Cryptanalysis of the Hash Functions MD4 and RIPEMD

Xiaoyun Wang[1], Xuejia Lai[2], Dengguo Feng[3], Hui Chen[1], Xiuyuan Yu[5]

[1] Shandong University, Jinan250100, China,
xywang@sdu.edu.cn
[2] Shanghai Jiaotong University, Shanghai200052, China
[3] Chinese Academy of Science China, Beijing100080, China
[4] Huangzhou Teacher College, Hangzhou310012, China

Abstract. MD4 is a hash function developed by Rivest in 1990. It serves as the basis for most of the dedicated hash functions such as MD5, SHAx, RIPEMD, and HAVAL. In 1996, Dobbertin showed how to find collisions of MD4 with complexity equivalent to 2^{20} MD4 hash computations. In this paper, we present a new attack on MD4 which can find a collision with probability 2^{-2} to 2^{-6}, and the complexity of finding a collision doesn't exceed 2^{8} MD4 hash operations. Built upon the collision search attack, we present a chosen-message pre-image attack on MD4 with complexity below 2^{8}. Furthermore, we show that for a weak message, we can find another message that produces the same hash value. The complexity is only a single MD4 computation, and a random message is a weak message with probability 2^{-122}.

The attack on MD4 can be directly applied to RIPEMD which has two parallel copies of MD4, and the complexity of finding a collision is about 2^{18} RIPEMD hash operations.

1 Introduction

MD4 [14] is an early-appeared hash function that is designed using basic arithmetic and Boolean operations that are readily available on modern computers. Such type of hash functions are often referred to as dedicated hash functions, and they are quite different from hash functions based on block ciphers. After the publication of MD4, several dedicated hash functions are successively designed, including MD5 [15], HAVAL [18], RIPEMD [13], RIPEMD-160 [9], SHA-1 [10], SHA-256 [11], etc. These hash functions, although more complex, all follow the same design philosophy as MD4 and have similar structures as MD4. In particular, RIPEMD consists of two parallel copies of MD4, and each copy is identical to MD4 except for some internal constants.

There have been several important cryptanalytical results for both MD4 and RIPEMD. In 1996, H. Dobbertin [5] gave a collision attack on MD4 which finds a collision with probability 2^{-22}. He also showed how to find collisions of meaningful messages. In 1998, H. Dobbertin [8] showed that the first two (out of the total three) rounds of MD4 is not one-way, and this means that there is an efficient attack for finding a preimage and a second preimage. For RIPEMD, H. Dobbertin [7] gave an attack that finds a collision of RIPEMD reduced to two rounds with 2^{31} hash operations.

Along with the development of the MD4-family of hash functions, there have also been security analysis on these functions. For example, B. den Boer and A. Bosselaers [3] found pseudo-collisions (same message with two different initial values) for MD5. In Eurocrypto'96, H. Dobbertin [6] presented a collisions of MD5, under another initial value. In Crypto'98, F. Chabaud and A. Joux [4] presented a differential attack on SHA-0 with probability 2^{-61}. At Asiacrypt 2003, B.V. Rompay etc. [16] gave a collision attack on HAVAL-128 with probability 2^{-29}.

Some very interesting results on hash functions came out simultaneously in Crypto 2004. Eli Biham and Rafi Chen [2] presented a near-collision attack on SHA-0, and described their improved results on SHA-0 and SHA-1 in the rump session. Then, A. Joux [12] presented a real collision of SHA-0 with four message blocks. X.Y. Wang etc. [17] also announced real collisions of a series of hash functions including MD4, MD5, HAVAL-128, and RIPEMD in the rump session. All these research work were done independently.

The purpose of this paper is to analyze the security of MD4 and RIPEMD and present more efficient attacks. The main results are summarized below.

1. Collision search attack on MD4: we can find collisions with probability 2^{-2} to 2^{-6} and with complexity less than 2^{8} MD4 hash operations.
2. A theoretical second pre-image attack on MD4 for weak messages: For a weak message, we can find another message that produces the same hash value. The complexity is only a single MD4 computation and a random selected message is a weak message with probability 2^{-122}.
3. Collision search attack on RIPEMD: we can find collisions with probability 2^{-16} and with complexity less than 2^{18} RIPEMD hash operations.

In addition to presenting the new attacks on MD4 and RIPEMD, we also introduce a set of new analytical techniques that are applicable to all the hash functions in the MD4-family. More specifically, we show how to derive a set of the sufficient conditions on the chaining values to ensure the differential path to hold, and how to use message modification techniques to greatly improve the success probability of the attack. Such techniques have proved to be very effective in cryptanalyzing other dedicated hash functions such as MD5, RIPEMD, HAVAL-128, HAVAL-160, SHA0, and especially SHA-1.

All the existing attacks on dedicated hash functions belong to differential attacks [1], since a collision can be regarded as a special differential which has non-zero input difference and zero output difference. We remark that unlike other existing attacks on hash functions, our attack presented in this paper is a "precise" differential attack in which the differential path is more restrictive since it depends on both the difference as well as the specific value of the bit involved.

The paper is organized as follows. In Section 2 we provide a description of MD4 and RIPEMD. In Section 3, we summarize some useful properties of the Boolean functions in two hash functions and introduce the notation used in the paper. As our main result, the collision attack on MD4 is presented in Section 4, the collision attack on RIPEMD is presented in Section 5. In Section 6, we describe a theoretical second pre-image attack on MD4. In Section 7, we summarize our work together with some remarks, especially on the implication for the analysis of the hash function SHA-0.

2 Description of MD4 and RIPEMD

2.1 MD4 Algorithm

The message digest algorithm MD4 compresses any arbitrary bit-length message into a 128-bit hash value. Given any message, the algorithm first pads it into a message with a length that is a multiple of 512 bits We omit the padding method here since it is irrelevant to our attack.

For each 512-bit message block, MD4 compresses it into a 128-bit hash value using a compression function. The MD4 compression function has three rounds. Each round uses a different nonlinear Boolean function defined as follows:

$$\begin{aligned} F(X,Y,Z) &= (X \wedge Y) \vee (\neg X \wedge Z) \\ G(X,Y,Z) &= (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z) \\ H(X,Y,Z) &= X \oplus Y \oplus Z \end{aligned}$$

Here X, Y, Z are 32-bit words. The operations of three functions are all bitwise. $\neg X$ is the bitwise complement of X, $\wedge$, $\oplus$ and $\vee$ are respectively the bitwise AND, XOR and OR.

Each round of the compression function repeats 16 similar step operations, and in each step one the four chaining variables a, b, c, d is updated.

$$\begin{aligned} \phi_0(a,b,c,d,m_k,s) &= ((a + F(b,c,d) + m_k) \bmod 2^{32}) \lll s \\ \phi_1(a,b,c,d,m_k,s) &= ((a + G(b,c,d) + m_k + \texttt{0x5a827999}) \bmod 2^{32}) \lll s \\ \phi_2(a,b,c,d,m_k,s) &= ((a + H(b,c,d) + m_k + \texttt{0x6ed9eba1}) \bmod 2^{32}) \lll s \end{aligned}$$

The initial value for MD4 is defined as:

$$(a,b,c,d) = (\texttt{0x67452301},\ \texttt{0xefcdab89},\ \texttt{0x98badcfe},\ \texttt{0x10325476})$$

MD4 Compression Function. For one 512-bit block M of the padded message $\overline{M}$, $M = (m_0, m_1, ..., m_{15})$, the compression function is defined as follows:

1. Let (aa, bb, cc, dd) be input chaining variables for M. If M is the first message block to be hashed, then (aa, bb, cc, dd) are set to be the initial value. Otherwise they are the output from compressing the previous message block.
2. Perform the following 48 steps (three rounds):

For $j = 0, 1, 2$ and $i = 0, 1, 2, 3$

$$\begin{aligned}
a &= \phi_j(a, b, c, d, w_{j,4i}, s_{j,4i}) \\
d &= \phi_j(d, a, b, c, w_{j,4i+1}, s_{j,4i+1}) \\
c &= \phi_j(c, d, a, b, w_{j,4i+2}, s_{j,4i+2}) \\
b &= \phi_j(b, c, d, a, w_{j,4i+3}, s_{j,4i+3})
\end{aligned}$$

Here $s_{j,4i+k}$ $(k = 0,\ 1,\ 2,\ 3)$ are step-dependent constants, $w_{j,4i+k}$ is a message word and $\lll s_{j,4i+k}$ is circularly left-shift by $s_{j,4i+k}$ bit positions. The specific message order and shift positions are given in Table 5.

3. Add the chaining variables a, b, c and d respectively to the input chaining variables to produce the final chaining variables for the current message block.

$$\begin{aligned}
aa &= (a + aa) \bmod 2^{32} \\
bb &= (b + bb) \bmod 2^{32} \\
cc &= (c + cc) \bmod 2^{32} \\
dd &= (d + dd) \bmod 2^{32}
\end{aligned}$$

If M is the last message block, $H(\overline{M}) = aa|bb|cc|dd$ is the hash value for the message $\overline{M}$. Otherwise repeat the above process with the next 512-bit message block and (aa, bb, cc, dd) as the input chaining variables.

2.2 RIPEMD Algorithm

RIPEMD employs the same nonlinear round functions as MD4 and they are used in the following six operations:

$$\begin{aligned}
\varphi_0(a, b, c, d, m_k, s) &= ((a + F(b, c, d) + m_k) \bmod 2^{32}) \lll s \\
\varphi_1(a, b, c, d, m_k, s) &= ((a + G(b, c, d) + m_k + \texttt{0x5a827999}) \bmod 2^{32}) \lll s \\
\varphi_2(a, b, c, d, m_k, s) &= ((a + H(b, c, d) + m_k + \texttt{0x6ed9eba1}) \bmod 2^{32}) \lll s \\
\psi_0(a, b, c, d, m_k, s) &= ((a + F(b, c, d) + m_k + \texttt{0x50a28be6}) \bmod 2^{32}) \lll s \\
\psi_1(a, b, c, d, m_k, s) &= ((a + G(b, c, d) + m_k) \bmod 2^{32}) \lll s \\
\psi_2(a, b, c, d, m_k, s) &= ((a + H(b, c, d) + m_k + \texttt{0x5c4dd124}) \bmod 2^{32}) \lll s
\end{aligned}$$

In order to easily describe the RIPEMD compression function, we denote MD4 compression function with three operations ϕ_0, ϕ_1 and ϕ_2 as MD4$(\phi_0, \phi_1, \phi_2, M)$.

RIPEMD Compression Function. The RIPEMD compression function employs two copies of MD4 compression function: the left copy is MD4$(\varphi_0, \varphi_1, \varphi_2, M)$, and the right copy is MD4$(\psi_0, \psi_1, \psi_2, M)$. Both copies have the same initial value as MD4. The details of the message order and shift positions are given in Table 7.

1. Let (a, b, c, d) be the input chaining variables for M which is the same as MD4.

2. Perform two copies of the MD4 operation

$$(aa, dd, cc, bb,) \leftarrow \mathrm{MD4}(\varphi_0, \varphi_1, \varphi_2, M),$$
$$(aaa, ddd, ccc, bbb) \leftarrow \mathrm{MD4}(\psi_0, \psi_1, \psi_2, M).$$

3. The output (a, b, c, d) for compressing M is the following:

$$\begin{aligned} a &= (b + cc + ddd) \bmod 2^{32} \\ b &= (c + dd + aaa) \bmod 2^{32} \\ c &= (d + aa + bbb) \bmod 2^{32} \\ d &= (a + bb + ccc) \bmod 2^{32} \end{aligned}$$

3 Preliminaries

3.1 Basic Properties of the Boolean functions

Some properties of three nonlinear Boolean functions are very helpful for determining sufficient conditions for the differential paths that are used in our collision search attack on MD4 and RIPEMD. In what follows, we summarize some well-known properties of these functions.

Proposition 1. *For the nonlinear function $F(X, Y, Z) = (X \wedge Y) \vee (\neg X \wedge Z)$ in the first round, there are the following properties:*

1. $F(x, y, z) = F(\neg x, y, z)$ *if and only if* $y = z$.
2. $F(x, y, z) = F(x, \neg y, z)$ *if and only if* $x = 0$.
3. $F(x, y, z) = F(x, y, \neg z)$ *if and only if* $x = 1$.

Proposition 2. *For the nonlinear function $G(X, Y, Z) = (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$ in the second round, there are the following properties:*

1. $G(x, y, z) = G(\neg x, y, z)$ *if and only if* $y = z$.
2. $G(x, y, z) = G(x, \neg y, z)$ *if and only if* $x = z$.
3. $G(x, y, z) = G(x, y, \neg z)$ *if and only if* $x = y$.

Proposition 3. *For the nonlinear function $H(X, Y, Z) = X \oplus Y \oplus Z$ in the third round, there are the following properties:*

1. $H(x, y, z) = \neg H(\neg x, y, z) = \neg H(x, \neg y, z) = \neg H(x, \neg y, z)$
2. $H(x, y, z) = H(\neg x, \neg y, z) = H(x, \neg y, \neg z) = H(\neg x, y, \neg z)$

3.2 Notation

Here we introduce the notation used in our analysis. Since our attack is a "precise" differential attack, we need to keep track of both the difference as well as the specific value of the bit involved. Therefore, the notation may seem quite complex at a first glance, but the intuition behind these notation will become more clear as we proceed in describing the attacks.

1. $M = (m_0, m_1, ..., m_{15})$ and $M' = (m'_0, m'_1, ..., m'_{15})$ represent two 512-bit messages.
2. a_i, d_i, c_i, b_i respectively denote the outputs of the $(4i-3)$-th, $(4i-2)$-th, $(4i-1)$-th and $4i$-th steps for compressing M, for $1 \leq i \leq 16$.
3. a'_i, b'_i, c'_i, d'_i respectively denote the outputs of the $(4i-3)$-th, $(4i-2)$-th, $(4i-1)$-th and $4i$-th steps for compressing M'.
4. $\Delta m_i = m'_i - m_i$ denotes the difference between two message words m_i and m'_i.
5. $a_{i,j}$, $b_{i,j}$, $c_{i,j}$, $d_{i,j}$ represent respectively the $j-th$ bit of a_i, b_i, c_i, d_i, where the least significant bit is the 1-st bit, and the most significant bit is 32-th bit.
6. $x_i[j]$, $x_i[-j]$ (x can be a, b, c, d) is the resulting values by only changing the $j-th$ bit of the word x_i. $x_i[j]$ is obtained by changing the j-th bit of x_i from 0 to 1. $x_i[-j]$ is obtained by changing the j-th bit of x_i from 1 to 0.
7. $x_i[\pm j_1, \pm j_2, ..., \pm j_l]$ is the value by change $j_1 - th$, $j_2 - th$, ..., $j_l - th$ bits of x_i. The "+" sign means that the bit is changed from 0 to 1, and the "−" sign means that the bit is changed from 1 to 0.

Note that we use integer modular subtraction difference as the measure of difference, not the exclusive-or difference. In addition, we also need to specify the precise values of *each bit* when considering the carry effect in the differential path. This is better understood using an example. Let us consider step 7 in Table 5. The output difference is

$$\Delta c_2 = c'_2 - c_2 = -2^{18} + 2^{21}.$$

Using our notation, $c'_2 = c_2[-19, 22]$. For the specific differential path, we need to expand the *one-bit* subtraction difference in bit 19 into a *three-bit* difference in bits 19,20,21. That is, we expand $c_2[19]$ as $c_2[19, 20, -21]$. Hence, the output c'_2 is represented as

$$c'_2 = c_2[19, 20, -21, 22],$$

as showed in the last column of Table 5.

4 The Collision Attack on MD4

In this section, we will describe a collision attack on MD4 with a success probability 2^{-2} to 2^{-6}. The complexity is below 2^8 MD4 computations. The attack includes three parts:

1. Find a collision differential in which M and M' produces a collision.
2. Derive a set of sufficient conditions which ensure the collision differential to hold.
3. For any random message M, make some modification to M such that almost all the sufficient conditions hold.

4.1 The Collision Differential for MD4

We select a collision differential for MD4 as follows:

$$\Delta H_0 = 0 \overset{(M,M')}{\longrightarrow} \Delta H = 0$$

such that

$$\triangle M = M' - M = (\triangle m_0, \triangle m_1, ..., \triangle m_{15})$$
$$\triangle m_1 = 2^{31},\ \triangle m_2 = 2^{31} - 2^{28},\ \triangle m_{12} = -2^{16}$$
$$\triangle m_i = 0,\ 0 \leq i \leq 15,\ i \neq 1, 2, 12.$$

All the characteristics in the collision differential can be found in Table 5. The first column denotes the step, the second column is the chaining variable in each step for M, the third is the message word for M in each step, the fourth is shift rotation, the fifth and the sixth are respectively the message word difference and chaining variable difference for M and M', and the seventh is the chaining variable for M'. Especially, the empty items both in fifth and sixth columns denote zero differences, and steps those aren't listed in the table have zero differences both for message words and chaining variables.

It is clear that the collision differential consists of two internal collisions respectively from 2-25 steps and 36-41 steps.

The sufficient conditions (Table 6) that ensure all the characteristics to hold can be easily verified by the properties of the Boolean functions given in Section 3. This further means that if M satisfies all the conditions in Table 6, M and M' consists of a collision.

The following is the derivation for the sufficient conditions in the step 9 of Table 5. The differential characteristic in step 9 is:

$$(b_2[-13, -14, 15], c_2[19, 20, -21, 22], d_2[14], a_2)$$
$$\longrightarrow (a_3[17], b_2[-13, -14, 15], c_2[19, 20, -21, 22], d_2[14])$$

1. According to (1) of Proposition 1, the conditions $c_{2,13} = d_{2,13}$ and $c_{2,15} = d_{2,15}$ ensure that the changes in 13-th and 15-th bits in b_2 result in no change in a_3.
2. According to (2) of Proposition 1, the conditions $b_{2,19} = 0$, $b_{2,20} = 0$, $b_{2,21} = 0$, and $b_{2,22} = 0$ ensure that the changes in 19-th, 20-th, 21-th and 22-th bits in c_2 result in no change in a_3.
3. From the property of function f, the conditions $b_{2,14} = 1$, $d_{2,14} = 0$ and $c_{2,14} = 0$ result in $f(b_{2,14}, c_{2,14}, d_{2,14}) = 0$ and $f(\neg b_{2,14}, c_{2,14}, \neg d_{2,14}) = 1$. So $\triangle a_3 = 2^{16}$.

4. The condition $a_{3,17} = 0$ ensures that $a'_3 = a_3[17]$.

Thus the above 10 conditions consists of a set of sufficient conditions for the differential characteristic in step 9.

4.2 Message Modification

From the conditions listed in Table 6, we know that the (M, M') is a collision with probability 2^{-122}. This is greatly lower than the birthday attack probability 2^{-64}. We can improve the probability to $2^{-6} \sim 2^{-2}$ by two types of message modification techniques, which we term as "single-step modification" and "multi-step modification."

Single-Step Modification. It is easy to modify M such that the conditions in round 1 hold. For example, m_1 can be modified as :

$$d_1 \leftarrow d_1 \oplus (d_{1,7} \lll 6) \oplus ((d_{1,8} \oplus a_{1,8}) \lll 7) \oplus ((d_{1,11} \oplus a_{1,11}) \lll 10)$$

$$m_1 \leftarrow (d_1 \ggg 7) - d_0 - F(a_1, b_0, c_0)$$

After simple-message modification, (M, M') is a collision with probability 2^{-25} by Table 6.

Multi-Step Modification. Although the probability 2^{-25} is high enough for us to find many collisions of MD4, we also introduce a multi-message modification to correct the conditions in second round, and that greatly improves the probability. This modification technique is very important for analyzing other hash functions such as MD5, SHA-0, especially SHA-1.

The principle for multi-message modification is that the modifications for some messages consist of a partial collision in the first round which remains all the conditions in the first round to hold, but only change a bit of the second round. The details are as follows:

1. Modify m_0, m_1, m_2, m_3, m_4 successively by Table 1 to correct 5 conditions of a_5 in Table 6. For example, if $a_{5,19} = \overline{c_{4,19}}$, modify m_0, m_1, m_2, m_3, m_4 by Table 1 ($i = 19$). The changed message words don't change any condition of first round in Table 6, but correct $a_{5,19} = \overline{c_{4,19}}$ to $a_{5,19} = c_{4,19}$.
   It is noted that, the conditions in step 17 should be corrected from low bit to high bit, i.e. the order of the bits needed to be changed is:

$$a_{5,19} \rightarrow a_{5,26} \rightarrow a_{5,27} \rightarrow a_{5,29} \rightarrow a_{5,32}$$

2. Similarly, modify m_4, m_5, m_6, m_7, m_8 successively to correct 4 conditions of d_5.

$$d_{5,19} = a_{5,19},\ d_{5,26} = b_{4,26},\ d_{5,27} = b_{4,27},\ d_{5,29} = b_{4,29}$$

Table 1. Message Modification for Correcting $a_{5,i}$, $i = 19, 26, 27, 29, 32$

			Modify m_i	Chaining values after message modification
1	m_0	3	$m_0 \longleftarrow m_0 \pm 2^{i-4}$	${a_1}^{new} = a_1[\pm i]$, b_0, c_0, d_0
2	m_1	7	$m_1 \longleftarrow (d_1 \ggg 7) - d_0 - f(a_1', b_0, c_0)$	d_1, ${a_1}^{new}$, b_0, c_0
3	m_2	11	$m_2 \longleftarrow (c_1 \ggg 11) - c_0 - f(d_1, a_1', b_0)$	c_1, d_1, ${a_1}^{new}$, b_0
4	m_3	19	$m_3 \longleftarrow (b_1 \ggg 19) - b_0 - f(c_1, d_1, a_1')$	b_1, c_1, d_1, ${a_1}^{new}$
5	m_4	3	$m_4 \longleftarrow (a_2 \ggg 3) - a_1' - f(b_1, c_1, d_1)$	a_2, b_1, c_1, d_1

Table 2. The Modification for Correcting $c_{5,i}$, $i = 26,\ 27,\ 29,\ 32$

				Modify m_i	Chaining values after message modification	The extra conditions in first round
6	d_2	m_5	7	$m_5 \leftarrow m_5 + 2^{i-17}$	$d_2[i-9]$, a_2, b_1, c_1	$d_{2,i-9} = 0$
7	c_2	m_6	11		c_2, $d_2[i-9]$, a_2, b_1	$a_{2,i-9} = b_{1,i-9}$
8	b_2	m_7	19		b_2, c_2, $d_2[i-9]$, a_2	$c_{2,i-9} = 0$
9	a_3	m_8	3	$m_8 \leftarrow m_8 - 2^{i-10}$	a_3, b_2, c_2, $d_2[i-9]$	$b_{2,i-9} = 0$
10	d_3	m_9	11	$m_9 \leftarrow m_9 - 2^{i-10}$	d_3, a_3, b_2, c_2	

3. Utilize more precise modification to correct some other conditions. For example, we can use the internal collision in Table 2 in which there are three message words are changed to correct $c_{5,i}$, $i = 26, 27, 29, 32$. The precise modification should add some extra conditions in the first rounds (see Table 2) in advance. There are many other precise modifications. $c_{5,30}$ can be corrected by other modification. By various modifications, besides two conditions in the third round, almost all the conditions in rounds 1-2 will be corrected. The probability can be among $2^{-6} \sim 2^{-2}$.

The complexity of finding a collision doesn't exceed 2^8 MD4 computations. To select a message M is only to change the last two words from the previous selected message M. So, finding (M, M') only needs about one-time single-message modification for the first 14 words. This time can be neglected. For each selected message M, it is only needs two-time single-message modifications for the last two words and about 20 -time advanced modifications for correcting 20 conditions in the second round, and each multi-message modification only needs about a few step operations, so the total time for both kinds of modifications is about two MD4 computations for each selected message. According to the probability of the collision differential, it is easy to know that the complexity of finding (M, M') does not exceed 2^8 MD4 computations. We give two collisions for MD4 in the Table 3.

Table 3. Two collisions for MD4. H is the hash value with little-endian and no message padding, and H^* is the hash value with big-endian and message padding

M_1	4d7a9c83 56cb927a b9d5a578 57a7a5ee de748a3c dcc366b3 b683a020 3b2a5d9f c69d71b3 f9e99198 d79f805e a63bb2e8 45dd8e31 97e31fe5 2794bf08 b9e8c3e9
M_1'	4d7a9c83 d6cb927a 29d5a578 57a7a5ee de748a3c dcc366b3 b683a020 3b2a5d9f c69d71b3 f9e99198 d79f805e a63bb2e8 45dc8e31 97e31fe5 2794bf08 b9e8c3e9
H	5f5c1a0d 71b36046 1b5435da 9b0d807a
H^*	4d7e6a1d efa93d2d de05b45d 864c429b
M_2	4d7a9c83 56cb927a b9d5a578 57a7a5ee de748a3c dcc366b3 b683a020 3b2a5d9f c69d71b3 f9e99198 d79f805e a63bb2e8 45dd8e31 97e31fe5 f713c240 a7b8cf69
M_2'	4d7a9c83 d6cb927a 29d5a578 57a7a5ee de748a3c dcc366b3 b683a020 3b2a5d9f c69d71b3 f9e99198 d79f805e a63bb2e8 45dc8e31 97e31fe5 f713c240 a7b8cf69
H	e0f76122 c429c56c ebb5e256 b809793
H^*	c6f3b3fe 1f4833e0 697340fb 214fb9ea

5 The Collision Attack on RIPEMD

We select a collision differential for RIPEMD as follows:

$$\Delta H_0 = 0 \overset{(M,M')}{\longrightarrow} \Delta H = 0$$

such that

$$m_3' = m_3 + 2^{20},\ m_{10}' = m_{10} + 2^{18} + 2^{31},\ m_{15}' = m_{15} + 2^{31},$$

$$m_i' = m_i,\ i \neq 3, 10, 15.$$

The reason for the choice of M' is that M and M' can easily collide in round 3 with probability 2^{-4}.

The differential characteristics and sufficient conditions can be referred to Table 7 and Table 8.

The following mainly describes the message modification for RIPEMD. Because RIPEMD has two copies of MD4, the modification is more complicated than that of MD4.

Message Modification for Correcting Conditions in the First Round. Select M, we make the modification for M word by word so that both copies with the modified M satisfy the conditions in the first round.

1. Modify m_{i-1} such that i-th step conditions in the left copy hold. The modification is the same as the single-message modification in Section 4.
2. Correct the conditions in the right copy from low bit to high bit. There are many kinds of modifications. The following gives two kinds of modification techniques.
   For example, we correct $aaa_{i,j} = 0$ to $aaa_{i,j} = 1$ by the following methods.

(a) Correct the condition by bit carry. If $j-1$-bit has no constraint condition in table 8, and $aa_{i,j-1} = \overline{aa_{i,j-1}}$, let

$$m_i \leftarrow m_i \pm 2^{j-2-s_i}.$$

We select the modification which results in bit carry in the right and no carry in the left.

(b) Correct the condition by changing $(j-s_i)-th$ bit of chaining variables in the nonlinear round function ψ.

i. Change $(j-s_i)-th$ bit of some chaining variables in the nonlinear round function F by modifying a previous message word, such that the changed bit doesn't occur in Table 8, and the changed bit only causes one of $aa_{i,j}$ and $aaa_{i,j}$ changes.

ii. If $aa_{i,j} = aaa_{i,j} = 0$, modify the next bit of aaa_i.

iii. If $aa_{i,j} = aaa_{i,j} = 1$, let

$$m_i \leftarrow m_i - 2^{j-1-s_i},$$

then modify the next bit of aaa_i.

By combining the above two methods, we can get some other methods to correct $aaa_{i,j}$. For example, if $j-1$-bit has no constraint condition in table 8, and $aa_{i,j-1} = aa_{i,j-1}$, the bit-carry correction of (a) isn't available. We can use (b) or the lower bit carry to change $aa_{i,j-1}$ or $aaa_{i,j-1}$ such that $aa_{i,j-1} = \overline{aaa_{i,j-1}}$, and then use the bit carry.

Remarks. For RIPEMD, a non-zero differential in the first round is an impossible differential with a very high probability. The reason that results in the phenomenon is that, the conditions of both copies in some step cannot hold simultaneously. Among 30 collision differentials we selected, only one can produce the real collisions.

Message Modification for Correcting Some Second Round Conditions. By the multi-message modification in Section 4 to correct the conditions of left copy in the second round. There are about 16 conditions are left, so the modified M and M' is a collision with probability 2^{-16}, and the complexity is about 2^{18} RIPEMD computations. Two collisions for RIPEMD can be seen in Table 4.

6 Theoretical Pre-image Attack on MD4

For a secure hash function, there are two important security properties, one property is collision-resistance, another is one-wayness which is to find a second pre-image or a pre-image. In this section, we will show that we can give a second pre-image attack on MD4 for a set of weak messages.

For a hash function with l-bit hash value, it's ideal security strength against the second pre-image attack is that, for any message M, to find another message M' such that $h(M) = h(M')$ is not higher than the exhaustive search probability of 2^{-l}.

Table 4. Two collisions for RIPEMD. H is the hash value with little-endian and no message padding, and H^* is the hash value with big-endian and message padding

M_1	579faf8e	9ecf579	574a6aba	78413511	a2b410a4	ad2f6c9f	b56202c	4d757911
	bdeaae7	78bc91f2	47bc6d7d	9abdd1b1	a45d2015	817104ff	264758a8	61064ea5
M_1'	579faf8e	9ecf579	574a6aba	78513511	a2b410a4	ad2f6c9f	b56202c	4d757911
	bdeaae7	78bc91f2	c7c06d7d	9abdd1b1	a45d2015	817104ff	264758a8	e1064ea5
H	1fab152	1654a31b	7a33776a	9e968ba7				
H^*	dd6478dd	9a7d821c	aa018648	e5e792e9				
M_2	579faf8e	9ecf579	574a6aba	78413511	a2b410a4	ad2f6c9f	b56202c	4d757911
	bdeaae7	78bc91f2	47bc6d7d	9abdd1b1	a45d2015	a0a504ff	b18d58a8	e70c66b6
M_2'	579faf8e	9ecf579	574a6aba	78513511	a2b410a4	ad2f6c9f	b56202c	4d757911
	bdeaae7	78bc91f2	c7c06d7d	9abdd1b1	a45d2015	a0a504ff	b18d58a8	670c66b6
H	1f2c159f	569b31a6	dfcaa51a	25665d24				
H^*	88cea096	c773c29f	04cd9698	4a41d139				

Theorem 1 (Second Pre-Image Attack for Weak Messages). *For a weak message, we can find another message such that these two different messages produce the same hash code. The complexity is only one-time MD4 computation. A random selected message is weak with probability* 2^{-122}.

Proof. For any message M, we select M' such that

$$M' = M + \triangle M$$

$$\triangle M = M' - M = (\triangle m_0, \triangle m_1, ..., \triangle m_{15})$$
$$\triangle m_1 = 2^{31}, \triangle m_2 = 2^{31} - 2^{28}, \triangle m_{12} = -2^{16},$$
$$\triangle m_i = 0, 0 \leq i \leq 15, i \neq 1, 2, 12.$$

From the conditions in Table 6, we know that, if M satisfies all the 122 conditions, M' is the second pre-image of $h(M)$.

There are $2^{512}/2^{122} = 2^{391}$ one-block messages satisfy all the conditions. This completes the proof. □

Any message M can be modified with the techniques in Section 4 so that almost all the conditions in rounds 1-2 hold. For the resulting message, say M', we then find a second pre-image M'' of $h(M')$ with probability 2^{-2} to 2^{-6}. This fact can be interpreted as a chosen-message 2nd pre-image attack, since M' is not chosen freely but "close" to M. One message "close" to other message implies that the hamming weight of the difference for two messages is low. For example, given any random message M, if we only fulfil the the single-message modification, the chosen message M' is the 2nd pre-image of other message M'' with probability 2^{23} (excluding two conditions in 17-step). According to the conditions in Table 6, we can get M' by modifying M about 50 bits, so the difference hamming weight for two messages is 50 on average. When applying the multi-message modification, although the probability can be improved to

2^{-2} to 2^{-6}, the hamming weight may be greatly increased. The best method is to fulfil a kind of precise message modification, and correct a condition by increasing about 3 hamming weights. So, the difference hamming weight can be controlled within 110 on average.

7 Conclusion

In this paper, we have presented efficient collision search attacks on MD4 and RIPEMD. We have shown that only about 4 to 64 random selected messages are needed in order to find a collision of MD4, and only about 2^{16} random selected messages to for RIPEMD.

We have introduced three important analytical techniques that are very important for the effectiveness of the attacks:

1. How to find an efficient differential that is composed of one collision.
2. Determine all the conditions under which the collision happens.
3. For any message M, make some modification to M to guarantee that almost all the conditions hold.

Remarks. Our collision search attack on MD4 implies that for a weak message a 2nd pre-image can be found with complexity below 2^8. The probability for a random messages to be weak with respect to MD4 is 2^{-122}. However, this can be improved significantly. In fact note that Theorem 1 directly come from the collision differential path in Section 4, where the differential path is chosen to minimize the complexity of our collision attack. Hence it is not optimal for our pre-image attack. The number of weak messages is determined by the number of conditions specified in Table 1. By finding other differential paths with the least number of conditions, we believe that the probability of weak messages can be increased significantly. In fact, the latest 2nd pre-image attack can be improved to 2^{-72} which is found by Hongbo Yu Gaoli Wang et al.

We also note that for SHA-0, given any random message, it is a weak message with about probability 2^{-107} which is a surprising result compared to the exhaustive search probability 2^{-160}.

Acknowledgements

It is a pleasure to acknowledge Hans Dobbertin, Magnus Daum for their important advice, corrections, and suggestions, and for spending their precious time on our research.

Xiaoyun Wang's research is supported by the National Natural Science Foundation of China (Grant No. 90304009). Dengguo Feng's research is supported by 973 project (Grant No. G19990358).

References

1. E. Biham, A. Shamir, Differential Cryptanalysis of the Data Encryption Standard, Springer-Verlag, 1993.
2. E. Biham, R. Chen, *Near collision for SHA-0,* Advances in Cryptology, Crypto'04, 2004, LNCS 3152, pp. 290-305.
3. B. den Boer, A. Bosselaers, *Collisions for the compression function of MD5,* Advances in Cryptology, Eurocrypt'93.
4. F. Chaband, A. Joux, Differential Collisions in SHA-0, Advances in Cryptology, Crypto'98 Proceedings, 1998.
5. H. Dobbertin, Cryptanalysis of MD4, Fast Software Encryption, LNCS 1039, D. Gollmann, Ed., Springer-Verlag, 1996.
6. H. Dobbertin, Cryptanaltysis of MD5 Compress, Presented at the Rump Session of Eurocrypt'96.
7. H. Dobbertin, RIPEMD with Two Round Compress Function Is Not Collision-Free, Journal of Cryptology(1997) 10:51-69, 1997.
8. H. Dobbertin, The First Two Rounds of MD4 are Not One-Way, Fast Software Encryption, 1998.
9. H. Dobbertin, A. Bosselaers, B. Preneel, RIPMEMD-160:A Strengthened Version of RIPMMD, Fast Software Encryption, LNCS 1039, 1996.
10. FIPS 180-1, Secure hash standard, NIST, US Department of Commerce, Washington D. C., April 1995. Springer-Verlag, 1996.
11. FIPS 180-2, Secure Hash Standard, http://csrc.nist.gov/publications/,2002.
12. Joux, A., Collisions for SHA-0, Rump Session of CRYPTO'04, 2004.
13. RIPE,Integrity Primitives for Secure Information Systems, Final Report of RACE Integrity Primitives Evalutiobn(RIPE-RACE 1040), LNCS 1007, 1995.
14. R. L., Rivest, The MD4 Message Digest Algorithm, Crypo'90 Proceedings, 1991.
15. R. L. Rivest, The MD5 Message-Digest Algorithm, Request for Comments (RFC 1320), Internet Activities Board, Internet Privacy Task Force, April 1992.
16. Bart Van Rompay, A. Biryukov, B. Preneel, J. Vandewalle, Cryptanalysis of 3-pass HAVAL, Asiacrypto'03 Proceedings, pp. 228-245, 2003.
17. X.Y. Wang, F.D. Guo, X.J. Lai, H.B. Yu, Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD, Rump Session of Crypto'04, E-print, 2004.
18. Y. Zheng, J. Pieprzyk, J. Seberry, HAVAL–A One-way Hashing Algorithm with Variable Length of Output, Auscrypto'92 Proceedings, pp.83-104.

Appendix

In the appendix, we give the tables for the differential paths and the set of sufficient conditions that are used in our collision search attacks on MD4 and RIPEMD.

Table 5. Differential Characteristics in the Collision Differential for MD4

Step	Chaining value for M	$w_{j,i}$	Shift	Δm_i	The i-th step difference	The i-th output for M'
1	a_1	m_0	3			a_1
2	d_1	m_1	7	2^{31}	2^6	$d_1[7]$
3	c_1	m_2	11	$-2^{28}+2^{31}$	-2^7+2^{10}	$c_1[-8,11]$
4	b_1	m_3	19		2^{25}	$b_1[26]$
5	a_2	m_4	3			a_2
6	d_2	m_5	7		2^{13}	$d_2[14]$
7	c_2	m_6	11		$-2^{18}+2^{21}$	$c_2[19,20,-21,22]$
8	b_2	m_7	19		2^{12}	$b_2[-13,-14,15]$
9	a_3	m_8	3		2^{16}	$a_3[17]$
10	d_3	m_9	7		$2^{19}+2^{20}-2^{25}$	$d_3[20,-21,-22,23,-26]$
11	c_3	m_{10}	11		-2^{29}	$c_3[-30]$
12	b_3	m_{11}	19		2^{31}	$b_3[32]$
13	a_4	m_{12}	3	-2^{16}	$2^{22}+2^{25}$	$a_4[23,26]$
14	d_4	m_{13}	7		$-2^{26}+2^{28}$	$d_4[-27,-29,30]$
15	c_4	m_{14}	11			c_4
16	b_4	m_{15}	19		2^{18}	$b_4[19]$
17	a_5	m_0	3		$2^{25}-2^{28}-2^{31}$	$a_5[-26,27,-29,-32]$
18	d_5	m_4	5			d_5
19	c_5	m_8	9			c_5
20	b_5	m_{12}	13	-2^{16}	$-2^{29}+2^{31}$	$b_5[-30,32]$
21	a_6	m_1	3	2^{31}	$2^{28}-2^{31}$	$a_6[-29,30,-32]$
22	d_6	m_5	5			d_6
23	c_6	m_9	9			c_6
24	b_6	m_{13}	13			b_6
25	a_7	m_2	3	$-2^{28}+2^{31}$		a_7
...	...					....
36	b_9	m_{12}	15	-2^{16}	2^{31}	$b_9[-32]$
37	a_{10}	m_2	3	$-2^{28}+2^{31}$	2^{31}	$a_{10}[-32]$
38	d_{10}	m_{10}	9			d_{10}
39	c_{10}	m_6	11			c_{10}
40	b_{10}	m_{14}	15			b_{10}
41	a_{11}	m_1	3	2^{31}		a_{11}

Table 6. A Set of Sufficient Conditions for Collisions of MD4

a_1	$a_{1,7} = b_{0,7}$
d_1	$d_{1,7} = 0$, $d_{1,8} = a_{1,8}$, $d_{1,11} = a_{1,11}$
c_1	$c_{1,7} = 1$, $c_{1,8} = 1$, $c_{1,11} = 0$, $c_{1,26} = d_{1,26}$
b_1	$b_{1,7} = 1$, $b_{1,8} = 0$, $b_{1,11} = 0$, $b_{1,26} = 0$
a_2	$a_{2,8} = 1$, $a_{2,11} = 1$, $a_{2,26} = 0$, $a_{2,14} = b_{1,14}$
d_2	$d_{2,14} = 0$, $d_{2,19} = a_{2,19}$, $d_{2,20} = a_{2,20}$, $d_{2,21} = a_{2,21}$, $d_{2,22} = a_{2,22}$, $d_{2,26} = 1$
c_2	$c_{2,13} = d_{2,13}$, $c_{2,14} = 0$, $c_{2,15} = d_{2,15}$, $c_{2,19} = 0$, $c_{2,20} = 0$, $c_{2,21} = 1$, $c_{2,22} = 0$
b_2	$b_{2,13} = 1$, $b_{2,14} = 1$, $b_{2,15} = 0$, $b_{2,17} = c_{2,17}$, $b_{2,19} = 0$, $b_{2,20} = 0$, $b_{2,21} = 0$ $b_{2,22} = 0$
a_3	$a_{3,13} = 1$, $a_{3,14} = 1$, $a_{3,15} = 1$, $a_{3,17} = 0$, $a_{3,19} = 0$, $a_{3,20} = 0$, $a_{3,21} = 0$, $a_{3,23} = b_{2,23}$ $a_{3,22} = 1$, $a_{3,26} = b_{2,26}$
d_3	$d_{3,13} = 1$, $d_{3,14} = 1$, $d_{3,15} = 1$, $d_{3,17} = 0$, $d_{3,20} = 0$, $d_{3,21} = 1$, $d_{3,22} = 1$, $d_{3,23} = 0$, $d_{3,26} = 1$, $d_{3,30} = a_{3,30}$
c_3	$c_{3,17} = 1$, $c_{3,20} = 0$, $c_{3,21} = 0$, $c_{3,22} = 0$, $c_{3,23} = 0$, $c_{3,26} = 0$, $c_{3,30} = 1$, $c_{3,32} = d_{3,32}$
b_3	$b_{3,20} = 0$, $b_{3,21} = 1$, $b_{3,22} = 1$, $b_{3,23} = c_{3,23}$, $b_{3,26} = 1$, $b_{3,30} = 0$, $b_{3,32} = 0$
a_4	$a_{4,23} = 0$, $a_{4,26} = 0$, $a_{4,27} = b_{3,27}$, $a_{4,29} = b_{3,29}$, $a_{4,30} = 1$, $a_{4,32} = 0$
d_4	$d_{4,23} = 0$, $d_{4,26} = 0$, $d_{4,27} = 1$, $d_{4,29} = 1$, $d_{4,30} = 0$, $d_{4,32} = 1$
c_4	$c_{4,19} = d_{4,19}$, $c_{4,23} = 1$, $c_{4,26} = 1$, $c_{4,27} = 0$, $c_{4,29} = 0$, $c_{4,30} = 0$
b_4	$b_{4,19} = 0$, $b_{4,26} = c_{4,26} = 1$, $b_{4,27} = 1$, $b_{4,29} = 1$, $b_{4,30} = 0$
a_5	$a_{5,19} = c_{4,19}$, $a_{5,26} = 1$, $a_{5,27} = 0$, $a_{5,29} = 1$, $a_{5,32} = 1$
d_5	$d_{5,19} = a_{5,19}$, $d_{5,26} = b_{4,26}$, $d_{5,27} = b_{4,27}$, $d_{5,29} = b_{4,29}$, $d_{5,32} = b_{4,32}$
c_5	$c_{5,26} = d_{5,26}$, $c_{5,27} = d_{5,27}$, $c_{5,29} = d_{5,29}$, $c_{5,30} = d_{5,30}$, $c_{5,32} = d_{5,32}$
b_5	$b_{5,29} = c_{5,29}$, $b_{5,30} = 1$, $b_{5,32} = 0$
a_6	$a_{6,29} = 1$, $a_{6,32} = 1$
d_6	$d_{6,29} = b_{5,29}$
c_6	$c_{6,29} = d_{6,29}$, $c_{6,30} = d_{6,30} + 1$, $c_{6,32} = d_{6,32} + 1$
b_9	$b_{9,32} = 1$
a_{10}	$a_{10,32} = 1$

Table 7. Differential Characteristics in a Collision Differential of RIPEMD

Step	Chaining value for M	$w_{j,i}$	Shift	Δm_i	The i-th step difference	The i-th output for M'
0	a_1	m_0	11			a_1
1	d_1	m_1	14			d_1
2	c_1	m_2	15			c_1
3	b_1	m_3	12	2^{20}	1	$b_1[-1, -2, -3, 4]$
4	a_2	m_4	5		2^6	$a_2[7]$
5	d_2	m_5	8		$2^9 - 2^{11}$	$d_2[10, -12]$
6	c_2	m_6	7		$2^{16} - 2^{18}$	$c_2[17, -19]$
7	b_2	m_7	9		$2^9 + 2^{25} + 2^{27}$	$b_2[10, -26, 27, 28]$
8	a_3	m_8	11		$-2^5 + 2^{17}$	$a_3[-6, 18]$
9	d_3	m_9	13		-2^{23}	$d_3[24, 25, -26]$
10	c_3	m_{10}	14	$2^{18} + 2^{31}$	$-2^{13} + 2^{30}$	$c_3[-14, 31]$
11	b_3	m_{11}	15		$2^{10} + 2^{24}$	$b_3[11, 25]$
12	a_4	m_{12}	6		$-2^{11} + 2^{23}$	$a_4[12, 13, -14, 24]$
13	d_4	m_{13}	7			d_4
14	c_4	m_{14}	9		$2^7 - 2^{23}$	$c_4[8, 24, -25]$
15	b_4	m_{15}	8	2^{31}	$-2^7 + 2^{18}$	$b_4[-8, 19]$
16	a_5	m_7	7		-2^{18}	$a_5[-19]$
17	d_5	m_4	6			d_5
18	c_5	m_{13}	8		2^{31}	$c_5[-32]$
19	b_5	m_1	13		-2^{20}	b_5[-21]
20	a_6	m_{10}	11	2^{18}+2^{31}		a_6
21	d_6	m_6	9			d_6
22	c_6	m_{15}	7	2^{31}		c_6
23	b_6	m_3	15	2^{20}		b_6
24	a_7	m_{12}	7			a_7
25	d_7	m_0	12			d_7
26	c_7	m_9	15			c_7
27	b_7	m_5	9			b_7
28	a_8	m_{14}	7			a_8
29	d_8	m_2	11			d_8
30	c_8	m_{11}	13			c_8
31	b_8	m_8	12			b_8
32	a_9	m_3	11	2^{20}	2^{31}	$a_9[32]$
33	d_9	m_{10}	13	$2^{18} + 2^{31}$	2^{31}	$d_9[32]$
34	c_9	m_2	14			c_9
35	b_9	m_4	7			b_9
36	a_{10}	m_9	14			a_{10}
37	d_{10}	m_{15}	9	2^{31}		d_{10}

Table 8. A Set of Sufficient Conditions for Collisions of RIPEMD

a_1	
d_1	$d_{1,2} = 1$
c_1	$c_{1,1} = d_{1,1}$, $c_{1,2} = 0$, $c_{1,3} = d_{1,3}$, $c_{1,4} = d_{1,4}$
b_1	$b_{1,1} = 1$, $b_{1,2} = 1$, $b_{1,3} = 1$, $b_{1,4} = 0$, $b_{1,7} = c_{1,7}$
a_2	$a_{2,7=0}$, $a_{2,1} = 0$, $a_{2,2} = 1$, $a_{2,3} = 1$, $a_{2,4} = 0$, $a_{2,10} = \overline{b_{1,10}} = 1$, $a_{2,12} = \overline{b_{1,12}} = 1$ $a_{2,17} = 0$
d_2	$d_{2,1} = 1$, $d_{2,2} = 1$, $d_{2,3} = 1$, $d_{2,4} = 1$, $d_{2,7} = 0$, $d_{2,10} = 0$, $d_{2,12} = 1$, $d_{2,17} = 1$, $d_{2,19} = \overline{a_{2,19}} = 0$
c_2	$c_{2,17} = 0$, $c_{2,19} = 1$, $c_{2,10} = 0$, $c_{2,7} = 1$, $c_{2,12} = 0$, $c_{2,26} = d_{2,26}$, $c_{2,27} = \overline{d_{2,27}} = 0$, $c_{2,28} = d_{2,28}$
b_2	$b_{2,6} = c_{2,6}$, $b_{2,10} = 0$, $b_{2,12} = 1$, $b_{2,17} = 0$, $b_{2,18} = c_{2,18}$, $b_{2,19} = 0$, $b_{2,26} = 1$, $b_{2,27} = 0$, $b_{2,28} = 0$
a_3	$a_{3,6} = 1$, $a_{3,10} = 1$, $a_{3,17} = 1$, $a_{3,18} = 0$,$a_{3,19} = 1$, $a_{3,24} = b_{2,24}$, $a_{3,25} = b_{2,25}$, $a_{3,26} = 0$, $a_{3,27} = 0$, $a_{3,28} = 0$
d_3	$d_{3,6} = 0$, $d_{3,10} = 1$, $d_{3,14} = a_{3,14}$, $d_{3,18} = 0$, $d_{3,24} = 0$, $d_{3,25} = 0$, $d_{3,26} = 1$, $d_{3,27} = 1$, $d_{3,28} = 1$, $d_{3,31} = a_{3,31}$
c_3	$c_{3,6} = 1$, $c_{3,11} = d_{3,11}$, $c_{3,14} = 1$, $c_{3,18} = 1$, $c_{3,24} = 0$, $c_{3,25} = 0$, $c_{3,26} = 1$, $c_{3,31} = 0$
b_3	$b_{3,11} = 0$, $b_{3,12} = c_{3,12}$, $b_{3,13} = c_{3,13}$, $b_{3,14} = 0$, $b_{3,24} = 1$, $b_{3,25} = 0$, $b_{3,26} = 1$, $b_{3,31} = 0$
a_4	$a_{4,11} = 0$, $a_{4,12} = 0$, $a_{4,13} = 0$, $a_{4,14} = 1$, $a_{4,24} = 0$, $a_{4,25} = 0$, $a_{4,31} = 1$
d_4	$d_{4,8} = a_{4,8}$, $d_{4,11} = 1$, $d_{4,12} = 0$, $d_{4,13} = 0$, $d_{4,14} = 1$, $d_{4,24} = 0$, $d_{4,25} = 1$,
c_4	$c_{4,8} = 0$, $c_{4,12} = 1$, $c_{4,13} = 1$, $c_{4,14} = 1$, $c_{4,19} = d_{4,19}$, $c_{4,24} = 0$, $c_{4,25} = 1$,
b_4	$b_{4,8} = 1$, $b_{4,19} = 0$, $b_{4,24} = \overline{d_{4,24}}$, $b_{4,25} = \overline{d_{4,25}}$
a_5	$a_{5,19} = 1$, $a_{5,24} = b_{4,24}$, $a_{5,25} = b_{4,25}$
d_5	$d_{5,8} = \overline{a_{5,8}}$, $d_{5,32} = a_{5,32}$
c_5	$c_{5,19} = \overline{d_{5,19}}$, $c_{5,21} = d_{5,21}$, $c_{5,32} = 1$
b_5	$b_{5,21} = 1$, $b_{5,32} = \overline{d_{5,32}}$
a_6	$a_{6,21} = c_{5,21}$, $a_{6,32} = b_{5,32}$
d_6	$d_{6,21} = a_{6,21}$
a_9	$a_{9,32} = 0$,
d_9	$d_{9,32} = 0$
a_{10}	