A note on the security proof of Knudsen-Preneel construction of a hash function

Dai Watanabe[1]

Systems Development Laboratory, Hitachi, Ltd.

daidai@sdl.hitachi.co.jp

Abstract

In this paper two attacks on a multiple length hash function whose construction is proposed by Knudsen and Preneel. One can violates the security bound claimed in the proposal paper [6] if $t = 1$ and $d > 3$.
Keywords. Hash function, Collision resistance, Linear code, Differential cryptanalysis

1 Introduction

A cryptographic hash function is one of the important cryptographic primitives. It compresses data of arbitrary length into a bit string of fixed length. It has a lot of applications such as a message authentication and a digital signature.

In these applications, hash functions are required to achieve some security criteria. The following three properties are the criteria which should be satisfied:

- one-wayness (OW): for all outputs, it is computationally infeasible to find any input whose hash value is equal to that output;
- second pre-image resistance (SPR): for all inputs, it is computationally infeasible to find a second input whose hash value is equal to that input;
- collision resistance (CR): it is computationally infeasible to find two distinct inputs whose hash values collide.

The detailed definition of the term "computationally infeasible" is not given in this paper because it is not essential for our discussion. Readers who intend to know it can refer [1], wherein their modern definitions are given.

These properties are mostly dependent on the output length of hash functions (hash length). For example, about 2^m calculations of a hash function are necessary to find a pre-image of an output if the hash length is m bits because the possible number of outputs is exactly 2^m. Nearly equal complexity is necessary to find a second pre-image. In the case of collision resistance, there is a well known technique to find a collision whose computational complexity is significantly smaller than that of finding a second pre-image. This technique is called *birthday attack* and the complexity is about $2^{m/2}$, the square root of the exhaustive trial (See [9] for the detail of a birthday attack).

A common way to construct a hash function is to apply a compression function iteratively, which maps a bit string of a fixed length to another bit string of a fixed (and shorter) length. Merkle [8] and Damgård [2] proposed a chaining construction (MD-construction) defined as follows and proved that it is secure if the underlying compression function is secure:

$$\begin{aligned} M &= M_1||M_2||\ldots||M_n, \\ H_0 &= Const, \\ H_i &= Compress(H_{i-1}, M_i) \quad \text{for} \quad 1 \leq i \leq n. \end{aligned}$$

Most of real hash functions are based on this scheme,

so that how to construct a secure compression function is a matter of concern in the study of hash functions.

There are many attempts to construct a compression function from a block cipher like as Data Encryption Standard (DES) [3] and Advanced Encryption Standard (AES) [5]. Matyas-Meyer-Oseas [7] and Davies-Meyer constructions are the examples that are secure if the underlying block cipher is secure as a block cipher. Preneel *et al.* comprehensively studied these constructions which call the underlying block cipher only once for each call of the compression function. They considered possible 64 constructions and resulted that 12 of them are secure as a compression function [10]. Black *et al.* added security proofs to these constructions by a current theoretical fashion [1].

These constructions are easy to implement if a block cipher is already used in the target system. On the other hand, the obvious weakness that the hash length is too short especially if DES is used as the underlying cipher. In addition, these constructions is much slower than dedicated hash functions.

The next objective is how to construct a compression function which achieves a better security than that of underlying compression functions, especially in the context of second pre-image resistance and collision resistance with a small supplement of operations. MDC-2 and MDC-4 are the double length hash functions, and MDC-4 achieves a better security. Knudsen and Preneel proposed a general method to construct a compression function whose security is better than that of underlying function under some acceptable assumptions. They proved that finding a collision of the proposed construction requires $2^{(d-1)m/2}$ calls of compression function [6].

In this paper we proposed an differential attack on Knudsen-Preneel construction of a secure hash function. This attack enable to find a collision less complexity than that claimed in [6].

The organization of the rest of this paper is as follows: Firstly necessary terms and the main contents of [6] including Knudsen-Preneel construction is given in Sect. 2 and the best known attack is introduced in Sect. 3. The detailed description of our new differential attack is given in Sect. 4. Sect. 5 presents yet another differential attack on Knudsen-Preneel construction. This attack does not violate the claimed security of Knudsen-Preneel construction, though it will help readers to understand the integral of the attack which should be considered. Finally the conclusion of this paper is presented in Sect. 6.

2 Knudsen-Preneel construction of a hash function

Knudsen and Preneel proposed how to construct a hash function from ideal compression functions (CFs), whose security (second pre-image resistance (SPR) and collision resistance (CR)) is really better than that of underlying CFs [6]. In this section, we introduce their construction.

2.1 Preliminaries

Before starting discussion we define terms and notations used in this paper.

Let m be the output length (hash length) of the underlying hash function h. Subscript j is accompanied and denoted by h_j if it is necessary to distinguish plural distinct CFs. For simplify the discussion let the input length of the CF h be multiple of hash length and let the multiple be $t+1$.

$$h : \{0,1\}^{tm} \times \{0,1\}^m \to \{0,1\}^m.$$

The first input of tm bits is the message input and the second input of m bits is the hash input. Essentially it is not necessary to divide the input of a CF into two distinct inputs: the hash input and the message input. But for convenience the notation above is used throughout this paper. The message input and the hash input are denoted by M_i and H_i respectively.

Definition 1 (Multiple Construction) Let $h_i(\cdot,\cdot)$ be ideal CFs and they are independent each other. Our target is *multiple construction* of CF

defined by:

$$\begin{aligned} H_i^1 &= h_1(X_i^1, Y_i^1), \\ H_i^2 &= h_2(X_i^2, Y_i^2), \\ \cdots & \\ H_i^n &= h_n(X_i^n, Y_i^n), \end{aligned}$$

where X_i^j, Y_i^j are linear combinations of H_{i-1}^j and $M_i^{j'}$.

We call h_j *subfunctions* (SCFs) of the multiple construction.

2.2 Security assumptions

Let $H_{i-1}^1, \ldots, H_{i-1}^n$, $M_i^1, \ldots, M_i^r$ and $H'^1_{i-1}, \ldots$, H'^n_{i-1}, $M'^1_i, \ldots, M'^r_i$ be the two distinct input to a multiple construction of CFs. *Active* inputs are defined as a set of pairs H_{i-1}^j and H'^j_{i-1} (or M_i^j and M'^j_i) such that $H_{i-1}^j \neq H'^j_{i-1}$ ($M_i^j \neq M'^j_i$). A CF h_j is called *active* if at least one of the input is active.

A set of CFs $h_{i_1}(X_{i_1}, Y_{i_1}), \ldots, h_{i_s}(X_{i_s}, Y_{i_s})$ can be attacked *independently* if for all $j \in \{1, \ldots, s\}$ it holds that: for all values of the input blocks affecting (X_{i_j}, Y_{i_j}) to h_{i_j} the arguments (X_{i_k}, Y_{i_k}) are fixed for $k \in \{1, \ldots, j-1, j+1, \ldots, s\}$.

Under the notations defined above we give the following assumption (this assumption is the quotation from [6] for the preciseness).

Assumption 1

1. The underlying CFs h_i are ideal functions.
2. What a collision for the CF of a multiple scheme has been found means it is found simultaneously for underlying SCFs $h_1, \ldots, h_n$.

2.3 Theorems of [6]

Theorem 1 ([6] Theorem 3) If there exists an $[n, k, d]$ linear code over GF(2^2) of length n, dimension k, and minimum distance d, with $2k > n$, for $m \gg \log_2 n$, then there exists a parallel hash function based on an ideal compression function $h : \{0,1\}^m \times \{0,1\}^m \to \{0,1\}^m$, for which finding a collision for the compression function requires at least $2^{(d-1)m/2}$ operations.

We follow a part of the proof because it includes the construction.

The CF consists of n different SCFs h_i with $1 \leq i \leq n$. The input to the CF consists of $2k$ m-bit blocks: the n hash blocks $H_{i-1}^1, \ldots, H_{i-1}^n$ (the output of the n subfunctions of the previous iteration) and the r message blocks $M_i^1, \ldots, M_i^r$. Note that $r = 2k - n > 0$ is the necessary condition for the CF to be a compression function, i.e., the input length is larger than the output length.

In Knudsen-Preneel construction firstly $2k$ m-bit is transformed into the concatenation of km elements of GF(2^2). For example j-th bits of H_{i-1}^1 and H_{i-1}^2 is treated as an element of GF(2^2). These km elements of GF(2^2) is mapped to nm elements of GF(2^2) by $[n, k, d]$ code over GF(2^2). These two pairs of two bits are separated into two distinct inputs of h_j. The hash value of the scheme is defined by the concatenation of the output H_i^j of all SCFs h_j, i.e., $H_i^1 || H_i^2 || \cdots || H_i^n$.

This construction bases its security on the basic property of $[n, k, d]$ linear code. An arbitrary input differential has an influence on at least $d - 1$ SCFs because the minimum distance of the code is d. In other word at least $d - 1$ SCFs are active so that Assumption 1 guarantees that more than $2^{(d-1)m/2}$ calculations are needed to find a collision.

A theorem of the same kind is approved for arbitrary SCFs.

Theorem 2 ([6] Theorem 4) Let b be a divisor of m. If there exists an $[n, k, d]$ linear code over GF($2^{(t+1)b}$) of length n, dimension k, and minimum distance d, with $(t+1)k > n$, for $m \gg \log_2 n$, then there exists a parallel hash function based on an ideal compression function $h : \{0,1\}^{tm} \times \{0,1\}^m \to \{0,1\}^m$, for which finding a collision for the compression function requires at least $2^{(d-1)m/2}$ operations.

The construction and the proof of the CF in Theorem 2 is given in almost the same manner as in

Theorem 1. Please refer [6] if the original descriptions and more detailed proofs of the theorems are needed.

We now call the hash functions of Theorem 1 and 2 as Knudsen-Preneel compression functions (KPCFs).

3 Generic attack (Inverse and collide)

[6] shows not only the construction and the security proof but also the claimed security is tight both for one-wayness and collision resistance. In this section the generic collision attack against KPCFs presented in [6] is introduced.

3.1 The attack

Definition 2 *multi-collision* is a input set which leads same a hash value:

$$\mathcal{S}_y \subset \{x \in \mathcal{D} | \mathcal{H}(x) = y\}.$$

In other words $\mathcal{S}_y$ is a subset of the inverse image of y.

Proposition 1 ([6] Proposition 4) Let $\mathcal{H}$ be a KPCF with $[n, k, d]$ linear code L over GF$(2^{(t+1)b})$ where $(t+1)k > n$. Then collisions for $\mathcal{H}$ can be found in

$$\max\left(2^{m(n-k/2)}, k \cdot 2^{\frac{(n+km)}{2k}}\right)$$

operations. The attack requires the storage of about $(t+1)k2^{(n-k)m/2k}$ m-bit values.

For the simplicity we assume that the code L is of the normal form. I.e., the hash value of the KPCF $\mathcal{H}$ is calculated as follows:

$$\begin{aligned} H_i^j &= h_j(M_i^{t(j-1)+1}, \ldots, M_i^{tj}, H_{i-1}^j) \quad 1 \le j \le k, \\ H_i^j &= h_j(L_j(M_i, H_{i-1})) \quad k+1 \le j \le n. \end{aligned}$$

It is clear that $h_1, \ldots, h_k$ can be attacked independently.

Algorithm 1 Generic attack on a KPCF [6]

Step 1. Calculate multi-collisions $S_1, \ldots, S_k$ for each SCFs $h_1, \ldots, h_k$.

Step 2. Search for a collision pair of $n-k$ SCFs $h_{k+1}, \ldots, h_n$ simultaneously by birthday attack with the input taken from $S = S_1 \times \cdots \times S_k$.

Algorithm 1 shows the outline of generic attack on KPCFs. In the generic attack firstly the attacker generates a set of multi-collision S_j for each SCFs h_j with $1 \le j \le k$. Secondly he searches for a collision for $h_{k+1}, \ldots, h_n$ simultaneously with elements of $S = S_1 \times \cdots \times S_k$. Any two elements of S lead same output for $h_1, \ldots, h_k$ so that a collision pair for SCFs $h_{k+1}, \ldots, h_n$ collides on all SCFs.

3.2 Complexity of the attack

By Assumption 1 it is necessary to search for a collision for SCFs $h_{k+1}, \ldots, h_n$ simultaneously in Step 2 so that the input set S must include more than $2^{(n-k)m/2}$ elements. The sufficient condition for the number of the element of S_j is $(\#S)^{1/k} = 2^{(n-k)m/2k}$ for each S_j which is the multi-collision for SCF h_j.

As a result, the calculation complexity of Step 1 for collecting multi-collision for $h_1, \ldots, h_k$ is $2^{(n-k)m/2k} \cdot 2^m = 2^{(n+k)m/2k}$ hash calculations. In the Step 2 $2^{(n-k)m/2}$ hash calculations is necessary to find a collision for $n-k$ SCFs $h_{k+1}, \ldots, h_n$ simultaneously.

4 Differential attack

In this section, we show an attack on KPCF which is a counter example for Theorem 1. It is just a differential attack. For a simple discussion, we assume $k > n-k$. Note that this assumption holds for all linear codes applied in Theorem 1 because $2k - n > 0$ is satisfied.

Let $h : \{0,1\}^{tm} \times \{0,1\}^m \to \{0,1\}^m$ be a SCF and L be an $[n, k, d]$ linear code over GF(2^{t+1}). An input to L is denoted by $X = (X_1, \ldots, X_k)$ and the corresponding output is denoted by $Y = (Y_1, \ldots, Y_n)$

Algorithm 2 Differential attack on a KPCF

Step 1. Consider a system of linear equations

$$L_i(X_1, \ldots, X_k) \quad = \quad 0, \quad k+1 \le i \le n. \quad (3)$$

Because of the assumption $k > n-k$, this system has a non-trivial solution $\Delta = (\Delta_1, \ldots, \Delta_k)$. Fix the solution Δ, then the encoded differential is $L(\Delta) = (\Delta_1, \ldots, \Delta_k, 0, \ldots, 0)$.

Step 2. For each SCF h_i $(1 \le i \le k)$, apply differential attack using differential Δ_i. In other words, calculate hash values of X and $X \oplus \Delta_i$ with the fixed Δ_i and variable Xs until $h_i(X)$ and $h_i(X \oplus \Delta_i)$ collide. Denote the collision pair of h_i by $(A_i, A_i \oplus \Delta_i)$.

Step 3. Let A is the concatenated vector consisting of A_i, i.e., $A = (A_1, \ldots, A_k)$. Then A and $A \oplus \Delta)$ are colliding pair.

where X_i and Y_i are elements of $\mathrm{GF}(2^{t+1})^m$. Then the encoding of L is described as follows:

$$Y_i \quad = \quad L_i(X_1, \ldots, X_k), \quad 1 \le i \le n. \quad (1)$$

We can assume that

$$L_i(X_1, \ldots, X_k) \quad = \quad X_i, \quad 1 \le i \le k. \quad (2)$$

4.1 The attack

Algorithm 2 shows the process of differential attack on KPCFs for $t = 1$. The step 3 holds basing on the properties of A and Δ as follows:

$$\begin{aligned}
& KPCF(A \oplus \Delta) \\
= \; & (h_1 \circ L_1(A \oplus \Delta), \ldots, h_k \circ L_k(A \oplus \Delta), \\
& h_{k+1} \circ L_{k+1}(A \oplus \Delta), \ldots, h_n \circ L_n(A \oplus \Delta)) \\
= \; & (h_1(A_1 \oplus \Delta_1), \ldots, h_k(A_k \oplus \Delta_k), \\
& h_{k+1}(L_{k+1}(A) \oplus L_{k+1}(\Delta)), \\
& \ldots, h_n(L_n(A) \oplus L_n(\Delta))) \\
= \; & (h_1(A_1), \ldots, h_k(A_k), \\
& h_{k+1} \circ L_{k+1}(A), \ldots, h_n \circ L_n(A)) \\
= \; & KPCF(A). \quad (4)
\end{aligned}$$

Note that $k > n-k$ is not always satisfied if $t > 1$, so that it is possible that this attack cannot be applied to KPCFs with $t > 1$. Let $\Delta = (\Delta_1, \ldots, \Delta_k)$ be a solution of Eq. 3. If $k > n-k$ is not satisfied the input of $h_{k+1}, \ldots, h_{n-k}$ cannot be handled independently so that the attack is not applicable. It may be fortunate for KPCFs that the standard block ciphers DES and AES support double length key size.

Contrarily the attack is applicable if $k > n-k$ is satisfied and is independent from t. Therefore it is desirable carefully to choose the parameter k, n and t (and the code) such that $t > 1$, $2k \le n < (t+1)k$.

4.2 Complexity of the attack

Now we estimate the computational complexity of this attack.

The complexity of the first step is negligible. We have to calculate about 2^m input pairs to find a collision for each SCF h_i for $1 \le i \le k$ in the second step. Note that we can attack distinct SCFs independently because we can assume that only X_i is the input of h_i for $1 \le i \le k$. Hence $k \cdot 2^m$ operations of SCFs is required for the second step. The third step is costless.

Therefore the total complexity of this attack is nearly equal to that of the second step, about $k \cdot 2^m$. This violates the security bound of Theorem 1 and 2 when $d > 3$.

5 Yet another differential attack

In this section we present another differential attack, which is combined with a birthday attack. It is not so effective as other proposed attacks, however it appeals that a combination of differential attacks and birthday attacks is somehow possible.

5.1 Differential birthday attack

The first idea is how to choose a set whose any two elements have a desirable differential property. Let

x be an element of $\mathrm{GF}(2^{t+1})^k$ such that the Hamming weight of $L(x)$ is equal to d. Then

$$\Delta(x) := \{ax | a \in \mathrm{GF}(2^{(t+1)})\}$$

is a set satisfying the desirable property: for all $x, y \in \Delta(x)$ $x \oplus y$ is a element of $\Delta(x)$.

If $t + 1 > d/2$, we can apply birthday attacks to d SCFs. Note that attacks must be done simultaneously because all elements of $\Delta(x)$ is a multiple of x. The total complexity of this attack is $2^{dm/2}$.

If t is not sufficiently large to satisfy above condition, we can extend the differential vector space Δ by relaxing the condition. In fact, $\Delta(x)$ can be represented by solutions of a system of linear equations. Deleting some equations from the system relaxes the condition, so that the resultant vector space Δ consisting of solutions of remaining equations becomes larger. On the other hand, the maximum Hamming weight of the differences (elements of Δ) becomes larger too, as some linear equations are not satisfied any more.

5.2 Calculation complexity of DBA

If we delete c linear equations over $\mathrm{GF}(2^{t+1})$, $\max\{\mathrm{ham}(\delta) | \delta \in \Delta\} = d+c$. Besides, the dimension of the vector space Δ over $\mathrm{GF}(2)$ is $(t + 1)(c + 1)$. Hence, if we choose sufficiently large c such that $(t+1)(c+1) \geq (d+c)/2$, a birthday attack on $d+c$ active SCFs is applicable. In this case, the complexity of the attack is $2^{(d+c)m/2}$.

6 Conclusion

In this paper two attacks on a Knudsen-Preneel construction of a hash function are proposed. One can violates the security bound claimed in the proposal paper [6] if $t = 1$ and $d > 3$. For the case $t > 1$, the condition for the security of Knudsen-Preneel construction to be decreased is not clear as the case $t = 1$.

References

[1] J. Black, P. Rogaway, and T. Shrimpton, "Black-box analysis of the block-cipher-based hash-function constructions from PGV," *Advances in Cryptology, CRYPTO 2002*, Springer-Verlag, LNCS 2442, pp. 320–335, 2002.

[2] I. Damgård, "A design principle for hash functions," *Advances in Cryptology, CRYPTO '89*, Springer-Verlag, LNCS 435, pp. 416–427, 1990.

[3] FIPS 46-3, "Data Encryption Standard," Federal Information Processing Standards Publication, National Institute of Standards and Technology, 1997 (revised as FIPS 46-1: 1988; FIPS 46-2: 1993; FIPS 46-3: 1999).

[4] FIPS 180-2, "Secure Hash Standard," Federal Information Processing Standards Publication, National Institute of Standards and Technology, 1993 (revised as FIPS 180-1: 1995; FIPS 180-2: 2002).

[5] FIPS 197, "Advanced Encryption Standard," Federal Information Processing Standards Publication, National Institute of Standards and Technology, 2001.

[6] L. Knudsen and B. Preneel, "Construction of secure and fast hash functions using nonbinary error-correcting codes," *IEEE TRANSACTIONS ON INFORMATION THEORY*, Vol. 48, No. 9, pp. 1–17, 2002.

[7] S. Matyas, C. Meyer, and J. Oseas, "Generating strong one-way functions with crypographic algorithm," *IBM Tech. Discl. Bull.*, vol. 27, no. 10A, pp. 5658–5659, 1985.

[8] R. Merkle, "One way hash functions and DES," *Advances in Cryptology, CRYPTO '89*, Springer-Verlag, LNCS 435, pp. 428–446, 1990.

[9] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC-Press, 1996.

[10] B. Preneel, R. Govaert, and J. Vandewalle, "Hash functions based on block ciphers: A synthetic approach," *Advances in Cryptology, CRYPTO 93*, Springer-Verlag, LNCS 773, pp. 368–378, 1994.