On Hashing With Tweakable Ciphers

Raphael C.-W. Phan
Loughborough University, UK
Email: R.Phan@lboro.ac.uk

Jean-Philippe Aumasson
FHNW, Windisch, Switzerland
Email: JeanPhilippe.Aumasson@gmail.com

***Abstract*—Cryptographic hash functions are often built on block ciphers in order to reduce the security of the hash to that of the cipher, and to minimize the hardware size. Proven secure constructions are used in international standards like MD5, SHA-1, or Whirlpool. But recently, researchers proposed new modes of operations for hash functions to protect against generic attacks, and it remains open how to base such function on block ciphers. An attracting and intuitive choice is to combine previous constructions with tweakable block ciphers. We investigate such constructions, and show the surprising result that combining a secure mode of operation with a secure tweakable cipher does not guarantee the security of the hash function built. In fact, simple attacks can be possible when the interaction between secure components leaves some additional "freedom" to an adversary. Our techniques are derived from the principle of slide attacks, which were introduced for attacking block ciphers.**

I. Introduction

Cryptographic hash functions are key ingredients in numerous schemes like public-key encryption, digital signatures, message-authentication codes, or multiparty functionalities. The last past years, the focus on hash functions has dramatically increased, because of new dedicated attacks on e.g. MD5 and SHA-1, and new generic attacks—that is, which apply to broad classes of functions. A hash function h should satisfy (at least)

- *collision resistance*: it should be hard to find distinct inputs x and x' such that $h(x) = h(x')$
- *second-preimage resistance*: given a random input x, it should be hard to find a distinct x' such that $h(x) = h(x')$
- *preimage resistance*: given $h(x)$ for a random unknown x, it should be hard to find a distinct x' such that $h(x) = h(x')$

Critical generic attacks [1]–[3] were presented against the classical Merkle-Damgård (MD) iterative mode of operation, thus threating all the functions using the MD operation mode (for example, MD5 and SHA-1). An MD hash function hashes a message $M = M_1 M_2 \dots M_\ell$ as follows: for $1 \leq i \leq \ell$, compute

$$h_i = f(h_{i-1}, M_i),$$

where f is called the *compression function*, and h_0 is a predefined initialization vector (IV). Finally the function returns the hash value $H(M) = h_\ell$.

To prevent from attacks on the MD mode, extended operation modes were proposed (e.g. HAIFA [4], [5]); in this work we focus on Rivest's *dithered MD* (DMD) mode, for its simplicity and better efficiency. DMD was proposed as a general framework for hash functions, and it remains to be seen how to concretely instantiate the underlying compression function. Furthermore, we know of no concrete hash function construction that employs DMD. The question has been discussed among the community as to whether the upcoming NIST hash function competition [6] should concentrate on only concrete hash function proposals, or split between proposals for operating modes and for compression functions (cf. [7]).

Block cipher-based constructions for hash functions used to build on the MD mode [8], [9] (the so-called *PGV* schemes) and there is no direct way to extend them to DMD, because of an additional input to the compression function. Recently, *ad-hoc* constructions were proposed [10], but it is unclear whether this approach is optimal. A natural approach—yet unexplored—is to use *tweakable block ciphers* [11] to instanciate DMD hash functions. The model of tweakable block cipher was originally proposed by Liskov, Rivest, and Wagner to define families of permutations for a fixed secret key, thus avoiding the slowdown caused by the key schedule operation.

A. Contribution

We first present two classes of constructions for DMD hash functions based on tweakable block ciphers, which combine[1]

1) a secure hash mode of operation
2) a secure tweakable block cipher
3) a secure block cipher-based construction

Then, we show that such constructions do not necessarily lead to a secure hash function. More precisely, we apply the idea of slid pairs to find *collisions* for one of the functions classes. Our attacks apply to broad classes of constructions, and are independent of the strength of the block cipher used.

B. Related Work

Dithering of hash functions appeared with the work of Kelsey and Schneier [1], with generalizations in [4], [5], [12], [13]. An analysis of dithered hash functions appears in [14], and constructive results were proposed in [10].

Hash functions based on block ciphers recently attracted considerable attention, with several results proving security bounds for constructions with one or more block cipher [15]–[18]. Concrete block cipher-based design include Maelstrom [19] and Grindahl [20], and implicitly the *de facto* standards MD5, SHA-1, and SHA-2.

[1]The notion of security differs for each of these constructions, see the corresponding papers [8], [11], [12] for details.

The idea of slide attacks was applied on hash functions to the *compression function* of SHA-1 in [21], [22], and later extended to the block cipher SHACAL-1. Recently, Gorski et al. made a more direct application of slide attacks against Sponge functions [23]. The attacks by Dean [24], and Kelsey and Schneier [1] exploit a fixed-point of the compression function, which is similar in spirit to slide attacks [25], [26] and to our attacks.

In [27], attacks were mounted on a hash function mode based on tweakable block ciphers, so called Tweak Chain Hash (TCH). This construction, however, is a conventional hash function (not dithered). The attack on TCH does not apply to our constructions.

II. Definitions

A *block cipher* is a map $E : \{0,1\}^k \times \{0,1\}^m \mapsto \{0,1\}^m$, such that $E_K(\cdot) = E(K,\cdot)$ is a permutation of $\{0,1\}^m$ for all $K \in \{0,1\}^k$, and its inverse permutation is written E^{-1}. The set of all blockciphers with k-bit key and m-bit messages is denoted $\mathsf{Bloc}(k,m)$. A *blockcipher-based hash function* is a map $H : \mathsf{Bloc}(k,m) \times D \mapsto R$, where $D \subseteq \{0,1\}^\star$ and $R = \{0,1\}^n$, defined iteratively by a compression function $f : \mathsf{Bloc}(k,m) \times \{0,1\}^{n_1} \times \{0,1\}^{n_2} \mapsto \{0,1\}^{n_2}$, where n_1 is the size of a message block, and n_2 the size of chaining values. In the remainder of the paper, we assume $m = n_1 = n_2 = n$.

A. Hashing Based on Block Ciphers

Among the 12 constructions presented in [8], we will focus on the most popular ones (which are used in all concrete hash designs):

- the Matyas-Meyer-Oseas [28] (MMO) scheme, which constructs the compression function by setting
$$h_i = E_{h_{i-1}}(M_i) \oplus M_i$$
- the Davies-Meyer scheme, somehow the dual of MMO, is used in the MD5 and SHA functions:
$$h_i = E_{M_i}(h_{i-1}) \oplus h_{i-1}$$
- the Miyaguchi-Preneel scheme [29], [30], notably employed in Whirlpool [31], with as blockcipher a variant of Rijndael:
$$h_i = E_{h_{i-1}}(M_i) \oplus h_{i-1} \oplus M_i$$

B. Tweakable Block Ciphers

Tweakable block ciphers [11] aim to achieve the "best of both worlds" (security and efficiency) for block cipher-based hashing, since they allow to use an input-dependent permutation, thus avoiding the time-consuming key schedule—the additional input (tweak) being injected in a simplistic fashion.

Formally, a tweakable block cipher has three inputs: a key $K \in \{0,1\}^k$, a tweak $T \in \{0,1\}^t$ and a message $M \in \{0,1\}^n$; it produces a ciphertext $C \in \{0,1\}^n$:

$$\tilde{E} : \{0,1\}^k \times \{0,1\}^t \times \{0,1\}^n \mapsto \{0,1\}^n.$$

We write $\tilde{E}_K(T,M)$ as shorthand for $\tilde{E}(K,T,M)$. In [11], two tweakable block ciphers are constructed from classical block ciphers:

- TEXE (tweakable cipher formed by two E boxes sandwiching an XOR), which is inspired from CBC-MAC and defines
$$\tilde{E}_K(T,M) = E_K(T \oplus E_K(M)).$$
It was proven [11] that TEXE is a secure tweakable cipher in the sense of indistinguishability from a family of random permutations parametrized by the tweak.
- TFX (inspired from FX of Kilian and Rogaway [32], [33]) defines the scheme
$$\tilde{E}_K(T,M) = E_K(M \oplus U(T)) \oplus U(T),$$
where U is a universal hash function TFX is *strongly* (chosen-ciphertext) secure in the sense of indistinguishability from a family of random permutations parametrized by the tweak (see [11] for details).

III. Dithered Hash Functions (DMD)

Dithering is a generalization of the countermeasure proposed by Kelsey and Schneier [1] to prevent attacks [1], [24] based on message block repetition and fixed-points. This type of iterated hash uses a sequence of dither values $D = d_1 \dots d_\ell$, which is *public and static*.

A dithered Merkle-Damgård (DMD) hash function, as defined in [12], [13], takes as input an IV, a message $M = M_1 M_2 \dots M_\ell$, a dither sequence $D = d_1 d_2 \dots d_\ell$, and produces an output $H_D(M)$ as follows: for $1 \le i \le \ell$, compute

$$h_i = f(h_{i-1}, M_i, d_i)$$

where f is the compression function, the h_i's are chaining variable; h_0 is the IV and the dither values $d_0, \dots, d_{\ell-1}$ have a *zero most significant bit* (MSB), and d_ℓ, the last dither value, has *nonzero MSB*.

In the above definition, a special MSB encoding of d_i differentiates the last block from other blocks. This features was proposed to avoid a complex message padding rule (unlike classical MD functions, which append to a message the encoding of its bit length).

IV. Constructions

We present constructions of DMD hash functions where the compression function is instantiated with a tweakable block cipher, in one of the 12 provably secure PGV modes. The dither input d_i of the compression function is directed to the tweak input T of the tweakable block cipher. We focus on the MMO, to simplify the description.

A. DMD-TEXE *in* MMO *Mode*

This construction of a DMD function combines the TEXE and MMO schemes, which respectively add a new input slot and construct a secure compression function. Following our

previous definitions, DMD-TEXE with the MMO scheme defines

$$\begin{aligned} h_i &= f(h_{i-1}, M_i, d_i) \\ &= \tilde{E}_{h_{i-1}}(d_i, M_i) \oplus M_i \\ &= E_{h_{i-1}}(d_i \oplus E_{h_{i-1}}(M_i)) \oplus M_i \\ &= E_{h_{i-1}}(d_i \oplus E_{h_{i-1}}(M_i)) \oplus M_i. \end{aligned}$$

This construction, however, is inefficient, since each call to the compression function requires two encryptions with the block cipher E, plus one key schedule (both encryptions use the same key).

B. DMD-TFX *in* MMO *Mode*

This construction is more efficient than DMD-TEXE, since requires only one encryption (and the key schedule), plus a call to a universal hash function, which is generally faster than the block cipher in practice. This construction defines:

$$\begin{aligned} h_i &= f(h_{i-1}, M_i, d_i) \\ &= \tilde{E}_{h_{i-1}}(d_i, M_i) \oplus M_i \\ &= E_{h_i - 1}(M_i \oplus U(d_i)) \oplus U(d_i) \oplus M_i \\ &= E_{h_i - 1}(M_i \oplus U(d_i)) \oplus (M_i \oplus U(d_i)). \end{aligned}$$

V. Collision Attack

We show how to mount a free-start collision attack on DMD-TFX in MMO mode, resulting in a pair of colliding messages M and M' for a predefined IV and another IV'. The attack goes as follows, for an arbitrary dither sequence $D = D = d_1 \dots d_{\ell+1}$:

1) choose an arbitrary $(\ell + 1)$-block message $M = M_1 M_2 \dots M_{\ell+1}$, and compute $H_D(M)$
2) define a ℓ-block message $M' = M'_1 \dots M'_\ell$, where

$$M'_i = M_{i+1} \oplus U(d_{i+1}) \oplus U(d_i). \quad (1)$$

3) compute $H_D(M')$ using the IV $h'_0 \neq h_0$ defined as

$$h'_0 = f(h_0, M_1, d_1) = h_1. \quad (2)$$

Now, Eq. (1) and (2) yield

$$\begin{aligned} h'_1 &= f(h'_0, M'_1, d_1) \\ &= E_{h'_0}(M'_1 \oplus U(d_1)) \oplus (M'_1 \oplus U(d_1)) \\ &= E_{h_1}(M_2 \oplus U(d_2)) \oplus (M_2 \oplus U(d_2)) \\ &= f(h_1, M_2, d_2) \\ &= h_2. \end{aligned}$$

Then, by induction,

$$\begin{aligned} h'_i &= f(h'_{i-1}, M'_i, d_i) \\ &= E_{h'_{i-1}}(M'_i \oplus U(d_i)) \oplus (M'_i \oplus U(d_i)) \\ &= E_{h_i}(M_{i+1} \oplus U(d_{i+1})) \oplus (M_{i+1} \oplus U(d_{i+1})) \\ &= f(h_i, M_{i+1}, d_{i+1}) \\ &= h_{i+1}. \end{aligned}$$

Eventually we have $h = h_{\ell+1}$ and $h' = h'_\ell$, and thus $H_D(M) = H_D(M')$, i.e. a collision.

A. Generalization

This attack generalizes to other DMD-TFX modes, for example when the underlying block cipher-based compression function sets

$$E_{h_{i-1}}(M_i \oplus h_{i-1}) \oplus M_i \oplus h_{i-1},$$

or

$$E_{h_{i-1}}(M_i \oplus h_{i-1}) \oplus M_i.$$

These are the constructions called f_2 and f_4 in [9]. Our attack also applies to the Miyaguchi-Preneel scheme.

More generally, our attack can in general be applied to DMD hash functions with *symmetric mixing* of the message block and the dither input. Examples of this kind would be inspired from perceptual hash functions in image authentication applications [34], [35] of the field of image processing, from which the notion of "dithering" is inspired. In such hash functions, the dither input is mixed within the compression function in the same way as how a message input is mixed, hence the mixing is symmetric.

We can describe a further generalization, when the compression function f can be expressed as

$$\begin{aligned} h_i &= f(h_{i-1}, M_i, d_i) \\ &= f'(h_{i-1}, f_1(M_i) \otimes f_2(d_i)) \end{aligned}$$

where $\otimes$ and f_1 are arbitrary *invertible* functions, and f_2 is an arbitrary function, not necessarily invertible.

Our attack in can be applied to this case too, by choosing M'_i such that

$$f_1(M'_i) \otimes f_2(d_i) = f_1(M_{i+1}) \otimes f_2(d_{i+1}),$$

i.e., we choose:

$$M'_i = f_1^{-1}\left\{[f_1(M_{i+1}) \otimes f_2(d_{i+1})] \otimes^{-1} f_2(d_i)\right\}.$$

Note that f_1 and f_2 can be defined to also take the chaining variable h_{i-1} as input, and our attack equally applies.

B. Applicability to DMD-TEXE Functions

Functions based on the TEXE construction resist our attacks since the mixing between M_i and d_i is not invertible, with respect to expressing M_i from the above equation in terms of the other variables h_i, h_{i-1} and d_i.

It might seem counter-intuitive that DMD is insecure against our attack when instantiated with TFX yet is resistant when instantiated with the essentially weaker TEXE. This might be explained as follows: the difference between the two security notions achieved by TFX and TEXE is in terms of the access to the decryption oracle, which does not appear to be useful since PGV modes only make use of the underlying block cipher in the encryption direction.

VI. Conclusions

Attacks such as [1], [24] seem to provide support for the hypothesis that the adversary attacking an MD hash function has too much control over the message block input to the compression function, and so this control should be restricted, e.g. with dithering. DMD hash functions therefore increase the security guarantees, by using different compressions of the message at each iteration. However, the dither value may expose the functions to new attacks, as our attacks showed.

Our attacks can be easily foiled by appending to the message its bit length (this makes the function a bit less efficient). It remains open, though, whether concrete hash functions should be based on block ciphers, or be dedicated designs in order to satisfy the very particular security requirements of a hash function, not necessarily captured by those of block ciphers (cf. notions like indifferentiability, seed-incompressibility, secure MAC, etc.).

References

[1] J. Kelsey and B. Schneier, "Second preimages on n-bit hash functions for much less than 2^n work." in *EUROCRYPT*, ser. LNCS, R. Cramer, Ed., vol. 3494. Springer, 2005, pp. 474–490.

[2] J. Kelsey and T. Kohno, "Herding hash functions and the Nostradamus attack," First NIST Cryptographic Hash Function Workshop, 2005.

[3] ——, "Herding hash functions and the Nostradamus attack." in *EUROCRYPT*, ser. LNCS, S. Vaudenay, Ed., vol. 4004. Springer, 2006, pp. 183–200.

[4] E. Biham, "Recent advances in hash functions - the way to go," 2005, presented at the ECRYPT Hash Function Workshop, 2005.

[5] E. Biham and O. Dunkelman, "A framework for iterative hash functions - HAIFA," Cryptology ePrint Archive, Report 2007/278, 2007, previously presented at the second NIST Hash Function Workshop, 2006.

[6] NIST, "Cryptographic hash competition," 2008, `http://www.nist.gov/hash-function/`.

[7] D. J. Bernstein, NIST's hash mailing list, post on June 8, 2007.

[8] B. Preneel, R. Govaerts, and J. Vandewalle, "Hash functions based on block ciphers: A synthetic approach," in *CRYPTO*, ser. LNCS, D. R. Stinson, Ed., vol. 773. Springer, 1993, pp. 368–378.

[9] J. Black, P. Rogaway, and T. Shrimpton, "Black-box analysis of the block-cipher-based hash-function constructions from pgv." in *CRYPTO*, ser. LNCS, M. Yung, Ed., vol. 2442. Springer, 2002, pp. 330–335.

[10] J.-P. Aumasson and R. C.-W. Phan, "How (not) to efficiently dither blockcipher-based hash functions?" in *AFRICACRYPT*, ser. LNCS, S. Vaudenay, Ed., vol. 5023. Springer, 2008, pp. 308–324.

[11] M. Liskov, R. L. Rivest, and D. Wagner, "Tweakable block ciphers." in *CRYPTO*, ser. LNCS, M. Yung, Ed., vol. 2442. Springer, 2002, pp. 31–46.

[12] R. L. Rivest, "Abelian square-free dithering for iterated hash functions," ECRYPT Workshop on Hash Functions, 2005.

[13] ——, "Abelian square-free dithering for iterated hash functions," NIST Hash Function Workshop, 2005.

[14] E. Andreeva, C. Bouillaguet, P.-A. Fouque, J. J. Hoch, J. Kelsey, A. Shamir, and S. Zimmer, "Second preimage attacks on dithered hash functions," in *EUROCRYPT*, ser. LNCS, N. P. Smart, Ed., vol. 4965. Springer, 2008, pp. 270–288.

[15] M. Stam, "Another glance at blockcipher based hashing," Cryptology ePrint Archive, Report 2008/071, 2008.

[16] T. Shrimpton and M. Stam, "Building a collision-resistant compression function from non-compressing primitives," in *ICALP (2)*, ser. LNCS, L. Aceto, I. Damgård, L. A. Goldberg, M. M. Halldórsson, A. Ingólfsdóttir, and I. Walukiewicz, Eds., vol. 5126. Springer, 2008, pp. 643–654.

[17] P. Rogaway and J. P. Steinberger, "Security/efficiency tradeoffs for permutation-based hashing," in *EUROCRYPT*, ser. LNCS, N. P. Smart, Ed., vol. 4965. Springer, 2008, pp. 220–236.

[18] ——, "Constructing cryptographic hash functions from fixed-key blockciphers," in *CRYPTO*, ser. LNCS, D. Wagner, Ed., vol. 5157. Springer, 2008, pp. 433–450.

[19] D. G. Filho, P. Barreto, and V. Rijmen, "The Maelstrom-0 hash function," in *6th Brazilian Symposium on Information and Computer Security*, 2006.

[20] L. R. Knudsen, C. Rechberger, and S. S. Thomsen, "The Grindahl hash functions," in *FSE*, ser. LNCS, A. Biryukov, Ed., vol. 4593. Springer, 2007, pp. 39–57.

[21] D. Wagner, "A slide attack on SHA-1," Unpublished manuscript, June 2001.

[22] M.-J. O. Saarinen, "Cryptanalysis of block ciphers based on SHA-1 and MD5." in *FSE*, ser. LNCS, T. Johansson, Ed., vol. 2887. Springer, 2003, pp. 36–44.

[23] M. Gorski, S. Lucks, and T. Peyrin, "Slide attacks on a class of hash functions," in *ASIACRYPT*, ser. LNCS, J. Pieprzyk, Ed., vol. 5350. Springer, 2008, pp. 143–160.

[24] R. D. Dean, "Formal aspects of mobile code security." Ph.D. dissertation, Princeton University, 1999.

[25] A. Biryukov and D. Wagner, "Slide attacks." in *FSE*, ser. LNCS, L. R. Knudsen, Ed., vol. 1636. Springer, 1999, pp. 245–259.

[26] ——, "Advanced slide attacks." in *EUROCRYPT*, ser. LNCS, B. Preneel, Ed., vol. 1807. Springer, 2000, pp. 589–606.

[27] J. Black, M. Cochran, and T. Shrimpton, "On the impossibility of highly-efficient blockcipher-based hash functions." in *EUROCRYPT*, ser. LNCS, R. Cramer, Ed., vol. 3494. Springer, 2005, pp. 526–541.

[28] S. Matyas, C. Meyer, and J. Oseas, "Generating strong one-way functions with cryptographic algorithm," *IBM Technical Disclosure Bulletin*, vol. 27, no. 10A, pp. 5658–5659, 1985.

[29] S. Miyaguchi, K. Ohta, and M. Iwata, "128-bit hash function (N-Hash)." *NTT Review*, vol. 2, pp. 128–132, 1990.

[30] B. Preneel, "Analysis and design of cryptographic hash functions." Ph.D. dissertation, Katholieke Universiteit Leuven, January 1993.

[31] P. Barreto and V. Rijmen, "The Whirlpool hashing function," First Open NESSIE Workshop, 2000.

[32] J. Kilian and P. Rogaway, "How to protect DES against exhaustive key search." in *CRYPTO*, ser. LNCS, N. Koblitz, Ed., vol. 1109. Springer, 1996, pp. 252–267.

[33] ——, "How to protect DES against exhaustive key search (an analysis of DESX)." *Journal of Cryptology*, vol. 14, no. 1, pp. 17–35, 2001.

[34] M. Johnson and K. Ramchandran, "Dither-based secure image hashing using distributed coding," in *ICIP*, vol. 2. IEEE, 2003, pp. 751–754.

[35] A. Swaminathan, Y. Mao, and M. Wu, "Robust and secure image hashing," *IEEE Transactions on Information Forensics and Security*, vol. 1, pp. 215–230, 2006.

[36] R. Cramer, Ed., *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, ser. LNCS, vol. 3494. Springer, 2005.

[37] M. Yung, Ed., *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, ser. LNCS, vol. 2442. Springer, 2002.

[38] N. P. Smart, Ed., *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, ser. LNCS, vol. 4965. Springer, 2008.