On the Security of Tandem-DM

Full version of the paper (2009/02/03)

Ewan Fleischmann, Michael Gorski, Stefan Lucks
{ewan.fleischmann,michael.gorski,stefan.lucks}@uni-weimar.de

Bauhaus-University Weimar, Germany

Abstract. We provide the first proof of security for Tandem-DM, one of the oldest and most well-known constructions for turning a blockcipher with n-bit blocklength and $2n$-bit keylength into a $2n$-bit cryptographic hash function. We prove, that when Tandem-DM is instantiated with AES-256, *i.e.* blocklength 128 bits and keylength 256 bits, any adversary that asks less than $2^{120.4}$ queries cannot find a collision with success probability greater than 1/2. We also prove a bound for preimage resistance of Tandem-DM. Interestingly, as there is only one practical construction known (FSE'06, Hirose) turning such an $(n, 2n)$-bit blockcipher into a $2n$-bit compression function that has provably birthday-type collision resistance, Tandem-DM is one out of two structures that possess this desirable feature.

Keywords: Cryptographic hash function, blockcipher based, proof of security, double-block length, ideal cipher model, Tandem-DM.

1 Introduction

A cryptographic hash function is a function which maps an input of arbitrary length to an output of fixed length. It should satisfy at least collision-, preimage- and second-preimage resistance and is is one of the most important primitives in cryptography [23].

Blockcipher-Based Hash Functions. Since their initial design by Rivest, MD4-family hash functions (*e.g.* MD4, MD5, RIPEMD, SHA-1, SHA2 [26, 27, 29, 30]) have dominated cryptographic practice. But in recent years, a sequence of attacks on these type of functions [7, 11, 36, 37] has led to a generalized sense of concern about the MD4-approach. The most natural place to look for an alternative is in blockcipher-based constructions, which in fact predate the MD4-approach [22]. Another reason for the resurgence of interest in blockcipher-based hash functions is due to the rise of size restricted devices such as RFID tags or smart cards: A hardware designer has to implement only a blockcipher in order to obtain an encryption function as well as a hash function. But since the output length of most practical encryption functions is far too short for a collision resistant hash function, *e.g.* 128-bit for AES, one is mainly interested in sound design principles for *double block length* (DBL) hash functions [2]. A DBL hash-function uses a blockcipher with n-bit output as the building block by which it maps possibly long strings to $2n$-bit ones.

Our Contribution. Four 'classical' DBL hash functions are known: MDC-2, MDC-4, Abreast-DM and Tandem-DM [3, 4, 21]. At EUROCRYPT'07, Steinberger [34] proved the first security bound for the hash function MDC-2: assuming a hash output length of 256 bits, any adversary asking less than $2^{74.9}$ queries cannot find a collision with probability greater than 1/2.

In this paper, we prove the first security bound for the compression function Tandem-DM in terms of collision resistance and preimage resistance. We will give an upper bound for success if an adversary is trying to find a collision. By assuming a hash output length of 256 bits, any adversary asking less than $2^{120.4}$ queries cannot find a collision with probability greater than 1/2. We will also prove an upper bound for success if an adversary is trying to find a preimage. This bound is rather weak as it essentially only states, that the success probability of an adversary asking strictly less than 2^n queries is asymptotically negligible.

Beyond providing such a proof of security for TANDEM-DM in the first place, our result even delivers one of the most secure rate 1/2 DBL compression functions known. The first *practical* DBL compression function with rate 1/2 (without bit-fixing and other artificial procedures like employing two different blockciphers) that has a birthday-type security guarantee was presented at FSE'06 by Hirose [14]. He essentially states (see Appendix B for a proof), that no adversary asking less than $2^{124.55}$ queries, again for $2n = 256$, can find a collision with probability greater then 1/2. These two compression functions (Hirose's FSE '06 proposal and TANDEM-DM) are the only rate 1/2 practical compression functions that are known to have a birthday-type security guarantee.

Outline. The paper is organized as follows: Section 2 includes formal notations and definitions as well as a review of related work. In Section 3, we proof that an adversary asking less than $2^{120.4}$ oracle queries has negligible advantage in finding a collision for the TANDEM-DM compression function. A bound for preimage resistance of TANDEM-DM is given in Section 4. In Section 5 we discuss our results and conclude the paper.

2 Preliminaries

2.1 Iterated DBL Hash Function Based on Blockciphers

Ideal Cipher Model. An (n, k)-bit blockcipher is a keyed family of permutations consisting of two paired algorithms $E : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n$ and $E^{-1} : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n$ both accepting a key of size k bits and an input block of size n bits. For simplicity, we will call it an (n, k)-blockcipher. Let $\mathrm{BC}(n, k)$ be the set of all (n, k)-blockciphers. Now, for any one fixed key $K \in \{0,1\}^k$, decryption $E_K^{-1} = E^{-1}(\cdot, K)$ is the inverse function of encryption $E_K = E(\cdot, K)$, so that $E_K^{-1}(E_K(x)) = x$ holds for any input $X \in \{0,1\}^n$.

Most of the attacks on hash functions based on block ciphers do not utilize the internal structure of the blockciphers. The security of such hash functions is usually analyzed in the *ideal cipher model* [2, 9, 18]. In the ideal cipher model the underlying primitive, the blockcipher E, is modeled as a family of random permutations $\{E_k\}$ whereas the random permutations are chosen independently for each key K, i.e. formally E is selected randomly from $\mathrm{BC}(n, k)$.

DBL Compression Functions. Iterated DBL hash functions with two blockcipher calls in their compression function are discussed in this article. A hash function $H : \{0,1\}^* \to \{0,1\}^{2n}$ can be built by iterating a compression function $F : \{0,1\}^{3n} \to \{0,1\}^{2n}$ as follows: Split the padded message M into n-bit blocks $M_1, \ldots, M_l$, fix (G_0, H_0), apply $(G_i, H_i) = F(G_{i-1}, H_{i-1}, M_i)$ for $i = 1, \ldots, l$ and finally set $H(M) := (G_l, H_l)$. Let the compression function F be such that

$$(G_i, H_i) = F(G_{i-1}, H_{i-1}, M_i),$$

where $G_{i-1}, H_{i-1}, G_i, H_i, M_i \in \{0,1\}^n$. We assume that the compression function F consists of F_T, the top row, and F_B, the bottom row. We explicitly allow the results of F_T to be fed into the calculation of F_B. Each of the component functions F_B and F_T performs exactly *one* call to the blockcipher and can be defined as follows:

$$\begin{aligned} G_i = \quad F_T(G_{i-1}, H_{i-1}, M_i) \quad &= E(X_T, K_T) \oplus Z_T, \\ H_i = F_B(G_i, G_{i-1}, H_{i-1}, M_i) &= E(X_B, K_B) \oplus Z_B, \end{aligned}$$

where X_T, K_T, Z_T are uniquely determined by G_{i-1}, H_{i-1}, M_i and X_B, K_B, Z_B are uniquely determined by $G_i, G_{i-1}, H_{i-1}, M_i$.
We define the rate r of a blockcipher based compression/hash function F by

$$r = \frac{|M_i|}{(\text{number of block cipher calls in F}) \times n}.$$

It is a measure of efficiency for such blockcipher based constructions.

2.2 The Tandem-DM Compression Function

The Tandem-DM compression function was proposed at EUROCRYPT'92 by Lai and Massey [21]. It uses two cascaded Davies-Meyer [2] schemes. The compression function is illustrated in Figure 1 and is formally defined in Definition 1.

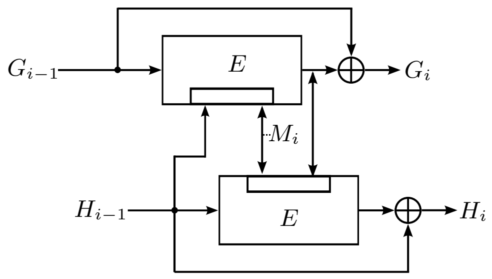

Figure 1. The compression function Tandem-DM F^{TDM} where E is an $(n, 2n)$-blockcipher, the small rectangle inside the cipher rectangle indicates which input is used as key

Definition 1. *Let $F^{TDM} : \{0,1\}^{2n} \times \{0,1\}^n \to \{0,1\}^{2n}$ be a compression function such that $(G_i, H_i) = F^{TDM}(G_{i-1}, H_{i-1}, M_i)$ where $G_i, H_i, M_i \in \{0,1\}^n$. F^{TDM} is built upon an $(n, 2n)$-blockcipher E as follows:*

$$\begin{aligned} W_i &= E(G_{i-1}, H_{i-1}|M_i) \\ G_i &= \mathrm{F}_T(G_{i-1}, H_{i-1}, M_i) = W_i \oplus G_{i-1} \\ H_i &= \mathrm{F}_B(G_{i-1}, H_{i-1}, M_i) = E(H_{i-1}, M_i|W_i) \oplus H_{i-1}. \end{aligned}$$

2.3 Related Work

Our work is largely motivated by Steinberger [34] in order to provide rigorous proofs for well-known blockcipher based hash functions. As is reviewed in the following, there are many papers on hash functions composed of blockciphers.

Schemes with non-optimal or unknown collision resistance. Preneel *et al.* [28] discussed the security of SBL hash functions against several generic attacks. They concluded that 12 out of 64 hash functions are secure against the attacks. However, formal proofs were first given by Black *et al.* [2] about 10 years later. Their most important result is that 20 hash functions – including the 12 mentioned above – are optimally collision resistant. Knudsen *et al.* [19] discussed the insecurity of DBL hash functions with rate 1 composed of (n, n)-blockciphers. Hohl *et al.* [15] analyzed the security of DBL compression functions with rate 1 and 1/2. Satoh *et al.* [33] and Hattoris *et al.* [12] discussed DBL hash functions with rate 1 composed of $(n, 2n)$-blockciphers. MDC-2 and MDC-4 [16, 1, 4] are (n, n)-blockcipher based DBL hash functions with rates 1/2 and 1/4, respectively. Steinberger [34] proved that for MDC-2 instantiated with, *e.g.*, AES-128 no adversary asking less than $2^{74.9}$ can usually find a collision. Nandi *et al.* [25] proposed a construction with rate 2/3 but it is not optimally collision resistant. Furthermore, Knudsen and Muller [20] presented some attacks against it. Gauravaram *et al.* [10] proposed a new approach based on iterated halving to design a hash function with a blockcipher. At EUROCRYPT'08 and CRYPTO'08, Steinberger [31, 32] proved some security bounds for fixed-key (n, n)-blockcipher based hash functions, *i.e.* permutation based hash functions, that all have small rates and low security guarantees. None of these schemes/techniques mentioned so far are known to have birthday-type collision resistance.

Schemes with Birthday-Type Collision Resistance. Merkle [24] presented three DBL hash functions composed of DES with rates of at most 0.276. They are optimally collision resistant in the ideal cipher model. Hirose [13] presented a class of DBL hash functions with rate 1/2 which are composed of two different and independent $(n, 2n)$ blockciphers that have birthday-type collision resistance. At FSE'06, Hirose [14] presented a rate 1/2 and $(n, 2n)$ blockcipher based DBL hash function that has birthday-type collision resistance. As he only stated the proof for the hash function, we have given the proof for his compression function in Appendix B.

3 Collision Resistance

In this section we will discuss the collision resistance of the compression function Tandem-DM.

3.1 Defining Security – Collision Resistance of a Compression Function

Insecurity is quantified by the success probability of an optimal resource-bounded adversary. The resource is the number of queries to the ideal cipher oracles E or E^{-1}. For a set S, let $z \overset{R}{\leftarrow} S$ represent random sampling from S under the uniform distribution. For a probabilistic algorithm $\mathcal{M}$, let $z \overset{R}{\leftarrow} \mathcal{M}$ mean that z is an output of $\mathcal{M}$ and its distribution is based on the random choices of $\mathcal{M}$.

An adversary is a computationally unbounded but always-halting collision-finding algorithm $\mathcal{A}$ with access to an oracle $E \in \mathrm{BC}(n, k)$. We can assume (by standard arguments) that $\mathcal{A}$ is deterministic. The adversary may make a *forward* query $(K, X)_{fwd}$ to discover the corresponding value $Y = E_K(X)$, or the adversary may make a *backward* query $(K, Y)_{bwd}$, so as to learn the corresponding value $X = E_K^{-1}(Y)$ for which $E_K(X) = Y$. Either way the result of the query is stored in a triple (X_i, K_i, Y_i) and the *query history*, denoted $\mathcal{Q}$, is the tuple $(Q_1, \ldots, Q_q)$ where $Q_i = (X_i, K_i, Y_i)$ is the result of the i-th query made by the adversary and where q is the total number of queries made by the adversary. The value $X_i \oplus Y_i$ is called 'XOR'-output of the query.

Without loss of generality, it is assumed that $\mathcal{A}$ asks at most only once on a triplet of a key K_i, a plaintext X_i and a ciphertext Y_i obtained by a query and the corresponding reply.

The adversary's goal is to output two different triplets (G, H, M) and (G', H', M') such that $F(G, H, M) = F(G', H', M')$. We impose the reasonable condition that the adversary must have made all queries necessary to compute $F(G, H, M)$ and $F(G', H', M')$. We will in fact dispense the adversary from having to output these two triplets, and simply determine whether the adversary has been successful or not by examining its query history $\mathcal{Q}$. Formally, we say that $\textsc{Coll}(\mathcal{Q})$ holds if there is such a collision and $\mathcal{Q}$ contains all the queries necessary to compute it.

Definition 2. ***(Collision resistance of a compression function)*** *Let F be a blockcipher based compression function, $F : \{0,1\}^{3n} \rightarrow \{0,1\}^{2n}$. Fix an adversary $\mathcal{A}$. Then the advantage of $\mathcal{A}$ in finding collisions in F is the real number*

$$\begin{aligned}\mathbf{Adv}_F^{\textsc{Coll}}(\mathcal{A}) = \Pr[E \overset{R}{\leftarrow} \mathrm{BC}(n,k); ((G,H,M),(G',H',M')) \overset{R}{\leftarrow} \mathcal{A}^{E,E^{-1}} :\\ ((G,H,M) \neq (G',H',M')) \wedge F(G,H,M) = F(G',H',M')].\end{aligned}$$

For $q \geq 1$ we write

$$\mathbf{Adv}_F^{\textsc{Coll}}(q) = \max_{\mathcal{A}}\{\mathbf{Adv}_F^{\textsc{Coll}}(\mathcal{A})\}$$

where the maximum is taken over all adversaries that ask at most q oracle queries (*i.e.* E and E^{-1} queries).

3.2 Security Results

Our discussion will result in a proof for the following upper bound:

Theorem 1. *Let $F := F^{TDM}$ as in Definition 1 and n, q be natural numbers with $q < 2^n$. Let $N' = 2^n - q$ and let α be any positive number with $eq/N' \leq \alpha$ and $\tau = \alpha N'/q$ (and e^x being the exponential function). Then*

$$\mathbf{Adv}_F^{\textsc{Coll}}(q) \leq q2^n e^{q\tau(1-\ln\tau)/N'} + 4q\alpha/N' + 6q/(N')^2 + 2q/(N')^3.$$

The proof is given on page 11 and is a simple corollary of the discussion and lemmas below. As this theorem is rather incomprehensible, we will investigate what this theorem means for AES-256. The bound obtained by this theorem depends an a parameter α. We do not require any specific value α as any α (meeting to the conditions mentioned in Theorem 1) leaves us with a correct bound. For Theorem 1 to give a *good* bound one must choose a suitable value for the parameter α. Choosing large values of α reduces the value of the first term but increases the value of the second term. There seems to be no good closed for for α as these will change with every q. The meaning of α will be explained in the proof. We will optimize the parameter α numerically as given in the following corollary.

Corollary 1. *For the compression function* Tandem-DM*, instantiated with AES-*256*, any adversary asking less than $2^{120.4}$ (backward or forward) oracle queries cannot usually find a collision. In this case, $\alpha = 24.0$.*

3.3 Proof of Theorem 1

Analysis Overview. We will analyze if the queries made by the adversary contain the means of constructing a collision for the compression function F^{TDM}. Effectively we look to see whether there exist four queries that form a collision (see Figure 2).

To upper bound the probability of the adversary obtaining queries than can be used to construct a collision, we upper bound the probability of the adversary making a query that can be used as the final query to complete such a collision. Namely for each i, $1 \leq i \leq q$, we upper bound the probability that the answer to the adversary's i-th query $(K_i, X_i)_{fwd}$ or $(K_i, Y_i)_{bwd}$ will allow the adversary to use the i-th query to complete the collision. In the latter case, we say that the i-th query is 'successful' and we give the attack to the adversary.

As the probability depends naturally on the first $i-1$ queries, we need to make sure that the adversary hasn't already been too lucky with these (or else the probability of the i-th query being successful would be hard to upper bound). Concretely, being lucky means, that there exists a large subset of the first $i-1$ queries that all have the same XOR output (see below for a formal definition). Our upper bound thus breaks down into two pieces: an upper bound for the probability of the adversary getting lucky in one defined specific way and the probability of the adversary ever making a successful i-th query, conditioned on the fact that the adversary has not yet become lucky by its $(i-1)$-th query.

Analysis Details. Fix numbers n, q and an adversary $\mathcal{A}$ asking q queries to its oracle. We upper bound $\Pr[\textsc{Coll}^{TDM}(\mathcal{Q})]$ by exhibiting predicates $\textsc{Lucky}(\mathcal{Q})$, $\textsc{Win1}(\mathcal{Q})$, $\textsc{Win2}(\mathcal{Q})$ and $\textsc{Win3}(\mathcal{Q})$ such that $\textsc{Coll}^{TDM}(\mathcal{Q}) \Rightarrow \textsc{Lucky}(\mathcal{Q}) \vee \textsc{Win1}(\mathcal{Q}) \vee \textsc{Win2}(\mathcal{Q}) \vee \textsc{Win3}(\mathcal{Q})$ and then by upper bounding separately the probabilities $\Pr[\textsc{Lucky}(\mathcal{Q})]$, $\Pr[\textsc{Win1}(\mathcal{Q})]$, $\Pr[\textsc{Win2}(\mathcal{Q})]$ and $\Pr[\textsc{Win3}(\mathcal{Q})]$. Then obviously $\Pr[\textsc{Coll}(\mathcal{Q})] \leq \Pr[\textsc{Lucky}(\mathcal{Q})] + \Pr[\textsc{Win1}(\mathcal{Q})] + \Pr[\textsc{Win2}(\mathcal{Q})] + \Pr[\textsc{Win3}(\mathcal{Q})]$. The event $\textsc{Lucky}(\mathcal{Q})$ happens if the adversary is lucky, whereas if the adversary is not lucky but makes a successful i-th query then one of the other predicates hold.

To state the predicates, we need one additional definition. Let $a(\mathcal{Q})$ be a function defined on query sequences of length q as follows:

$$a(\mathcal{Q}) = \max_{Z \in \{0,1\}^n} |\{i : X_i \oplus Y_i = Z\}| \text{ is the maximum size of a set of queries}$$
$$\text{in } \mathcal{Q} \text{ whose XOR outputs are all the same.}$$

The event $\textsc{Lucky}(\mathcal{Q})$ is now defined by

$$\textsc{Lucky}(\mathcal{Q}) = a(\mathcal{Q}) > \alpha,$$

where α is the constant from Theorem 1 (it is chosen depending on n and q by a numerical optimization process). Thus as α is chosen larger $\Pr[\textsc{Lucky}(\mathcal{Q})]$ diminishes. The other events, $\textsc{Win1}(\mathcal{Q})$, $\textsc{Win2}(\mathcal{Q})$ and $\textsc{Win3}(\mathcal{Q})$ are different in nature from the event $\textsc{Lucky}(\mathcal{Q})$. Simply put, they consider mutually exclusive configurations on how to find a collision for TANDEM-DM (see Figure 2 for an overview).

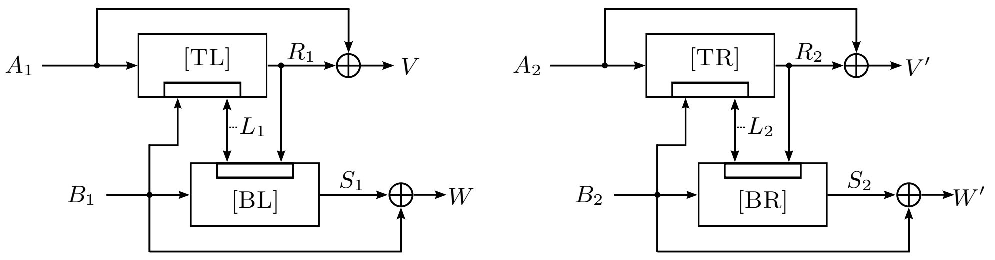

Figure 2. Generic configuration for a collision if ($V = V'$, $W = W'$) for the TANDEM-DM compression function.

We will call the configuration necessary for, e.g., predicate WIN1$a(\mathcal{Q})$ simply 1a. Now, take for example just this configuration of predicate WIN1$a(\mathcal{Q})$ (i.e. all four queries are different and a collision is found; it will be defined in Definition 3). Formally, the four queries $Q_i, Q_j, Q_k, Q_l \in \mathcal{Q}$ fit configuration 1a if and only if

$$\begin{aligned}&(i \neq j) \wedge (i \neq k) \wedge (i \neq l) \wedge (j \neq k) \wedge (j \neq l) \wedge (k \neq l) \wedge \\ &(X_i \oplus Y_i = X_k \oplus Y_k) \wedge (X_j \oplus Y_j = X_l \oplus Y_l) \wedge \\ &(K_i = X_j | K_j^{(1 \ldots n/2)}) \wedge (K_j = K_i^{(n/2+1 \ldots n)} | Y_i) \wedge \\ &(K_k = X_l | K_k^{(1 \ldots n/2)}) \wedge (K_l = K_k^{(n/2+1 \ldots n)} | Y_k).\end{aligned}$$

We say, that FIT1$a(\mathcal{Q})$ holds if there exist $i, j, k, l \in \{1, 2, \ldots, q\}$ such that queries Q_i, Q_j, Q_k, Q_l fit configuration 1a. The other predicates FIT1$b(\mathcal{Q})$, FIT1$c(\mathcal{Q})$, FIT1$d(\mathcal{Q})$, FIT2$a(\mathcal{Q}), \ldots,$ FIT2$d(\mathcal{Q})$ and FIT3$a(\mathcal{Q}), \ldots$ FIT3$d(\mathcal{Q})$ whose configurations are given in Definition 3 are likewise defined. We also let FIT$j(\mathcal{Q}) =$ FIT$ja(\mathcal{Q}) \vee \ldots \vee$ FIT$jd(\mathcal{Q})$ for $j = 1, 2, 3$.

Definition 3. Fit1$(\mathcal{Q})$**:** *The last query is used only once in position* TL. *Note that this is equal to the case where the last query is used only once in position* TR.

Fit1$a(\mathcal{Q})$ *all queries used in the collision are pairwise different,*
Fit1$b(\mathcal{Q})$ BL = TR *and* BR *is different to* TL*,* BL*,* TR*,*
Fit1$c(\mathcal{Q})$ BL = BR *and* TR *is different to* TL*,* BL*,* BR*,*
Fit1$d(\mathcal{Q})$ TR = BR *and* BL *is different to* TL*,* TR*,* BR*,*

Fit2$(\mathcal{Q})$**:** *The last query is used only once in position* BL. *Note that this is equal to the case where the last query is used only once in position* BR.

Fit2$a(\mathcal{Q})$ *all queries used in the collision are pairwise different,*
Fit2$b(\mathcal{Q})$ TL = TR *and* BR *is different to* TL*,* BL*,* TR*,*
Fit2$c(\mathcal{Q})$ TL = BR *and* TR *is different to* TL*,* BL*,* BR*,*
Fit2$d(\mathcal{Q})$ TR = BR *and* TL *is different to* BL*,* TR*,* BR*,*

Fit3$(\mathcal{Q})$**:** *The last query is used twice in a collision. Then:*

Fit3$a(\mathcal{Q})$ *last query used in* TL, BL (TL = BL) *and* TR $\neq$ BR*,*
Fit3$b(\mathcal{Q})$ *last query used in* TL, BL (TL = BL) *and* TR = BR*,*
Fit3$c(\mathcal{Q})$ *last query used in* TL, BR (TL = BR) *and* BL $\neq$ TR*,*
Fit3$d(\mathcal{Q})$ *last query used in* TL, BR (TL = BR) *and* BL = TR.

In Lemma 1 we will show that these configurations cover all possible cases of a collision.
We now define

$$\begin{aligned}
\textsc{Win}1(\mathcal{Q}) &= \neg\textsc{Lucky}(\mathcal{Q}) \wedge \textsc{Fit}1(\mathcal{Q}),\\
\textsc{Win}2(\mathcal{Q}) &= \neg(\textsc{Lucky}(\mathcal{Q}) \vee \textsc{Fit}1(\mathcal{Q})) \wedge \textsc{Fit}2(\mathcal{Q}),\\
\textsc{Win}3(\mathcal{Q}) &= \neg(\textsc{Lucky}(\mathcal{Q}) \vee \textsc{Fit}1(\mathcal{Q}) \vee \textsc{Fit}2(\mathcal{Q})) \wedge \textsc{Fit}3(\mathcal{Q}).
\end{aligned}$$

Thus $\textsc{Win}3(\mathcal{Q})$, for example, is the predicate which is true if and only if $a(\mathcal{Q}) < \alpha$ (i.e. $\neg\textsc{Lucky}(\mathcal{Q})$) and $\mathcal{Q}$ contains queries that fit configurations $3a, 3b, 3c$ or $3d$ but $\mathcal{Q}$ does *not* contain queries fitting configurations $1a, \ldots, 1d, 2a, \ldots 2d$.

We now show, that $\textsc{Coll}^{TDM}(\mathcal{Q}) \implies \textsc{Lucky}(\mathcal{Q}) \vee \textsc{Win}1(\mathcal{Q}) \vee \textsc{Win}2(\mathcal{Q}) \vee \textsc{Win}3(\mathcal{Q})$.

Lemma 1. *$\textsc{Coll}^{TDM}(\mathcal{Q}) \implies \textsc{Lucky}(\mathcal{Q}) \vee \textsc{Win}1(\mathcal{Q}) \vee \textsc{Win}2(\mathcal{Q}) \vee \textsc{Win}3(\mathcal{Q})$.*

Proof. As $\textsc{Fit}1a(\mathcal{Q}) \vee \ldots \vee \textsc{Fit}3d(\mathcal{Q}) \implies \textsc{Win}1a(\mathcal{Q}) \vee \ldots \textsc{Win}3d(\mathcal{Q})$, it is sufficient to show that $\textsc{Coll}^{TDM}(\mathcal{Q}) \implies \textsc{Fit}1a(\mathcal{Q}) \vee \ldots \vee \textsc{Fit}3d(\mathcal{Q})$. Now, say $\textsc{Coll}^{TDM}(\mathcal{Q})$. Then a collision can be constructed from the queries $\mathcal{Q}$. That is, our query history $\mathcal{Q}$ contains queries Q_i, Q_j, Q_k, Q_l (see Figure 2) such that we have a collision, i.e. $V = V'$ and $W = W'$ and TL $\neq$ TR. Note, that the last condition suffices to ensure a real collision (a collision from two different inputs).

First assume that the last query is used once in the collision. If it is used in position TL, then we have to consider the queries BL, TR and BR. If these three queries are all different (and as the last query is only used *once*), then $\textsc{Fit}1a(\mathcal{Q})$. If BL = TR and BR is different, then $\textsc{Fit}1b(\mathcal{Q})$. If BL = BR and TR is different, then $\textsc{Fit}1c(\mathcal{Q})$. If TR = BR and BL is different, then $\textsc{Fit}1d(\mathcal{Q})$. If BL = TR = BR, then we have BL = BR and TL = TR and this would not result in a collision since the inputs to the two compression functions would be the same. As no cases are left, we are done (for the case that the last query is used only in position TL).

If the last query is used once in the collision and is used in position BL, then we have to consider the queries TL, TR and BR. If these three queries are all different (and as the last query is only used *once*), then $\textsc{Fit}2a(\mathcal{Q})$. If TL = TR and BR is different, then $\textsc{Fit}2b(\mathcal{Q})$. If TL = BR and TR is different, then $\textsc{Fit}2c(\mathcal{Q})$. If TR = BR and TL is different, then $\textsc{Fit}2d(\mathcal{Q})$. If TL = TR = BR, it follows TL = TR and BL = BR and this would not result in a collision since the inputs to the two compression functions would be the same. As no cases are left, we are done.

We now analyze the case when the last query is used twice in the collision. First, assume that the query is used for the positions TL and BL (TL = BL). If TR $\neq$ BR, then $\textsc{Fit}3a(\mathcal{Q})$, if TR = BR, then $\textsc{Fit}3b(\mathcal{Q})$. Now assume that the query is employed for the pair TL and BR (TL = BR). Note, that this case is equal to the case where the query is employed for BL and TR. If BL $\neq$ TR, then $\textsc{Fit}3c(\mathcal{Q})$, if BL = TR, then $\textsc{Fit}3d(\mathcal{Q})$. The other cases, *i.e.* the last query is employed either for TL = TR or BL = BR, do not lead to a real collision as this would imply the same compression function input. As no cases are left, we are done.

If the last query is used more than twice for the collision we do not get a real collision as this case would imply either TL = TR or BL = BR and we have the same input, again, for both compression functions. ■

The next step is to upper bound the probability of the predicates $\textsc{Lucky}(\mathcal{Q})$, $\textsc{Win}1(\mathcal{Q})$, $\textsc{Win}2(\mathcal{Q})$ and $\textsc{Win}3(\mathcal{Q})$.

Lemma 2. *Let α be as in Theorem 1. If $\alpha > e$ and $\tau = N'\alpha/q$, then*

$$\Pr[\textsc{Lucky}(\mathcal{Q})] \leq q2^n e^{\tau\nu(1-\ln\tau)}.$$

The proof is quite technical and is given in Appendix A.

Lemma 3. $\Pr[\textsc{Win1}(\mathcal{Q})] \leq q\alpha/N' + 2q/(N')^2 + q/(N')^3$.

Proof. As $\textsc{Win1}(\mathcal{Q}) = \neg\textsc{Lucky}(\mathcal{Q}) \wedge \textsc{Fit1}(\mathcal{Q})$, we will upper bound the probabilities of $\textsc{Fit1}a(\mathcal{Q})$, $\textsc{Fit1}b(\mathcal{Q})$, $\textsc{Fit1}c(\mathcal{Q})$ and $\textsc{Fit1}d(\mathcal{Q})$ separately in order to get an upper bound for $\Pr[\textsc{Fit1}(\mathcal{Q})] \leq \textsc{Fit1}a(\mathcal{Q}) + \ldots + \textsc{Fit1}d(\mathcal{Q})$. We will use the notations given in Figure 3.

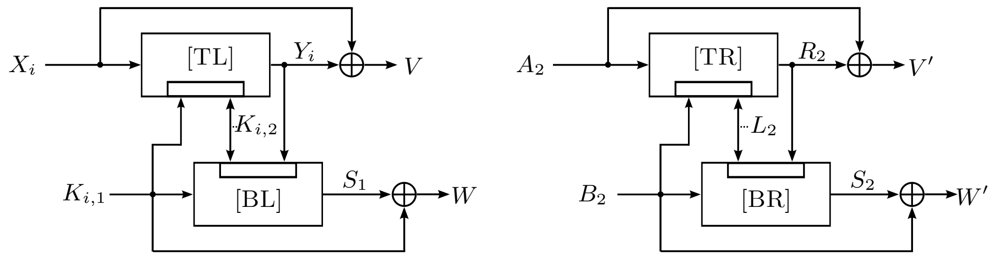

Figure 3. Notations used for $\textsc{Win1}(\mathcal{Q})$

Let $\mathcal{Q}_i$ denote the first i queries made by the adversary. The term 'last query' means the latest query made by the adversary (we examine the adversary's queries $(K_i, X_i)_{fwd}$ or $(K_i, Y_i)_{bwd}$ one at a time as they come in). The last query is always given index i. We say the last query is successful if the output X_i or Y_i for the last query is such that $a(\mathcal{Q}_i) < \alpha$ and such that the adversary can use the query (X_i, K_i, Y_i) to fit the configuration given in Figure 3 using only queries in $\mathcal{Q}_i$ (in particular, the last query *must* be used once in the fitting for that query to count as successful). The goal is thus to upper bound the adversary's chance of ever making a successful last query. The basic setup for upper bounding the probability of success in a given case is to upper bound the maximum number of different outputs Y_i or X_i (depending on whether the last query is a forward or a backward query) that would allow the query (X_i, K_i, Y_i) to be used to fit the configuration, and then divide this number by $N' = 2^n - q$ (since either Y_i or X_i, depending, is chosen randomly among a set of at least $2^n - q$ different values). The ratio is then multiplied by q, since the adversary makes q queries in all, each of which could become a successful last query.

(i) $\textsc{Fit1}a(\mathcal{Q})$: The last query, wlog. $(X_i, K_{i,1}|K_{i,2}, Y_i)$, is used in position TL. We do not care whether the last query was a forward or backward query since the analysis below is the same. All queries are, as claimed, pairwise different. We give the adversary for free the answer to the forward query BL, $(K_{i,1}, K_{i,2}|Y_i, S_1)$. Then we have $V = Y_i \oplus X_i$ and $W = S_1 \oplus K_{i,1}$. This pair of queries is successful if the adversary's query history $\mathcal{Q}_{i-1}$ contains a pair $(A_2, B_2|L_2, R_2), (B_2, L_2|R_2, S_2)$ such that $V = X_i \oplus Y_i = R_2 \oplus A_2 = V'$ and $W = S_1 \oplus K_{i,1} = S_2 \oplus B_2 = W'$. There are at most α queries in $\mathcal{Q}_{i-1}$ that can possibly be used for query in TR that all lead to a collision in the top row, i.e. $V = V'$. Therefore we have at most α possibilities for the query in BRsince the query in TR uniquely determines the query BR. Thus, the last query has a chance of $\leq \alpha/N'$ of succeeding. So the total chance of making a successful query of this type is $\leq q\alpha/N'$.

(ii) Fit1$b(\mathcal{Q})$: Again, the last query, wlog. $(X_i, K_{i,1}|K_{i,2}, Y_i)$, is used in position TL. We give the adversary for free the answer to the forward query BL, $(K_{i,1}, K_{i,2}|Y_i, S_1)$. By our claim, as BL=TR, we have $A_2 = K_{i,1}, B_2 = K_{i,2}, L_2 = Y_i$ and $R_2 = S_1$. It follows that for any given query i for TL, we have at most one query for TR to form a collision $V = V'$ (as the query TL uniquely determines the query BL and the queries BL and TR are equal) and therefore have at most one query BR in our query history to form a collision $W = W'$. The last query has a chance of $\leq 1/(N' \cdot N')$ of succeeding and so the total chance of making a successful query in the attack is $\leq q/(N')^2$.

(iii) Fit1$c(\mathcal{Q})$: As this analysis is essentially the same as for Fit1$b(\mathcal{Q})$ we conclude with a total chance of success for this type of query of $\leq q/(N')^2$.

(iv) Fit1$d(\mathcal{Q})$: Again, the last query, wlog. $(X_i, K_{i,1}|K_{i,2}, Y_i)$, is used in position TL. We give the adversary for free the answer to the forward query BL, $(K_{i,1}, K_{i,2}|Y_i, S_1)$. Note, that this query is trivially different from the query in TL as we assume that the last query is only used once in this configuration (the case in which the two queries, TL and BL, are equal is discussed in the analysis of Win3$(\mathcal{Q})$). We have $V = Y_i \oplus X_i$ and $W = S_1 \oplus K_{i,1}$. As by our claim, we assume TR = BR. The pair of queries for TL and BL is successful if the adversary's query history $\mathcal{Q}_{i-1}$ contains a query $(A_2, B_2|L_2, R_2)$ such that $V = R_2 \oplus A_2 = V'$ and $W = R_2 \oplus A_2 = W'$, i.e. $V = W = V' = W'$. Moreover, it follows from $B_2 = R_2 = L_2$ that $V = W = V' = W' = 0$. As at least three of them are chosen randomly by the initial query input (wlog. V, W, V'), the query has a chance of success in the i-th query $\leq 1/(N' \cdot N' \cdot N')$ and therefore a total chance of success $\leq q/(N')^3$.

The claim follows by adding up the individual results.

■

Lemma 4. $\Pr[\text{Win2}(\mathcal{Q})] \leq q\alpha/N' + 2q/(N')^2 + q/(N')^3$.

As the proof and the result is (in principle) identical to the proof of $\Pr[\text{Win1}(\mathcal{Q})]$ we omitted the details of the proof.

Lemma 5. $\Pr[\text{Win3}(\mathcal{Q})] \leq 2q\alpha/N' + 2q/(N')^2$

Proof. The same notations and preliminaries as in the proof of Lemma 3 are used.

(i) Win3$a(\mathcal{Q})$: The last query, wlog. $(X_i, K_{i,1}|K_{i,2}, Y_i)$ is used in positions TL and BL. We do not care whether the last query is a forward or backward query since the analysis is the same. It follows, that $X_i = K_{i,1} = K_{i,2} = Y_i$ and therefore $V = X_i \oplus Y_i = W = 0$. We assume that the adversary is successful concerning these restraints, i.e. has found a query TL that can also be used for BL such as $X_i = Y_i = K_{i,1} = K_{i,2}$. (Note, that this condition is quite hard.) We do have at most α queries in $\mathcal{Q}_{i-1}$ that can possibly be used for a query in TR and that lead to a collision in the top row, i.e. $0 = V = V'$. For every such query TR we have at most one corresponding query in $\mathcal{Q}_{i-1}$ that can be used in position BR. So the last query has a chance of $\leq \alpha/N'$ of succeeding and so the total chance of making a successful query of this type during the attack is $\leq q\alpha/N'$.

(ii) Win3$b(\mathcal{Q})$: The last query, wlog. $(X_i, K_{i,1}|K_{i,2}, Y_i)$ is used in positions TL and BL. We do not care whether the last query is a forward or backward query since the analysis is the same. It follows, that $X_i = K_{i,1} = K_{i,2} = Y_i$ and therefore $V = X_i \oplus Y_i = W = 0$. We assume again that the adversary is successful concerning these restraints, i.e. has found a query TL that can also be used for BL. We do have at most α queries in $\mathcal{Q}_{i-1}$ that can possibly be used for a query in TR and that lead to a collision in the top row, i.e. $0 = V = V'$. We assume that we can use any such query equally as the corresponding query for BR. In reality, this gives the adversary with high probability more power than he will have. Thus, the last query has a chance of $\leq \alpha/N'$ of succeeding and so the total chance of making a successful query of this type during the attack is $\leq q\alpha/N'$. As discussed above, this upper bound is likely to be generous.

(iii) Win3$c(\mathcal{Q})$: The last query, wlog. $(X_i, K_{i,1}|K_{i,2}, Y_i)$ is used in positions TL and BR. Note, that this situation is equal to the last query being used in position BL and TR. We do not care whether the last query is a forward or backward query. We give the adversary for free the answer to the forward query BL, $(K_{i,1}, K_{i,2}|Y_i, S_1)$. We also give the adversary for free the answer to the backward query TR, $(A_2, X_i|K_{i,1}, K_{i,2})$. The probability for the i-th query to be successful is equal to $\Pr[V = V'] \cdot \Pr[W = W']$, and as W and V' are guaranteed to be chosen independently and randomly the chance of success is $\leq 1/(N')^2$. The total chance of success is therefore $\leq q/(N')^2$.

(iv) Win3$d(\mathcal{Q})$: The last query, wlog. $(X_i, K_{i,1}|K_{i,2}, Y_i)$ is used in positions TL and BR. Note, that this situation is equal to the last query being used in position BL and TR. We do not care whether the last query is a forward or backward query. We give the adversary for free the answer to the forward query BL, $(K_{i,1}, K_{i,2}|Y_i, S_1)$. (This query is also used for position TR and it follows (by comparing the input values of query BL that is used for TR with them of BR) $K_{i,2}|Y_i = X_i|K_{i,1}$ and $S_1 = K_{i,2}$. Comparing the outputs we get a collision in the top row of the compression functions $\Pr[V = V'] = \Pr[E_{K_{i,1}|K_{i,2}}(X_i) \oplus X_i = E_{K_{i,2}|Y_i}(K_{i,1}) \oplus K_{i,1}]$, where $Y_i = E_{K_{i,1}|K_{i,2}}$, with probability $\leq 1/N'$. This is, because the input values $X_i, K_{i,1}, K_{i,2}$ have to be in such a way that the two inputs to the E oracle are different (if they are not, we wold have no colliding inputs for the two compression functions). For the bottom row of the compression function we get, similarly, a collision with probability $\leq 1/N'$. So the total chance for succeeding is in this case $\leq q/(N')^2$ as we have again at most q queries by the adversary.

■

We now give the proof for Theorem 1.

Proof. (of Theorem 1)
The proof follows directly with Lemma 1, 2, 3, 4 and Lemma 5. ■

4 Preimage Resistance

Although, the main focus is on collision resistance, we are also interested in the difficulty of inverting the compression function of Tandem-DM. Generally speaking, second-preimage resistance is a stronger security requirement than preimage resistance. A preimage may have some information of another preimage which produces the same output. However, in the ideal cipher model, for the compression function Tandem-DM, a second-preimage has no information useful to find another preimage. Thus, only preimage resistance is analyzed. Note, that there have be various results that discuss attacks on iterated hash functions in terms of pre- and second-preimage, *e.g.* long-message second-preimage attacks [6, 17], in such a way that the preimage-resistance level cannot easily be transferred to an iterated hash function built on it.

The adversary's goal is to output a preimage (G, H, M) for a given σ, where σ is taken randomly from the output domain, such as $F(G, H, M) = \sigma$. As in the proof of Theorem 1 we will again dispense the adversary from having to output such a preimage. We will determine whether the adversary has been successful or not by examining its query history $\mathcal{Q}$. We say, that $\textsc{PreImg}(\mathcal{Q})$ holds if there is such a preimage and $\mathcal{Q}$ contains all the queries necessary to compute it.

Definition 4. ***(Inverting random points of a compression function)*** *Let F be a blockcipher based compression function, $F : \{0,1\}^{3n} \to \{0,1\}^{2n}$. Fix an adversary $\mathcal{A}$ that has access to oracles E, E^{-1}. Then the advantage of $\mathcal{A}$ of inverting F is the real number*

$$\mathbf{Adv}_F^{Inv}(\mathcal{A}) = \Pr[E \overset{R}{\leftarrow} \mathrm{BC}(n,k); \sigma \overset{R}{\leftarrow} \{0,1\}^{2n} : (G,H,M) \overset{R}{\leftarrow} \mathcal{A}^{E,E^{-1}}(\sigma) : F(G,H,M) = \sigma].$$

Again, for $q \geq 1$, we write

$$\mathbf{Adv}_F^{Inv}(q) = \max_{\mathcal{A}}\{\mathbf{Adv}_F^{Inv}(\mathcal{A})\}$$

where the maximum is taken over all adversaries that ask at most q oracle queries.

Note, that there has been a discussion on formalizations of preimage resistance. For details we refer to [2, Section 2, Appendix B].

4.1 Preimage Security

The preimage resistance of the compression function $\textsc{Tandem}$-DM is given in the following Theorem.

Theorem 2. *Let $F := F^{TDM}$ be as in Definition 1. For every $N' = 2^n - q$ and $q > 1$*

$$\mathbf{Adv}_F^{Inv}(q) \leq 2q/(N')^2.$$

Proof. Fix $\sigma = (\sigma_1, \sigma_2) \in \{0,1\}^{2n}$ where $\sigma_1, \sigma_2 \in \{0,1\}^n$ and an adversary $\mathcal{A}$ asking q queries to its oracles. We upper bound the probability that $\mathcal{A}$ finds a preimage for a given σ by examining the oracle queries as they come in and upper bound the probability that the last query can be used to create a preimage, i.e. we upper bound $\Pr[\textsc{PreImg}(\mathcal{Q})]$. Let $\mathcal{Q}_i$ denote the first i queries made by the adversary. The term 'last query' means the latest query made by the adversary since we examine again the adversary's queries $(K_i, X_i)_{fwd}$ or $(K_i, X_i)_{bwd}$ one at a time as they come in. The last query is always given index i.

Case 1: The last query (X_i, K_i, Y_i) is used in the top row. Either X_i or Y_i was randomly assigned by the oracle from a set of at least the size N'. The query is successful *in the top row* if $X_i \oplus Y_i = \sigma_1$ and thus has a chance of success of $\leq 1/N'$. In $\mathcal{Q}_{i-1}$ there is at most one query Q_j that matches for the bottom row. If there is no such query in $\mathcal{Q}$ we give this query Q_j the adversary for free. This 'bottom' query is successful if $X_j \oplus Y_j = \sigma_2$ and therefore has a chance of success of $\leq 1/N'$. So the total chance of success after q queries is $\leq q/(N')^2$

Case 2: The last query (X_i, K_i, Y_i) is used in the bottom row. The analysis is essentially the same as in case 1. The total chance of success if $\leq q/(N')^2$, too.

As any query can be either used in the top or the bottom row the claim follows.

5 Discussion and Conclusion

In this paper, we have investigated the security of TANDEM-DM, a long outstanding DBL compression function based on an $(n, 2n)$-blockcipher. In the ideal cipher model, we showed that this construction has birthday-type collision resistance. As there are some generous margins in the proof it is likely, that TANDEM-DM is even more secure. Our bound for preimage resistance is far from optimal, but we have not found an attack that would classify this bound as tight.

Somewhat surprisingly, there seems to be only one practical rate 1/2 DBL compression function that also has a birthday-type security guarantee. It was presented at FSE'06 by Hirose [14]. Taking into account that it was presented about 15 years after TANDEM-DM, it is clear that there needs still to be a lot of research done in the field of blockcipher based hash functions, *e.g.* there are still security proofs missing for the aforementioned ABREAST-DM and MDC-4 compression or hash functions.

References

[1] ANSI. *ANSI X9.31:1998: Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA).* American National Standards Institute, pub-ANSI:adr, 1998.

[2] John Black, Phillip Rogaway, and Thomas Shrimpton. Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV. In Moti Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 320–335. Springer, 2002.

[3] C. Meyer and S. Matyas. Secure program load with manipulation detection code, 1988.

[4] D. Coppersmith, S. Pilpel, C. H. Meyer, S. M. Matyas, M. M. Hyden, J. Oseas, B. Brachtl, and M. Schilling. Data authentication using modification dectection codes based on a public one way encryption function. U.S. Patent No. 4,908,861, March 13, 1990.

[5] Ronald Cramer, editor. *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*. Springer, 2005.

[6] Richard Drews Dean. *Formal aspects of mobile code security.* PhD thesis, Princeton, NJ, USA, 1999. Adviser-Andrew Appel.

[7] Bert den Boer and Antoon Bosselaers. Collisions for the Compressin Function of MD5. In *EUROCRYPT*, pages 293–304, 1993.

[8] Eberhard Freitag and Rolf Busam. Complex Analysis, Springer; 1st edition, 2005.

[9] Shimon Even and Yishay Mansour. A Construction of a Cipher From a Single Pseudorandom Permutation. In Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto, editors, *ASIACRYPT*, volume 739 of *Lecture Notes in Computer Science*, pages 210–224. Springer, 1991.

[10] Praveen Gauravaram, William Millan, and Lauren May. CRUSH: A New Cryptographic Hash Function using Iterated Halving Technique. In Ed Dawson and Wolfgang Klemm, editors, *Cryptographic Algorithms and their Uses*, pages 28–39. Queensland University of Technology, 2004.

[11] H. Dobbertin. The status of MD5 after a recent attack, 1996.

[12] Mitsuhiro Hattori, Shoichi Hirose, and Susumu Yoshida. Analysis of Double Block Length Hash Functions. In Kenneth G. Paterson, editor, *IMA Int. Conf.*, volume 2898 of *Lecture Notes in Computer Science*, pages 290–302. Springer, 2003.

[13] Shoichi Hirose. Provably Secure Double-Block-Length Hash Functions in a Black-Box Model. In Choonsik Park and Seongtaek Chee, editors, *ICISC*, volume 3506 of *Lecture Notes in Computer Science*, pages 330–342. Springer, 2004.

[14] Shoichi Hirose. Some Plausible Constructions of Double-Block-Length Hash Functions. In Matthew J. B. Robshaw, editor, *FSE*, volume 4047 of *Lecture Notes in Computer Science*, pages 210–225. Springer, 2006.

[15] Walter Hohl, Xuejia Lai, Thomas Meier, and Christian Waldvogel. Security of Iterated Hash Functions Based on Block Ciphers. In Stinson [35], pages 379–390.

[16] ISO/IEC. *ISO DIS 10118-2: Information technology - Security techniques - Hash-functions, Part 2: Hash-functions using an n-bit block cipher algorithm. First released in 1992*, 2000.

[17] John Kelsey and Bruce Schneier. Second Preimages on n-Bit Hash Functions for Much Less than 2^{n} Work. In Cramer [5], pages 474–490.

[18] Joe Kilian and Phillip Rogaway. How to Protect DES Against Exhaustive Key Search. In Neal Koblitz, editor, *CRYPTO*, volume 1109 of *Lecture Notes in Computer Science*, pages 252–267. Springer, 1996.

[19] Lars R. Knudsen, Xuejia Lai, and Bart Preneel. Attacks on Fast Double Block Length Hash Functions. *J. Cryptology*, 11(1):59–72, 1998.

[20] Lars R. Knudsen and Frédéric Muller. Some Attacks Against a Double Length Hash Proposal. In Bimal K. Roy, editor, *ASIACRYPT*, volume 3788 of *Lecture Notes in Computer Science*, pages 462–473. Springer, 2005.

[21] Xuejia Lai and James L. Massey. Hash Function Based on Block Ciphers. In *EUROCRYPT*, pages 55–70, 1992.

[22] M. Rabin. Digitalized Signatures. *In R. DeMillo, D. Dobkin, A. Jones and R.Lipton, editors, Foundations of Secure Computation, Academic Press*, pages 155–168, 1978.

[23] Alfred Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1996.

[24] Ralph C. Merkle. One Way Hash Functions and DES. In Gilles Brassard, editor, *CRYPTO*, volume 435 of *Lecture Notes in Computer Science*, pages 428–446. Springer, 1989.

[25] Nandi, Lee, Sakurai, and Lee. Security Analysis of a 2/3-Rate Double Length Compression Function in the Black-Box Model. In *IWFSE: International Workshop on Fast Software Encryption, LNCS*, 2005.

[26] NIST National Institute of Standards and Technology. FIPS 180-1: Secure Hash Standard. April 1995. See http://csrc.nist.gov.

[27] NIST National Institute of Standards and Technology. FIPS 180-2: Secure Hash Standard. April 1995. See http://csrc.nist.gov.

[28] Bart Preneel, René Govaerts, and Joos Vandewalle. Hash Functions Based on Block Ciphers: A Synthetic Approach. In Stinson [35], pages 368–378.

[29] R. L. Rivest. *RFC 1321: The MD5 Message-Digest Algorithm.* Internet Activities Board, April 1992.

[30] Ronald L. Rivest. The MD4 Message Digest Algorithm. In Alfred Menezes and Scott A. Vanstone, editors, *CRYPTO*, volume 537 of *Lecture Notes in Computer Science*, pages 303–311. Springer, 1990.

[31] Phillip Rogaway and John P. Steinberger. Constructing Cryptographic Hash Functions from Fixed-Key Blockciphers. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 433–450. Springer, 2008.

[32] Phillip Rogaway and John P. Steinberger. Security/Efficiency Tradeoffs for Permutation-Based Hashing. In Nigel P. Smart, editor, *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 220–236. Springer, 2008.

[33] Satoh, Haga, and Kurosawa. Towards Secure and Fast Hash Functions. *TIEICE: IEICE Transactions on Communications/Electronics/Information and Systems*, 1999.

[34] John P. Steinberger. The Collision Intractability of MDC-2 in the Ideal-Cipher Model. In Moni Naor, editor, *EUROCRYPT*, volume 4515 of *Lecture Notes in Computer Science*, pages 34–51. Springer, 2007.

[35] Douglas R. Stinson, editor. *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, volume 773 of *Lecture Notes in Computer Science*. Springer, 1994.

[36] Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, and Xiuyuan Yu. Cryptanalysis of the Hash Functions MD4 and RIPEMD. In Cramer [5], pages 1–18.

[37] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding Collisions in the Full SHA-1. In Victor Shoup, editor, *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 17–36. Springer, 2005.

A Proof of Lemma 2

We can rephrase the problem of upper bounding $\Pr[\textsc{Lucky}(\mathcal{Q})] = \Pr[a(\mathcal{Q}) > \alpha]$ as a balls-in-bins question. Let $N = 2^n$ be the number of bins and q be the number of balls to be thrown. The i-th ball falls into the j-th bin if the XOR output of the i-th query is equal to the XOR output of the j-th query, i.e. $X_i \oplus Y_i = X_j \oplus Y_j$. In the following we will upper bound the probability that some bin contains more than α balls. As the balls are thrown independent of each other, the i-th ball always has probability $\leq p = 1/(2^n - q)$ of falling in the j-th bin. This is because the XOR output of the i-th query is chosen uniformly at random from a set of size at least $2^n - q$. If we let $B(k)$ be the probability of having exactly k balls in a particular bin, say bin 1, then

$$B(k) \leq p^k \binom{q}{k}.$$

Let $\nu = qp$, where ν is an upper bound for the expected number of balls in any bin. By Stirlings approximation [8] (and e^x being the exponential function)

$$n! \leq \sqrt{2\pi n} \cdot \left(\frac{n}{e}\right)^n \cdot e^{1/(12n)}$$

we can upper bound $B(k)$ as follows

$$\begin{aligned} B(k) &\leq p^k \frac{q!}{k!(q-k)!} \\ &\leq \frac{p^k}{\sqrt{2\pi}} \sqrt{\frac{q}{k(q-k)}} \cdot \frac{q^q}{k^k (q-k)^{1-k}} \cdot \frac{e^k \cdot e^{q-k}}{e^q} \cdot e^{\frac{1}{12}(q-k-(q-k))} \\ &\leq k^{-k} \nu^k \left(\frac{q}{q-k}\right) \\ &\leq \nu^k \cdot k^{-k} \cdot e^k. \end{aligned}$$

Since $\alpha = \tau\nu$ we get

$$B(\alpha) \leq \frac{\nu^{\tau\nu} e^{\tau\nu}}{(\tau\nu)^{\tau\nu}} = \frac{e^{\tau\nu}}{\tau^{\tau\nu}} = e^{\tau\nu(1-\ln\tau)}.$$

As $B(\alpha)$ is a decreasing function of α if $(1 - \ln\tau) < 0$. It follows that $B(\alpha)$ is a decreasing function if $\alpha > e$. And so we have

$$\begin{aligned} \Pr[a(\mathcal{Q}) > \alpha] \quad &\leq \quad 2^n \cdot \sum_{j=\alpha}^{q} B(j) \\ &\leq \quad q2^n B(\alpha) \quad \leq \quad q2^n e^{\tau\nu(1-\ln\tau)}. \end{aligned}$$

This proves our claim. ■

B Security of the FSE'06 Proposal by Hirose for a DBL Compression Function

At FSE'06, Hirose [14] proposed a DBL compression function (Definition 5 and Figure 4). He proved that when this compression function F^{Hirose} is employed in an iterated hash function H, that no adversary asking less than $2^{126.71}$ (assuming AES-256 as the blockcipher) can have an even chance in finding a collision. As he has not stated a security result for the compression function we do here for comparison with Tandem-DM.

B.1 Compression Function

Definition 5. *Let $F^{Hirose} : \{0,1\}^{2n} \times \{0,1\}^n \to \{0,1\}^{2n}$ be a compression function such that $(G_i, H_i) = F^{Hirose}(G_{i-1}, H_{i-1}, M_i)$ where $G_i, H_i, M_i \in \{0,1\}^n$. F^{Hirose} is built upon a $(n, 2n)$-blockcipher E as follows:*

$$G_i = \mathrm{F}_T(G_{i-1}, H_{i-1}, M_i) = E(G_{i-1}, H_{i-1}|M_i) \oplus G_{i-1}$$
$$H_i = \mathrm{F}_B(G_{i-1}, H_{i-1}, M_i) = E(G_{i-1} \oplus C, H_{i-1}|M_i) \oplus G_{i-1} \oplus C,$$

where $|$ represents concatenation and $c \in \{0,1\}^n - \{0^n\}$ is a constant.

A visualization of this compression function is given in the following Figure 4.

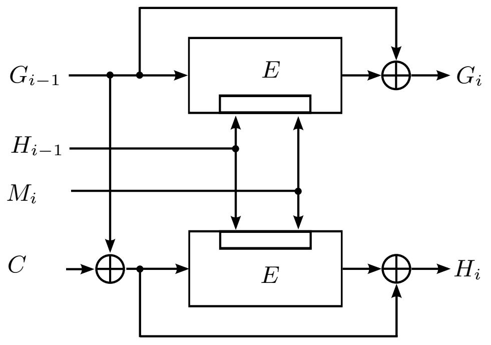

Figure 4. The compression function F^{Hirose}, E is an $(n, 2n)$-blockcipher, the small rectangle inside the cipher rectangle indicates the position of the key

B.2 Security of Compression Function

As the security proof of Hirose [14, Theorem 4] only states the collision resistance of a hash function built using F^{Hirose} we will give a security proof for the compression function alone. In particular, we will show

Theorem 3. *Let $F := F^{Hirose}$ be a compression function as in Defintion 5. Then,*

$$\mathbf{Adv}_F^{\text{COLL}}(q) \leq \frac{2q^2}{(2^n - 2q)^2} + \frac{2q}{2^n - 2q}.$$

In numerical terms, it means that no adversary performing less than $2^{124.55}$ oracle calls has more than an even chance, *i.e.* 0.5, in finding a collision.

Due to the special structure of the compression function, the following definition is useful for the proof.

Definition 6. *A pair of distinct inputs $(G_{i-1}, H_{i-1}, M_i), (G'_{i-1}, H'_{i-1}, M'_i)$ to F^{Hirose} is called a* matching *pair if $(G'_{i-1}, H'_{i-1}, M'_i) = (G_{i-1}, H_{i-1}, M_i \oplus C$. Otherwise they are called a* non-matching *pair.*

Note, that the proof is essentially due to Hirose [14], but as he stated it only for the hash function and not for the compression function itself. We will give a proof here for the compression function.

Proof. Let $\mathcal{A}$ be an adversary that asks q queries to oracles E, E^{-1}. Since

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

depends both on the plaintext and the ciphertext of E and one of them is fixed by a query and the other is determined by the answer it follows that G_i is determined randomly. We give the adversary for free the answer to the query for H_i. Let $(X_i, K_{i,1}|K_{i,2}, Y_i)$ and $(X_i \oplus C, K_{i,1}|K_{i,2}, Z_i)$ be the triplest of E obtained by the i-th pair of queries and the corresponding answers.

For every $2 \leq i \leq q$, let C_i be the event that a colliding pair of non-matching inputs are found for F with the i-th pair of queries. Namely, it is the event that, for some $i' < i$

$$F(X_i, K_{1,i}, K_{2,i}) = F(X_{i'}, K_{1,i'}, K_{2,i'}) \text{ or } F(X_{i'} \oplus C, K_{1,i'}, K_{2,i'})$$

or

$$F(X_i \oplus C, K_{1,i}, K_{2,i}) = F(X_{i'}, K_{1,i'}, K_{2,i'}) \text{ or } F(X_{i'} \oplus C, K_{1,i'}, K_{2,i'})$$

which is equivalent to

$$(Y_i \oplus X_i, Z_i \oplus X_i \oplus C) = (Y_{i'} \oplus X_{i'}, Z_{i'} \oplus X_{j'} \oplus C) \text{ or } (Z_{i'} \oplus X_{i'} \oplus C, Y_{i'} \oplus X_{j'}).$$

It follows, that

$$\Pr[\mathrm{C}_i] \leq \frac{2(i-1)}{(2^n - (2i-2))(2^n - (2i-1))} \leq \frac{2q}{(2^n - 2q)^2}.$$

Let C be the event that a colliding pair of non-matching inputs are found for F^{Hirose} with q (pairs) of queries. Then,

$$\Pr[\mathrm{C}] \leq \sum_{i=2}^{q} \Pr[\mathrm{C}_j] \leq \sum_{i=2}^{q} \frac{2q}{(2^n - 2q)^2} \leq \frac{2q^2}{(2^n - 2q)^2}.$$

Now, let $\hat{\mathrm{C}}_i$ be the event that a colliding pair of matching inputs is found for F. It follows, that

$$\Pr[\hat{\mathrm{C}}_i] \leq \frac{2}{(2^n - 2q)}.$$

Let $\hat{\mathrm{C}}$ be the event that a colliding pair of matching inputs are found for F^{Hirose} with q (pairs) of queries. Then,

$$\Pr[\hat{\mathrm{C}}] \leq \sum_{i=2}^{q} \Pr[\hat{\mathrm{C}}_i] \leq \frac{2q}{2^n - 2q}.$$

Since $\mathbf{Adv}_F^{\mathrm{COLL}}(q) = \Pr[\mathrm{C} \vee \hat{\mathrm{C}}] \leq \Pr[\mathrm{C}] + \Pr[\hat{\mathrm{C}}]$, the claim follows.