E}[k,x] \leftarrow y$;
15 $\mathsf{D}[k,y] \leftarrow x$;
16 Ret x;

$\underline{\mathcal{O}_F(M)}$
21 $M_1||\ldots||M_i \leftarrow \mathsf{sfpad}(M)$; $x_1 \leftarrow IV[0]$; $x_2 \leftarrow IV[1]$;
22 For $j = 1, \ldots, i$,
23 $\quad y_1 \leftarrow E2_I(x_2||M_j, x_1)$;
24 $\quad$ If $x_1 = c_1$ or $x_1 = c_2$, $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{y_1 \leftarrow \mathcal{O}_E(x_2||M_j, x_1);}$
25 $\quad y_2 = E2_I(x_2||M_j, x_1 \oplus C)$;
26 $\quad$ If $x_1 \oplus C = c_1$ or $x_1 \oplus C = c_2$, $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{y_1 \leftarrow \mathcal{O}_E(x_2||M_j, x_1 \oplus C);}$
27 $\quad x_2 \leftarrow x_1 \oplus C \oplus y_2$;
28 $\quad x_1 \leftarrow x_1 \oplus y_1$;
29 $z_1 \leftarrow E3_I(x_1||x_2, c_1)$; $z_2 \leftarrow E3_I(x_1||x_2, c_2)$;
30 Ret $z_1||z_2$;

When boxed statements are removed, the figure explicitly illustrates Game G3. So we demonstrate that when boxed statements are included, the figure illustrates Game G2. If A sets bad, Lines 23-24 and 25-26 simulates S_E. If A does not set bad, in Lines 23-24 the output of $E2_I(x_2||M_j, x_1)$ is used and in Lines 25-26 the output of $E2_I(x_2||M_j, x_1 \oplus C)$ is used. Since if $x \neq c_1$ and $x \neq c_2$ the output of $S_E(k,x)$ is always defined by $E2_I(k,x)$, in this case, Lines 23-24 and 25-26 simulate S_E. Therefore, when boxed statements are included, the figure illustrates Game G2. Thus the A's view in Game G3 is identical with that in Game G2 unless A sets bad and we have that

$$|Pr[G3] - Pr[G2]| \leq Pr[\mathsf{bad} \leftarrow \mathsf{true}]|.$$

We evaluate the probability $Pr[\mathsf{bad} \leftarrow \mathsf{true}]$. When A sets bad, for some input-output triple (k,x,y) of $E2_I$ where $y = E2_I(k,x)$, $x \oplus y$ is equal to $c_1, c_2, c_1 \oplus C$ or $c_2 \oplus C$. Since the number of queries to $\mathcal{C}^2_{2n,n}$ is at most $2lq_F + q_E + q_D$,

$$Pr[\mathsf{bad} \leftarrow \mathsf{true}] \leq 4 \times \frac{2lq_F + q_E + q_D}{2^n - (2lq_F + q_E + q_D)}$$

Consequently, we can obtain the bound of the theorem. □

4 Other Applications of Indifferentiability from a Hash Function.

Indifferentiability from a hash function may be useful for reducing the number of underlying primitives. For example, we consider the polynomial-based hash function [17] which uses two random oracles and is PRO. Using indifferentiability from a hash function, this hash function can be easily modified that using a single random oracle with the same PRO security. In this case, it is proven that the hash function using a single ideal cipher is indifferentiable from that using two ideal ciphers. In this proof, we have only to consider the difference of the number of the ideal ciphers and don't need to consider the gap between a random oracle and the hash function.

Other applications are PRO proofs for the MD variants (e.g. prefix-free MD [4], EMD [1] and MDP [14]). The PRO security can be proven by (1) the NMAC hash function is PRO (this can be easily

proven by the PrA design framework [6]) and (2) the MD variant is indifferentiable from the NMAC hash function. The inner hash function of both the NMAC hash function and the MD variant uses the iterated hash function of a compression function. But the final calculations of the two hash function are different. Therefore, in the second step, we have only to consider the difference between the final functions. The proof of the first step is easy, since the PRO security of the NMAC hash function can be easily proven by the PrA design framework and PRO from a small primitive, like the proof of our double-length hash functions. Therefore, the proof method is easier than the traditional proof.

5 Block Cipher-based Double-pipe Hash Function

Since a chopped random oracle is also a random oracle, the hash function $chop \circ F^{\mathcal{C}_{2n,n}}$ is PRO with $\mathcal{O}(2^n)$ PRO security where $F^{\mathcal{C}_{2n,n}}$ is our hash function proposed in Section 3 and *chop* is a chop function that chops n-bits of the input and outputs remaining bits. That is, the chopped hash function is the double-pipe hash function using a single block cipher with the PRO security beyond the birthday bound. Note that several double-pipe hash functions using a random oracle compression function with the PRO security beyond the birthday bound were proposed (e.g. [3]). The output length of these hash function is n-bits and the compression function maps a d-bit input to a n-bit output where $d > 2n$. Since a fixed plaintext ideal cipher is a random oracle, these double-pipe hash function using a d-bit key length ideal cipher is also PRO with the security beyond the birthday bound. However, the key length of the block cipher used in our double-pipe hash functions is shorter than that used in existing hash functions. Therefore, our double-pipe hash functions are better than existing double-pipe hash functions.

6 Conclusion

In this paper, we proposed the first time double-pipe hash function based on a *practical size* block cipher with the PRO *birthday* security. We proved the PRO security by using three techniques: the PrA design framework [6], PRO [4] for a small function, and indifferentiability from a hash function. First, we proved that Hirose's compression function [13], Tandem-DM [15], and Abreast-DM [15], are PrA up to $\mathcal{O}(2^n)$ query complexity. To our knowledge, this is the first time PrA proof of double-length compression functions (hash functions). Second, we proposed the block cipher-based function f and proved that f is PRO up to $\mathcal{O}(2^n)$ query complexity. Thus, the first result and the second result yield the double-length hash functions using two block ciphers which is PRO up to $\mathcal{O}(2^n)$ query complexity. Finally, using indifferentiability from a hash function which is a new application of indifferentiability [19], we reduced the number of the underlying block ciphers. Thus the final result yields the double-length hash functions using a single block cipher which are PROs up to $\mathcal{O}(2^n)$ query complexity.

References

1. Mihir Bellare and Thomas Ristenpart. Multi-Property-Preserving Hash Domain Extension and the EMD Transform. In *ASIACRYPT*, pages 299–314, 2006.
2. Donghoon Chang, Sangjin Lee, Mridul Nandi, and Moti Yung. Indifferentiable Security Analysis of Popular Hash Functions with Prefix-Free Padding. In *ASIACRYPT*, volume 4284 of *Lecture Notes in Computer Science*, pages 283–298. Springer, 2006.
3. Donghoon Chang and Mridul Nandi. Improved Indifferentiability Security Analysis of chopMD Hash Functionl. In *FSE*, pages pages 429–443, 2008.
4. Jean-Sébastien Coron, Yevgeniy Dodis, Cécile Malinaud, and Prashant Puniya. Merkle-Damgård Revisited: How to Construct a Hash Function. In *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 430–448. Springer, 2005.

5. Ivan Damgård. A Design Principle for Hash Functions. In *CRYPTO*, volume 435 of *Lecture Notes in Computer Science*, pages 416–427. Springer, 1989.
6. Yevgeniy Dodis, Thomas Ristenpart, and Thomas Shrimpton. Salvaging Merkle-Damgård for Practical Applications. In *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 371–388. Springer, 2009.
7. Yevgeniy Dodis, Thomas Ristenpart, and Thomas Shrimpton. Salvaging Merkle-Damgård for Practical Applications. In *ePrint 2009/177*, 2009.
8. Ewan Fleischmann, Christian Forler, Michael Gorski, and Stefan Lucks. Collision Resistant Double-Length Hashingl. In *ProvSec*, pages 102–118, 2010.
9. Ewan Fleischmann, Michael Gorski, and Stefan Lucksl. On the Security of Tandem-DM. In *FSE*, pages 84–103, 2009.
10. Ewan Fleischmann, Michael Gorski, and Stefan Lucksl. Security of Cyclic Double Block Length Hash Functions. In *IMA Int. Conf*, pages 153–175, 2009.
11. National Institute for Standards and Technology. cryptographic hash project . 2007.
12. Zheng Gong, Xuejia Lai, and Kefei Chen. A synthetic indifferentiability analysis of some block-cipher-based hash functions. In *Des. Codes Cryptography 48*, pages 293–305, 2008.
13. Shoichi Hirose. Some Plausible Constructions of Double-Block-Length Hash Functions. In *FSE*, pages 210–225, 2006.
14. Shoichi Hirose, Je Hong Park, and Aaram Yun. A Simple Variant of the Merkle-Damgård Scheme with a Permutation. In *ASIACRYPT*, volume 4833 of *Lecture Notes in Computer Science*, pages 113–129. Springer, 2007.
15. Xuejia Lai and James L. Massey. Hash Function Based on Block Ciphers. In *EUROCRYPT*, pages 55–70, 1992.
16. Jooyoung Lee, Martijn Stam, and John Steinberger. The collision security of Tandem-DM in the ideal cipher model. ePrint 2010/409, 2010.
17. Jooyoung Lee and John Steinberger. Multi-property-preserving Domain Extension Using Polynomial-based Modes of Operation. In *EUROCRYPT and ePrint 2010/131*, 2010.
18. S. Matyas, C. Meyer, and J. Oseas. Generating strong one-way functions with cryptographic algorithms. In *IBM Technical Disclosure Bulletin 27(10a)*, pages 5658–5659, 1985.
19. Ueli M. Maurer, Renato Renner, and Clemens Holenstein. Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 21–39. Springer, 2004.
20. Alfred Menezes, Paul C. van Oorschot, and Scott A. Vanstone. Handbook of Applied Cryptography. In *CRC Press*, 1996.
21. Ralph C. Merkle. One Way Hash Functions and DES. In *CRYPTO*, volume 435 of *Lecture Notes in Computer Science*, pages 428–446. Springer, 1989.
22. National Institute of Standards and Technoloty. FIPS PUB 180-3 Secure Hash Standard. In *FIPS PUB*, 2008.
23. Onur Özen and Martijn Stam. Another Glance at Double-Length Hashing. In *IMA Int. Conf*, pages 176–201, 2009.
24. Bart Preneel, Antoon Bosselaers, Rene Govaerts, and Joos Vandewalle. Collision-free Hashfunctions Based on Blockcipher Algorithmsl. In *Proceedings of 1989 International Carnahan Conference on Security Technology*, pages pages 203–210, 1989.
25. Bart Preneel, René Govaerts, and Joos Vandewalle. Hash Functions Based on Block Ciphers: A Synthetic Approach. In *CRYPTO*, volume 773 of *Lecture Notes in Computer Science*, pages 368–378. Springer, 1993.
26. Ronald L. Rivest. The MD4 Message Digest Algorithm. In *CRYPTO*, Lecture Notes in Computer Science, pages 303–311. Springer, 1990.
27. Ronald L. Rivest. The MD5 Message Digest Algorithm. In *RFC 1321*, 1992.

A Abreast-DM Is PrA

Abreast-DM [15] incorporates two Davies-Meyer (DM) single block length compression functions which are used side-by-side. The compression function is formally given in Definition 3.

Definition 3. *Let* $\mathsf{BC}_{2n,n} = (E, D)$ *be a block cipher. Let* $\mathsf{CF}^{\mathsf{ADM}}[\mathsf{BC}_{2n,n}] : \{0,1\}^{2n} \times \{0,1\}^n \rightarrow \{0,1\}^{2n}$ *be a compression function such that* $(G_i, H_i) = \mathsf{CF}^{\mathsf{ADM}}[\mathsf{BC}_{2n,n}](G_{i-1}||H_{i-1}, M_i)$ *where* $G_i, H_i, M_i, G_{i-1}, H_{i-1} \in \{0,1\}^n$. (G_i, H_i) *is calculated as follows:*

$$G_i = G_{i-1} \oplus E(H_{i-1}||M_i, G_{i-1})$$
$$H_i = H_{i-1} \oplus E(M_i||G_{i-1}, \overline{H_{i-1}})$$

where $\overline{H}$ *denotes the bit-by-bit complement of* H*. We call the first procedure "first block" and the second procedure "second block".*

We show that the Abreast-DM compression function is PrA with $\mathcal{O}(2^n)$ security.

Theorem 4 (Abreast-DM is PrA). *Let* $\mathcal{C}_{2n,n} = (E_I, D_I)$ *be an ideal cipher. There exists an extractor* $\mathcal{E}$ *such that for any adversary* A *making at most* q_P *queries to* $\mathcal{C}_{2n,n}$ *and* q_e *extraction queries we have*

$$\mathsf{Adv}^{\mathsf{pra}}_{\mathsf{CF}^{\mathsf{ADM}}[\mathcal{C}_{2n,n}],\mathcal{C}_{2n,n},\mathcal{E}}(A) \leq 18\left(\frac{q_P}{2^{n-1}}\right)^2 + \frac{2q_Pq_e}{(2^n - q_P)^2}$$

where $\mathcal{E}$ *runs in time at most* $\mathcal{O}(q_eq_P)$*.*

Proof. We will prove that any such compression function is 1-WPrA, and then Lemma 1 gives the final bound. We note that Theorem 1 of [10] upperbounds the cr-advantage by $18(q_P/2^{n-1})^2$, yielding the first term above. Let us define the multi-point extractor $\mathcal{E}^+$ as follows.

algorithm $\mathcal{E}^+(z, \alpha)$
Let L be an empty list;
Parse $(k_1, x_1, y_1), \ldots, (k_i, x_i, y_i) \leftarrow \alpha$;
For $j = 1$ to i do
  If $z[0] = x_j \oplus y_j$ then
    $y \leftarrow E_I(k_j[1]||x_j, \overline{k_j[0]})$;
    If $z[1] = \overline{k_j[0]} \oplus y$ then $L \xleftarrow{\cup} (x_j||k_j[0], k_j[1])$;
  If $z[1] = x_j \oplus y_j$ then
    $y \leftarrow E_I(\overline{x_j}||k_j[0], k_j[1])$;
    If $z[0] = k_j[1] \oplus y$ then $L \xleftarrow{\cup} (k_j[1]||\overline{x}_j, k_j[0])$;
If L is not an empty list then return L and otherwise return $\perp$;

If an input-output triple of the first block is defined, automatically the input of the second block is defined, and vice versa, from the definition of the compression function. For a query (z, α) to $\mathcal{E}^+$, when there is an input-output triple (k, x, y) such that $x \oplus y = z[0]$, the multi-point extractor $\mathcal{E}^+$ checks whether the output of the second block is equal to $z[1]$ or not and if this holds the multi-point extractor stores it in the return list L, and vice versa. Therefore, A must find a preimage (k, x) of z to win the 1-WPrA experiment. Thus one can straightforwardly adapt the preimage resistant advantage of the compression function (Theorem 2 in [10]). The advantage is at most $2q_P/(2^n - q_P)^2$. □

B Tandem-DM Is PrA

Tandem-DM [15] incorporates two Davies-Meyer (DM) single block length compression functions which are used side-by-side. The compression function is formally given in Definition 4.

Definition 4. *Let* $\mathsf{BC}_{2n,n} = (E, D)$ *be a block cipher. Let* $\mathsf{CF}^{\mathsf{TDM}}[\mathsf{BC}_{2n,n}] : \{0,1\}^{2n} \times \{0,1\}^n \rightarrow \{0,1\}^{2n}$ *be a compression function such that* $(G_i, H_i) = \mathsf{CF}^{\mathsf{TDM}}[\mathsf{BC}_{2n,n}](G_{i-1}||H_{i-1}, M_i)$ *where* $G_i, H_i, M_i, G_{i-1}, H_{i-1} \in \{0,1\}^n$. (G_i, H_i) *is calculated as follows:*

$$W_i = E(H_{i-1}||M_i, G_{i-1}) \tag{3}$$
$$G_i = G_{i-1} \oplus W_i \tag{4}$$
$$H_i = H_{i-1} \oplus E(M_i||W_i, H_{i-1}) \tag{5}$$

We call the procedures of 3 and 4 "first block" and the procedures of 5 "second block".

We show that the Tandem-DM compression function is PrA with $\mathcal{O}(2^n)$ security.

Theorem 5 (Tandem-DM is PrA). *Let $\mathcal{C}_{2n,n} = (E_I, D_I)$ be an ideal cipher. There exists an extractor $\mathcal{E}$ such that for any adversary A making at most q_P queries to $\mathcal{C}_{2n,n}$ and q_e extraction queries we have*

$$\mathsf{Adv}^{\mathsf{pra}}_{\mathsf{CF}^{\mathsf{TDM}},\mathcal{C}_{d,n},\mathcal{E}}(A) \leq p + \frac{2q_P q_e}{(2^n - q_P)^2}$$

where $\mathcal{E}$ runs in time at most $\mathcal{O}(q_e q_P)$ and p is the cr-advantage of Tandem-DM described in Theorem 1 of [16].

Proof. We will prove that any such compression function is 1-WPrA, and then Lemma 1 to give the final bound. We note that Theorem 1 of [9] upperbounds the cr-advantage by p, yielding the terms excluding the last term. Let us define the multi-point extractor $\mathcal{E}^+$ as follows:

algorithm $\mathcal{E}^+(z, \alpha)$
L be an empty list;
Parse $(k_1, x_1, y_1), \ldots, (k_i, x_i, y_i) \leftarrow \alpha$;
For $j = 1$ to i do
  If $z[0] = x_j \oplus y_j$ then
    $y \leftarrow E_I(k_j[1]||y_j, k_j[0])$;
    If $z[1] = k_j[0] \oplus y$ then $L \xleftarrow{\cup} (x_j||k_j[0], k_j[1])$;
  If $z[1] = x_j \oplus y_j$ then
    $x \leftarrow D_I(x_j||k_j[0], k_j[1])$;
    If $z[0] = k_j[1] \oplus x$ then $L \xleftarrow{\cup} (x||x_j, k_j[0])$;
If L is an empty list then return L otherwise return $\perp$;

If an input-output triple of the first block is defined, automatically the input triple of the second block is defined, and vice versa, from the definition of the compression function. For a query (z, α) to $\mathcal{E}^+$, when there is an input-output triple (k, x, y) such that $x \oplus y = z[0]$, the multi-point extractor $\mathcal{E}^+$ checks whether the output of the second block is equal to $z[1]$ or not and if this holds the multi-point extractor stores it in the return list L, and vice versa. Therefore, A must find a preimage (k, x) of z to win the 1-WPrA experiment. Then one can straightforwardly adapt the preimage resistant advantage of Tandem-DM (Theorem 2 in [9]). This advantage is at most $2q_P/(2^n - q_P)^2$. □