Optimal Collision Security in Double Block Length Hashing with Single Length Key

Bart Mennink

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

Abstract. The idea of double block length hashing is to construct a compression function on $2n$ bits using a block cipher with an n-bit block size. All optimally secure double block length hash functions known in the literature employ a cipher with a key space of double block size, $2n$-bit. On the other hand, no optimally secure compression functions built from a cipher with an n-bit key space are known. Our work deals with this problem. Firstly, we prove that for a wide class of compression functions with two calls to its underlying n-bit keyed block cipher collisions can be found in about $2^{n/2}$ queries. This attack applies, among others, to functions where the output is derived from the block cipher outputs in a linear way. This observation demonstrates that all security results of designs using a cipher with $2n$-bit key space crucially rely on the presence of these extra n key bits. The main contribution of this work is a proof that this issue can be resolved by allowing the compression function to make one extra call to the cipher. We propose a family of compression functions making three block cipher calls that asymptotically achieves optimal collision resistance up to $2^{n(1-\varepsilon)}$ queries and preimage resistance up to $2^{3n(1-\varepsilon)/2}$ queries, for any $\varepsilon > 0$. To our knowledge, this is the first optimally collision secure double block length construction using a block cipher with single length key space. We additionally prove this class of functions indifferentiable from random functions in about $2^{n/2}$ queries, and demonstrate that no other function in this direction achieves a bound of similar kind.

Keywords. double block length; compression function; collision resistance; preimage resistance; indifferentiability; beyond birthday bound.

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 [29]. In recent years, the design methodology got renewed attention in the works of [3, 6, 11, 14, 15, 18, 27, 34, 42]. 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 [36, 41]: 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 and most-studied type of compression functions, namely functions that compress $3n$ to $2n$ bits. Those can be classified into two classes: compression functions that internally evaluate a $2n$-bit keyed block cipher $E : \{0,1\}^{2n} \times \{0,1\}^n \to \{0,1\}^n$ (which we will call the DBL^{2n} class), and ones that employ an n-bit keyed block cipher $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ (the DBL^n class). The DBL^{2n} class is well understood. It includes the classical compression functions Tandem-DM and Abreast-DM [13] and Hirose's function [8] (see Figure 1), as well as Stam's supercharged single call Type-I compression function design [40, 41] (reconsidered in [20]) and the generalized designs by Hirose [7] and Özen and Stam [34]. As illustrated in Table 1, all of these functions provide optimal collision security guarantees (up to about 2^n queries), and Tandem-DM, Abreast-DM, and Hirose's function are additionally proven optimally preimage resistant (up to about 2^{2n} queries). These bounds carry over to iterative designs, when a proper domain extender is applied [2]. Lucks [22] 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. Members of the DBL^n class are the MDC-2 and MDC-4 compression functions [29], the MJH construction [15], and a construction by Jetchev et al. [11]. For the MDC-2 and MJH compression functions, collisions and preimages can be found in about $2^{n/2}$ and 2^n queries, respectively.[1] The MDC-4 compression function achieves a higher level of collision and preimage resistance than MDC-2 [27], but contrary to the other functions it makes four block cipher calls. Jetchev et al.'s construction makes two block cipher calls and achieves $2^{2n/3}$ collision security. Stam also introduced a design based on two calls, and proved it optimally collision secure in a restricted security model where the adversary must fix its queries in advance. Therefore we did not include this design in the table.

Further related results include the work of Nandi et al. [32], who presented a $3n$-to-$2n$-bit compression function making three calls to a $2n$-to-n-bit one-way function, achieving collision security up to $2^{2n/3}$ queries. They extended this result to a $4n$-to-$2n$-bit function using three $2n$-bit keyed block ciphers. Peyrin et al. [35] introduced double length compression functions based on five compression function calls with n-bit output. Related are also Lucks wide-pipe design [21] and its generalization by Nandi [31].

Unlike the DBL^{2n} class, for the DBL^n class no optimally secure compression function is known. The situation is the same for the iteration, where none of these designs has been proven to achieve optimal security. Determinative to this gap is the difference in the underlying primitive: in the DBL^{2n} class, the underlying primitive maps $3n$ bits to n bits and thus allows for more compression. In particular, if we look at Tandem-DM, Abreast-DM, and Hirose's function (Figure 1), the first cipher call already compresses the entire input (u, v, w) to the compression function, and the second cipher call is simply used to assure a $2n$-bit output. In fact, these designs achieve their level of security merely due to this property, for their proofs crucially rely on this (see also Section 4).

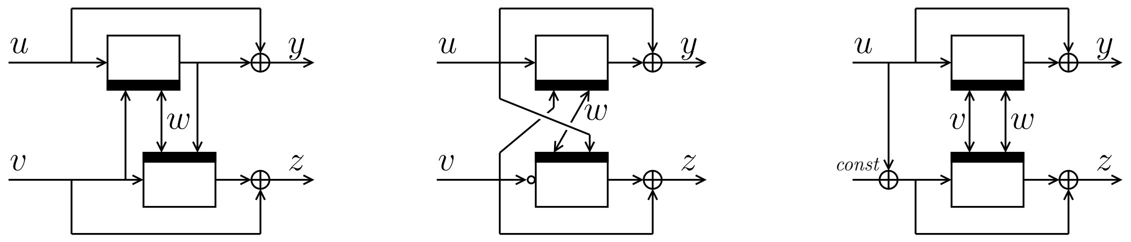


Fig. 1. From left to right, Tandem-DM, Abreast-DM, and Hirose's compression function [8,13]. All wires carry n bits. For Abreast-DM, the circle $\circ$ denotes bit complementation. For Hirose's function, *const* is any non-zero constant.

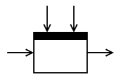
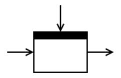
Our Contributions

Thus, from a theoretical point of view it is unreasonable to compare DBL^{2n} and DBL^n. But the gap between the two classes leaves us with an interesting open problem: starting from a single block cipher $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$, is it possible to construct a double length compression function that achieves optimal collision and preimage security? This is the central research question of this work. Note that Stam's bound [40,43,44] does not help us here: it claims that collisions can be found in at most $(2^n)^{(2r-1)/(r+1)}$ queries, where r denotes the number of block cipher calls, which results in the trivial bound for $r \geq 2$. For $r \geq 2$, denote by $F^r : \{0,1\}^{3n} \to \{0,1\}^{2n}$ a compression function that makes r calls to its primitive E.

Impossibility Result for Two-call Hashing. As a first contribution, we consider F^2 based on two potentially distinct blockciphers E_1, E_2, and prove that for a very large class of functions of

[1] In the iteration collision resistance is proven up to $2^{3n/5}$ queries for MDC-2 [42] and $2^{2n/3}$ queries for MJH [15]; the latter result got recently improved to 2^n asymptotically [16].

Table 1. Asymptotic ideal cipher model security guarantees of known functions in the classes DBL^{2n} (first) and DBL^{n} (second). All results printed in **bold** are derived in this work.

compression function	E-calls	collision security	preimage security	indifferen-tiability	underlying cipher
Stam's	1	2^n [41]	2^n [41]	**2** **(App. B.1)**	
Tandem-DM	2	2^n [18]	2^{2n} [3,19]	**2** **(App. B.1)**	
Abreast-DM	2	2^n [6,14]	2^{2n} [3,19]	**2** **(App. B.1)**	
Hirose's	2	2^n [8]	2^{2n} [3,19]	**2** **(App. B.1)**	
Hirose-class	2	2^n [7]	2^n [7]	**2** **(App. B.1)**	
Özen-Stam-class	2	2^n [34]	2^n [34]	**2** **(App. B.2)**	
MDC-2	2	$2^{n/2}$	2^n	**2** **(App. C)**	
MJH	2	$2^{n/2}$	2^n	**2** **(App. C)**	
JOS	2	$2^{2n/3}$ [11]	2^n [11]	**2** **(App. D)**	
Our Proposal	**3**	$\mathbf{2^n}$ **(Sect. 5)**	$\mathbf{2^{3n/2}}$ **(Sect. 6)**	$\mathbf{2^{n/2}}$ **(Sect. 7)**	
MDC-4	4	$2^{5n/8}$ [27]	$2^{5n/4}$ [27]	$\mathbf{2^{n/4}}$ **(App. E)**	

this form one expects collisions in approximately $2^{n/2}$ queries. Covered by the attack are among others designs with linear finalization function (the function that produces the $2n$-bit output given the $3n$-bit input and the block cipher responses). We note that the compression function by Jetchev et al. [11] is not vulnerable to the attack due to its non-linear finalization function. Nevertheless, these results strengthen the claim that no practical optimally collision secure F^2 function exists.

Toward Optimally Collision and Preimage Secure F^3. Motivated by this, we increase the number of calls to E, and consider F^3. In this setting, we derive a family of compression functions which we prove asymptotically optimal collision resistant up to $2^{n(1-\varepsilon)}$ queries and preimage resistant up to $2^{3n(1-\varepsilon)/2}$ queries, for any $\varepsilon > 0$. Our compression function family, thus, achieves the same level of collision security as the well-established Tandem-DM, Abreast-DM, and Hirose's function, albeit based on a much weaker assumption. In the DBL^n class, our design clearly compares favorably to MDC-4 that makes four block cipher evaluations, and from a provable security point of view it beats MDC-2 and MJH, still, an extra E evaluation has to be made which results in an efficiency loss. The introduced class of compression functions is simple and easy to understand: they are defined by 4×4 matrices over the field $GF(2^n)$ which are required to comply with easily satisfied conditions. Two example compression functions in this class are given in Figure 2.

The collision and preimage security proofs of our compression function family rely on basic principles from previous proofs, but in order to accomplish optimal collision security (and as our designs use n-bit keyed block ciphers) our proofs have become significantly more complex. The collision and preimage security proofs of all known DBL^{2n} functions (see Table 1) crucially rely on the property that one block cipher evaluation defines the input to the second one. For F^3 this cannot be achieved as each primitive call fixes at most $2n$ bits of the function input. Although one may expect this to cause an optimal proof to become unlikely, this is not the case. Using a new proof approach—we smartly apply the methodology of "wish lists" (by Lee et al. [17,19] and Armknecht et al. [3]) to collision resistance—we manage to achieve asymptotically the close to 2^n collision security for our family of functions.

Nonetheless, the bound on preimage resistance does not reach the optimal level of 2^{2n} queries. One can see this as the price we pay for using single key length rather than double key length block ciphers: a rather straightforward generalization of the pigeonhole-birthday attack of Rogaway and Steinberger [39] shows that, when the compression function behaves "sufficiently random", one

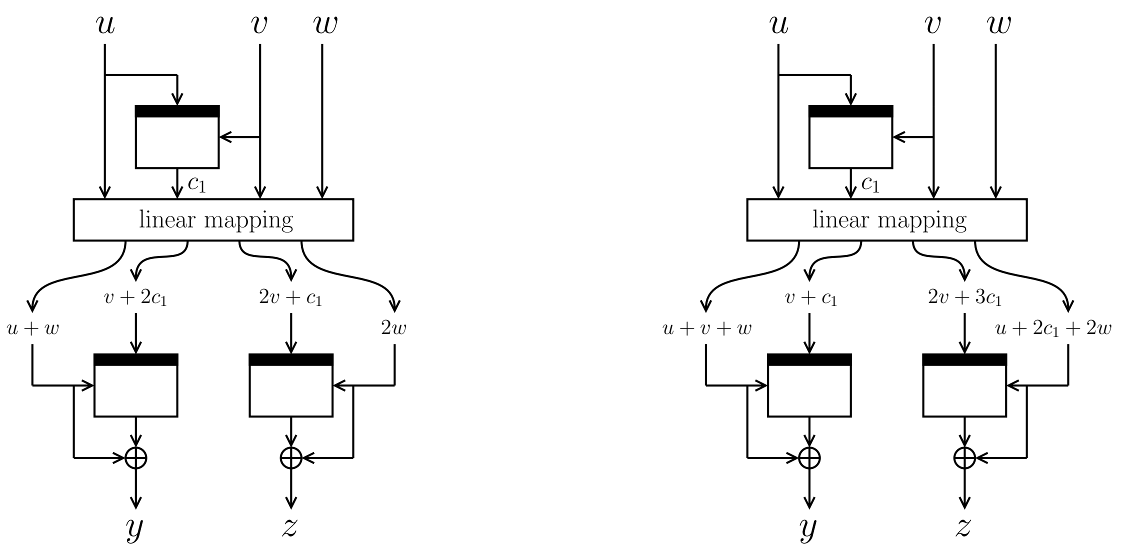


Fig. 2. Two example compression functions from the family of functions introduced and evaluated in this work. For these constructions, all wires carry $n = 128$ bits, and the arithmetic is done over $GF(2^{128})$. We further elaborate on these designs and their derivations in Section 4.

may expect a preimage in approximately $2^{5n/3}$ queries (cf. Section 2). The asymptotic preimage bound of $2^{3n/2}$ found in this work closely approaches this generic bound.

Indifferentiability. Beyond these security notions, the *indifferentiability* framework of Maurer et al. [23] 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. Indifferentiability 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 [4, 5, 9] and compression functions are usually easier to analyze than hash functions, and therefore it is of interest to study the indifferentiability of compression functions.

We prove that our compression function design is indifferentiable from a random compression function up to about $2^{n/2}$ queries (tight). This bound is worse than the collision and preimage bounds, but this is in fact as expected. Indeed, as for single block length compression functions, the PGV functions are known to be differentiable from random functions [12], and it turns out that this problematic situation also applies to double length functions: we demonstrate in Appendices B-D that 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. In general, indifferentiability appears to be much harder to achieve then "simply" collision and preimage security.

Our indifferentiability proof crucially relies on the above-mentioned key characteristics of F^3, but is in general made possible by the sequential block cipher evaluation of the design. We additionally show that a similar proof approach results in a tight $2^{n/4}$ indifferentiability bound for MDC-4 based on two distinct block ciphers.[2]

Publication History and Subsequent Work

The results in this article have appeared in the proceedings of ASIACRYPT 2012 [25] and of IMA Cryptography and Coding 2013 [26]. This article is the full version including all proofs that were

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

not present in the proceedings versions. The impossibility proof for two-call hashing from [25] turned out to miss a subtle side condition. In this article, the condition is added, and we explain the necessity of it.

Several follow-up works on [25, 26] have appeared, and we highlight the most relevant ones. Firstly, Hong and Kwon [10] reconsidered the preimage security of the compression function of [25], and found an alternative preimage attack in approximately $2^{3n/2}$ queries. They also look at the compression function in Merkle-Damgård mode and note that preimages for the resulting hash function can be found in about $2^{7n/4}$ queries, due to a meet-in-the-middle attack.

Abed et al. [1] introduced Counter-bDM : $\{0,1\}^{(b+1)n} \to \{0,1\}^{bn}$, a generalization of Hirose's compression function to $b \geq 2$ parallel evaluations of a block cipher with bn-bit key. Using the principle of wish lists similarly as in [25], the authors derive asymptotically optimally collision and preimage security. However, for $b \geq 2$ the scheme requires a block cipher with a much larger key.

Miyaji and Rashed [30] recently introduced a three-call compression function in the DBL^n class which is claimed to achieve optimal 2^n collision and 2^{2n} preimage resistance in a Merkle-Damgård mode of operation. These results surprisingly contradict the generalizations of the pigeonhole-birthday attacks of Rogaway and Steinberger [39] as derived in [25] (see also Section 2). In more detail, these generic attacks show that a preimage for the compression function can be expected in at most $2^{5n/3}$ queries, and for the iterated design in at most $2^{11n/6} \ll 2^{2n}$ queries (due to the meet-in-the-middle attack). In fact, it turns out that the compression function proposal of Miyaji and Rashed is flawed, allowing for collisions in $2^{2n/3}$ and preimages in 2^n queries. The latter can be used to derive preimages for the iterated design in about $2^{3n/2}$ queries. A more detailed discussion of the scheme of [30], along with the collision and preimage attacks, is given in Appendices A.

Outline

We present and formalize the security model in Section 2. Then, in Section 3 we derive our impossibility result on F^2. We propose and analyze our family of compression functions in Section 4. Collision resistance, preimage resistance, and indifferentiability of this family of functions is proven in Sections 5-7. This work is concluded in Section 8. Supporting indifferentiability guarantees for all other functions in Table 1 is derived in Appendices B-E.

2 Security Model

For $n \geq 1$, we denote by $\mathrm{Bloc}(n)$ the set of all block ciphers with a key and message space of n bits. Let $E \in \mathrm{Bloc}(n)$. For $r \geq 1$, let $F^r : \{0,1\}^{3n} \to \{0,1\}^{2n}$ be a double length compression function making r calls to its block cipher E. We can represent F^r by mappings $f_i : \{0,1\}^{(i+2)n} \to \{0,1\}^{2n}$ for $i = 1, \ldots, r+1$ as follows:

```
$F^r(u, v, w) = (y, z)$, where:
  for $i = 1, \ldots, r$:
    $(k_i, m_i) \leftarrow f_i(u, v, w; c_1, \ldots, c_{i-1})$,
    $c_i \leftarrow E(k_i, m_i)$,
  $(y, z) \leftarrow f_{r+1}(u, v, w; c_1, \ldots, c_r)$.
```

For $r = 3$, the F^r compression function design is depicted in Figure 3. This generic design is a generalization of the permutation based hash function construction described by Rogaway and Steinberger [39]. In fact, it is straightforward to generalize the main findings of [39] to our F^r design and we state them as preliminary results. If the collision- and preimage-degeneracies are sufficiently small (these values intuitively capture the degree of non-randomness of the design with respect to the occurrence of collisions and preimages), one can expect collisions after approximately $2^{n(2-2/r)}$ queries and preimages after approximately $2^{n(2-1/r)}$ queries. We refer to [39] for the details. First of all, these findings confirm that at least two cipher calls are required to get 2^n collision resistance. More importantly, from these results we can conclude that F^r can impossibly achieve optimal 2^{2n} preimage resistance. Yet, it may still be possible to construct a function that achieves optimal collision resistance and almost-optimal preimage resistance.

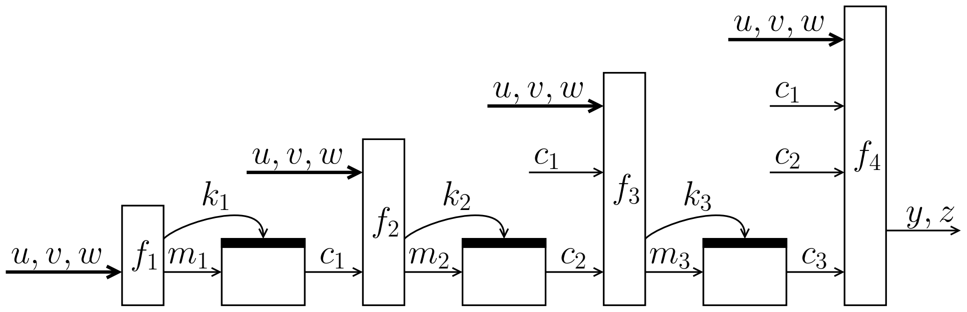


Fig. 3. $F^3 : \{0,1\}^{3n} \to \{0,1\}^{2n}$ making three block cipher evaluations.

Collision and Preimage Resistance

Throughout, we consider security in the ideal cipher model: we consider an adversary $\mathcal{A}$ that is a probabilistic algorithm with oracle access to a block cipher $E \xleftarrow{\$} \text{Bloc}(n)$ randomly sampled from $\text{Bloc}(n)$. We consider $\mathcal{A}$ to be computationally unbounded, and its complexity is measured by the number of queries made to its oracles. The adversary can make forward queries and inverse queries to E, and these are stored in a query history Q as indexed tuples of the form (k_i, m_i, c_i), where k_i denotes the key input, and (m_i, c_i) the plaintext/ciphertext pair. For $q \geq 0$, by Q_q we define the query history after q queries. We assume that the adversary never makes queries to which it knows the answer in advance.

A collision-finding adversary $\mathcal{A}$ for F^r aims at finding two distinct inputs to F^r that compress to the same range value. In more detail, we say that $\mathcal{A}$ succeeds if it finds two distinct tuples $(u, v, w), (u', v', w')$ such that $F^r(u, v, w) = F^r(u', v', w')$ and Q contains all queries required for these evaluations of F^r. Formally, we define the collision security of F^r as follows.

Definition 1. *The advantage of a collision-finding adversary $\mathcal{A}$ is defined as*

$$\mathbf{adv}_{F^r}^{\mathrm{coll}}(\mathcal{A}) = \mathbf{Pr}\left(\begin{array}{r} E \xleftarrow{\$} Bloc(n),\ (u,v,w),(u',v',w') \leftarrow \mathcal{A}^{E,E^{-1}}\ : \quad \\ (u,v,w) \neq (u',v',w')\ \wedge\ F^r(u,v,w) = F^r(u',v',w') \end{array} \right).$$

We define by $\mathbf{adv}_{F^r}^{\mathrm{coll}}(q)$ the maximum advantage of any adversary making q queries to its oracle.

For preimage resistance, we focus on everywhere preimage resistance [38], which captures preimage security for every point of $\{0,1\}^{2n}$. Consider any range point $(y, z) \in \{0,1\}^{2n}$. We say that $\mathcal{A}$ succeeds in finding a preimage if it obtains a tuple (u, v, w) such that $F^r(u, v, w) = (y, z)$ and Q contains all queries required for this evaluation of F^r. Formally, we define the preimage security of F^r as follows.

Definition 2. *The advantage of an everywhere preimage-finding adversary $\mathcal{A}$ is defined as*

$$\mathbf{adv}_{F^r}^{\mathrm{epre}}(\mathcal{A}) = \max_{(y,z)\,\in\,\{0,1\}^{2n}} \mathbf{Pr}\left(\begin{array}{c} E \xleftarrow{\$} Bloc(n),\ (u,v,w) \leftarrow \mathcal{A}^{E,E^{-1}}(y,z)\ : \\ F^r(u,v,w) = (y,z) \end{array} \right)$$

We define by $\mathbf{adv}_{F^r}^{\mathrm{epre}}(q)$ the maximum advantage of any adversary making q queries to its oracle.

Indifferentiability

The indifferentiability framework, introduced by Maurer et al. [23], 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 F^r using an ideal subcomponent $E \xleftarrow{\$} \mathrm{Bloc}(n)$. We employ the adaption and simplification by Coron et al. [5]. Recent results by Ristenpart et al. [37] 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 3. *Let $\mathcal{R} : \{0,1\}^{3n} \to \{0,1\}^{2n}$ be a random function. Let $\mathcal{S}$ be a simulator with the same domain and range as E 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*

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

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

3 Impossibility Result for Two-call Double Length Hashing

We present an attack on a wide class of double block length compression functions with two calls. To suit the analysis, we consider a slightly more restricted design (compared with Section 2), namely, where the two underlying primitives are two different block ciphers $E_1, E_2 : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$. In more detail, the block cipher used at the ith iteration (for $i = 1, 2$) is E_i. Let F^2 be a compression function of this form. We pose a condition on the finalization function f_3, such that if this condition is satisfied, collisions for F^2 can be found in about $2^{n/2}$ queries. Although we are not considering all possible compression functions, we cover the most interesting and intuitive ones, such as compression functions with linear finalization function f_3. Compression functions with non-linear f_3 are covered up to some degree (but we note that the attack does not apply to the compression function of [11], for which collision security up to $2^{2n/3}$ queries is proven).

We first state the attack. Then, by ways of examples, we illustrate its generality. For the purpose of the attack, we introduce the function left_n which on input of a bit string of length $2n$ bits outputs the leftmost n bits.

Proposition 1. *Let $E_1 \leftarrow Bloc(n)$ and $E_2 \xleftarrow{\$} Bloc(n)$. Let $F^2 : \{0,1\}^{3n} \to \{0,1\}^{2n}$ be a compression function as described in Section 2, with first block cipher being E_1 and second block cipher E_2. Suppose there exists a bijective function L such that for any $u, v, w, c_1, c_2 \in \{0,1\}^n$ we have*

$$\mathsf{left}_n \circ L \circ f_3(u, v, w; c_1, c_2) = \mathsf{left}_n \circ L \circ f_3(u, v, w; c_1, 0) . \quad (1)$$

Suppose furthermore that for any $(u, v, w; c_1) \neq (u', v', w'; c_1')$,

$$\mathbf{Pr}\begin{pmatrix} c_2 \leftarrow E_2(f_2(u, v, w; c_1)),\ c_2' \leftarrow E_2(f_2(u', v', w'; c_1')) \ : \\ \mathsf{right}_n \circ L \circ f_3(u, v, w; c_1, c_2) = \mathsf{right}_n \circ L \circ f_3(u', v', w'; c_1', c_2') \end{pmatrix} \geq \frac{1}{2^n} , \quad (2)$$

where the probability is taken over the randomness of E_2. Then, there is an adversary against F^2 that makes $2 \cdot (2^{n/2} + 1)$ queries, for which the expected number of collisions it finds is at least $1/2$.

Proof. Let F^2 be a compression function and let L be a bijection such that (1) holds. First, we consider the case of L being the identity function, and next we show how this attack extends to the case L is an arbitrary bijection.

Suppose (1) holds with L the identity function. This means that the first n bits of $f_3(u, v, w; c_1, c_2)$ do not depend on c_2 and we can write f_3 as a concatenation of two functions $g_1 : \{0,1\}^{4n} \to \{0,1\}^n$ and $g_2 : \{0,1\}^{5n} \to \{0,1\}^n$ as follows:

$$f_3(u, v, w; c_1, c_2) = g_1(u, v, w; c_1) \| g_2(u, v, w; c_1, c_2)\,.$$

Let $\alpha \in \mathbb{N}$. We present an adversary $\mathcal{A}$ for F^2. The first part of the attack is derived from [39].

- Make α queries $(k_1, m_1) \to c_1$ to E_1 that maximize the number of tuples (u, v, w) with $f_1(u, v, w)$ hitting any of these values (k_1, m_1). By the pigeonhole principle,[3] the adversary obtains at least $\alpha \cdot 2^{3n}/2^{2n} = \alpha 2^n$ tuples $(u, v, w; c_1)$ for which it knows the first block cipher evaluation;
- Again by the balls-and-bins principle, there exists a value y such that at least α tuples satisfy $g_1(u, v, w; c_1) = y$;
- Varying over these α tuples, compute $(k_2, m_2) = f_2(u, v, w; c_1)$ and query (k_2, m_2) to E_2 to obtain a c_2. $\mathcal{A}$ finds a collision for F^2 if it obtains two tuples $(u, v, w; c_1, c_2)$, $(u', v', w'; c_1', c_2')$ that satisfy $g_2(u, v, w; c_1, c_2) = g_2(u', v', w'; c_1', c_2')$.

Due to (2) and by linearity of expectation, which applies even though the responses by E_2 are not mutually independent, the expected number of collisions in the last round is at least $\binom{\alpha}{2}\frac{1}{2^n}$. Putting $\alpha = 2^{n/2} + 1$, the expected number of collisions is at least $1/2$. In total, the attack is done in approximately $2 \cdot (2^{n/2} + 1)$ queries.

It remains to consider the case of L being an arbitrary bijection. Define $\overline{F}^2$ as F^2 with f_3 replaced by $\overline{f_3} = L \circ f_3$. Using the idea of equivalence classes on compression functions [28] we prove that F^2 and $\overline{F}^2$ are equally secure with respect to collisions. Let $\overline{\mathcal{A}}$ be a collision finding adversary for $\overline{F}^2$. We construct a collision finding adversary $\mathcal{A}$ for F^2, with oracle access to E_1, E_2, that uses $\overline{\mathcal{A}}$ to output a collision for F^2. Adversary $\mathcal{A}$ proceeds as follows. It forwards all queries made by $\overline{\mathcal{A}}$ to its own oracle. Eventually, $\overline{\mathcal{A}}$ outputs two tuples $(u, v, w), (u', v', w')$ such that $\overline{F}^2(u, v, w) = \overline{F}^2(u', v', w')$. Denote by c_1 the block cipher outcome on input of $f_1(u, v, w)$ and by c_2 the outcome on input of $f_2(u, v, w; c_1)$. Define c_1' and c_2' similarly. By construction, as (u, v, w) and (u', v', w') form a collision for $\overline{F}^2$, we have

$$L \circ f_3(u, v, w; c_1, c_2) = L \circ f_3(u', v', w'; c_1', c_2')\,.$$

Now, bijectivity of L implies that $f_3(u, v, w; c_1, c_2) = f_3(u', v', w'; c_1', c_2')$, and hence (u, v, w) and (u', v', w') form a collision for F^2. (Recall that F^2 and $\overline{F}^2$ only differ in the finalization function f_3, the functions f_1 and f_2 are the same.) We thus obtain $\mathbf{adv}^{\mathrm{coll}}_{\overline{F}^2}(q) \leq \mathbf{adv}^{\mathrm{coll}}_{F^2}(q)$. The derivation in reverse order is the same by symmetry. But $\overline{F}^2$ satisfies (1) for L the identity function. Therefore, the attack described in the first part of the proof applies to $\overline{F}^2$, and thus to F^2. $\square$

We demonstrate the impact of the attack by giving several example functions that fall in the categorization. We stress that the requirements of Proposition 1 are in fact solely requirements on f_2 and f_3; f_1 can be any function.

Suppose F^2 uses a linear finalization function f_3. Say, f_3 is defined as follows:

$$\begin{pmatrix} \mathsf{a}_{11} & \mathsf{a}_{12} & \mathsf{a}_{13} & \mathsf{a}_{14} & \mathsf{a}_{15} \\ \mathsf{a}_{21} & \mathsf{a}_{22} & \mathsf{a}_{23} & \mathsf{a}_{24} & \mathsf{a}_{25} \end{pmatrix} (u, v, w, c_1, c_2)^\top = (y, z)^\top,$$

[3] If k balls are thrown in l bins, the α fullest bins in total contain at least $\alpha k/l$ balls.

where addition and multiplication is done over the field $GF(2^n)$. Now, if $\mathsf{a}_{25} = 0$ we set $L = \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$ which corresponds to swapping y and z. If $\mathsf{a}_{25} \neq 0$, we set $L = \left(\begin{smallmatrix} 1 & -\mathsf{a}_{15}\mathsf{a}_{25}^{-1} \\ 0 & 1 \end{smallmatrix}\right)$, which corresponds to subtracting the second equation $\mathsf{a}_{15}\mathsf{a}_{25}^{-1}$ times from the first one.

The attack also covers designs whose finalization function f_3 rotates or shuffles its inputs, where one defines L so that the rotation gets undone. For instance, for MDC-2 f_3 is defined over $n/2$-bit words as $f_3(u^l, u^r, v^l, v^r, w^l, w^r; c_1^l, c_1^r, c_2^l, c_2^r) = (c_1^l \oplus w^l, c_2^r \oplus w^r, c_2^l \oplus w^l, c_1^r \oplus w^r)$, where u^l and u^r denote the left and right half of u, and it satisfies (1) for

$$L = \begin{pmatrix} 1\,0\,0\,0 \\ 0\,0\,0\,1 \\ 0\,0\,1\,0 \\ 0\,1\,0\,0 \end{pmatrix}.$$

Re-shuffling the output of f_3 can even be defined at bit level, in which case L is a $2^{2n} \times 2^{2n}$ permutation matrix.

In general, if f_3 is a sufficiently simple add-rotate-xor function, it is possible to derive a bijective L that makes (1) satisfied. This is possible by composing multiple bijective mappings L. Up to a degree, the attack also covers general non-linear finalization functions. However, it clearly does not cover all functions and it remains an open problem to either close this gap or to come with a (possibly impractical) F^2 compression function that provable achieves optimal collision resistance. One direction may be to start from the compression function with non-linear finalization f_3 by Jetchev et al. [11], for which collision resistance up to $2^{2n/3}$ queries is proven.

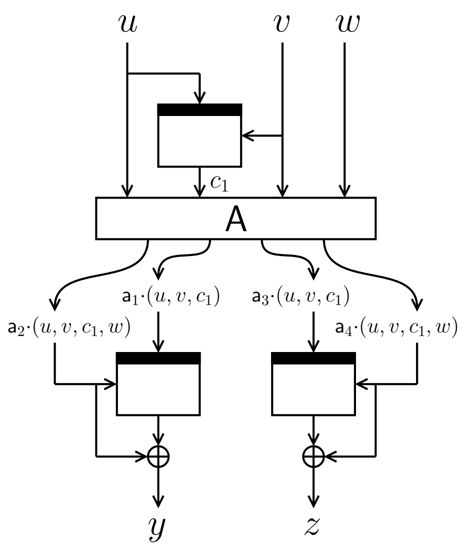
Remark 1. The original impossibility result for two-call double length hashing [25] was missing the condition (2). Consequently, it was possible to derive (contrived) designs invulnerable to the attack. For instance, if at step 3, for all α tuples $(u, v, w; c_1)$, f_2 outputs the same key k_2 but the messages m_2 are all distinct, then the outcomes c_2 are distinct for all α tuples. If, additionally, $g_2(u, v, w; c_1, c_2) = c_2$, then collisions happen with probability 0. Also if g_2 is a variant that only uses a few bits of $(u, v, w; c_1)$ to mask c_2, collisions happen with probability at most $\mathcal{O}(\alpha/2^n)$. The new condition (2) allows us to avoid these cases.

4 Double Length Hashing with Three E-calls

Motivated by the negative result of Section 3, we target the existence of double length hashing with three block cipher calls. We introduce a family of double length compression functions making three cipher calls that achieve asymptotically optimal 2^n collision resistance and preimage resistance significantly beyond the birthday bound (up to $2^{3n/2}$ queries). We note that, although the preimage bound is non-optimal, it is close to the generic bound dictated by the pigeonhole-birthday attack (Section 2).

Let $GF(2^n)$ be the field of order 2^n. We identify bit strings from $\{0,1\}^n$ and finite field elements in $GF(2^n)$ to define addition and scalar multiplication over $\{0,1\}^n$. In the family of double block length functions we propose in this section, the functions f_1, f_2, f_3, f_4 of Figure 3 will be linear functions over $GF(2^n)$. For two tuples $x = (x_1, \ldots, x_l)$ and $y = (y_1, \ldots, y_l)$ of elements $x_i, y_i \in \{0,1\}^n$, we define by $x \cdot y$ their inner product $\sum_{i=1}^{l} x_i y_i \in \{0,1\}^n$.

Before introducing the design, we first explain the fundamental consideration upon which the family is based. The security proofs of all DBL^{2n} functions known in the literature (cf. Table 1) crucially rely on the property that one block cipher evaluation defines the input to the other one. For DBL^{2n} functions this can easily be achieved: any block cipher evaluation can take as input the full $3n$-bit input state (u, v, w). Considering the class of functions DBL^n, and F^r of Figure 3 in particular, this cannot be achieved: one block cipher "processes" at most $2n$ out of $3n$ input



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. 4. The family of compression functions F^3_{A} where A is a 4×4 matrix as specified in the text. Arithmetics is done over $GF(2^n)$.

bits. In our design, we slightly relax this requirement, by requiring that any *two* block cipher evaluations define the input to the third one. Although from a technical point of view one may expect that this change causes optimal collision resistance to be harder or even impossible to be achieved, we will demonstrate that this is not the case due to new proof techniques employed to analyze the collision resistance.

Based on this key observation we propose the compression function design F^3_{A} of Figure 4. Here,

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

is a 4×4 matrix over $GF(2^n)$. Here, as $\mathsf{a}_{14} = \mathsf{a}_{34} = 0$, for simplicity we will write $\mathsf{a}_1(u,v,c_1) := \mathsf{a}_1(u,v,c_1,0)$ and $\mathsf{a}_3(u,v,c_1) := \mathsf{a}_3(u,v,c_1,0)$. Note that, provided A is invertible and $\mathsf{a}_{24},\mathsf{a}_{44} \neq 0$, any two block cipher evaluations of F^3_{A} define (the inputs of) the third one. For instance, evaluations of the second and third block cipher fix the vector $\mathsf{A}(u,v,c_1,w)^\top$, which by invertibility of A fixes (u,v,c_1,w) and thus the first block cipher evaluation. Evaluations of the first and second block cipher fix the inputs of the third block cipher as $\mathsf{a}_{24} \neq 0$. For the proofs of collision and preimage resistance, however, we will need to posit additional requirements on A. As we will explain, these requirements are easily satisfied.

In the remainder of this section, we state our results on the collision resistance of F^3_{A} in Section 5 and on the preimage resistance in Section 6.

5 Collision Resistance of F^3_{A}

We prove that, provided its underlying matrix A satisfies some simple conditions, F^3_{A} satisfies optimal collision resistance. In more detail, we pose the following requirements on A:

- A is invertible;
- $\mathsf{a}_{12},\mathsf{a}_{13},\mathsf{a}_{24},\mathsf{a}_{32},\mathsf{a}_{33},\mathsf{a}_{44} \neq 0$;
- $\mathsf{a}_{12} \neq \mathsf{a}_{32}$ and $\mathsf{a}_{13} \neq \mathsf{a}_{33}$.

We refer to the logical AND of these requirements as `colreq`.

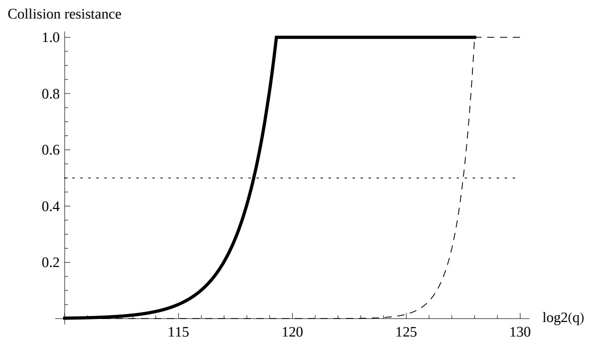


Fig. 5. For $n = 128$, the function $\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q)$ of (4) for $\varepsilon = 1/35$ and for the particular choice of values t_1, t_2 (solid line) and the optimal bound of $q(q+1)/2^{2n}$ (dashed line).

Theorem 1. *Let $E \xleftarrow{\$} Bloc(n)$. Suppose A satisfies `colreq`. Then, for any positive integral values t_1, t_2,*

$$\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q) \leq \frac{2t_2^2 q + 3t_2 q + 11q + 3t_1 t_2^2 + 7t_1 t_2}{2^n - q} + \frac{q^2}{t_1(2^n - q)} + 3 \cdot 2^n \left(\frac{eq}{t_2(2^n - q)}\right)^{t_2}. \qquad (4)$$

The proof is given in Section 5.1. The basic proof idea is similar to existing proofs in the literature (e.g. [27,42]) and is based on the usage of thresholds t_1, t_2. For increasing values of t_1, t_2 the first term of the bound increases, while the second two terms decrease. Although the proof derives basic proof principles from literature, for the technical part we deviate from existing proof techniques in order to get a bound that is "as tight as possible". In particular, we introduce the usage of wish lists in the context of collisions, an approach that allows for significantly better bounds. Wish lists have been introduced by Lee et al. [17,19] and Armknecht et al. [3] for the preimage resistance analysis of DBL^{2n} functions, but they have never been used for collision resistance as there never was a need to do so. Our analysis relies on this proof methodology, but as for collisions more block cipher evaluations are involved (one collision needs six block cipher calls while a preimage requires three) this makes the analysis more technical and delicate.

The goal now is to find a good threshold between the first term and the latter two terms of (4). To this end, let $\varepsilon > 0$ be any parameter. We put $t_1 = q$ and $t_2 = 2^{n\varepsilon}$ (we can assume t_2 to be integral). Then, the bound simplifies to

$$\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q) \leq \frac{5 \cdot 2^{2n\varepsilon} q + 10 \cdot 2^{n\varepsilon} q + 12q}{2^n - q} + 3 \cdot 2^n \left(\frac{eq}{2^{n\varepsilon}(2^n - q)}\right)^{2^{n\varepsilon}}.$$

From this, we find that for any $\varepsilon > 0$ we have

$$\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(2^n/2^{3n\varepsilon}) \to 0 \text{ for } n \to \infty\,.$$

Hence, given that the bound holds for any ε, this implies that the F^3_{A} compression function achieves close to optimal 2^n collision security for $n \to \infty$, asymptotically. For $n = 128$, the bound on $\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}$ is depicted in Figure 5, where we take a slightly different value for t_1 to achieve a better bound (to be precise, $t_1 = q/(3t_2^2 + 7t_2)^{1/2}$). The collision advantage hits $1/2$ for $\log_2 q \approx 118.3$, relatively close to the threshold 127.5 for $q(q+1)/2^{2n}$. For larger values of n this gap approaches 0.

5.1 Proof of Theorem 1

The proof of collision resistance of F^3_{A} follows the basic spirit of [27], but crucially differs in the way the probability bounds are computed. A new approach here is the usage of wish lists. While

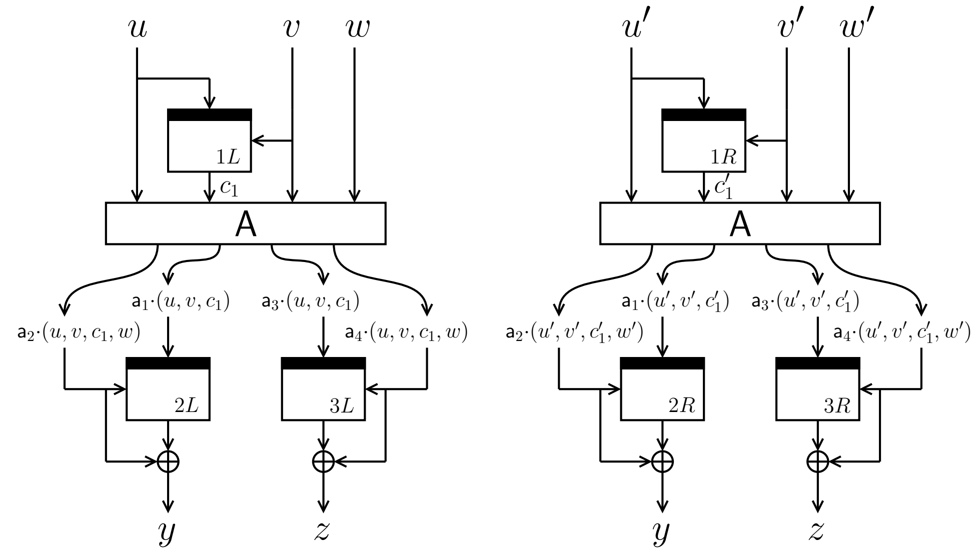


Fig. 6. Configuration $\mathsf{coll}(Q)$. The configuration is satisfied if Q contains six (possibly the same) queries that satisfy this setting. We require $(u,v,w) \neq (u',v',w')$.

the idea of wish lists is not new—it has been introduced by Lee et al. [17, 19] and Armknecht et al. [3] for double block length compression functions, and used by Mennink [27] for the analysis of MDC-4—in these works wish lists are solely used for the analysis of preimage resistance rather than collision resistance. Given that in a collision more block cipher evaluations are involved, the analysis becomes more complex. At a high level, wish lists rely on the idea that in order to find a collision, the adversary must at some point make a query that "completes this collision" together with some other queries already in the query history. Wish lists keep track of such query tuples, and the adversary's goal is to ever obtain a query tuple that is in such wish list. A more technical treatment can be found in the proof of Lemma 1.

We consider any adversary that has query access to its oracle E and makes q queries stored in a query history Q_q. Its goal is to find a collision for F^3_{A}, in which it by definition only succeeds if it obtains a query history Q_q that satisfies configuration $\mathsf{coll}(Q_q)$ of Figure 6. Formally, $\mathsf{coll}(Q_q)$ is set if for some $(u,v,w) \neq (u',v',w')$ there exists query tuples $(k_1,m_1,c_1),(k_2,m_2,c_2),(k_3,m_3,c_3) \in Q_q$ and $(k_1',m_1',c_1'),(k_2',m_2',c_2'),(k_3',m_3',c_3') \in Q_q$ such that:

1. $(k_1,m_1) = (u,v)$ and $(k_2,m_2,k_3,m_3)^\top = \mathsf{A}(u,v,c_1,w)^\top$;
2. $(k_1',m_1') = (u',v')$ and $(k_2',m_2',k_3',m_3')^\top = \mathsf{A}(u',v',c_1',w')^\top$;
3. $m_2 \oplus c_2 = m_2' \oplus c_2'$;
4. $m_3 \oplus c_3 = m_3' \oplus c_3'$.

This means,

$$\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q) = \mathbf{Pr}\left(\mathsf{coll}(Q_q)\right). \tag{5}$$

Above set of tuples is also called a "solution" to the configuration. For the sake of readability of the proof, we label the block cipher positions in Figure 6 as follows. In the left F^3_{A} evaluation (on input (u,v,w)), the block ciphers are labeled $1L$ (the one on input (u,v)), $2L$ (the bottom left one), and $3L$ (the bottom right one). The block ciphers for the right F^3_{A} evaluation are labeled $1R, 2R, 3R$ in a similar way. When we say "a query $1L$", we refer to a query that in a collision occurs at position $1L$.

For the analysis of $\mathbf{Pr}\left(\mathsf{coll}(Q_q)\right)$ we introduce an auxiliary event $\mathsf{aux}(Q_q)$. Let $t_1, t_2 > 0$ be any integral values. We define $\mathsf{aux}(Q_q) = \mathsf{aux}_1(Q_q) \vee \cdots \vee \mathsf{aux}_4(Q_q)$, where

$$\begin{aligned}
\mathsf{aux}_1(Q_q):&\quad \left|\left\{(k_i, m_i, c_i), (k_j, m_j, c_j) \in Q_q \;:\; i \neq j \;\wedge\; m_i + c_i = m_j + c_j\right\}\right| > t_1\,;\\
\mathsf{aux}_2(Q_q):&\quad \max\nolimits_{Z \in \{0,1\}^n} \left|\left\{(k_i, m_i, c_i) \in Q_q \;:\; \mathsf{a}_1 \cdot (k_i, m_i, c_i) = Z\right\}\right| > t_2\,;\\
\mathsf{aux}_3(Q_q):&\quad \max\nolimits_{Z \in \{0,1\}^n} \left|\left\{(k_i, m_i, c_i) \in Q_q \;:\; \mathsf{a}_3 \cdot (k_i, m_i, c_i) = Z\right\}\right| > t_2\,;\\
\mathsf{aux}_4(Q_q):&\quad \max\nolimits_{Z \in \{0,1\}^n} \left|\left\{(k_i, m_i, c_i) \in Q_q \;:\; m_i + c_i = Z\right\}\right| > t_2\,.
\end{aligned}$$

By basic probability theory, we obtain for (5):

$$\mathbf{Pr}\left(\mathsf{coll}(Q_q)\right) \leq \mathbf{Pr}\left(\mathsf{coll}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) + \mathbf{Pr}\left(\mathsf{aux}(Q_q)\right). \tag{6}$$

We start with the analysis of $\mathbf{Pr}\left(\mathsf{coll}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right)$. For obtaining a query history that fulfills configuration $\mathsf{coll}(Q_q)$, it may be the case that a query appears at multiple positions. For instance, the queries at positions $1L$ and $2R$ are the same. We split the analysis of $\mathsf{coll}(Q_q)$ into essentially all different possible cases, but we do this in two steps. In the first step, we make a distinction among the cases a query occurs in both words at the same position. For binary $\alpha_1, \alpha_2, \alpha_3$, we define by $\mathsf{coll}_{\alpha_1\alpha_2\alpha_3}(Q)$ the configuration $\mathsf{coll}(Q)$ of Figure 6 restricted to

$$\begin{aligned}
\alpha_1 = 1 \iff 1L = 1R \iff (k_1, m_1, c_1) = (k_1', m_1', c_1')\,,\\
\alpha_2 = 1 \iff 2L = 2R \iff (k_2, m_2, c_2) = (k_2', m_2', c_2')\,,\\
\alpha_3 = 1 \iff 3L = 3R \iff (k_3, m_3, c_3) = (k_3', m_3', c_3')\,.
\end{aligned}$$

In other words, a solution to $\mathsf{coll}(Q_q)$ is a solution to $\mathsf{coll}_{\alpha_1\alpha_2\alpha_3}(Q_q)$ if the queries at positions κL and κR are the same if and only if $\alpha_\kappa = 1$. Moreover, a solution to $\mathsf{coll}_{\alpha_1\alpha_2\alpha_3}(Q_q)$ for $\alpha_1, \alpha_2, \alpha_3 \in \{0,1\}$ is a solution to $\mathsf{coll}(Q_q)$. Thus, $\mathsf{coll}(Q_q) \Rightarrow \bigvee_{\alpha_1,\alpha_2,\alpha_3 \in \{0,1\}} \mathsf{coll}_{\alpha_1\alpha_2\alpha_3}(Q_q)$, and from (5-6) we obtain the following bound on $\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q)$:

$$\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q) \leq \sum_{\alpha_1,\alpha_2,\alpha_3 \in \{0,1\}} \mathbf{Pr}\left(\mathsf{coll}_{\alpha_1\alpha_2\alpha_3}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) + \mathbf{Pr}\left(\mathsf{aux}(Q_q)\right). \tag{7}$$

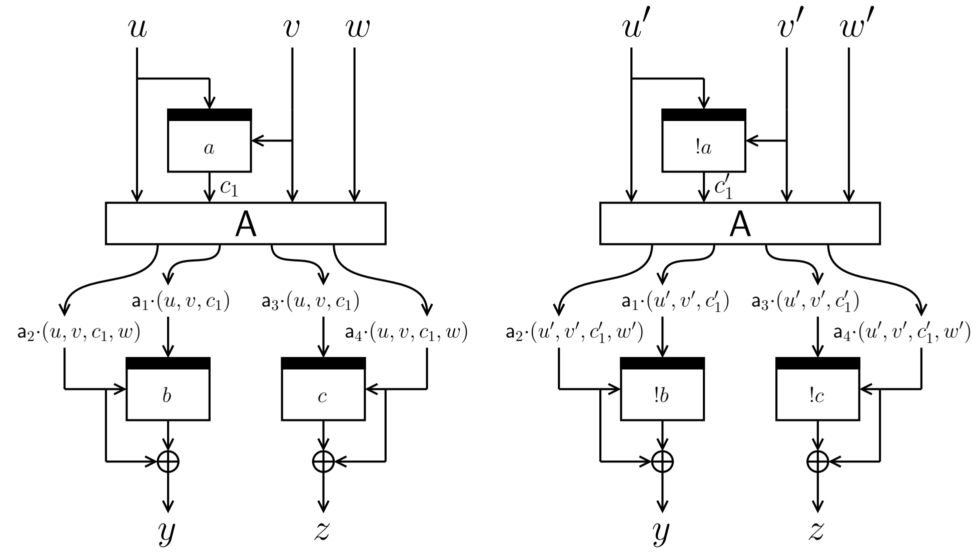
Note that we did not make a distinction yet whether or not a query occurs at two "different" positions (e.g. at positions $1L$ and $2R$). These cases are analyzed for each of the sub-configurations separately, as becomes clear later. Probabilities $\mathbf{Pr}\left(\mathsf{coll}_{\alpha_1\alpha_2\alpha_3}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right)$ for the different choices of $\alpha_1, \alpha_2, \alpha_3$ are bounded in Lemmas 1-4. The proofs are rather similar, and we only bound the probability on $\mathsf{coll}_{000}(Q_q)$ in full detail (Lemma 1). A bound on $\mathbf{Pr}\left(\mathsf{aux}(Q_q)\right)$ is derived in Lemma 5.

Lemma 1. $\mathbf{Pr}\left(\mathsf{coll}_{000}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) \leq \frac{t_2 q + 7q + 3t_1 t_2^2 + 3t_1 t_2}{2^n - q}$.

Proof. Sub-configuration $\mathsf{coll}_{000}(Q_q)$ is given in Figure 7. The block cipher queries at positions a and $!a$ are required to be different, and so are the ones at positions $b, !b$ and $c, !c$.

We consider the probability of the adversary finding a solution to configuration $\mathsf{coll}_{000}(Q_q)$ such that Q_q satisfies $\neg\mathsf{aux}(Q_q)$. Consider the ith query, for $i \in \{1, \ldots, q\}$. We say this query is a winning query if it makes $\mathsf{coll}_{000}(Q_i) \wedge \neg\mathsf{aux}(Q_i)$ satisfied for some set of other queries in the query history Q_{i-1}. We can assume the ith query does not make $\mathsf{aux}(Q_i)$ satisfied: if it would, by definition it cannot be a winning query.

Recall that, although we narrowed down the number of possible positions for a winning query to occur (in $\mathsf{coll}_{000}(Q_q)$ it cannot occur at both $1L$ and $1R$, at both $2L$ and $2R$, or at both $3L$ and $3R$), it may still be the case that such a query appears at multiple "different" positions, e.g. $1L$ and $2R$. Note that by construction, a winning query can appear at at most three block cipher positions of Figure 7. In total, there are 26 sets of positions at which the winning query



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

Fig. 7. Configuration $\mathsf{coll}_{000}(Q)$. We require $(u,v,w) \neq (u',v',w')$.

can appear at at the same time. Discarding symmetric cases caused by swapping (u,v,w) and (u',v',w'), one identifies the following 13 sets of positions:

$$\begin{array}{llll}
\mathcal{S}_1 = \{1L\}\,, & \mathcal{S}_4 = \{1L,2L\}\,, & \mathcal{S}_7 = \{1L,2R\}\,, & \mathcal{S}_{10} = \{1L,2L,3L\}\,,\\
\mathcal{S}_2 = \{2L\}\,, & \mathcal{S}_5 = \{1L,3L\}\,, & \mathcal{S}_8 = \{1L,3R\}\,, & \mathcal{S}_{11} = \{1L,2L,3R\}\,,\\
\mathcal{S}_3 = \{3L\}\,, & \mathcal{S}_6 = \{2L,3L\}\,, & \mathcal{S}_9 = \{2L,3R\}\,, & \mathcal{S}_{12} = \{1L,2R,3L\},\\
 & & & \mathcal{S}_{13} = \{1L,2R,3R\}\,.
\end{array}$$

Note that there are many more symmetric cases among these, but we are not allowed to discard those as these may result in effectively different collisions. For $j = 1,\ldots,13$ we denote by $\mathsf{coll}_{000:\mathcal{S}_j}(Q)$ configuration $\mathsf{coll}_{000}(Q)$ with the restriction that the winning query *must* appear at the positions in $\mathcal{S}_j$. By basic probability theory,

$$\mathbf{Pr}\left(\mathsf{coll}_{000}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) \leq \sum_{j=1}^{13} \mathbf{Pr}\left(\mathsf{coll}_{000:\mathcal{S}_j}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right). \tag{8}$$

$\mathsf{coll}_{000:\mathcal{S}_1}(Q_q)$. Rather than considering the success probability of the ith query, and then sum over $i = 1,\ldots,q$ (as is done in the analysis of [6–8,11,14,18,27,34,41], hence all collision security proofs of Table 1), the approach in this proof is to focus on "wish lists". Intuitively, a wish list is a continuously updated sequence of query tuples that would make configuration $\mathsf{coll}_{000:\mathcal{S}_j}(Q_q)$ satisfied. During the attack of the adversary, we maintain an initially empty wish list $\mathcal{W}_{\mathcal{S}_1}$. Consider configuration $\mathsf{coll}_{000}(Q)$ with the query at position $\mathcal{S}_1 = \{1L\}$ left out (see Figure 8). Note that every set of five query tuples (some of which may be the same) that satisfy configuration $\mathsf{coll}_{000:\mathcal{S}_1}(Q_q)$, uniquely define a tuple (u,v,c_1). This tuple is, indeed, uniquely determined by the queries at $2L$ and $3L$ by invertibility of A. Concretely, this means that if this exact query will ever be made, this will be a winning query (due to the corresponding set of five queries in Q_q that satisfy $\mathsf{coll}_{000:\mathcal{S}_1}(Q_q)$). The idea of the wish list is to maintain a list of all such query tuples. We remark that, technically, there may be different solutions to configuration $\mathsf{coll}_{000:\mathcal{S}_1}(Q_q)$ that define the same wish, but this is not a problem: it simply implies that the number of wishes is *at most* the number of solutions to configuration $\mathsf{coll}_{000:\mathcal{S}_1}(Q_q)$ (and in the end we will bound the size of $\mathcal{W}_{\mathcal{S}_1}$ by the number of solutions to $\mathsf{coll}_{000:\mathcal{S}_1}(Q_q)$). The wish list $\mathcal{W}_{\mathcal{S}_1}$ will be maintained continuously: suppose the adversary makes a query, and denote by Q the updated query history. Then, identify all sets of five query tuples in Q that satisfy configuration $\mathsf{coll}_{000:\mathcal{S}_1}(Q_q)$ and such

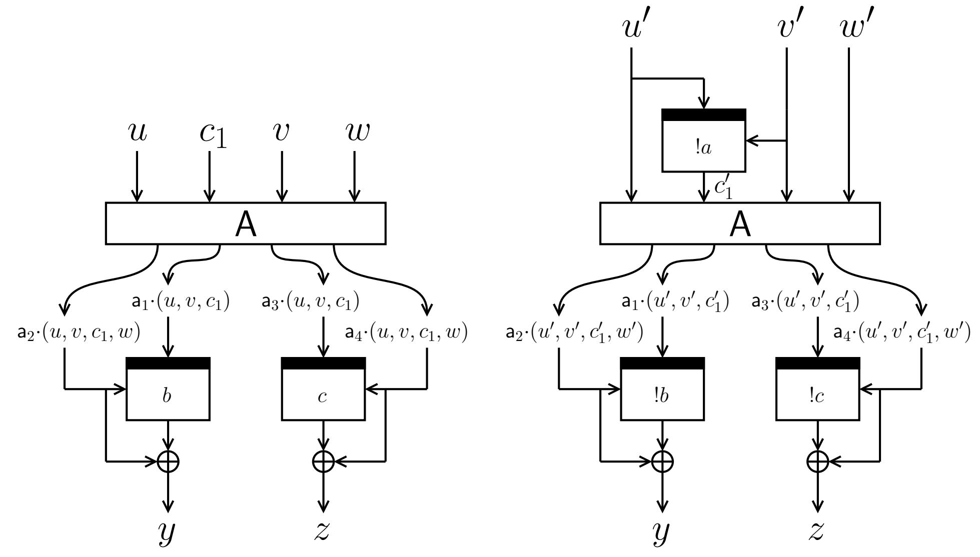

u
c_1
v
w
A
$a_1 \cdot (u, v, c_1)$
$a_3 \cdot (u, v, c_1)$
$a_2 \cdot (u, v, c_1, w)$
$a_4 \cdot (u, v, c_1, w)$
b
c
y
z
u'
v'
w'
$!a$
c_1'
A
$a_1 \cdot (u', v', c_1')$
$a_3 \cdot (u', v', c_1')$
$a_2 \cdot (u', v', c_1', w')$
$a_4 \cdot (u', v', c_1', w')$
$!b$
$!c$
y
z

Fig. 8. Configuration $\mathsf{coll}_{000:\mathcal{S}_1}(Q)$. We require $(u, v, w) \neq (u', v', w')$.

that the new query appears *at least* once. For each of these solutions to the configuration, add the corresponding (u, v, c_1) to the wish list.

As we have restricted to the case the winning query only occurring at the position of $\mathcal{S}_1$, we can assume a query never adds itself to a wish list.[4] Clearly, in order to find a collision for F^3_{A} in this sub-configuration, the adversary needs to wish for a query at least once. Suppose the adversary makes a query $E(k, m)$ where $(k, m, c) \in \mathcal{W}_{\mathcal{S}_1}$ for some c. We say that (k, m, c) is wished for, and the wish is granted if the query response equals c. As the adversary makes at most q queries, such wish is granted with probability at most $1/(2^n - q)$, and the same for inverse queries. By construction, each element from $\mathcal{W}_{\mathcal{S}_1}$ can be wished for only once, and we find that the adversary finds a collision with probability at most $\frac{|\mathcal{W}_{\mathcal{S}_1}|}{2^n - q}$.

Now, it suffices to upper bound the size of the wish list $\mathcal{W}_{\mathcal{S}_1}$ after q queries, and to this end we bound the number of solutions to the configuration of Figure 8. By $\neg\mathsf{aux}_1(Q_q)$, the configuration has at most t_1 choices for $2L, 2R$. For any such choice, by $\neg\mathsf{aux}_2(Q_q)$ we have at most t_2 choices for $1R$. Any such choice fixes w' (as $\mathsf{a}_{24} \neq 0$), and thus the query at position $3R$, and consequently z. By $\neg\mathsf{aux}_4(Q_q)$, we have at most t_2 choices for $3L$. The queries at positions $2L$ and $3L$ uniquely fix (u, v, c_1) by invertibility of A. We find $|\mathcal{W}_{\mathcal{S}_1}| \leq t_1 t_2^2$, and hence in this setting a collision is found with probability at most $t_1 t_2^2/(2^n - q)$.

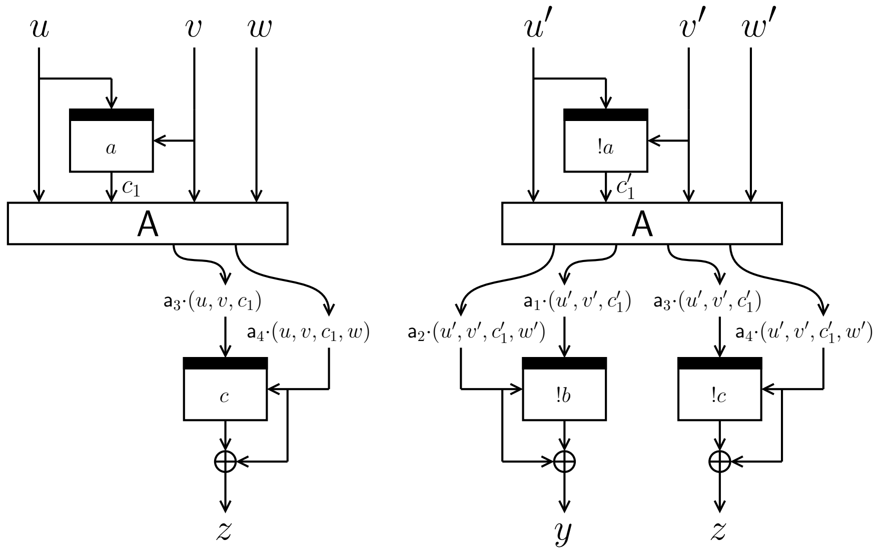
$\mathsf{coll}_{000:\mathcal{S}_j}(Q_q)$ for $j = 2, 3$. Both cases are the same by symmetry, and we consider $\mathcal{S}_2$ only.

The analysis is similar to the one for $\mathcal{S}_1$, and we only present the computation of the bound on the wish list $\mathcal{W}_{\mathcal{S}_2}$ after q queries. Consider configuration $\mathsf{coll}_{000}(Q)$ with the query at position $\mathcal{S}_2 = \{2L\}$ left out (see Figure 9). By $\neg\mathsf{aux}_1(Q_q)$, the configuration has at most t_1 choices for $3L, 3R$. For any such choice, by $\neg\mathsf{aux}_3(Q_q)$ we have at most t_2 choices for $1L$ and at most t_2 choices for $1R$. Any such choice fixes w' (as $\mathsf{a}_{44} \neq 0$), and thus the query at position $2R$, and consequently y. The query at position $1L$ fixes (u, v, c_1) and together with query $3L$ this fixes w. Any choice of queries thus uniquely fixes $(\mathsf{a}_1 \cdot (u, v, c_1), \mathsf{a}_2 \cdot (u, v, c_1, w), y - \mathsf{a}_2 \cdot (u, v, c_1, w))$. We find $|\mathcal{W}_{\mathcal{S}_2}| \leq t_1 t_2^2$, and hence in this setting a collision is found with probability at most $t_1 t_2^2/(2^n - q)$.

$\mathsf{coll}_{000:\mathcal{S}_j}(Q_q)$ for $j = 4, 5$. Both cases are the same by symmetry, and we consider $\mathcal{S}_4$ only.

The analysis differs from the ones before, because in this setting the success probability can be analyzed more easily. As the winning query (k, m, c) should appear at positions $1L$ and $2L$, we require it to satisfy $k = \mathsf{a}_1 \cdot (k, m, c)$. Any query satisfies this equation with probability at most

[4] A winning query that would appear at multiple positions is counted in $\mathsf{coll}_{000:\mathcal{S}_j}(Q_q)$ for some other set $\mathcal{S}_j$.



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

Fig. 9. Configuration $\mathsf{coll}_{000:\mathcal{S}_2}(Q)$. We require $(u,v,w) \neq (u',v',w')$.

$1/(2^n - q)$ (as $\mathsf{a}_{12}, \mathsf{a}_{13} \neq 0$). As the adversary makes at most q queries, in this setting a collision is found with probability at most $q/(2^n - q)$.

$\mathsf{coll}_{000:\mathcal{S}_6}(Q_q)$. By construction, there must be a query (k,m,c) in the query history (corresponding to position $1L$) that satisfies $\mathsf{a}_1\cdot(k,m,c) = \mathsf{a}_3\cdot(k,m,c)$. So the problem shifts to bounding the probability that the adversary ever finds a query (k,m,c) that satisfies this equation. As $\mathsf{a}_{12} \neq \mathsf{a}_{32}$ and $\mathsf{a}_{13} \neq \mathsf{a}_{33}$, the adversary never obtains such query except with probability at most $q/(2^n - q)$ (for the same reasoning as for $\mathcal{S}_4$).

$\mathsf{coll}_{000:\mathcal{S}_j}(Q_q)$ for $j = 7, 8$. Both cases are the same by symmetry, and we consider $\mathcal{S}_7$ only.

The analysis is similar to the one for $\mathcal{S}_1$, and we only present the definition of the wish list $\mathcal{W}_{\mathcal{S}_7}$ and the computation of the bound on $\mathcal{W}_{\mathcal{S}_7}$. Consider configuration $\mathsf{coll}_{000}(Q)$ with the queries at positions $\mathcal{S}_7 = \{1L, 2R\}$ left out (see Figure 10). For any set of four queries that satisfy the configuration at positions $\{2L, 3L, 1R, 3R\}$, the tuple (u, v, c_1) is added to $\mathcal{W}_{\mathcal{S}_7}$. By construction, this tuple is required to satisfy $(u, v, c_1) = (\mathsf{a}_1\cdot(u',v',c_1'), \mathsf{a}_2\cdot(u',v',c_1',w'), y - \mathsf{a}_2\cdot(u',v',c_1',w'))$.

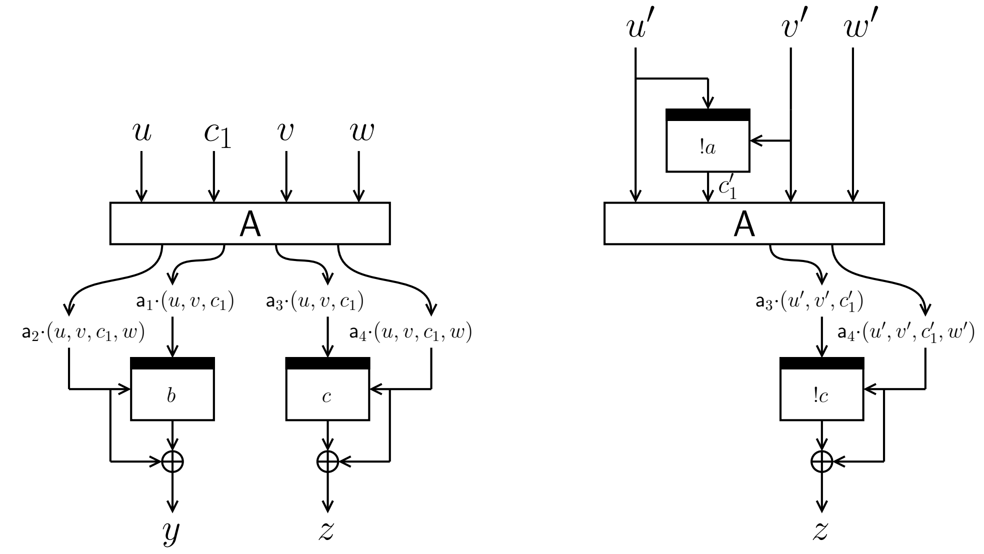
We upper bound the number of solutions to the configuration of Figure 10 after q queries. By $\neg\mathsf{aux}_1(Q_q)$, the configuration has at most t_1 choices for $3L, 3R$. For any such choice, by $\neg\mathsf{aux}_3(Q_q)$ we have at most t_2 choices for $1R$. The queries at positions $1R$ and $3R$ uniquely fix (u',v',c_1',w') (as $\mathsf{a}_{44} \neq 0$), and thus the values $u = \mathsf{a}_1\cdot(u',v',c_1')$ and $v = \mathsf{a}_2\cdot(u',v',c_1',w')$. Together with the query $3L$ this fixes c_1 (as $\mathsf{a}_{33} \neq 0$). Any choice of queries thus uniquely fixes (u, v, c_1). We find $|\mathcal{W}_{\mathcal{S}_7}| \leq t_1 t_2$, and hence in this setting a collision is found with probability at most $t_1 t_2/(2^n - q)$.

$\mathsf{coll}_{000:\mathcal{S}_9}(Q_q)$. The analysis is similar to the one for $\mathcal{S}_1$, and we only present the definition of the wish list $\mathcal{W}_{\mathcal{S}_9}$ and the computation of the bound on $\mathcal{W}_{\mathcal{S}_9}$. Consider configuration $\mathsf{coll}_{000}(Q)$ with the queries at positions $\mathcal{S}_9 = \{2L, 3R\}$ left out (see Figure 11). For any set of queries that satisfy this configuration at positions $\{1L, 3L, 1R, 2R\}$, the tuple

$$\begin{aligned}&(\mathsf{a}_1\cdot(u,v,c_1), \mathsf{a}_2\cdot(u,v,c_1,w), y - \mathsf{a}_2\cdot(u,v,c_1,w))\\ &= (\mathsf{a}_3\cdot(u',v',c_1'), \mathsf{a}_4\cdot(u',v',c_1',w'), z - \mathsf{a}_4\cdot(u',v',c_1',w'))\,.\end{aligned}$$

is added to $\mathcal{W}_{\mathcal{S}_9}$. Note that we particularly require $y = z$.

We split this wish list up into two sets: $\mathcal{W}_{\mathcal{S}_9}^{=}$ contains only wishes of which the corresponding queries at positions $3L, 2R$ are the same, and $\mathcal{W}_{\mathcal{S}_9}^{\neq}$ contains only wishes of which the corresponding queries at positions $3L, 2R$ are different. Note that by construction, a wish may occur in both



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

Fig. 10. Configuration $\mathsf{coll}_{000:\mathcal{S}_7}(Q)$. We require $(u,v,w) \neq (u',v',w')$ and $(u,v,c_1) = (\mathsf{a}_1 \cdot (u',v',c_1'), \mathsf{a}_2\cdot(u',v',c_1',w'), y - \mathsf{a}_2\cdot(u',v',c_1',w'))$.

sets, but this does not invalidate the security analysis: it only results in a slightly worse bound. As before, each element from $\mathcal{W}_{\mathcal{S}_9}$ can be wished for only once, and we find that the adversary finds a collision with probability at most

$$\frac{|\mathcal{W}^{=}_{\mathcal{S}_9}| + |\mathcal{W}^{\neq}_{\mathcal{S}_9}|}{2^n - q}\,.$$

We upper bound the number of solutions to the configuration of Figure 11, restricted to either $3L = 2R$ and $3L \neq 2R$, after q queries.

- $\boldsymbol{3L = 2R}$**.** We have at most q choices for $3L = 2R$. For any such choice, by $\neg\mathsf{aux}_3(Q_q)$ we have at most t_2 choices for $1L$. The queries at positions $1L$ and $3L$ uniquely fix (u,v,c_1,w). The query $3L = 2R$ fixes $y = z$. Any choice of queries thus uniquely fixes $(\mathsf{a}_1\cdot(u,v,c_1), \mathsf{a}_2\cdot(u,v,c_1,w), y - \mathsf{a}_2\cdot(u,v,c_1,w))$. We find $|\mathcal{W}^{=}_{\mathcal{S}_9}| \leq t_2 q$.
- $\boldsymbol{3L \neq 2R}$**.** As we require $y = z$, by $\neg\mathsf{aux}_1(Q_q)$ the configuration has at most t_1 choices for $3L, 2R$. The remainder is the same, and we find $|\mathcal{W}^{\neq}_{\mathcal{S}_9}| \leq t_1 t_2$.

Hence, in this setting a collision is found with probability at most $(t_1 t_2 + t_2 q)/(2^n - q)$.

$\boldsymbol{\mathsf{coll}_{000:\mathcal{S}_j}(Q_q)}$ **for** $\boldsymbol{j = 10, 11, 12}$**.** For these cases, the analysis for $\mathcal{S}_4$ directly applies.

$\boldsymbol{\mathsf{coll}_{000:\mathcal{S}_{13}}(Q_q)}$**.** For this case, the analysis for $\mathcal{S}_6$ directly applies.

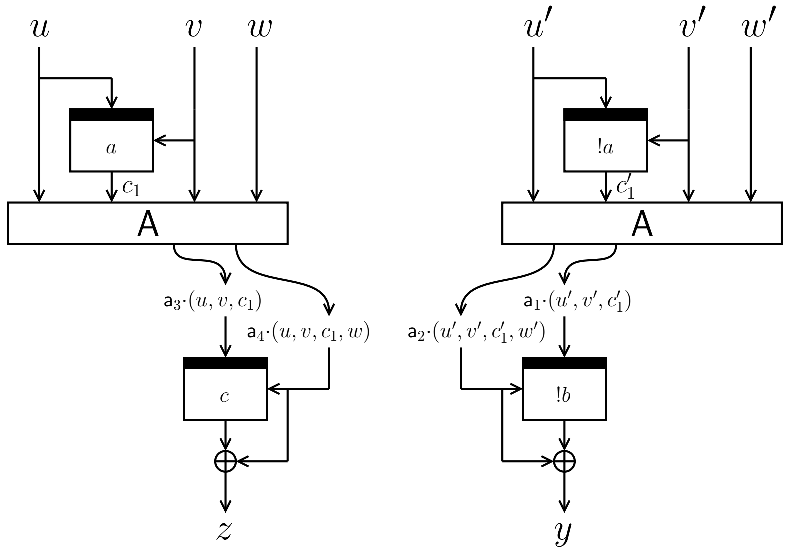
The proof is now completed by adding all bounds in accordance with (8). $\square$

Lemma 2. $\mathbf{Pr}\left(\mathsf{coll}_{100}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) \leq \frac{2q + 2t_1 t_2}{2^n - q}$.

Proof. Sub-configuration $\mathsf{coll}_{100}(Q_q)$ is given in Figure 12. Note that here we in particular have $(u,v,c_1) = (u',v',c_1')$ as $1L = 1R$.

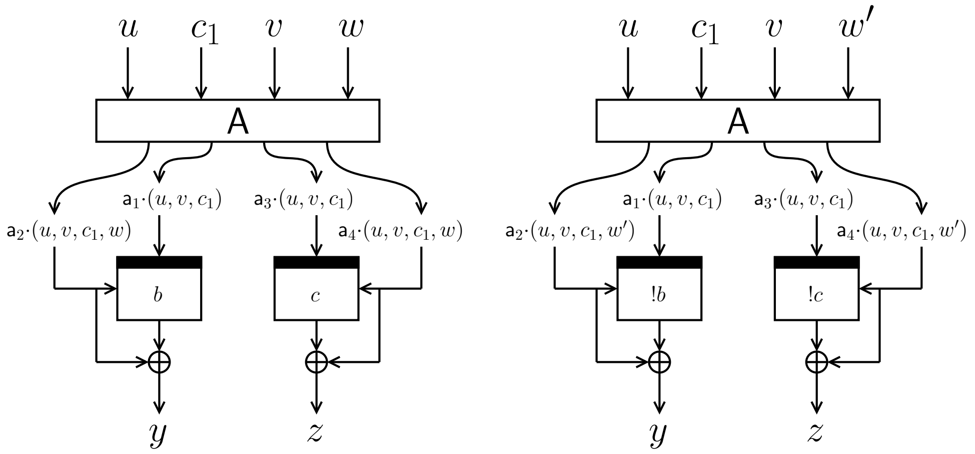
The approach is similar to the one for Lemma 1 and we only highlight the structural differences. Discarding symmetric cases caused by swapping w and w', one identifies 4 sets of positions in which the winning query can appear:

$$\mathcal{S}_1 = \{2L\}\,, \qquad \mathcal{S}_2 = \{3L\}\,, \qquad \mathcal{S}_3 = \{2L, 3L\}\,, \qquad \mathcal{S}_4 = \{2L, 3R\}\,.$$



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

Fig. 11. Configuration $\mathsf{coll}_{000:\mathcal{S}_9}(Q)$. We require $(u,v,w) \neq (u',v',w')$ and $(\mathsf{a}_1\cdot(u,v,c_1), \mathsf{a}_2\cdot(u,v,c_1,w), y-\mathsf{a}_2\cdot(u,v,c_1,w)) = (\mathsf{a}_3\cdot(u',v',c_1'), \mathsf{a}_4\cdot(u',v',c_1',w'), z-\mathsf{a}_4\cdot(u',v',c_1',w'))$.



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

Fig. 12. Configuration $\mathsf{coll}_{100}(Q)$. We require $w \neq w'$.

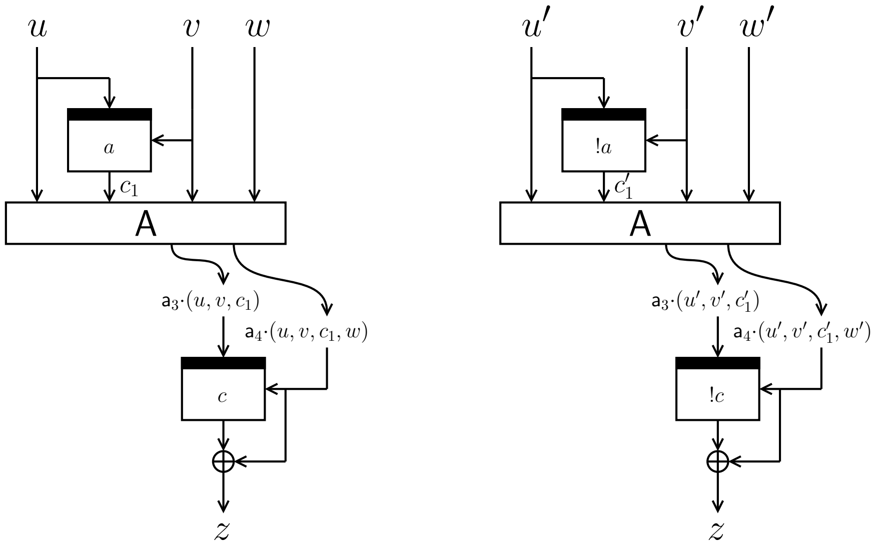
As before, we find

$$\mathbf{Pr}\left(\mathsf{coll}_{100}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) \leq \sum_{j=1}^{4} \mathbf{Pr}\left(\mathsf{coll}_{100:\mathcal{S}_j}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right). \tag{9}$$

$\mathsf{coll}_{100:\mathcal{S}_j}(Q_q)$ for $j = 1, 2$. Both cases are the same by symmetry, and we consider $\mathcal{S}_1$ only.

Consider configuration $\mathsf{coll}_{100}(Q)$ with the query at position $\mathcal{S}_1 = \{2L\}$ left out. By $\neg\mathsf{aux}_1(Q_q)$, the configuration has at most t_1 choices for $3L, 3R$. For any such choice, by $\neg\mathsf{aux}_3(Q_q)$ we have at most t_2 choices for $1L = 1R$. (Note that the query at $1L = 1R$ may be made *after* the winning query. This is because in this case "winning query" refers to a winning query for configuration $\mathsf{coll}_{100}(Q_q)$. We stress that this does not invalidate the security analysis.) Any such choice fixes u, v, c_1, w' (as $\mathsf{a}_{44} \neq 0$), and thus the query at position $2R$, and consequently y. The query at position $1L$ fixes (u, v, c_1) and together with query $3L$ this fixes w. Any choice of queries thus uniquely fixes $(\mathsf{a}_1\cdot(u,v,c_1), \mathsf{a}_2\cdot(u,v,c_1,w), y-\mathsf{a}_2\cdot(u,v,c_1,w))$. We find $|\mathcal{W}_{\mathcal{S}_1}| \leq t_1t_2$, and hence in this setting a collision is found with probability at most $t_1t_2/(2^n - q)$.

$\mathsf{coll}_{100:\mathcal{S}_3}(Q_q)$. By construction, there must be a query (k, m, c) in the query history (corresponding to position $1L$) that satisfies $\mathsf{a}_1\cdot(k,m,c) = \mathsf{a}_3\cdot(k,m,c)$. So the problem shifts to bounding the



u
v
w
a
c_1
A
$\mathsf{a}_3\cdot(u,v,c_1)$
$\mathsf{a}_4\cdot(u,v,c_1,w)$
c
z
u'
v'
w'
$!a$
c_1'
A
$\mathsf{a}_3\cdot(u',v',c_1')$
$\mathsf{a}_4\cdot(u',v',c_1',w')$
$!c$
z

Fig. 13. Configuration $\mathsf{coll}_{010}(Q)$. We require $(u,v,w) \neq (u',v',w')$, $\mathsf{a}_1\cdot(u,v,c_1) = \mathsf{a}_1\cdot(u',v',c_1')$, and $\mathsf{a}_3\cdot(u,v,c_1,w) = \mathsf{a}_3\cdot(u',v',c_1',w')$.

probability that the adversary ever finds a query (k,m,c) that satisfies this equation. As $\mathsf{a}_{12} \neq \mathsf{a}_{32}$ and $\mathsf{a}_{13} \neq \mathsf{a}_{33}$, the adversary never obtains such query except with probability at most $q/(2^n - q)$.

$\mathbf{coll_{100:\mathcal{S}_4}(Q_q)}$. As in the current case we have $(u,v,c_1) = (u',v',c_1')$, the approach for $\mathcal{S}_3$ applies.

The proof is now completed by adding all bounds in accordance with (9). $\square$

Lemma 3. $\mathbf{Pr}\left(\mathsf{coll}_{\alpha_1\alpha_2\alpha_3}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) \leq \frac{t_2^2 q + t_2 q + q + t_1 t_2}{2^n - q}$ *for* $\alpha_1\alpha_2\alpha_3 \in \{010, 001\}$.

Proof. Both cases are the same by symmetry, and we consider $\alpha_1\alpha_2\alpha_3 = 010$ only. Subconfiguration $\mathsf{coll}_{010}(Q)$ is given in Figure 13. Note that here we in particular have $\mathsf{a}_1\cdot(u,v,c_1) = \mathsf{a}_1\cdot(u',v',c_1')$ and $\mathsf{a}_3\cdot(u,v,c_1,w) = \mathsf{a}_3\cdot(u',v',c_1',w')$.

The approach is similar to the one for Lemma 1 and we only highlight the structural differences. Discarding symmetric cases caused by swapping (u,v,w) and (u',v',w'), one identifies 4 sets of positions in which the winning query can appear:

$$\mathcal{S}_1 = \{1L\}\,, \qquad \mathcal{S}_2 = \{3L\}\,, \qquad \mathcal{S}_3 = \{1L, 3L\}\,, \qquad \mathcal{S}_4 = \{1L, 3R\}\,.$$

As before, we find

$$\mathbf{Pr}\left(\mathsf{coll}_{010}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) \leq \sum_{j=1}^{4} \mathbf{Pr}\left(\mathsf{coll}_{010:\mathcal{S}_j}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right). \tag{10}$$

$\mathbf{coll_{010:\mathcal{S}_1}(Q_q)}$. Consider configuration $\mathsf{coll}_{010}(Q)$ with the query at position $\mathcal{S}_1 = \{1L\}$ left out. By $\neg\mathsf{aux}_1(Q_q)$, the configuration has at most t_1 choices for $3L, 3R$. For any such choice, by $\neg\mathsf{aux}_3(Q_q)$ we have at most t_2 choices for $1R$. Any such choice fixes u', v', c_1', w' (as $\mathsf{a}_{44} \neq 0$), and thus the values $\mathsf{a}_1\cdot(u,v,c_1) = \mathsf{a}_1\cdot(u',v',c_1')$ and $\mathsf{a}_3\cdot(u,v,c_1,w) = \mathsf{a}_3\cdot(u',v',c_1',w')$. Together with the query $3L$ this fixes (u,v,c_1) by invertibility of A. We find $|\mathcal{W}_{\mathcal{S}_1}| \leq t_1 t_2$, and hence in this setting a collision is found with probability at most $t_1 t_2/(2^n - q)$.

$\mathbf{coll_{010:\mathcal{S}_2}(Q_q)}$. Consider configuration $\mathsf{coll}_{010}(Q)$ with the query at position $\mathcal{S}_2 = \{3L\}$ left out. We have at most q choices for $2L = 2R$. For any such choice, by $\neg\mathsf{aux}_2(Q_q)$ we have at most t_2 choices for $1L$ and at most t_2 choices for $1R$. Any such choice fixes the query at position $3R$, and thus z. The query at position $1L$ fixes (u, v, c_1) and together with query $2L$ this fixes w. Any choice of queries thus uniquely fixes $(\mathsf{a}_3 \cdot (u, v, c_1), \mathsf{a}_4 \cdot (u, v, c_1, w), z - \mathsf{a}_4 \cdot (u, v, c_1, w))$. We find $|\mathcal{W}_{\mathcal{S}_2}| \le t_2^2 q$, and hence in this setting a collision is found with probability at most $t_2^2 q/(2^n - q)$.

$\mathbf{coll}_{\mathbf{010}:\boldsymbol{\mathcal{S}_3}}\boldsymbol{(Q_q)}$**.** As the winning query (k, m, c) should appear at positions $1L$ and $3L$, we require it to satisfy $k = \mathsf{a}_3 \cdot (k, m, c)$. Any query satisfies this equation with probability at most $1/(2^n - q)$ (as $\mathsf{a}_{32}, \mathsf{a}_{33} \neq 0$). As the adversary makes at most q queries, in this setting a collision is found with probability at most $q/(2^n - q)$.

$\mathbf{coll}_{\mathbf{010}:\boldsymbol{\mathcal{S}_4}}\boldsymbol{(Q_q)}$**.** Consider any query, without loss of generality a forward query on input (k, m). Note that, as the query appears at positions $1L$ and $3R$, we have $k = u = \mathsf{a}_3 \cdot (u', v', c_1')$ and $m = v = \mathsf{a}_4 \cdot (u', v', c_1', w')$. By $\neg\mathsf{aux}_3(Q_q)$, the configuration has at most t_2 choices for $1R$. Any such query fixes (u', v', c_1'). Recall that, as $2L = 2R$, we require $\mathsf{a}_1 \cdot (u, v, c_1) = \mathsf{a}_1 \cdot (u', v', c_1')$. Now, the query succeeds only if c_1 satisfies this equation, hence with probability at most $t_2/(2^n - q)$ (as $\mathsf{a}_{13} \neq 0$). Exactly the same statement holds for inverse queries (as $\mathsf{a}_{12} \neq 0$). As the adversary makes at most q queries, in this setting a collision is found with probability at most $t_2 q/(2^n - q)$.

The proof is now completed by adding all bounds in accordance with (10). □

Lemma 4. $\mathbf{Pr}\left(\mathsf{coll}_{\alpha_1\alpha_2\alpha_3}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) = 0$ *when* $\alpha_1 + \alpha_2 + \alpha_3 \ge 2$.

Proof. First suppose $\alpha_2 = \alpha_3 = 1$. By design of F^3_{A} we require $\mathsf{A}(u, v, c_1, w)^\top = \mathsf{A}(u', v', c_1', w')^\top$. By invertibility of A this gives $(u, v, w) = (u', v', w')$ and this implies that the collision is trivial. Now, suppose $\alpha_1 = \alpha_2 = 1$ (the case of $\alpha_1 = \alpha_3 = 1$ is the same by symmetry). In this case, by design of F^3_{A} we have $(u, v, c_1) = (u', v', c_1')$ (by $\alpha_1 = 1$) and $\mathsf{a}_2 \cdot (u, v, c_1, w) = \mathsf{a}_2 \cdot (u', v', c_1', w')$ (by $\alpha_2 = 1$). As $\mathsf{a}_{24} \neq 0$ this implies $w = w'$ and thus that the collision is trivial. □

Lemma 5. $\mathbf{Pr}\left(\mathsf{aux}(Q_q)\right) \le \frac{q^2}{t_1(2^n - q)} + 3 \cdot 2^n \left(\frac{eq}{t_2(2^n - q)}\right)^{t_2}$.

Proof. Note that $\mathsf{aux}_1(Q_q)$ essentially equals $\mathsf{help}_1(Q_q)$ of [27, Section 3.1], and the proof and bound directly carry over. The analysis for $\mathsf{aux}_2(Q_q)$, $\mathsf{aux}_3(Q_q)$, and $\mathsf{aux}_4(Q_q)$ essentially equals the one for $\mathsf{help}_4(Q_q)$ of [27, Section 3.1]. We include the proof for completeness.

It suffices to consider the events $\mathbf{Pr}\left(\mathsf{aux}_k(Q_q)\right)$ $(k = 1, \ldots, 4)$ separately.

$\mathbf{aux_1}\boldsymbol{(Q_q)}$**.** For $i \neq j$, the two queries (k_i, m_i, c_i) and (k_j, m_j, c_i) satisfy $m_i + c_i = m_j + c_j$ with probability at most $\frac{1}{2^n - q}$. Hence, the expected value $\mathsf{E}(m_i + c_i = m_j + c_j)$ is at most $\frac{1}{2^n - q}$, and consequently

$$\mathsf{E}\left(\left|\left\{(k_i, m_i, c_i), (k_j, m_j, c_j) \in Q_q \ : \ i \neq j \ \wedge \ m_i + c_i = m_j + c_j\right\}\right|\right) \le \sum_{i \neq j} \frac{1}{2^n - q} \le \frac{q^2}{2^n - q}.$$

By Markov's inequality, we obtain

$$\mathbf{Pr}\left(\mathsf{aux}_1(Q_q)\right) \le \frac{q^2}{t_1(2^n - q)}. \tag{11}$$

$\mathbf{aux}_{\boldsymbol{k}}\boldsymbol{(Q_q)}$ **for** $\boldsymbol{k \in \{2, 3, 4\}}$**.** For the proof to go through we use $\mathsf{a}_{12}, \mathsf{a}_{13} \neq 0$ (for $\mathsf{aux}_2(Q_q)$) and $\mathsf{a}_{32}, \mathsf{a}_{33} \neq 0$ (for $\mathsf{aux}_3(Q_q)$). The cases are equivalent by symmetry, and we consider $\mathsf{aux}_2(Q_q)$ only. Let $Z \in \{0, 1\}^n$. Consider the ith query (k_i, m_i, c_i). This query makes equation $\mathsf{a}_1 \cdot (k_i, m_i, c_i) = Z$ satisfied with probability at most $\frac{1}{2^n - q}$. More than t_2 queries result in a solution with probability at most $\binom{q}{t_2}\left(\frac{1}{2^n - q}\right)^{t_2} \le \left(\frac{eq}{t_2(2^n - q)}\right)^{t_2}$, where we use Stirling's approximation $(t! \ge (t/e)^t$ for any $t)$. Considering any possible choice for Z, we obtain for $k = 2, 3, 4$:

$$\mathbf{Pr}\left(\mathsf{aux}_k(Q_q)\right) \le 2^n \left(\frac{eq}{t_2(2^n - q)}\right)^{t_2}. \tag{12}$$

The claim is obtained by adding (11-12). □

From (7) and the results of Lemmas 1-5 we conclude for $\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q)$:

$$\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q) \leq \frac{2t_2^2q + 3t_2q + 11q + 3t_1t_2^2 + 7t_1t_2}{2^n - q} + \frac{q^2}{t_1(2^n-q)} + 3\cdot 2^n\left(\frac{eq}{t_2(2^n-q)}\right)^{t_2}.$$

This completes the proof of Theorem 1.

6 Preimage Resistance of F^3_{A}

In this section we consider the preimage resistance of F^3_{A}. Though we do not obtain optimal preimage resistance—which is impossible to achieve after all, due to the generic bounds of the pigeonhole-birthday attack (Section 2)—we achieve preimage resistance up to $2^{3n/2}$ queries, much better than the preimage bounds on MDC-2 and MDC-4 [27], relatively close to the generic bound. Yet, for the proof to hold we need to put slightly stronger requirements on A.

- $\mathsf{A} - \begin{pmatrix} \mathsf{B}_1 & \begin{matrix} 0\,0 \\ 0\,0 \end{matrix} \\ \mathsf{B}_2 & \begin{matrix} 0\,0 \\ 0\,0 \end{matrix} \end{pmatrix}$ is invertible for any $\mathsf{B}_1, \mathsf{B}_2 \in \left\{\left(\begin{smallmatrix}0&0\\0&0\end{smallmatrix}\right), \left(\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right), \left(\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right)\right\}$. In the remainder, we write $\left[\mathsf{B}_1/\mathsf{B}_2\right]$ to denote the subtracted matrix;
- $\mathsf{a}_{12}, \mathsf{a}_{13}, \mathsf{a}_{24}, \mathsf{a}_{32}, \mathsf{a}_{33}, \mathsf{a}_{44} \neq 0$;
- $\mathsf{a}_{12} \neq \mathsf{a}_{32}$, $\mathsf{a}_{13} \neq \mathsf{a}_{33}$, and $\mathsf{a}_{24} \neq \mathsf{a}_{44}$.

We refer to the logical AND of these requirements as `prereq`. We remark that `prereq` $\Rightarrow$ `colreq`, and that matrices satisfying `prereq` are easily found. Simple matrices complying with these conditions over the field $GF(2^{128})$ are

$$\begin{pmatrix} 0&1&2&0\\ 1&0&0&1\\ 0&2&1&0\\ 0&0&0&2 \end{pmatrix}, \qquad \begin{pmatrix} 0&1&1&0\\ 1&1&0&1\\ 0&2&3&0\\ 1&0&2&2 \end{pmatrix}. \tag{13}$$

These are the matrices corresponding to the compression functions of Figure 2. Here, we use $x^{128}+x^{127}+x^{126}+x^{121}+1$ as our irreducible polynomial and we represent bit strings as polynomials in the obvious way ($1 = 1$, $2 = x$, $3 = 1 + x$). Note that the choice of matrix A influences the efficiency of the construction. The first matrix of (13) has as minimal zeroes as possible, which reduces the amount of computation.

Theorem 2. *Let $E \xleftarrow{\$} Bloc(n)$. Suppose A satisfies `prereq`. Then, for any positive integral value t, provided $t \leq q$,*

$$\mathbf{adv}^{\mathrm{epre}}_{F^3_{\mathsf{A}}}(q) \leq \frac{6t^2 + 18t + 26}{2^n - 2} + 4\cdot 2^n\left(\frac{4eq}{t2^n}\right)^{t/2} + 8q\left(\frac{8eq}{t2^n}\right)^{\frac{t2^n}{4q}}. \tag{14}$$

The proof is given in Section 6.1. As for the bound on the collision resistance (Theorem 1), the idea is to make a smart choice of t to minimize this bound. Let $\varepsilon > 0$ be any parameter. Then, for $t = q^{1/3}$, the bound simplifies to

$$\mathbf{adv}^{\mathrm{epre}}_{F^3_{\mathsf{A}}}(q) \leq \frac{6q^{2/3} + 18q^{1/3} + 26}{2^n - 2} + 4\cdot 2^n\left(\frac{4eq^{2/3}}{2^n}\right)^{q^{1/3}/2} + 8q\left(\frac{8eq^{2/3}}{2^n}\right)^{\frac{2^n}{4q^{2/3}}}.$$

From this, we find that for any $\varepsilon > 0$ we have

$$\mathbf{adv}^{\mathrm{epre}}_{F^3_{\mathsf{A}}}(2^{3n/2}/2^{n\varepsilon}) \to 0 \text{ for } n \to \infty\,.$$

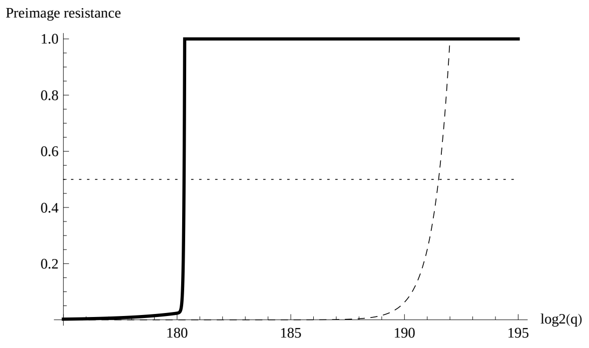


Fig. 14. For $n = 128$, the function $\mathbf{adv}^{\text{epre}}_{F^3_{\mathsf{A}}}(q)$ of (14) for the particular choice of values t (solid line) and the bound of $q^2/2^{3n}$ (dashed line). The steepness of our bound is caused by the last term of (14) which explodes for q approaching $t2^n$ due to its decreasing exponent.

Hence, the F^3_{A} compression function achieves close to $2^{3n/2}$ preimage security for $n \to \infty$. For $n = 128$, the bound on $\mathbf{adv}^{\text{epre}}_{F^3_{\mathsf{A}}}$ is depicted in Figure 14. The preimage advantage hits 1/2 for $\log_2 q \approx 180.3$, relatively close to the threshold 191.5 for $q^2/2^{3n}$. For larger values of n this gap approaches 0.

The result shows that F^3_{A} with A compliant to `prereq` satisfies preimage resistance up to about $2^{3n/2}$ queries. We note that our proof is the best possible for this design, by demonstrating in Proposition 2 a preimage-finding adversary that with high probability succeeds in at most $O(2^{3n/2})$ queries.

Proposition 2. *Let $E \overset{\$}{\leftarrow} Bloc(n)$. Then, one can expect a preimage for F^3_{A} after $2 \cdot 2^{3n/2} + 2^n$ queries.*

Proof. Let $(y, z) \in \{0, 1\}^{2n}$ be a range value. Let $\alpha \in \mathbb{N}$. The adversary proceeds as follows.

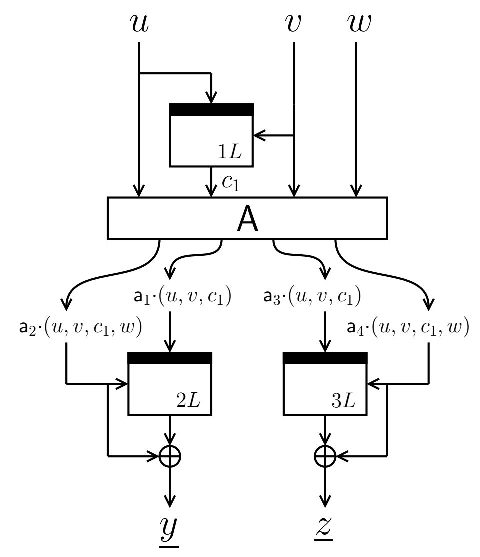
(i) $\mathcal{A}$ makes $\alpha 2^n$ queries to the block cipher corresponding to the bottom-left position of Figure 4. One expects to find α tuples (k_2, m_2, c_2) that satisfy $m_2 + c_2 = y$;
(ii) It repeats the first step for the bottom-right position. One expects to find α tuples (k_3, m_3, c_3) satisfying $m_3 + c_3 = z$;
(iii) By invertibility of A, any choice of (k_2, m_2, c_2) and (k_3, m_3, c_3) uniquely defines a tuple (u, v, c_1, w) for the F^3_{A} evaluation. Likely, the emerged tuples (u, v, c_1) are all different, and we find about α^2 such tuples;
(iv) Varying over all α^2 tuples (u, v, c_1), query (u, v) to the block cipher. If it responds c_1, we have obtained a preimage for F^3_{A}.

In the last round one expects to find a preimage if $\alpha^2/2^n = 1$, or equivalently if $\alpha = 2^{n/2}$. The first and second round both require approximately $2^{3n/2}$ queries, and the fourth round takes 2^n queries. In total, the attack is done in approximately $2 \cdot 2^{3n/2} + 2^n$ queries. □

6.1 Proof of Theorem 2

The proof of preimage resistance of F^3_{A} follows the basic spirit of [27]. Let (y, z) be a range value. We consider any adversary that has query access to its oracle E and makes q queries stored in a query history Q_q. Its goal is to find a preimage for F^3_{A}, in which it by definition only succeeds if it obtains a query history Q_q that satisfies configuration $\mathsf{pre}(Q_q)$ of Figure 15. Formally, $\mathsf{pre}(Q_q)$ is set if for some (u, v, w) there exists query tuples $(k_1, m_1, c_1), (k_2, m_2, c_2), (k_3, m_3, c_3) \in Q_q$ such that:

1. $(k_1, m_1) = (u, v)$ and $(k_2, m_2, k_3, m_3)^\top = \mathsf{A}(u, v, c_1, w)^\top$;



u
v
w
$1L$
c_1
A
$\mathsf{a}_1\cdot(u,v,c_1)$
$\mathsf{a}_3\cdot(u,v,c_1)$
$\mathsf{a}_2\cdot(u,v,c_1,w)$
$\mathsf{a}_4\cdot(u,v,c_1,w)$
$2L$
$3L$
$\underline{y}$
$\underline{z}$

Fig. 15. Configuration $\mathsf{pre}(Q)$. The configuration is satisfied if Q contains three (possibly the same) queries that satisfy this setting.

2. $m_2 \oplus c_2 = y$;
3. $m_3 \oplus c_3 = z$.

This means,

$$\mathbf{adv}^{\mathrm{epre}}_{F^3_{\mathsf{A}}}(q) = \mathbf{Pr}\left(\mathsf{pre}(Q_q)\right). \tag{15}$$

Above set of tuples is also called a "solution" to the configuration. We inherit the notation of Section 5.1. The underlining of y and z means that these are fixed (by the adversary) from the start. We name the block ciphers $1L, 2L, 3L$ similarly.
For the analysis of the preimage resistance, we use the idea of free super queries [3,17,19,27]. The issuance of free super queries is a well-established proof trick for achieving preimage resistance beyond the birthday bound. If under some key the adversary has made 2^{n-1} queries to E, it receives the remaining 2^{n-1} queries for this key for free. As in [3,19], we call this query a super query. Free queries can be formalized as queries the adversary is forced to make, but for which it will not be charged. For convenience, we use Q_q to denote the query history after q normal queries: it thus contains all normal queries plus all super queries made so far. A super query is a set of 2^{n-1} single queries, and any query in the query history is either a normal query or a part of a super query, but not both. Notice that at most $q/2^{n-1}$ super queries will occur: the adversary makes q queries, and needs 2^{n-1} queries as preparatory work to enforce one super query.

For the analysis of $\mathbf{Pr}\left(\mathsf{pre}(Q_q)\right)$ we introduce an auxiliary event $\mathsf{aux}(Q_q)$. Let $t > 0$ be any integral value. We define $\mathsf{aux}(Q_q) = \mathsf{aux}_2(Q_q) \vee \cdots \vee \mathsf{aux}_5(Q_q)$, where

$$\begin{aligned}
\mathsf{aux}_2(Q_q): &\quad \max\nolimits_{Z\in\{0,1\}^n} \left|\left\{(k_i,m_i,c_i)\in Q_q \ :\ \mathsf{a}_1\cdot(k_i,m_i,c_i) = Z\right\}\right| > t\,;\\
\mathsf{aux}_3(Q_q): &\quad \max\nolimits_{Z\in\{0,1\}^n} \left|\left\{(k_i,m_i,c_i)\in Q_q \ :\ \mathsf{a}_3\cdot(k_i,m_i,c_i) = Z\right\}\right| > t\,;\\
\mathsf{aux}_4(Q_q): &\quad \max\nolimits_{Z\in\{0,1\}^n} \left|\left\{(k_i,m_i,c_i)\in Q_q \ :\ m_i + c_i = Z\right\}\right| > t\,;\\
\mathsf{aux}_5(Q_q): &\quad \max\nolimits_{Z\in\{0,1\}^n} \left|\left\{(k_i,m_i,c_i)\in Q_q \ :\ \mathsf{a}_1\cdot(k_i,m_i,c_i) - \mathsf{a}_3\cdot(k_i,m_i,c_i) = Z\right\}\right| > t\,.
\end{aligned}$$

Note that $\mathsf{aux}_2(Q_q), \mathsf{aux}_3(Q_q), \mathsf{aux}_4(Q_q)$ equal the ones of Section 5.1, but we reintroduce them for convenience. By basic probability theory, we obtain for (15):

$$\mathbf{Pr}\left(\mathsf{pre}(Q_q)\right) \le \mathbf{Pr}\left(\mathsf{pre}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) + \mathbf{Pr}\left(\mathsf{aux}(Q_q)\right). \tag{16}$$

Probability $\mathbf{Pr}\left(\mathsf{pre}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right)$ is bounded in Lemma 6. A bound on $\mathbf{Pr}\left(\mathsf{aux}(Q_q)\right)$ is derived in Lemma 7.

Lemma 6. $\mathbf{Pr}\left(\mathsf{pre}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) \le \frac{6t^2+18t+26}{2^n-2}$.

Proof. We consider the probability of the adversary finding a solution to configuration $\mathsf{pre}(Q_q)$ of Figure 15 such that Q_q satisfies $\neg\mathsf{aux}(Q_q)$. The proof shows similarities with the proof of Lemma 1. Consider the ith query, for $i \in \{1, \dots, q\}$. We say this (normal or super) query is a winning query if it makes $\mathsf{pre}(Q_i) \wedge \neg\mathsf{aux}(Q_i)$ satisfied for some set of other queries in the query history Q_{i-1}. We can assume the ith query does not make $\mathsf{aux}(Q_i)$ satisfied: if it would, by definition it cannot be a winning query. It may be the case that a winning query appears at two or three positions in the configuration. In more detail, one can identify the following 7 sets of positions in which the winning query can appear:

$$\begin{aligned}
\mathcal{S}_1 &= \{1L\}\,, & \mathcal{S}_4 &= \{1L, 2L\}\,, & \mathcal{S}_7 &= \{1L, 2L, 3L\}\,,\\
\mathcal{S}_2 &= \{2L\}\,, & \mathcal{S}_5 &= \{1L, 3L\}\,, & &\\
\mathcal{S}_3 &= \{3L\}\,, & \mathcal{S}_6 &= \{2L, 3L\}\,. & &
\end{aligned}$$

For $j = 1, \dots, 7$ we denote by $\mathsf{pre}_{\mathcal{S}_j}(Q_q)$ configuration $\mathsf{pre}(Q_q)$ with the restriction that the winning query *must* appear at the positions in $\mathcal{S}_j$. Recall that a winning query may consist of different queries if it is a super query. By basic probability theory,

$$\mathbf{Pr}\left(\mathsf{pre}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) \le \sum_{j=1}^{7} \mathbf{Pr}\left(\mathsf{pre}_{\mathcal{S}_j}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right). \tag{17}$$

$\mathsf{pre}_{\mathcal{S}_1}(Q_q)$**.** In this case, the winning query may be a normal query or a super query. As is done in the proof of Lemma 1, we use wish lists for the analysis. Consider configuration $\mathsf{pre}(Q_q)$ with the query at position $\mathcal{S}_1 = \{1L\}$ left out (see Figure 16). For any pair of queries that satisfy this configuration at positions $\{2L, 3L\}$, the tuple (u, v, c_1) is added to $\mathcal{W}_{\mathcal{S}_1}$. Note that this tuple is uniquely determined by the queries at $2L$ and $3L$ by invertibility of A.

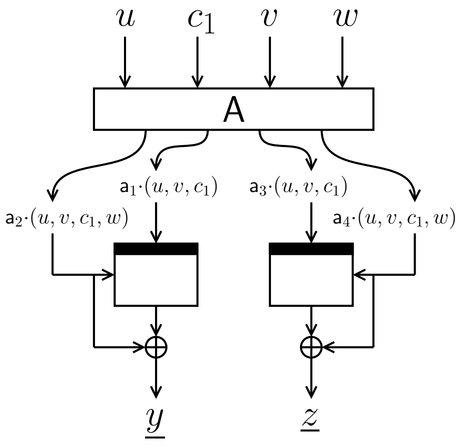


Fig. 16. Configuration $\mathsf{pre}_{\mathcal{S}_1}(Q)$.

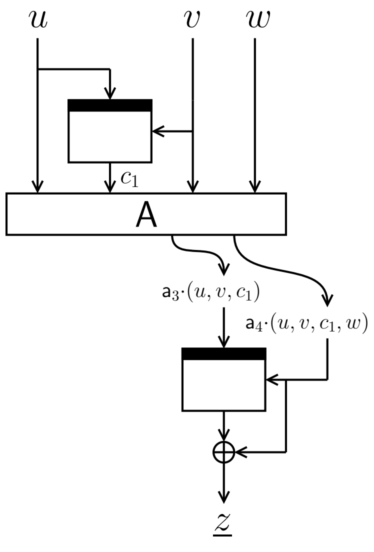


Fig. 17. Configuration $\mathsf{pre}_{\mathcal{S}_2}(Q)$.

As before, as the winning query only occurs at $\mathcal{S}_1$, we can assume a query never adds itself to a wish list. In order to find a preimage for F^3_{A} in this sub-configuration the adversary needs to get a wish granted at least once. The adversary can make each wish at most once. Note that it can make multiple wishes at the same time (in case of super queries), but this does not invalidate the analysis. Suppose the adversary makes a query $E(k, m)$ where $(k, m, c) \in \mathcal{W}_{\mathcal{S}_1}$ for some c. If the query is a normal query, the answer is drawn uniformly at random from a set of size at least 2^{n-1}. If, on the other hand, this wish is a part of a super query, the answer is generated from a set of size 2^{n-1}. In both cases, the wish is granted with probability at most $1/2^{n-1}$ (and the same for inverse queries). Thus, by construction, in this setting the adversary finds a preimage with probability at most $\frac{|\mathcal{W}_{\mathcal{S}_1}|}{2^{n-1}}$.

Now, it suffices to upper bound the size of the wish list $\mathcal{W}_{\mathcal{S}_1}$ after q queries, and to this end we bound the number of solutions to the configuration of Figure 16. By $\neg\mathsf{aux}_4(Q_q)$, the configuration has at most t choices for $2L$ and at most t choices for $3L$. For any such choice, the queries at positions $2L$ and $3L$ uniquely fix (u, v, c_1). We find $|\mathcal{W}_{\mathcal{S}_1}| \leq t^2$, and hence in this setting a preimage is found with probability at most $t^2/2^{n-1}$.

$\mathsf{pre}_{\mathcal{S}_j}(Q_q)$ for $j = 2, 3$. Both cases are the same by symmetry, and we consider $\mathcal{S}_2$ only.

The analysis is similar to the one for $\mathcal{S}_1$, and we only present the computation of the bound on the wish list $\mathcal{W}_{\mathcal{S}_2}$ after q queries. Consider configuration $\mathsf{pre}(Q_q)$ with the query at position $\mathcal{S}_2 = \{2L\}$ left out (see Figure 17). By $\neg\mathsf{aux}_4(Q_q)$, the configuration has at most t choices for $3L$. For any such choice, by $\neg\mathsf{aux}_3(Q_q)$ we have at most t choices for $1L$. The query at position $1L$ fixes (u, v, c_1) and together with query $3L$ this fixes w (as $\mathsf{a}_{44} \neq 0$). Any choice of queries thus uniquely fixes $(\mathsf{a}_1 \cdot (u, v, c_1), \mathsf{a}_2 \cdot (u, v, c_1, w), y - \mathsf{a}_2 \cdot (u, v, c_1, w))$. We find $|\mathcal{W}_{\mathcal{S}_2}| \leq t^2$, and hence in this setting a preimage is found with probability at most $t^2/2^{n-1}$.

$\mathsf{pre}_{\mathcal{S}_j}(Q_q)$ for $j = 4, 5$. Both cases are the same by symmetry, and we consider $\mathcal{S}_4$ only.

We make the distinction between whether or not the two queries at positions $\mathcal{S}_4 = \{1L, 2L\}$ are the same (normal or super query), or are different (super query).

- **$1L = 2L$.** In this case, the wish list contains tuples of the form (k, m, c) that by construction are required to satisfy $k = \mathsf{a}_1 \cdot (k, m, c)$ and $m = \mathsf{a}_2 \cdot (k, m, c, w)$ for some w. As was the case with $\mathcal{S}_1$, each wish is granted with probability at most $1/2^{n-1}$. By $\neg\mathsf{aux}_4(Q_q)$, the configuration has at most t choices for $3L$. For any such choice, this query fixes values $\mathsf{a}_3 \cdot (k, m, c)$ and $\mathsf{a}_4 \cdot (k, m, c, w)$. Together with the equations on a_1 and a_2 this uniquely fixes (k, m, c) by invertibility of $\mathsf{A} - \left[\left(\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right)/\left(\begin{smallmatrix}0&0\\0&0\end{smallmatrix}\right)\right]$. We find $|\mathcal{W}_{\mathcal{S}_4}| \leq t$, and hence in this setting a preimage is found with probability at most $t/2^{n-1}$.
- **$1L \neq 2L$.** In this case, the wish list contains tuples of the form (k, m_1, c_1, m_2, c_2), where (k, m_1, c_1) is the wished query at $1L$ and (k, m_2, c_2) is the wished query at $2L$. By construction, these tuples are required to satisfy $k = \mathsf{a}_1 \cdot (k, m_1, c_1)$ and $m_2 = \mathsf{a}_2 \cdot (k, m_1, c_1, w)$ for some w. Additionally the wish is required to satisfy $m_2 + c_2 = y$. As in a super query the answers are generated from a set of size 2^{n-1}, a wish is granted with probability at most $\frac{1}{2^{n-1}(2^{n-1}-1)}$. Thus, by construction, in this setting the adversary finds a preimage with probability at most
$$\frac{|\mathcal{W}_{\mathcal{S}_4}|}{2^{n-1}(2^{n-1}-1)}\,.$$
By $\neg\mathsf{aux}_4(Q_q)$, the configuration has at most t choices for $3L$. For any such choice, this query fixes values $\mathsf{a}_3 \cdot (k, m_1, c_1)$ and $\mathsf{a}_4 \cdot (k, m_1, c_1, w)$. We have 2^n choices for c_2. This uniquely fixes m_2. Now, this uniquely fixes (k, m_1, c_1) by invertibility of $\mathsf{A} - \left[\left(\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right)/\left(\begin{smallmatrix}0&0\\0&0\end{smallmatrix}\right)\right]$. We find $|\mathcal{W}_{\mathcal{S}_4}| \leq t2^n$, and hence in this setting a preimage is found with probability at most $\frac{t2^n}{2^{n-1}(2^{n-1}-1)}$.

$\mathsf{pre}_{\mathcal{S}_6}(Q_q)$. We make the distinction between whether or not the two queries at positions $\mathcal{S}_6 = \{2L, 3L\}$ are the same (normal or super query), or are different (super query).

- **$2L = 3L$.** In this case, the wish list contains tuples of the form (k, m, c) that by construction are required to satisfy $k = \mathsf{a}_1 \cdot (u, v, c_1) = \mathsf{a}_3 \cdot (u, v, c_1)$, and $m = \mathsf{a}_2 \cdot (u, v, c_1, w) = \mathsf{a}_4 \cdot (u, v, c_1, w)$

for some u, v, c_1, w. Additionally the wish is required to satisfy $m + c = y = z$. As was the case with $\mathcal{S}_1$, each wish is granted with probability at most $1/2^{n-1}$.
By $\neg\mathsf{aux}_5(Q_q)$, noting that $\mathsf{a}_1\cdot(u, v, c_1) = \mathsf{a}_3\cdot(u, v, c_1)$, the configuration has at most t choices for $1L$. For any such choice, this query fixes values (u, v, c_1), and thus k. Equation $\mathsf{a}_2\cdot(u, v, c_1, w) = \mathsf{a}_4\cdot(u, v, c_1, w)$ fixes w (as $\mathsf{a}_{24} \neq \mathsf{a}_{44}$), and thus m. Using $m+c = y$ this uniquely fixes (k, m, c). We find $|\mathcal{W}_{\mathcal{S}_6}| \leq t$, and hence in this setting a preimage is found with probability at most $t/2^{n-1}$.

- $\boldsymbol{2L \neq 3L}$**.** In this case, the wish list contains tuples of the form (k, m_2, c_2, m_3, c_3), where (k, m_2, c_2) is the wished query at $2L$ and (k, m_3, c_3) is the wished query at $3L$. By construction, these tuples are required to satisfy $k = \mathsf{a}_1\cdot(u, v, c_1) = \mathsf{a}_3\cdot(u, v, c_1)$, $m_2 = \mathsf{a}_2\cdot(u, v, c_1, w)$, and $m_3 = \mathsf{a}_4\cdot(u, v, c_1, w)$ for some u, v, c_1, w. Additionally the wish is required to satisfy $m_2+c_2 = y$ and $m_3 + c_3 = z$. As before, in this setting the adversary finds a preimage with probability at most
$$\frac{|\mathcal{W}_{\mathcal{S}_6}|}{2^{n-1}(2^{n-1}-1)}\,.$$
By $\neg\mathsf{aux}_5(Q_q)$, noting that $\mathsf{a}_1\cdot(u, v, c_1) = \mathsf{a}_3\cdot(u, v, c_1)$, the configuration has at most t choices for $1L$. For any such choice, this query fixes values (u, v, c_1) and thus k. We have 2^n choices for c_3. This uniquely fixes m_3. This uniquely fixes w, and subsequently m_2 and c_2. We find $|\mathcal{W}_{\mathcal{S}_6}| \leq t2^n$, and hence in this setting a preimage is found with probability at most $\frac{t2^n}{2^{n-1}(2^{n-1}-1)}$.

$\mathsf{pre}_{\mathcal{S}_7}\boldsymbol{(Q_q)}$**.** We make the following distinction: $1L = 2L = 3L$, $1L = 2L \neq 3L$, $1L = 3L \neq 2L$, $2L = 3L \neq 1L$, and $\{1L, 2L, 3L\}$ all different.

- $\boldsymbol{1L = 2L = 3L}$**.** In this case, the wish list contains tuples of the form (k, m, c) that by construction are required to satisfy $k = \mathsf{a}_1\cdot(k, m, c) = \mathsf{a}_3\cdot(k, m, c)$ and $m = \mathsf{a}_2\cdot(k, m, c, w) = \mathsf{a}_4\cdot(k, m, c, w)$ for some w. These equations uniquely determine (k, m, c, w) by invertibility of $\mathsf{A} - \left[\begin{pmatrix}1&0\\0&1\end{pmatrix}/\begin{pmatrix}1&0\\0&1\end{pmatrix}\right]$, and we find $|\mathcal{W}_{\mathcal{S}_1}| = 1$. Hence, in this setting a preimage is found with probability at most $1/2^{n-1}$.
- $\boldsymbol{1L = 2L \neq 3L}$ **or** $\boldsymbol{1L = 3L \neq 2L}$**.** Both cases are the same by symmetry, and we consider $1L = 2L \neq 3L$ only.
In this case, the wish list contains tuples of the form (k, m, c, m_3, c_3), where (k, m, c) is the wished query at $1L = 2L$ and (k, m_3, c_3) is the wished query at $3L$. By construction, these tuples are required to satisfy $k = \mathsf{a}_1\cdot(k, m, c) = \mathsf{a}_3\cdot(k, m, c)$, $m = \mathsf{a}_2\cdot(k, m, c, w)$, and $m_3 = \mathsf{a}_4\cdot(k, m, c, w)$ for some w. Additionally the wish is required to satisfy $m+c = y$ and $m_3+c_3 = z$. As before, in this setting the adversary finds a preimage with probability at most
$$\frac{|\mathcal{W}_{\mathcal{S}_7}|}{2^{n-1}(2^{n-1}-1)}\,.$$
We have 2^n choices for c_3. This uniquely fixes m_3. This uniquely fixes (k, m, c) by invertibility of $\mathsf{A} - \left[\begin{pmatrix}1&0\\0&1\end{pmatrix}/\begin{pmatrix}1&0\\0&0\end{pmatrix}\right]$. We find $|\mathcal{W}_{\mathcal{S}_7}| \leq 2^n$, and hence in this setting a preimage is found with probability at most $\frac{2^n}{2^{n-1}(2^{n-1}-1)}$.
- $\boldsymbol{2L = 3L \neq 1L}$**.** In this case, the wish list contains tuples of the form (k, m_1, c_1, m, c), where (k, m_1, c_1) is the wished query at $1L$ and (k, m, c) is the wished query at $2L = 3L$. By construction, these tuples are required to satisfy $k = \mathsf{a}_1\cdot(k, m_1, c_1) = \mathsf{a}_3\cdot(k, m_1, c_1)$ and $m = \mathsf{a}_2\cdot(k, m_1, c_1, w) = \mathsf{a}_4\cdot(k, m_1, c_1, w)$ for some w. Additionally the wish is required to satisfy $m + c = y = z$. As before, in this setting the adversary finds a preimage with probability at most
$$\frac{|\mathcal{W}_{\mathcal{S}_7}|}{2^{n-1}(2^{n-1}-1)}\,.$$

We have 2^n choices for c. This uniquely fixes m. This uniquely fixes (k, m_1, c_1) by invertibility of $\mathsf{A} - \left[\left(\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right)/\left(\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right)\right]$. We find $|\mathcal{W}_{\mathcal{S}_7}| \leq 2^n$, and hence in this setting a preimage is found with probability at most $\frac{2^n}{2^{n-1}(2^{n-1}-1)}$.

- **$\{1L, 2L, 3L\}$ all different.** In this case, the wish list contains tuples of the form $(k, m_1, c_1, m_2, c_2, m_3, c_3)$, where (k, m_1, c_1) is the wished query at $1L$, (k, m_2, c_2) is the wished query at $2L$, and (k, m_3, c_3) is the wished query at $3L$. By construction, these tuples are required to satisfy $k = \mathsf{a}_1 \cdot (k, m_1, c_1) = \mathsf{a}_3 \cdot (k, m_1, c_1)$, $m_2 = \mathsf{a}_2 \cdot (k, m_1, c_1, w)$, and $m_3 = \mathsf{a}_4 \cdot (k, m_1, c_1, w)$ for some w. Additionally the wish is required to satisfy $m_2 + c_2 = y$ and $m_3 + c_3 = z$. As before, in this setting the adversary finds a preimage with probability at most
$$\frac{|\mathcal{W}_{\mathcal{S}_7}|}{2^{n-1}(2^{n-1}-1)(2^{n-1}-2)}\,.$$
We have 2^n choices for both c_2 and c_3. These uniquely fix m_2 and m_3. Any such choice uniquely fixes (k, m_1, c_1) by invertibility of $\mathsf{A} - \left[\left(\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right)/\left(\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right)\right]$. We find $|\mathcal{W}_{\mathcal{S}_7}| \leq 2^n(2^n - 1)$, and hence in this setting a preimage is found with probability at most $\frac{2^{2n}}{2^{n-1}(2^{n-1}-1)(2^{n-1}-2)}$.

The proof is now completed by adding and simplifying all bounds in accordance with (17):
$$\begin{aligned}\mathbf{Pr}\left(\mathsf{pre}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) &\leq \frac{3t^2+3t+1}{2^{n-1}} + \frac{(3t+3)2^n}{2^{n-1}(2^{n-1}-1)} + \frac{2^{2n}}{2^{n-1}(2^{n-1}-1)(2^{n-1}-2)} \\ &\leq \frac{6t^2+18t+26}{2^n-2}\,,\end{aligned}$$
where we use that $1/(2^{n-1}-2) \leq 3/2^n$ for $n \geq 4$. $\square$

Lemma 7. *Provided $t \leq q$, we have* $\mathbf{Pr}\left(\mathsf{aux}(Q_q)\right) \leq 4 \cdot 2^n \left(\frac{4eq}{t2^n}\right)^{t/2} + 4 \cdot 2q \left(\frac{8eq}{t2^n}\right)^{\frac{t2^n}{4q}}$.

Proof. Note that $\mathsf{aux}_2(Q_q), \ldots, \mathsf{aux}_5(Q_q)$ essentially equal $\mathsf{help}_3(Q_q)$ of [27, Section 4.1], and the proof and bound directly carries over. We include the proof for completeness.

It suffices to consider the events $\mathbf{Pr}\left(\mathsf{aux}_k(Q_q)\right)$ $(k = 2, \ldots, 5)$ separately. For the proof to go through we use $\mathsf{a}_{12}, \mathsf{a}_{13} \neq 0$ (for $\mathsf{aux}_2(Q_q)$), $\mathsf{a}_{32}, \mathsf{a}_{33} \neq 0$ (for $\mathsf{aux}_3(Q_q)$), and $\mathsf{a}_{12} \neq \mathsf{a}_{32}$ and $\mathsf{a}_{13} \neq \mathsf{a}_{33}$ (for $\mathsf{aux}_5(Q_q)$). The cases are equivalent by symmetry, and we consider $\mathsf{aux}_2(Q_q)$ only.

Let $Z \in \{0,1\}^n$. Denote by $Q_q^{(n)}$ the restriction of Q_q to normal queries, and by $Q_q^{(s)}$ the restriction of Q_q to queries that belong to super queries. In order for Q_q to have more than t solutions to $\mathsf{a}_1 \cdot (k_i, m_i, c_i) = Z$, at least one of the following criteria needs to hold:

1. $Q_q^{(n)}$ has more than $t/2$ solutions;
2. $Q_q^{(s)}$ has more than $t/2$ solutions.

We consider these two scenarios separately. In case of normal queries, each query (k_i, m_i, c_i) is answered with a value generated at random from a set of size at least 2^{n-1}, and hence it satisfies $\mathsf{a}_1 \cdot (k_i, m_i, c_i) = Z$ with probability at most $\frac{1}{2^{n-1}} = \frac{2}{2^n}$. More than $t/2$ queries result in a solution with probability at most $\binom{q}{t/2}\left(\frac{2}{2^n}\right)^{t/2} \leq \left(\frac{4eq}{t2^n}\right)^{t/2}$.

The analysis for super queries is more elaborate. In order for $Q_q^{(s)}$ to have more than $t/2$ solutions, as at most $q/2^{n-1}$ super queries occur, at least one of the super queries needs to provide more than $t' := \frac{t}{2q/2^{n-1}} = \frac{t2^n}{4q}$ solutions. Consider any super query, consisting of 2^{n-1} queries. It provides more than t' solutions with probability at most
$$\binom{2^{n-1}}{t'} \prod_{j=0}^{t'-1} \frac{1}{2^{n-1}-j} \leq \binom{2^{n-1}}{t'} \left(\frac{1}{2^{n-1}-t'}\right)^{t'} \leq \left(\frac{e2^{n-1}}{t'(2^{n-1}-t')}\right)^{t'}.$$

Provided $t \leq q$, we have $t' = \frac{t2^n}{4q} \leq 2^{n-2}$, and thus $\frac{1}{2^{n-1}-t'} \leq \frac{1}{2^{n-2}}$. Consequently, this super query adds more than $\frac{t2^n}{4q}$ solutions with probability at most $\left(\frac{8eq}{t2^n}\right)^{\frac{t2^n}{4q}}$. In order to cover any super query, we need to multiply this probability with $q/2^{n-1}$.

Considering any possibly choice for Z, we obtain for $k = 2, \ldots, 5$:

$$\mathbf{Pr}\left(\mathsf{aux}_k(Q_q)\right) \leq 2^n \left(\frac{4eq}{t2^n}\right)^{t/2} + 2^n \cdot \frac{q}{2^{n-1}} \left(\frac{8eq}{t2^n}\right)^{\frac{t2^n}{4q}} .$$

The claim is obtained by multiplying this equation with 4. $\square$

From (15-16) and Lemmas 6-7 we conclude for $\mathbf{adv}^{\text{epre}}_{F^3_{\mathsf{A}}}(q)$:

$$\mathbf{adv}^{\text{epre}}_{F^3_{\mathsf{A}}}(q) \leq \frac{6t^2 + 18t + 26}{2^n - 2} + 4 \cdot 2^n \left(\frac{4eq}{t2^n}\right)^{t/2} + 8q \left(\frac{8eq}{t2^n}\right)^{\frac{t2^n}{4q}} .$$

This completes the proof of Theorem 2.

7 Indifferentiability of F^3_{A}

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 Sections 5-6. Differentiability is discussed in Section 7.1, and indifferentiability in Section 7.2.

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.

7.1 Differentiability

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

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 $2^{n/2} + 2$ queries to its oracles, such that*

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

Proof. Recall that $\mathcal{D}$ has access to either (F^3_{A}, E) (in the real world) or $(\mathcal{R}, \mathcal{S})$ (in the ideal world). 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 $j \in \{1, 3\}$:

$$\mathsf{a}_j \cdot (k_1, m_1, c_1) = \mathsf{a}_j \cdot (k_1', m_1', c_1') . \tag{18}$$

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 (18), then for any $m \in \{0,1\}^n$, the following condition naturally holds in the real world:

$$y = y' \text{ if } j = 1 \text{ and } z = z' \text{ if } j = 3 , \tag{19}$$

where

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

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 (18). 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 (18), $\mathcal{D}$ 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 (18) is satisfied;
(iv) Take $m \xleftarrow{\$} \{0,1\}^n$. If (19) 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 (18) 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 (19) holds (which means that his guess in step (iv) is wrong). Therefore, $\mathbf{adv}^{\text{iff}}_{F^3_{\mathsf{A}}, \mathcal{S}}(\mathcal{D}) \geq 1 - \mathbf{Pr}(\mathsf{C}_1) - \mathbf{Pr}(\mathsf{C}_2)$. 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 (18) 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 $\mathcal{R}(k_1, m_1, \text{get}w(j+1, m, k_1, m_1, c_1))$. 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 up to about $2^{n/2}$ queries. Together with the lower bound of Section 7.1 this implies tightness.

Theorem 3. *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,*

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

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

Recall that we consider $\mathcal{D}$ to have access to either (F^3_{A}, E) (in the real world) or $(\mathcal{R}, \mathcal{S})$ (in the ideal world). 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 $(m,c) \in \mathcal{L}E[k]$.

The F^3_{A} class of functions is based on the key principle that any two block cipher evaluations define the inputs to the third one (using invertibility of A). The simulator we use for the proof of Theorem 3 enormously benefits from some of these characteristics. In more detail, the simulator is given in Figure 18.

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 Figure 4) 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.[5] 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 $c \xleftarrow{\$} \{0,1\}^n \backslash \mathcal{L}E^+[k]$
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 $m \xleftarrow{\$} \{0,1\}^n \backslash \mathcal{L}E^-[k]$
12 return $\mathcal{L}E^-[k](c) \leftarrow m$
```

Fig. 18. The simulator $\mathcal{S}$ for E used in the proof of Theorem 3. See footnote 5 regarding the case where there is more than one solution to the if-clause of line 02.

Proof of Theorem 3

We formally prove Theorem 3. Let $\mathcal{S}$ be the simulator of Figure 18, 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 Definition 3, the goal is to bound:

$$\mathbf{adv}^{\mathrm{iff}}_{F^3_{\mathsf{A}},\mathcal{S}}(\mathcal{D}) = \left|\mathbf{Pr}\left(\mathcal{D}^{F^3_{\mathsf{A}},E}=1\right) - \mathbf{Pr}\left(\mathcal{D}^{\mathcal{R},\mathcal{S}}=1\right)\right| . \qquad (20)$$

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 Figure 18 with the difference that random sampling from $\{0,1\}^n$ is done in lines 01 and 11. Now,

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

[5] Note that if there are two different solutions that make the condition of the if-clause satisfied, the simulator will naturally never be able to maintain consistency with the random cipher. This event will later be considered as a bad event. If this happens, the simulator will simply take any of the solutions and execute the if-clause based on that solution. This design decision is merely for simplicity; the simulator can just as well abort.

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 (20):[6]

$$\mathbf{adv}^{\mathrm{iff}}_{F^3_{\mathsf{A}},\mathcal{S}}(\mathcal{D})\leq\left|\mathbf{Pr}\left(\mathcal{D}^{F^3_{\mathsf{A}},\widetilde{E}}=1\right)-\mathbf{Pr}\left(\mathcal{D}^{\mathcal{R},\widetilde{\mathcal{S}}}=1\right)\right|+\frac{(3q_L+q_R)^2}{2^n}\,. \tag{21}$$

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 Figure 19. 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 $\mathcal{L}\mathcal{R}$ as an initially empty list of all left oracle queries.

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

$$\mathsf{cond}(\mathcal{L}E)=\left(\begin{array}{c}\exists\ j,j'\in\{1,3\},(k,m,c),(k',m',c')\in\mathcal{L}E:\\ (k,m,c)\text{ newer than }(k',m',c')\text{ and}\\ \mathsf{a}_j\cdot(k,m,c)\in\{k,k',\mathsf{a}_{j'}\cdot(k',m',c')\}\end{array}\right). \tag{22}$$

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 Figure 19 is always satisfied by at most one $(j,(k_1,m_1,c_1))$. In the remainder, we prove in Lemma 8 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 Lemma 9 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 (21), this completes the proof.

Lemma 8. *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 (u,v,w).** We make the following distinction:

1. $\mathcal{L}E^+[u](v)=\perp$. 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 "$\mathcal{L}E^+[u](v)=\perp$," $\widetilde{\mathcal{S}}$ 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 $j\in\{1,3\}$ 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. $\mathcal{L}E^+[u](v)\neq\perp$. 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:

[6] Technically, we could have taken $\widetilde{\mathcal{S}}$ as our simulator, therewith obtaining an improved indifferentiability bound for Theorem 3. However, for clarity and ease of presentation, we opted for simulator $\mathcal{S}$.

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

- $\mathcal{L}E^+[k_2](m_2) = \perp$ and $\mathcal{L}E^+[k_3](m_3) = \perp$. 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 $\{0,1\}^{2n}$;
- $\mathcal{L}E^+[k_2](m_2) \neq \perp$ and/or $\mathcal{L}E^+[k_3](m_3) \neq \perp$. 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 $\mathcal{L}E^+[k_3](m_3) \neq \perp$, the same reasoning applies to z, the second half of the query response. On the other hand, in case $\mathcal{L}E^+[k_3](m_3) = \perp$, the previous bullet carries over to the z-part.

***R*-query (k,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 9. $\mathbf{Pr}\left(\mathsf{cond}(\mathcal{LE})\ \textit{for}\ (F^3_{\mathsf{A}}, \widetilde{E})\right) \leq \frac{3(3q_L+q_R)^2}{2^n}$ *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:

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

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 Conclusions

In the area of double block length hashing, where a $3n$-to-$2n$-bit compression function is constructed from n-bit block ciphers, all optimally secure constructions known in the literature employ a block cipher with $2n$-bit key space. We have reconsidered the principle of double length hashing, focusing on double length hashing from a block cipher with *n-bit message and key space*. Unlike in the DBL^{2n} class, we demonstrate that there does not exist any optimally secure design with reasonably simple finalization function that makes two cipher calls. By allowing one extra call, optimal collision resistance can nevertheless be achieved, as we have proven by introducing our family of designs F^3_{A}.

In our quest for optimal collision secure compression function designs, we had to resort to designs with three block cipher calls rather than two, which moreover are not parallelizable. This entails an efficiency loss compared to MDC-2, MJH, and Jetchev et al.'s construction. On the other hand, our family of functions is based on simple arithmetic in the finite field: unlike constructions by Stam [40, 41], Lee and Steinberger [20], and Jetchev et al. [11], our design does not make use of full field multiplications. The example matrices A given in (13) are designed to use a minimal amount of non-zero elements. We note that specific choices of A may be more suited for this construction to be used in an iterated design.

This work provides new insights in double length hashing, but also results in interesting research questions. Most importantly, is it possible to construct other collision secure F^3 constructions (beyond our family of functions F^3_{A}), that achieve optimal $2^{5n/3}$ preimage resistance? An open problem of a more general kind is to design compression functions with better than $2^{n/2}$ indifferentiability security.[7] We note that the differentiability attacks in Appendices B-D rely on the fact that these functions make only two primitive calls, which gives the impression at least three primitive calls are needed for the achievement of such bound.

Also, despite the negative collision resistance result of Proposition 1, the open problems regarding F^2 are plentiful, and we name some. Firstly, can Proposition 1 be generalized to cover, e.g., Jetchev et al.'s construction? Secondly, is it possible to achieve optimal collision security *in the iteration* anyhow? Thirdly, can we derive a two-call compression function with the good "overall" security guarantees? We note that the indifferentiability proofs in this work (Theorems 3 and 4) crucially rely on the presence of a third block cipher call, and the latter question may not be trivial. Further open research directions include, in line with ideas of [28], the consideration of F^3 restricted to the xor-only design (where $f_1, \ldots, f_4$ only xor their parameters).

Acknowledgments. This work was supported in part by the Research Council KU Leuven: GOA TENSE (GOA/11/007). Bart Mennink is a Postdoctoral Fellows of the Research Foundation – Flanders (FWO). The author would like to thank Elena Andreeva, Atul Luykx, and the anonymous reviewers of ASIACRYPT 2012, IMA Cryptography and Coding 2013, and Designs, Codes and Cryptography for their valuable help and feedback. The author is particularly indebted to the anonymous reviewer that pointed out the flaw in the original Proposition 1 (see the remark at the end of Section 3).

References

[1] Abed, F., Forler, C., List, E., Lucks, S., Wenzel, J.: Counter-bDM: A provably secure family of multi-block-length compression functions. In: Progress in Cryptology - AFRICACRYPT 2014. Lecture Notes in Computer Science, vol. 8469, pp. 440–458. Springer, Heidelberg (2014)

[2] 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)

[3] 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)

[4] 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)

[5] 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)

[7] Without going into detail, we refer to a slightly related work of Maurer and Tessaro [24] on indifferentiable domain extenders from random functions.

[6] 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)

[7] 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)

[8] 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)

[9] 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)

[10] Hong, D., Kwon, D.: Cryptanalysis of some double-block-length hash modes of block ciphers with n-bit block and n-bit key. Cryptology ePrint Archive, Report 2013/174 (2013)

[11] 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)

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

[13] 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)

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

[15] 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)

[16] Lee, J., Stam, M.: MJH: A faster alternative to MDC-2. Des. Codes Cryptography 76(2), 179–205 (2015)

[17] Lee, J., Stam, M., Steinberger, J.: The collision security of Tandem-DM in the ideal cipher model. Cryptology ePrint Archive, Report 2010/409 (2010), full version of [18]

[18] 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)

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

[20] 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)

[21] Lucks, S.: A failure-friendly design principle for hash functions. In: Advances in Cryptology - ASIACRYPT 2005. Lecture Notes in Computer Science, vol. 3788, pp. 474–494. Springer, Heidelberg (2005)

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

[23] 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)

[24] 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)

[25] 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)

[26] Mennink, B.: Indifferentiability of double length compression functions. In: IMA International Conference on Cryptography and Coding - IMACC 2013. Lecture Notes in Computer Science, vol. 8308, pp. 232–251. Springer, Heidelberg (2013)

[27] Mennink, B.: On the collision and preimage security of MDC-4 in the ideal cipher model. Designs, Codes and Cryptography 73(1), 121–150 (2014)

[28] Mennink, B., Preneel, B.: Hash functions based on three permutations: A generic security analysis. In: Advances in Cryptology - CRYPTO 2012. Lecture Notes in Computer Science, vol. 7417, pp. 330–347. Springer, Heidelberg (2012)

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

[30] Miyaji, A., Rashed, M.: A new (n, n) blockcipher hash function using Feistel network: Apposite for RFID security. In: Computational Intelligence in Data Mining - Volume 3. Smart Innovation, Systems and Technologies, vol. 33, pp. 519–528. Springer India (2015)

[31] Nandi, M.: Towards optimal double-length hash functions. In: Progress in Cryptology - INDOCRYPT 2005. Lecture Notes in Computer Science, vol. 3797, pp. 77–89. Springer, Heidelberg (2009)

[32] Nandi, M., Lee, W., Sakurai, K., Lee, S.: Security analysis of a 2/3-rate double length compression function in the black-box model. In: Fast Software Encryption 2005. Lecture Notes in Computer Science, vol. 3557, pp. 243–254. Springer, Heidelberg (2005)

[33] Özen, O.: Design and Analysis of Multi-Block-Length Hash Functions. Ph.D. thesis, École Polytechnique Fédérale de Lausanne, Lausanne (2012)
[34] 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)
[35] Peyrin, T., Gilbert, H., Muller, F., Robshaw, M.: Combining compression functions and block cipher-based hash functions. In: Advances in Cryptology - ASIACRYPT 2006. Lecture Notes in Computer Science, vol. 4284, pp. 315–331. Springer, Heidelberg (2006)
[36] 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)
[37] 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)
[38] Rogaway, P., Shrimpton, T.: Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In: Fast Software Encryption 2004. Lecture Notes in Computer Science, vol. 3017, pp. 371–388. Springer, Heidelberg (2004)
[39] Rogaway, P., Steinberger, J.: Security/efficiency tradeoffs for permutation-based hashing. In: Advances in Cryptology - EUROCRYPT 2008. Lecture Notes in Computer Science, vol. 4965, pp. 220–236. Springer, Heidelberg (2008)
[40] 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)
[41] Stam, M.: Blockcipher-based hashing revisited. In: Fast Software Encryption 2009. Lecture Notes in Computer Science, vol. 5665, pp. 67–83. Springer, Heidelberg (2009)
[42] 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)
[43] Steinberger, J.: Stam's collision resistance conjecture. In: Advances in Cryptology - EUROCRYPT 2010. Lecture Notes in Computer Science, vol. 6110, pp. 597–615. Springer, Heidelberg (2010)
[44] Steinberger, J., Sun, X., Yang, Z.: Stam's conjecture and threshold phenomena in collision resistance. In: Advances in Cryptology - CRYPTO 2012. Lecture Notes in Computer Science, vol. 7417, pp. 384–405. Springer, Heidelberg (2012)

A Attack on Recently Proposed Three-Call Compression Function

We consider the compression function $F^3_{\mathrm{MR}} : \{0,1\}^{3n} \to \{0,1\}^{2n}$, that was recently introduced by Miyaji and Rashed [30]. It makes three calls to its block cipher E as follows:

$$\begin{aligned}
&F^3_{\mathrm{MR}}(u,v,w) = (y,z), \text{ where:}\\
&\quad c_1 \leftarrow E(u,w)\,,\\
&\quad k_2 \leftarrow v \oplus c_1\,,\ c_2 \leftarrow E(k_2,w)\,,\\
&\quad k_3 \leftarrow u \oplus c_2\,,\ c_3 \leftarrow E(k_3,w)\,,\\
&\quad (y,z) \leftarrow (u \oplus c_2, v \oplus c_1 \oplus c_3)\,.
\end{aligned}$$

The authors claim that a hash function based on F^3_{MR} achieves optimal 2^n collision and 2^{2n} preimage security, contradicting the generalizations of the pigeonhole-birthday attacks as outlined in (see also Section 2). In fact, we show that improved versions of these generic attacks render collisions for F^3_{MR} in about $2^{2n/3}$ queries and preimages in about 2^n queries.

Proposition 4. *Let $E \overset{\$}{\leftarrow} Bloc(n)$. Then, one can expect a collision for F^3_{MR} after $4 \cdot 2^{2n/3}$ queries, and a preimage after $2 \cdot 2^n$ queries.*

Proof. We apply the pigeonhole-birthday attacks to F^3_{MR}. The attack relies on making block cipher calls that "cover" the evaluation of at least approximately 2^n images (for collision resistance) and 2^{2n} images (for preimage resistance). The adversary makes $2q$ queries as follows.

(i) $\mathcal{A}$ fixes a w;
(ii) $\mathcal{A}$ fixes distinct $k^{(1)}, \ldots, k^{(q)}$ and queries $c^{(i)} \leftarrow E(k^{(i)}, w)$;

(iii) Any choice of $1 \leq i, j \leq q$ uniquely defines a tuple (u, v) such that the first two block cipher calls of the evaluation $F^3_{\mathrm{MR}}(u, v, w)$ are $E(k^{(i)}, w)$ and $E(k^{(j)}, w)$. In more detail, these are

$$u = k^{(i)}\,, \qquad\qquad v = k^{(j)} \oplus c^{(i)}\,,$$

and the key input to the third block cipher will be $k^{(i)} \oplus c^{(j)}$;

(iv) $\mathcal{A}$ identifies a list of q values $l^{(1)}, \ldots, l^{(q)}$ that maximizes the number of solutions to

$$\exists\, i, j, \kappa \in \{1, \ldots, q\} :\; k^{(i)} \oplus c^{(j)} = l^{(\kappa)}\,,$$

and queries $c_3 \leftarrow E(l^{(\kappa)}, w)$.

The q block cipher evaluations in the last round assure that the adversary learns the evaluation of at least $q \cdot q^2/2^n = q^3/2^n$ compression function calls (by the pigeonhole principle).

One expects a collision within these compression function calls if $\binom{q^3/2^n}{2} \approx 2^{2n}$, hence for $q = 2 \cdot 2^{2n/3}$. The total number of queries made by the adversary is $2q = 4 \cdot 2^{2n/3}$. Similarly, for preimages, let $(y, z) \in \{0, 1\}^{2n}$ be a range value. The adversary knows a preimage for (y, z) with certainty if $q^3/2^n \geq 2^{2n}$, hence if $q \geq 2^n$. The total number of queries made by the adversary is $2q = 2 \cdot 2^n$. □

Via a meet-in-the-middle approach, the preimage attack on F^3_{MR} can be used to find a preimage for Merkle-Damgård based on F^3_{MR} in about $2^{3n/2}$ queries. In more detail, let $(y, z) \in \{0, 1\}^{2n}$ be a given range value, and let $(iv_1, iv_2) \in \{0, 1\}^{2n}$ be the initial value. Firstly the attacker finds $K \geq 1$ free-start preimages for (y, z), which requires $K \cdot 2 \cdot 2^n$ queries. Then, starting from (iv_1, iv_2), $\mathcal{A}$ varies the message block to hit any of the K preimages, which requires $2^{2n}/K$ queries. The total complexity is $2K2^n + 2^{2n}/K$. Putting $K = 2^{n/2}$, the total complexity is about $3 \cdot 2^{3n/2}$.

B Indifferentiability Analysis of Functions in $\mathbf{DBL}^{2n}$

The functions analyzed in this section do not exactly fit the model of Section 2, as the underlying block ciphers have a $2n$-bit key. 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. Now, the model, as well as the indifferentiability definition, Definition 3, carries over the natural way. We stress that some of the designs analyzed in this section 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). However, 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.

B.1 Indifferentiability Analysis of Stam's, Tandem-DM, Abreast-DM, Hirose's, and Hirose-Class

In this section, we consider Tandem-DM and Abreast-DM [13] (cf. Figure 1), Hirose's compression function [8] (cf. Figure 1) and its generalized Hirose-class [7],[8] as well as Stam's supercharged single call Type-I compression function design [40,41], or more specifically the block cipher based variant considered in [20]:

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

[8] 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.

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 [34], and our attack on their class (in Appendix B.2) 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 5. *Let $E \overset{\$}{\leftarrow} Bloc(2n, n)$, 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:

$$\mathrm{TDM}(u, v, w) = (u, z) , \tag{23}$$

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 (23) is satisfied). Clearly, in the real world, (23) 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. $\square$

B.2 Indifferentiability Analysis of Özen-Stam-Class

Özen and Stam [34] analyzed a wide class of double length compression functions, extending the single-length compression function result of Stam [41].

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

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 6. *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 Proposition 5, and we only highlight the differences. Our distinguisher $\mathcal{D}$ aims at finding an evaluation of OS that satisfies:

$$\mathrm{OS}(u, v, w) = (C_1^{\mathrm{post}}(u, v, w, 0), z) , \tag{24}$$

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 (24) is satisfied. The remainder of the analysis is the same as in the proof of Proposition 5. $\square$

C Indifferentiability Analysis of MDC-2 and MJH

In this section, we consider the MDC-2 and MJH compression functions.[9] 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):

MDC-2$(u, v, w) = (y, z)$, where:

$$\begin{aligned} c_1 &\leftarrow E(u, w) , \\ y &\leftarrow c_1 + w , \\ c_2 &\leftarrow E(v, w) , \\ z &\leftarrow c_2 + w . \end{aligned}$$

MJH$(u, v, w) = (y, z)$, where:

$$\begin{aligned} c_1 &\leftarrow E(v, u + w) , \\ y &\leftarrow c_1 + u + w , \\ c_2 &\leftarrow E(v, \sigma(u + w)) , \\ z &\leftarrow (c_2 + \sigma(u + w)) \cdot \theta + u . \end{aligned}$$

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

Proposition 7. *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.

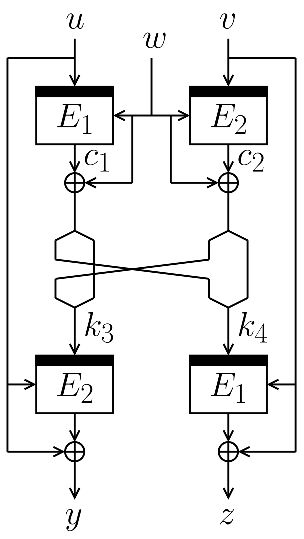
Proof. The proof is similar to the one of Proposition 5. Now, our distinguisher aims at finding an evaluation of MDC-2 that satisfies MDC-2$(u, v, w) = (w, z)$, and the same for MJH. The remainder of the analysis is almost identical to the proof of Proposition 5, and therefore omitted. $\square$

D Indifferentiability Analysis of JOS

In this section, we consider Jetchev et al.'s compression function (called JOS).[10] The analysis is slightly more complicated but in fact not much different. We consider the block cipher based

[9] We note that for our analysis it is not relevant whether the underlying ciphers are distinct or the same. Therefore, we consider the designs to be based on one single block cipher.

[10] We note that for our analysis it is not relevant whether the underlying ciphers are distinct or the same. Therefore, we consider the design to be based on one single block cipher.



$$
\begin{aligned}
&\text{MDC-4}(u,v,w) = (y,z),\ \text{where:}\\
&\quad c_1 \leftarrow E_1(u,w)\,,\\
&\quad c_2 \leftarrow E_2(v,w)\,,\\
&\quad k_3 \leftarrow c_2^l \| c_1^r + w\,,\\
&\quad y \leftarrow E_2(k_3,u) + u\,,\\
&\quad k_4 \leftarrow c_1^l \| c_2^r + w\,,\\
&\quad z \leftarrow E_1(k_4,v) + v\,.
\end{aligned}
$$

Fig. 20. The MDC-4 compression function. For convenience, the swapping at the end is omitted.

variant with the underlying matrix A as suggested in [33, Section 5.4.2].

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

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

Proposition 8. *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*

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

Proof. The proof is similar to the one of Proposition 5. Now, our distinguisher aims at finding an evaluation of $\text{JOS}(u,v,w)=(y,z)$ that satisfies $y+uz = u^2+u+v$. The remainder of the analysis is almost identical to the proof of Proposition 5, and therefore omitted. □

E Indifferentiability Analysis of 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 Figure 20. 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 as MDC-2, mainly due to the two sequential rounds. Differentiability is discussed in Appendix E.1, and indifferentiability in Appendix E.2.

E.1 Differentiability

In Proposition 9 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 Proposition 3, 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 9. *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:

$$c_2^l = {c_2'}^l . \tag{25}$$

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 (25), then the following condition naturally holds in the real world:

$$y := \text{MDC-4}(u, v, w)^l = \text{MDC-4}(u, v', w)^l =: y' . \tag{26}$$

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 (25). 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 (25), $\mathcal{D}$ returns 1;
(iii) Let (v, w, c_2) and (v', w, c_2') be such that (25) is satisfied;
(iv) Take $u \xleftarrow{\$} \{0,1\}^n$. If (26) 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 (25) 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 (26) 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 (18) 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 Proposition 3 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. □

E.2 Indifferentiability

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

Theorem 4. *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 $\mathcal{R}$.

The proof of Theorem 4 is similar to the proof of Theorem 3: 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 Section 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 Figure 21. 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 Figure 20) 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 $\mathcal{L}E_j^+[k](m) \neq \perp$ 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 $\mathcal{L}E_j^-[k](c) \neq \perp$ 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. 21. The simulator $\mathcal{S}$ for E used in the proof of Theorem 4. Here, $\bar{j} \in \{1, 2\}$ is the complement of $j \in \{1, 2\}$. Similar to Figure 18, see footnote 5 regarding the case where there is more than one solution to the if-clause of line 02–03.

Proof of Theorem 4

We formally prove Theorem 4. The proof is similar to the proof of Theorem 3, with only minor modifications. Let $\mathcal{S}$ be the simulator of Figure 21, 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 $q_S \leq q_R$ queries. By Definition 3, the goal is to bound:

$$\mathbf{adv}^{\text{iff}}_{\text{MDC-4},\mathcal{S}}(\mathcal{D}) = \left|\mathbf{Pr}\left(\mathcal{D}^{\text{MDC-4},E} = 1\right) - \mathbf{Pr}\left(\mathcal{D}^{\mathcal{R},\mathcal{S}} = 1\right)\right| . \tag{27}$$

As in Section 7.2, we first perform a PRP-PRF switch. $\widetilde{E}$ and $\widetilde{\mathcal{S}}$ are defined similarly as before, and we obtain for (27):

$$\mathbf{adv}^{\text{iff}}_{\text{MDC-4},\mathcal{S}}(\mathcal{D}) \leq \left|\mathbf{Pr}\left(\mathcal{D}^{\text{MDC-4},\widetilde{E}} = 1\right) - \mathbf{Pr}\left(\mathcal{D}^{\mathcal{R},\widetilde{\mathcal{S}}} = 1\right)\right| + \frac{2(4q_L + q_R)^2}{2^n} . \tag{28}$$

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 Figure 22. The notations $\mathcal{L}E_1$, $\mathcal{L}E_2$ and $\mathcal{L}\mathcal{R}$ are defined similarly as before.

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

$$\mathsf{cond}(\mathcal{L}E) = \left(\begin{array}{c} \exists\ j \in \{l, r\}, (k, m, c), (k', m', c') \in \mathcal{L}E : \\ (k, m, c) \text{ newer than } (k', m', c') \text{ and} \\ (c + m)^j \in \{k^j, k'^j, (c' + m')^j\} \end{array}\right) . \tag{29}$$

Event $\mathsf{cond}(\mathcal{L}E)$ is fairly the same as the event for the proof in Section 7.2 (equation (22)). 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 Figure 22 is always satisfied by at most one $((u, w, c_1), (v, w, c_2))$. In the remainder, we prove in Lemma 10 that $(\text{MDC-4}, \widetilde{E})$ 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 Lemma 11 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 (28), 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 $\mathcal{L}\mathcal{R}(u, v, w) \neq \perp$ return $(y, z) = \mathcal{L}\mathcal{R}(u, v, w)$
31 $(y, z) \xleftarrow{\$} \{0,1\}^{2n}$
32 return $\mathcal{L}\mathcal{R}(u, v, w) \leftarrow (y, z)$
```

```
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 $k = c_{\bar{j}}^l \| c_j^r + w$
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. 22. The worlds $(\text{MDC-4}, \widetilde{E})$ (left) and $(\mathcal{R}, \widetilde{\mathcal{S}})$ (right).

Lemma 10. *As long as* $\neg\mathsf{cond}(\mathcal{L}E)$*,* $(\text{MDC-4}, \widetilde{E})$ *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. $\mathcal{LE}_1^+[u](w) \neq \bot$ and $\mathcal{LE}_2^+[v](w) \neq \bot$. Note that in the real world, these elements could have been added to $\mathcal{LE}$ via $\mathcal{D}$ or via MDC-4. Let $c_1 = \mathcal{LE}_1^+[u](w)$ and $c_2 = \mathcal{LE}_2^+[v](w)$, 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:
   - $\mathcal{LE}_2^+[k_3](u) = \bot$ and $\mathcal{LE}_1^+[k_4](v) = \bot$. 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 $(u, w, c_1) \in \mathcal{LE}_1$ and $(v, w, c_2) \in \mathcal{LE}_2$ such that $c_j^l \| c_{\bar{j}}^r + w = k'$ and $u[j=2] + v[j=1] = m'$ for some $(k', m', c') \in \mathcal{LE}_j$. The former implies $k' = k_3[j=2] + k_4[j=1]$. This contradicts the condition that (k_3, u) is not in $\mathcal{LE}_2$ and (k_4, v) not in $\mathcal{LE}_1$. Therefore, the query (u, v, w) to $\mathcal{R}$ is new, and the response is randomly drawn from $\{0,1\}^{2n}$;
   - $\mathcal{LE}_2^+[k_3](u) \neq \bot$ and/or $\mathcal{LE}_1^+[k_4](v) \neq \bot$. Without loss of generality, assume the former and write $c_3 = \mathcal{LE}_2^+[k_3](u)$. In the real world, this query could not have been made in an earlier evaluation of MDC-4 (by virtue of $\mathsf{cond}(\mathcal{LE})$). 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 $\neg\mathsf{cond}(\mathcal{LE})$, 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 $\mathcal{LE}_1^+[k_4](v) \neq \bot$, the same reasoning applies to z, the second half of the query response. On the other hand, in case $\mathcal{LE}_1^+[k_4](v) = \bot$, the previous bullet carries over to the z-part.

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

1. $\neg\, \exists\ (u, w, c_1) \in \mathcal{LE}_1, (v, w, c_2) \in \mathcal{LE}_2 : m = u[j=2] + v[j=1]$ and $k = c_j^l \| c_{\bar{j}}^r + w$. 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. $\exists\ (u, w, c_1) \in \mathcal{LE}_1, (v, w, c_2) \in \mathcal{LE}_2 : m = u[j=2] + v[j=1]$ and $k = c_j^l \| c_{\bar{j}}^r + w$. By $\neg\mathsf{cond}(\mathcal{LE})$, 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 $\neg\mathsf{cond}(\mathcal{LE})$), 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 $\mathcal{LE}^+[k](m) = c$ 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 11. $\mathbf{Pr}\left(\mathsf{cond}(\mathcal{LE})\ \textit{for}\ (\text{MDC-4}, \widetilde{E})\right) \leq \frac{2(4q_L + q_R)^2}{2^{n/2}}$ *and* $\mathbf{Pr}\left(\mathsf{cond}(\mathcal{LE})\ \textit{for}\ (\mathcal{R}, \widetilde{\mathcal{S}})\right) \leq \frac{2q_R^2}{2^{n/2}}$.

Proof. The proof is the same as the proof of Lemma 9, 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 $4q_L + q_R$. $\square$