On the Collision and Preimage Security of MDC-4 in the Ideal Cipher Model

Abstract. We present a collision and preimage security analysis of MDC-4, a 24 years old construction for transforming an n-bit block cipher into a $2n$-bit hash function. We start with the MDC-4 compression function based on two independent block ciphers, and prove that any adversary with query access to the underlying block ciphers requires at least $2^{5n/8}$ queries (asymptotically) to find a collision, and at least $2^{5n/4}$ queries to find a preimage. These results then directly carry over to the MDC-4 hash function design. With these results, we are the first to formally confirm that MDC-4 offers a higher level of provable security compared to MDC-2. Next, we consider MDC-4 based on one single block cipher, and confirm that the collision bound carries over to the single block cipher setting. In case of preimage resistance we present a surprising negative result: for a target image with the same left and right half, a MDC-4 preimage in the single block cipher setting can be found in approximately 2^n queries. Yet, restricted to target images with different left and right halves, the preimage security bound of $2^{5n/4}$ queries is nevertheless retained.
Keywords. MDC-4; double block length; hash function; collision resistance; preimage resistance.

1 Introduction

The focus of this work is the classical block cipher based hash function MDC-4. MDC-4 and its related hash function MDC-2 have first been described by Meyer and Schilling in 1988 [20], and have been patented by Brachtl et al. in 1990 [4]. MDC-2 has been standardized in ISO/IEC 10118-2 [10] and is used in numerous applications (see [11, 23] for an exposition). In their original specification, MDC-2 and MDC-4 are instantiated using the block cipher DES. In this work, we step away from this design criterion and consider the designs based on any block cipher $E : \mathbb{Z}_2^n \times \mathbb{Z}_2^n \to \mathbb{Z}_2^n$ with key and message length n bits (throughout, the first input to the block cipher is the key input).

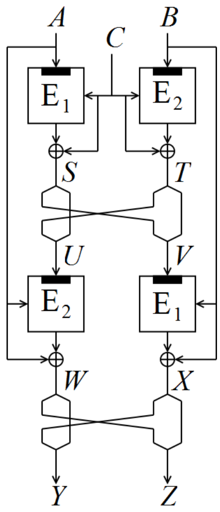
MDC-2 is a Merkle-Damgård (MD) hash function design [5, 19] using a compression function $f_{\text{MDC-2}} : \mathbb{Z}_2^{3n} \to \mathbb{Z}_2^{2n}$ that internally calls the block cipher E twice. It additionally employs two mappings v and w, applied on the key inputs to the two block cipher calls. As v and w are originally constructed so as to have a distinct range[1], we can consider MDC-2 to be based on two block ciphers $E_1(\cdot,\cdot) = E(v(\cdot),\cdot)$ and $E_2(\cdot,\cdot) = E(w(\cdot),\cdot)$ [23]. The compression function $f_{\text{MDC-2}} : \mathbb{Z}_2^{3n} \to \mathbb{Z}_2^{2n}$ is defined as follows.

$$\begin{aligned}
&f_{\text{MDC-2}}(A, B, C)\\
&\quad W \leftarrow E_1(A, C) \oplus C, \quad X \leftarrow E_2(B, C) \oplus C,\\
&\quad Y \leftarrow W^l \| X^r, \qquad\qquad Z \leftarrow X^l \| W^r,\\
&\quad \textbf{return } (Y, Z).
\end{aligned}$$

Here, for a bit string X of even length we denote by X^l and X^r its left and right halves. $f_{\text{MDC-2}}$ can be considered as a parallel evaluation of two Matyas-Meyer-Oseas (MMO) constructions [18] followed by a swapping of the right halves of both states. Consequently, $f_{\text{MDC-2}}$ does not achieve the desired level of security: finding a collision or a preimage for $f_{\text{MDC-2}}$ is as hard as finding it for the two MMO constructions independently, hence requires about $2^{n/2}$ or 2^n block cipher calls, respectively. For the full MDC-2 hash function, Knudsen et al. [11] demonstrated that roughly $2^n/n$ block cipher calls suffice for finding a collision, and about 2^n calls for finding a preimage. In 2007, Steinberger derived a non-trivial security lower bound on MDC-2 [23]. Steinberger considers the MDC-2 hash function using one single block cipher E modeled as a random cipher, and proves that any adversary with query access to E requires at least $2^{3n/5}$ queries (asymptotically) to find a collision. His proof relies on the observation that a collision for MDC-2 implies a collision for the last two rounds of MDC-2, except for some special cases. These results on MDC-2 are summarized in Table 1; all of these results hold for the case E_1 and E_2 are different block ciphers as for the case they are the same.

The MDC-4 hash function differs from MDC-2 in the sense that its underlying compression function $f_{\text{MDC-4}}$ makes four calls to the underlying block cipher rather than two. $f_{\text{MDC-4}}$ is defined as two consecutive evaluations of $f_{\text{MDC-2}}$, where the message inputs to the MMO executions in the second

[1] The original specification (using block cipher DES) defines v and w as mappings from the ciphertext space to the key space, where the parity bits are omitted, and the second and third bits are set to 10 and 01, respectively.



$f_{\text{MDC-4}}(A, B, C)$
$S \leftarrow E_1(A, C) \oplus C,$
$T \leftarrow E_2(B, C) \oplus C,$
$U \leftarrow T^l \| S^r,$
$V \leftarrow S^l \| T^r,$
$W \leftarrow E_2(U, A) \oplus A,$
$X \leftarrow E_1(V, B) \oplus B,$
$Y \leftarrow X^l \| W^r,$
$Z \leftarrow W^l \| X^r,$
return (Y, Z).

Fig. 1. The $f_{\text{MDC-4}}$ compression function. For convenience, we swapped the left and right block ciphers of the second $f_{\text{MDC-2}}$ evaluation.

evaluation are B for E_1 and A for E_2. The definition of $f_{\text{MDC-4}} : \mathbb{Z}_2^{3n} \to \mathbb{Z}_2^{2n}$ is given in Fig. 1. Knudsen and Preneel [12] showed that approximately $2^{3n/4}$ block cipher executions suffice to find a collision for $f_{\text{MDC-4}}$. With respect to preimage resistance, the same authors report that $2^{3n/2}$ calls suffice for finding a preimage for $f_{\text{MDC-4}}$ and $2^{7n/4}$ calls result in a preimage for MDC-4. These results are summarized in Table 1. In independent concurrent research, Fleischmann et al. [6] analyzed the collision resistance of MDC-4. They proved that MDC-4 is collision secure up to approximately $2^{3n/5}$ queries. Hence, they prove that the bound of Steinberger for MDC-2 also holds for MDC-4 (in fact, numerically their bound is slightly worse than the bound on MDC-2). Since the introduction of the functions in [4, 20], however, the MDC-4 hash function has always been considered the more secure variant of MDC-2. Although mainly a matter of theoretical interest (given that MDC-2 is twice as fast as MDC-4), formal security lower bounds for MDC-4 or $f_{\text{MDC-4}}$ that confirm this longstanding claim have never been derived. In particular, for years the MDC-4 structure has not been thoroughly analyzed, which makes it impossible to classify MDC-4 among other double block length constructions known in literature (see "Related Work").

Our Contributions

In this work, we formally analyze the collision *and* (everywhere) preimage security of MDC-4 and its underlying $f_{\text{MDC-4}}$ compression function of Fig. 1. We start with the original double block cipher setting, where $f_{\text{MDC-4}}$ is built on two block ciphers E_1 and E_2. Then, we generalize our findings to the single block cipher setting where $E_1 = E_2$.

Double block cipher setting. In the setting of two independent block ciphers, our starting point is the collision and (everywhere) preimage security of the $f_{\text{MDC-4}}$ compression function. At the first glance, as $f_{\text{MDC-4}}$ roughly consists of two evaluations of $f_{\text{MDC-2}}$, one is inclined to say the results of Steinberger [23] directly carry over. However, this is not true due to the differences at the second $f_{\text{MDC-2}}$ evaluation where the inputs are swapped and the message inputs differ for the left and right cipher. Instead, we conduct a new collision resistance proof for $f_{\text{MDC-4}}$, and although it shows some similarities with the proof of Steinberger, it differs in many aspects and uses several new ideas to facilitate the analysis. We formally prove that any adversary with query access to E_1 and E_2, which are modeled as independent random ciphers, needs at least $2^{5n/8}$ queries (asymptotically) to find a collision for $f_{\text{MDC-4}}$. This is significantly beyond the collision bound $2^{3n/5}$ on MDC-2 by Steinberger [23] and on MDC-4 concurrently derived by Fleischmann et al. [6].

For the preimage resistance of $f_{\text{MDC-4}}$ we prove that at least $2^{5n/4}$ queries (asymptotically) are required, hence security beyond the birthday bound is achieved. In order to achieve security beyond 2^n queries, we employ the ideas of free queries and wish lists. These proof tools have been used before by

Table 1. Known security results for the MDC-2 and MDC-4 compression and hash functions. The security bound gives a lower bound and the attack bound gives an upper bound on the number of queries in order to find an attack. By "(triv)" we note that the bound is trivial; these bounds come from the security of the MMO construction [3], or correspond to generic attacks. The results printed in **bold** are derived in this work.

	collision		preimage		ideal primitives
	security	attack	security	attack	
$f_{\text{MDC-2}}$	$2^{n/2}$ (triv)	$2^{n/2}$ (triv)	2^n (triv)	2^n (triv)	E or (E_1, E_2)
MDC-2	$2^{3n/5}$ [23]	$2^n/n$ [11]	2^n (triv)	2^n [11]	
$f_{\text{MDC-4}}$	$\mathbf{2^{5n/8}}$	$2^{3n/4}$ [12]	$\mathbf{2^{5n/4}}$	$2^{3n/2}$ [12]	(E_1, E_2)
MDC-4	$\mathbf{2^{5n/8}}$	2^n (triv)	$\mathbf{2^{5n/4}}$	$2^{7n/4}$ [12]	
$f_{\text{MDC-4}}$	$2^{n/2}$ (triv)	$2^{n/2}$ (triv)	2^n (triv)	2^n (triv)	E
MDC-4	$\mathbf{2^{5n/8}}$	2^n (triv)	2^n (triv)	$\mathbf{2^n}$	

Armknecht et al. and Lee et al. [2, 17] for compression functions based on two block cipher calls (see "Related Work"), but because MDC-4 makes four block cipher calls rather than two, and additionally the corresponding wish lists are much harder to bound, the security proof has become considerably more complex.

Because MDC-4 is a MD transform, it preserves collision and (everywhere) preimage resistance [1], which means that if the compression function satisfies these security notions, then so does the hash function. Therefore, the results for $f_{\text{MDC-4}}$ directly carry over to the MDC-4 hash function. We have included these findings in Table 1.

Single block cipher setting. In this work we also consider the generalized MDC-4 design where the two block ciphers are the same, $E = E_1 = E_2$. Note that in this setting collisions and preimages for $f_{\text{MDC-4}}$ can be found in at most $2^{n/2}$ and 2^n block cipher calls, respectively: one focuses on state values with the same left and right half (for Fig. 1 this means $A = B$ and $Y = Z$, and consequently $S = T$, $U = V$, and $W = X$), and the security boils down to the security of two subsequent MMO evaluations. For the preimage resistance, these type of target images (with $Y = Z$) can be considered as weak images, these give the adversary significantly more power. In any case, the security preservation approach does not help us out here, and instead we consider the security of the MDC-4 hash function design directly.

Starting with the collision resistance, we prove that the above-mentioned bound of $2^{5n/8}$ queries still holds for the MDC-4 hash function with $E_1 = E_2$. The proof combines previous result and the fact that in a MDC-4 evaluation the state values always consist of two different halves except with a sufficiently small probability.

For (everywhere) preimage resistance, we demonstrate that the bound of $2^{5n/4}$ queries *does not* carry over. In more detail, we show that when the target image satisfies $Y = Z$, a preimage for MDC-4 can be found in approximately 2^n queries. The attack resembles ideas from the preimage attack on MDC-2 by Knudsen et al. [11] and from above-described preimage attack on $f_{\text{MDC-4}}$ in the single block cipher setting. We stress that the attack does not apply to the original MDC-4 design due to the domain separation by the functions v and w. Also, restricted to target images with different halves ($Y \neq Z$) we prove that the bound of $2^{5n/4}$ queries still applies, to both $f_{\text{MDC-4}}$ as MDC-4. We remark that target images with the same left and right half are rather rare, 2^n out of 2^{2n} target images satisfy this property. If we had opted for preimage resistance where the challenge is randomly generated, we obtain in the single block cipher setting a security bound of approximately $2^{5n/4}$ queries for $f_{\text{MDC-4}}$ and MDC-4, rather than the bound of 2^n queries.

The findings on MDC-4 based on one block cipher E are included in Table 1.

With the $2^{5n/8}$ collision and $2^{5n/4}$ preimage security bounds we have formally confirmed the widespread believe that MDC-4 offers a higher level of security compared to MDC-2. Despite that this security gain is obtained at the price of efficiency loss, it is an interesting and important result that allows us to make a fairer comparison among the double block length hash functions and that gives us more insight in the possibilities and impossibilities of block cipher based hashing. In particular, to our knowledge this is the first time the preimage resistance of a double block length compression function design with more than two block cipher evaluations is analyzed. Given that MDC-4 is originally constructed from an efficiency rather than a security point of view, more elaborate designs with more than two block cipher calls may likely offer a higher level of security; our work is a good starting point for this research direction. Although our findings improve the existing bounds on MDC-4 significantly, large gaps between the security bounds and the best known attacks remain. A more technical and elaborate analysis may result in better bounds, and it remains an interesting open problem to improve the security bounds or the generic attacks for MDC-2 and MDC-4.

Related Work

Closely related to this work are the classical double block length compression functions Abreast-DM and Tandem-DM [13] and Hirose's compression function [9], as well as the general compression function designs by Hirose [8] and Özen and Stam [21]. And in fact, these constructions all beat MDC-2 and MDC-4 with respect to collision and preimage resistance. Each of these constructions is provided with almost optimal collision security (see Fleischmann et al. [7] and Lee and Kwon [14] for Abreast-DM, see Lee et al. [16] for Tandem-DM). With respect to preimage resistance, Armknecht et al. and Lee et al. [2, 17] prove security of Abreast-DM, Tandem-DM and Hirose's compression function up to almost 2^{2n} queries (as mentioned, our preimage resistance proof of MDC-4 employs some proof ideas from [2, 17]). These double block length constructions, however, all fundamentally differ from MDC-2 and MDC-4 in the sense that their underlying block ciphers have a double key size, i.e. they use a block cipher $E : \mathbb{Z}_2^{2n} \times \mathbb{Z}_2^n \to \mathbb{Z}_2^n$ that clearly allows for higher compression and that renders a much stronger underlying assumption.

Outline

In Sect. 2, we introduce some mathematical background and the security model used in this work. The security result on the collision resistance of $f_{\text{MDC-4}}$ (based on two block ciphers E_1, E_2) is given in Sect. 3, along with the formal security proof in Sect. 3.1. In Sect. 4, we present our security result on the preimage resistance of $f_{\text{MDC-4}}$ (based on E_1, E_2), the proof of which is given in Sect. 4.1. In Sect. 5 the results are then generalized to the setting where the two block ciphers are the same.

2 Preliminaries

For $n \in \mathbb{N}$, by $\mathbb{Z}_2^n$ we denote the set of bit strings of length n. For two bit strings X, Y, by $X \| Y$ we denote their concatenation and by $X \oplus Y$ their bitwise XOR. If X is of even length, we denote by X^l and X^r its left and right halves. Throughout, we assume n is even. We denote by $\text{Bloc}(n)$ the set of all block ciphers $E : \mathbb{Z}_2^n \times \mathbb{Z}_2^n \to \mathbb{Z}_2^n$, where the first input corresponds to the key input.

An adversary $\mathcal{A}$ is a probabilistic algorithm with oracle access to two block ciphers $E_1, E_2 \xleftarrow{\$} \text{Bloc}(n)$ randomly sampled from $\text{Bloc}(n)$. We consider the adversary to be information-theoretic, which means that it has unbounded computational power, and that its complexity is measured by the number of queries made to its oracles. The adversary can make forward and inverse queries to E_1 and E_2. The queries are stored in a query history $\mathcal{Q}$ as elements (k_i, K_i, x_i, y_i), where i is the index of the query, $k_i \in \{1, 2\}$ indicates the corresponding block cipher, K_i is the key input, and x_i and y_i denote the (plaintext) input and (ciphertext) output of the block cipher. By $x_i \oplus y_i$, we define its XOR-output. The index i and the parameter k are omitted if they are irrelevant or clear from the context. For $q \geq 0$, by $\mathcal{Q}_q$ we define the query history after q queries. We assume that the adversary never makes queries to which it knows the answer in advance. In this work, we consider two types of adversaries, namely one that aims at finding collisions and one that aims at finding preimages for $f_{\text{MDC-4}}$.

We say that adversary $\mathcal{A}$ finds a collision for $f_{\text{MDC-4}}$ if it obtains a query history $\mathcal{Q}$ that allows it to output two distinct tuples $(A_1, B_1, C_1), (A_2, B_2, C_2) \in \mathbb{Z}_2^{3n}$ such that $f_{\text{MDC-4}}(A_1, B_1, C_1) =$

$f_{\text{MDC-4}}(A_2, B_2, C_2)$ and for which $\mathcal{Q}$ contains all block cipher queries required to compute the two evaluations of $f_{\text{MDC-4}}$. We define by

$$\mathbf{adv}^{\text{col}}_{f_{\text{MDC-4}}}(\mathcal{A}) = \mathbf{Pr}\Big(E_1, E_2 \stackrel{\$}{\leftarrow} \text{Bloc}(n),\ (A_1, B_1, C_1), (A_2, B_2, C_2) \leftarrow \mathcal{A}^{E_i, E_i^{-1}}\ :\ (A_1, B_1, C_1) \neq (A_2, B_2, C_2),\ f_{\text{MDC-4}}(A_1, B_1, C_1) = f_{\text{MDC-4}}(A_2, B_2, C_2)\Big)$$

the probability that $\mathcal{A}$ succeeds in finding such query history, and define by $\mathbf{adv}^{\text{col}}_{f_{\text{MDC-4}}}(q)$ the maximum collision advantage taken over all adversaries making q queries.

With respect to preimage resistance, we opt for the notion of everywhere preimage resistance [22]. This security notion intuitively guarantees preimage security for every range point. Before making queries to its oracles, the (preimage finding) adversary $\mathcal{A}$ decides on a range point $(Y, Z) \in \mathbb{Z}_2^{2n}$. We say that $\mathcal{A}$ finds a preimage for $f_{\text{MDC-4}}$ if it obtains a query history $\mathcal{Q}$ that allows it to output a tuple $(A, B, C) \in \mathbb{Z}_2^{3n}$ such that $f_{\text{MDC-4}}(A, B, C) = (Y, Z)$ and for which $\mathcal{Q}$ contains all block cipher queries required to compute the evaluation of $f_{\text{MDC-4}}$. We define by

$$\mathbf{adv}^{\text{epre}}_{f_{\text{MDC-4}}}(\mathcal{A}) = \max_{(Y,Z) \in \mathbb{Z}_2^{2n}} \mathbf{Pr}\Big(E_1, E_2 \stackrel{\$}{\leftarrow} \text{Bloc}(n),\ (A, B, C) \leftarrow \mathcal{A}^{E_i, E_i^{-1}}(Y, Z)\ :\ (Y, Z) = f_{\text{MDC-4}}(A, B, C)\Big)$$

the maximum probability that $\mathcal{A}$ succeeds in finding such query history, and define by $\mathbf{adv}^{\text{epre}}_{f_{\text{MDC-4}}}(q)$ the maximum (everywhere) preimage advantage taken over all adversaries making q queries.

The security definitions for the full MDC-4 hash function, $\mathbf{adv}^{\text{col}}_{\text{MDC-4}}$ and $\mathbf{adv}^{\text{epre}}_{\text{MDC-4}}$, are defined similarly. Here, rather than tuples from $\mathbb{Z}_2^{3n}$ the adversary outputs messages of arbitrary length. Throughout, we denote the initial state value of MDC-4 by (F_0, G_0). In the single block cipher setting, we consider one block cipher E to be generated randomly from $\text{Bloc}(n)$ rather than two, and the definitions follow immediately. In the remainder of this work, it is clear from the context which of the security models we consider.

3 Collision Resistance of MDC-4 with Two Block Ciphers E_1, E_2

We derive a collision resistance lower bound for the $f_{\text{MDC-4}}$ compression function of Fig. 1. The result directly carries over to MDC-4 as it is a MD transform, which preserves collision resistance [1].

Theorem 1. *Let $n \in \mathbb{Z}_2^n$ and $q < 2^{n-1}$. Let $t_1, t_2, t_3 > 0$ be any integral values. Then,*

$$\mathbf{adv}^{\text{col}}_{f_{\text{MDC-4}}}(q) \leq \frac{2(t_1 + 7t_1t_2 + 3t_1t_2t_3 + 5t_2 + 4t_2^2)q}{2^n} + \frac{2q^2}{t_1 2^n} + 2 \cdot 2^{n/2}\left(\frac{2eq}{t_2 2^{n/2}}\right)^{t_2} + 2^n\left(\frac{2eq}{t_3 2^n}\right)^{t_3}. \quad (1)$$

The proof of Thm. 1 is given in Sect. 3.1. It shows similarities with the proof by Steinberger [23] of collision resistance of the MDC-2 hash function, but its structure is entirely different so as to facilitate the proof.

One can divide the bound of (1) into two parts. The first term forms the first part and increases for increasing parameters t_1, t_2, t_3. The remaining three terms form the second part that decreases for increasing t_1, t_2, t_3. For obtaining a sharp bound, we need to fine tune the integral positive parameters t_1, t_2, t_3. The trick is to take parameters t_1, t_2, t_3 minimal so that the second part of (1) still goes to 0 for $n \to \infty$. We will show that the advantage of any adversary making slightly less than $2^{5n/8}$ queries approaches 0 when n goes to infinity. To this end, let $\varepsilon > 0$ be any parameter, we consider any adversary making at most $q = 2^{5n/8}/n^{\varepsilon}$ queries to its oracle. We set $t_1 = 2^{2n/8}$, $t_2 = 2^{n/8}$, and $t_3 = 3$. Here, for simplicity we assume t_1, t_2 to be integral. If n is no multiple of 8, one sets t_1, t_2 to the nearest integers. Note that these parameters satisfy $t_1 > \frac{q^2}{2^n}$, $t_2 > \frac{q}{2^{n/2}}$ and $t_3 > \frac{q}{2^n}$, which are minimal requirements for the second part of (1) to be < 1. Now, it is an easy exercise to verify that the bound of (1) approaches 0 for $n \to \infty$ when $q = 2^{5n/8}/n^{\varepsilon}$ and t_1, t_2, t_3 are as specified.

Corollary 1. *For any $\varepsilon > 0$, we obtain* $\lim_{n\to\infty} \mathbf{adv}^{\text{col}}_{f_{\text{MDC-4}}}\left(2^{5n/8}/n^{\varepsilon}\right) = 0$.

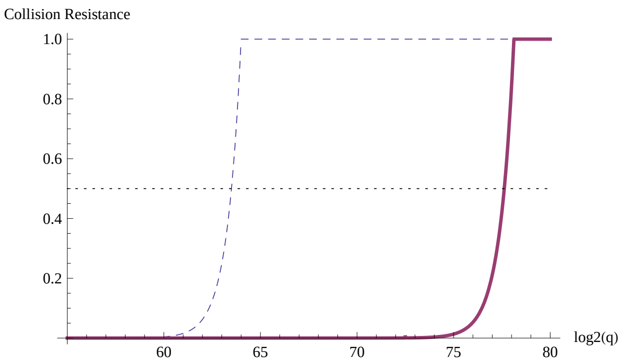


Fig. 2. The function $\mathbf{adv}^{\mathrm{col}}_{f_{\text{MDC-4}}}(q)$ of (1) for $n = 128$, in comparison with the trivial bound $q(q+1)/2^n$ (dashed line).

The result means that for $n \to \infty$ the function $\mathbf{adv}^{\mathrm{col}}_{f_{\text{MDC-4}}}$ behaves as $q^8/2^{5n}$. A graphical representation of $\mathbf{adv}^{\mathrm{col}}_{f_{\text{MDC-4}}}$ for $n = 128$ is given in Fig. 2. In this graph, where we have slightly adjusted the parameters t_1, t_2 to facilitate the analysis for smaller n and smaller q (the previously chosen values were set to analyze limiting behavior for n, q), we see a significant improvement over the best known bound and the bound independently derived in [6]. For $n = 128$ the collision resistance advantage hits $1/2$ for $\log_2 q \approx 77.6$, which is smaller than the threshold for $q^8/2^{5n}$, 79.9. For larger values of n, by Cor. 1 the difference goes to 0 for $n \to \infty$.

3.1 Proof of Thm. 1

The collision resistance proof shows some similarities with the proof of Steinberger for MDC-2 [23], but fundamentally differs in various aspects and is as such of independent interest. In particular, due to a different and more structured case distinction we obtain a sharper bound (security up to $2^{5n/8}$ queries) than the bound of Steinberger for MDC-2 (security up to $2^{3n/5}$ queries). Also, our proof significantly improves over the proof by Fleischmann et al. [6], who simply confirmed that the $2^{3n/5}$ bound of MDC-2 applies to MDC-4 too.

We consider any adversary making q queries to its oracles E_1, E_2, which tries to find a collision for $f_{\text{MDC-4}}$. Finding a collision corresponds to obtaining a query history $\mathcal{Q}_q$ of size q that satisfies configuration $\mathsf{col}(\mathcal{Q}_q)$ of Fig. 3. In other words,

$$\mathbf{adv}^{\mathrm{col}}_{f_{\text{MDC-4}}}(q) = \mathsf{Pr}\left(\mathsf{col}(\mathcal{Q}_q)\right), \tag{2}$$

and we consider the probability of obtaining any query history $\mathcal{Q}_q$ that satisfies configuration $\mathsf{col}(\mathcal{Q}_q)$. Notice that in this configuration, we omit the shifting at the end: as this shifting is bijective, it does not influence the collision finding advantage. In Fig. 3, as well as in all subsequent figures in this section, we label the block ciphers as follows to uniquely identify their positions. In the left word of Fig. 3 (with inputs (A_1, B_1, C_1)) the block ciphers are labeled 1tl, 1tr, 1bl, 1br, for top/bottom left/right. For the right word the block ciphers are identified as 2tl, 2tr, 2bl, 2br. In the remainder, when talking about "a query 1tl", we mean "a query that in a collision occurs at position 1tl" (and the same for the other positions). The labels "1" and "2" in the block ciphers correspond to the block cipher index. The capitalized variables in the figures may take any value, and are simply used to accentuate relations among the two words.

We need to evaluate the probability of the adversary finding a query history $\mathcal{Q}_q$ that satisfies configuration $\mathsf{col}(\mathcal{Q}_q)$ of Fig. 3. For this analysis we introduce a helping event $\mathsf{help}(\mathcal{Q}_q)$. Let $t_1, t_2, t_3 > 0$ be integral. Event $\mathsf{help}(\mathcal{Q}_q)$ is satisfied if either of the following sub-events $\mathsf{help}_k(\mathcal{Q}_q)$ $(k = 1, \ldots, 4)$ occurs.

$$\begin{aligned}
&\mathsf{help}_1(\mathcal{Q}_q): && \left|\left\{(K_i, x_i, y_i), (K_j, x_j, y_j) \in \mathcal{Q}_q \;\middle|\; i \neq j \;\wedge\; x_i \oplus y_i = x_j \oplus y_j\right\}\right| > t_1;\\
&\mathsf{help}_2(\mathcal{Q}_q): && \text{for some } z \in \mathbb{Z}_2^{n/2}: \;\left|\left\{(K_i, x_i, y_i) \in \mathcal{Q}_q \;\middle|\; (x_i \oplus y_i)^l = z\right\}\right| > t_2;\\
&\mathsf{help}_3(\mathcal{Q}_q): && \text{for some } z \in \mathbb{Z}_2^{n/2}: \;\left|\left\{(K_i, x_i, y_i) \in \mathcal{Q}_q \;\middle|\; (x_i \oplus y_i)^r = z\right\}\right| > t_2;\\
&\mathsf{help}_4(\mathcal{Q}_q): && \text{for some } z \in \mathbb{Z}_2^{n}: \;\left|\left\{(K_i, x_i, y_i) \in \mathcal{Q}_q \;\middle|\; x_i \oplus y_i = z\right\}\right| > t_3.
\end{aligned}$$

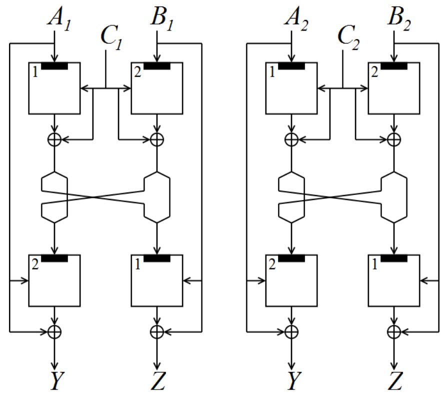


Fig. 3. Configuration $\mathsf{col}(\mathcal{Q})$. We require $(A_1, B_1, C_1) \neq (A_2, B_2, C_2)$.

Note that we do not distinguish between queries to E_1 or E_2. This is done for simplicity of notation, it has no affect on the security analysis. By basic probability theory, we obtain for (2):

$$\mathsf{Pr}\left(\mathsf{col}(\mathcal{Q}_q)\right) \leq \mathsf{Pr}\left(\mathsf{col}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) + \mathsf{Pr}\left(\mathsf{help}(\mathcal{Q}_q)\right). \tag{3}$$

For the analysis of the event $\mathsf{col}(\mathcal{Q}_q)$, it may be the case that a single query occurs at multiple positions in the configuration. Therefore, we divide $\mathsf{col}(\mathcal{Q}_q)$ into sub-configurations. For two distinct positions $a, b \in \{1\mathsf{tl}, 1\mathsf{tr}, 1\mathsf{bl}, 1\mathsf{br}, 2\mathsf{tl}, 2\mathsf{tr}, 2\mathsf{bl}, 2\mathsf{br}\}$ and a binary value $\alpha \in \{0, 1\}$, by $a = b \equiv \alpha$ we say that the same query occurs at both positions a and b if and only if $\alpha = 1$. Now, we define for $\alpha_{\mathsf{tl}}, \alpha_{\mathsf{tr}}, \alpha_{\mathsf{bl}}, \alpha_{\mathsf{br}} \in \{0, 1\}$ the sub-configuration $\mathsf{col}_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q})$ as $\mathsf{col}(\mathcal{Q})$ of Fig. 3 with the restriction that

$$1\mathsf{tl} = 2\mathsf{tl} \equiv \alpha_{\mathsf{tl}}, \qquad 1\mathsf{tr} = 2\mathsf{tr} \equiv \alpha_{\mathsf{tr}}, \qquad 1\mathsf{bl} = 2\mathsf{bl} \equiv \alpha_{\mathsf{bl}}, \qquad 1\mathsf{br} = 2\mathsf{br} \equiv \alpha_{\mathsf{br}}.$$

Clearly,

$$\mathsf{col}(\mathcal{Q}_q) \Rightarrow \bigvee_{\substack{\alpha_{\mathsf{tl}},\alpha_{\mathsf{tr}},\alpha_{\mathsf{bl}},\\ \alpha_{\mathsf{br}}\in\{0,1\}}} \mathsf{col}_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q}_q). \tag{4}$$

It may be the case that the same query occurs at positions $1\mathsf{tl}$ and $1\mathsf{br}$ or $2\mathsf{br}$, but as becomes clear these cases are included in the analysis. By (2-4), we obtain the following bound on $\mathbf{adv}^{\mathrm{col}}_{f_{\text{MDC-4}}}(q)$:

$$\mathbf{adv}^{\mathrm{col}}_{f_{\text{MDC-4}}}(q) \leq \sum_{\substack{\alpha_{\mathsf{tl}},\alpha_{\mathsf{tr}},\alpha_{\mathsf{bl}},\\ \alpha_{\mathsf{br}}\in\{0,1\}}} \mathsf{Pr}\left(\mathsf{col}_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) + \mathsf{Pr}\left(\mathsf{help}(\mathcal{Q}_q)\right). \tag{5}$$

The probabilities constituting to the sum of (5) are analyzed in Lems. 1-6 as further set forth in Table 2. Probability $\mathsf{Pr}\left(\mathsf{help}(\mathcal{Q}_q)\right)$ is analyzed in Lem. 7. In this section we only include the proof of Lems. 1. The proofs of Lems. 2-7 are given in App. A.

Table 2. For $\alpha_{\mathsf{tl}}, \alpha_{\mathsf{tr}}, \alpha_{\mathsf{bl}}, \alpha_{\mathsf{br}} \in \{0, 1\}$, the probability bound on $\mathsf{col}_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)$ (cf. (5)) is analyzed in the corresponding lemma.

$\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}$	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Lemma	1	2	2	3	4	5	6	6	4	6	5	6	6	6	6	6

Lemma 1. $\mathsf{Pr}\left(\mathsf{col}_{0000}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) \leq \frac{(t_1t_2t_3+t_1t_2+t_2)q}{2^n-q}$.

Proof. A visualization of configuration $\mathsf{col}_{0000}(\mathcal{Q}_q)$ can be found in Fig. 4. In this figure, the queries corresponding to locations a and $!a$ are required to be different, and the same for the queries at positions $(b, !b)$, $(c, !c)$ and $(d, !d)$. For the analysis of $\mathsf{col}_{0000}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)$, we say that the i-th query ($i \in$

$\{1,\ldots,q\}$) is successful if it makes configuration $\mathsf{col}_{0000}(\mathcal{Q}_i)$ satisfied and $\neg\mathsf{help}(\mathcal{Q}_i)$ holds. Now, by basic probability theory, we can analyze the probability of the i-th query being successful, and sum over $i=1,\ldots,q$.

Let $i\in\{1,\ldots,q\}$. We will analyze the probability of the i-th query to be successful, i.e. to satisfy $\mathsf{col}_{0000}(\mathcal{Q}_i)\wedge\neg\mathsf{help}(\mathcal{Q}_i)$. If $\mathsf{help}(\mathcal{Q}_i)$ holds, the i-th query can certainly not be successful, so we assume $\neg\mathsf{help}(\mathcal{Q}_i)$ and analyze the probability the i-th query makes $\mathsf{col}_{0000}(\mathcal{Q}_i)$ satisfied.

Without loss of generality (by symmetry), the i-th query is a query to E_1 and occurs at position $1\mathsf{tl}$ and/or $1\mathsf{br}$. We distinguish among these three cases. It may be the case that the i-th query also occurs in the right word, but as becomes clear from the proof, these cases are automatically included.

Query occurs at $1\mathsf{tl}$ but not at $1\mathsf{br}$. By $\neg\mathsf{help}_1(\mathcal{Q}_i)$, there are $\leq t_1$ choices for queries at positions $(1\mathsf{br},2\mathsf{br})$ (we note that the query at $2\mathsf{br}$ may equal the i-th query, but this does not invalidate the ongoing analysis). For any of these $\leq t_1$ choices, let $K_{1\mathsf{br}}$ and $K_{2\mathsf{br}}$ be the key inputs corresponding to positions $1\mathsf{br}$ and $2\mathsf{br}$. By $\neg\mathsf{help}_2(\mathcal{Q}_i)$, there are $\leq t_2$ choices for $2\mathsf{tl}$. For any of these $\leq t_1t_2$ choices $2\mathsf{tl}$, the query at position $2\mathsf{tr}$ and consequently the query at position $2\mathsf{bl}$ is uniquely determined (if they exist at all), and so is the XOR-output Y of $2\mathsf{bl}$. By $\neg\mathsf{help}_4(\mathcal{Q}_i)$, there are $\leq t_3$ choices for $1\mathsf{bl}$. For any of these $\leq t_1t_2t_3$ choices $1\mathsf{bl}$, let $K_{1\mathsf{bl}}$ be the key input corresponding to position $1\mathsf{bl}$. The i-th query is successful only if its XOR-output equals $K^l_{1\mathsf{br}}\|K^r_{1\mathsf{bl}}$, which happens with probability at most $\frac{1}{2^n-q}$. The total probability is at most $\frac{t_1t_2t_3}{2^n-q}$.

Query occurs at $1\mathsf{br}$ but not at $1\mathsf{tl}$. By $\neg\mathsf{help}_1(\mathcal{Q}_i)$, there are $\leq t_1$ choices for $(1\mathsf{bl},2\mathsf{bl})$. For any of these $\leq t_1$ choices, let $K_{2\mathsf{bl}}$ be the key input corresponding to position $2\mathsf{bl}$. By $\neg\mathsf{help}_2(\mathcal{Q}_i)$, there are $\leq t_2$ choices for $2\mathsf{tr}$. For any of these $\leq t_1t_2$ choices $2\mathsf{tr}$, the query at position $2\mathsf{tl}$ and consequently the query at position $2\mathsf{br}$ is uniquely determined, and so is the XOR-output Z of $2\mathsf{br}$. The i-th query is successful only if its XOR-output equals this value Z, which happens with probability at most $\frac{1}{2^n-q}$. The total success probability is at most $\frac{t_1t_2}{2^n-q}$.

Query occurs at $1\mathsf{tl}$ and $1\mathsf{br}$. Let K be the key input for the i-th query. As the query occurs at both positions, we require $K^l=Z^l$, which fixes Z^l. By $\neg\mathsf{help}_2(\mathcal{Q}_i)$, there are $\leq t_2$ choices for $2\mathsf{br}$. For any of these $\leq t_2$ choices $2\mathsf{br}$, we obtain a different Z. The i-th query is successful only if its XOR-output equals this value Z, which happens with probability at most $\frac{1}{2^n-q}$. The total success probability is at most $\frac{t_2}{2^n-q}$.

The i-th query is successful with probability at most $\frac{t_1t_2t_3+t_1t_2+t_2}{2^n-q}$. The claimed bound is obtained by summing over $i=1,\ldots,q$. $\square$

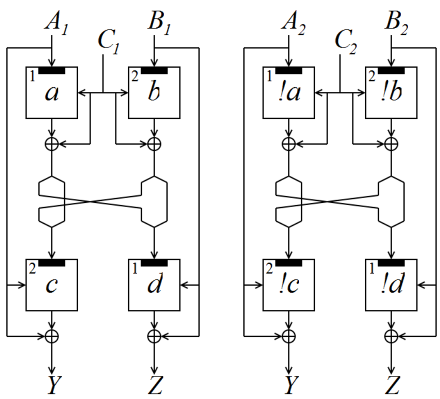


Fig. 4. Configuration $\mathsf{col}_{0000}(\mathcal{Q})$ of Lem. 1. We require $(A_1,B_1,C_1)\neq(A_2,B_2,C_2)$.

Lemma 2. $\Pr\left(\mathsf{col}_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q}_q)\wedge\neg\mathsf{help}(\mathcal{Q}_q)\right)\leq\frac{(t_1t_2+t_2+t_2^2)q}{2^n-q}$ *for* $\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}\in\{0001,0010\}$.

Lemma 3. $\Pr\left(\mathsf{col}_{0011}(\mathcal{Q}_q)\wedge\neg\mathsf{help}(\mathcal{Q}_q)\right)\leq\frac{t_1q}{2^n-q}$.

Lemma 4. $\Pr\left(\mathsf{col}_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) \leq \frac{(t_1t_2t_3+t_1t_2+t_2)q}{2^n-q}$ *for* $\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}} \in \{0100, 1000\}$.

Lemma 5. $\Pr\left(\mathsf{col}_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) \leq \frac{(t_1t_2+t_2^2)q}{2^n-q}$ *for* $\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}} \in \{0101, 1010\}$.

Lemma 6. $\Pr\left(\mathsf{col}_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) = 0$ *for* $\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}} \in \{11{*}{*}, 1{*}{*}1, {*}11{*}\}$.

Lemma 7. $\Pr\left(\mathsf{help}(\mathcal{Q}_q)\right) \leq \frac{q^2}{t_1(2^n-q)} + 2\cdot 2^{n/2}\left(\frac{eq2^{n/2}}{t_2(2^n-q)}\right)^{t_2} + 2^n\left(\frac{eq}{t_3(2^n-q)}\right)^{t_3}$.

We are ready to finish the proof of Thm. 1. Lemmas 1-7 imply for $\mathbf{adv}^{\mathrm{col}}_{f_{\text{MDC-4}}}(q)$ of (5):

$$\mathbf{adv}^{\mathrm{col}}_{f_{\text{MDC-4}}}(q) \leq \frac{(t_1+7t_1t_2+3t_1t_2t_3+5t_2+4t_2^2)q}{2^n-q} + \frac{q^2}{t_1(2^n-q)} + 2\cdot 2^{n/2}\left(\frac{eq2^{n/2}}{t_2(2^n-q)}\right)^{t_2} + 2^n\left(\frac{eq}{t_3(2^n-q)}\right)^{t_3},$$

where $t_1, t_2, t_3 > 0$ are integral. The result of Thm. 1 is obtained by observing that $2^n - q > 2^{n-1}$ for $q < 2^{n-1}$.

4 Preimage Resistance of MDC-4 with Two Block Ciphers E_1, E_2

For the everywhere preimage resistance of the $f_{\text{MDC-4}}$ compression function of Fig. 1, we derive the following results. The findings directly carry over to MDC-4 as it is a MD transform, which preserves everywhere preimage resistance [1].

Theorem 2. *Let* $n \in \mathbb{Z}_2^n$. *Let* $t_1, t_2 > 0$ *be any integral values with* $t_1 \leq q$. *Then,*

$$\mathbf{adv}^{\mathrm{epre}}_{f_{\text{MDC-4}}}(q) \leq \frac{4t_2^3+4t_1t_2}{2^n} + \frac{16t_1t_2}{2^{2n}} + \frac{8}{2^n} + 2\cdot 2^{n/2}\left(\frac{4eq}{t_12^{n/2}}\right)^{t_1/2} + \frac{4q}{2^{n/2}}\left(\frac{8eq}{t_12^{n/2}}\right)^{\frac{t_12^n}{4q}} + 2^n\left(\frac{4eq}{t_22^n}\right)^{t_2/2} + 2q\left(\frac{8eq}{t_22^n}\right)^{\frac{t_22^n}{4q}}. \quad (6)$$

The proof of Thm. 2 is given in Sect. 4.1. It employs ideas of the preimage resistance proof by Armknecht et al. [2] and Lee et al. [15, 17] for double block length compression functions, namely the issuance of free queries and the usage of wish lists. However, the analysis has become considerably more complex because the MDC-4 compression function uses four block ciphers rather than two, and consequently the derivation of bounds on the sizes of the wish lists has become more elaborate.

The bound of (6) can be analyzed in a similar manner as is done in Sect. 3, and we skip the details. Let $\varepsilon > 0$ be any parameter, we consider any adversary making at most $q = 2^{5n/4}/n^{\varepsilon}$ queries to its oracle. We set $t_1 = 2^{3n/4}$ and $t_2 = 2^{n/4}/n^{\varepsilon/2}$. Again, t_1, t_2 are assumed to be integral. Note that for interesting values of ε, we have $t_1 \leq q$ as desired. As before, it immediately follows that the bound of (6) approaches 0 for $n \to \infty$ when $q = 2^{5n/4}/n^{\varepsilon}$ and t_1, t_2 are as specified.

Corollary 2. *For any* $\varepsilon > 0$, *we obtain* $\lim_{n\to\infty} \mathbf{adv}^{\mathrm{epre}}_{f_{\text{MDC-4}}}\left(2^{5n/4}/n^{\varepsilon}\right) = 0$.

The result means that for $n \to \infty$ the function $\mathbf{adv}^{\mathrm{epre}}_{f_{\text{MDC-4}}}$ behaves as $q^4/2^{5n}$. a graphical representation of $\mathbf{adv}^{\mathrm{epre}}_{f_{\text{MDC-4}}}$ for $n = 128$ is given in Fig. 5. As in the case of Sect. 3, we have slightly adjusted the parameters t_1, t_2 to facilitate the analysis for smaller n and smaller q. For $n = 128$ the preimage resistance advantage hits 1/2 for $\log_2 q \approx 151.9$. Also in this case, the gap between this value and threshold for $q^4/2^{5n}$, 159.75, is caused by the choice for small n. By Cor. 2 the difference goes to 0 for $n \to \infty$.

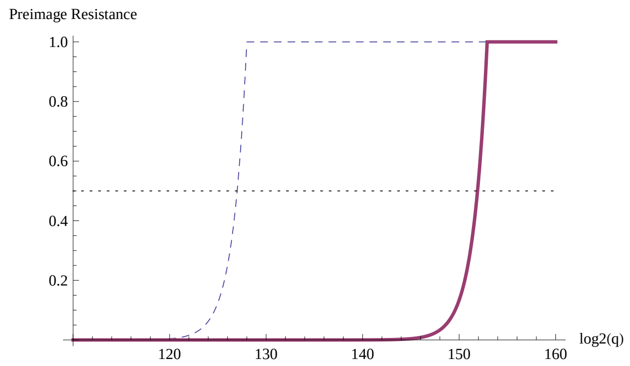


Fig. 5. The function $\mathbf{adv}^{\text{epre}}_{f_{\text{MDC-4}}}(q)$ of (6) for $n = 128$, in comparison with the best known bound $q/2^n$ (dashed line).

4.1 Proof of Thm. 2

We consider any adversary making q queries to its oracles E_1, E_2, which tries to find a preimage for $f_{\text{MDC-4}}$. Let $(Y, Z) \in \mathbb{Z}_2^{2n}$ be the point to invert, chosen by the adversary prior to making any query. Finding a preimage for (Y, Z) corresponds to obtaining a query history $\mathcal{Q}_q$ of size q that satisfies configuration $\mathsf{pre}(\mathcal{Q}_q)$ of Fig. 6. In other words,

$$\mathbf{adv}^{\text{epre}}_{f_{\text{MDC-4}}}(q) = \mathsf{Pr}\left(\mathsf{pre}(\mathcal{Q}_q)\right), \tag{7}$$

and we consider the probability of obtaining any query history $\mathcal{Q}_q$ that satisfies configuration $\mathsf{pre}(\mathcal{Q}_q)$. As is done in Sect. 3.1, we again omit the bijective shifting at the end as it does not influence the preimage security. We use the same convention for the figures as is used in Sect. 3.1, with the difference that in Fig. 6 the variables Y, Z are underlined to denote that these are fixed. As we only consider one word (rather than two, in Sect. 3.1), we label the block ciphers simply as tl, tr, bl, br for top/bottom left/right.

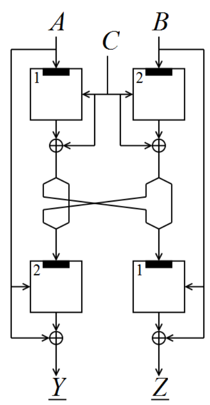


Fig. 6. Configuration $\mathsf{pre}(\mathcal{Q})$.

The analysis in this section relies on the issuance of free super queries [2, 15, 17]. If the adversary has made 2^{n-1} queries to either E_1 or E_2 under the same key, it will receive the remaining 2^{n-1} queries for this key for free. As in [2, 17], we call this query a super query. Formally, these free queries can be modeled as queries the adversary is forced to make, but at no charge. For convenience, we use $\mathcal{Q}_q$ to denote the query history after q normal queries. This query history 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 the adversary needs 2^{n-1} queries as preparatory work to enforce a super query. As the adversary makes at most q queries, at most $q/2^{n-1}$ super queries will occur.

For the analysis of $\mathsf{pre}(\mathcal{Q}_q)$, we introduce a helping event $\mathsf{help}(\mathcal{Q}_q)$. Let $t_1, t_2 > 0$ be integral. Event $\mathsf{help}(\mathcal{Q}_q)$ is satisfied if either of the following sub-events $\mathsf{help}_k(\mathcal{Q}_q)$ $(k = 1, 2, 3)$ occurs.

$$\begin{aligned}
&\mathsf{help}_1(\mathcal{Q}_q): && \text{for some } z \in \mathbb{Z}_2^{n/2}: \ \left|\left\{(K_i, x_i, y_i) \in \mathcal{Q}_q \mid (x_i \oplus y_i)^l = z\right\}\right| > t_1;\\
&\mathsf{help}_2(\mathcal{Q}_q): && \text{for some } z \in \mathbb{Z}_2^{n/2}: \ \left|\left\{(K_i, x_i, y_i) \in \mathcal{Q}_q \mid (x_i \oplus y_i)^r = z\right\}\right| > t_1;\\
&\mathsf{help}_3(\mathcal{Q}_q): && \text{for some } z \in \mathbb{Z}_2^{n}: \ \left|\left\{(K_i, x_i, y_i) \in \mathcal{Q}_q \mid x_i \oplus y_i = z\right\}\right| > t_2.
\end{aligned}$$

These helping events are the same as the ones used in the proof of collision resistance in Sect. 3.1, but are reintroduced for simplicity. Note that $\mathsf{help}_3(\mathcal{Q}_q)$ particularly covers the values Y, Z as XOR-outputs. By basic probability theory, we obtain for (7):

$$\mathsf{Pr}\left(\mathsf{pre}(\mathcal{Q}_q)\right) \leq \mathsf{Pr}\left(\mathsf{pre}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) + \mathsf{Pr}\left(\mathsf{help}(\mathcal{Q}_q)\right). \tag{8}$$

In Lem. 8, we bound $\mathsf{Pr}\left(\mathsf{pre}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right)$ and probability $\mathsf{Pr}\left(\mathsf{help}(\mathcal{Q}_q)\right)$ is analyzed in Lem. 9. The proofs are given in App. B.

Lemma 8. $\mathsf{Pr}\left(\mathsf{pre}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) \leq \frac{4t_2^3 + 4t_1t_2}{2^n} + \frac{16t_1t_2}{2^{2n}} + \frac{8}{2^n}$.

Lemma 9. *Provided $t_1 \leq q$, we have*

$$\mathsf{Pr}\left(\mathsf{help}(\mathcal{Q}_q)\right) \leq 2 \cdot 2^{n/2}\left(\frac{4eq}{t_1 2^{n/2}}\right)^{t_1/2} + \frac{4q}{2^{n/2}}\left(\frac{8eq}{t_1 2^{n/2}}\right)^{\frac{t_1 2^n}{4q}} + 2^n\left(\frac{4eq}{t_2 2^n}\right)^{t_2/2} + 2q\left(\frac{8eq}{t_2 2^n}\right)^{\frac{t_2 2^n}{4q}}.$$

With respect to Lem. 9, we note that $\mathsf{help}_3(\mathcal{Q})$ is similar to the event $\mathsf{Lucky}(\mathcal{Q})$ analyzed by Armknecht et al. [2] and Lee et al. [17]: the only difference is that $\mathsf{help}_3(\mathcal{Q})$ is required to hold for any $z \in \mathbb{Z}_2^n$. In their analysis of $\mathsf{Lucky}(\mathcal{Q})$, [2, 17] make a distinction between the normal and super queries (just as we do in the proof of Lem. 9) but for the super queries their analysis is based on Markov's inequality and is consequently much simpler. However, because our helping events are required to hold for any $z \in \mathbb{Z}_2^n$ (event $\mathsf{help}_3(\mathcal{Q})$) and for any $z \in \mathbb{Z}_2^{n/2}$ (event $\mathsf{help}_{1\vee 2}(\mathcal{Q})$), a similar approach using Markov's inequality would result in a trivial bound and a more elaborate treatment was required.

The proof of Thm. 2 is finished by adding the bounds of Lems. 8-9, as set forth in (7-8).

5 Security of MDC-4 with One Block Cipher E

In this section we consider the collision and preimage resistance of MDC-4 in the particular setting where the two block ciphers are identical, i.e. $E = E_1 = E_2$. Although MDC-4 with two different block ciphers E_1, E_2 is closer the original design specification, it is of general interest to see what security is achieved when the two block ciphers are the same. In this setting, however, the security preservation approach from Sects. 3 and 4 does not help us out: as is already explained in Sect. 1, in the single block cipher setting the $f_{\text{MDC-4}}$ compression function does not offer a sufficiently high level of security. Therefore, we resort to collision and preimage security of MDC-4 *in the iteration.*

5.1 Collision Resistance

For collision resistance of MDC-4 using a single block cipher E, we obtain the following result.

Theorem 3. *Let $n \in \mathbb{Z}_2^n$ and $q < 2^{n-1}$. Let $t_1, t_2, t_3 > 0$ be any integral values. Then,*

$$\mathbf{adv}^{\text{col}}_{\text{MDC-4}}(q) \leq \frac{2(2t_1 + 11t_1t_2 + 3t_1t_2t_3 + 14t_2 + 5t_2^2)q}{2^n} + \frac{2q^2}{t_1 2^n} + 2 \cdot 2^{n/2}\left(\frac{2eq}{t_2 2^{n/2}}\right)^{t_2} + 2^n\left(\frac{2eq}{t_3 2^n}\right)^{t_3}. \tag{9}$$

The proof of Thm. 3 is similar to the one of Thm. 1 and is given in App. C. Here, we will only give the intuition. Recall Fig. 3 of Sect. 3.1. The proof of Thm. 1 uses the independence of E_1, E_2 essentially to guarantee that a block cipher query can never occur at both positions tl and tr (or at both bl and br) for one of the two words. In the single block cipher setting, this cannot be guaranteed for the $f_{\text{MDC-4}}$ compression function and the collision attack of Sect. 1 relies on this. However, for a full MDC-4 iteration, the initial state value consists of two different halves, and in fact all intermediate state values consist of two different halves, except with some small probability. Using this property, one can guarantee that the state inputs (A_1, B_1) and (A_2, B_2) both consist of two different halves and thus that a query cannot occur at both positions tl and tr (or bl and br). Then, as we demonstrate in the proof, the bound of (1) carries over up to a constant factor and it suffices to consider the probability that the adversary ends up with a state value with two the same halves. This probability results in an additional term $\frac{2(t_1+2t_2+t_2^2)q}{2^n}$ in (9). For the choice of t_1, t_2, t_3 as in Sect. 3, we obtain the same corollary result for $\mathbf{adv}^{\text{col}}_{\text{MDC-4}}(q)$ in the single block cipher setting.

Corollary 3. *For any $\varepsilon > 0$, we obtain $\lim_{n\to\infty} \mathbf{adv}^{\text{col}}_{MDC\text{-}4}\left(2^{5n/8}/n^{\varepsilon}\right) = 0$.*

5.2 Preimage Resistance

For preimage resistance, the trick of Sect. 5.1 does not work: a collision on the state halves (the additional event) happens with a significantly higher probability then what we are aiming for. In fact, for some class of target images this drawback turns out to be crucial and results in a preimage attack in 2^n queries. Let (Y, Z) be the target image. We distinguish between $Y \neq Z$ and $Y = Z$.

$\boldsymbol{Y \neq Z}$**.** We denote by $\mathbf{adv}^{\text{epre}(\neq)}_{f_{\text{MDC-4}}}(q)$ the limitation of $\mathbf{adv}^{\text{epre}}_{f_{\text{MDC-4}}}(q)$ to $Y \neq Z$. Consider Fig. 6 of Sect. 4.1 (and ignore the block cipher indices 1 and 2). If $Y \neq Z$, this necessarily implies that in the last round of the MDC-4 iteration the queries at positions bl and br must be different, and similarly that the queries at positions tl and tr are different. Consequently, the proof of Thm. 2 carries over, except for some additional cases to be analyzed. We obtain the following result for $f_{\text{MDC-4}}$ provided that $Y \neq Z$.

Theorem 4. *Let $n \in \mathbb{Z}_2^n$. Let $t_1, t_2 > 0$ be any integral values with $t_1 \leq q$. Then, provided the image (Y, Z) satisfies $Y \neq Z$,*

$$\mathbf{adv}^{\text{epre}(\neq)}_{f_{\text{MDC-4}}}(q) \leq \frac{4t_2^3 + 4t_1t_2 + 20 + 4\cdot 2^{n/2}}{2^n} + \frac{16t_1t_2 + 24t_1t_2 2^{n/2}}{2^{2n}} + \frac{64q}{2^{3n}} + 2\cdot 2^{n/2}\left(\frac{4eq}{t_1 2^{n/2}}\right)^{t_1/2} + \frac{4q}{2^{n/2}}\left(\frac{8eq}{t_1 2^{n/2}}\right)^{\frac{t_1 2^n}{4q}} + 2^n\left(\frac{4eq}{t_2 2^n}\right)^{t_2/2} + 2q\left(\frac{8eq}{t_2 2^n}\right)^{\frac{t_2 2^n}{4q}}. \quad (10)$$

Thm. 4 is proven in App. D. The findings directly carry over to MDC-4 as it is a MD transform, which preserves everywhere preimage resistance [1]. Note that this bound of Thm. 4 is similar to the bound of Thm. 2, except for an additional term $\frac{12+4\cdot 2^{n/2}}{2^n} + \frac{24t_1t_2 2^{n/2}}{2^{2n}} + \frac{64q}{2^{3n}}$. For the same choice of t_1, t_2 as in Sect. 4, this additional term is of order $O(2^{-n/2})$ for $q < 2^{5n/4}$ and thus of negligible size compared to the other terms. The further analysis is the same, and Cor. 2 still holds in the single block cipher setting when $Y \neq Z$.

$\boldsymbol{Y = Z}$**.** If the two halves of the image are the same, preimage for the compression function $f_{\text{MDC-4}}$ can be found in about 2^n queries (cf. Sect. 1): one focuses on preimages with the same left and right halves $A = B$, in which case it suffices to find A, C such that

$$E(E(A, C) \oplus C, A) \oplus A = Y = Z.$$

We demonstrate that this weakness propagates through the iteration of the MDC-4 hash function, resulting in an everywhere preimage attack for the MDC-4 hash function in 2^n queries (on average). We recall that everywhere preimage resistance is defined as the maximum advantage over all images, thus including the weak images consisting of two identical halves. If we had opted for preimage resistance where the challenge is randomly generated, this preimage attack succeeds only with small probability as 2^n out of 2^{2n} target images are weak.

The attack uses ideas from Knudsen et al. [11] to find preimages for MDC-2. It is a meet-in-the-middle attack and at a high level works as follows. First, one constructs a tree with 2^n leaves with root (Y, Z). The edges in this tree correspond to evaluations of $f_{\text{MDC-4}}$. In the general case, the construction of this tree requires the adversary to find approximately 2^{n+1} preimages, but as turns out for $f_{\text{MDC-4}}$ the workload is significantly lower. Then, starting from the initial value (F_0, G_0), one varies the message input C to hit any of the 2^n leaves. In more detail, the attack works as follows:

1. Fix any $m_0, m_1 \in \mathbb{Z}_2^n$ such that $X \| m_b$ is a correct padding for any $X \in \mathbb{Z}_2^{n \cdot n}$ and $b \in \{0, 1\}$;[2]
2. For $b = 0, 1$ operate as follows. For any $A \in \mathbb{Z}_2^n$ query $V \leftarrow E(A, m_b)$ and $W \leftarrow E(V \oplus m_b, A)$. These queries correspond to the evaluation $f_{\text{MDC-4}}(A, A, m_b) = (W \oplus A, W \oplus A)$. Add the input-output tuple $((A, A); (W \oplus A, W \oplus A))$ to a list L_b;
3. Let (Z, Z) be the target image. On average, this item occurs once in each list L_0, L_1, which results in two $f_{\text{MDC-4}}$ preimages for (Z, Z). It may result in more than two $f_{\text{MDC-4}}$ preimages if (Z, Z) occurs multiple times in one of the lists. The same procedure can be iteratively executed for all resulting preimages, until a tree of approximately 2^n leaves is formed, with from each leave a path of n edges to (Z, Z);[3]
4. Starting from initial value (F_0, G_0), vary m until $f_{\text{MDC-4}}(F_0, G_0, m)$ hits any of the 2^n leaves.

Step 1 requires 2^{n+2} block cipher queries. Step 4 is a brute force attack and requires approximately $4 \cdot (2^{2n}/2^n)$ evaluations of E. In total, this attack requires approximately 2^{n+3} queries. The attack has time and space complexity $O(2^n)$.

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-Verlag, Berlin (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-Verlag, Berlin (2011)

[3] Black, J., Rogaway, P., Shrimpton, T.: Black-box analysis of the block-cipher-based hash-function constructions from PGV. In: Advances in Cryptology - CRYPTO 2002. Lecture Notes in Computer Science, vol. 2442, pp. 320–335. Springer-Verlag, Berlin (2002)

[4] Brachtl, B., Coppersmith, D., Hyden, M., Matyas, S., Meyer, C., Oseas, J., Pilpel, S., Schilling, M.: Data Authentication Using Modification Detection Codes Based on a Public One Way Encryption Function. U.S. Patent Number 4,908,861 (March 13, 1990)

[5] Damgård, I.: A design principle for hash functions. In: Advances in Cryptology - CRYPTO '89. Lecture Notes in Computer Science, vol. 435, pp. 416–427. Springer-Verlag, Berlin (1990)

[6] Fleischmann, E., Forler, C., Lucks, S., Wenzel, J.: The collision security of MDC-4. Cryptology ePrint Archive, Report 2012/096 (2012)

[7] 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-Verlag, Berlin (2009)

[8] 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-Verlag, Berlin (2005)

[9] 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-Verlag, Berlin (2006)

[10] ISO/IEC 10118-2:2010. Information technology – Security techniques – Hash-functions – Part 2: Hash-functions using an n-bit block cipher (1994, revised in 2010)

[11] Knudsen, L., Mendel, F., Rechberger, C., Thomsen, S.: Cryptanalysis of MDC-2. In: Advances in Cryptology - EUROCRYPT 2009. Lecture Notes in Computer Science, vol. 5479, pp. 106–120. Springer-Verlag, Berlin (2009)

[12] Knudsen, L., Preneel, B.: Fast and secure hashing based on codes. In: Advances in Cryptology - CRYPTO '97. Lecture Notes in Computer Science, vol. 1294, pp. 485–498. Springer-Verlag, Berlin (1997)

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

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

[15] Lee, J., Stam, M., Steinberger, J.: The collision security of Tandem-DM in the ideal cipher model. Cryptology ePrint Archive, Report 2010/409 (2010), full version of [16]

[2] We assume the padding takes less than n bits.

[3] Due to collisions in the lists L_0, L_1, the amount of 2^n will usually not be reached. Elaborate statistical analysis shows that the average number of leaves at distance n from the root (Z, Z) varies between 2^{n-1} and 2^n.

[16] 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-Verlag, Berlin (2011)
[17] Lee, J., Stam, M., Steinberger, J.: The preimage security of double-block-length compression functions. Cryptology ePrint Archive, Report 2011/210 (2011)
[18] Matyas, S., Meyer, C., Oseas, J.: Generating strong one-way functions with cryptographic algorithm. IBM Techn. Disclosure Bull. 27(10A), 5658–5659 (1985)
[19] Merkle, R.: One way hash functions and DES. In: Advances in Cryptology - CRYPTO '89. Lecture Notes in Computer Science, vol. 435, pp. 428–446. Springer-Verlag, Berlin (1990)
[20] Meyer, C., Schilling, M.: Secure program load with manipulation detection code. In: Proc. Securicom. pp. 111–130 (1988)
[21] 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-Verlag, Berlin (2009)
[22] 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-Verlag, Berlin (2004)
[23] 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-Verlag, Berlin (2007)

A Appendix to Sect. 3.1: Proofs of Lems. 2-7

In this appendix, we prove Lems. 2-7 of Sect. 3.1. The proofs of Lems. 2-6 are supported by Figs. 7-10, and for these figures the same convention is used as for Fig. 4. In particular, the queries corresponding to locations a and $!a$ are required to be different, and the same for the queries at positions $(b, !b)$ and $(c, !c)$.

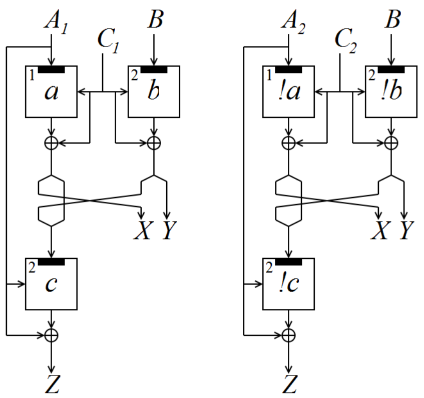


Fig. 7. Configuration $\mathsf{col}_{0001}(\mathcal{Q})$ of Lem. 2. We require $(A_1, C_1) \neq (A_2, C_2)$.

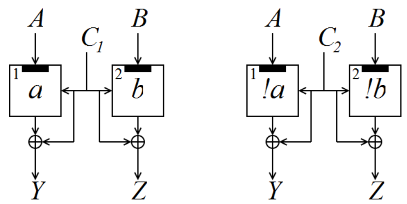


Fig. 8. Configuration $\mathsf{col}_{0011}(\mathcal{Q})$ of Lem. 3. We require $C_1 \neq C_2$.

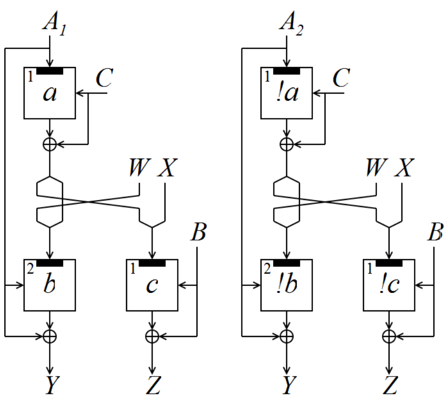


Fig. 9. Configuration $\mathsf{col}_{0100}(\mathcal{Q})$ of Lem. 4. We require $A_1 \neq A_2$.

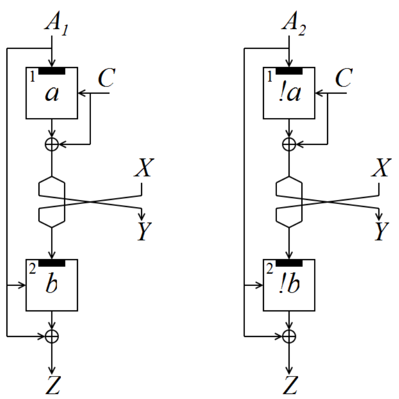


Fig. 10. Configuration $\mathsf{col}_{0101}(\mathcal{Q})$ of Lem. 5. We require $A_1 \neq A_2$.

A.1 Proof of Lem. 2

The cases are equivalent by symmetry, and we consider $\mathsf{col}_{0001}(\mathcal{Q}_q)$ only. A visualization of configuration $\mathsf{col}_{0001}(\mathcal{Q}_q)$ can be found in Fig. 7. For the basic proof idea, we refer to the proof of Lem. 1. Let $i \in \{1, \ldots, q\}$. As in Lem. 1, we assume $\neg\mathsf{help}(\mathcal{Q}_i)$ and analyze the probability the i-th query makes $\mathsf{col}_{0001}(\mathcal{Q}_i)$ satisfied.

Without loss of generality (by symmetry), the i-th query is either a query to E_1 and occurs at position $1\mathsf{tl}$, or it is a query to E_2 and occurs at position $1\mathsf{tr}$ and/or $1\mathsf{bl}$. We distinguish among these four cases. It may be the case that the i-th query also occurs in the right word, but these cases are automatically included.

Query occurs at $1\mathsf{tl}$**.** By $\neg\mathsf{help}_1(\mathcal{Q}_i)$, there are $\leq t_1$ choices for $(1\mathsf{bl}, 2\mathsf{bl})$. For any of these $\leq t_1$ choices, let $K_{1\mathsf{bl}}$ and $K_{2\mathsf{bl}}$ be the key inputs corresponding to positions $1\mathsf{bl}$ and $2\mathsf{bl}$. By $\neg\mathsf{help}_3(\mathcal{Q}_i)$, there are $\leq t_2$ choices for $2\mathsf{tl}$. For any of these $\leq t_1 t_2$ choices $2\mathsf{tl}$, we obtain a different X. The i-th query is successful only if its XOR-output equals $X \| K_{1\mathsf{bl}}^r$, which happens with probability at most $\frac{1}{2^n - q}$. The total success probability is at most $\frac{t_1 t_2}{2^n - q}$.

Query occurs at $1\mathsf{tr}$ **but not at** $1\mathsf{bl}$**.** By $\neg\mathsf{help}_1(\mathcal{Q}_i)$, there are $\leq t_1$ choices for $(1\mathsf{bl}, 2\mathsf{bl})$. For any of these $\leq t_1$ choices, let $K_{1\mathsf{bl}}$ and $K_{2\mathsf{bl}}$ be the key inputs corresponding to positions $1\mathsf{bl}$ and $2\mathsf{bl}$. By $\neg\mathsf{help}_2(\mathcal{Q}_i)$, there are $\leq t_2$ choices for $2\mathsf{tr}$. For any of these $\leq t_1 t_2$ choices $2\mathsf{tr}$, we obtain a different Y. The i-th query is successful only if its XOR-output equals $K_{1\mathsf{bl}}^l \| Y$, which happens with probability at most $\frac{1}{2^n - q}$. The total success probability is at most $\frac{t_1 t_2}{2^n - q}$.

Query occurs at $1\mathsf{bl}$ **but not at** $1\mathsf{tr}$**: inverse query** $x \leftarrow E_2^{-1}(K, y)$**.** By $\neg\mathsf{help}_3(\mathcal{Q}_i)$, there are $\leq t_2$ choices for $1\mathsf{tl}$. For any of these $\leq t_2$ choices, let $K_{1\mathsf{tl}}$ be the key input corresponding to position $1\mathsf{tl}$. The i-th query is successful only if $x = K_{1\mathsf{tl}}$, which happens with probability at most $\frac{1}{2^n - q}$. The total success probability is at most $\frac{t_2}{2^n - q}$.

Query occurs at $1\mathsf{bl}$ **but not at** $1\mathsf{tr}$**: forward query** $y \leftarrow E_2(K, x)$**.** By $\neg\mathsf{help}_2(\mathcal{Q}_i)$, there are $\leq t_2$ choices for $1\mathsf{tr}$. For any of these $\leq t_2$ choices $1\mathsf{tr}$, the query at position $1\mathsf{tl}$ is uniquely determined (it requires key input x and message input C_1 defined by query $1\mathsf{tr}$), and so are the strings (B, X). By $\neg\mathsf{help}_2(\mathcal{Q}_i)$, there are $\leq t_2$ choices for $2\mathsf{tl}$. For any of these $\leq t_2^2$ choices $2\mathsf{tl}$, the query at position $2\mathsf{tr}$ is uniquely determined (it requires key input B and message input C_2 defined by query $2\mathsf{tl}$). Consequently, the query at position $2\mathsf{br}$ is uniquely determined, and so is the XOR-output Z of $2\mathsf{br}$. The i-th query is successful only if its XOR-output equals this value Z, which happens with probability at most $\frac{1}{2^n - q}$. The total success probability is at most $\frac{t_2^2}{2^n - q}$.

Query occurs at $1\mathsf{tr}$ **and** $1\mathsf{bl}$**.** Let K be the key input for the i-th query. As the query occurs at both positions, we require $K^l = Z^l$, which fixes Z^l. By $\neg\mathsf{help}_2(\mathcal{Q}_i)$, there are $\leq t_2$ choices for $2\mathsf{bl}$. For any of these $\leq t_2$ choices $2\mathsf{bl}$, we obtain a different Z. The i-th query is successful only if its XOR-output equals this value Z, which happens with probability at most $\frac{1}{2^n - q}$. The total success probability is at most $\frac{t_2}{2^n - q}$.

The i-th query is successful with probability at most $\frac{t_1 t_2}{2^n - q}$ if it is a query to E_1 and at most $\frac{t_1 t_2 + t_2 + t_2^2}{2^n - q}$ if it is a query to E_2. Overall, the query is successful with probability at most $\frac{t_1 t_2 + t_2 + t_2^2}{2^n - q}$. The claimed bound is obtained by summing over $i = 1, \ldots, q$.

A.2 Proof of Lem. 3

A visualization of configuration $\mathsf{col}_{0011}(\mathcal{Q}_q)$ can be found in Fig. 8. For the basic proof idea, we refer to the proof of Lem. 1. Let $i \in \{1, \ldots, q\}$. As in Lem. 1, we assume $\neg\mathsf{help}(\mathcal{Q}_i)$ and analyze the probability the i-th query makes $\mathsf{col}_{0011}(\mathcal{Q}_i)$ satisfied.

Without loss of generality (by symmetry), the i-th query is a query to E_1 and occurs at position $1\mathsf{tl}$. It may be the case that the i-th query also occurs in the right word, but this case is automatically included.

Query occurs at 1tl**.** By $\neg\mathsf{help}_1(\mathcal{Q}_i)$, there are $\leq t_1$ choices for $(1\mathsf{tr}, 2\mathsf{tr})$. For any of these $\leq t_1$ choices, as A is fixed (it equals the key input for the i-th query) the query at position 2tl is uniquely determined, and so is the XOR-output Y of 2tl. The i-th query is successful only if its XOR-output equals this value Y, which happens with probability at most $\frac{1}{2^n-q}$. The total success probability is at most $\frac{t_1}{2^n-q}$.

Hence, either for the case the query is made to E_1 or E_2, it is successful with probability at most $\frac{t_1}{2^n-q}$. The claimed bound is obtained by summing over $i = 1, \ldots, q$.

A.3 Proof of Lem. 4

The cases are equivalent by symmetry, and we consider $\mathsf{col}_{0100}(\mathcal{Q}_q)$ only. A visualization of configuration $\mathsf{col}_{0100}(\mathcal{Q}_q)$ can be found in Fig. 9. For the basic proof idea, we refer to the proof of Lem. 1. Let $i \in \{1, \ldots, q\}$. As in Lem. 1, we assume $\neg\mathsf{help}(\mathcal{Q}_i)$ and analyze the probability the i-th query makes $\mathsf{col}_{0100}(\mathcal{Q}_i)$ satisfied.

Without loss of generality (by symmetry), the i-th query is either a query to E_2 and occurs at position 1bl, or it is a query to E_1 and occurs at position 1tl and/or 1br. We distinguish among these four cases. It may be the case that the i-th query also occurs in the right word, but these cases are automatically included.

Query occurs at 1bl**.** By $\neg\mathsf{help}_1(\mathcal{Q}_i)$, there are $\leq t_1$ choices for $(1\mathsf{br}, 2\mathsf{br})$. For any of these $\leq t_1$ choices, let $K_{2\mathsf{br}}$ be the key input corresponding to position 2br. By $\neg\mathsf{help}_2(\mathcal{Q}_i)$, there are $\leq t_2$ choices for 2tl. For any of these $\leq t_1 t_2$ choices 2tl, the query at position 2bl is uniquely determined, and so is the XOR-output Y of 2bl. The i-th query is successful only if its XOR-output equals this value Y, which happens with probability at most $\frac{1}{2^n-q}$. The total success probability is at most $\frac{t_1 t_2}{2^n-q}$.

Query occurs at 1tl **but not at** 1br**.** We note that the query at position $1\mathsf{tr} = 2\mathsf{tr}$ is not depicted in Fig. 9 but is defined as a query $(2, B, C, W \| X \oplus C)$. By $\neg\mathsf{help}_1(\mathcal{Q}_i)$, there are $\leq t_1$ choices for queries at positions $(1\mathsf{br}, 2\mathsf{br})$. For any of these $\leq t_1$ choices, let $K_{1\mathsf{br}}$ and $K_{2\mathsf{br}}$ be the key inputs corresponding to positions 1br and 2br. By $\neg\mathsf{help}_2(\mathcal{Q}_i)$, there are $\leq t_2$ choices for 2tl. For any of these $\leq t_1 t_2$ choices 2tl, the query at position $1\mathsf{tr} = 2\mathsf{tr}$ and consequently the query at position 2bl is uniquely determined, and so is the XOR-output Y of 2bl. By $\neg\mathsf{help}_4(\mathcal{Q}_i)$, there are $\leq t_3$ choices for 1bl. For any of these $\leq t_1 t_2 t_3$ choices 1bl, let $K_{1\mathsf{bl}}$ be the key input corresponding to position 1bl. The i-th query is successful only if its XOR-output equals $K^l_{1\mathsf{br}} \| K^r_{1\mathsf{bl}}$, which happens with probability at most $\frac{1}{2^n-q}$. The total probability is at most $\frac{t_1 t_2 t_3}{2^n-q}$.

Query occurs at 1br **but not at** 1tl**.** By $\neg\mathsf{help}_1(\mathcal{Q}_i)$, there are $\leq t_1$ choices for $(1\mathsf{bl}, 2\mathsf{bl})$. For any of these $\leq t_1$ choices, let $K_{2\mathsf{bl}}$ be the key input corresponding to position 2bl. By $\neg\mathsf{help}_3(\mathcal{Q}_i)$, there are $\leq t_2$ choices for 2tl. For any of these $\leq t_1 t_2$ choices 2tl, the key input to the query at position 2br, say $K_{2\mathsf{br}}$, is uniquely determined. Suppose the i-th query is a forward query $y \leftarrow E_1(K, x)$ (exactly the same reasoning applies to inverse queries). If $K_{2\mathsf{br}} = K$, the queries at positions 1br and 2br must be the same and the collision is invalid. Therefore, we assume $K_{2\mathsf{br}} \neq K$. For the key $K_{2\mathsf{br}}$, let $(K_{2\mathsf{br}}, x_{2\mathsf{br}}, y_{2\mathsf{br}})$ be any query in the query history. The i-th query makes the configuration satisfied if $x_{2\mathsf{br}} = x$ and $x_{2\mathsf{br}} \oplus y_{2\mathsf{br}} = x \oplus y$, or more concretely if

$$x_{2\mathsf{br}} = x \text{ and } y_{2\mathsf{br}} = y. \tag{11}$$

This means that, irrespectively of whether the i-th query is a forward or inverse query, the query at position 2br is uniquely determined. The i-th query is successful only if it satisfies (11), which happens with probability at most $\frac{1}{2^n-q}$. The total probability is at most $\frac{t_1 t_2}{2^n-q}$.

Query occurs at 1tl **and** 1br**.** Let K be the key input for the i-th query. As the query occurs at both positions, we require $K^l = Z^l$, which fixes Z^l. By $\neg\mathsf{help}_2(\mathcal{Q}_i)$, there are $\leq t_2$ choices for 2br. For any of these $\leq t_2$ choices 2br, we obtain a different Z. The i-th query is successful only if its XOR-output equals this value Z, which happens with probability at most $\frac{1}{2^n-q}$. The total success probability is at most $\frac{t_2}{2^n-q}$.

The i-th query is successful with probability at most $\frac{t_1 t_2}{2^n-q}$ if it is a query to E_2 and at most $\frac{t_1t_2t_3+t_1t_2+t_2}{2^n-q}$ if it is a query to E_1. Overall, the query is successful with probability at most $\frac{t_1t_2t_3+t_1t_2+t_2}{2^n-q}$. The claimed bound is obtained by summing over $i = 1, \dots, q$.

A.4 Proof of Lem. 5

The cases are equivalent by symmetry, and we consider $\mathsf{col}_{0101}(\mathcal{Q}_q)$ only. A visualization of configuration $\mathsf{col}_{0101}(\mathcal{Q}_q)$ can be found in Fig. 10. For the basic proof idea, we refer to the proof of Lem. 1. Let $i \in \{1, \dots, q\}$. As in Lem. 1, we assume $\neg\mathsf{help}(\mathcal{Q}_i)$ and analyze the probability the i-th query makes $\mathsf{col}_{0101}(\mathcal{Q}_i)$ satisfied.

Without loss of generality (by symmetry), the i-th query is either a query to E_1 and occurs at position $1\mathsf{tl}$, or it is a query to E_2 and occurs at position $1\mathsf{bl}$. We distinguish among these two cases. It may be the case that the i-th query also occurs in the right word, but these cases are automatically included.

Query occurs at $1\mathsf{tl}$**.** By $\neg\mathsf{help}_1(\mathcal{Q}_i)$, there are $\leq t_1$ choices for $(1\mathsf{bl}, 2\mathsf{bl})$. For any of these $\leq t_1$ choices, let $K_{1\mathsf{bl}}$ and $K_{2\mathsf{bl}}$ be the key inputs corresponding to positions $1\mathsf{bl}$ and $2\mathsf{bl}$. By $\neg\mathsf{help}_3(\mathcal{Q}_i)$, there are $\leq t_2$ choices for $2\mathsf{tl}$. For any of these $\leq t_1 t_2$ choices $2\mathsf{tl}$, we obtain a different Y. The i-th query is successful only if its XOR-output equals $Y \| K_{1\mathsf{bl}}^r$, which happens with probability at most $\frac{1}{2^n-q}$. The total success probability is at most $\frac{t_1 t_2}{2^n-q}$.

Query occurs at $1\mathsf{bl}$**.** Let K be the key input for the i-th query. By $\neg\mathsf{help}_3(\mathcal{Q}_i)$, there are $\leq t_2$ choices for $1\mathsf{tl}$. For any of these $\leq t_2$ choices, we obtain a different Y. By $\neg\mathsf{help}_2(\mathcal{Q}_i)$, there are $\leq t_2$ choices for $2\mathsf{tl}$. For any of these $\leq t_2^2$ choices $2\mathsf{tl}$, the query at position $2\mathsf{bl}$ is uniquely determined, and so is the XOR-output Z of $2\mathsf{bl}$. The i-th query is successful only if its XOR-output equals this value Z, which happens with probability at most $\frac{1}{2^n-q}$. The total success probability is at most $\frac{t_2^2}{2^n-q}$.

The i-th query is successful with probability at most $\frac{t_1t_2+t_2^2}{2^n-q}$. The claimed bound is obtained by summing over $i = 1, \dots, q$.

A.5 Proof of Lem. 6

If $1\mathsf{tl} = 2\mathsf{tl}$ and $1\mathsf{tr} = 2\mathsf{tr}$, we obtain $(A_1, B_1, C_1) = (A_2, B_2, C_2)$ in the configuration of Fig. 3, and the collision is invalid. The same observation applies if $(1\mathsf{tl}, 1\mathsf{br}) = (2\mathsf{tl}, 2\mathsf{br})$ or $(1\mathsf{tr}, 1\mathsf{bl}) = (2\mathsf{tr}, 2\mathsf{bl})$.

A.6 Proof of Lem. 7

It suffices to consider the events $\mathsf{Pr}(\mathsf{help}_k(\mathcal{Q}_q))$ $(k = 1, \dots, 4)$ separately.
$\mathsf{help}_1(\mathcal{Q}_q)$**.** We copy the approach of Steinberger [23]. For $i \neq j$, the two queries (K_i, x_i, y_i) and (K_j, x_j, y_i) have the same XOR-output with probability at most $\frac{1}{2^n-q}$. Hence, the expected value $\mathsf{E}(x_i \oplus y_i = x_j \oplus y_j)$ is at most $\frac{1}{2^n-q}$, and consequently

$$\mathsf{E}\left(\left\{(K_i, x_i, y_i), (K_j, x_j, y_j) \in \mathcal{Q}_q \;\middle|\; i \neq j \;\wedge\; x_i \oplus y_i = x_j \oplus y_j\right\}\right) \leq \sum_{i \neq j} \frac{1}{2^n - q} \leq \frac{q^2}{2^n - q}.$$

By Markov's inequality, we obtain

$$\mathsf{Pr}\left(\mathsf{help}_1(\mathcal{Q}_q)\right) \leq \frac{q^2}{t_1(2^n - q)}. \tag{12}$$

$\mathsf{help}_k(\mathcal{Q}_q)$ **for** $k \in \{2, 3\}$**.** The cases are equivalent by symmetry, and we consider $\mathsf{help}_2(\mathcal{Q}_q)$ only. Let $z \in \mathbb{Z}_2^{n/2}$. Consider the i-th query (K_i, x_i, y_i). This query makes equation $(x_i \oplus y_i)^l = z$ satisfied with probability at most $\frac{2^{n/2}}{2^n-q}$. More than t_2 queries result in a solution with probability at most

$\binom{q}{t_2}\left(\frac{2^{n/2}}{2^n-q}\right)^{t_2} \leq \left(\frac{eq2^{n/2}}{t_2(2^n-q)}\right)^{t_2}$, where we use Stirling's approximation ($t! \geq (t/e)^t$ for any t). Considering any possible choice for z, we obtain for $k = 2, 3$:

$$\Pr(\mathsf{help}_k(\mathcal{Q}_q)) \leq 2^{n/2}\left(\frac{eq2^{n/2}}{t_2(2^n-q)}\right)^{t_2}. \tag{13}$$

$\mathsf{help}_4(\mathcal{Q}_q)$**.** A similar analysis as for $\mathsf{help}_2(\mathcal{Q}_q)$ results in the following bound:

$$\Pr(\mathsf{help}_4(\mathcal{Q}_q)) \leq 2^{n}\left(\frac{eq}{t_3(2^n-q)}\right)^{t_3}. \tag{14}$$

The claim is obtained by adding (12-14).

B Appendix to Sect. 4.1: Proofs of Lems. 8-9

B.1 Proof of Lem. 8

We consider the probability of the adversary finding a solution to configuration $\mathsf{pre}(\mathcal{Q}_q)$ of Fig. 6, in such a way that $\mathcal{Q}_q$ satisfies $\neg\mathsf{help}(\mathcal{Q}_q)$. For a set of solutions complying with configuration $\mathsf{pre}(\mathcal{Q}_q)$, it may be the case that two queries are the same or belong to the same super query. 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. Note that a winning query can contribute to at most two positions, due to the usage of different block ciphers. We distinguish among the following cases.

1. The winning query contributes to exactly one position of configuration $\mathsf{pre}(\mathcal{Q}_q)$;
2. The winning query contributes to exactly two positions of configuration $\mathsf{pre}(\mathcal{Q}_q)$ (either positions $(\mathsf{tl}, \mathsf{br})$ or $(\mathsf{tr}, \mathsf{bl})$).

Case 1. In this case, the winning query may be a normal query or a super query. As in [15, 17], we make use of "wish lists" for the analysis of this case. Intuitively, a wish list is a continuously updated sequence of query tuples that would make configuration $\mathsf{pre}(\mathcal{Q})$ satisfied. During the attack of the adversary, we maintain four initially empty wish lists $\mathcal{W}_{\mathsf{tl}}$, $\mathcal{W}_{\mathsf{tr}}$, $\mathcal{W}_{\mathsf{bl}}$, $\mathcal{W}_{\mathsf{br}}$, corresponding to the four positions of configuration $\mathsf{pre}(\mathcal{Q})$. If a query is made, the wish lists are updated according to the following requirements:

- If the query fits $\mathsf{pre}_{\mathsf{tl}}(\mathcal{Q})$ of Fig. 11 for any two other queries in the query history, the corresponding tuple $(1, A, C, (W\|X) \oplus C)$ is added to $\mathcal{W}_{\mathsf{tl}}$;
- If the query fits $\mathsf{pre}_{\mathsf{tr}}(\mathcal{Q})$ of Fig. 11 for any two other queries in the query history, the corresponding tuple $(2, B, C, (W\|X) \oplus C)$ is added to $\mathcal{W}_{\mathsf{tr}}$;
- If the query fits $\mathsf{pre}_{\mathsf{bl}}(\mathcal{Q})$ of Fig. 11 for any two other queries in the query history, the corresponding tuple $(2, W\|X, A, Y \oplus A)$ is added to $\mathcal{W}_{\mathsf{bl}}$;
- If the query fits $\mathsf{pre}_{\mathsf{br}}(\mathcal{Q})$ of Fig. 11 for any two other queries in the query history, the corresponding tuple $(1, W\|X, B, Z \oplus B)$ is added to $\mathcal{W}_{\mathsf{br}}$.

As in this case we consider the winning query to be different from all other queries made before, we can assume a query never adds itself to a wish list. It is clear that the adversary finds a preimage for MDC-4 (in this case) only if it makes a query that is already a member of any of the wish lists. Suppose the adversary makes a query $E_1(K, x)$ (either as a normal query or as a part of a super query), and suppose $(1, K, x, y) \in \mathcal{W}_{\mathsf{tl}} \cup \mathcal{W}_{\mathsf{br}}$ for some y. Then, we say that $(1, K, x, y)$ is wished for, and the wish is granted if the response of the block cipher is y. Similar naming is used for inverse queries to E_1 and for queries to E_2. Notice that the adversary may wish for multiple queries at the same time, but this does not invalidate the analysis. Additionally, each wish list element can be wished for only once. In order to find a preimage, the adversary needs at least a wish to be granted. Let (k, K, x, y) be an element in any of the wish lists, and suppose the adversary makes a query $E_k(K, x)$ or $E_k^{-1}(K, y)$. In case of normal queries, the answer is generated from a set of size at least 2^{n-1}, and the wish is granted with probability at most $\frac{1}{2^{n-1}}$. In case the query is a part of a super query, the answer is generated from a set of size

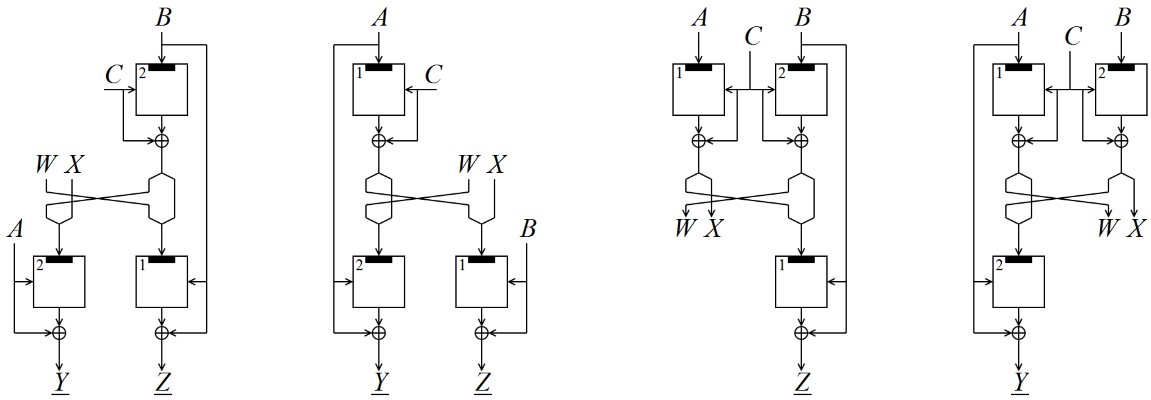


Fig. 11. From left to right: configurations $\mathsf{pre}_{\mathsf{tl}}(\mathcal{Q})$, $\mathsf{pre}_{\mathsf{tr}}(\mathcal{Q})$, $\mathsf{pre}_{\mathsf{bl}}(\mathcal{Q})$, and $\mathsf{pre}_{\mathsf{br}}(\mathcal{Q})$.

exactly 2^{n-1} and the wish is also granted with probability at most $\frac{1}{2^{n-1}}$. Because each element of the wish lists can be wished for only once, the adversary finds a preimage with probability at most

$$\frac{|\mathcal{W}_{\mathsf{tl}}| + |\mathcal{W}_{\mathsf{tr}}| + |\mathcal{W}_{\mathsf{bl}}| + |\mathcal{W}_{\mathsf{br}}|}{2^{n-1}}.$$

It remains to bound the sizes of the wish lists after q queries. Configuration $\mathsf{pre}_{\mathsf{tl}}(\mathcal{Q}_q)$ of Fig. 11 has $\le t_2$ solutions for each bl and br (by $\neg\mathsf{help}_3(\mathcal{Q}_q)$), and consequently $\le t_2$ solutions for tr (by $\neg\mathsf{help}_3(\mathcal{Q}_q)$). Thus $|\mathcal{W}_{\mathsf{tl}}| \le t_2^3$, and similarly we obtain $|\mathcal{W}_{\mathsf{tr}}| \le t_2^3$. Configuration $\mathsf{pre}_{\mathsf{bl}}(\mathcal{Q}_q)$ of Fig. 11 has $\le t_2$ solutions for br (by $\neg\mathsf{help}_3(\mathcal{Q}_q)$), and consequently $\le t_1$ solutions for tl (by $\neg\mathsf{help}_1(\mathcal{Q}_q)$). For any of these $\le t_1 t_2$ choices, the query at position tr is uniquely determined (if it exists at all). Thus $|\mathcal{W}_{\mathsf{bl}}| \le t_1 t_2$, and similarly $|\mathcal{W}_{\mathsf{br}}| \le t_1 t_2$ (using $\neg\mathsf{help}_2(\mathcal{Q}_q)$). Hence, in this case a preimage is found with probability at most $\frac{4t_2^3 + 4t_1 t_2}{2^n}$.

Case 2. We make the following distinction, and consider the two sub-cases separately.

1. The contributed queries are different for both positions;
2. The contributed queries are the same for both positions.

Case 2.1. In this particular case, the winning query must be a super query. Similar to case 1, we make use of wish lists, but now for the specific case that a super query contributes two queries to a configuration. Note that if a super query contributes to positions $(\mathsf{tl}, \mathsf{br})$, the left half of the XOR-output of tl should equal the left half of the key input to br, which is the same as the key input to tl (similar for super queries contributing to $(\mathsf{tr}, \mathsf{bl})$). During the attack of the adversary, we maintain two initially empty wish lists $\mathcal{W}_1$, $\mathcal{W}_2$, corresponding to super queries to the block ciphers E_1 and E_2. If a query is made by the adversary, the wish lists are updated according to the following requirements:

- If the query fits $\mathsf{pre}_1(\mathcal{Q})$ of Fig. 12 for any query in the query history, the corresponding tuple $(1, V\|X, C, (V\|W) \oplus C, B, Z \oplus B)$ is added to $\mathcal{W}_1$;
- If the query fits $\mathsf{pre}_2(\mathcal{Q})$ of Fig. 12 for any query in the query history, the corresponding tuple $(2, V\|X, C, (V\|W) \oplus C, A, Y \oplus A)$ is added to $\mathcal{W}_2$.

Of these tuples, the first element identifies the corresponding block cipher, the second element the key for which the super query is made, the third and fourth element define the input and output of the cipher in the top row (either left or right), and the fifth and sixth element define the input and output of the cipher in the bottom row (either right or left). A query trivially does not add itself to the wish list as a query to E_1 only affects $\mathcal{W}_2$ and a query to E_2 only affects $\mathcal{W}_1$. Suppose the adversary makes a super query to E_k for key K, and suppose $(k, K, x_{\mathsf{tl}}, y_{\mathsf{tl}}, x_{\mathsf{br}}, y_{\mathsf{br}}) \in \mathcal{W}_1$ for some $x_{\mathsf{tl}}, y_{\mathsf{tl}}, x_{\mathsf{br}}, y_{\mathsf{br}}$. This wish is then granted if the response satisfies $y_{\mathsf{tl}} = E_k(K, x_{\mathsf{tl}})$ and $y_{\mathsf{br}} = E_k(K, x_{\mathsf{br}})$. In order to find a preimage, the adversary needs at least a wish to be granted. As the answers are generated from a set of size exactly 2^{n-1}, a wish is granted with probability at most $\frac{1}{2^{n-1}(2^{n-1}-1)}$. Because each element of the wish lists can be wished for only once, the adversary finds a preimage with probability at most

$$\frac{|\mathcal{W}_1| + |\mathcal{W}_2|}{2^{n-1}(2^{n-1}-1)}.$$

It remains to bound the sizes of the wish lists after q queries. Configuration $\mathsf{pre}_1(\mathcal{Q}_q)$ has $\leq t_2$ solutions for bl (by $\neg\mathsf{help}_3(\mathcal{Q}_q)$), and at most $\leq t_1$ solutions for tr (by $\neg\mathsf{help}_1(\mathcal{Q}_q)$). Thus, $|\mathcal{W}_1| \leq t_1 t_2$ and similarly $|\mathcal{W}_2| \leq t_1 t_2$. Hence, in this case a preimage is found with probability at most $\frac{16 t_1 t_2}{2^{2n}}$.

Fig. 12. Configuration $\mathsf{pre}_1(\mathcal{Q})$ (left) and $\mathsf{pre}_2(\mathcal{Q})$ (right).

Case 2.2. In this case, the winning query may be a normal query or a super query. We first consider the winning query to contribute to $\mathsf{tl} = \mathsf{br}$. Suppose the adversary makes a query $y \leftarrow E_1(K, x)$ (either as a normal query or as a part of a super query). As it occurs at position br we require $x \oplus y = Z$ (because of this, the analysis for inverse queries is equivalent). Additionally, the query at position tr should have key input as well as message input equal to x. This particularly means that the key input K to $\mathsf{tl} = \mathsf{br}$ must satisfy $K = Z^l \| (E_2(x,x) \oplus x)^r$. By construction of the queries at positions bl, br, the adversary can only succeed if it ever finds an $x \in \mathbb{Z}_2^n$ that satisfies

$$E_2((E_2(x,x) \oplus x)^l \| Z^r, Z^l \| (E_2(x,x) \oplus x)^r) \oplus Z^l \| (E_2(x,x) \oplus x)^r = Y,$$
$$E_1(Z^l \| (E_2(x,x) \oplus x)^r, x) \oplus x = Z.$$

As Y and Z are fixed, the adversary finds such x with probability at most $\frac{2^n}{2^{n-1}2^{n-1}} = \frac{4}{2^n}$. The same probability bound is obtained for winning queries to appear at $(\mathsf{tr}, \mathsf{bl})$. Consequently, a preimage is found in this case with probability at most $\frac{8}{2^n}$.

The claim is obtained by summing the bounds obtained for the two cases.

B.2 Proof of Lem. 9

It suffices to consider the events $\mathsf{Pr}(\mathsf{help}_k(\mathcal{Q}_q))$ $(k = 1, 2, 3)$ separately.
$\mathsf{help}_k(\mathcal{Q}_q)$ **for** $k \in \{1, 2\}$**.** The cases are equivalent by symmetry, and we consider $\mathsf{help}_1(\mathcal{Q}_q)$ only. Let $z \in \mathbb{Z}_2^{n/2}$. Denote by $\mathcal{Q}_q^{(n)}$ the restriction of $\mathcal{Q}_q$ to normal queries, and by $\mathcal{Q}_q^{(s)}$ the restriction of $\mathcal{Q}_q$ to queries that belong to super queries. In order for $\mathcal{Q}_q$ to have more than t_1 solutions to $(x_i \oplus y_i)^l = z$, at least one of the following criteria needs to hold:

1. $\mathcal{Q}_q^{(n)}$ has more than $t_1/2$ solutions;
2. $\mathcal{Q}_q^{(s)}$ has more than $t_1/2$ solutions.

We consider these two scenarios separately. In case of normal queries, each query (K_i, x_i, y_i) is answered with a value generated at random from a set of size at least 2^{n-1}, and hence it satisfies $(x_i \oplus y_i)^l = z$ with probability at most $\frac{2^{n/2}}{2^{n-1}} = \frac{2}{2^{n/2}}$. More than $t_1/2$ queries result in a solution with probability at most $\binom{q}{t_1/2} \left(\frac{2}{2^{n/2}}\right)^{t_1/2} \leq \left(\frac{4eq}{t_1 2^{n/2}}\right)^{t_1/2}$.

The analysis for super queries is more elaborate. In order for $\mathcal{Q}_q^{(s)}$ to have more than $t_1/2$ solutions, as at most $q/2^{n-1}$ super queries occur, at least one of the super queries needs to provide more than

$t'_1 := \frac{t_1}{2q/2^{n-1}} = \frac{t_1 2^n}{4q}$ solutions. Consider any super query, consisting of 2^{n-1} queries. It provides more than t'_1 solutions with probability at most

$$\binom{2^{n-1}}{t'_1} \prod_{j=0}^{t'_1-1} \frac{2^{n/2}}{2^{n-1}-j} \leq \binom{2^{n-1}}{t'_1} \left(\frac{2^{n/2}}{2^{n-1}-t'_1}\right)^{t'_1} \leq \left(\frac{e2^{n-1}2^{n/2}}{t'_1(2^{n-1}-t'_1)}\right)^{t'_1} .$$

Provided $t_1 \leq q$, we have $t'_1 = \frac{t_1 2^n}{4q} \leq 2^{n-2}$, and thus $\frac{1}{2^{n-1}-t'_1} \leq \frac{1}{2^{n-2}}$. Consequently, this super query adds more than $\frac{t_1 2^n}{4q}$ solutions with probability at most $\left(\frac{8eq}{t_1 2^{n/2}}\right)^{\frac{t_1 2^n}{4q}}$. 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 $k = 1, 2$:

$$\mathsf{Pr}\left(\mathsf{help}_k(\mathcal{Q}_q)\right) \leq 2^{n/2} \left(\frac{4eq}{t_1 2^{n/2}}\right)^{t_1/2} + 2^{n/2} \cdot \frac{q}{2^{n-1}} \left(\frac{8eq}{t_1 2^{n/2}}\right)^{\frac{t_1 2^n}{4q}} . \tag{15}$$

$\mathsf{help}_3(\mathcal{Q}_q)$**.** A similar analysis as for $\mathsf{help}_1(\mathcal{Q}_q)$ results in the following bound:

$$\mathsf{Pr}\left(\mathsf{help}_3(\mathcal{Q}_q)\right) \leq 2^n \left(\frac{4eq}{t_2 2^n}\right)^{t_2/2} + 2^n \cdot \frac{q}{2^{n-1}} \left(\frac{8eq}{t_2 2^n}\right)^{\frac{t_2 2^n}{4q}} . \tag{16}$$

The claim is obtained by adding (15) (twice) and (16).

C Appendix to Sect. 5.1: Proof of Thm. 3

In this appendix, we will extend the proof of Thm. 1 to Thm. 3. To to so, as explained in Sect. 5.1 we essentially only need to consider the probability that an adversary finds an $f_{\text{MDC-4}}$ evaluation where the input state consists of two different halves and the output state consists of two the same halves. The formal treatment of this is more elaborate.

We consider any adversary making q queries to its oracle $E = E_1 = E_2$, which tries to find a collision for MDC-4. Denote by (F_0, G_0) the initial state value of MDC-4, where $F_0 \neq G_0$. Suppose the adversary finds a collision, i.e. two lists

$$(F_0, G_0) \xrightarrow{m_1} (F_1, G_1) \xrightarrow{m_2} \cdots \xrightarrow{m_k} (F_k, G_k),$$
$$(F_0, G_0) \xrightarrow{m'_1} (F'_1, G'_1) \xrightarrow{m'_2} \cdots \xrightarrow{m'_k} (F'_{k'}, G'_{k'})$$

of internal state values of the two evaluations, where $k, k' \geq 1$ and $(F_k, G_k) = (F'_{k'}, G'_{k'})$. The collision is non-trivial if $k \neq k'$ or if $(F_i, G_i, m_i) \neq (F'_i, G'_i, m'_i)$ for some $i = 1, \ldots, k = k'$, and we consider non-trivial collisions only. If the adversary finds a collision of this form, we can distinguish between the following two cases:

(1) $F_i \neq G_i$ for all $i \in \{1, \ldots, k\}$ and $F'_i \neq G'_i$ for all $i \in \{1, \ldots, k'\}$;
(2) $F_i = G_i$ for some $i \in \{1, \ldots, k\}$ or $F'_i = G'_i$ for some $i \in \{1, \ldots, k'\}$.

Suppose the adversary finds a collision in case (1). This implies (by basic collision security preservation [1]) the adversary necessarily needs to obtain a query history $\mathcal{Q}_q$ of size q that satisfies configuration $\mathsf{col}'(\mathcal{Q}_q)$ of Fig. 13. Note that $\mathsf{col}'(\mathcal{Q}_q)$ differs from $\mathsf{col}(\mathcal{Q}_q)$ of Fig. 3 only in the fact that $A_1 \neq B_1$ and $A_2 \neq B_2$ (and that the function employs one block cipher rather than two).

On the other hand, suppose the adversary finds a collision in case (2). Without loss of generality, a state-half collision occurs in the first word. As $F_0 \neq G_0$, there exists an i such that $F_i = G_i$ but $F_{i-1} \neq G_{i-1}$. This means that in this case the adversary necessarily needs to obtain a query history $\mathcal{Q}_q$ of size q that satisfies configuration $\mathsf{statecol}'(\mathcal{Q}_q)$ of Fig. 14. Here, Z represents $F_i = G_i$ and (A, B) represents (F_{i-1}, G_{i-1}).

Concluding, we find

$$\mathbf{adv}^{\text{col}}_{\text{MDC-4}}(q) \leq \mathsf{Pr}\left(\mathsf{col}'(\mathcal{Q}_q) \vee \mathsf{statecol}'(\mathcal{Q}_q)\right), \tag{17}$$

and we consider the probability of obtaining any query history $\mathcal{Q}_q$ that satisfies configuration $\mathsf{col}'(\mathcal{Q}_q)$ or $\mathsf{statecol}'(\mathcal{Q}_q)$. As in Sect. 3.1, in these configurations we have omitted the shifting at the end. We stress that this does not harm the security analysis. We use the same block cipher labeling as before, i.e. in Fig. 13 the block ciphers are labeled $1\mathsf{tl},\dots,2\mathsf{br}$ and in Fig. 14 they are labeled $\mathsf{tl},\dots,\mathsf{br}$.

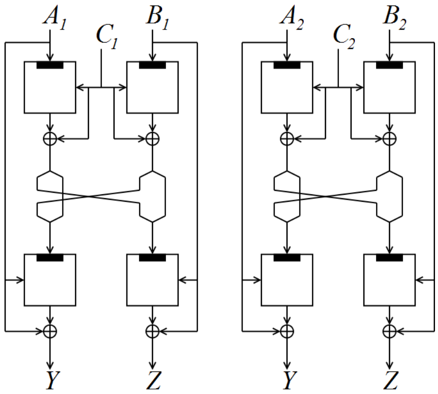


Fig. 13. Configuration $\mathsf{col}'(\mathcal{Q})$. We require $(A_1,B_1,C_1)\neq(A_2,B_2,C_2)$, $A_1\neq B_1$, and $A_2\neq B_2$.

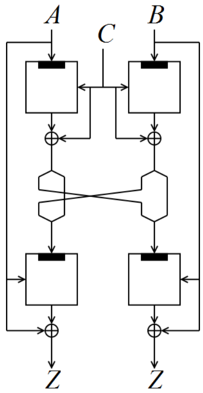


Fig. 14. Configuration $\mathsf{statecol}'(\mathcal{Q})$. We require $A\neq B$.

To analyze the probability of the adversary finding a query history $\mathcal{Q}_q$ that satisfies configuration $\mathsf{col}'(\mathcal{Q}_q)$ or $\mathsf{statecol}'(\mathcal{Q}_q)$, we employ the helping event $\mathsf{help}(\mathcal{Q}_q)$ from Sect. 3.1. We obtain for (17):

$$\mathsf{Pr}\left(\mathsf{col}'(\mathcal{Q}_q)\vee\mathsf{statecol}'(\mathcal{Q}_q)\right)\leq\mathsf{Pr}\left(\mathsf{col}'(\mathcal{Q}_q)\wedge\neg\mathsf{help}(\mathcal{Q}_q)\right)+\\ \mathsf{Pr}\left(\mathsf{statecol}'(\mathcal{Q}_q)\wedge\neg\mathsf{help}(\mathcal{Q}_q)\right)+\mathsf{Pr}\left(\mathsf{help}(\mathcal{Q}_q)\right).$$

As before, $\mathsf{col}'(\mathcal{Q}_q)$ is separated into 16 sub-configurations $\mathsf{col}'_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q})$ for $\alpha_{\mathsf{tl}},\alpha_{\mathsf{tr}},\alpha_{\mathsf{bl}},\alpha_{\mathsf{br}}\in\{0,1\}$ (cf. (4)), and we eventually obtain the following bound on $\mathbf{adv}^{\mathrm{col}}_{\mathrm{MDC\text{-}4}}(q)$:

$$\mathbf{adv}^{\mathrm{col}}_{\mathrm{MDC\text{-}4}}(q)\leq\sum_{\substack{\alpha_{\mathsf{tl}},\alpha_{\mathsf{tr}},\alpha_{\mathsf{bl}},\\ \alpha_{\mathsf{br}}\in\{0,1\}}}\mathsf{Pr}\left(\mathsf{col}'_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q}_q)\wedge\neg\mathsf{help}(\mathcal{Q}_q)\right)+\\ \mathsf{Pr}\left(\mathsf{statecol}'(\mathcal{Q}_q)\wedge\neg\mathsf{help}(\mathcal{Q}_q)\right)+\mathsf{Pr}\left(\mathsf{help}(\mathcal{Q}_q)\right). \tag{18}$$

Probability $\mathsf{Pr}\left(\mathsf{help}(\mathcal{Q}_q)\right)$ is already bounded in Lem. 7 of Sect. 3.1; the original proof carries over as it does not make any distinction between the two block ciphers. The evaluations of the probabilities constituting to the sum of (18) are different from the analysis in Lems. 1-6 of Sect. 3.1, in the sense that some additional cases need to be analyzed. In Lems. 10-15 we revisit the original lemmas for the case of $\mathsf{col}'(\mathcal{Q}_q)$. The probability bound on $\mathsf{statecol}'(\mathcal{Q}_q)\wedge\neg\mathsf{help}(\mathcal{Q}_q)$ is analyzed in Lem. 16.

Lemma 10. $\mathsf{Pr}\left(\mathsf{col}'_{0000}(\mathcal{Q}_q)\wedge\neg\mathsf{help}(\mathcal{Q}_q)\right)\leq\frac{(t_1t_2t_3+t_1t_2+2t_2)q}{2^n-q}$.

Proof. Configuration $\mathsf{col}'_{0000}(\mathcal{Q}_q)$ is the same as $\mathsf{col}_{0000}(\mathcal{Q}_q)$ of Fig. 4 with the additional restriction that $A_1\neq B_1$ and $A_2\neq B_2$ (and that only one block cipher is used).

For the basic proof idea, we refer to the proof of Lem. 1. The idea is for the i-th query, for $i\in\{1,\dots,q\}$, to assume $\neg\mathsf{help}(\mathcal{Q}_i)$ and to analyze the probability the i-th query makes $\mathsf{col}'_{0000}(\mathcal{Q}_i)$ satisfied. Then, we consider the success probability of the i-th query, and sum over all queries. Given that now the block ciphers E_1,E_2 are the same, additional (combinations of) positions have to be analyzed.

Without loss of generality (by symmetry), the i-th query occurs in the left word. It may be the case that the i-th query also occurs in the right word, but as becomes clear from the proof, these cases are automatically included. Note that, as $A_1\neq B_1$, it can impossibly occur at $(1\mathsf{tl},1\mathsf{tr})$ or $(1\mathsf{bl},1\mathsf{br})$. Therefore, without loss of generality it suffices to analyze the cases the query occurs at the following positions: $1\mathsf{tl}$ only, $1\mathsf{br}$ only, $(1\mathsf{tl},1\mathsf{br})$ only, or $(1\mathsf{tl},1\mathsf{bl})$ only. For the first three cases, the analysis is identical to Lem. 1. Remains to consider the case the query occurs at $(1\mathsf{tl},1\mathsf{bl})$ but not at $(1\mathsf{tr},1\mathsf{br})$.

Let K be the key input for the i-th query. As the query occurs at both positions, we require $K^r = Y^r$, which fixes Y^r. By $\neg\mathsf{help}_3(\mathcal{Q}_i)$, there are $\leq t_2$ choices for $2\mathsf{bl}$. For any of these $\leq t_2$ choices $2\mathsf{bl}$, we obtain a different Y. The i-th query is successful only if its XOR-output equals this value Y, which happens with probability at most $\frac{1}{2^n-q}$. The total success probability is at most $\frac{t_2}{2^n-q}$.

The i-th query is successful with probability at most $\frac{t_1t_2t_3+t_1t_2+2t_2}{2^n-q}$. The claimed bound is obtained by summing over $i = 1, \ldots, q$. □

Lemma 11. $\mathsf{Pr}\left(\mathsf{col}'_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) \leq \frac{(2t_1t_2+2t_2+t_2^2)q}{2^n-q}$ *for* $\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}} \in \{0001, 0010\}$.

Proof. The cases are equivalent by symmetry, and we consider $\mathsf{col}'_{0001}(\mathcal{Q}_q)$ only. Configuration $\mathsf{col}'_{0001}(\mathcal{Q}_q)$ is the same as $\mathsf{col}_{0001}(\mathcal{Q}_q)$ of Fig. 7 with the additional restriction that $A_1 \neq B$ and $A_2 \neq B$ (and that only one block cipher is used). For the basic proof idea, we refer to the proof of Lem. 10. Let $i \in \{1, \ldots, q\}$. We assume $\neg\mathsf{help}(\mathcal{Q}_i)$ and analyze the probability the i-th query makes $\mathsf{col}'_{0001}(\mathcal{Q}_i)$ satisfied.

Without loss of generality (by symmetry), the i-th query occurs at the following positions: $1\mathsf{tl}$ only, $1\mathsf{tr}$ only, $1\mathsf{bl}$ only, $(1\mathsf{tr}, 1\mathsf{bl})$ only, or $(1\mathsf{tl}, 1\mathsf{bl})$ only. For the first four cases, the analysis is identical to Lem. 2. For the remaining case, the analysis for configuration $\mathsf{col}'_{0000}(\mathcal{Q}_q)$ (Lem. 10) can be copied with Y replaced by Z.

The i-th query is successful with probability at most $\frac{2t_1t_2+2t_2+t_2^2}{2^n-q}$. The claimed bound is obtained by summing over $i = 1, \ldots, q$. □

Lemma 12. $\mathsf{Pr}\left(\mathsf{col}'_{0011}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) \leq \frac{t_1q}{2^n-q}$.

Proof. Configuration $\mathsf{col}'_{0011}(\mathcal{Q}_q)$ is the same as $\mathsf{col}_{0011}(\mathcal{Q}_q)$ of Fig. 8 with the additional restriction that $A \neq B$. As a consequence, a winning query cannot occur at positions $(1\mathsf{tl}, 1\mathsf{tr})$. Thus, without loss of generality (by symmetry), a winning query can only occur at $1\mathsf{tl}$ and the proof of Lem. 3 directly carries over. □

Lemma 13. $\mathsf{Pr}\left(\mathsf{col}'_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) \leq \frac{(t_1t_2t_3+2t_1t_2+2t_2)q}{2^n-q}$ *for* $\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}} \in \{0100, 1000\}$.

Proof. The cases are equivalent by symmetry, and we consider $\mathsf{col}'_{0100}(\mathcal{Q}_q)$ only. Configuration $\mathsf{col}'_{0100}(\mathcal{Q}_q)$ is the same as $\mathsf{col}_{0100}(\mathcal{Q}_q)$ of Fig. 9 with the additional restriction that $A_1 \neq B$ and $A_2 \neq B$ (and that only one block cipher is used). For the basic proof idea, we refer to the proof of Lem. 10. Let $i \in \{1, \ldots, q\}$. We assume $\neg\mathsf{help}(\mathcal{Q}_i)$ and analyze the probability the i-th query makes $\mathsf{col}'_{0100}(\mathcal{Q}_i)$ satisfied.

Without loss of generality (by symmetry), the i-th query occurs at the following positions: $1\mathsf{tl}$ only, $1\mathsf{bl}$ only, $1\mathsf{br}$ only, $(1\mathsf{tl}, 1\mathsf{br})$ only, or $(1\mathsf{tl}, 1\mathsf{bl})$ only. For the first four cases, the analysis is identical to Lem. 4. For the remaining case, the analysis for configuration $\mathsf{col}'_{0000}(\mathcal{Q}_q)$ (Lem. 10) can be copied.

The i-th query is successful with probability at most $\frac{t_1t_2t_3+2t_1t_2+2t_2}{2^n-q}$. The claimed bound is obtained by summing over $i = 1, \ldots, q$. □

Lemma 14. $\mathsf{Pr}\left(\mathsf{col}'_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) \leq \frac{(t_1t_2+t_2+t_2^2)q}{2^n-q}$ *for* $\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}} \in \{0101, 1010\}$.

Proof. The cases are equivalent by symmetry, and we consider $\mathsf{col}'_{0101}(\mathcal{Q}_q)$ only. Configuration $\mathsf{col}'_{0101}(\mathcal{Q}_q)$ is the same as $\mathsf{col}_{0101}(\mathcal{Q}_q)$ of Fig. 10 with the additional restriction that only one block cipher is used. For the basic proof idea, we refer to the proof of Lem. 10. Let $i \in \{1, \ldots, q\}$. We assume $\neg\mathsf{help}(\mathcal{Q}_i)$ and analyze the probability the i-th query makes $\mathsf{col}'_{0101}(\mathcal{Q}_i)$ satisfied.

Without loss of generality (by symmetry), the i-th query occurs at the following positions: $1\mathsf{tl}$ only, $1\mathsf{bl}$ only, or $(1\mathsf{tl}, 1\mathsf{bl})$ only. For the first two cases, the analysis is identical to Lem. 5. For the remaining case, the analysis for configuration $\mathsf{col}'_{0000}(\mathcal{Q}_q)$ (Lem. 10) can be copied with Y replaced by Z.

The i-th query is successful with probability at most $\frac{t_1t_2+t_2+t_2^2}{2^n-q}$. The claimed bound is obtained by summing over $i = 1, \ldots, q$. □

Lemma 15. $\Pr\left(\mathsf{col}'_{\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}}}(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) = 0$ *for* $\alpha_{\mathsf{tl}}\alpha_{\mathsf{tr}}\alpha_{\mathsf{bl}}\alpha_{\mathsf{br}} \in \{11**, 1**1, *11*\}$.

Proof. See Lem. 6. □

Lemma 16. $\Pr\left(\mathsf{statecol}'(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) \leq \frac{(t_1+2t_2+t_2^2)q}{2^n-q}$.

Proof. We consider configuration $\mathsf{statecol}'(\mathcal{Q}_q)$ of Fig. 14. The proof idea is the same as the proof of Lem. 1. Let $i \in \{1,\dots,q\}$. As in Lem. 1, we assume $\neg\mathsf{help}(\mathcal{Q}_i)$ and analyze the probability the i-th query makes $\mathsf{statecol}'(\mathcal{Q}_i)$ satisfied.

Recall that the positions in Fig. 14 are simply referred to as $\mathsf{tl}, \mathsf{tr}, \mathsf{bl}, \mathsf{br}$, without a leading 1. Without loss of generality (by symmetry), the i-th query occurs at the following positions: tl only, bl only, $(\mathsf{tl}, \mathsf{br})$ only, or $(\mathsf{tl}, \mathsf{bl})$ only. Note that, as $A \neq B$, it can impossibly occur at $(\mathsf{tl}, \mathsf{tr})$ or $(\mathsf{bl}, \mathsf{br})$.

Query occurs at tl but not at $(\mathsf{tr}, \mathsf{bl}, \mathsf{br})$. By $\neg\mathsf{help}_1(\mathcal{Q}_i)$, there are $\leq t_1$ choices for $(\mathsf{bl}, \mathsf{br})$. For any of these $\leq t_1$ choices, let K_{bl} and K_{br} be the key inputs corresponding to positions bl and br. The i-th query is successful only if its XOR-output equals $K_{\mathsf{br}}^l \| K_{\mathsf{bl}}^r$, which happens with probability at most $\frac{1}{2^n-q}$. The total success probability is at most $\frac{t_1}{2^n-q}$.

Query occurs at bl but not at $(\mathsf{tl}, \mathsf{tr}, \mathsf{br})$. Let K be the key input for the i-th query. By $\neg\mathsf{help}_2(\mathcal{Q}_i)$ and $\neg\mathsf{help}_3(\mathcal{Q}_i)$, there are $\leq t_2$ choices for tl and $\leq t_2$ choices for tr. For any of these $\leq t_2^2$ choices, the query at position br is uniquely determined, and so is the XOR-output Z of br. The i-th query is successful only if its XOR-output equals this value Z, which happens with probability at most $\frac{1}{2^n-q}$. The total success probability is at most $\frac{t_2^2}{2^n-q}$.

Query occurs at $(\mathsf{tl}, \mathsf{br})$ but not at $(\mathsf{tr}, \mathsf{bl})$. Let K be the key input for the i-th query. As the query occurs at both positions, we require $K^l = Z^l$, which fixes Z^l. By $\neg\mathsf{help}_2(\mathcal{Q}_i)$, there are $\leq t_2$ choices for bl. For any of these $\leq t_2$ choices bl, we obtain a different Z. The i-th query is successful only if its XOR-output equals this value Z, which happens with probability at most $\frac{1}{2^n-q}$. The total success probability is at most $\frac{t_2}{2^n-q}$.

Query occurs at $(\mathsf{tl}, \mathsf{bl})$ but not at $(\mathsf{tr}, \mathsf{br})$. Let K be the key input for the i-th query. As the query occurs at both positions, we require $K^r = Z^r$, which fixes Z^r. By $\neg\mathsf{help}_3(\mathcal{Q}_i)$, there are $\leq t_2$ choices for br. For any of these $\leq t_2$ choices br, we obtain a different Z. The i-th query is successful only if its XOR-output equals this value Z, which happens with probability at most $\frac{1}{2^n-q}$. The total success probability is at most $\frac{t_2}{2^n-q}$.

The i-th query is successful with probability at most $\frac{t_1+2t_2+t_2^2}{2^n-q}$. The claimed bound is obtained by summing over $i = 1,\dots,q$. □

We are ready to finish the proof of Thm. 3. From (18) and Lems. 7, 10-15, and 16 we find:

$$\mathbf{adv}^{\mathrm{col}}_{\text{MDC-4}}(q) \leq \frac{(t_1+11t_1t_2+3t_1t_2t_3+12t_2+4t_2^2)q}{2^n-q} + \frac{(t_1+2t_2+t_2^2)q}{2^n-q} + \frac{q^2}{t_1(2^n-q)} + 2\cdot 2^{n/2}\left(\frac{eq2^{n/2}}{t_2(2^n-q)}\right)^{t_2} + 2^n\left(\frac{eq}{t_3(2^n-q)}\right)^{t_3},$$

where $t_1, t_2, t_3 > 0$ are integral. The result of Thm. 3 is obtained by observing that $2^n - q > 2^{n-1}$ for $q < 2^{n-1}$.

D Appendix to Sect. 5.2: Proof of Thm. 4

The proof of Thm. 4 follows the analysis of Thm. 2 in Sect. 4.1. We consider any adversary making q queries to its oracle $E = E_1 = E_2$, which tries to find a preimage for $f_{\text{MDC-4}}$. Let $(Y, Z) \in \mathbb{Z}_2^{2n}$ with $Y \neq Z$ be the point to invert, chosen by the adversary prior to making any query. Finding a preimage for (Y, Z) corresponds to obtaining a query history $\mathcal{Q}_q$ of size q that satisfies configuration $\mathsf{pre}'(\mathcal{Q}_q)$ of Fig. 15. In other words,

$$\mathbf{adv}^{\mathrm{epre}(\neq)}_{f_{\text{MDC-4}}}(q) = \Pr\left(\mathsf{pre}'(\mathcal{Q}_q)\right), \tag{19}$$

and we consider the probability of obtaining any query history $\mathcal{Q}_q$ that satisfies configuration $\mathsf{pre}'(\mathcal{Q}_q)$. Note that $\mathsf{pre}'(\mathcal{Q}_q)$ differs from $\mathsf{pre}(\mathcal{Q}_q)$ of Fig. 6 only in the fact that $Y \neq Z$ (and that the function employs one block cipher rather than two). We use the same convention for the figures as is used in Sect. 4.1.

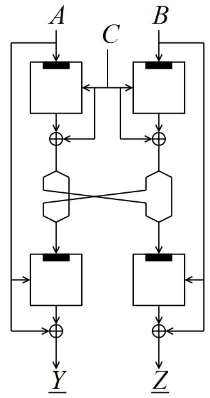


Fig. 15. Configuration $\mathsf{pre}'(\mathcal{Q})$. We have $Y \neq Z$.

As before, we use the notion of free super queries, and we refer to Sect. 4.1 for a formalization of them. We employ the helping event $\mathsf{help}(\mathcal{Q}_q)$ from Sect. 4.1. We obtain for (19):

$$\mathsf{Pr}\left(\mathsf{pre}'(\mathcal{Q}_q)\right) \leq \mathsf{Pr}\left(\mathsf{pre}'(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) + \mathsf{Pr}\left(\mathsf{help}(\mathcal{Q}_q)\right). \tag{20}$$

Probability $\mathsf{Pr}\left(\mathsf{help}(\mathcal{Q}_q)\right)$ is already bounded in Lem. 9 of Sect. 4.1. In Lem. 17, we bound probability $\mathsf{Pr}\left(\mathsf{pre}'(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right)$.

Lemma 17. $\mathsf{Pr}\left(\mathsf{pre}'(\mathcal{Q}_q) \wedge \neg\mathsf{help}(\mathcal{Q}_q)\right) \leq \frac{4t_2^3+4t_1t_2+20+4\cdot 2^{n/2}}{2^n} + \frac{16t_1t_2+24t_1t_2 2^{n/2}}{2^{2n}} + \frac{64q}{2^{3n}}$.

Proof. The proof follows the analysis of Lem. 8. We make the following distinction:

1. The winning query contributes to exactly one position of configuration $\mathsf{pre}'(\mathcal{Q}_q)$;
2. The winning query contributes to exactly two positions of configuration $\mathsf{pre}'(\mathcal{Q}_q)$;
3. The winning query contributes to exactly three positions of configuration $\mathsf{pre}'(\mathcal{Q}_q)$.

Note that a winning query cannot occur at all four positions: if this would be the case, we would have $A = B$ and thus $Y = Z$. In particular in the remainder of the proof we will use that a winning *normal* query cannot occur at positions $(\mathsf{bl}, \mathsf{br})$, and a winning query (*normal or super*) cannot contribute at positions $(\mathsf{tl}, \mathsf{tr})$.

Case 1. The analysis of Lem. 8 carries over; a preimage is found with probability at most $\frac{4t_2^3+4t_1t_2}{2^n}$.

Case 2. We make the following distinction, and consider the two sub-cases separately.

1. The contributed queries are different for both positions;
2. The contributed queries are the same for both positions.

Case 2.1. We follow the analysis of case 2.1 for Lem. 8 (App. B). In this case, the winning query must be a super query. As a super query cannot contribute to $(\mathsf{tl}, \mathsf{tr})$, it can only contribute to positions $(\mathsf{tl}, \mathsf{br})$, $(\mathsf{tr}, \mathsf{bl})$, $(\mathsf{tl}, \mathsf{bl})$, $(\mathsf{tr}, \mathsf{br})$, or $(\mathsf{bl}, \mathsf{br})$. The first two cases are already covered in the two wish lists $\mathcal{W}_1, \mathcal{W}_2$ of Lem. 8, with the minor exception that the block cipher indices are redundant. We introduce three extra wish lists for queries contributing in the latter three cases. Note that if a super query contributes to positions $(\mathsf{tl}, \mathsf{bl})$, the key input to bl equals the key input to tl which equals the message input to bl (similar for super queries contributing to $(\mathsf{tr}, \mathsf{br})$). Also, note that if a super query contributes to positions $(\mathsf{bl}, \mathsf{br})$, the XOR-outputs of tl and tr must be the same. If a query is made by the adversary, the wish lists are updated according to the following requirements:

– If the query fits $\mathsf{pre}'_3(\mathcal{Q})$ of Fig. 16 for any query in the query history, the corresponding tuple $(X\|W, C, (V\|W) \oplus C, X\|W, Y \oplus (X\|W))$ is added to $\mathcal{W}_3$;
– If the query fits $\mathsf{pre}'_4(\mathcal{Q})$ of Fig. 16 for any query in the query history, the corresponding tuple $(X\|W, C, (V\|W) \oplus C, X\|W, Z \oplus (X\|W))$ is added to $\mathcal{W}_4$;
– If the query fits $\mathsf{pre}'_5(\mathcal{Q})$ of Fig. 16 for any query in the query history, the corresponding tuple $(W\|X, A, Y \oplus A, B, Z \oplus B)$ is added to $\mathcal{W}_5$;

Of these tuples, the first element identifies the key for which the super query is made, the second and third element define the input and output of the cipher in the top row (either left or right), and the fourth and fifth element define the input and output of the cipher in the bottom row (either right or left). For $\mathcal{W}_5$, the second and third element correspond to bl and the fourth and fifth element to br. Because each element of the wish lists can be wished for only once, the adversary finds a preimage with probability at most

$$\frac{|\mathcal{W}_1| + |\mathcal{W}_2| + |\mathcal{W}_3| + |\mathcal{W}_4| + |\mathcal{W}_5|}{2^{n-1}(2^{n-1} - 1)},$$

where $\mathcal{W}_1, \mathcal{W}_2$ are already bounded in Lem. 8. It remains to bound the sizes of $\mathcal{W}_3, \mathcal{W}_4, \mathcal{W}_5$ after q queries. Configuration $\mathsf{pre}'_3(\mathcal{Q}_q)$ has $\leq t_2$ solutions for br (by $\neg\mathsf{help}_3(\mathcal{Q}_q)$), and at most $\leq t_1$ solutions for tr (by $\neg\mathsf{help}_2(\mathcal{Q}_q)$). Additionally, there are $2^{n/2}$ possibilities for W. Thus $|\mathcal{W}_3| \leq t_1 t_2 2^{n/2}$ and similarly $|\mathcal{W}_4| \leq t_1 t_2 2^{n/2}$. Configuration $\mathsf{pre}'_5(\mathcal{Q}_q)$ has $2^{n/2}$ choices for W, for any of these choices it has $\leq t_1$ solutions for tr (by $\neg\mathsf{help}_1(\mathcal{Q}_q)$), and consequently $\leq t_2$ solutions for tl (by $\neg\mathsf{help}_3(\mathcal{Q}_q)$). Thus also $|\mathcal{W}_5| \leq t_1 t_2 2^{n/2}$. Hence, in this case a preimage is found with probability at most $\frac{16 t_1 t_2 + 24 t_1 t_2 2^{n/2}}{2^{2n}}$.

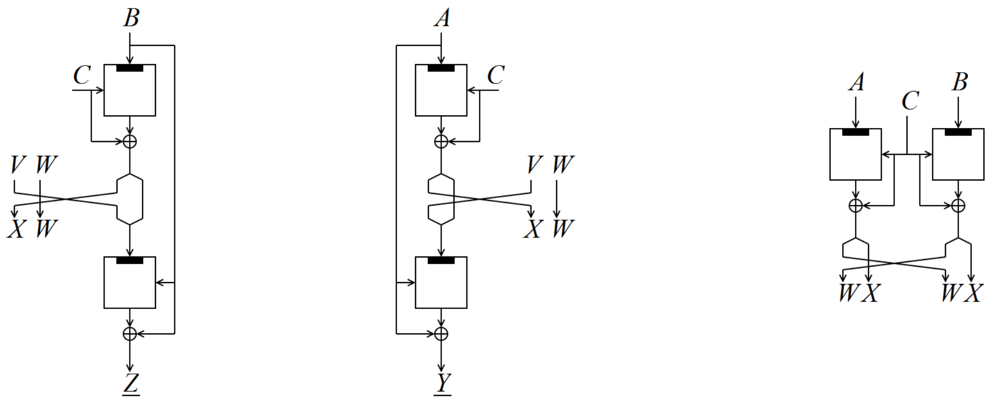


Fig. 16. From left to right: configurations $\mathsf{pre}'_3(\mathcal{Q})$, $\mathsf{pre}'_4(\mathcal{Q})$, and $\mathsf{pre}'_5(\mathcal{Q})$.

Case 2.2. We follow the analysis of case 2.2 for Lem. 8 (App. B). In this case, the winning query may be a normal query or a super query. The winning query can only contribute to positions ($\mathsf{tl} = \mathsf{br}$), ($\mathsf{tr} = \mathsf{bl}$), ($\mathsf{tl} = \mathsf{bl}$), or ($\mathsf{tr} = \mathsf{br}$). The first two cases are already covered in Lem. 8. Consider a query contributing to $\mathsf{tl} = \mathsf{bl}$. By construction, this query must be of the form $E(K, K) = y$ where $K \oplus y = Y$ and $K^r = Y^r$. As Y is fixed, the adversary finds such query with probability at most $\frac{2^{n/2}}{2^{n-1}} = \frac{2 \cdot 2^{n/2}}{2^n}$ (either in case of forward or inverse query). The same probability bound is obtained for winning queries to appear at $(\mathsf{tr}, \mathsf{br})$. Consequently, a preimage is found in this case with probability at most $\frac{8 + 4 \cdot 2^{n/2}}{2^n}$.

Case 3. Recall that a query can never contribute to $(\mathsf{tl}, \mathsf{tr})$ at the same time. Therefore, we only need to consider queries contributing at positions $(\mathsf{tl}, \mathsf{bl}, \mathsf{br})$ or $(\mathsf{tr}, \mathsf{bl}, \mathsf{br})$. We make the following distinction, and consider the two sub-cases separately.

1. The contributed queries are different for all positions;
2. The contributed queries are the same at two positions.

Note that as the queries at $(\mathsf{bl}, \mathsf{br})$ cannot be the same, there is no need to consider the case all three queries are the same.

Case 3.1. As before, we consider two wish lists $\mathcal{W}_{\mathsf{tl}}$, $\mathcal{W}_{\mathsf{tr}}$, corresponding to position to which the winning query does not contribute. Note that if a super query contributes to positions $(\mathsf{tr}, \mathsf{bl}, \mathsf{br})$, the XOR-outputs of tl and tr must be the same, and equal to the key input to tr. In order words, any query to tl fixes exactly one wish list tuple in $\mathcal{W}_{\mathsf{tl}}$. In more detail, if a query $E(K, x) = y$ is made the wish lists are updated as follows:

- The tuple $(x \oplus y, x, y, K, Y \oplus K, x \oplus y, Z \oplus x \oplus y)$ is added to $\mathcal{W}_{\mathsf{tl}}$;
- The tuple $(x \oplus y, x, y, x \oplus y, Y \oplus x \oplus y, K, Z \oplus K)$ is added to $\mathcal{W}_{\mathsf{tr}}$.

Of these tuples, the first element identifies the key, the second and third element define the input and output of the cipher in the top row (either left or right), the fourth and fifth element define the input and output of the cipher at bl and the sixth and seventh element the input and output of the cipher at br. Because each element of the wish lists can be wished for only once, the adversary finds a preimage with probability at most

$$\frac{|\mathcal{W}_{\mathsf{tl}}| + |\mathcal{W}_{\mathsf{tr}}|}{2^{n-1}(2^{n-1} - 1)(2^{n-1} - 2)}.$$

Clearly, $|\mathcal{W}_{\mathsf{tl}}|, |\mathcal{W}_{\mathsf{tr}}| \leq q$. Hence, in this case a preimage is found with probability at most $\frac{64q}{2^{3n}}$.
Case 3.2. Note that the same query can only occur at positions $(\mathsf{tl} = \mathsf{br})$, $(\mathsf{tr} = \mathsf{bl})$, $(\mathsf{tl} = \mathsf{bl})$, or $(\mathsf{tr} = \mathsf{br})$. The analysis of case 2.2 carries over directly, but for the latter two scenarios we can do better. Consider a query contributing to $\mathsf{tl} = \mathsf{bl}$. Note that, as the super query also contributes to position br, the key inputs to $\mathsf{tl}, \mathsf{bl}, \mathsf{br}$ are the same. By construction, the query at $\mathsf{tl} = \mathsf{bl}$ must be of the form $E(K, K) = y$ where $K \oplus y = Y$ and $K = Y$. As Y is fixed, the adversary finds such query with probability at most $\frac{1}{2^{n-1}} = \frac{2}{2^n}$ (either in case of forward or inverse query). The same probability bound is obtained for winning queries to appear at $(\mathsf{tr} = \mathsf{br})$. Consequently, a preimage is found in this case with probability at most $\frac{12}{2^n}$.

The claim is obtained by summing the bounds obtained for the three cases. □

The proof of Thm. 4 is finished by adding the bounds of Lems. 9 and 17, as set forth in (19-20).