A Collision Attack on a Double-Block-Length Compression Function Instantiated with 8-/9-Round AES-256

PAPER Special Section on Cryptography and Information Security

A Collision Attack on a Double-Block-Length Compression Function Instantiated with 8-/9-Round AES-256*

Jiageng CHEN[†], **Shoichi HIROSE**[††a)], ***Members***, **Hidenori KUWAKADO**[†††], ***Senior Member***, ***and*** **Atsuko MIYAJI**[††††,†††††,††††††], ***Member***

SUMMARY This paper presents the first non-trivial collision attack on the double-block-length compression function presented at FSE 2006 instantiated with round-reduced AES-256: $f_0(h_0\|h_1, M)\|f_1(h_0\|h_1, M)$ such that

$$f_0(h_0\|h_1, M) = E_{h_1\|M}(h_0) \oplus h_0 \ ,$$
$$f_1(h_0\|h_1, M) = E_{h_1\|M}(h_0 \oplus c) \oplus h_0 \oplus c \ ,$$

where $\|$ represents concatenation, E is AES-256 and c is a 16-byte non-zero constant. The proposed attack is a free-start collision attack using the rebound attack proposed by Mendel et al. The success of the proposed attack largely depends on the configuration of the constant c: the number of its non-zero bytes and their positions. For the instantiation with AES-256 reduced from 14 rounds to 8 rounds, it is effective if the constant c has at most four non-zero bytes at some specific positions, and the time complexity is 2^{64} or 2^{96}. For the instantiation with AES-256 reduced to 9 rounds, it is effective if the constant c has four non-zero bytes at some specific positions, and the time complexity is 2^{120}. The space complexity is negligible in both cases.
key words: *double-block-length compression function, free-start collision attack, rebound attack, AES-256*

1. Introduction

(1) Background.

Cryptographic hash functions are very important primitives and used in almost all cryptographic protocols. They are often called hash functions, and we follow this convention.

There are several design strategies of hash functions, and the most popular ones are block-cipher-based and permutation-based. The block-cipher-based approach is much more classical than the permutation-based approach. The permutation-based approach is fairly new, and the SHA-3 Keccak [4] is designed with the approach. Well-known hash functions such as MD5 [36], SHA-1 and SHA-2 [14] can be regarded as being designed with the block-cipher-based approach using dedicated block ciphers. Hash functions MDC-2 and MDC-4 [8] using DES predate them.

How to construct secure hash functions using a block cipher has been an important research topic [6], [33]. When using existing block ciphers such as AES, one should adopt double-block-length construction [8], [17], [18], [24], [31] for sufficient level of collision-resistance. Hash functions using AES [11], [15] may be an option for high-end CPUs with AES-NI and low-end microcontrollers.

(2) Our Contribution.

This paper presents a non-trivial collision attack on the double-block-length (DBL) compression function [18] instantiated with round-reduced AES-256. As far as the authors know, this is the first collision attack on the DBL compression function instantiated with AES-256. The DBL compression function is defined as $f_0(h_0\|h_1, M)\|f_1(h_0\|h_1, M)$ such that

$$f_0(h_0\|h_1, M) = E_{h_1\|M}(h_0) \oplus h_0 \ ,$$
$$f_1(h_0\|h_1, M) = E_{h_1\|M}(h_0 \oplus c) \oplus h_0 \oplus c \ ,$$

where $\|$ represents concatenation, E is AES-256 and c is a non-zero constant. The proposed collision attack assumes that the final round of round-reduced AES-256 does not have the `MixColumns` operation.

The proposed collision attack makes use of the following fact [9]: if $(h_0\|h_1, M)$ and $((h_0 \oplus c)\|h_1, M)$ are a colliding pair for f_0, then they are also a colliding pair for f_1. The rebound attack [25] is used to find such a colliding pair for f_0. Thus, its success largely depends on the configuration of the 16-byte constant c: the number of its non-zero bytes and their positions. For the instantiation with 8-round AES-256, the attack finds a colliding pair of inputs with time complexity 2^{64} or 2^{96} if the constant c has at most four non-zero bytes at some specific positions. For the instantiation with 9-round AES-256, it finds a colliding pair of inputs with time complexity 2^{120} if the constant c has four non-zero bytes at some specific positions. The space complexity is negligible in both cases.

Manuscript received March 23, 2015.
Manuscript revised August 19, 2015.
[†]The author is with Computer School, Central China Normal University, Wuhan, 430079, China.
[††]The author is with Graduate School of Engineering, University of Fukui, Fukui-shi, 910-8507 Japan.
[†††]The author is with Faculty of Informatics, Kansai University, Takatsuki-shi, 569-1095 Japan.
[††††]The author is with Graduate School of Engineering, Osaka University, Suita-shi, 565-0871 Japan.
[†††††]The author is with School of Information Science, Japan Advanced Institute of Science and Technology, Nomi-shi, 923-1292 Japan.
[††††††]The author is with CREST, JST, Kawaguchi-shi, 332-0012 Japan.
*A preliminary version of this paper was presented at ICISC 2014 [10]. This paper concentrates on the collision attack against the instantiation with 8-/9-round AES-256. The contents in Sect. 4.4 of the paper do not appear in the preliminary version.
a) E-mail: hrs_shch@u-fukui.ac.jp
DOI: 10.1587/transfun.E99.A.14

(3) Related Work.

The rebound attack was proposed by Mendel et al. [30], and was applied to the hash functions Whirlpool [34] and Grøstl [21], which have similar structure to AES. The rebound attack on Whirlpool was further improved by Lamberger et al. [25]. The rebound attack was also applied to a few other SHA-3 finalists [12], [20], [35].

There is some work on cryptanalyses of single-block-length hashing modes of AES. Biryukov, Khovratovich and Nikolić [5] presented a q-multicollision attack on the Davies-Meyer (DM) compression function instantiated with full-round AES-256. It is very powerful and its time complexity is $q \cdot 2^{67}$. Mendel et al. [29] presented a collision attack on the DM compression function instantiated with 5-round AES-128 with time complexity 2^{56}. The collision attack on 5.5-round Whirlpool [25] can easily be extended to a collision attack on the DM, Matyas-Meyer-Oseas (MMO), Miyaguchi-Preneel (MP) compression functions instantiated with 6-round AES-128 or the DM compression function instantiated with 6-round AES-192/256. Its time complexity is 2^{56}. Jean, Naya-Plasencia and Peyrin [19] presented a collision attack on the DM compression function instantiated with 6-round AES-128 with time complexity 2^{32}. Sasaki presented preimage and second-preimage attacks on DM, MMO and MP modes of 7-round AES [37].

There is little work on cryptanalyses of instantiations of DBL hashing modes. Ferguson [13] presented a few generic attacks on H-PRESENT-128 [7]. Kobayashi and Hirose [22] presented a collision attack on H-PRESENT-128 instantiated with 10-round PRESENT with time complexity 2^{60}. Wei et al. [38] presented collision and preimage attacks on various hashing modes instantiated with the block cipher IDEA [23]. They concluded that IDEA should not be used for hashing. The hashing modes include the DBL modes such as Abreast-DM, Tandem-DM [24], the mode by Hirose [18], the mode by Peyrin et al. [32] and MJH [27]. Our proposed collision attack is unlikely to be applied to them except for the Hirose mode.

The collision resistance and the preimage resistance were provided proofs in the ideal cipher model for Abreast-DM [3], [16], [26], Tandem-DM [3], [28], the Hirose compression function [3], [18]. In particular, Abreast-DM and the Hirose compression function were shown to be optimally collision-resistant in the ideal cipher model.

Chang [9] pointed out that, for the Hirose compression function, the constant c can be used as a backdoor by first constructing a collision for f_0 such that $f_0(h_0 \| h_1, M) = f_0(h'_0 \| h_1, M)$ with complexity of 2^{64} and then choosing $c = h_0 \oplus h'_0$. Attacks in the similar setting have recently been proposed for SHA-1 [1] and GOST [2].

(4) Organization.

A brief description of AES is given in Sect. 2. An overview of the proposed collision attack on the DBL compression function is described in Sect. 3. The collision attacks on the compression function instantiated with AES-256 of 8 rounds and 9 rounds are detailed in Sect. 4 and Sect. 5, respectively. A concluding remark is given in Sect. 6.

2. Preliminaries

2.1 AES

This section gives a description of AES [11], [15] together with some properties of its components necessary for the discussions later.

AES is a block cipher with 128-bit block length and 128/192/256-bit key length. AES with κ-bit key length is often denoted by AES-κ.

The transformations of AES are performed on a (4×4)-byte array called the state. Each byte is regarded as an element in $GF(2^8)$. Multiplication is performed modulo $\mathtt{x}^8 + \mathtt{x}^4 + \mathtt{x}^3 + \mathtt{x} + 1$. The state is initially a plaintext.

The encryption of AES consists of four transformations: `SubBytes`, `ShiftRows`, `MixColumns` and `AddRoundKey`. It starts with the `AddRoundKey` transformation followed by iteration of a round function. The round function applies `SubBytes`, `ShiftRows`, `MixColumns` and `AddRoundKey` transformations in this order to the state. The final round does not have the `MixColumns` transformation. AES-128/192/256 have 10/12/14 rounds, respectively.

The `SubBytes` transformation is byte-wise application of the nonlinear S-box function. For the S-box S, an input x satisfying the equation $\mathsf{S}(x) \oplus \mathsf{S}(x \oplus \Delta I) = \Delta O$ is called an admissible input for the pair of an input difference ΔI and an output difference ΔO. We will say that an input difference and an output difference are compatible with each other if there exist admissible inputs for the pair.

Table 1 shows the numbers of the pairs of input and output differences which have the specified numbers of admissible inputs. The probability that there exist any admissible inputs for a pair of input and output differences $(\Delta I, \Delta O)$ chosen uniformly at random is about 1/2. An admissible input of the `SubBytes` transformation is defined similarly.

The `ShiftRows` transformation is byte-wise cyclic transposition of each row. It shifts the i-th row by i-bytes cyclically to left for $0 \le i \le 3$.

The `MixColumns` transformation is linear transformation of each column. It can be represented with a matrix. For a 4-byte column b of a state, it is represented by Tb, where

$$T = \begin{pmatrix} \mathtt{02} & \mathtt{03} & \mathtt{01} & \mathtt{01} \\ \mathtt{01} & \mathtt{02} & \mathtt{03} & \mathtt{01} \\ \mathtt{01} & \mathtt{01} & \mathtt{02} & \mathtt{03} \\ \mathtt{03} & \mathtt{01} & \mathtt{01} & \mathtt{02} \end{pmatrix} \quad \text{and}$$

Table 1 Correspondence between the number of input-/output-difference pairs and the number of their admissible inputs for the AES S-box.

No. of admissible inputs	0	2	4	256
No. of difference pairs	33150	32130	255	1

$$T^{-1} = \begin{pmatrix} \mathtt{0e} & \mathtt{0b} & \mathtt{0d} & \mathtt{09} \\ \mathtt{09} & \mathtt{0e} & \mathtt{0b} & \mathtt{0d} \\ \mathtt{0d} & \mathtt{09} & \mathtt{0e} & \mathtt{0b} \\ \mathtt{0b} & \mathtt{0d} & \mathtt{09} & \mathtt{0e} \end{pmatrix} .$$

The `AddRoundKey` transformation is bitwise XOR of a round key to a state. The round keys are generated by a key expansion algorithm. The round keys of AES-256 are generated in the following way. Let (4×4)-byte array K_r be the round key of the r-th round for $r \geq 0$, where K_0 is for the initial `AddRoundKey` transformation. The 256-bit key input is given to K_0 and K_1. Let $K_r[j]$ be the j-th column of K_r for $0 \leq j \leq 3$. For $r \geq 2$, if r is even, then

$$K_r[0] = K_{r-2}[0] \oplus \mathsf{SW}(K_{r-1}[3]^{\uparrow}) \oplus C_r \ ,$$
$$K_r[j] = K_{r-2}[j] \oplus K_r[j-1] \quad \text{for } 1 \leq j \leq 3 \ ,$$

where SW represents byte-wise application of the AES S-box, $K_{r-1}[3]^{\uparrow}$ represents cyclic 1-byte shift of $K_{r-1}[3]$ to the top, and C_r is a specified constant. If r is odd, then

$$K_r[0] = K_{r-2}[0] \oplus \mathsf{SW}(K_{r-1}[3]) \ ,$$
$$K_r[j] = K_{r-2}[j] \oplus K_r[j-1] \quad \text{for } 1 \leq j \leq 3 \ .$$

For simplicity, the transformations `SubBytes`, `ShiftRows`, `MixColumns` and `AddRoundKey` are denoted by SB, SR, MC and AK, respectively.

The state in the r-th round is denoted by S_r. S_r^{SB}, S_r^{SR}, S_r^{MC} and S_r^{AK} represent the state S_r just after SB, SR, MC and AK transformations, respectively. S_{-1} represents a plaintext input.

For $0 \leq i \leq 3$ and $0 \leq j \leq 3$, $S_r[i][j]$ represents the byte of S_r in the i-th row and the j-th column. $S_r[j]$ represents the j-th column of S_r.

3. Overview

This section gives an overview of the proposed free-start collision attack on a DBL compression function [18] instantiated with round-reduced AES-256. The target DBL compression function

$$v_0 \| v_1 = F(h_0 \| h_1, M) = f_0(h_0 \| h_1, M) \| f_1(h_0 \| h_1, M)$$

is defined by

$$f_0(h_0 \| h_1, M) = E_{h_1 \| M}(h_0) \oplus h_0 \ ,$$
$$f_1(h_0 \| h_1, M) = E_{h_1 \| M}(h_0 \oplus c) \oplus h_0 \oplus c \ ,$$

where $\|$ represents concatenation, E is a block cipher and c is a non-zero constant. F is depicted in Fig. 1.

A free-start collision attack on a compression function is successful if it simply finds a colliding pair of inputs of the compression function, that is, a pair of distinct inputs mapped to the same output.

To find a collision of F, the proposed attack uses the following fact [9]:

Fact 1: Suppose that a collision for f_0 is caused by $(h_0 \| h_1, M)$ and $((h_0 \oplus \Delta h_0) \| h_1, M)$, that is, $f_0(h_0 \| h_1, M) = f_0((h_0 \oplus \Delta h_0) \| h_1, M)$ and that $\Delta h_0 = c$. Then, a collision for f_1 is also caused by $(h_0 \| h_1, M)$ and $((h_0 \oplus \Delta h_0) \| h_1, M)$.

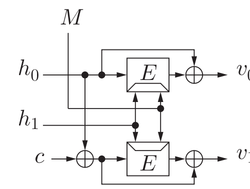


Fig. 1 The target DBL compression function. c is a non-zero constant and E is a block cipher.

The algorithm of the collision attack on F is given below:

1. Find a colliding pair of inputs $(h_0 \| h_1, M)$ and $((h_0 \oplus \Delta h_0) \| h_1, M)$ for f_0.

2. Output $(h_0 \| h_1, M)$ and $((h_0 \oplus \Delta h_0) \| h_1, M)$ if $\Delta h_0 = c$. Otherwise, return to Step 1.

The first step returns a colliding pair of inputs for f_0 such that the non-zero bytes of Δh_0 are located at the same positions as the non-zero bytes of the constant c. Sections 4 and 5 present how the collision attack is applied to F instantiated with AES-256 of 8 and 9 rounds, respectively.

4. Collision Attack on Instantiation with 8-Round AES-256

This section first presents a free-start collision attack on f_0 instantiated with 8-round AES-256. It returns a pair of colliding inputs with difference Δh_0 whose bytes are zero except for the first byte. The time complexity is 2^{56}, and the space complexity is negligible. The probability that $\Delta h_0 = c$ is 2^{-8} if c has a single non-zero byte at the same position as the non-zero byte of Δh_0. Thus, the total time complexity of the collision attack on F instantiated with 8-round AES-256 is 2^{64}.

This section also shows by an exhaustive search that the attack mentioned above can be extended to apply to other constants c with at most four non-zero bytes at some specific positions.

4.1 Collision Attack on f_0

The collision attack on f_0 is based on the rebound attack on the 7.5-round Whirlpool compression function by Lamberger et al. [25]. The goal of the attack is to find a pair of inputs, $(h_0 \| h_1, M)$ and $(h_0' \| h_1, M)$, which follow the differential path given in Fig. 2.

The proposed attack uses two inbound phases: the first one is in the second and the third rounds, and the second one is in the fifth and the sixth rounds. The algorithm of the attack is described below. It first selects the values of differences of the two inbounds (Steps 1 and 2) and those between

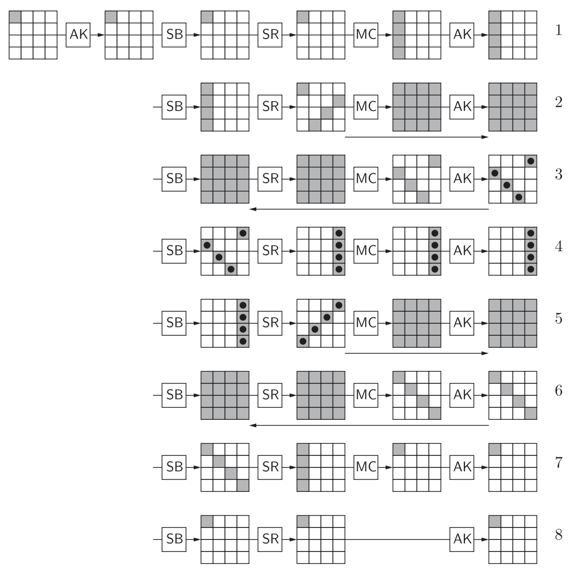


Fig. 2 The differential path used by the collision attack on the compression function f_0 instantiated with 8-round AES-256. Colored bytes are non-zero differences.

the two inbounds (Step 3). Then, for each pair of an admissible input of SB in the third round and that of SB in the sixth round, it connects them with the round keys (Steps 4a to 4c), and extends the state transformation to the outbounds to check if a colliding pair of inputs are obtained (Steps 4d to 4f).

1. This step looks for a pair of compatible input/output differences of `SubBytes` of the third round in the following way:

   a. Select the non-zero differences in ΔS_2^{SR} and $\Delta S_3^{\mathsf{AK}} = \Delta S_3^{\mathsf{MC}}$ uniformly at random. Then, compute

   $$\Delta S_2^{\mathsf{AK}} = \Delta S_2^{\mathsf{MC}} = \mathsf{MC}(\Delta S_2^{\mathsf{SR}}) \ ,$$
   $$\Delta S_3^{\mathsf{SB}} = \mathsf{SR}^{-1}(\mathsf{MC}^{-1}(\Delta S_3^{\mathsf{MC}})) \ .$$

   For linear transformations, output differences are uniquely determined only by the input differences.

   b. If there are no admissible inputs for the pair of ΔS_2^{AK} and ΔS_3^{SB}, then return to Step 1a.

   The expected number of repetitions of this step is 2^{16}. The number of admissible inputs obtained for S_2^{AK} with this step is 2^{16}. Actually, this step can be made more efficient since the trials can be done column by column. However, this speed-up does not change the time complexity of the overall algorithm.

2. This step looks for a pair of compatible input/output differences of `SubBytes` of the sixth round in the same way as the step 1. 2^{16} admissible inputs are obtained for S_5^{AK} with this step.

3. Select ΔS_4^{SB} compatible with ΔS_3^{AK} uniformly at random until $\Delta S_4^{\mathsf{AK}} = \Delta S_4^{\mathsf{MC}} = \mathsf{MC}(\mathsf{SR}(\Delta S_4^{\mathsf{SB}}))$ is compatible with ΔS_5^{SB}. The expected number of repetitions of this step is 2^4.

4. Perform the following procedure:

   a. Select a new pair among the 2^{32} pairs of S_2^{AK} and S_5^{AK}. If there exists no new pair, then return to Step 1.

   b. Compute $S_3^{\mathsf{SB}} = \mathsf{SB}(S_2^{\mathsf{AK}})$. Then, run the algorithm for connecting two inbound phases, which is given in Sect. 4.2, and obtain the round keys K_3, K_4 and K_5.

   c. Compute the round keys K_0, K_1, K_2, K_6 and K_7.

   d. Compute the corresponding input S_{-1} to AES and the difference ΔS_{-1} from S_2^{AK} and ΔS_2^{AK}. If any byte of ΔS_{-1} other than $\Delta S_{-1}[0][0]$ is non-zero, then return to Step 4a.

   e. Compute the corresponding output S_8^{AK} from AES and the difference ΔS_8^{AK} from S_5^{AK} and ΔS_5^{AK}. If any byte of ΔS_8^{AK} other than $\Delta S_8^{\mathsf{AK}}[0][0]$ is non-zero, then return to Step 4a.

   f. If $\Delta S_{-1} = \Delta S_8^{\mathsf{AK}}$, then proceed to Step 5. Otherwise, return to Step 4a.

5. Output the pair of inputs (K, S_{-1}) and $(K, S_{-1} \oplus \Delta S_{-1})$, which are mapped to the same hash value by f_0 instantiated with 8-round AES-256, where $K = K_0 \| K_1$.

For Step 4d in the algorithm above, the probability that only $\Delta S_{-1}[0][0]$ is non-zero (the transition from ΔS_1^{MC} to ΔS_1^{SR} is successful) is 2^{-24}. Similarly, for Step 4e, the probability that only $\Delta S_8^{\mathsf{AK}}[0][0]$ is non-zero is 2^{-24}. For Step 4f, the probability that $\Delta S_{-1} = \Delta S_8^{\mathsf{AK}}$ is 2^{-8}. Thus, the estimated time complexity of the algorithm above is $2^{24\times2+8} = 2^{56}$.

For Step 1, the number of the pairs of compatible differences $(\Delta S_2^{\mathsf{AK}}, \Delta S_3^{\mathsf{SB}})$ is $255^4 \times 255^4 \times 2^{-16} \approx 2^{48}$. For Step 2, the number of the pairs of compatible differences $(\Delta S_5^{\mathsf{AK}}, \Delta S_6^{\mathsf{SB}})$ is also 2^{48}. For each pair of compatible differences $(\Delta S_2^{\mathsf{AK}}, \Delta S_3^{\mathsf{SB}})$ and $(\Delta S_5^{\mathsf{AK}}, \Delta S_6^{\mathsf{SB}})$, the number of the pairs of admissible inputs S_2^{AK} and S_5^{AK} is 2^{32} as mentioned in Step 4a. Thus, we have in total about $2^{48+48+32} = 2^{128}$ candidates for the outbound phase.

4.2 Algorithm to Connect Two Inbound Phases

An algorithm to connect two inbound phases is described in this section. It gives a pair of sequences of state values between SB in the third round and SB in the sixth round whose differences follow the differential path in Fig. 2. The initial and final state values of the sequences are given to the algorithm as input as well as the values of the differences. The algorithm outputs the round keys (K_3, K_4 and K_5) which connect these values. The algorithm pays specific attention to the bytes of states with black circles in Fig. 2. They are given priority simply because they are bytes with non-zero differences.

Input: S_3^{SB}, S_5^{AK}, and ΔS_3^{SB}, ΔS_4^{SB}, ΔS_5^{AK}.
Output: Round keys K_3, K_4 and K_5.
Procedure:

1. Compute ΔS_3^{AK}, ΔS_4^{AK} and ΔS_5^{SB}:
$$\Delta S_3^{\mathsf{AK}} = \Delta S_3^{\mathsf{MC}} = \mathsf{MC}(\mathsf{SR}(\Delta S_3^{\mathsf{SB}}))$$
$$\Delta S_4^{\mathsf{AK}} = \Delta S_4^{\mathsf{MC}} = \mathsf{MC}(\mathsf{SR}(\Delta S_4^{\mathsf{SB}}))$$
$$\Delta S_5^{\mathsf{SB}} = \mathsf{SR}^{-1}(\mathsf{MC}^{-1}(\Delta S_5^{\mathsf{MC}})) \ ,$$
where $\Delta S_5^{\mathsf{MC}} = \Delta S_5^{\mathsf{AK}}$.
2. Select admissible inputs of the S-boxes with non-zero differences of SB in the fourth round: $S_3^{\mathsf{AK}}[0][3]$, $S_3^{\mathsf{AK}}[1][0]$, $S_3^{\mathsf{AK}}[2][1]$ and $S_3^{\mathsf{AK}}[3][2]$.
3. Compute $K_3[0][3]$, $K_3[1][0]$, $K_3[2][1]$ and $K_3[3][2]$ from the corresponding bytes of $S_3^{\mathsf{MC}} = \mathsf{MC}(\mathsf{SR}(S_3^{\mathsf{SB}}))$ and S_3^{AK}.
4. Select admissible inputs of the S-boxes with non-zero differences of SB in the fifth round: $S_4^{\mathsf{AK}}[3]$. Then, compute $S_5^{\mathsf{SB}}[3]$.
5. $K_4[3] = S_4^{\mathsf{MC}}[3] \oplus S_4^{\mathsf{AK}}[3]$, where $S_4^{\mathsf{MC}}[3]$ can be computed from the corresponding bytes of S_3^{AK}.
6. Compute the round key K_5 satisfying the conditions obtained so far. They can be expressed by 8 linear equations on the bytes of K_5, which form an underdetermined and consistent system of linear equations. They are given in Sect. 4.3.
7. Compute the remaining bytes of K_3 from K_5 and $K_4[3]$.
8. Compute S_4^{MC} and S_4^{AK} from S_3^{SB} with K_3 and from S_5^{AK} with K_5, respectively. Then, compute $K_4[j] = S_4^{\mathsf{MC}}[j] \oplus S_4^{\mathsf{AK}}[j]$ for $0 \le j \le 2$.

4.3 Conditions on the Round Key K_5

The following four conditions are led from the key expansion algorithm:

$$K_5[1][0] = K_3[1][0] \oplus \mathsf{S}(K_4[1][3])$$
$$K_5[2][0] \oplus K_5[2][1] = K_3[2][1]$$
$$K_5[3][1] \oplus K_5[3][2] = K_3[3][2]$$
$$K_5[0][2] \oplus K_5[0][3] = K_3[0][3] \ .$$

Notice that all the bytes of K_3 and K_4 on the right side are already fixed by the algorithm.

The other condition comes from the fixed bytes of $S_5^{\mathsf{SB}}[3]$:

$$\mathsf{SR}(S_5^{\mathsf{SB}}[3]) = \mathsf{MC}^{-1}(S_5^{\mathsf{AK}})[3] \oplus \mathsf{MC}^{-1}(K_5)[3] \ .$$

Notice that S_5^{AK} is given to the algorithm as input. They can be expanded to the following four equations:

$$(\mathtt{0b}, \mathtt{0d}, \mathtt{09}, \mathtt{0e})K_5[0] = S_5^{\mathsf{SR}}[3][0] \oplus (\mathtt{0b}, \mathtt{0d}, \mathtt{09}, \mathtt{0e})S_5^{\mathsf{AK}}[0]$$
$$(\mathtt{0d}, \mathtt{09}, \mathtt{0e}, \mathtt{0b})K_5[1] = S_5^{\mathsf{SR}}[2][1] \oplus (\mathtt{0d}, \mathtt{09}, \mathtt{0e}, \mathtt{0b})S_5^{\mathsf{AK}}[1]$$
$$(\mathtt{09}, \mathtt{0e}, \mathtt{0b}, \mathtt{0d})K_5[2] = S_5^{\mathsf{SR}}[1][2] \oplus (\mathtt{09}, \mathtt{0e}, \mathtt{0b}, \mathtt{0d})S_5^{\mathsf{AK}}[2]$$
$$(\mathtt{0e}, \mathtt{0b}, \mathtt{0d}, \mathtt{09})K_5[3] = S_5^{\mathsf{SR}}[0][3] \oplus (\mathtt{0e}, \mathtt{0b}, \mathtt{0d}, \mathtt{09})S_5^{\mathsf{AK}}[3] \, .$$

4.4 Effectiveness of the Attack

We exhaustively checked the values of constant c against which the proposed attack is effective. The attack is effective if and only if its time complexity is smaller than that of the birthday attack: 2^{128} in the current case. We will call the constant c vulnerable if the attack is effective against it.

Vulnerable constants turned out to have at most four non-zero bytes. Non-zero bytes can have any value from `01` to `ff`. Figures 3, 4, 5 and 6 give all the vulnerable constants with one, two, three and four non-zero bytes, respectively.

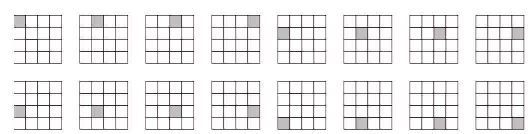


Fig. 3 Vulnerable constants with one non-zero byte. Colored bytes are non-zero.

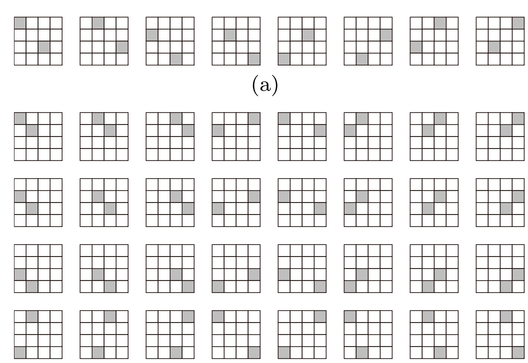


Fig. 4 Vulnerable constants with two non-zero bytes. Colored bytes are non-zero.

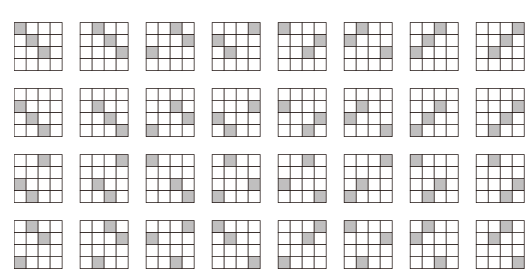


Fig. 5 Vulnerable constants with three non-zero bytes. Colored bytes are non-zero.

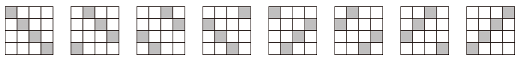


Fig. 6 Vulnerable constants with four non-zero bytes. Colored bytes are non-zero.

The total number of vulnerable constants is

$$16 \times 255 + 40 \times 255^2 + 32 \times 255^3 + 8 \times 255^4 \approx 2^{35} \ .$$

All the constants with one non-zero byte are vulnerable. Non-zero bytes of the vulnerable constants with two non-zero bytes are

- $c[i][j]$ and $c[i+2][j+2]$,
- $c[i][j]$ and $c[i+1][j+1]$, or
- $c[i][j]$ and $c[i+1][j-1]$

for $0 \le i \le 3$ and $0 \le j \le 3$. Non-zero bytes of the vulnerable constants with three non-zero bytes are

- $c[i][j]$, $c[i+1][j+1]$ and $c[i+2][j+2]$, or
- $c[i][j]$, $c[i+1][j-1]$ and $c[i+2][j-2]$

for $0 \le i \le 3$ and $0 \le j \le 3$. Non-zero bytes of the vulnerable constants with four non-zero bytes are

- $c[0][j]$, $c[1][j+1]$, $c[2][j+2]$ and $c[3][j+3]$, or
- $c[0][j]$, $c[1][j-1]$, $c[2][j-2]$ and $c[3][j-3]$

for $0 \le j \le 3$. All the additions above are modulo 4.

The estimated time complexity of the collision attack on the compression function with any constant

- in Figs. 3 and 4(a) is 2^{64}.
- in Figs. 4(b), 5 and 6 is 2^{96}.

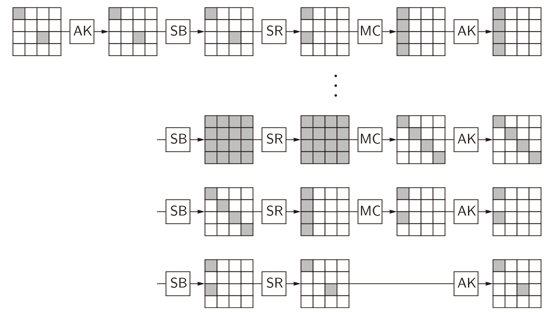
Two examples of differential paths for vulnerable constants with two non-zero bytes are given in Fig. 7. It is not difficult to construct differential paths for the other vulnerable constants.

For the differential path in Fig. 7(b), the probability of the transition from ΔS_1^{MC} to ΔS_1^{SR} is 2^{-16}. The probability of the transition from ΔS_7^{SR} to ΔS_7^{MC} is 2^{-48}. The probability that $\Delta S_{-1} = \Delta S_8^{\mathsf{AK}}$ is 2^{-16}. The probability that $\Delta S_{-1} = c$ is also 2^{-16}. Thus, the estimated time complexity of the attack is $2^{16+48+16+16} = 2^{96}$.

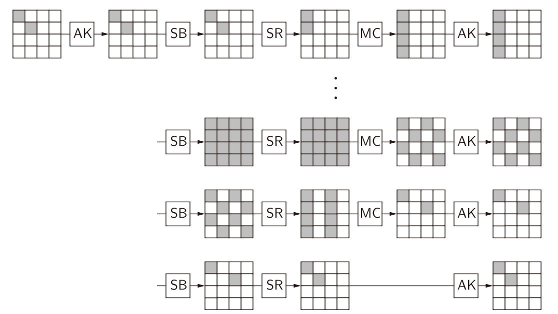
5. Collision Attack on Instantiation with 9-Round AES-256

The collision attack on f_0 instantiated with 8-round AES-256 can be extended to the collision attack on f_0 instantiated with 9-round AES-256 for different choice of the constant c. The differential path used by this attack is presented in Fig. 8. $\Delta h_0 = \Delta S_{-1}$ has four non-zero bytes on its diagonal. Notice that the differential path from the third round to the sixth round is identical to the differential path from the second round to the fifth round in Fig. 2.

The first inbound phase of the attack is in the third and the fourth rounds, and the second one is in the sixth and the seventh rounds. The algorithm shown in Sect. 4.2 can be used to connect these inbound phases and obtain K_4, K_5 and K_6.



(a) The estimated time complexity is $2^{64} (= 2^{16+16+16+16})$.



(b) The estimated time complexity is $2^{96} (= 2^{16+48+16+16})$.

Fig. 7 Examples of differential paths used by the collision attack on the compression function with vulnerable constants with two non-zero bytes. Colored bytes are non-zero differences. The differential paths from the second round to the fifth round are omitted since they are identical to the differential path from the second round to the fifth round in Fig. 2.

In the outbound phase of the attack, S_{-1} and ΔS_{-1} are computed from S_3^{AK} and ΔS_3^{AK}, and S_9^{AK} and ΔS_9^{AK} are computed from S_6^{AK} and ΔS_6^{AK}. For these computations,

- the success probability of the transition from ΔS_2^{MC} to ΔS_2^{SR} is 2^{-24},
- the success probability of the transition from ΔS_8^{SR} to ΔS_8^{MC} is 2^{-32}, and
- the probability that $\Delta S_{-1} = \Delta S_9^{\mathsf{AK}}$ is 2^{-32}.

Thus, the estimated time complexity of the attack is $2^{24+32+32} = 2^{88}$. Though it is beyond the complexity of the birthday attack for f_0, it is effective for our purpose.

Due to the symmetry of AES, the attack also works with the same kind of the differential paths with ΔS_{-1} such that the four non-zero bytes of ΔS_{-1} are

- $\Delta S_{-1}[0][1]$, $\Delta S_{-1}[1][2]$, $\Delta S_{-1}[2][3]$, $\Delta S_{-1}[3][0]$,
- $\Delta S_{-1}[0][2]$, $\Delta S_{-1}[1][3]$, $\Delta S_{-1}[2][0]$, $\Delta S_{-1}[3][1]$, or
- $\Delta S_{-1}[0][3]$, $\Delta S_{-1}[1][0]$, $\Delta S_{-1}[2][1]$, $\Delta S_{-1}[3][2]$.

The total time complexity of the collision attack on F is $2^{88+32} = 2^{120}$ since the probability that $\Delta h_0 = c$ is 2^{-32} for constant c which has four non-zero bytes at the same positions as the non-zero bytes of Δh_0. The total number of vulnerable constants is $4 \times 255^4 \approx 2^{33.98}$.

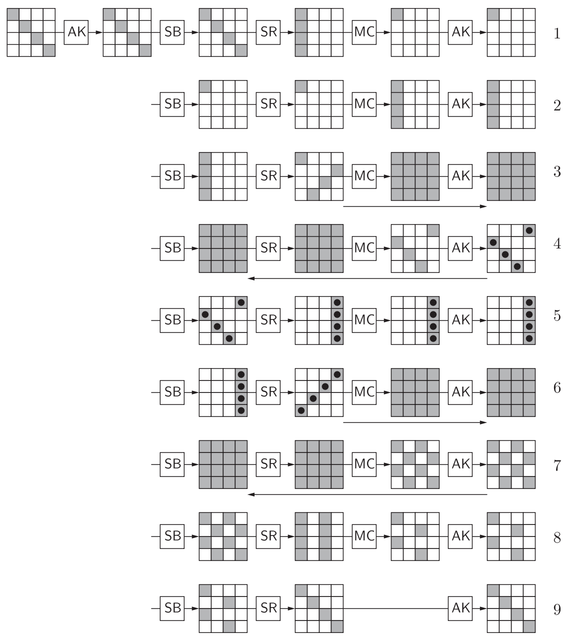


Fig. 8 The differential path used by the collision attack on the compression function f_0 instantiated with 9-round AES-256. Colored bytes are non-zero differences. The differential path from the third round to the sixth round is identical to the differential path from the second round to the fifth round in Fig. 2.

6. Conclusion

This paper has presented a free-start collision attack on the DBL compression function [18] instantiated with round-reduced AES-256. A drawback of the attack is that it is effective against restricted constants. It is interesting if the restriction is reduced. It is also interesting to apply the attack to instantiations with other block ciphers.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments. This work was supported by JSPS KAKENHI Grant Numbers 21240001 and 25330150. It was also supported by Grant-in-Aid for Scientific Research (C) (15K00183) and (15K00189).

References

[1] A. Albertini, J.-P. Aumasson, M. Eichlseder, F. Mendel, and M. Schläffer, "Malicious Hashing: Eve's Variant of SHA-1," Selected Areas in Cryptography — SAC 2014, Lecture Notes in Computer Science, vol.8781, pp.1–19, Springer, 2014.

[2] R. AlTawy and A.M. Youssef, "Watch your constants: Malicious Streebog." Cryptology ePrint Archive, Report 2014/879, 2014. http://eprint.iacr.org/

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

[4] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche, "The Keccak sponge function family," 2008. http://keccak.noekeon.org

[5] A. Biryukov, D. Khovratovich, and I. Nikolić, "Distinguisher and related-key attack on the full AES-256," CRYPTO, ed. S. Halevi, Lecture Notes in Computer Science, vol.5677, pp.231–249, Springer, 2009. An extended version is "Cryptology ePrint Archive: Report 2009/241" at http://eprint.iacr.org/

[6] J. Black, P. Rogaway, T. Shrimpton, and M. Stam, "An analysis of the blockcipher-based hash functions from PGV," J. Cryptol., vol.23, no.4, pp.519–545, 2010.

[7] A. Bogdanov, G. Leander, C. Paar, A. Poschmann, M.J.B. Robshaw, and Y. Seurin, "Hash functions and RFID tags: Mind the gap," CHES, ed. E. Oswald and P. Rohatgi, Lecture Notes in Computer Science, vol.5154, pp.283–299, Springer, 2008.

[8] B.O. Brachtl, D. Coppersmith, M.M. Hyden, S.M. Matyas Jr., C.H.W. Meyer, J. Oseas, S. Pilpel, and M. Schilling, "Data authentication using modification detection codes based on a public one-way encryption function," March 1990. U.S. Patent # 4,908,861.

[9] D. Chang, "A practical limit of security proof in the ideal cipher model: Possibility of using the constant as a trapdoor in several double block length hash functions," Cryptology ePrint Archive, Report 2006/481, 2006. http://eprint.iacr.org/

[10] J. Chen, S. Hirose, H. Kuwakado, and A. Miyaji, "A collision attack on a double-block-length compression function instantiated with round-reduced AES-256," Information Security and Cryptology — ICISC 2014, Lecture Notes in Computer Science, vol.8949, pp.271–285, Springer, 2015.

[11] J. Daemen and V. Rijmen, The Design of Rijndael, Springer, 2002.

[12] A. Duc, J. Guo, T. Peyrin, and L. Wei, "Unaligned rebound attack: Application to Keccak," Fast Software Encryption, Lecture Notes in Computer Science, vol.7549, pp.402–421, Springer, 2012.

[13] N. Ferguson, "Observations on H-PRESENT-128," Crypto 2011 Rump Session, 2011. http://www.iacr.org/cryptodb/archive/2011/CRYPTO/video/rump/

[14] FIPS PUB 180-4, "Secure hash standard (SHS)," March 2012.

[15] FIPS PUB 197, "Advanced encryption standard (AES)," 2001.

[16] E. Fleischmann, M. Gorski, and S. Lucks, "Security of cyclic double block length hash functions," Cryptography and Coding, Lecture Notes in Computer Science, vol.5921, pp.153–175, Springer, 2009.

[17] S. Hirose, "Provably secure double-block-length hash functions in a black-box model," Information Security and Cryptology — ICISC 2004, Lecture Notes in Computer Science, vol.3506, pp.330–342, Springer, 2005.

[18] S. Hirose, "Some plausible constructions of double-block-length hash functions," Fast Software Encryption, Lecture Notes in Computer Science, vol.4047, pp.210–225, Springer, 2006.

[19] J. Jean, M. Naya-Plasencia, and T. Peyrin, "Multiple limited-birthday distinguishers and applications," Selected Areas in Cryptography — SAC 2013, Lecture Notes in Computer Science, vol.8282, pp.533–550, Springer, 2014.

[20] D. Khovratovich, I. Nikolić, and C. Rechberger, "Rotational rebound attacks on reduced skein," Advances in Cryptology — ASIACRYPT 2010, Lecture Notes in Computer Science, vol.6477, pp.1–19, Springer, 2010.

[21] L.R. Knudsen, P. Gauravaram, K. Matusiewicz, F. Mendel, C. Rechberger, M. Schläffer, and S.S. Thomsen, "Grøstl — A SHA-3 candidate," http://www.groestl.info, 2008.

[22] T. Kobayashi and S. Hirose, "Collision attack on double-block-length compression function using round-reduced PRESENT," IEICE Trans. Fundamentals (Japanese Edition), vol.J96-A, no.8, pp.541–550, 2013.

[23] X. Lai and J.L. Massey, "A proposal for a new block encryption standard," Advances in Cryptology — EUROCRYPT'f90, Lecture Notes in Computer Science, vol.473, pp.389–404, Springer, 1991.

[24] X. Lai and J.L. Massey, "Hash function based on block ciphers," EUROCRYPT, ed. R.A. Rueppel, Lecture Notes in Computer Science, vol.658, pp.55–70, Springer, 1992.

[25] M. Lamberger, F. Mendel, M. Schläffer, C. Rechberger, and V.

Rijmen, "The rebound attack and subspace distinguishers: Application to whirlpool," J. Cryptol., vol.28, no.2, pp.257–296, 2015.
[26] J. Lee and D. Kwon, "The security of abreast-DM in the ideal cipher model," IEICE Trans. Fundamentals, vol.E94-A, no.1, pp.104–109, 2011.
[27] J. Lee and M. Stam, "MJH: A faster alternative to MDC-2," Topics in Cryptology — CT-RSA 2011, Lecture Notes in Computer Science, vol.6558, pp.213–236, Springer, 2011.
[28] J. Lee, M. Stam, and J. Steinberger, "The collision security of tandem-DM in the ideal cipher model," Advances in Cryptology — CRYPTO 2011, Lecture Notes in Computer Science, vol.6841, pp.561–577, Springer, 2011.
[29] F. Mendel, T. Peyrin, C. Rechberger, and M. Schläffer, "Improved cryptanalysis of the reduced Grøstl compression function, ECHO permutation and AES block cipher," Selected Areas in Cryptography, Lecture Notes in Computer Science, vol.5867, pp.16–35, Springer, 2009.
[30] F. Mendel, C. Rechberger, M. Schläffer, and S.S. Thomsen, "The rebound attack: Cryptanalysis of reduced whirlpool and Grøstl," Fast Software Encryption, Lecture Notes in Computer Science, vol.5665, pp.260–276, Springer, 2009.
[31] O. Özen and M. Stam, "Another glance at double-length hashing," Cryptography and Coding, Lecture Notes in Computer Science, vol.5921, pp.176–201, Springer, 2009.
[32] T. Peyrin, H. Gilbert, F. Muller, and M. Robshaw, "Combining compression functions and block cipher-based hash functions," Advances in Cryptology — ASIACRYPT 2006, Lecture Notes in Computer Science, vol.4284, pp.315–331, Springer, 2006.
[33] B. Preneel, R. Govaerts, and J. Vandewalle, "Hash functions based on block ciphers: A synthetic approach," CRYPTO, ed. D.R. Stinson, Lecture Notes in Computer Science, vol.773, pp.368–378, Springer, 1993.
[34] V. Rijmen and P.S.L.M. Barreto, "The Whirlpool hash function." http://www.larc.usp.br/~pbarreto/WhirlpoolPage.html, 2000.
[35] V. Rijmen, D. Toz, and K. Varici, "Rebound attack on reduced-round versions of JH," Fast Software Encryption, Lecture Notes in Computer Science, vol.6147, pp.286–303, Springer, 2010.
[36] R. Rivest, "The MD5 message-digest algorithm." Request for Comments 1321 (RFC 1321), The Internet Engineering Task Force, 1992.
[37] Y. Sasaki, "Meet-in-the-middle preimage attacks on AES hashing modes and an application to whirlpool," IEICE Trans. Fundamentals, vol.E96-A, no.1, pp.121–130, 2013.
[38] L. Wei, T. Peyrin, P. Sokołowski, S. Ling, J. Pieprzyk, and H. Wang, "On the (in)security of IDEA in various hashing modes," FSE, ed. A. Canteaut, Lecture Notes in Computer Science, vol.7549, pp.163–179, Springer, 2012. The full version is "Cryptology ePrint Archive: Report 2012/264" at http://eprint.iacr.org/

Jiageng Chen received the B.Sc. in computer science from Huazhong University of Science and Technology (HUST), Wuhan, China in 2004, the M.Sc. and Dr.Sci. degrees in information science from Japan Advanced Institute of Science and Technology (JAIST), Nomi, Japan in 2007 and 2012 respectively. He was an assistant professor at Japan Advanced Institute of Science and Technology (JAIST) from 2012 to 2015. Currently, he is an Associate Professor at the Computer School of Central China Normal University. His research areas mainly include cryptanalysis on symmetric key cryptography, fast implementation and so on.

Shoichi Hirose received the B.E., M.E. and D.E. degrees in information science from Kyoto University, Kyoto, Japan, in 1988, 1990 and 1995, respectively. From 1990 to 1998, he was a research associate at Faculty of Engineering, Kyoto University. From 1998 to 2005, he was a lecturer at Graduate School of Informatics, Kyoto University. From 2005 to 2009, he was an associate professor at Faculty of Engineering, University of Fukui. From 2009, he is a professor at Graduate School of Engineering, University of Fukui. His current interests include cryptography and information security. He received Young Engineer Award from IEICE in 1997, and KDDI Foundation Research Award in 2008.

Hidenori Kuwakado received the B.E., M.E. and D.E. degrees from Kobe University in 1990, 1992, and 1999 respectively. He worked for Nippon Telegraph and Telephone Corporation from 1992 to 1996. From 1996 to 2002, he was a research associate in the Faculty of Engineering, Kobe University. From 2002 to 2007, he was an associate professor in the Faculty of Engineering, Kobe University. From 2007 to 2013, he was an associate professor in Graduate School of Engineering, Kobe University. Since 2013, he has been a professor in Faculty of Informatics, Kansai University. His research interests are in cryptography and information security.

Atsuko Miyaji received the B.Sc., the M.Sc., and the Dr. Sci. degrees in mathematics from Osaka University, Osaka, Japan in 1988, 1990, and 1997 respectively. She joined Panasonic Co., LTD from 1990 to 1998 and engaged in research and development for secure communication. She was an associate professor at the Japan Advanced Institute of Science and Technology (JAIST) in 1998. She has joined the computer science department of the University of California, Davis since 2002. She has been a professor at the Japan Advanced Institute of Science and Technology (JAIST) since 2007 and the director of Library of JAIST from 2008 to 2012. She has been a professor at Graduate School of Engineering, Osaka University since 2015. Her research interests include the application of number theory into cryptography and information security. She received Young Paper Award of SCIS'93 in 1993, Notable Invention Award of the Science and Technology Agency in 1997, the IPSJ Sakai Special Researcher Award in 2002, the Standardization Contribution Award in 2003, Engineering Sciences Society: Certificate of Appreciation in 2005, the AWARD for the contribution to CULTURE of SECURITY in 2007, IPSJ/ITSCJ Project Editor Award in 2007, 2008, 2009, 2010, and 2012, the Director-General of Industrial Science and Technology Policy and Environment Bureau Award in 2007, Editorial Committee of Engineering Sciences Society: Certificate of Appreciation in 2007, DoCoMo Mobile Science Awards in 2008, Advanced Data Mining and Applications (ADMA 2010) Best Paper Award, The chief of air staff: Letter of Appreciation Award, Engineering Sciences Society: Contribution Award in 2012, and Prizes for Science and Technology, The Commendation for Science and Technology by the Minister of Education, Culture, Sports, Science and Technology. She is a member of the International Association for Cryptologic Research, the Institute of Electronics, Information and Communication Engineers, the Information Processing Society of Japan, and the Mathematical Society of Japan.