Indifferentiability of Double Length Compression Functions

Bart Mennink

Dept. Electrical Engineering, ESAT/COSIC, KU Leuven, and iMinds, Belgium
bart.mennink@esat.kuleuven.be

Abstract. Double block length hashing covers the idea of constructing a compression function on $2n$ bits using an n-bit block cipher. In this work, we present a comprehensive indifferentiability analysis of all relevant double length compression functions. Indifferentiability is a stronger security notion than collision and preimage resistance and ensures that a design has no structural flaws. It is very well suited for composition: using an indifferentiable compression function in a proper mode of operation supplies an indifferentiable hash function. Yet, as we demonstrate compression function indifferentiability is not at all a triviality: almost all double length compression functions, including Tandem-DM and Jetchev et al.'s, appear to be differentiable from a random function in 2 queries. Nevertheless, we also prove that two known functions are indifferentiable: the MDC-4 compression function (up to $2^{n/4}$ queries tight) and Mennink's function (up to $2^{n/2}$ queries tight).

Keywords. double block length; block cipher based; compression function; indifferentiability.

1 Introduction

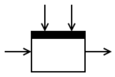
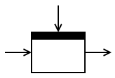
Double (block) length hashing is a well-established method for constructing a compression function with $2n$-bit output based only on n-bit block ciphers. The idea dates back to the designs of MDC-2 and MDC-4 in 1988 by Meyer and Schilling [22]. Double length hash functions have an obvious advantage over classical block cipher based functions such as Davies-Meyer and Matyas-Meyer-Oseas, and more generally the PGV class of functions [25, 28]: the same type of underlying primitive allows for a larger compression function. Yet, for double length compression functions it is harder to achieve optimal n-bit collision and $2n$-bit preimage security.

We focus on the simplest type of double length compression functions, namely those that compress $3n$ to $2n$ bits. Adopting the convention of Mennink [20], we divide the state of the art into two classes: DBL^{2n}, consisting of all functions that internally evaluate a $2n$-bit keyed block cipher $E : \{0,1\}^{2n} \times \{0,1\}^n \to \{0,1\}^n$, and DBL^n, of functions based on an n-bit keyed block cipher $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$. A classification on the collision and preimage security of the known functions is given in Table 1 (this classification is accredited to [20]). Regarding these security properties, DBL^{2n} is well-understood. For instance, the most notable Tandem-DM and Abreast-DM [11] and Hirose's function [7], that all make two underlying block cipher calls, are proven optimally secure with respect to both security notions. Hirose [6] and Özen and Stam [24] presented generalizations of these compression function designs (but for convenience all of these results are handled separately in this work). Stam introduced a single-call compression function [27, 28] (reconsidered in [16]) which is proven optimally collision secure. Lucks [17] introduced a compression function that allows for collisions in about $2^{n/2}$ queries—and is therefore not included in the classification—but achieves optimal collision resistance in the iteration. On the other hand, in the DBL^n class the first provably optimally collision secure function was presented only recently by Mennink [20]: his function is proven collision secure up to 2^n queries and preimage secure up to $2^{3n/2}$ queries. Earlier designs in this class are the MDC-2 and MDC-4 compression functions [22] and MJH [13], which are merely constructed to achieve security in the iteration,[1] and Jetchev et al.'s construction (which we

[1] In the iteration, collision resistance is proven up to $2^{3n/5}$ for MDC-2 [29] and $2^{2n/3}$ for MJH [13].

will call JOS) [9], a clever design achieving collision security up to $2^{2n/3}$ queries (with preimage security guaranteed up to 2^n queries).

Table 1. Asymptotic ideal cipher model security guarantees of known functions in the classes DBL^{2n} (first) and DBL^n (second). The collision and preimage results are taken from [20]; all indifferentiability results (in **bold**) are derived in this paper.

compression function	E-calls	collision security	preimage security	indifferentiability	underlying cipher
Stam's	1	2^n [28]	2^n [28]	**2** (Sect. 3)	
Tandem-DM	2	2^n [14]	2^{2n} [2,15]	**2** (Sect. 3)	
Abreast-DM	2	2^n [5,12]	2^{2n} [2,15]	**2** (Sect. 3)	
Hirose's	2	2^n [7]	2^{2n} [2,15]	**2** (Sect. 3)	
Hirose-class	2	2^n [6]	2^n [6]	**2** (Sect. 3)	
Özen-Stam-class	2	2^n [24]	2^n [24]	**2** (Sect. 4)	
MDC-2	2	$2^{n/2}$	2^n	**2** (Sect. 5)	
MJH	2	$2^{n/2}$	2^n	**2** (Sect. 5)	
JOS	2	$2^{2n/3}$ [9]	2^n [9]	**2** (Sect. 6)	
Mennink's	3	2^n [20]	$2^{3n/2}$ [20]	$\mathbf{2^{n/2}}$ (Sect. 7)	
MDC-4	4	$2^{5n/8}$ [21]	$2^{5n/4}$ [21]	$\mathbf{2^{n/4}}$ (Sect. 8)	

So far, these results only concern the collision and preimage security of the compression functions. If such compression function is used in a proper iteration, these carry over to the hash function design [1]. Beyond these notions, the *indifferentiability* framework of Maurer et al. [18] has gained recent attention. Indifferentiability is an important security criterion as it guarantees that a construction based on an underlying idealized primitive shows no structural flaws: generic attacks on such a design are impossible up to the proven bound, and weaknesses, if any, come from the underlying primitive. It is well suited for composition: a hash function indifferentiability result (based on an underlying compression function) and a compression function indifferentiability result (based on, say, a block cipher) compose to security of the hash function based on the ideality of the block cipher. Several hash function indifferentiability results exist [3,4,8] and compression functions are usually easier to analyze than hash functions, and therefore it is of interest to study the indifferentiability of compression functions.

But, returning to block cipher based compression functions, the state of affairs is entirely topsy-turvy when it comes to indifferentiability. First of all, as for single block length compression functions, the PGV functions are known to be differentiable from random functions [10]. As a first contribution of this work, we show that this problematic situation also applies to double length functions: all functions in the DBL^{2n} class, as well as MDC-2, MJH, and JOS (in the DBL^n class), are trivially differentiable from a random function in 2 queries. The attacks show similarities with the differentiability attacks on the PGV functions. In general, indifferentiability appears to be much harder to achieve then "simply" collision and preimage security.

However, on the positive side, we derive non-trivial indifferentiability results for Mennink's and the MDC-4 compression function. Starting with Mennink's compression function class, called F^3_{A} (see Fig. 2). These functions make three block ciphers calls and are indexed by a 4×4 matrix A that is required to comply with certain simple conditions. We prove that any F^3_{A} meeting these conditions is indifferentiable from a random function in about $2^{n/2}$ queries (tight). This bound is worse than the collision and preimage bounds, but this is as expected, given the negative indifferentiability results so far. The proof crucially relies on two key characteristics of F^3_{A}: that any two block cipher evaluations of F^3_{A} define the inputs to the third one, but more importantly, that at least two such calls *are needed* to learn something about an F^3_{A} evaluation. In general, the proof is made possible by the sequential block cipher evaluation of the design.

Next, for the MDC-4 compression function (see Fig. 5) based on two distinct block ciphers, we prove it indifferentiable from a random function up to $2^{n/4}$ queries (tight).[2] The proof is very similar to the one of Mennink's function, and in particular also crucially relies on the sequential block cipher evaluation.

All indifferentiability results are summarized in Table 1, in which we also mention the corresponding section of this paper. The work is concluded in Sect. 9.

2 Indifferentiability

The indifferentiability framework, introduced by Maurer et al. [18], is a security notion that formally captures the "distance" between a cryptographic construction and its random equivalent. Informally, it gives a sufficient condition under which an ideal primitive $\mathcal{R}$ can be replaced by some construction $\mathcal{C}^{\mathcal{P}}$ using an ideal subcomponent $\mathcal{P}$. In this paper, we employ the adaption and simplification by Coron et al. [4]. Recent results by Ristenpart et al. [26] show that indifferentiability does not capture all properties of a random oracle, it applies to single stage games only. Nevertheless, this notion captures pretty many games and remains the best way to prove that a hash or compression function behaves like a random oracle.

Definition 1. *Let $\mathcal{C}$ be a cryptographic primitive with oracle access to an ideal primitive $\mathcal{P}$. Let $\mathcal{R}$ be an ideal primitive with the same domain and range as $\mathcal{C}$. Let $\mathcal{S}$ be a simulator with the same domain and range as $\mathcal{P}$ with oracle access to $\mathcal{R}$ and making at most $q_{\mathcal{S}}$ queries, and let $\mathcal{D}$ be a distinguisher making at most $q_{\mathcal{D}}$ queries. The differentiability advantage of $\mathcal{D}$ is defined as*

$$\mathbf{adv}^{\mathrm{iff}}_{\mathcal{C},\mathcal{S}}(\mathcal{D}) = \left|\mathbf{Pr}\left(\mathcal{D}^{\mathcal{C},\mathcal{P}} = 1\right) - \mathbf{Pr}\left(\mathcal{D}^{\mathcal{R},\mathcal{S}} = 1\right)\right| .$$

We refer to $(\mathcal{C},\mathcal{P})$ as the real world, and to $(\mathcal{R},\mathcal{S})$ as the simulated world. We denote $\mathcal{D}$'s left oracle ($\mathcal{C}$ or $\mathcal{R}$) by L and its right oracle ($\mathcal{P}$ or $\mathcal{S}$) by R.

For $k, n \geq 1$, we denote by $\mathrm{Bloc}(k, n)$ the set of all block ciphers with a k-bit key and n-bit message space. We simply write $\mathrm{Bloc}(n)$ if $k = n$. Throughout, $\mathcal{C} : \{0,1\}^{3n} \to \{0,1\}^{2n}$ corresponds to a compression function design, and $\mathcal{P}$ represents block ciphers from $\mathrm{Bloc}(k, n)$ (where $k = 2n$ for functions in DBL^{2n} and $k = n$ for functions in DBL^{n}). We stress that some of the designs analyzed in this work are defined to make use of two distinct block ciphers (e.g., one call to a cipher E_1 and one call to E_2). Except for our indifferentiability result on MDC-4, for all of our results it is not relevant whether the underlying ciphers are distinct or the same. Therefore, we consider all designs simply to be based on one single block cipher, unless stated otherwise.

3 Stam's, Tandem-DM, Abreast-DM, Hirose's, and Hirose-class

In this section, we consider Tandem-DM and Abreast-DM [11] (cf. Fig. 1), Hirose's compression function [7] (cf. Fig. 1) and its generalized Hirose-class [6],[3] as well as Stam's supercharged

[2] The MDC-4 compression function based on one single block cipher is differentiable in 2 queries.

[3] Hirose's function can be seen as a special case of Hirose-class (using that in the attack it is not relevant whether the underlying block ciphers are distinct or the same), and our attack directly carries over.

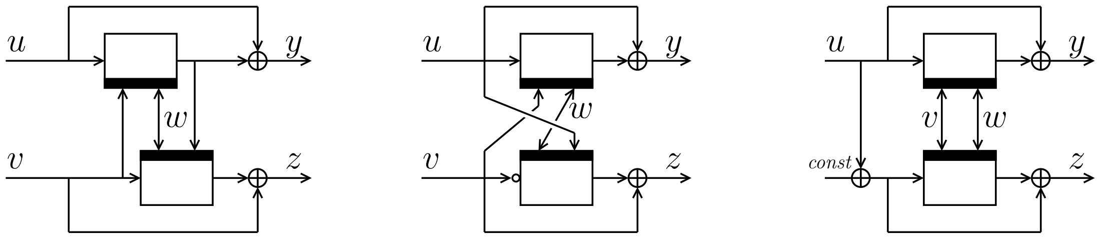


Fig. 1. Tandem-DM (left), Abreast-DM (middle), and Hirose's compression function (right) [7, 11]. All wires carry n bits. For Abreast-DM, the circle $\circ$ denotes bit complementation. For Hirose's function, *const* is any non-zero constant.

single call Type-I compression function design [27, 28], or more specifically the block cipher based variant considered in [16]:

$$\begin{aligned}&\mathrm{Stam}(u,v,w)=(y,z),\text{ where:}\\&\quad c_1\leftarrow E(v\|w,u)\,,\\&\quad y\leftarrow c_1+u\,,\\&\quad z\leftarrow wy^2+vy+u\,.\end{aligned}$$

Here, additions and finite field multiplications are done over the field $GF(2^n)$. The differentiability attacks are identical for all designs, and we only consider Tandem-DM (abbreviated to TDM). The attack is a direct generalization of the fixed-point attack on the Davies-Meyer (DM) compression function.

We note that Özen and Stam presented a generalized double length design [24], and our attack on their class (in Sect. 4) can be seen as a true generalization of the attacks in this section on Abreast-DM and Hirose's functions (given that in these attacks it is not relevant whether the underlying block ciphers are distinct or the same). Nevertheless, these functions are handled separately for clarity and as an illustration.

Proposition 1. *Let $E \xleftarrow{\$} Bloc(2n,n)$, and let $\mathcal{R}:\{0,1\}^{3n}\to\{0,1\}^{2n}$ be a random compression function. For any simulator $\mathcal{S}$ that makes at most $q_\mathcal{S}$ queries to $\mathcal{R}$, there exists a distinguisher $\mathcal{D}$ that makes 2 queries to its oracles, such that*

$$\mathbf{adv}^{\mathrm{iff}}_{\mathrm{TDM},\mathcal{S}}(\mathcal{D})\geq 1-\frac{q_\mathcal{S}+1}{2^n}\,.$$

Proof. Our distinguisher $\mathcal{D}$ aims at finding an evaluation of TDM that satisfies:

$$\mathrm{TDM}(u,v,w)=(u,z)\,,\tag{1}$$

for some values u,v,w,z. $\mathcal{D}$ operates as follows. First, it fixes some values v,w, and queries $u\leftarrow R^{-1}(v\|w,0)$. Next, it queries its left oracle L on input of (u,v,w), and outputs 0 if and only if the first half of the response equals u (hence if (1) is satisfied). Clearly, in the real world, (1) holds with certainty, and $\mathcal{D}$ succeeds except if $\mathcal{S}$ or $\mathcal{D}$ obtains a solution to $\mathcal{R}(u,v,w)=(u,z)$. As $\mathcal{R}$ is a random function, any query satisfies this equation with probability $\frac{1}{2^n}$, and $\mathcal{R}$ is consulted at most $q_\mathcal{S}+1$ times. This completes the proof. □

4 Özen-Stam-class

Özen and Stam [24] analyzed a wide class of double length compression functions, extending the single-length compression function result of Stam [28].

$$
\begin{aligned}
&\mathrm{OS}(u,v,w) = (y,z), \text{ where:}\\
&\quad (k_1,m_1) \leftarrow C_1^{\mathrm{pre}}(u,v,w)\,,\\
&\quad c_1 \leftarrow E(k_1,m_1)\,,\\
&\quad (k_2,m_2) \leftarrow C_2^{\mathrm{pre}}(u,v,w)\,,\\
&\quad c_2 \leftarrow E(k_2,m_2)\,,\\
&\quad (y,z) \leftarrow C^{\mathrm{post}}(u,v,w,c_1,c_2)\,.
\end{aligned}
$$

Here, it is required that C_1^{pre} and C_2^{pre} are bijections, as is $C^{\mathrm{post}}(u,v,w,\cdot,\cdot)$ for fixed (u,v,w). Additionally, certain requirements are posed on C_1^{aux} and C_2^{aux} (combinations of the three functions), but these are not relevant for our analysis.

We assume the existence of a bijection $M : \{0,1\}^{2n} \to \{0,1\}^{2n}$ such that the left half of $M \circ C^{\mathrm{post}}(u,v,w,c_1,c_2)$ is independent of c_2, and consider the compression function design with M appended. (Note that this does not affect the security result.) For convenience, we simply assume the existence of C_1^{post} and C_2^{post} such that

$$
\begin{aligned}
&y \leftarrow C_1^{\mathrm{post}}(u,v,w,c_1)\,,\\
&z \leftarrow C_2^{\mathrm{post}}(u,v,w,c_1,c_2)\,.
\end{aligned}
$$

Proposition 2. *Let $E \xleftarrow{\$} Bloc(2n,n)$, and let $\mathcal{R} : \{0,1\}^{3n} \to \{0,1\}^{2n}$ be a random compression function. For any simulator $\mathcal{S}$ that makes at most $q_{\mathcal{S}}$ queries to $\mathcal{R}$, there exists a distinguisher $\mathcal{D}$ that makes 2 queries to its oracles, such that*

$$\mathbf{adv}^{\mathrm{iff}}_{\mathrm{OS},\mathcal{S}}(\mathcal{D}) \geq 1 - \frac{q_{\mathcal{S}}+1}{2^n}\,.$$

Proof. The proof is similar to the one of Prop. 1, and we only highlight the differences. Our distinguisher $\mathcal{D}$ aims at finding an evaluation of OS that satisfies:

$$\mathrm{OS}(u,v,w) = (C_1^{\mathrm{post}}(u,v,w,0),z)\,, \tag{2}$$

for some values u,v,w,z. First, the adversary fixes k_1, and queries $m_1 \leftarrow R^{-1}(k_1,0)$. Then, it computes $(u,v,w) \leftarrow C_1^{-\mathrm{pre}}(k_1,m_1)$. Next, it queries its left oracle L on input of (u,v,w), and outputs 0 if and only if (2) is satisfied. The remainder of the analysis is the same as in the proof of Prop. 1. □

5 MDC-2 and MJH

In this section, we consider the MDC-2 and MJH compression functions. For MDC-2, we leave out the swapping at the end as it is of no influence to the indifferentiability proof. The functions are defined as follows (for MJH, σ is an involution and θ a constant):

$$
\begin{aligned}
&\text{MDC-2}(u,v,w) = (y,z), \text{ where:}\\
&\quad c_1 \leftarrow E(u,w)\,,\\
&\quad y \leftarrow c_1 + w\,,\\
&\quad c_2 \leftarrow E(v,w)\,,\\
&\quad z \leftarrow c_2 + w\,.
\end{aligned}
\qquad
\begin{aligned}
&\mathrm{MJH}(u,v,w) = (y,z), \text{ where:}\\
&\quad c_1 \leftarrow E(v,u+w)\,,\\
&\quad y \leftarrow c_1 + u + w\,,\\
&\quad c_2 \leftarrow E(v,\sigma(u+w))\,,\\
&\quad z \leftarrow (c_2 + \sigma(u+w)) \cdot \theta + u\,.
\end{aligned}
$$

Recall that for our results, it is not relevant whether the underlying ciphers are distinct or the same.

Proposition 3. *Let $E \xleftarrow{\$} Bloc(n)$, and let $\mathcal{R} : \{0,1\}^{3n} \to \{0,1\}^{2n}$ be a random compression function. For any simulator $\mathcal{S}$ that makes at most $q_\mathcal{S}$ queries to $\mathcal{R}$, there exists a distinguisher $\mathcal{D}$ that makes* 2 *queries to its oracles, such that*

$$\mathbf{adv}^{\mathrm{iff}}_{\text{MDC-2},\mathcal{S}}(\mathcal{D}) \geq 1 - \frac{q_\mathcal{S}+1}{2^n} .$$

The same result holds for MJH.

Proof. The proof is similar to the one of Prop. 1. Now, our distinguisher aims at finding an evaluation of MDC-2 that satisfies MDC-2$(u,v,w) = (w,z)$, and the same for MJH. The remainder of the analysis is almost identical to the proof of Prop. 1, and therefore omitted. □

6 JOS

In this section, we consider Jetchev et al.'s compression function (called JOS). The analysis is slightly more complicated but in fact not much different. We consider the block cipher based variant with the underlying matrix A as suggested in [23, Sect. 5.4.2].

$$\begin{aligned} &\mathrm{JOS}(u,v,w) = (y,z), \text{ where:}\\ &\quad c_1 \leftarrow E(w,u)\,,\\ &\quad c_2 \leftarrow E(w+uv,v)\,,\\ &\quad y \leftarrow u+v+(u+c_1)(v+c_2)\,,\\ &\quad z \leftarrow u+v+c_1+c_2\,. \end{aligned}$$

Here, additions and finite field multiplications are done over the field $GF(2^n)$.

Proposition 4. *Let $E \xleftarrow{\$} Bloc(n)$, and let $\mathcal{R} : \{0,1\}^{3n} \to \{0,1\}^{2n}$ be a random compression function. For any simulator $\mathcal{S}$ that makes at most $q_\mathcal{S}$ queries to $\mathcal{R}$, there exists a distinguisher $\mathcal{D}$ that makes* 2 *queries to its oracles, such that*

$$\mathbf{adv}^{\mathrm{iff}}_{\mathrm{JOS},\mathcal{S}}(\mathcal{D}) \geq 1 - \frac{q_\mathcal{S}+1}{2^n} .$$

Proof. The proof is similar to the one of Prop. 1. Now, our distinguisher aims at finding an evaluation of JOS$(u,v,w) = (y,z)$ that satisfies $y + uz = u^2 + u + v$. The remainder of the analysis is almost identical to the proof of Prop. 1, and therefore omitted. □

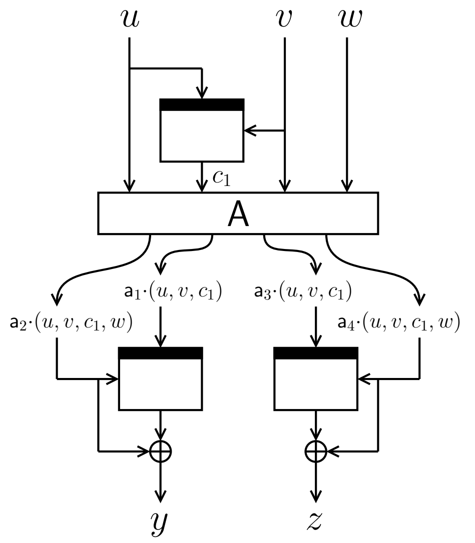
7 Mennink's

Mennink's double length compression function design, dubbed $F^3_{\mathsf{A}} : \{0,1\}^{3n} \to \{0,1\}^{2n}$ (depicted in Fig. 2), makes three calls to a single block cipher, and is indexed by a 4×4 matrix

$$\mathsf{A} = \begin{pmatrix} \mathsf{a}_1 \\ \mathsf{a}_2 \\ \mathsf{a}_3 \\ \mathsf{a}_4 \end{pmatrix} = \begin{pmatrix} \mathsf{a}_{11} & \mathsf{a}_{12} & \mathsf{a}_{13} & 0 \\ \mathsf{a}_{21} & \mathsf{a}_{22} & \mathsf{a}_{23} & \mathsf{a}_{24} \\ \mathsf{a}_{31} & \mathsf{a}_{32} & \mathsf{a}_{33} & 0 \\ \mathsf{a}_{41} & \mathsf{a}_{42} & \mathsf{a}_{43} & \mathsf{a}_{44} \end{pmatrix} \tag{3}$$

over the field $GF(2^n)$.[4]

[4] Bit strings from $\{0,1\}^n$ and finite field elements in $GF(2^n)$ are identified to define addition and scalar multiplication over $\{0,1\}^n$. For two tuples $x = (x_1, \dots, x_l)$ and $y = (y_1, \dots, y_l)$ of elements from $\{0,1\}^n$, $x \cdot y$ denotes inner product $\sum_{i=1}^{l} x_i y_i \in \{0,1\}^n$.


u
v
w
c_1
A
$\mathsf{a}_1\cdot(u,v,c_1)$
$\mathsf{a}_3\cdot(u,v,c_1)$
$\mathsf{a}_2\cdot(u,v,c_1,w)$
$\mathsf{a}_4\cdot(u,v,c_1,w)$
y
z

$F^3_{\mathsf{A}}(u,v,w)=(y,z)$, where:

$$\begin{aligned}
&c_1 \leftarrow E(u,v)\,,\\
&k_2 \leftarrow \mathsf{a}_1\cdot(u,v,c_1)\,,\\
&m_2 \leftarrow \mathsf{a}_2\cdot(u,v,c_1,w)\,,\\
&y \leftarrow E(k_2,m_2)+m_2\,,\\
&k_3 \leftarrow \mathsf{a}_3\cdot(u,v,c_1)\,,\\
&m_3 \leftarrow \mathsf{a}_4\cdot(u,v,c_1,w)\,,\\
&z \leftarrow E(k_3,m_3)+m_3\,.
\end{aligned}$$

Fig. 2. Mennink's compression function class F^3_{A} where A is a 4×4 matrix as in (3).

The security of F^3_{A} is based on the key principle that any *two* block cipher evaluations define the input to the third one. Indeed, invertibility of A guarantees that evaluations of the second and third block cipher define the values (u,v,c_1). Also, if $\mathsf{a}_{24}\neq 0$ (resp. $\mathsf{a}_{44}\neq 0$), the first and second (resp. third) block cipher define the inputs to the third (resp. second) one. However, in order to achieve collision and preimage security, Mennink posed a slightly larger set of conditions on A, which he called `colreq` (for collision security) and `prereq` (for preimage security). For the indifferentiability results, it suffices to pose a much weaker condition on A. In detail, we require the following from A (called `indreq`): A is invertible and $\mathsf{a}_{12},\mathsf{a}_{13},\mathsf{a}_{24},\mathsf{a}_{32},\mathsf{a}_{33},\mathsf{a}_{44}\neq 0$. As `prereq` $\Rightarrow$ `colreq` $\Rightarrow$ `indreq`, our results particularly apply to *all* schemes proven secure in [20].

Suiting the analysis, we define a function get$w$ that, on input of $j\in\{2,4\}$, $m\in\{0,1\}^n$, and $(k_1,m_1,c_1)\in\{0,1\}^{3n}$, outputs w such that $\mathsf{a}_j\cdot(k_1,m_1,c_1,w)=m$. Note that $\mathsf{a}_{24},\mathsf{a}_{44}\neq 0$ implies uniqueness of w. Differentiability is discussed in Sect. 7.1, and indifferentiability in Sect. 7.2.

7.1 Differentiability

In Prop. 5 we show that F^3_{A} is differentiable from a random oracle in at most about $2^{n/2}$ queries.

Proposition 5. *Let $E \xleftarrow{\$} \mathit{Bloc}(n)$, and let $\mathcal{R}:\{0,1\}^{3n}\to\{0,1\}^{2n}$ be a random compression function. For any simulator $\mathcal{S}$ that makes at most $q_{\mathcal{S}}$ queries to $\mathcal{R}$, there exists a distinguisher $\mathcal{D}$ that makes $2^{n/2}+2$ queries to its oracles, such that*

$$\mathbf{adv}^{\mathrm{iff}}_{F^3_{\mathsf{A}},\mathcal{S}}(\mathcal{D}) \geq \frac{1}{2}-\frac{1}{2^{n/2+1}}-\frac{q_{\mathcal{S}}+1}{2^n-q_{\mathcal{S}}}\,.$$

Proof. Our distinguisher $\mathcal{D}$ aims at finding two different evaluations of F^3_{A} with the same key inputs to the second (or third) block cipher call. In more detail, the distinguisher aims at finding two distinct block cipher calls (k_1,m_1,c_1) and (k_1',m_1',c_1') such that for $j\in\{1,3\}$:

$$\mathsf{a}_j\cdot(k_1,m_1,c_1)=\mathsf{a}_j\cdot(k_1',m_1',c_1')\,. \tag{4}$$

Note that in the real world, for F^3_{A}, such collisions are expected to be found in about $2^{n/2}$ queries to E (here we use that $\mathsf{a}_{12}, \mathsf{a}_{13}, \mathsf{a}_{32}, \mathsf{a}_{33} \neq 0$). If the distinguisher eventually finds a collision as in (4), then for any $m \in \{0,1\}^n$, the following condition naturally holds in the real world:

$$y = y' \text{ if } j = 1 \text{ and } z = z' \text{ if } j = 3\,, \tag{5}$$

where

$$\begin{aligned}(y, z) &= F^3_{\mathsf{A}}(k_1, m_1, \mathrm{get}w(j+1, m, k_1, m_1, c_1))\,,\\ (y', z') &= F^3_{\mathsf{A}}(k'_1, m'_1, \mathrm{get}w(j+1, m, k'_1, m'_1, c'_1))\,.\end{aligned}$$

In the random world, with F^3_{A} replaced by $\mathcal{R}$, this equation only holds with small probability. Note that the simulator never learns the value m, yet, it may simply try to avoid collisions as in (4). However, in this case, the responses from $\mathcal{S}$ are too biased, which allows the distinguisher to succeed.

Formally, the distinguisher $\mathcal{D}$ proceeds as follows.

(i) $\mathcal{D}$ makes $2^{n/2}$ queries to its right oracle R for different key and different message values, obtaining $2^{n/2}$ distinct tuples (k_1, m_1, c_1);
(ii) If there is no solution to (4), $\mathcal{D}$ returns 1;
(iii) Let $j \in \{1, 3\}$ and (k_1, m_1, c_1) and (k'_1, m'_1, c'_1) be such that (4) is satisfied;
(iv) Take $m \xleftarrow{\$} \{0,1\}^n$. If (5) holds, $\mathcal{D}$ returns 0, and otherwise it returns 1.

Distinguisher $\mathcal{D}$ succeeds except in the following two cases: "C_1" it is conversing with the real world and (4) does not have a solution (which means that his guess in step (ii) is wrong), or "C_2" it is conversing with the simulated world and (5) holds (which means that his guess in step (iv) is wrong). Therefore, $\mathbf{adv}^{\mathrm{iff}}_{F^3_{\mathsf{A}},\mathcal{S}}(\mathcal{D}) \geq 1 - \mathbf{Pr}(\mathsf{C}_1) - \mathbf{Pr}(\mathsf{C}_2)$. Regarding C_1: note that all queries are made with different key inputs, and E is a random cipher. Therefore, all responses are randomly drawn from a set of size 2^n, and a collision (4) occurs with probability at least $\binom{2^{n/2}}{2}\frac{1}{2^n}$ (as $\mathsf{a}_{12}, \mathsf{a}_{13}, \mathsf{a}_{32}, \mathsf{a}_{33} \neq 0$). Thus,

$$\mathbf{Pr}(\mathsf{C}_1) \leq 1 - \binom{2^{n/2}}{2}\frac{1}{2^n} = \frac{1}{2} + \frac{1}{2^{n/2+1}}\,.$$

Regarding C_2, denote by E the event that $\mathcal{S}$ ever queries $\mathcal{R}(k_1, m_1, \mathrm{get}w(j+1, m, k_1, m_1, c_1))$. Then,

$$\mathbf{Pr}(\mathsf{C}_2) \leq \mathbf{Pr}(\mathsf{C}_2 \mid \neg\mathsf{E}) + \mathbf{Pr}(\mathsf{E}) \leq \frac{1}{2^n - q_{\mathcal{S}}} + \frac{q_{\mathcal{S}}}{2^n - (q_{\mathcal{S}} - 1)} = \frac{q_{\mathcal{S}} + 1}{2^n - q_{\mathcal{S}}}\,,$$

where we use that $\mathsf{a}_{24}, \mathsf{a}_{44} \neq 0$. This completes the proof. □

7.2 Indifferentiability

We prove that F^3_{A} is indifferentiable from a random function.

Theorem 1. *Let $E \xleftarrow{\$} Bloc(n)$, and let $\mathcal{R} : \{0,1\}^{3n} \to \{0,1\}^{2n}$ be a random function. There exists a simulator $\mathcal{S}$ such that for any distinguisher $\mathcal{D}$ that makes at most q_L left queries and q_R right queries,*

$$\mathbf{adv}^{\mathrm{iff}}_{F^3_{\mathsf{A}},\mathcal{S}}(\mathcal{D}) \leq \frac{7(3q_L + q_R)^2}{2^n}\,,$$

where $\mathcal{S}$ makes $q_{\mathcal{S}} \leq q_R$ queries to $\mathcal{R}$.

The simulator $\mathcal{S}$ used in the proof mimics the behavior of random cipher E such that queries to $\mathcal{S}$ and queries to $\mathcal{R}$ are consistent, which means that relations among the query outputs in the real world hold in the simulated world as well. In the remainder of the section, we first introduce our simulator and accommodate it with an intuition, and next present the formal proof.

Simulator Intuition

For $k \in \{0,1\}^n$, the simulator maintains an initially empty list $\mathcal{L}E[k]$. In this list, it stores tuples (m, c) such that $\mathcal{S}(k, m) = c$. We write $\mathcal{L}E^+[k]$ for all input values m and $\mathcal{L}E^-[k]$ for all output values c. Sometimes, we abuse notation and write $(k, m, c) \in \mathcal{L}E$ to denote that $(m, c) \in \mathcal{L}E[k]$.

Mennink's F^3_{A} class of functions is based on the key principle that any two block ciphers define the inputs to the third one. The simulator we use for the proof of Thm. 1 enormously benefits from some of these characteristics. In more detail, the simulator is given in Fig. 3.

Apart from the **if**-clause of lines 02-06, the simulator identically mimics an ideal cipher. In this particular clause, the simulator checks whether a query (k, m) may appear in an F^3_{A} evaluation (see Fig. 2) as a bottom query (left or right) for some other query appearing in the top. In more detail, this happens if $(k, m) = (\mathsf{a}_j \cdot (k_1, m_1, c_1), \mathsf{a}_{j+1} \cdot (k_1, m_1, c_1, w))$ for some $j \in \{1, 3\}$ and some earlier query $(k_1, m_1, c_1) \in \mathcal{L}E$. In this case, the simulator should consult $\mathcal{R}$ to derive the query response. At a higher level, the simulator is based on the idea that, with high probability, a distinguisher can only compare (F^3_{A}, E) and $(\mathcal{R}, \mathcal{S})$ if it makes the queries to $E/\mathcal{S}$ "in correct order": for any evaluation of F^3_{A} that can be derived from $\mathcal{L}E$, the top query is made prior to the two bottom queries.

```
Forward Query $\mathcal{S}(k, m)$
00 if $\mathcal{L}E^+[k](m) \neq \perp$ return $c = \mathcal{L}E^+[k](m)$
01 $c \overset{\$}{\leftarrow} \{0,1\}^n \backslash \mathcal{L}E^+[k]$
02 if $\exists\, j \in \{1,3\}, (k_1, m_1, c_1) \in \mathcal{L}E :\ k = \mathsf{a}_j \cdot (k_1, m_1, c_1)$
03     $w \leftarrow \mathrm{get}w(j+1, m, k_1, m_1, c_1)$
04     $(y, z) \leftarrow \mathcal{R}(k_1, m_1, w)$
05     $c \leftarrow m + (y[j=1] + z[j=3])$
06 end if
07 return $\mathcal{L}E^+[k](m) \leftarrow c$
```

```
Inverse Query $\mathcal{S}^{-1}(k, c)$
10 if $\mathcal{L}E^-[k](c) \neq \perp$ return $m = \mathcal{L}E^-[k](c)$
11 $m \overset{\$}{\leftarrow} \{0,1\}^n \backslash \mathcal{L}E^-[k]$
12 return $\mathcal{L}E^-[k](c) \leftarrow m$
```

Fig. 3. The simulator $\mathcal{S}$ for E used in the proof of Thm. 1.

Proof of Theorem 1

We formally proof Thm. 1. Let $\mathcal{S}$ be the simulator of Fig. 3, and let $\mathcal{D}$ be any distinguisher that makes at most q_L left queries and q_R right queries. Note that $\mathcal{S}$ makes $q_{\mathcal{S}} \leq q_R$ queries. By Def. 1, the goal is to bound:

$$\mathbf{adv}^{\mathrm{iff}}_{F^3_{\mathsf{A}}, \mathcal{S}}(\mathcal{D}) = \left| \mathbf{Pr}\left(\mathcal{D}^{F^3_{\mathsf{A}}, E} = 1\right) - \mathbf{Pr}\left(\mathcal{D}^{\mathcal{R}, \mathcal{S}} = 1\right) \right| . \tag{6}$$

As a first step, we apply a PRP-PRF switch to both worlds. More formally, we define $\widetilde{E}$ as E with the difference that all responses are randomly drawn from $\{0,1\}^n$. Similarly, $\widetilde{\mathcal{S}}$ is defined as $\mathcal{S}$ of Fig. 3 with the difference that random sampling from $\{0,1\}^n$ is done in lines 01 and 11. Now,

$$\left| \mathbf{Pr}\left(\mathcal{D}^{F^3_{\mathsf{A}}, E} = 1\right) - \mathbf{Pr}\left(\mathcal{D}^{F^3_{\mathsf{A}}, \widetilde{E}} = 1\right) \right| \leq \frac{(3q_L + q_R)^2}{2^{n+1}} ,$$

and

$$\left|\mathbf{Pr}\left(\mathcal{D}^{\mathcal{R},\widetilde{\mathcal{S}}}=1\right)-\mathbf{Pr}\left(\mathcal{D}^{\mathcal{R},\mathcal{S}}=1\right)\right|\leq\frac{q_R^2}{2^{n+1}}\,,$$

and we obtain for (6):[5]

$$\mathbf{adv}^{\mathrm{iff}}_{F^3_{\mathsf{A}},\mathcal{S}}(\mathcal{D})\leq\left|\mathbf{Pr}\left(\mathcal{D}^{F^3_{\mathsf{A}},\widetilde{E}}=1\right)-\mathbf{Pr}\left(\mathcal{D}^{\mathcal{R},\widetilde{\mathcal{S}}}=1\right)\right|+\frac{(3q_L+q_R)^2}{2^n}\,. \tag{7}$$

It remains to analyze the probability of $\mathcal{D}$ to distinguish $(F^3_{\mathsf{A}},\widetilde{E})$ from $(\mathcal{R},\widetilde{\mathcal{S}})$. Abusing notation, we remain calling these worlds the real and simulated world. These worlds are described in Fig. 4. Here, in both worlds, $\mathcal{L}E$ represents an initially empty list of all right oracle queries, and in the simulated world only we furthermore use $\mathcal{LR}$ as an initially empty list of all left oracle queries.

Let event $\mathsf{cond}(\mathcal{L}E)$ be defined as follows:

$$\mathsf{cond}(\mathcal{L}E)=\left(\begin{array}{c}\exists\ j,j'\in\{1,3\},(k,m,c),(k',m',c')\in\mathcal{L}E:\\(k,m,c)\text{ newer than }(k',m',c')\text{ and}\\\mathsf{a}_j\cdot(k,m,c)\in\{k,k',\mathsf{a}_{j'}\cdot(k',m',c')\}\end{array}\right). \tag{8}$$

Event $\mathsf{cond}(\mathcal{L}E)$ covers the case of two distinct top queries that result to the same key input to two bottom queries, as well as the case of a top query accidentally hitting the key k' of a bottom query (which may be the equal to the top query). Particularly, as long as $\neg\mathsf{cond}(\mathcal{L}E)$, the condition in line 42 of Fig. 4 is always satisfied by at most one $(j,(k_1,m_1,c_1))$. In the remainder, we prove in Lem. 1 that $(F^3_{\mathsf{A}},\widetilde{E})$ and $(\mathcal{R},\widetilde{\mathcal{S}})$ are perfectly indistinguishable as long as $\mathsf{cond}(\mathcal{L}E)$ does not occur in both worlds. Then, in Lem. 2 we prove that $\mathsf{cond}(\mathcal{L}E)$ occurs in the real world with probability at most $\frac{3(3q_L+q_R)^2}{2^n}$ and in the simulated world with probability at most $\frac{3q_R^2}{2^n}$. Together with (7), this completes the proof.

```
Query $F^3_{\mathsf{A}}(u,v,w)$
00 $c_1 \leftarrow \widetilde{E}(u,v)$
01 $k_2 \leftarrow \mathsf{a}_1\cdot(u,v,c_1)$
02 $m_2 \leftarrow \mathsf{a}_2\cdot(u,v,c_1,w)$
03 $y \leftarrow \widetilde{E}(k_2,m_2)+m_2$
04 $k_3 \leftarrow \mathsf{a}_3\cdot(u,v,c_1)$
05 $m_3 \leftarrow \mathsf{a}_4\cdot(u,v,c_1,w)$
06 $z \leftarrow \widetilde{E}(k_3,m_3)+m_3$
07 return $(y,z)$
```

```
Query $\widetilde{E}(k,m)$
10 if $\mathcal{L}E^+[k](m)\neq\perp$ return $c=\mathcal{L}E^+[k](m)$
11 $c \xleftarrow{\$} \{0,1\}^n$
12 return $\mathcal{L}E^+[k](m)\leftarrow c$
```

```
Query $\widetilde{E}^{-1}(k,c)$
20 if $\mathcal{L}E^-[k](c)\neq\perp$ return $m=\mathcal{L}E^-[k](c)$
21 $m \xleftarrow{\$} \{0,1\}^n$
22 return $\mathcal{L}E^-[k](c)\leftarrow m$
```

```
Query $\mathcal{R}(u,v,w)$
30 if $\mathcal{LR}(u,v,w)\neq\perp$ return $(y,z)=\mathcal{LR}(u,v,w)$
31 $(y,z) \xleftarrow{\$} \{0,1\}^{2n}$
32 return $\mathcal{LR}(u,v,w)\leftarrow(y,z)$
```

```
Query $\widetilde{\mathcal{S}}(k,m)$
40 if $\mathcal{L}E^+[k](m)\neq\perp$ return $c=\mathcal{L}E^+[k](m)$
41 $c \xleftarrow{\$} \{0,1\}^n$
42 if $\exists\ j\in\{1,3\},(k_1,m_1,c_1)\in\mathcal{L}E:\ k=\mathsf{a}_j\cdot(k_1,m_1,c_1)$
43     $w\leftarrow \mathrm{get}w(j+1,m,k_1,m_1,c_1)$
44     $(y,z)\leftarrow\mathcal{R}(k_1,m_1,w)$
45     $c\leftarrow m+(y[j=1]+z[j=3])$
46 end if
47 return $\mathcal{L}E^+[k](m)\leftarrow c$
```

```
Query $\widetilde{\mathcal{S}}^{-1}(k,c)$
50 if $\mathcal{L}E^-[k](c)\neq\perp$ return $m=\mathcal{L}E^-[k](c)$
51 $m \xleftarrow{\$} \{0,1\}^n$
52 return $\mathcal{L}E^-[k](c)\leftarrow m$
```

Fig. 4. The worlds $(F^3_{\mathsf{A}},\widetilde{E})$ (left) and $(\mathcal{R},\widetilde{\mathcal{S}})$ (right).

[5] Technically, we could have taken $\widetilde{\mathcal{S}}$ as our simulator, therewith obtaining an improved indifferentiability bound for Thm. 1. However, for clarity and ease of presentation, we opted for simulator $\mathcal{S}$.

Lemma 1. *As long as* $\neg\mathsf{cond}(\mathcal{L}E)$, $(F^3_{\mathsf{A}}, \widetilde{E})$ *from* $(\mathcal{R}, \widetilde{\mathcal{S}})$ *are perfectly indistinguishable.*

Proof. We consider any query made by the distinguisher, either to the left oracle L (either F^3_{A} or $\mathcal{R}$) and the right oracle R/R^{-1} (either $\widetilde{E}/\widetilde{E}^{-1}$ or $\widetilde{\mathcal{S}}/\widetilde{\mathcal{S}}^{-1}$), and show that the query responses are equally distributed in both worlds (irrespectively of the query history). Without loss of generality, we consider new queries only: if the distinguisher makes a repetitive query, the answer is known and identically distributed in both worlds.

***L*-query $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$.** We make the following distinction:

1. $\mathcal{L}E^{+}[u](v) = \bot$. In the real world, this means that the first cipher call $\widetilde{E}(u, v)$ is new, and answered with a fresh value. As $\mathsf{cond}(\mathcal{L}E)$ does not occur, also the second *and* third call, $\widetilde{E}(k_2, m_2)$ and $\widetilde{E}(k_3, m_3)$, are fresh, and both their responses are drawn from $\{0,1\}^n$. Regarding the simulated world, by the condition "$\mathcal{L}E^{+}[u](v) = \bot$," $\widetilde{\mathcal{S}}$ has never queried $\mathcal{R}$ on input of (u, v, w). Indeed, it had only queried $\mathcal{R}$ if the condition of line 42 was satisfied for some $j \in \{1, 3\}$ and *existing* $(u, v, c_1) \in \mathcal{L}E$. Thus, also in this world the response is randomly generated from $\{0,1\}^{2n}$;
2. $\mathcal{L}E^{+}[u](v) \neq \bot$. Note that in the real world, this element could have been added to $\mathcal{L}E$ via $\mathcal{D}$ or via F^3_{A}. Let $c_1 = \mathcal{L}E^{+}[u](v)$, and write $(k_2, m_2) = (\mathsf{a}_1 \cdot (u, v, c_1), \mathsf{a}_2 \cdot (u, v, c_1, w))$ and $(k_3, m_3) = (\mathsf{a}_3 \cdot (u, v, c_1), \mathsf{a}_4 \cdot (u, v, c_1, w))$. We make the following distinction:
   - $\mathcal{L}E^{+}[k_2](m_2) = \bot$ and $\mathcal{L}E^{+}[k_3](m_3) = \bot$. In the real world, the answers to the queries $\widetilde{E}(k_2, m_2)$ and $\widetilde{E}(k_3, m_3)$ are both fresh and randomly drawn from $\{0,1\}^n$. Regarding the simulated world, by contradiction we prove that $\mathcal{R}(u, v, w)$ has never been queried before by $\widetilde{\mathcal{S}}$. Indeed, suppose it has been queried before. This necessarily means that there exist $j \in \{1, 3\}$ and $(u, v, c_1) \in \mathcal{L}E$ such that $\mathsf{a}_j \cdot (u, v, c_1) = k'$ and $w = \mathrm{get}w(j + 1, m', u, v, c_1)$ for some $(k', m', c') \in \mathcal{L}E$. The former implies $k' = k_2[j = 1] + k_3[j = 3]$, and the latter implies $m' = \mathsf{a}_{j+1} \cdot (u, v, c_1, w)$ and thus $m' = m_2[j = 1] + m_3[j = 3]$. This contradicts the condition that (k_2, m_2) and (k_3, m_3) are not in $\mathcal{L}E$. Therefore, the query (u, v, w) to $\mathcal{R}$ is new, and the response is randomly drawn from $\{0,1\}^{2n}$;
   - $\mathcal{L}E^{+}[k_2](m_2) \neq \bot$ and/or $\mathcal{L}E^{+}[k_3](m_3) \neq \bot$. Without loss of generality, assume the former and write $c_2 = \mathcal{L}E^{+}[k_2](m_2)$. In the real world, this query could not have been made in an earlier evaluation of F^3_{A} (by virtue of $\mathsf{cond}(\mathcal{L}E)$). Therefore, the distinguisher must have made this query, and particular knows $y = c_2 + m_2$, which is the left half of the query response. In the simulated world, a similar story applies: by $\neg\mathsf{cond}(\mathcal{L}E)$, this query to $\widetilde{\mathcal{S}}$ must have been made after (u, v, c_1), and thus, the response value c_2 equals $m + y$ by line 45, where y equals the left half of $\mathcal{R}(u, v, w)$. Thus also in this case, the distinguisher knows the left half of the query response.

     If also $\mathcal{L}E^{+}[k_3](m_3) \neq \bot$, the same reasoning applies to z, the second half of the query response. On the other hand, in case $\mathcal{L}E^{+}[k_3](m_3) = \bot$, the previous bullet carries over to the z-part.

***R*-query $(\boldsymbol{k}, \boldsymbol{m})$.** We make the following distinction:

1. $\neg \exists\, j \in \{1, 3\}, (k_1, m_1, c_1) \in \mathcal{L}E : k = \mathsf{a}_j \cdot (k_1, m_1, c_1)$. In the simulated world, the response is randomly drawn from $\{0,1\}^n$ by construction. Regarding the real world, first assume (k, m) has never been queried to $\widetilde{E}$ via a query to F^3_{A}. Then, the response is clearly fresh and randomly drawn from $\{0,1\}^n$. However, it may be the case that the $\widetilde{E}$-query could have been triggered by an earlier F^3_{A}-query. However, by the condition, it could have impossibly appeared in such evaluation as a bottom left/right query. It may have appeared as a top query in an F^3_{A} evaluation, which means that (k, m, w) has been queried to F^3_{A} for some w. However, in this setting, the adversary never learnt c_1, and thus the response to the R-query appears completely randomly drawn from $\{0,1\}^n$;

2. $\exists\, j \in \{1,3\}, (k_1, m_1, c_1) \in \mathcal{LE} : k = \mathsf{a}_j\cdot(k_1, m_1, c_1)$. By $\neg\mathsf{cond}(\mathcal{LE})$, these values are unique. Let $w = \mathrm{get}w(j+1, m, k_1, m_1, c_1)$. In the simulated world, the response c is defined as $m+y$ (if $j=1$) or $m+z$ (if $j=3$), where $(y,z) = \mathcal{R}(k_1, m_1, w)$. Clearly, if the distinguisher has queried $\mathcal{R}(k_1, m_1, w)$ before, it knows the response in advance. Otherwise, it is randomly drawn from $\{0,1\}^n$ by construction. Regarding the real world, the same reasoning applies: either the query is new, or it must have appeared as a bottom query (left if $j=1$, right if $j=3$) of an earlier F^3_{A} evaluation (by $\neg\mathsf{cond}(\mathcal{LE})$), in which case the distinguisher knows the response.

***R^{-1}*-query (k, c).** In the simulated world, queries are always answered with a random answer from $\{0,1\}^n$. In the real world, this is also the case, except if a certain query (k,m) with $\mathcal{LE}^+[k](m) = c$ has ever been triggered via a call to F^3_{A}. However, in this case, the response will still appear completely random to the distinguisher, similar to the first item of forward queries to R. □

Lemma 2. $\mathbf{Pr}\left(\mathsf{cond}(\mathcal{LE})\ \textit{for}\ (F^3_{\mathsf{A}}, \widetilde{E})\right) \leq \frac{3(3q_L+q_R)^2}{2^n}$ *and* $\mathbf{Pr}\left(\mathsf{cond}(\mathcal{LE})\ \textit{for}\ (\mathcal{R}, \widetilde{\mathcal{S}})\right) \leq \frac{3q_R^2}{2^n}$.

Proof. We start with the real world $(F^3_{\mathsf{A}}, \widetilde{E})$. At the end of the proof, we highlight the differences that give rise to the bound for the simulated world $(\mathcal{R}, \widetilde{\mathcal{S}})$.

Let $1 \leq i \leq 3q_L + q_R$, and denote by $\mathcal{LE}_i$ the set $\mathcal{LE}$ after the ith query. We assume $\neg\mathsf{cond}(\mathcal{LE}_{i-1})$ and consider the probability $\mathsf{cond}(\mathcal{LE}_i)$ gets satisfied. More detailed, we consider the probability that the ith query makes the condition satisfied for some $j, j' \in \{1,3\}$ and some earlier query $(k', m', c') \in \mathcal{LE}$. Note that $\mathsf{cond}(\mathcal{LE}_i)$ can only be triggered by the values derived in lines 11 and 21. In fact, these values are always randomly generated from $\{0,1\}^n$.

Decomposing $\mathsf{cond}(\mathcal{LE}_i)$, the ith query satisfies the condition if it satisfies any of the following three:

$$\begin{aligned}
&\mathsf{a}_j\cdot(k,m,c) = k && \text{for } j \in \{1,3\}\,,\\
&\mathsf{a}_j\cdot(k,m,c) = k' && \text{for } j \in \{1,3\} \text{ and } (k',m',c') \in \mathcal{LE}_{i-1}\,,\\
&\mathsf{a}_j\cdot(k,m,c) = \mathsf{a}_{j'}\cdot(k',m',c') && \text{for } j,j' \in \{1,3\} \text{ and } (k',m',c') \in \mathcal{LE}_{i-1}\,.
\end{aligned}$$

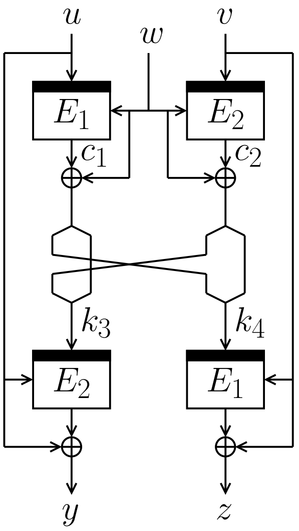
Therefore, $\mathsf{cond}(\mathcal{LE}_i)$ gets satisfied with probability at most $\frac{6(i-1)+2}{2^n}$ (as $\mathsf{a}_{12}, \mathsf{a}_{13}, \mathsf{a}_{32}, \mathsf{a}_{33} \neq 0$). We thus find:

$$\begin{aligned}
\mathbf{Pr}\left(\mathsf{cond}(\mathcal{LE})\right) &\leq \sum_{i=1}^{3q_L+q_R} \mathbf{Pr}\left(\mathsf{cond}(\mathcal{LE}_i) \mid \neg\mathsf{cond}(\mathcal{LE}_{i-1})\right)\\
&\leq \sum_{i=1}^{3q_L+q_R} \frac{6(i-1)+2}{2^n} \leq \frac{3(3q_L+q_R)^2}{2^n}\,.
\end{aligned}$$

Now, for the simulated world, first note that $1 \leq i \leq q_R$. In this setting, $\mathsf{cond}(\mathcal{LE}_i)$ can only be triggered by the values derived in lines 41, 45, and 51. We remark that in line 45, the value c is indeed always a random n-bit value by $\neg\mathsf{cond}(\mathcal{LE}_{i-1})$. □

8 MDC-4

For MDC-4, we leave out the swapping at the end as it is of no influence to the indifferentiability proof. The function is given in Fig. 5. Here, for a bit string x, we write x^l and x^r to denote its left and right halves where $|x^l| = |x^r|$. MDC-4 achieves a higher level of indifferentiability security than MDC-2, mainly due to the two sequential rounds. Differentiability is discussed in Sect. 8.1, and indifferentiability in Sect. 8.2.



MDC-4$(u,v,w)=(y,z)$, where:

$$\begin{aligned}
&c_1 \leftarrow E_1(u,w)\,,\\
&c_2 \leftarrow E_2(v,w)\,,\\
&k_3 \leftarrow c_2^l \| c_1^r + w\,,\\
&y \leftarrow E_2(k_3,u)+u\,,\\
&k_4 \leftarrow c_1^l \| c_2^r + w\,,\\
&z \leftarrow E_1(k_4,v)+v\,.
\end{aligned}$$

Fig. 5. The MDC-4 compression function. For convenience, the swapping at the end is omitted.

8.1 Differentiability

In Prop. 6 we show that MDC-4 is differentiable from a random oracle in at most about $2^{n/4}$ queries. The attack is very similar to the attack of Prop. 5, but is included for convenience. We briefly note that if $E_1 = E_2$, MDC-4 is clearly differentiable in 2 queries, exploiting that MDC-4(u,u,w) has the same left and right half for any $u, w \in \{0,1\}^n$.

Proposition 6. *Let $E_1, E_2 \xleftarrow{\$} Bloc(n)$, and let $\mathcal{R} : \{0,1\}^{3n} \to \{0,1\}^{2n}$ be a random compression function. For any simulator $\mathcal{S}$ that makes at most $q_\mathcal{S}$ queries to $\mathcal{R}$, there exists a distinguisher $\mathcal{D}$ that makes $2^{n/4}+2$ queries to its oracles, such that*

$$\mathbf{adv}^{\text{iff}}_{\text{MDC-4},\mathcal{S}}(\mathcal{D}) \geq \frac{1}{2} - \frac{1}{2^{n/4+1}} - \frac{q_\mathcal{S}+1}{2^n - q_\mathcal{S}}\,.$$

Proof. Our distinguisher $\mathcal{D}$ aims at finding two different evaluations of MDC-4 with the same key inputs to the bottom left block cipher call. In more detail, the distinguisher fixes u and w and aims at finding two distinct block cipher calls (v,w,c_2) and (v',w,c_2') such that:

$$c_2^l = {c_2'}^l\,. \tag{9}$$

Note that in the real world, for MDC-4, such collisions are expected to be found in about $2^{n/4}$ queries to E. If the distinguisher eventually finds a collision as in (9), then the following condition naturally holds in the real world:

$$y := \text{MDC-4}(u,v,w)^l = \text{MDC-4}(u,v',w)^l =: y'\,. \tag{10}$$

In the random world, with MDC-4 replaced by $\mathcal{R}$, this equation only holds with small probability. Note that the simulator never learns the value u, yet, it may simply try to avoid collisions as in (9). However, in this case, the responses from $\mathcal{S}$ are too biased, which allows the distinguisher to succeed.

Formally, the distinguisher $\mathcal{D}$ proceeds as follows.

(i) $\mathcal{D}$ makes $2^{n/4}$ queries to its right oracle R for different key values and for a fixed message value w, obtaining $2^{n/4}$ distinct tuples (v,w,c_2);
(ii) If there is no solution to (9), $\mathcal{D}$ returns 1;

(iii) Let (v, w, c_2) and (v', w, c_2') be such that (9) is satisfied;

(iv) Take $u \xleftarrow{\$} \{0,1\}^n$. If (10) holds, $\mathcal{D}$ returns 0, and otherwise it returns 1.

Distinguisher $\mathcal{D}$ succeeds except in the following two cases: "C_1" it is conversing with the real world and (9) does not have a solution (which means that his guess in step (ii) is wrong), or "C_2" it is conversing with the simulated world and (10) holds (which means that his guess in step (iv) is wrong). Therefore, $\mathbf{adv}^{\text{iff}}_{\text{MDC-4},\mathcal{S}}(\mathcal{D}) \geq 1 - \mathbf{Pr}(\mathsf{C}_1) - \mathbf{Pr}(\mathsf{C}_2)$. Regarding C_1: note that all queries are made with different key inputs, and E_2 is a random cipher. Therefore, all responses are randomly drawn from a set of size 2^n, and a collision (4) occurs with probability at least $\binom{2^{n/4}}{2}\frac{2^{n/2}}{2^n}$. Thus,

$$\mathbf{Pr}(\mathsf{C}_1) \leq 1 - \binom{2^{n/4}}{2}\frac{2^{n/2}}{2^n} = \frac{1}{2} + \frac{1}{2^{n/4+1}}\,.$$

Regarding C_2, the proof of Prop. 5 carries over and we find $\mathbf{Pr}(\mathsf{C}_2) \leq \frac{q_{\mathcal{S}}+1}{2^n - q_{\mathcal{S}}}$. This completes the proof. □

8.2 Indifferentiability

We prove that MDC-4 is indifferentiable from a random function.

Theorem 2. *Let $E_1, E_2 \xleftarrow{\$} Bloc(n)$, and let $\mathcal{R} : \{0,1\}^{3n} \to \{0,1\}^{2n}$ be a random function. There exists a simulator $\mathcal{S}$ such that for any distinguisher $\mathcal{D}$ that makes at most q_L left queries and q_R right queries,*

$$\mathbf{adv}^{\text{iff}}_{\text{MDC-4},\mathcal{S}}(\mathcal{D}) \leq \frac{6(4q_L + q_R)^2}{2^{n/2}}\,,$$

where $\mathcal{S}$ makes $q_{\mathcal{S}} \leq q_R$ queries to $\mathcal{R}$.

For ease of presentation, the proof is given in App. A. It is truly similar to the proof of Thm. 1.

9 Conclusions

Being the only known double length compression function that achieves optimal collision security and a non-trivial indifferentiability bound, Mennink's compression function class appears to be stronger than its alternatives. Yet, this additional level of security does not come for free: the function makes three block cipher calls, rather than "the usual" two, which are moreover not parallelizable. It would be of both theoretical and practical interest to derive a two-call compression function (for either choice of k) with the same or even better security guarantees.[6] We note, however, that the indifferentiability proof in this work relies on the presence of the third block cipher call, and all attacks on functions with $k = 2n$ rely on the fact that these make only two primitive calls.

Acknowledgments. This work has been funded in part by the IAP Program P6/26 BCRYPT of the Belgian State (Belgian Science Policy), in part by the European Commission through the ICT program under contract ICT-2007-216676 ECRYPT II, and in part by the Research Council K.U.Leuven: GOA TENSE. The author is supported by a Ph.D. Fellowship from the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen). The author would like to thank Atul Luykx for his valuable help and feedback.

[6] Without going into detail, we refer to a slightly related work of Maurer and Tessaro [19] on indifferentiable domain extenders from random functions.

References

[1] Andreeva, E., Neven, G., Preneel, B., Shrimpton, T.: Seven-property-preserving iterated hashing: ROX. In: Advances in Cryptology - ASIACRYPT 2007. Lecture Notes in Computer Science, vol. 4833, pp. 130–146. Springer, Heidelberg (2007)

[2] Armknecht, F., Fleischmann, E., Krause, M., Lee, J., Stam, M., Steinberger, J.: The preimage security of double-block-length compression functions. In: Advances in Cryptology - ASIACRYPT 2011. Lecture Notes in Computer Science, vol. 7073, pp. 233–251. Springer, Heidelberg (2011)

[3] Bellare, M., Ristenpart, T.: Multi-property-preserving hash domain extension and the EMD transform. In: Advances in Cryptology - ASIACRYPT 2006. Lecture Notes in Computer Science, vol. 4284, pp. 299–314. Springer, Heidelberg (2006)

[4] Coron, J., Dodis, Y., Malinaud, C., Puniya, P.: Merkle-Damgård revisited: How to construct a hash function. In: Advances in Cryptology - CRYPTO 2005. Lecture Notes in Computer Science, vol. 3621, pp. 430–448. Springer, Heidelberg (2005)

[5] Fleischmann, E., Gorski, M., Lucks, S.: Security of cyclic double block length hash functions. In: IMA International Conference 2009. Lecture Notes in Computer Science, vol. 5921, pp. 153–175. Springer, Heidelberg (2009)

[6] Hirose, S.: Provably secure double-block-length hash functions in a black-box model. In: Information Security and Cryptology 2004. Lecture Notes in Computer Science, vol. 3506, pp. 330–342. Springer, Heidelberg (2005)

[7] Hirose, S.: Some plausible constructions of double-block-length hash functions. In: Fast Software Encryption 2006. Lecture Notes in Computer Science, vol. 4047, pp. 210–225. Springer, Heidelberg (2006)

[8] Hirose, S., Park, J., Yun, A.: A simple variant of the Merkle-Damgård scheme with a permutation. In: Advances in Cryptology - ASIACRYPT 2007. Lecture Notes in Computer Science, vol. 4833, pp. 113–129. Springer, Heidelberg (2007)

[9] Jetchev, D., Özen, O., Stam, M.: Collisions are not incidental: A compression function exploiting discrete geometry. In: Theory of Cryptography Conference 2012. Lecture Notes in Computer Science, vol. 7194, pp. 303–320. Springer, Heidelberg (2012)

[10] Kuwakado, H., Morii, M.: Indifferentiability of single-block-length and rate-1 compression functions. IEICE Transactions 90-A(10), 2301–2308 (2007)

[11] Lai, X., Massey, J.: Hash function based on block ciphers. In: Advances in Cryptology - EUROCRYPT '92. Lecture Notes in Computer Science, vol. 658, pp. 55–70. Springer, Heidelberg (1992)

[12] Lee, J., Kwon, D.: The security of Abreast-DM in the ideal cipher model. Cryptology ePrint Archive, Report 2009/225 (2009)

[13] Lee, J., Stam, M.: MJH: A faster alternative to MDC-2. In: CT-RSA 2011. Lecture Notes in Computer Science, vol. 6558, pp. 213–236. Springer, Heidelberg (2011)

[14] Lee, J., Stam, M., Steinberger, J.: The collision security of Tandem-DM in the ideal cipher model. In: Advances in Cryptology - CRYPTO 2011. Lecture Notes in Computer Science, vol. 6841, pp. 561–577. Springer, Heidelberg (2011)

[15] Lee, J., Stam, M., Steinberger, J.: The preimage security of double-block-length compression functions. Cryptology ePrint Archive, Report 2011/210 (2011)

[16] Lee, J., Steinberger, J.: Multi-property-preserving domain extension using polynomial-based modes of operation. In: Advances in Cryptology - EUROCRYPT 2010. Lecture Notes in Computer Science, vol. 6110, pp. 573–596. Springer, Heidelberg (2010)

[17] Lucks, S.: A collision-resistant rate-1 double-block-length hash function (Symmetric Cryptography, Dagstuhl Seminar Proceedings 07021, 2007)

[18] Maurer, U., Renner, R., Holenstein, C.: Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In: Theory of Cryptography Conference 2004. Lecture Notes in Computer Science, vol. 2951, pp. 21–39. Springer, Heidelberg (2004)

[19] Maurer, U., Tessaro, S.: Domain extension of public random functions: Beyond the birthday barrier. In: Advances in Cryptology - CRYPTO 2007. Lecture Notes in Computer Science, vol. 4622, pp. 187–204. Springer, Heidelberg (2007)

[20] Mennink, B.: Optimal collision security in double block length hashing with single length key. In: Advances in Cryptology - ASIACRYPT 2012. Lecture Notes in Computer Science, vol. 7658, pp. 526–543. Springer, Heidelberg (2012)

[21] Mennink, B.: On the collision and preimage security of MDC-4 in the ideal cipher model. Designs, Codes and Cryptography (2013), to appear

[22] Meyer, C., Schilling, M.: Secure program load with manipulation detection code. In: Proc. Securicom. pp. 111–130 (1988)

[23] Özen, O.: Design and Analysis of Multi-Block-Length Hash Functions. Ph.D. thesis, École Polytechnique Fédérale de Lausanne, Lausanne (2012)

[24] Özen, O., Stam, M.: Another glance at double-length hashing. In: IMA International Conference 2009. Lecture Notes in Computer Science, vol. 5921, pp. 176–201. Springer, Heidelberg (2009)

[25] Preneel, B., Govaerts, R., Vandewalle, J.: Hash functions based on block ciphers: A synthetic approach. In: Advances in Cryptology - CRYPTO '93. Lecture Notes in Computer Science, vol. 773, pp. 368–378. Springer, Heidelberg (1993)

[26] Ristenpart, T., Shacham, H., Shrimpton, T.: Careful with composition: Limitations of the indifferentiability framework. In: Advances in Cryptology - EUROCRYPT 2011. Lecture Notes in Computer Science, vol. 6632, pp. 487–506. Springer, Heidelberg (2011)

[27] Stam, M.: Beyond uniformity: Better security/efficiency tradeoffs for compression functions. In: Advances in Cryptology - CRYPTO 2008. Lecture Notes in Computer Science, vol. 5157, pp. 397–412. Springer, Heidelberg (2008)

[28] Stam, M.: Blockcipher-based hashing revisited. In: Fast Software Encryption 2009. Lecture Notes in Computer Science, vol. 5665, pp. 67–83. Springer, Heidelberg (2009)

[29] Steinberger, J.: The collision intractability of MDC-2 in the ideal-cipher model. In: Advances in Cryptology - EUROCRYPT 2007. Lecture Notes in Computer Science, vol. 4515, pp. 34–51. Springer, Heidelberg (2007)

A Indifferentiability of MDC-4

In this appendix, we prove Thm. 2. The proof is similar to the proof of Thm. 1: it differs in various aspects and therefore deserves a separate proof, but we skip the redundant details. In the remainder of the section, we first introduce our simulator and accommodate it with an intuition, and next present the formal proof.

Simulator Intuition

Similar to Sect. 7.2, the simulator maintains an initially empty lists $\mathcal{L}E_1[k]$ (corresponding to E_1) and $\mathcal{L}E_2[k]$ (corresponding to E_2) for $k \in \{0,1\}^n$. Abusing notation, we also write $\mathcal{L}E = \mathcal{L}E_1 \cup \mathcal{L}E_2$. The simulator is given in Fig. 6. It consists of four interfaces: $\mathcal{S}_1/\mathcal{S}_1^{-1}$ corresponding to E_1/E_1^{-1}, and $\mathcal{S}_2/\mathcal{S}_2^{-1}$ corresponding to E_2/E_2^{-1}.

Again, apart from the **if**-clause of lines 02-06, the simulator identically mimics an ideal cipher. In this particular clause, the simulator checks whether a query (k, m) may appear in an MDC-4 evaluation (see Fig. 5) as a bottom query (left or right) for some other pair of queries appearing in the top. In this case, the simulator should consult $\mathcal{R}$ to derive the query response.

```
Forward Query $\mathcal{S}_j(k, m)$ ($j \in \{1,2\}$)
00 if $\mathcal{L}E_j^+[k](m) \neq \bot$ return $c = \mathcal{L}E_j^+[k](m)$
01 $c \xleftarrow{\$} \{0,1\}^n \backslash \mathcal{L}E_j^+[k]$
02 if $\exists\ (u, w, c_1) \in \mathcal{L}E_1, (v, w, c_2) \in \mathcal{L}E_2 :\ \dots$
03         $\dots\ m = u[j = 2] + v[j = 1]$ and $k = c_j^l \| c_{\bar{j}}^r + w$
04     $(y, z) \leftarrow \mathcal{R}(u, v, w)$
05     $c \leftarrow m + (y[j = 2] + z[j = 1])$
06 end if
07 return $\mathcal{L}E_j^+[k](m) \leftarrow c$
```

```
Inverse Query $\mathcal{S}_j^{-1}(k, c)$ ($j \in \{1,2\}$)
10 if $\mathcal{L}E_j^-[k](c) \neq \bot$ return $m = \mathcal{L}E_j^-[k](c)$
11 $m \xleftarrow{\$} \{0,1\}^n \backslash \mathcal{L}E_j^-[k]$
12 return $\mathcal{L}E_j^-[k](c) \leftarrow m$
```

Fig. 6. The simulator $\mathcal{S}$ for E used in the proof of Thm. 2. Here, $\bar{j} \in \{1,2\}$ is the complement of $j \in \{1,2\}$.

Proof of Theorem 2

We formally proof Thm. 2. The proof is similar to the proof of Thm. 1, with only minor modifications. Let $\mathcal{S}$ be the simulator of Fig. 6, and let $\mathcal{D}$ be any distinguisher that makes at most q_L left queries and q_R right queries. Note that $\mathcal{S}$ makes $q_\mathcal{S} \leq q_R$ queries. By Def. 1, the goal is to bound:

$$\mathbf{adv}^{\text{iff}}_{\text{MDC-4},\mathcal{S}}(\mathcal{D}) = \left|\mathbf{Pr}\left(\mathcal{D}^{\text{MDC-4},E} = 1\right) - \mathbf{Pr}\left(\mathcal{D}^{\mathcal{R},\mathcal{S}} = 1\right)\right| . \tag{11}$$

As in Sect. 7.2, we first perform a PRP-PRF switch. $\widetilde{E}$ and $\widetilde{\mathcal{S}}$ are defined similarly as before, and we obtain for (11):

$$\mathbf{adv}^{\text{iff}}_{\text{MDC-4},\mathcal{S}}(\mathcal{D}) \leq \left|\mathbf{Pr}\left(\mathcal{D}^{\text{MDC-4},\widetilde{E}} = 1\right) - \mathbf{Pr}\left(\mathcal{D}^{\mathcal{R},\widetilde{\mathcal{S}}} = 1\right)\right| + \frac{2(4q_L + q_R)^2}{2^n} . \tag{12}$$

It remains to analyze the probability of $\mathcal{D}$ to distinguish (MDC-4, $\widetilde{E}$) (real world) from $(\mathcal{R}, \widetilde{\mathcal{S}})$ (simulated world). These worlds are described in Fig. 7. The notations $\mathcal{L}E_1$, $\mathcal{L}E_2$ and $\mathcal{LR}$ are defined similarly as before.

Let event $\mathsf{cond}(\mathcal{L}E)$ be defined as follows:

$$\mathsf{cond}(\mathcal{L}E) = \left(\begin{array}{c} \exists\, j \in \{l, r\}, (k, m, c), (k', m', c') \in \mathcal{L}E : \\ (k, m, c) \text{ newer than } (k', m', c') \text{ and} \\ (c + m)^j \in \{k^j, k'^j, (c' + m')^j\} \end{array} \right) . \tag{13}$$

Event $\mathsf{cond}(\mathcal{L}E)$ is fairly the same as the event for the proof in Sect. 7.2 (equation (8)). Therefore, we skip the detailed explanation, and just point out that as long as $\neg\mathsf{cond}(\mathcal{L}E)$, the condition in line 42 of Fig. 7 is always satisfied by at most one $((u, w, c_1), (v, w, c_2))$. In the remainder, we prove in Lem. 3 that (MDC-4, $\widetilde{E}$) and $(\mathcal{R}, \widetilde{\mathcal{S}})$ are perfectly indistinguishable as long as $\mathsf{cond}(\mathcal{L}E)$ does not occur in both worlds. Then, in Lem. 4 we prove that $\mathsf{cond}(\mathcal{L}E)$ occurs in the real world with probability at most $\frac{2(4q_L+q_R)^2}{2^{n/2}}$ and in the simulated world with probability at most $\frac{2q_R^2}{2^{n/2}}$. Together with (12), this completes the proof.

```
Query MDC-4(u, v, w)
00 $c_1 \leftarrow \widetilde{E}_1(u, w)$
01 $c_2 \leftarrow \widetilde{E}_2(v, w)$
02 $k_3 \leftarrow c_2^l \| c_1^r + w$
03 $y \leftarrow \widetilde{E}_2(k_3, u) + u$
04 $k_4 \leftarrow c_1^l \| c_2^r + w$
05 $z \leftarrow \widetilde{E}_1(k_4, v) + v$
06 return $(y, z)$
```

```
Query $\widetilde{E}_j(k, m)$ $(j \in \{1, 2\})$
10 if $\mathcal{L}E_j^+[k](m) \neq \perp$ return $c = \mathcal{L}E_j^+[k](m)$
11 $c \xleftarrow{\$} \{0,1\}^n$
12 return $\mathcal{L}E_j^+[k](m) \leftarrow c$
```

```
Query $\widetilde{E}_j^{-1}(k, c)$ $(j \in \{1, 2\})$
20 if $\mathcal{L}E_j^-[k](c) \neq \perp$ return $m = \mathcal{L}E_j^-[k](c)$
21 $m \xleftarrow{\$} \{0,1\}^n$
22 return $\mathcal{L}E_j^-[k](c) \leftarrow m$
```

```
Query $\mathcal{R}(u, v, w)$
30 if $\mathcal{LR}(u, v, w) \neq \perp$ return $(y, z) = \mathcal{LR}(u, v, w)$
31 $(y, z) \xleftarrow{\$} \{0,1\}^{2n}$
32 return $\mathcal{LR}(u, v, w) \leftarrow (y, z)$
```

```
Query $\widetilde{\mathcal{S}}_j(k, m)$ $(j \in \{1, 2\})$
40 if $\mathcal{L}E_j^+[k](m) \neq \perp$ return $c = \mathcal{L}E_j^+[k](m)$
41 $c \xleftarrow{\$} \{0,1\}^n$
42 if $\exists\, (u, w, c_1) \in \mathcal{L}E_1, (v, w, c_2) \in \mathcal{L}E_2 :\ \ldots$
43         $\ldots\ m = u[j = 2] + v[j = 1]$ and $k = c_j^l \| c_{\bar{j}}^r + w$
44     $(y, z) \leftarrow \mathcal{R}(u, v, w)$
45     $c \leftarrow m + (y[j = 2] + z[j = 1])$
46 end if
47 return $\mathcal{L}E_j^+[k](m) \leftarrow c$
```

```
Query $\widetilde{\mathcal{S}}_j^{-1}(k, c)$ $(j \in \{1, 2\})$
50 if $\mathcal{L}E_j^-[k](c) \neq \perp$ return $m = \mathcal{L}E_j^-[k](c)$
51 $m \xleftarrow{\$} \{0,1\}^n$
52 return $\mathcal{L}E_j^-[k](c) \leftarrow m$
```

Fig. 7. The worlds (MDC-4, $\widetilde{E}$) (left) and $(\mathcal{R}, \widetilde{\mathcal{S}})$ (right).

Lemma 3. *As long as* $\neg\mathsf{cond}(\mathcal{L}E)$*,* (MDC-4, $\widetilde{E}$) *from* $(\mathcal{R}, \widetilde{\mathcal{S}})$ *are perfectly indistinguishable.*

Proof. We consider any query made by the distinguisher, either to the left oracle L (either MDC-4 or $\mathcal{R}$) and the right oracle R/R^{-1} (either $\widetilde{E}/\widetilde{E}^{-1}$ or $\widetilde{\mathcal{S}}/\widetilde{\mathcal{S}}^{-1}$), and show that the query responses are equally distributed in both worlds (irrespectively of the query history). Without loss of generality, we consider new queries only: if the distinguisher makes a repetitive query, the answer is known and identically distributed in both worlds.

***L*-query (u, v, w).** We make the following distinction:

1. $\mathcal{L}E_1^+[u](w) = \perp$ and/or $\mathcal{L}E_2^+[v](w) = \perp$. In the real world, this means that the first cipher call $\widetilde{E}_1(u, w)$ or the second call $\widetilde{E}_2(v, w)$ is new, and answered with a fresh value. As $\mathsf{cond}(\mathcal{L}E)$ does not occur, also the third *and* fourth call, $\widetilde{E}_2(k_3, u)$ and $\widetilde{E}_1(k_4, v)$, are fresh, and both their responses are drawn from $\{0,1\}^n$. Regarding the simulated world, by the condition "$\mathcal{L}E_1^+[u](w) = \perp$ or $\mathcal{L}E_2^+[v](w) = \perp$," $\widetilde{\mathcal{S}}$ has never queried $\mathcal{R}$ on input of (u, v, w). Indeed, it had only queried $\mathcal{R}$ if the condition of line 42 was satisfied for some *existing* $(u, w, c_1) \in \mathcal{L}E_1$ and $(v, w, c_2) \in \mathcal{L}E_2$. Thus, also in this world the response is randomly generated from $\{0,1\}^{2n}$;
2. $\mathcal{L}E_1^+[u](w) \neq \perp$ and $\mathcal{L}E_2^+[v](w) \neq \perp$. Note that in the real world, these elements could have been added to $\mathcal{L}E$ via $\mathcal{D}$ or via MDC-4. Let $c_1 = \mathcal{L}E_1^+[u](w)$ and $c_2 = \mathcal{L}E_2^+[v](w)$, and write $k_3 = c_2^l \| c_1^r + w$ and $k_4 = c_1^l \| c_2^r + w$. We make the following distinction:
   - $\mathcal{L}E_2^+[k_3](u) = \perp$ and $\mathcal{L}E_1^+[k_4](v) = \perp$. In the real world, the answers to the queries $\widetilde{E}_2(k_3, u)$ and $\widetilde{E}_1(k_4, v)$ are both fresh and randomly drawn from $\{0,1\}^n$. Regarding the simulated world, by contradiction we prove that $\mathcal{R}(u, v, w)$ has never been queried before by $\widetilde{\mathcal{S}}$. Indeed, suppose it has been queried before. This necessarily means that there exist $(u, w, c_1) \in \mathcal{L}E_1$ and $(v, w, c_2) \in \mathcal{L}E_2$ such that $c_{\bar{j}}^l \| c_j^r + w = k'$ and $u[j = 2] + v[j = 1] = m'$ for some $(k', m', c') \in \mathcal{L}E_j$. The former implies $k' = k_3[j = 2] + k_4[j = 1]$. This contradicts the condition that (k_3, u) is not in $\mathcal{L}E_2$ and (k_4, v) not in $\mathcal{L}E_1$. Therefore, the query (u, v, w) to $\mathcal{R}$ is new, and the response is randomly drawn from $\{0,1\}^{2n}$;
   - $\mathcal{L}E_2^+[k_3](u) \neq \perp$ and/or $\mathcal{L}E_1^+[k_4](v) \neq \perp$. Without loss of generality, assume the former and write $c_3 = \mathcal{L}E_2^+[k_3](u)$. In the real world, this query could not have been made in an earlier evaluation of MDC-4 (by virtue of $\mathsf{cond}(\mathcal{L}E)$). Therefore, the distinguisher must have made this query, and particular knows $y = c_3 + u$, which is the left half of the query response. In the simulated world, a similar story applies: by $\neg\mathsf{cond}(\mathcal{L}E)$, this query to $\widetilde{\mathcal{S}}$ must have been made after (u, w, c_1) and (v, w, c_2), and thus, the response value c_3 equals $u + y$ by line 45, where y equals the left half of $\mathcal{R}(u, v, w)$. Thus also in this case, the distinguisher knows the left half of the query response.

     If also $\mathcal{L}E_1^+[k_4](v) \neq \perp$, the same reasoning applies to z, the second half of the query response. On the other hand, in case $\mathcal{L}E_1^+[k_4](v) = \perp$, the previous bullet carries over to the z-part.

***R_j*-query (k, m) $(j \in \{1, 2\})$.** We make the following distinction:

1. $\neg \exists\ (u, w, c_1) \in \mathcal{L}E_1, (v, w, c_2) \in \mathcal{L}E_2 : m = u[j = 2] + v[j = 1]$ and $k = c_{\bar{j}}^l \| c_j^r + w$. In the simulated world, the response is randomly drawn from $\{0,1\}^n$ by construction. Regarding the real world, first assume (k, m) has never been queried to $\widetilde{E}_j$ via a query to MDC-4. Then, the response is clearly fresh and randomly drawn from $\{0,1\}^n$. However, it may be the case that the $\widetilde{E}_j$-query could have been triggered by an earlier MDC-4-query. However, by the condition, it could have impossibly appeared in such evaluation as a bottom left/right query. It may have appeared as a top left/right query in an MDC-4 evaluation, which means that (k, v, m) has been queried to MDC-4 for some v (if $j = 1$) or (u, k, m) for some u (if $j = 2$). However, in this setting, the adversary never learnt c, and thus the response to the R-query appears completely randomly drawn from $\{0,1\}^n$;

2. $\exists\ (u, w, c_1) \in \mathcal{L}E_1, (v, w, c_2) \in \mathcal{L}E_2 : m = u[j = 2] + v[j = 1]$ and $k = c_j^l \| c_{\bar{j}}^r + w$. By $\neg\mathsf{cond}(\mathcal{L}E)$, these values are unique. In the simulated world, the response c is defined as $m + y$ (if $j = 2$) or $m + z$ (if $j = 1$), where $(y, z) = \mathcal{R}(u, v, w)$. Clearly, if the distinguisher has queried $\mathcal{R}(u, v, w)$ before, it knows the response in advance. Otherwise, it is randomly drawn from $\{0, 1\}^n$ by construction. Regarding the real world, the same reasoning applies: either the query is new, or it must have appeared as a bottom query (left if $j = 2$, right if $j = 1$) of an earlier MDC-4 evaluation (by $\neg\mathsf{cond}(\mathcal{L}E)$), in which case the distinguisher knows the response.

R_j^{-1}-query (k, c) $(j \in \{1, 2\})$. In the simulated world, queries are always answered with a random answer from $\{0, 1\}^n$. In the real world, this is also the case, except if a certain query (k, m) with $\mathcal{L}E^+[k](m) = c$ has ever been triggered via a call to MDC-4. However, in this case, the response will still appear completely random to the distinguisher, similar to the first item of forward queries to R_j. □

Lemma 4. $\mathbf{Pr}\left(\mathsf{cond}(\mathcal{L}E) \textit{ for } (\text{MDC-4}, \widetilde{E})\right) \leq \frac{2(4q_L+q_R)^2}{2^{n/2}}$ *and* $\mathbf{Pr}\left(\mathsf{cond}(\mathcal{L}E) \textit{ for } (\mathcal{R}, \widetilde{\mathcal{S}})\right) \leq \frac{2q_R^2}{2^{n/2}}$.

Proof. The proof is the same as the proof of Lem. 2, with the differences that $\mathsf{cond}(\mathcal{L}E_i)$ gets satisfied with probability at most $\frac{4(i-1)+2}{2^{n/2}}$ and that for the real world $(\text{MDC-4}, \widetilde{E})$ i ranges from 1 to $4q_L + q_R$. □