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

Bart Mennink

Dept. Electrical Engineering, ESAT/COSIC, KU Leuven, and IBBT, 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.
Keywords. double block length; hash function; collision resistance; preimage resistance, 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 of double length hashing dates back to the work of Meyer and Schilling [19], with the introduction of the MDC-2 and MDC-4 compression functions in 1988. In recent years, the design methodology got renewed attention in the works of [2, 4, 7, 9, 10, 12, 17, 22, 29]. Double length hash functions have an obvious advantage over classical block cipher based functions such as Davies-Meyer and Matyas-Meyer-Oseas [24, 28]: the same type of underlying primitive allows for a larger compression function. Yet, for double length compression functions it is harder to achieve optimal n-bit collision and $2n$-bit preimage security.

We focus on the simplest 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 [8] and Hirose's function [6] (see Fig. 1), as well as Stam's supercharged single call Type-I compression function design [27, 28] (reconsidered in [14]) and the generalized designs by Hirose [5] and Özen and Stam [22]. 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 also hold in the iteration, when a proper domain extender is applied [1]. Lucks [16] introduced a compression function that allows for collisions in about $2^{n/2}$ queries, but achieves optimal collision resistance in the iteration. Members of the DBL^n class are the MDC-2 and MDC-4 compression functions [19], the MJH construction [10], and a construction by Jetchev et al. [7]. 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

[1] In the iteration collision resistance is proven up to $2^{3n/5}$ queries for MDC-2 [29] and $2^{2n/3}$ queries for MJH [10].

a higher level of collision and preimage resistance than MDC-2 [17], 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. [21], 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. [23] introduced double length compression functions based on five compression function calls with n-bit output. Related are also Lucks wide-pipe design [15] and its generalization by Nandi [20].

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 (Fig. 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 Sect. 4).

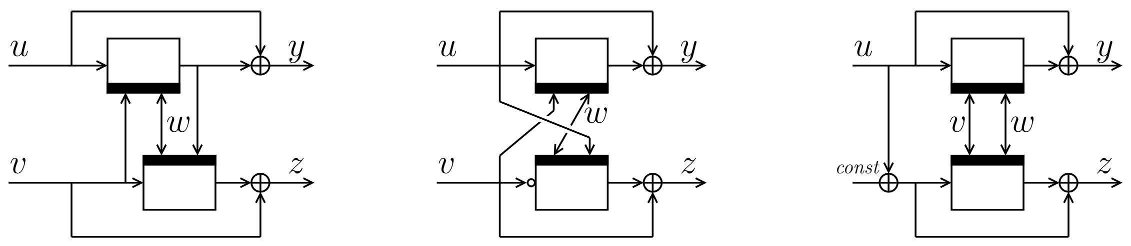
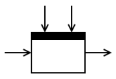
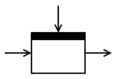


Fig. 1. From left to right, Tandem-DM, Abreast-DM, and Hirose's compression function [8, 6]. All wires carry n bits. For Abreast-DM, the circle $\circ$ denotes bit complementation. For Hirose's function, *const* is any non-zero constant.

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 [27] 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.

As a first contribution, we consider F^2, and prove that for a very large class of functions of 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. [7] 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. 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

Table 1. Asymptotic ideal cipher model security guarantees of known double length compression functions in the classes DBL^{2n} (first) and DBL^{n} (next). A more detailed security and efficiency comparison of some of these functions is presented by Bos et al. [3, App. A].

compression function	E-calls	collision security	preimage security	underlying cipher
Lucks'	1	$2^{n/2}$	2^{n}	
Stam's	1	2^{n} [28]	2^{n} [28]	
Tandem-DM	2	2^{n} [12]	2^{2n} [2, 13]	
Abreast-DM	2	2^{n} [4, 9]	2^{2n} [2, 13]	
Hirose's	2	2^{n} [6]	2^{2n} [2, 13]	
Hirose-class	2	2^{n} [5]	2^{n} [5]	
Özen-Stam-class	2	2^{n} [22]	2^{n} [22]	
MDC-2	2	$2^{n/2}$	2^{n}	
MJH	2	$2^{n/2}$	2^{n}	
Jetchev et al.'s	2	$2^{2n/3}$ [7]	2^{n} [7]	
MDC-4	4	$2^{5n/8}$ [17]	$2^{5n/4}$ [17]	
Our proposal	**3**	$\mathbf{2^{n}}$	$\mathbf{2^{3n/2}}$	

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 Fig. 2.

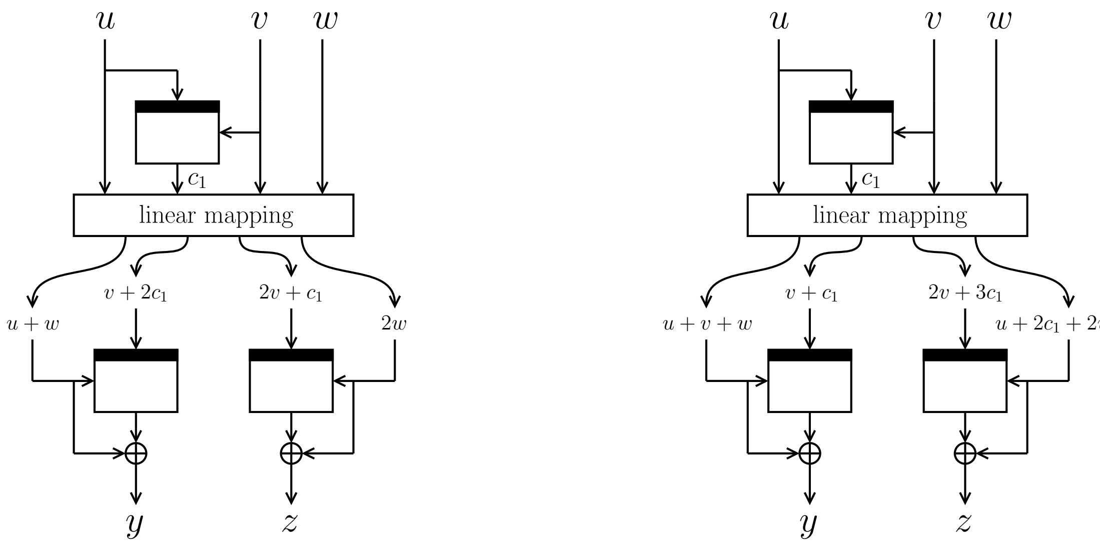


Fig. 2. Two example compression functions from the family of functions introduced and evaluated in this work. For these constructions, all wires carry $n = 128$ bits, and the arithmetic is done over $GF(2^{128})$. We further elaborate on these designs and their derivations in Sect. 4.

The 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 Armknecht et al. and Lee et al. [2, 13]) 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 [26] shows that, when the compression function behaves "sufficiently random", one may expect a preimage in approximately $2^{5n/3}$ queries (cf. Sect. 2). The asymptotic preimage bound of $2^{3n/2}$ found in this work closely approaches this generic bound.

Outline. We present and formalize the security model in Sect. 2. Then, in Sect. 3 we derive our impossibility result on F^2. We propose and analyze our family of compression functions in Sects. 4-6. This work is concluded in Sect. 7.

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 Fig. 3. This generic design is a generalization of the permutation based hash function construction described by Rogaway and Steinberger [26]. In fact, it is straightforward to generalize the main findings of [26] 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 [26] 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.
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 $E \overset{\$}{\leftarrow} \mathrm{Bloc}(n)$ randomly sampled from $\mathrm{Bloc}(n)$. $\mathcal{A}$ is information-theoretic: it has unbounded computational power, 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

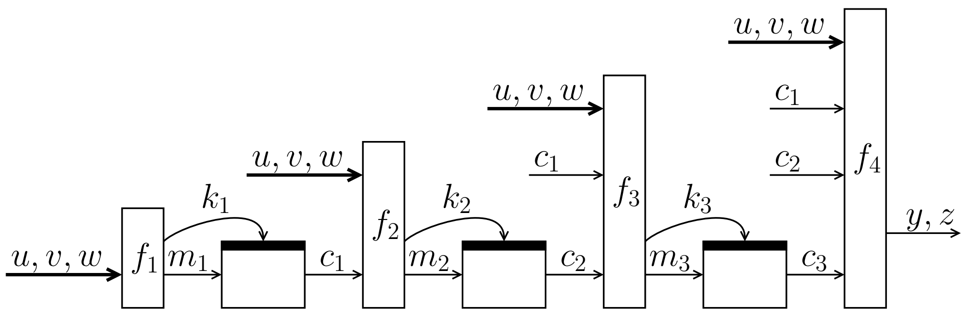


Fig. 3. $F^3 : \{0,1\}^{3n} \to \{0,1\}^{2n}$ making three block cipher evaluations.

$(u,v,w),(u',v',w')$ such that $F^r(u,v,w) = F^r(u',v',w')$ and Q contains all queries required for these evaluations of F^r. We define by

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

the probability that $\mathcal{A}$ succeeds in this. By $\mathbf{adv}^{\mathrm{coll}}_{F^r}(q)$ we define the maximum collision advantage taken over all adversaries making q queries.

For preimage resistance, we focus on everywhere preimage resistance [25], which captures preimage security for every point of $\{0,1\}^{2n}$. Before making any queries to its oracle, a preimage-finding adversary $\mathcal{A}$ first decides on a range point $(y,z) \in \{0,1\}^{2n}$. Then, 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. We define by

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

the probability that $\mathcal{A}$ succeeds, maximized over all possible choices for (y,z). By $\mathbf{adv}^{\mathrm{epre}}_{F^r}(q)$ we define the maximum (everywhere) preimage advantage taken over all adversaries making q queries.

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 their underlying block cipher $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$. 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 [7], 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 $F^2 : \{0,1\}^{3n} \to \{0,1\}^{2n}$ be a compression function as described in Sect. 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*

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

Then, one can expect collisions for F^2 after $2^{n/2}$ queries.

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 [26].

- Make α queries $(k_1,m_1) \to c_1$ 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 balls-and-bins principle[2], 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 the cipher 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 $g_2(u,v,w;c_1,c_2) = g_2(u',v',w';c_1',c_2')$.

In the last round one expects to find a collision if $\alpha^2/2^n = 1$, or equivalently if $\alpha = 2^{n/2}$. In total, the attack is done in approximately $2 \cdot 2^{n/2}$ 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 [18] 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, 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. □

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

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

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

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.

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

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. [7], for which collision resistance up to $2^{2n/3}$ queries is proven.

4 Double Length Hashing with Three *E*-calls

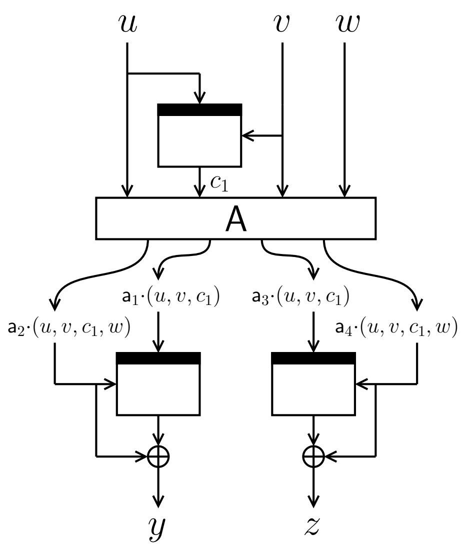
Motivated by the negative result of Sect. 3, we target the existence of double length hashing with three block cipher calls. We introduce a family of double length compression functions making three cipher calls that achieve asymptotically optimal 2^n collision resistance and preimage resistance significantly beyond the birthday bound (up to $2^{3n/2}$ queries). We note that, although the preimage bound is non-optimal, it closely approaches the generic bound dictated by the pigeonhole-birthday attack (Sect. 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 Fig. 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 from $\{0,1\}^n$, 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 Fig. 3 in particular, this can impossibly be achieved: one block cipher "processes" at most $2n$ out of $3n$ input 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 Fig. 4. Here,

$$\mathsf{A} = \begin{pmatrix} \mathsf{a}_1 \\ \mathsf{a}_2 \\ \mathsf{a}_3 \\ \mathsf{a}_4 \end{pmatrix} = \begin{pmatrix} \mathsf{a}_{11} & \mathsf{a}_{12} & \mathsf{a}_{13} & 0 \\ \mathsf{a}_{21} & \mathsf{a}_{22} & \mathsf{a}_{23} & \mathsf{a}_{24} \\ \mathsf{a}_{31} & \mathsf{a}_{32} & \mathsf{a}_{33} & 0 \\ \mathsf{a}_{41} & \mathsf{a}_{42} & \mathsf{a}_{43} & \mathsf{a}_{44} \end{pmatrix} \qquad (2)$$



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 $GF(2^n)$.

is a 4×4 matrix over $GF(2^n)$. 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 Sect. 4.1 and on the preimage resistance in Sect. 4.2.

4.1 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`.

Theorem 1. *Let $n \in \{0,1\}^n$. Suppose A satisfies `colreq`. Then, for any positive integral values t_1, t_2,*

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

The proof is given in Sect. 5. The basic proof idea is similar to existing proofs in the literature (e.g. [17, 29]) 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 Armknecht et al. [2] and Lee et al. [11, 13] for the preimage resistance

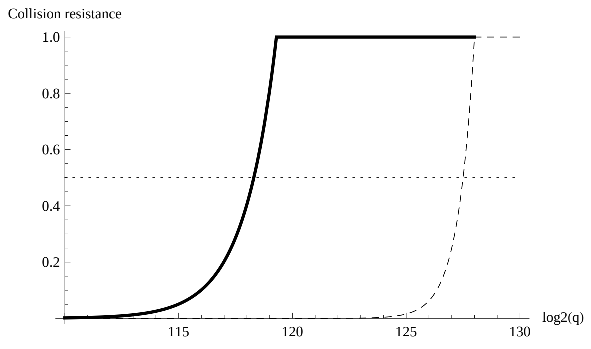


Fig. 5. For $n = 128$, the function $\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q)$ of (3) for $\varepsilon = 1/35$ and for the particular choice of values t_1, t_2 (solid line) and the optimal bound of $q(q+1)/2^{2n}$ (dashed line).

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 (3). 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

$$\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q) \le \frac{5 \cdot 2^{2n\varepsilon} q + 10 \cdot 2^{n\varepsilon} q + 11q}{2^n - q} + \frac{q}{2^n - q} + 3 \cdot 2^n \left(\frac{eq}{2^{n\varepsilon}(2^n - q)} \right)^{2^{n\varepsilon}} .$$

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, the F^3_{A} compression function achieves close to optimal 2^n collision security for $n \to \infty$. For $n = 128$, the bound on $\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}$ is depicted in Fig. 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.

4.2 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 (Sect. 2)—we achieve preimage resistance up to $2^{3n/2}$ queries, much better than the preimage bounds on MDC-2 and MDC-4 [17], 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

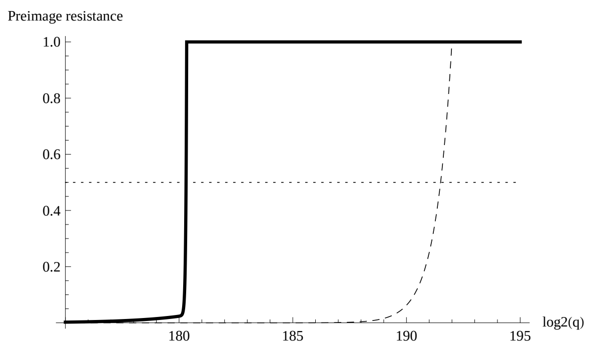


Fig. 6. For $n = 128$, the function $\mathbf{adv}^{\text{epre}}_{F^3_{\mathsf{A}}}(q)$ of (5) for the particular choice of values t (solid line) and the optimal bound of $q^2/2^{3n}$ (dashed line). The steepness of our bound is caused by the last term of (5) which explodes for q approaching $t2^n$ due to its decreasing exponent.

conditions over the field $GF(2^{128})$ are

$$\begin{pmatrix} 0\,1\,2\,0 \\ 1\,0\,0\,1 \\ 0\,2\,1\,0 \\ 0\,0\,0\,2 \end{pmatrix}, \qquad \begin{pmatrix} 0\,1\,1\,0 \\ 1\,1\,0\,1 \\ 0\,2\,3\,0 \\ 1\,0\,2\,2 \end{pmatrix}. \tag{4}$$

These are the matrices corresponding to the compression functions of Fig. 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 (4) has as minimal zeroes as possible, which reduces the amount of computation.

Theorem 2. *Let $n \in \{0,1\}^n$. Suppose A satisfies* `prereq`*. Then, for any positive integral value t, provided $t \leq q$,*

$$\mathbf{adv}^{\text{epre}}_{F^3_{\mathsf{A}}}(q) \leq \frac{6t^2 + 18t + 26}{2^n - 2} + 4 \cdot 2^n \left(\frac{4eq}{t2^n}\right)^{t/2} + 8q \left(\frac{8eq}{t2^n}\right)^{\frac{t2^n}{4q}}. \tag{5}$$

The proof is given in Sect. 6. As for the bound on the collision resistance (Thm. 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

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

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

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

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 Fig. 6. 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 a preimage-finding adversary that with high probability succeeds in at most $O(2^{3n/2})$ queries. Let $\alpha \in \mathbb{N}$. The adversary proceeds as follows.

- Make $\alpha 2^n$ queries to the block cipher corresponding to the bottom-left position of Fig. 4. One expects to find α tuples (k_2, m_2, c_2) that satisfy $m_2 + c_2 = y$;
- Repeat the first step for the bottom-right position. One expects to find α tuples (k_3, m_3, c_3) satisfying $m_3 + c_3 = z$;
- 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;
- 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.

5 Proof of Thm. 1

The proof of collision resistance of F^3_{A} follows the basic spirit of [17], but crucially differs in the way the probability bounds are computed. A new approach here is the usage of wish lists. While the idea of wish lists is not new—it has been introduced by Armknecht et al. [2] and Lee et al. [11, 13] for double block length compression functions, and used by Mennink [17] 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 Lem. 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 Fig. 7. This means,

$$\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q) = \mathbf{Pr}\left(\mathsf{coll}(Q_q)\right). \tag{6}$$

For the sake of readability of the proof, we label the block cipher positions in Fig. 7 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

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

By basic probability theory, we obtain for (6):

$$\mathbf{Pr}\left(\mathsf{coll}(Q_q)\right) \leq \mathbf{Pr}\left(\mathsf{coll}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) + \mathbf{Pr}\left(\mathsf{aux}(Q_q)\right). \tag{7}$$

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

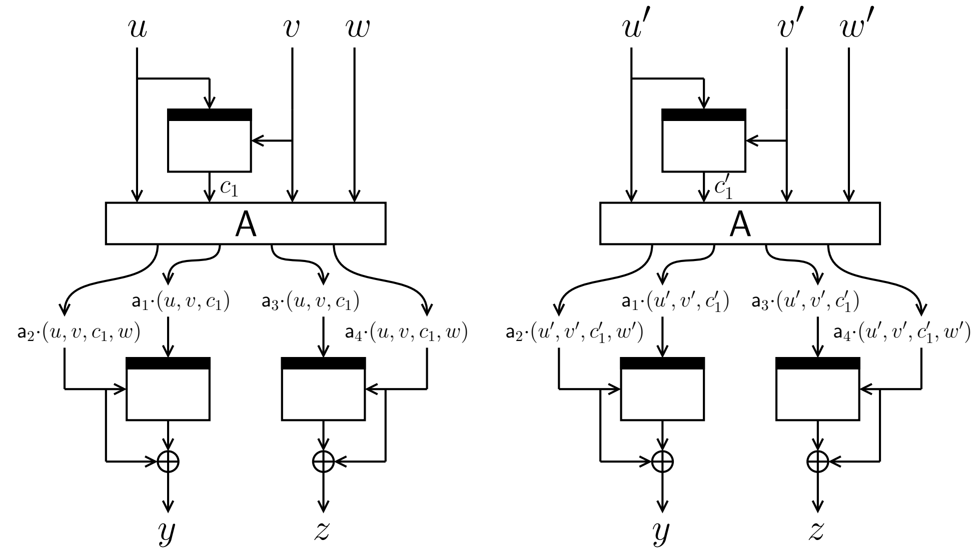

u
v
w
c_1
A
$\mathsf{a}_1\cdot(u,v,c_1)$
$\mathsf{a}_3\cdot(u,v,c_1)$
$\mathsf{a}_2\cdot(u,v,c_1,w)$
$\mathsf{a}_4\cdot(u,v,c_1,w)$
y
z
u'
v'
w'
c_1'
A
$\mathsf{a}_1\cdot(u',v',c_1')$
$\mathsf{a}_3\cdot(u',v',c_1')$
$\mathsf{a}_2\cdot(u',v',c_1',w')$
$\mathsf{a}_4\cdot(u',v',c_1',w')$
y
z

Fig. 7. Configuration $\mathsf{coll}(Q)$. The configuration is satisfied if Q contains six (possibly the same) queries that satisfy this setting. We require $(u, v, w) \neq (u', v', w')$.

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 distinct among the cases a query occurs in both words at the same position. We define for binary $\alpha_1, \alpha_2, \alpha_3$ by $\mathsf{coll}_{\alpha_1\alpha_2\alpha_3}(Q)$ the configuration $\mathsf{coll}(Q)$ of Fig. 7 restricted to

$$1L = 1R \iff \alpha_1 = 1\,, \qquad 2L = 2R \iff \alpha_2 = 1\,, \qquad 3L = 3R \iff \alpha_3 = 1\,.$$

By construction, $\mathsf{coll}(Q_q) \Rightarrow \bigvee_{\alpha_1,\alpha_2,\alpha_3\in\{0,1\}} \mathsf{coll}_{\alpha_1\alpha_2\alpha_3}(Q_q)$, and from (6-7) we obtain the following bound on $\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q)$:

$$\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q) \leq \sum_{\alpha_1,\alpha_2,\alpha_3\in\{0,1\}} \mathbf{Pr}\left(\mathsf{coll}_{\alpha_1\alpha_2\alpha_3}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) + \mathbf{Pr}\left(\mathsf{aux}(Q_q)\right). \tag{8}$$

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 Lems. 1-4. The proofs are rather similar, and we only bound the probability on $\mathsf{coll}_{000}(Q_q)$ in full detail (Lem. 1). A bound on $\mathbf{Pr}\left(\mathsf{aux}(Q_q)\right)$ is derived in Lem. 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 Fig. 8. The block cipher queries at positions a and $!a$ are required to be different, and so are the ones are 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 any 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 contributes to multiple "different" positions, e.g. $1L$ and $2R$. Note that by construction, a winning query can contribute to at most three block

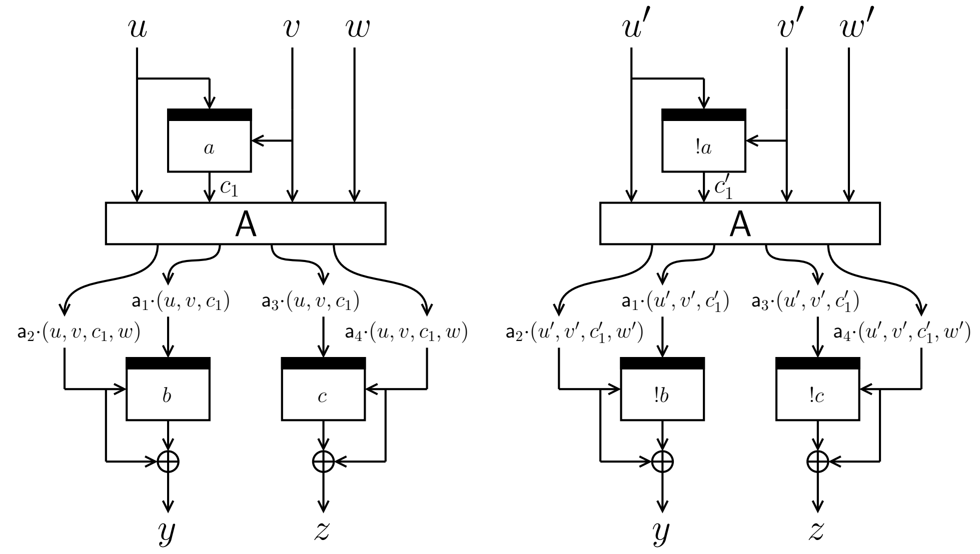

u
v
w
a
c_1
A
$\mathsf{a}_1\cdot(u, v, c_1)$
$\mathsf{a}_3\cdot(u, v, c_1)$
$\mathsf{a}_2\cdot(u, v, c_1, w)$
$\mathsf{a}_4\cdot(u, v, c_1, w)$
b
c
y
z
u'
v'
w'
$!a$
c'_1
A
$\mathsf{a}_1\cdot(u', v', c'_1)$
$\mathsf{a}_3\cdot(u', v', c'_1)$
$\mathsf{a}_2\cdot(u', v', c'_1, w')$
$\mathsf{a}_4\cdot(u',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

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

cipher positions of Fig. 8. In total, there are 26 sets of positions at which the winning query can contribute 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:

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

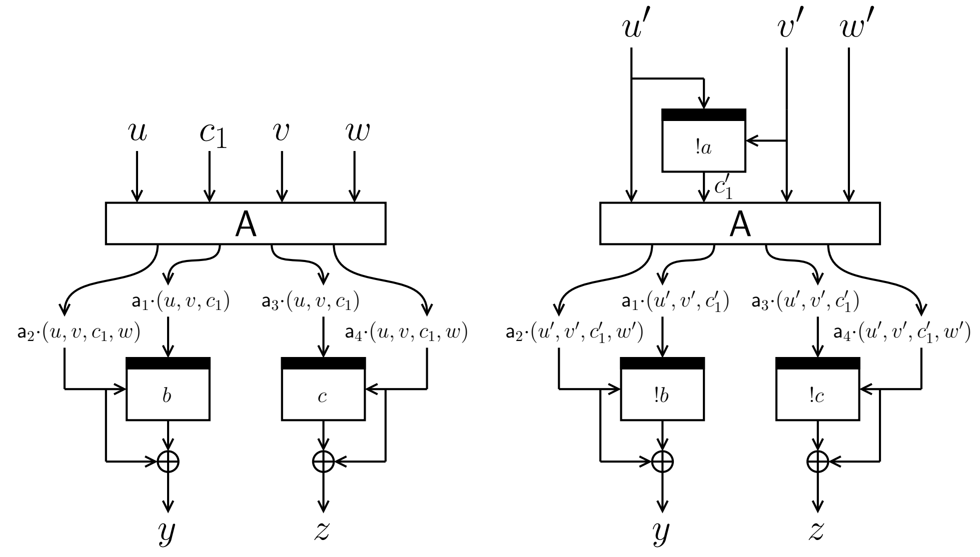
Note that there are many more symmetric cases among these, but we are not allowed to discard those as these may result in effectively different collisions. For $j = 1, \dots, 13$ we denote by $\mathsf{coll}_{000:\mathcal{S}_j}(Q)$ configuration $\mathsf{coll}_{000}(Q)$ with the restriction that the winning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

$$\mathbf{Pr}\left(\mathsf{coll}_{000}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) \leq \sum_{j=1}^{13} \mathbf{Pr}\left(\mathsf{coll}_{000:\mathcal{S}_j}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right). \tag{9}$$

$\boldsymbol{\mathsf{coll}_{000:\mathcal{S}_1}(Q_q)}$**.** Rather than considering the success probability of the ith query, and then sum over $i = 1, \dots, q$ (as is done in the analysis of [4, 5, 6, 7, 9, 12, 17, 22, 28], 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 Fig. 9). If a new query is made, suppose it fits this configuration for some other queries in the query history (the new query appearing at least once), jointly representing queries at positions $\{2L, 3L, 1R, 2R, 3R\}$. Then the corresponding 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, but different combinations of queries may define the same wish. The latter does, however, not invalidate the analysis: this is covered by the upper bound on $\mathcal{W}_{\mathcal{S}_1}$ that will be computed later in the proof, and will simply render a slightly worse bound.

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[3]. Clearly, in order to find a collision for

[3] 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
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
$\mathsf{a}_1{\cdot}(u',v',c'_1)$
$\mathsf{a}_3{\cdot}(u',v',c'_1)$
$\mathsf{a}_2{\cdot}(u',v',c'_1,w')$
$\mathsf{a}_4{\cdot}(u',v',c'_1,w')$
$!b$
$!c$
y
z

Fig. 9. Configuration $\mathsf{coll}_{000:\mathcal{S}_1}(Q)$. We require $(u,v,w) \neq (u',v',w')$.

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 Fig. 9. 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 $t_1 t_2^2/(2^n - q)$.

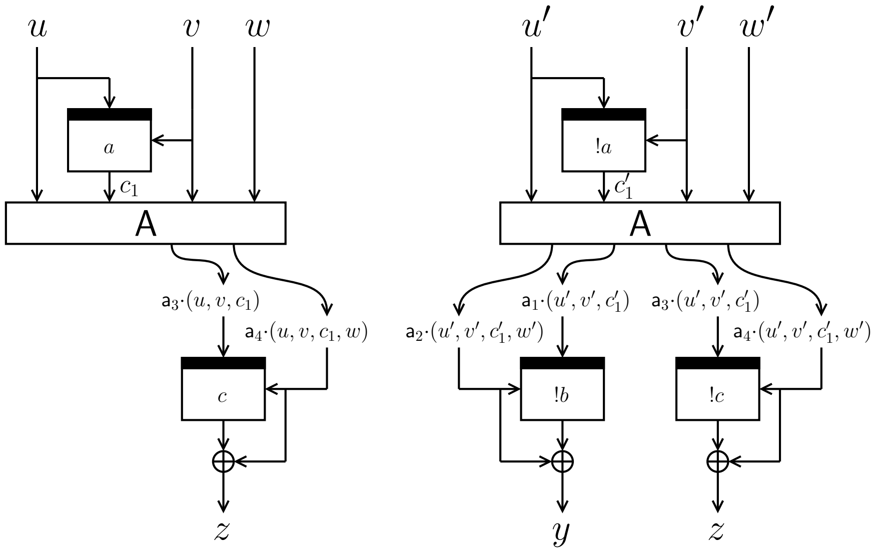
$\mathsf{coll}_{000:\mathcal{S}_j}(Q_q)$ for $j = 2, 3$. Both cases are the same by symmetry, and we consider $\mathcal{S}_2$ only.

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 Fig. 10). 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 $1/(2^n - q)$ (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 $\mathsf{a}_{12} \neq \mathsf{a}_{32}$



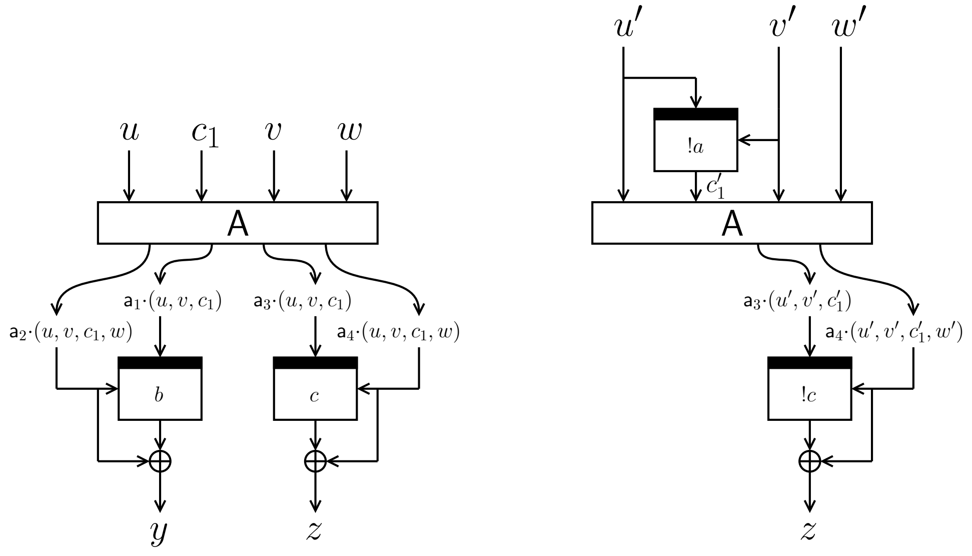
u
v
w
a
c_1
A
$\mathsf{a}_3\cdot(u,v,c_1)$
$\mathsf{a}_4\cdot(u,v,c_1,w)$
c
z
u'
v'
w'
$!a$
c_1'
A
$\mathsf{a}_1\cdot(u',v',c_1')$
$\mathsf{a}_3\cdot(u',v',c_1')$
$\mathsf{a}_2\cdot(u',v',c_1',w')$
$\mathsf{a}_4\cdot(u',v',c_1',w')$
$!b$
$!c$
y
z

Fig. 10. Configuration $\mathsf{coll}_{000:\mathcal{S}_2}(Q)$. We require $(u,v,w) \neq (u',v',w')$.

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 Fig. 11). For any set of queries that fits this 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 $(u, v, c_1) = (\mathsf{a}_1 \cdot (u', v', c_1'), \mathsf{a}_2 \cdot (u', v', c_1', w'), y - \mathsf{a}_2 \cdot (u', v', c_1', w'))$.



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

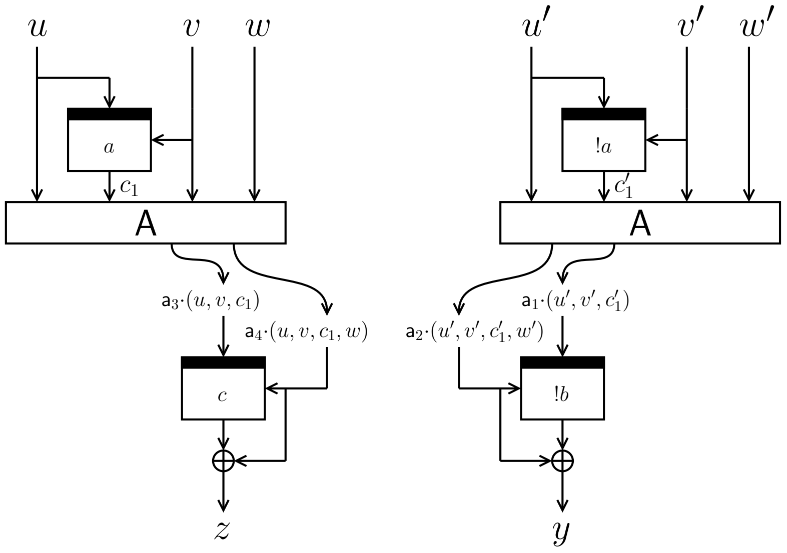
We upper bound the number of solutions to the configuration of Fig. 11 after q queries. By $\neg\mathsf{aux}_1(Q_q)$, the configuration has at most t_1 choices for $3L, 3R$. For any such choice, by $\neg\mathsf{aux}_3(Q_q)$ we have at most t_2 choices for $1R$. The queries at positions $1R$ and $3R$ 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 Fig. 12). For any set of queries that fits 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 $y = z$.



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

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 $\mathcal{W}^{\neq}_{\mathcal{S}_9}$ contains only wishes of which the corresponding queries at positions $3L, 2R$ are different. Note that by construction, a wish may occur in both 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 Fig. 12, 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 $(t_1t_2 + t_2q)/(2^n - q)$.

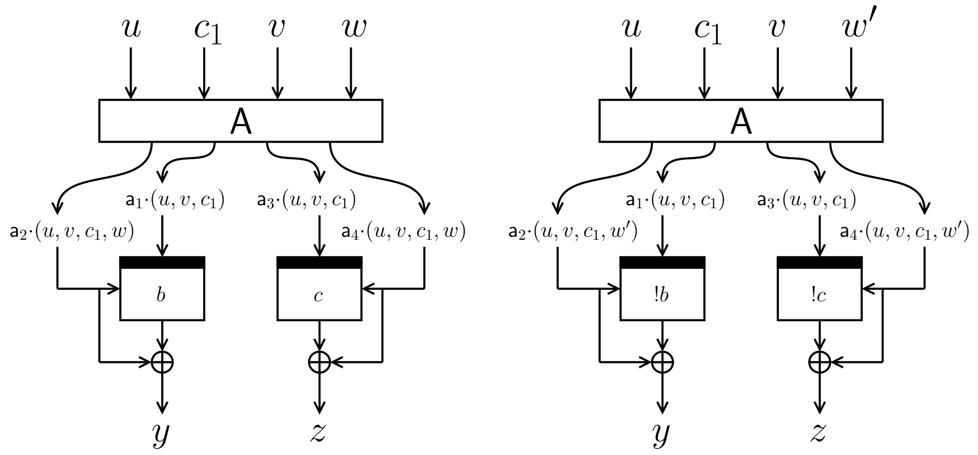
$\mathsf{coll}_{000:\mathcal{S}_j}(Q_q)$ for $j = 10, 11, 12$. For these cases, the analysis for $\mathcal{S}_4$ directly applies.

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

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

Lemma 2. $\mathbf{Pr}\left(\mathsf{coll}_{100}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) \leq \frac{2q+2t_1t_2}{2^n-q}$.

Proof. Sub-configuration $\mathsf{coll}_{100}(Q_q)$ is given in Fig. 13. Note that here we in particular have $(u, v, c_1) = (u', v', c_1')$ as $1L = 1R$.



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

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

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

As before, we find

$$\mathbf{Pr}\left(\mathsf{coll}_{100}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) \leq \sum_{j=1}^{4} \mathbf{Pr}\left(\mathsf{coll}_{100:\mathcal{S}_j}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right). \tag{10}$$

$\mathsf{coll}_{100:\mathcal{S}_j}(Q_q)$ 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 $t_1t_2/(2^n - q)$.

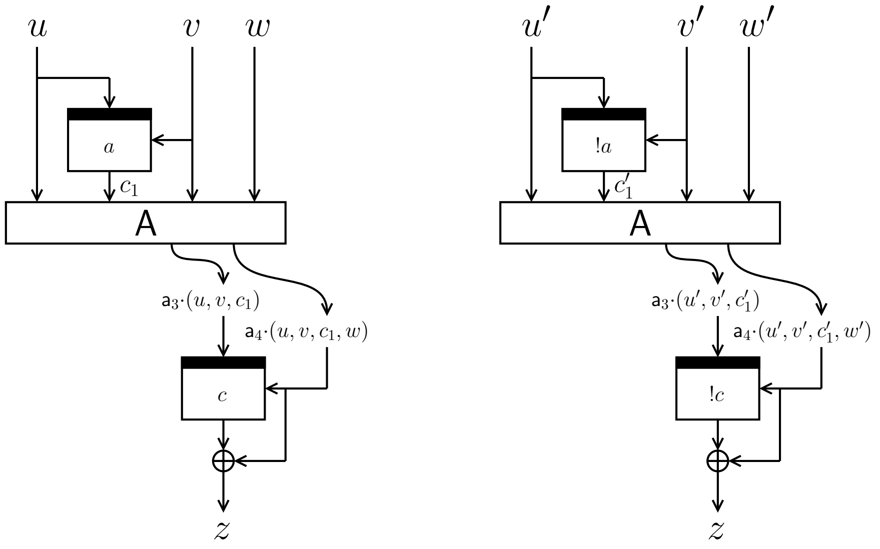
$\mathsf{coll}_{100:\mathcal{S}_3}(Q_q)$. By construction, there must be a query (k, m, c) in the query history (corresponding to position $1L$) that satisfies $\mathsf{a}_1\cdot(k, m, c) = \mathsf{a}_3\cdot(k, m, c)$. So the problem shifts to bounding the probability that the adversary ever finds a 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 (10). □

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 Fig. 14. Note that here we in particular have $\mathsf{a}_1 \cdot (u, v, c_1) = \mathsf{a}_1 \cdot (u', v', c_1')$ and $\mathsf{a}_3 \cdot (u, v, c_1, w) = \mathsf{a}_3 \cdot (u', v', c_1', w')$.



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. 14. 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 $\mathsf{a}_3 \cdot (u, v, c_1, w) = \mathsf{a}_3 \cdot (u', v', c_1', w')$.

The approach is similar to the one for Lem. 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 at which the winning query can contribute:

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

As before, we find

$$\mathbf{Pr}\left(\mathsf{coll}_{010}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) \leq \sum_{j=1}^{4} \mathbf{Pr}\left(\mathsf{coll}_{010:\mathcal{S}_j}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right). \tag{11}$$

$\mathbf{coll_{010:\mathcal{S}_1}(Q_q)}$. 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 $|\mathcal{W}_{\mathcal{S}_2}| \leq t_2^2 q$, and hence in this setting a collision is found with probability at most $t_2^2 q/(2^n - q)$.

$\mathsf{coll}_{010:\mathcal{S}_3}(Q_q)$. 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 $q/(2^n - q)$.

$\mathsf{coll}_{010:\mathcal{S}_4}(Q_q)$. 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 (11). □

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

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

Proof. Note that $\mathsf{aux}_1(Q_q)$ essentially equals $\mathsf{help}_1(Q_q)$ of [17, Sect. 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 [17, Sect. 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.

$\mathsf{aux}_1(Q_q)$. 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

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

By Markov's inequality, we obtain

$$\mathbf{Pr}\left(\mathsf{aux}_1(Q_q)\right) \leq \frac{q^2}{t_1(2^n - q)}. \tag{12}$$

$\mathsf{aux}_k(Q_q)$ for $k \in \{2, 3, 4\}$. 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 $\binom{q}{t_2}\left(\frac{1}{2^n - q}\right)^{t_2} \leq \left(\frac{eq}{t_2(2^n - q)}\right)^{t_2}$, where we use Stirling's approximation ($t! \geq (t/e)^t$ for any t). Considering any possible choice for z, we obtain for $k = 2, 3, 4$:

$$\mathbf{Pr}\left(\mathsf{aux}_k(Q_q)\right) \leq 2^n \left(\frac{eq}{t_2(2^n - q)}\right)^{t_2}. \tag{13}$$

The claim is obtained by adding (12-13). □

From (8) and the results of Lems. 1-5 we conclude for $\mathbf{adv}^{\mathrm{coll}}_{F^3_{\mathsf{A}}}(q)$:

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

This completes the proof of Thm. 1.

6 Proof of Thm. 2

The proof of preimage resistance of F^3_{A} follows the basic spirit of [17]. 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 Fig. 15. This means,

$$\mathbf{adv}^{\mathrm{epre}}_{F^3_{\mathsf{A}}}(q) = \mathbf{Pr}\left(\mathsf{pre}(Q_q)\right). \tag{14}$$

We inherit the notation of Sect. 5. 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.

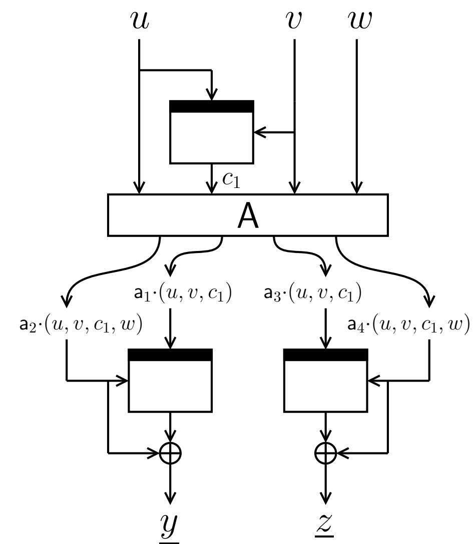

u
v
w
c_1
A
$\mathsf{a}_1\cdot(u,v,c_1)$
$\mathsf{a}_3\cdot(u,v,c_1)$
$\mathsf{a}_2\cdot(u,v,c_1,w)$
$\mathsf{a}_4\cdot(u,v,c_1,w)$
$\underline{y}$
$\underline{z}$

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

For the analysis of the preimage resistance, we use the idea of free super queries [2, 11, 13, 17]. 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 [2, 13], 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 $\mathbf{Pr}(\mathsf{pre}(Q_q))$ 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

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

Note that $\mathsf{aux}_2(Q_q), \mathsf{aux}_3(Q_q), \mathsf{aux}_4(Q_q)$ equal the ones of Sect. 5, but we reintroduce them for convenience. By basic probability theory, we obtain for (14):

$$\mathbf{Pr}(\mathsf{pre}(Q_q)) \le \mathbf{Pr}(\mathsf{pre}(Q_q)\wedge\neg\mathsf{aux}(Q_q)) + \mathbf{Pr}(\mathsf{aux}(Q_q))\,. \tag{15}$$

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

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

Proof. We consider the probability of the adversary finding a solution to configuration $\mathsf{pre}(Q_q)$ of Fig. 15 such that Q_q satisfies $\neg\mathsf{aux}(Q_q)$. The proof shows similarities with the proof of Lem. 1, We call a (normal or super) query winning if it makes the configuration satisfied for any other queries in the query history *strictly before* this winning query is made. It may be the case that a winning query contributes to two or three positions in the configuration. In more detail, one can identify the following 7 sets of positions at which the winning query can contribute:

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

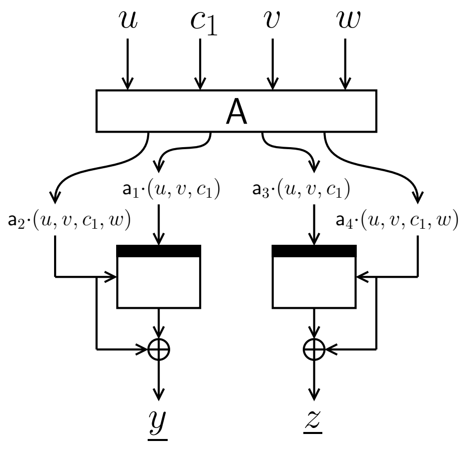
For $j = 1,\ldots,7$ we denote by $\mathsf{pre}_{\mathcal{S}_j}(Q_q)$ configuration $\mathsf{pre}(Q_q)$ with the restriction that the winning query *must* contribute to the positions in $\mathcal{S}_j$. Recall that a winning query may consist of different queries if it is a super query. By basic probability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

$$\mathbf{Pr}(\mathsf{pre}(Q_q)\wedge\neg\mathsf{aux}(Q_q)) \le \sum_{j=1}^{7} \mathbf{Pr}\left(\mathsf{pre}_{\mathcal{S}_j}(Q_q)\wedge\neg\mathsf{aux}(Q_q)\right). \tag{16}$$

$\boldsymbol{\mathsf{pre}_{\mathcal{S}_1}(Q_q)}$**.** In this case, the winning query may be a normal query or a super query. As is done in the proof of Lem. 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 Fig. 16). For any pair of queries that fits 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.

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 Fig. 16. By $\neg\mathsf{aux}_4(Q_q)$, the configuration



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

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

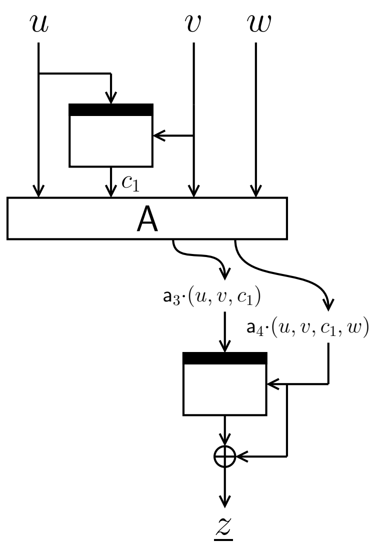


Fig. 17. Configuration $\mathsf{pre}_{\mathcal{S}_2}(Q)$.

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 Fig. 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 $t/2^{n-1}$.
- $2L \neq 3L$**.** 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 $\frac{t2^n}{2^{n-1}(2^{n-1}-1)}$.

$\mathsf{pre}_{\mathcal{S}_7}(Q_q)$**.** 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.

- $1L = 2L = 3L$**.** 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 $\mathsf{A} - \left[\left(\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right)/\left(\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right)\right]$, and we find $|\mathcal{W}_{\mathcal{S}_1}| = 1$. Hence, in this setting a preimage is found with probability at most $1/2^{n-1}$.
- $1L = 2L \neq 3L$ **or** $1L = 3L \neq 2L$**.** 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 $\mathsf{A} - \left[\left(\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right)/\left(\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right)\right]$. We find $|\mathcal{W}_{\mathcal{S}_7}| \leq 2^n$, and hence in this setting a preimage is found with probability at most $\frac{2^n}{2^{n-1}(2^{n-1}-1)}$.

- $\boldsymbol{2L = 3L \neq 1L}$**.** 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 $\frac{2^n}{2^{n-1}(2^{n-1}-1)}$.
- $\boldsymbol{\{1L, 2L, 3L\}}$ **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 (16):
$$\begin{aligned}\mathbf{Pr}\left(\mathsf{pre}(Q_q) \wedge \neg\mathsf{aux}(Q_q)\right) &\leq \frac{3t^2+3t+1}{2^{n-1}} + \frac{(3t+3)2^n}{2^{n-1}(2^{n-1}-1)} + \frac{2^{2n}}{2^{n-1}(2^{n-1}-1)(2^{n-1}-2)} \\ &\leq \frac{6t^2+18t+26}{2^n-2} \, ,\end{aligned}$$
where we use that $1/(2^{n-1}-2) \leq 3/2^n$ for $n \geq 4$. □

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 [17, Sect. 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 (14-15) and Lems. 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 Thm. 2.

7 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 [27, 28], Lee and Steinberger [14], and Jetchev et al. [7], our design does not make use of full field multiplications. The example matrices A given in (4) 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? Given the negative collision resistance result for a wide class of compression functions F^2, is it possible to achieve optimal collision security *in the iteration* anyhow? This question is beyond the scope of this work. On the other hand, in line with ideas of [18], is it possible to achieve an impossibility result for F^3 restricted to the xor-only design (where $f_1, \ldots, f_4$ only xor their parameters)?

Acknowledgments. This work has been funded in part by the IAP Program P6/26 BCRYPT of the Belgian State (Belgian Science Policy), in part by the European Commission through the ICT program under contract ICT-2007-216676 ECRYPT II, and in part by the Research Council K.U.Leuven: GOA TENSE. The author is supported by a Ph.D. Fellowship from the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen). The author would like to thank Elena Andreeva and the anonymous ASIACRYPT 2012 reviewers for their valuable help and feedback.

References

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

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

[3] Bos, J., Özen, O., Stam, M.: Efficient hashing using the AES instruction set. In: Cryptographic Hardware and Embedded Systems - CHES 2011. Lecture Notes in Computer Science, vol. 6917, pp. 507–522. Springer, Heidelberg (2011)

[4] 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)

[5] 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)

[6] 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)

[7] 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)

[8] 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)

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

[10] 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)

[11] 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 [12]

[12] 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)

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

[14] 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)

[15] 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)

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

[17] Mennink, B.: On the collision and preimage security of MDC-4 in the ideal cipher model. Cryptology ePrint Archive, Report 2012/113 (2012)

[18] 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)

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

[20] 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)
[21] 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)
[22] 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)
[23] 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)
[24] 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)
[25] 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)
[26] 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)
[27] Stam, M.: Beyond uniformity: Better security/efficiency tradeoffs for compression functions. In: Advances in Cryptology - CRYPTO 2008. Lecture Notes in Computer Science, vol. 5157, pp. 397–412. Springer, Heidelberg (2008)
[28] Stam, M.: Blockcipher-based hashing revisited. In: Fast Software Encryption 2009. Lecture Notes in Computer Science, vol. 5665, pp. 67–83. Springer, Heidelberg (2009)
[29] Steinberger, J.: The collision intractability of MDC-2 in the ideal-cipher model. In: Advances in Cryptology - EUROCRYPT 2007. Lecture Notes in Computer Science, vol. 4515, pp. 34–51. Springer, Heidelberg (2007)