Cryptanalysis of the Modified Version of the Hash Function Proposed at PKC'98

Daewan Han, Sangwoo Park, and Seongtaek Chee

National Security Research Institute
161 Gajeong-dong, Yuseong-gu, Daejeon, 305-350, Korea
{dwh,psw,chee}@etri.re.kr

Abstract. In the conference PKC'98, Shin *et al.* proposed a dedicated hash function of the MD family. In this paper, we study the security of Shin's hash function. We analyze the property of the Boolean functions, the message expansion, and the data dependent rotations of the hash function. We propose a method for finding the collisions of the modified Shin's hash function and show that we can find collisions with probability 2^{-30}.

1 Introduction

Hash functions are used for many cryptographic applications, such as message authentication and digital signature. A hash function is a computationally efficient function which maps binary strings of arbitrary length to binary strings of some fixed length. Cryptographic hash functions should satisfy the following properties [3]:

– *pre-image resistance*: given a y in the image of a hash function h, it is computationally infeasible to find any pre-image x such that $h(x) = y$.
– *2nd pre-image resistance*: given x and $h(x)$, it is computationally infeasible to find a $x' \neq x$ such that $h(x) = h(x')$.
– *collision resistance*: it is computationally infeasible to find any two distinct inputs x, x' which hash to the same output.

Since the hash function MD4 [6] was introduced by R. Rivest, many dedicated hash functions based on design principles of MD4 have been proposed. MD5 [7], HAVAL [10], RIPEMD [5], RIPEMD-160 [2], and SHA-1 [4] are the dedicated hash functions of the MD family.

In the conference PKC'98, Shin *et al.* proposed a dedicated hash function of the MD family [8]. We call it Shin's hash function. The compression function of Shin's hash function processes a message block of 512 bits and consists of 4 rounds. Each of the rounds consists of 24 steps. Shin's hash function employs the message expansion similar to SHA-1, and Boolean functions similar to HAVAL. Another feature of Shin's hash function is to adopt the data-dependent rotations: rotations are processed by variable amounts determined by message words.

In this paper, we study the security of Shin's hash function. We analyze the property of the Boolean functions, the message expansion, and the data dependent rotations of Shin's hash function. We indicate that, unlike the designer's intention, some of the Boolean functions of Shin's hash function fail to satisfy the Strict Avalanche Criterion(SAC) [9]. Also, we point out that there can be some weakness of the message expansion and the data dependent rotations. We consider the modified Shin's hash function which is Shin's hash function whose Boolean functions all satisfy the SAC, and propose a method for finding the collisions for the modified Shin's hash function.

2 The compression function of Shin's hash function

Throughout this paper, the symbol $+$ represents a modulo 2^{32} addition, $X \oplus Y$, $X \wedge Y$ and $X \vee Y$ represent the bitwise exclusive OR, AND, and OR of X and Y, respectively. The symbol $X^{\lll s}$ denotes the left cyclic shift of X by s bit positions to the left.

The compression function of Shin's hash function processes a 16-word message block of 512 bits, $(X_0, X_1, \ldots, X_{15})$, and consists of 4 rounds. The 16-word message block is expanded to a 24-word message block, $(X_0, X_1, \ldots, X_{23})$. In the 24-word message block, $X_i (i = 0, 1, \ldots, 15)$ are the same as the message words of the original 16-word message blocks and the additional 8 message words, $X_i (i = 16, 17, \cdots, 23)$ are determined by the 16-word message blocks as follows:

$$X_{16+i} = (X_{0+i} \oplus X_{2+i} \oplus X_{7+i} \oplus X_{12+i})^{\lll 1}, i = 0, 1, \cdots, 7. \tag{1}$$

With the expanded 24-word message block, the compression function transforms a 5-word(160 bits) initial value (A, B, C, D, E) into a 160-bit output value. The 5-word initial values are the followings:

$$A = \text{0x}67452301,\ B = \text{0x}efcdab89,\ C = \text{0x}98badcfe,$$

$$D = \text{0x}10325476,\ E = \text{0x}c3d2e1f0.$$

Each round of the compression function consists of 24 steps and each step processes a different word. The orders in which the words are processed differ from round to round. The word processing orders are determined by the following:

Round 1	Round 2	Round 3	Round 4
id	ρ	ρ^2	ρ^3

and the permutation ρ is as follows:

i	0	1	2	3	4	5	6	7	8	9	10	11
$\rho(i)$	4	21	17	1	23	18	12	10	5	16	8	0
i	12	13	14	15	16	17	18	19	20	21	22	23
$\rho(i)$	20	3	22	6	11	19	15	2	7	14	9	13

In addition, each round employs a different constant. The constant $K_i(i = 1, 2, 3, 4)$ is adopted by i-th round.

$$K_1 = 0\text{x}0,\ K_2 = 0\text{x}5a827999,\ K_3 = 0\text{x}6ed9eba1,\ K_4 = 0\text{x}8f1bbcdc.$$

In each round, one of the following Boolean functions is employed.

$$\begin{aligned}
&f_0(x_1, x_2, x_3, x_4, x_5) = (x_1 \wedge x_2) \oplus (x_3 \wedge x_4) \oplus (x_2 \wedge x_3 \wedge x_4) \oplus x_5\\
&f_1(x_1, x_2, x_3, x_4, x_5) = x_2 \oplus ((x_4 \wedge x_5) \vee (x_1 \wedge x_3))\\
&f_2(x_1, x_2, x_3, x_4, x_5) = x_1 \oplus (x_2 \wedge (x_1 \oplus x_4)) \oplus (((x_1 \wedge x_4) \oplus x_3) \vee x_5)
\end{aligned}$$

The Boolean functions perform bitwise operations on words. f_0, f_1, f_2, f_0 are adopted by the 1st, 2nd, 3rd, and 4th round, respectively.

Now, we describe the step function of Shin's hash function. Let $T_{i,j}(j = 0, 1, \cdots, 4)$ be the input of the step function at step i. Then, the step function of Shin's hash function has a transformation of the form

$$\begin{aligned}
&T_{i,0} = (f(T_{i,0}, T_{i,1}, T_{i,2}, T_{i,3}, T_{i,4}) + X_i + K)^{\lll s}, \quad T_{i,1} = T_{i,1}^{\lll 10}\\
&T_{i+1,1} = T_{i,0}, T_{i+1,2} = T_{i,1}, T_{i+1,3} = T_{i,2}, T_{i+1,4} = T_{i,3}, T_{i+1,0} = T_{i,4}.
\end{aligned}$$

The rotation amount s_i at the step i is determined by the following:

$$s_i = X_{ord(i)} \bmod 32,$$

where $ord(i)$ is determined by the following permutations:

Round 1	Round 2	Round 3	Round 4
ρ^3	ρ^2	ρ	id

3 Some properties of Shin's hash function

In this section, we analyze the property of the Boolean functions, the message expansion, and the data dependent rotations of Shin's hash function.

3.1 The property of the Boolean functions

The designers of Shin's hash function claimed that each of the Boolean functions of the hash function is 0-1 balanced, has a high nonlinearity, and satisfies the SAC [8]. Yet, it is easy to find out that some of the Boolean functions of Shin's hash function fail to satisfy the SAC.

We define the Boolean function f as satisfying the SAC if whenever one input bit of f is changed, each output bit is changed with probability 1/2[9]. In case of Shin's hash function, it is easy to know that the Boolean function f_0 and f_1 do not satisfy the SAC. In case of f_0, we can know that, whenever the input bit, x_5, is changed, the output bit is changed with probability 1. Similarly, in case of f_1, whenever the input bit, x_2, is changed, the output bit is changed with probability 1.

$$f_0(x_1, x_2, x_3, x_4, x_5) = (x_1 \wedge x_2) \oplus (x_3 \wedge x_4) \oplus (x_2 \wedge x_3 \wedge x_4) \oplus x_5,$$
$$f_1(x_1, x_2, x_3, x_4, x_5) = x_2 \oplus ((x_4 \wedge x_5) \vee (x_1 \wedge x_3)).$$

Since the designers of Shin's hash function intended that each of the Boolean functions satisfies the SAC, it can be adequate that we consider the Shin's hash function whose Boolean functions all satisfy the SAC. We call it the modified Shin's hash function.

3.2 The property of the message expansion

The additional 8 message words $X_{16}, X_{17}, \cdots, X_{23}$ are determined by the 16-word message block by the equation (1). The equation (1) can be restated as follows:

$$\begin{aligned}
X_{16} &= (X_0 \oplus X_2 \oplus X_7 \oplus X_{12})^{\lll 1} \\
X_{17} &= (X_1 \oplus X_3 \oplus X_8 \oplus X_{13})^{\lll 1} \\
X_{18} &= (X_2 \oplus X_4 \oplus X_9 \oplus X_{14})^{\lll 1} \\
X_{19} &= (X_3 \oplus X_5 \oplus X_{10} \oplus X_{15})^{\lll 1} \\
X_{20} &= (X_4 \oplus X_6 \oplus X_{11} \oplus X_{16})^{\lll 1} \\
X_{21} &= (X_5 \oplus X_7 \oplus X_{12} \oplus X_{17})^{\lll 1} \\
X_{22} &= (X_6 \oplus X_8 \oplus X_{13} \oplus X_{18})^{\lll 1} \\
X_{23} &= (X_7 \oplus X_9 \oplus X_{14} \oplus X_{19})^{\lll 1}
\end{aligned}$$

For two 32-bit words X and $\tilde{X}$, we will define the difference of X and $\tilde{X}$ as follows:

$$\Delta X = X - \tilde{X} \pmod{2^{32}}.$$

To find a collision for Shin's hash function, we should find two distinct message blocks X and $\tilde{X}$ which have the same hash value. In the two distinct message blocks X and $\tilde{X}$, if non-zero difference occurs between X_i and $\tilde{X}_i$ $(0 \leq i \leq 15)$, non-zero difference can occur between some of the additional 8 message words which are generated from X_i and $\tilde{X}_i$. For example, if non-zero difference occurs between X_0 and $\tilde{X}_0$, then non-zero difference can occur between X_{16} and $\tilde{X}_{16}$. Furthermore, non-zero difference between X_{16} and $\tilde{X}_{16}$ can make non-zero difference between X_{20} and $\tilde{X}_{20}$. Thus, the message expansion can increase the difficulty of finding collisions for the hash function.

Table 1 shows the property of the message expansion of Shin's hash function. It shows that X_{10} and X_{15} affect X_{19} and X_{23} simultaneously. It means that, although ΔX_{10} and ΔX_{15} are non-zero, we can have $\Delta X_{19} = \Delta X_{23} = 0$ in the case $X_{10} = X_{15}$ and $\tilde{X}_{10} = \tilde{X}_{15}$. Similarly, although ΔX_{17}, ΔX_{21}, and ΔX_{22} are non-zero, we can have $\Delta X_8 = \Delta X_{13} = 0$ in the case $X_{17} \oplus X_{21} \oplus X_{22} = 0$ and $\tilde{X}_{17} \oplus \tilde{X}_{21} \oplus \tilde{X}_{22} = 0$.

Table 1. The effect of message expansion of the SHF

X_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}
X_{16}	X_{17}	X_{16}	X_{17}	X_{18}	X_{19}	X_{20}	X_{16}	X_{17}	X_{18}	X_{19}	X_{20}	X_{16}	X_{17}	X_{18}	X_{19}
X_{20}	X_{21}	X_{18}	X_{19}	X_{20}	X_{21}	X_{22}	X_{20}	X_{21}	X_{22}	X_{23}		X_{20}	X_{21}	X_{22}	X_{23}
		X_{20}	X_{21}	X_{22}	X_{23}		X_{21}	X_{22}	X_{23}			X_{21}	X_{22}	X_{23}	
		X_{22}	X_{23}				X_{23}								

Now, we know that there are instances in Shin's hash function where non-zero difference between some original message words cannot be diffused to non-zero difference between some additional message words. We can use this property of the message expansion to find the collisions for the modified Shin's hash function.

3.3 The property of the data dependent rotations

In Shin's hash function, the data dependent rotations are adopted by the equation

$$s_i = X_{ord(i)} \bmod 32,$$

at step i. We can know that in the 1st round, $ord(i)$ is determined by the word processing orders of the 4th round, and $ord(i)$ of the 2nd round is determined by the word processing orders of the 3rd round, and so on.

For example, at step 1, s_1 is determined by the word X_{13}(see Appendix A). So, if we have the word X_{13} such that $X_{13} = 0 \bmod 32$, then, $s_1 = 0$. It means that we can make the data dependent rotations ineffective by choosing the appropriate message block, i.e. if we have the message block $(X_i, i = 0, 1, \ldots, 15)$ such that $X_i = 0 \bmod 32$, we can have all s_i equal to 0.

Also, if we have X_i and $\tilde{X}_i$ such that $\Delta X_i \neq 0$ and $X_i = \tilde{X}_i \bmod 32$, then we know that the shift amounts determined by X_i and $\tilde{X}_i$ are the same although X_i and $\tilde{X}_i$ are different.

4 Attack on the modified Shin's hash function

Although some of the Boolean functions of Shin's hash function do not satisfy the SAC, since the designers of Shin's hash function intended that each of the Boolean functions satisfies the SAC, it seems to be adequate that we study the security of the modified Shin's hash function. In this section, we propose a method for finding the collisions for the modified Shin's hash function.

We define some notations. A_i, B_i, C_i, D_i, E_i represent the chaining variables after step i for a message block $X = (X_0, \ldots, X_{23})$, and $\tilde{A}_i, \tilde{B}_i, \tilde{C}_i, \tilde{D}_i, \tilde{E}_i$ represent the chaining variables after step i for a message block $\tilde{X} = (\tilde{X}_0, \ldots, \tilde{X}_{23})$. s_i and $\tilde{s}_i$ represent the shift value used in step i for X and $\tilde{X}$, respectively.

4.1 Attack on the 6 consecutive steps of the modified Shin's hash function

We analyze the 6 consecutive steps of the modified Shin's hash function and show how to find the collisions for 6 consecutive steps. For convenience, we consider the 6 consecutive steps from step 1 to step 6.

To find the collisions for 6 consecutive steps, we should find two distinct message blocks $X = (X_0, X_1, \ldots, X_5)$ and $\tilde{X} = (\tilde{X}_0, \tilde{X}_1, \ldots, \tilde{X}_5)$ which have the same chaining variables after step 6, i.e. $A_6 = \tilde{A}_6$, $B_6 = \tilde{B}_6$, $C_6 = \tilde{C}_6$, $D_6 = \tilde{D}_6$, and, $E_6 = \tilde{E}_6$.

We consider two distinct message blocks X and $\tilde{X}$ such that $\Delta X_0 = 1^{\lll 31}$ and $\Delta X_1 = \Delta X_2 = \Delta X_3 = \Delta X_4 = \Delta X_5 = 0$. Then, we can have a situation that $s_i = \tilde{s}_i (i = 1, 2, \ldots, 6)$. We assume that $A_0 = \tilde{A}_0$, $B_0 = \tilde{B}_0$, $C_0 = \tilde{C}_0$, $D_0 = \tilde{D}_0$, and, $E_0 = \tilde{E}_0$. Note that the Boolean function f satisfies the SAC.

Now, we analyze each step from step 1 to step 6 and find the probability with which $(X, \tilde{X})$ can be a collision of the 6 consecutive steps. Table 2 shows the updated chaining variables at each step of the 6 consecutive steps. The boxed variables represent the updated chaining variables at each step.

Table 2. chaining variables updated in each step

Step	A_0	B_0	C_0	D_0	E_0	Input message
1	$\boxed{A_1}$	B_1	C_1	D_1	E_1	X_0
2	A_2	B_2	C_2	D_2	$\boxed{E_2}$	X_1
3	A_3	B_3	C_3	$\boxed{D_3}$	E_3	X_2
4	A_4	B_4	$\boxed{C_4}$	D_4	E_4	X_3
5	A_5	$\boxed{B_5}$	C_5	D_5	E_5	X_4
6	$\boxed{A_6}$	B_6	C_6	D_6	E_6	X_5

A_1 and $\tilde{A}_1$ are updated at step 1 as follows:

$$A_1 = (f(A_0, B_0, C_0, D_0, E_0) + X_0 + K)^{\lll s_1}, \qquad B_1 = B_0^{\lll 10}$$
$$\tilde{A}_1 = (f(\tilde{A}_0, \tilde{B}_0, \tilde{C}_0, \tilde{D}_0, \tilde{E}_0) + \tilde{X}_0 + K)^{\lll \tilde{s}_1}, \qquad \tilde{B}_1 = \tilde{B}_0^{\lll 10}$$

Since $\Delta X_0 = 1^{\lll 31}$, we can have a situation that $X_0 \oplus \tilde{X}_0 = 1^{\lll 31}$ with probability 1. Also, we know that $\Delta A_0 = \Delta B_0 = \Delta C_0 = \Delta D_0 = \Delta E_0$ and $s_1 = \tilde{s}_1$. So, we have the following:

$$A_1 \oplus \tilde{A}_1 = 1^{\lll s_1 (\text{or } \tilde{s}_1) - 1}.$$

At step 2, E_2 and $\tilde{E}_2$ are updated as follows:

$$E_2 = (f(E_1, A_1, B_1, C_1, D_1) + X_1 + K)^{\lll s_2}, \qquad A_2 = A_1^{\lll 10},$$
$$\tilde{E}_2 = (f(\tilde{E}_1, \tilde{A}_1, \tilde{B}_1, \tilde{C}_1, \tilde{D}_1) + \tilde{X}_1 + K)^{\lll \tilde{s}_2}, \qquad \tilde{A}_2 = \tilde{A}_1^{\lll 10}.$$

Since $\Delta E_1 = \Delta B_1 = \Delta C_1 = \Delta D_1 = 0$, $\Delta X_1 = 0$, $s_2 = \tilde{s}_2$, we can have the following equation:

$$\Delta E_2 = 0 \Longleftrightarrow f(E_1, A_1, B_1, C_1, D_1) = f(E_1, \tilde{A}_1, B_1, C_1, D_1).$$

Note that $A_1 \oplus \tilde{A}_1 = 1^{\lll s_1 (\text{or } \tilde{s}_1) - 1}$. Since f satisfies the SAC, we can have a result such that

$$f(E_1, A_1, B_1, C_1, D_1) = f(E_1, \tilde{A}_1, B_1, C_1, D_1)$$

with probability 1/2, i.e. $\Delta E_2 = 0$ with probability 1/2.

We assume that we have $\Delta E_2 = 0$ at step 2. At the next step, D_3 and $\tilde{D}_3$ are updated as follows:

$$\begin{aligned} D_3 &= (f(D_2, E_2, A_2, B_2, C_2) + X_2 + K)^{\lll s_3}, \quad & E_3 = E_2^{\lll 10}, \\ \tilde{D}_3 &= (f(\tilde{D}_2, \tilde{E}_2, \tilde{A}_2, \tilde{B}_2, \tilde{C}_2) + \tilde{X}_2 + K)^{\lll s_3}, \quad & \tilde{E}_3 = \tilde{E}_2^{\lll 10}. \end{aligned}$$

Since $\Delta D_2 = \Delta E_2 = \Delta B_2 = \Delta C_2$, $\Delta X_2 = 0$, and $s_3 = \tilde{s}_3$, we can have the following equation:

$$\Delta D_3 = 0 \Longleftrightarrow f(D_2, E_2, A_2, B_2, C_2) = f(D_2, E_2, \tilde{A}_2, B_2, C_2).$$

Since f satisfies the SAC, we can have a result such that

$$f(D_2, E_2, A_2, B_2, C_2) = f(D_2, E_2, \tilde{A}_2, B_2, C_2)$$

with probability 1/2, i.e. $\Delta D_3 = 0$ with probability 1/2.

Similarly, we can have that $\Delta C_4 = 0$ with probability 1/2 at step 4. Also, we can have that $\Delta B_5 = 0$ with probability 1/2 at step 5, and $\Delta A_6 = 0$ with probability 1/2 at step 6.

As a result, for two distinct message blocks $X = (X_0, X_1, \ldots, X_5)$ and $\tilde{X} = (\tilde{X}_0, \tilde{X}_1, \ldots, \tilde{X}_5)$ such that $\Delta X_0 = 1^{\lll 31}$ and $\Delta X_1 = \Delta X_2 = \Delta X_3 = \Delta X_4 = \Delta X_5 = 0$, we can have that $\Delta A_6 = 0$, $\Delta B_6 = 0$, $\Delta C_6 = 0$, $\Delta D_6 = 0$, $\Delta E_6 = 0$ with probability 2^{-5}. So, we can find a collision for the 6 consecutive steps of the modified Shin's hash function with about 2^5 operations.

4.2 Attack on the full steps of the modified Shin's hash function

Now, we propose a method for finding a collision for the full steps of the modified Shin's hash function. We consider two distinct message blocks $X = (X_0, X_1, \ldots, X_{15})$ and $\tilde{X} = (\tilde{X}_0, \tilde{X}_1, \ldots, \tilde{X}_{15})$ which satisfy the following conditions.

- Condition 1 : X_i is arbitrary for $i \neq 8$, 9, and 10.
- Condition 2 : $X_8 =$ 0x00000016 and $X_9 = X_2 \oplus X_4 \oplus X_{14} \oplus$ 0x0000000b
- Condition 3 : $X_{10} = X_{15}$
- Condition 4 : $\tilde{X}_i = X_i$ for $i \neq 10$, 15
- Condition 5 : $\tilde{X}_{10} = \tilde{X}_{15} = X_{15} + 1^{\lll 31}$

Condition 2 implies that $X_8 = 22 \bmod 32$ and $X_{18} = 22 \bmod 32$. So, the shift amounts determined by X_8 and the shift amounts determined by X_{18} are equal to 22. From Condition 3, 4, and 5, we can have that $X_i = \tilde{X}_i (i = 16, 17, \cdots, 23)$ from the property of the message expansion. Also, since $X_i = \tilde{X}_i \bmod 32 (0 \le i \le 23)$, it is easy to know that $s_i = \tilde{s}_i$ $(1 \le i \le 96)$. Finally, we notice that $\Delta X_{10} = \Delta X_{15} = 1^{\lll 31}$.

We denote I_i as the section of the 6 consecutive steps from step i to step $i+5$ and consider the sections I_{11}, I_{32}, I_{43}, I_{54}, I_{69}, and, I_{81}. Note that the first input message word of the sections is X_{10} or X_{15}, and $\Delta X_{10} = \Delta X_{15} = 1^{\lll 31}$(see Appendix A).

First, we consider the section I_{32}. We have two distinct message blocks (X_{10}, X_5, X_{16}, X_8, X_0, X_{20}) and ($\tilde{X}_{10}$, $\tilde{X}_5$, $\tilde{X}_{16}$, $\tilde{X}_8$, $\tilde{X}_0$, $\tilde{X}_{20}$) such that $\Delta X_{10} = 1^{\lll 31}$ and $\Delta X_5 = \Delta X_{16} = \Delta X_8 = \Delta X_0 = \Delta X_{20} = 0$. So, to the section I_{32}, we can apply the attack on the 6 consecutive steps of the modified Shin's hash function, i.e. if $\Delta A_{31} = \Delta B_{31} = \Delta C_{31} = \Delta D_{31} = \Delta E_{31} = 0$, we can have $\Delta A_{37} = \Delta B_{37} = \Delta C_{37} = \Delta D_{37} = \Delta E_{37} = 0$ with probability 2^{-5}. Similarly, the attack on the 6 consecutive steps can be applied to I_{43}, I_{54}, and I_{69} with the same probability.

However, to the section I_{11}, the attack on the 6 consecutive steps cannot be directly applied because $\Delta X_{15} \neq 0$, where X_{15} and $\tilde{X}_{15}$ are the input message words of the last step of I_{11}. However, by using the property of the data dependent rotations, this problem can be solved. Note that, at step 16, A_{16} and $\tilde{A}_{16}$ are updated as follows:

$$\begin{array}{ll} A_{16} = (f_0(A_{15}, B_{15}, C_{15}, D_{15}, E_{15}) + X_{15} + K_1)^{\lll s_{16}}, & B_{16} = B_{15}^{\lll 10} \\ \tilde{A}_{16} = (f_0(\tilde{A}_{15}, \tilde{B}_{15}, \tilde{C}_{15}, \tilde{D}_{15}, \tilde{E}_{15}) + \tilde{X}_{15} + K_1)^{\lll s_{16}}, & \tilde{B}_{15} = \tilde{B}_{15}^{\lll 10} \end{array}$$

We can have that $\Delta B_{15} = \Delta C_{15} = \Delta D_{15} = \Delta E_{15} = 0$ with probability 2^{-4}. Note that at step 11, A_{11} and $\tilde{A}_{11}$ are updated by the message words X_{10} and $\tilde{X}_{10}$ such that $\Delta X_{10} = 1^{\lll 31}$, and s_{11} and $\tilde{s}_{11}$ are determined by X_{18} and $\tilde{X}_{18}$, respectively. Since we have $X_{18} = \tilde{X}_{18}$ and $X_{18} = 22 \bmod 32$ from Condition 2, s_{11} and $\tilde{s}_{11}$ are equal to 22. Thus, we know that $A_{11} \oplus \tilde{A}_{11} = 1^{\lll 21}$. At step 12, A_{11} and $\tilde{A}_{11}$ are left-rotated by 10 bit positions, i.e. $A_{12} \oplus \tilde{A}_{12} = 1^{\lll 31}$. Since, from step 13 to step 15, $A_{13} = A_{14} = A_{15}$ and $\tilde{A}_{13} = \tilde{A}_{14} = \tilde{A}_{15}$, the equation $A_{15} \oplus \tilde{A}_{15} = 1^{\lll 31}$ holds. Now, we know that $\Delta X_{15} = \Delta A_{15} = 1^{\lll 31}$, so we can have the following equation:

$$\begin{array}{l} \Delta A_{16} = 0 \\ \Longleftrightarrow f_0(A_{15}, B_{15}, C_{15}, D_{15}, E_{15}) \oplus f_0(\tilde{A}_{15}, B_{15}, C_{15}, D_{15}, E_{15}) = 1^{\lll 31} \end{array}$$

Since f_0 satisfies the SAC, we can have that

$$f_0(A_{15}, B_{15}, C_{15}, D_{15}, E_{15}) \oplus f_0(\tilde{A}_{15}, B_{15}, C_{15}, D_{15}, E_{15}) = 1^{\lll 31}$$

with probability 1/2. So, $\Delta A_{16} = 0$ with probability 2^{-5}. Similarly, by using the property of the data dependent rotations, this attack can be applied to the section I_{81}. In this case, by the value of X_8 in Condition 2, we have that $s_{81} = \tilde{s}_{81} = 22$.

As our main result, if we have the two distinct message blocks X and $\tilde{X}$ which satisfy the Condition 1,2,3,4, and 5, we can find a collision of the modified Shin's hash function with probability $(2^{-5})^6 = 2^{-30}$.

Now, we find a collision of the modified Shin's hash function by computer simulation. We employ the Boolean functions $f_i (i = 0, 1, 2, 3)$ which satisfy the SAC as follows:

$$\begin{aligned}
f_0(x_1, x_2, x_3, x_4, x_5) &= x_2 \oplus (x_3 \wedge (x_2 \oplus x_5)) \oplus (((x_2 \wedge x_5) \oplus x_4) \vee x_1) \\
f_1(x_1, x_2, x_3, x_4, x_5) &= x_3 \oplus (x_4 \wedge (x_3 \oplus x_1)) \oplus (((x_3 \wedge x_1) \oplus x_5) \vee x_2) \\
f_2(x_1, x_2, x_3, x_4, x_5) &= x_1 \oplus (x_2 \wedge (x_1 \oplus x_4)) \oplus (((x_1 \wedge x_4) \oplus x_3) \vee x_5) \\
f_3(x_1, x_2, x_3, x_4, x_5) &= x_4 \oplus (x_5 \wedge (x_4 \oplus x_2)) \oplus (((x_4 \wedge x_2) \oplus x_1) \vee x_3)
\end{aligned}$$

$f_i (i = 0, 1, 2, 3)$ is the modified version of the Boolean function f_2 of Shin's hash function, which satisfies the SAC. Note that our attack does not use the specific properties of the Boolean functions except the SAC.

As a result of computer simulation, we give a collision for the modified Shin's hash function in Table 3 which has the following hash value:

0xdfe4e58f, 0x1f21fb34, 0x9956457f, 0x8726dff2, 0x0a45bef3

Table 3. A collision for the modified Shin's hash function

X_0 = 0xe64ec066	$\tilde{X}_0$ = 0xe64ec066
X_1 = 0xfd126b95	$\tilde{X}_1$ = 0xfd126b95
X_2 = 0x6d80c03e	$\tilde{X}_2$ = 0x6d80c03e
X_3 = 0x09d32e0c	$\tilde{X}_3$ = 0x09d32e0c
X_4 = 0x767d3ff5	$\tilde{X}_4$ = 0x767d3ff5
X_5 = 0x2bc1b633	$\tilde{X}_5$ = 0x2bc1b633
X_6 = 0x40727b94	$\tilde{X}_6$ = 0x40727b94
X_7 = 0xd7e17540	$\tilde{X}_7$ = 0xd7e17540
X_8 = 0x00000016	$\tilde{X}_8$ = 0x00000016
X_9 = 0x278364e1	$\tilde{X}_9$ = 0x278364e1
X_{10} = 0xe7e7d228	$\tilde{X}_{10}$ = 0x67e7d228
X_{11} = 0x8014bf7d	$\tilde{X}_{11}$ = 0x8014bf7d
X_{12} = 0xd5a3b0de	$\tilde{X}_{12}$ = 0xd5a3b0de
X_{13} = 0x5a70ffd6	$\tilde{X}_{13}$ = 0x5a70ffd6
X_{14} = 0x3c7e9b21	$\tilde{X}_{14}$ = 0x3c7e9b21
X_{15} = 0xe7e7d228	$\tilde{X}_{15}$ = 0x67e7d228

5 Conclusion

In this paper, we have studied the security of Shin's hash function proposed by Shin *et al.* in the conference PKC'98. We have pointed out that, unlike the designer's intention, some of the Boolean functions of Shin's hash function do not satisfy the SAC. Also, we have indicated that there are instances in Shin's hash function that the message expansion is not effective. We have proposed a method for finding the collisions with probability 2^{-30} for the modified Shin's hash function. Furthermore, we have found a collision of the modified Shin's hash function by computer simulation.

Recently, the collisions of the original Shin's hash function have been found by Chang *et al.*. They have extended our attack, and the complexity of the attack is 2^{37} hashing operations [1].

This paper has provided a good example that, although it is known that the SAC is one of the important properties of the cryptographic Boolean functions, it can be absolutely irrelevant for the dedicated hash functions. So, we recommend that the Boolean functions of the dedicated hash function of the MD family be carefully chosen. Also, the message expansion should be carefully designed, and we conjecture that the data dependent rotations seem to be inadequate for the dedicated hash functions.

References

1. Donghoon Chang, Jaechul Sung, Soo Hak Sung, Sangjin Lee, and Jongin Lim. Full-Round Differential Attack on the Hash Function Proposed at PKC'98. Proceedings of Koreacrypt'01, pages 24–35, 2002.
2. Hans Dobbertin, Antoon Bosselaers, and Bart Preneel. RIPEMD-160: A strengthened version of RIPEMD. *ftp.esat.kuleuven.ac.be/pub/COSIC/bossselae/ripemd*, April 1996.
3. Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1996.
4. National Institute of Standards and Technology. FIPS PUB 180-1 : Secure Hash Standard, April 1995.
5. Research and Development in Advanced Communications Technologies in Europe. RIPE: Integrity primitives for secure information systems. Final Report of RACE Integrity Primitives Evaluation(R1040),RACE, 1995.
6. Ronald L. Rivest. The MD4 message digest algorithm. In Alfred J. Menezes and Scott A. Vanstone, editors, *Advances in Cryptology - Crypto'90*, volume 537 of *Lecture Notes in Computer Science*, pages 303–311. Springer-Verlag, 1991.
7. Ronald L. Rivest. The MD5 message digest algorithm. In *Request for Comments(RFC) 1321*, April. Internet Activities Board, Internet Privacy Task Force, 1992.
8. Sang Uk Shin, Kyung Hyune Rhee, Dae Hyun Ryu, and Sang Jin Lee. A new hash function based on MDx-family and its application to MAC. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography - PKC'98*, volume 1431 of *Lecture Notes in Computer Science*, pages 234–246. Springer, 1998.
9. A. F. Webster and Stafford E. Tavares. On the design of S-boxes. In Hugh C. Williams, editor, *Advances in Cryptology - Crypto'85*, volume 218 of *Lecture Notes in Computer Science*, pages 523–534. Springer-Verlag, New York, 1986.
10. Yuliang Zheng, Josef Pieprzyk, and Jennifer Seberry. HAVAL-A One-Way Hashing Algorithm with Variable Length of Output. In Jennifer Seberry and Yuliang Zheng, editors, *Advances in Cryptology - Auscrypt'92*, volume 718 of *Lecture Notes in Computer Science*, pages 83–104. Springer, 1992.

A Message processing orders of Shin's hash function

Step	Word	Step	Word	Step	Word	Step	Word
1	X_0	25	X_4	49	X_{23}	73	X_{13}
2	X_1	26	X_{21}	50	X_{14}	74	X_{22}
3	X_2	27	X_{17}	51	X_{19}	75	X_2
4	X_3	28	X_1	52	X_{21}	76	X_{14}
5	X_4	29	X_{23}	53	X_{13}	77	X_3
6	X_5	30	X_{18}	54	$\boxed{X_{15}}$	78	X_6
7	X_6	31	X_{12}	55	X_{20}	79	X_7
8	X_7	32	$\boxed{X_{10}}$	56	X_8	80	X_5
9	X_8	33	X_5	57	X_{18}	81	$\boxed{X_{15}}$
10	X_9	34	X_{16}	58	X_{11}	82	X_0
11	$\boxed{X_{10}}$	35	X_8	59	X_5	83	X_{18}
12	X_{11}	36	X_0	60	X_4	84	X_{23}
13	X_{12}	37	X_{20}	61	X_7	85	$\boxed{X_{10}}$
14	X_{13}	38	X_3	62	X_1	86	X_{21}
15	X_{14}	39	X_{22}	63	X_9	87	X_{16}
16	$\boxed{X_{15}}$	40	X_6	64	X_{12}	88	X_{20}
17	X_{16}	41	X_{11}	65	X_0	89	X_4
18	X_{17}	42	X_{19}	66	X_2	90	X_{17}
19	X_{18}	43	$\boxed{X_{15}}$	67	X_6	91	X_{12}
20	X_{19}	44	X_2	68	X_{17}	92	X_{19}
21	X_{20}	45	X_7	69	$\boxed{X_{10}}$	93	X_8
22	X_{21}	46	X_{14}	70	X_{22}	94	X_9
23	X_{22}	47	X_9	71	X_{16}	95	X_{14}
24	X_{23}	48	X_{13}	72	X_3	96	X_1