Fast Collision Attack on MD5

Tao Xie[1,2], Fanbao Liu[2**], Dengguo Feng[3]

[1]The Center for Soft-Computing and Cryptology, NUDT, Changsha, China
[2]School of Computer, NUDT, Changsha, 410073, Hunan, China
[3] State Key Lab of Information Security, Chinese Academy of Sciences, Beijing, 100190, China
hamishxie@vip.sina.com
liufanbao@gmail.com[†]

Abstract. We presented the first single block collision attack on MD5 with complexity of 2^{47} MD5 compressions and posted the challenge for another completely new one in 2010. Last year, Stevens presented a single block collision attack to our challenge, with complexity of 2^{50} MD5 compressions. We really appreciate Stevens's hard work. However, it is a pity that he had not found even a better solution than our original one, let alone a completely new one and the very optimal solution that we preserved and have been hoping that someone can find it, whose collision complexity is about 2^{41} MD5 compressions. In this paper, we propose a method how to choose the optimal input difference for generating MD5 collision pairs. First, we divide the sufficient conditions into two classes: strong conditions and weak conditions, by the degree of difficulty for condition satisfaction. Second, we prove that there exist strong conditions in only 24 steps (one and a half rounds) under specific conditions, by utilizing the weaknesses of compression functions of MD5, which are difference inheriting and message expanding. Third, there should be no difference scaling after state word q_{25} so that it can result in the least number of strong conditions in each differential path, in such a way we deduce the distribution of strong conditions for each input difference pattern. Finally, we choose the input difference with the least number of strong conditions and the most number of free message words. We implement the most efficient 2-block MD5 collision attack, which needs only about 2^{18} MD5 compressions to find a collision pair, and show a single-block collision attack with complexity 2^{41}.

Keywords: Hash Function; MD5 Differential Cryptanalysis; Collision Attack; Single-Block Collision

1 Introduction

Hash function, mapping input message with arbitrary lengths to fixed lengths output, is an one-way cryptographic primitive. Hash functions are mainly used to generate digital fingerprint, and widely applied in the area of Random Number Generation (RNG), message integrity check, password shadow, challenge-and-response, Message Authentication Code (MAC), digital signature, digital certification, et al.

The most widely used hash functions are MD4 family iterated hash functions [6, 1], derived from MD4 [9] designed by Rivest in 1990. The family includes MD4, MD5 [10], SHA [7, 3] and SHA-2 [8], et al. The first one used in practical is MD5 [10], designed as the strengthened version of MD4 by Rivest in 1992.

We presented the first single block collision attack on MD5 with complexity of 2^{47} MD5 compressions with no details disclosed, and posted the challenge for another completely new one in 2010 [14]. In 2012, Stevens presented a single block collision attack to answer our challenge, with complexity of 2^{50} MD5 compressions [12]. The input difference pattern of Stevens's can be easily derived from ours.

In this paper, we propose a method how to choose the optimal input difference for generating MD5 collision pairs. First, we divide the sufficient conditions into two classes: strong condition and weak condition, according to the degree of difficulty for condition satisfaction. Second, we prove that there exist strong conditions in only 24 steps (one and a half rounds) under specific conditions, by utilizing the weaknesses of compression functions of MD5, which are difference inheriting and message expanding. Third, there should be no difference scaling after state word q_{25} so that it can result in the least number of strong conditions for each differential path, in such a way we deduce the distribution of strong conditions for each input difference pattern. Finally, we choose the input difference with the least number of strong conditions and the most number of free message words. We further apply the divide-and-conquer strategy to cut the MD5 collision searching into stages, to make the relations of all stages' complexity to be additive instead of multiplicative. We also propose a scheme named group satisfaction —to determinately satisfy the strong conditions of the first three steps in the last tunnel under the divide-and-conquer strategy, and randomly satisfy other strong conditions using the rest of free bits of the tunnel, so as to greatly reduce the complexity of

[†] Corresponding author.

MD5 collision searching. Hence, we should construct differential paths with the most number of free bits to support the divide-and-conquer strategy and tunnel technique. The details of such method will be appeared in our full paper [15]. Applying the above methods, we implement the most efficient MD5 collision attack, which only needs about 2^{18} MD5 compressions to find a collision pair. These methods are also applicable to other hash functions with MD (Merkle-Damgård) construction.

We also show how to find right input differences for single block collision attack on MD5. Moreover, we compare Stevens' work [12] to ours and we find that his response may not achieve our original target of the challenge, and that is why we have decided to give him a half of the award.

2 Preliminaries

2.1 MD5 Algorithm

MD5 [10] is a typical Merkle-Damgård structure hash function, it takes a variable-length message M as an input and outputs a 128-bit hash value $MD5(M)$.

The input message M should be pre-processed before being hashed, which is divided into the following three stages:

1. M is padded with padding bits (a '1' followed by several '0's to 448 mod 512) and the length of M with 64 bits, to the exact multiples of 512 bits.
2. The padded M' is divided into chunks of 512-bit blocks $(M_0, M_1, \ldots, M_{(|M'|/512-1)})$.
3. Each block M_i is further divided into sixteen 32-bit words $(m_0, m_1, \ldots, m_{15})$.

Compression Function of MD5. Each block is processed by MD5 compression function (CF). CF takes M_i and a 128-bit chaining variable H_i as input, and outputs H_{i+1}. The initiate chaining variable H_0 is set to certain constants, $a_0 =$ `0x67452301`, $b_0 =$ `0xefcdab89`, $c_0 =$ `0x98badcfe`, $d_0 =$ `0x10325476`. The iterated procedure of MD5 algorithm is shown as follows, where H_n is the exact $MD5(M)$.

$$H_1 = CF(M_0, H_0),\ H_2 = CF(M_1, H_1), \ldots, H_n = CF(M_{n-1}, H_{n-1}). \tag{1}$$

CF consists of 64 steps. Steps 1-16, steps 17-32, steps 33-48 and steps 49-64 are called round r_1, r_2, r_3 and r_4, respectively. Let q_i $(1 \le i \le 64)$ represent the 32-bit state of step i, and $q_{i,j}$ stand for the value of the j-th $(j\ (0 \le j \le 31))$ bit of q_i. With initiated chaining variables $q_{-3} = a_0$, $q_0 = b_0$, $q_{-1} = c_0$, $q_{-2} = d_0$, q_i $(1 \le i \le 64)$ is updated in (2).

$$q_i = q_{i-1} + (q_{i-4} + f_i(q_{i-1}, q_{i-2}, q_{i-3}) + w_i + t_i) \lll s_i \tag{2}$$

Each state word q_i uses modular addition $+$, left rotation $\lll$ and round dependent Boolean function f_i.

The details of f_i are shown in (3).

$$f_i = \begin{cases} F(B,C,D) = (B \wedge C) \vee (\neg B \wedge D), & i \in r_1, \\ G(B,C,D) = (B \wedge D) \vee (C \wedge \neg D), & i \in r_2, \\ H(B,C,D) = B \oplus C \oplus D, & i \in r_3, \\ I(B,C,D) = C \oplus (B \vee \neg D), & i \in r_4. \end{cases} \tag{3}$$

where $\oplus$, $\wedge$, $\vee$ and $\neg$ denote the logic operations *XOR*, *AND*, *OR* and *NOT*, respectively. B, C and D are 32-bit state words.

Message word w_i is one of $(m_0, m_1, \ldots, m_{15})$, the distribution of w_i is called message expanding, which is shown in (4).

$$w_i = \begin{cases} m_{i-1}, & i \in r_1, \\ m_{(5i-4) \mod 16}, & i \in r_2, \\ m_{(3i+2) \mod 16}, & i \in r_3, \\ m_{7(i-1) \mod 16}, & i \in r_4. \end{cases} \tag{4}$$

The constant t_i is defined in (5).

$$t_i = \lfloor 2^{32} \cdot |sin(i)| \rfloor \tag{5}$$

$\lll s_i$ denote the left rotation of s_i bits, $\ggg$ denote the corresponding right rotation. The details of the rotations are shown in (6).

$$(s_i,\ s_{i+1},\ s_{i+2},\ s_{i+3}) = \begin{cases} (7, 12, 17, 22), & i = 1, 5, 9, 13, \\ (5, 9, 14, 20), & i = 17, 21, 25, 29, \\ (4, 11, 16, 23), & i = 33, 37, 41, 45, \\ (6, 10, 15, 21), & i = 49, 53, 57, 61. \end{cases} \tag{6}$$

If all of the 64 steps are computed, the chaining variables are updated by adding the last four state words to finish one call to the compression function.

3 Differential Cryptanalysis on MD5

3.1 Differences

Definition 1. *Let F_2 be the binary field, F_2^n be an n-dimensional vector space over F_2, and X, $X' \in F_2^n$. A bitwise XOR difference (bitwise addition modulo 2) between X and X' is called XOR difference, denoted as $\triangle^{\oplus}X$. A integer subtraction modulo 2^n between X and X' is called modular difference, denoted as ΔX. A bitwise subtraction modulo 2 difference between X and X' is called signed difference, denoted as $\triangle^{\pm}X$.*

For the sake of simplicity, let $n = 10$, $X = 1001000101$, $X' = 0000111010$, then $\triangle^{\oplus}X$, ΔX and $\triangle^{\pm}X$ are computed as (7), (8) and (9), respectively.

$$\triangle^{\oplus}X = X \oplus X' = \coprod_{i=0}^{n-1} X_i \oplus X'_i = 1001111111 \tag{7}$$

$$\Delta X = (X - X') \mod 2^n = (\sum_{i=0}^{n-1} 2^i X_i - \sum_{i=0}^{n-1} 2^i X'_i) \mod 2^n = 1000001011 \tag{8}$$

$$\triangle^{\pm}X = \coprod_{i=0}^{n-1} (X_i - X'_i) = 1001 - 1 - 1 - 11 - 11 \tag{9}$$

We omit those '0's in the signed difference $\triangle^{\pm}X$, and index the positions of the non-zero differences with their signs ('+', '-'). For example, the signed difference $\triangle^{\pm}X = 1001 - 1 - 1 - 11 - 11$ can be represented simply by $[9, -4 + 6, 2 - 3, 0 - 1]$.

Definition 2. *The hamming weight of $\omega(\Delta X)$ denotes the number of non-zero difference of ΔX, and δ_i denote the difference corresponding to the i_{th} bit indexed starting from the least significant bit (LSB) of ΔX. Let ν_i denote the number of signed differences of δ_i.*

For example, for the modular difference $\Delta X = 2^0 + 2^5 + 2^{29}$, where $\omega(\Delta X) = 3$, $\delta_1 = 2^0$, $\delta_2 = 2^5$ and $\delta_3 = 2^{29}$. The values of ν_i can be computed as follows.

$$\nu_i = \begin{cases} n - \log \delta_{\omega(\Delta X)} + 1 & i = \omega(\Delta X) \\ \log \delta_{i+1} - \log \delta_i & i \neq \omega(\Delta X) \end{cases} \tag{10}$$

For the above example, we have $\nu_1 = \log 2^5 - \log 2^0 = 5$, $\nu_2 = 24$ and $\nu_3 = 3$.

Lemma 1. *Let X, $X' \in F_2^n$ be a pair of n-dimensional vectors, a signed difference $\triangle^{\pm}X$ can determine only one XOR difference $\triangle^{\oplus}X$ or modular difference ΔX. [2]*

Here, we give a more intuitive proof than that in [2] as follows.

Proof. From the definitions of XOR difference $\triangle^{\oplus}X$ and signed difference $\triangle^{\pm}X$ between X, $X' \in F_2^n$, we know that as follows.

$$\triangle^{\oplus}X = \coprod_{i=0}^{n-1} (X_i \oplus X'_i) = \coprod_{i=0}^{n-1} |X_i - X'_i| = \coprod_{i=0}^{n-1} |\triangle^{\pm}X_i| \tag{11}$$

For each $\triangle^{\pm}X_i = X_i - X'_i$, the value of $\triangle^{\pm}X_i$ has three possibilities: 0, 1 and -1. If $\triangle^{\pm}X_i = 0$, then $X_i = X'_i$ contributes nothing (0) for modular difference ΔX, we have $0 \cdot 2^i$. If $\triangle^{\pm}X_i = 1$, then $X_i = 1$ and $X'_i = 0$, the corresponding contribution to modular difference is $1 \cdot 2^i$. Similarly, $\triangle^{\pm}X_i = -1$ contributes $-2^i = -1 \cdot 2^i$ to modular difference ΔX. Hence, we have:

$$\triangle^{\pm}X = \coprod_{i=0}^{n-1}(X_i - X'_i) = \coprod_{i=0}^{n-1}\triangle^{\pm}X_i \Rightarrow \sum_{i=0}^{n-1} 2^i \triangle^{\pm}X_i \mod 2^n = \Delta X \tag{12}$$

□

Lemma 2. *Let X, $X' \in F_2^n$ be a pair of n-dimensional vectors, a signed difference $\triangle^{\pm}X$ can be determined uniquely by a modular difference ΔX and a XOR difference $\triangle^{\oplus}X$.*

Proof. The proof is listed as follows.

$$\begin{aligned}
\Delta X &= (X - X') \mod 2^n = (\sum_{i=0}^{n-1} 2^i X_i - \sum_{i=0}^{n-1} 2^i X'_i) \mod 2^n \\
&= \sum_{i=0}^{n-1} 2^i (X_i - X'_i) \mod 2^n = \sum_{i=0}^{n-1} 2^i (X_i - X'_i)|X_i - X'_i| \mod 2^n \\
&\Rightarrow \sum_{i=0}^{n-1} 2^i \triangle^{\pm}X_i \triangle^{\oplus}X_i \mod 2^n = \triangle^{\pm}X
\end{aligned} \tag{13}$$

Theorem 1. *Let X, $X' \in F_2^n$ be a pair of n-dimensional vectors, a signed difference $\triangle^{\pm}X$ can be determined uniquely by a modular difference ΔX and a XOR difference $\triangle^{\oplus}X$, and vice versa. Hence, a signed difference $\triangle^{\pm}X$ is equivalent to a modular difference ΔX combined with XOR difference $\triangle^{\oplus}X$.*

Proof. This theorem can be deduced directly by Lemma 1 and Lemma 2. □

Definition 3. *Difference scaling: If $\Delta q_i = \Delta q'_i$ and further $\omega(\triangle^{\pm}q'_i) > \omega(\triangle^{\pm}q_i)$, then we call ($\Delta q'_i$, $\triangle^{\pm}q'_i$) is a scaling case of (Δq_i, $\triangle^{\pm}q_i$). The process from $\triangle^{\pm}q_i$ to $\triangle^{\pm}q'_i$ is called a difference scaling.*

Theorem 2. *Δq_i may have as many as $\prod_{i=1}^{\omega(\Delta X)} \nu_i - 1$ ways of difference scaling.*

Proof. Each Δq_i may have $\prod_{i=1}^{\omega(\Delta q_i)} \nu_i$ kinds of different signed differences. When it excludes the case of itself, Δq_i may have as many as $\prod_{i=1}^{\omega(\Delta X)} \nu_i - 1$ different ways of difference scaling.

3.2 MD5 Weakness: Message Expanding

Proposition 1. *The message modification technique can not be applied in state words after q_{26} of the second round r_2 of MD5.*

Proof. If the differential path before the state word q_i ($i < 27$) is not changed (at least keep the values of q_{i-1} to q_{i-4} unchanged)when a message modification is applied, we call it a successful application of message modification. The only way is to modify the free bits of input messages so that the differential path can be hold unchanged, hence enough free bits are necessary for message modification.

The input words m_0, m_1, m_4, m_5, m_6, m_9, m_{10}, m_{11}, m_{14} and m_{15} are used in the state updating from q_{17} to q_{26} in round r_2, so these input words can not be used once more as free words after sate q_{26}. Otherwise, the differential path before q_{26} may be destroyed and a message modification fails.

For example, considering $w_{27} = m_3$ of q_{27}. Since m_4 to m_6 have been used in the second round r_2, a message modification can not be successfully employed in q_{27}. Similarly, we can deduce that the rest of state words in r_2 can not employ message modification technique.

Weak conditions and strong conditions Each sufficient condition is satisfied with probability of 1/2, by using the random input messages. Therefore, we can divide these conditions into two classes: weak conditions and strong conditions, in an ideal way[1], by the degree of difficulty of satisfaction.

Definition 4. *The sufficient conditions before state q_{25} are called weak conditions, in a ideal situation. The sufficient conditions after state q_{25} are called strong conditions.*

The sufficient conditions after state q_{27} could be satisfied randomly, according to the limitation of applying message modification technique. The sufficient conditions of state q_{25} and q_{26} may be satisfied determinately, depending on the number of free bits. Hence, it is hard to some extent to satisfy the conditions after state q_{25}.

3.3 MD5 Weakness: Difference Inheritance

MSB Difference of Round r_3. The round function H of r_3 is a linear bit-by-bit XOR function, four continuous state differences 2^{31} can transfer with probability 1 if no input difference exists.

Proposition 2. *Let $\Delta q_{i-1} = \Delta q_{i-2} = \Delta q_{i-3} = \Delta q_{i-4} = 0$, where $33 \leq i \leq 44$, the sufficient conditions of generating four continuous state differences $\Delta q_i = \Delta q_{i+1} = \Delta q_{i+2} = \Delta q_{i+3} = 2^{31}$ are shown as follows.*

$$\begin{cases} \Delta w_i = 2^{31} \ggg s_i \\ \omega(\triangle^{\pm} \sum q_i) = 1 \\ \Delta w_{i+1} = 2^{31} \\ \Delta w_{i+2} = 0 \\ \Delta w_{i+3} = 2^{31} \end{cases} \tag{14}$$

Proof. We have $\Pr(H(b,c,d) = \neg H(\neg b, c, d)) = 1$, and $\omega(2^{31}) = 1$, applying the properties of round function H. Hence, the following equation holds

$$\Pr(\Delta f_{i+1}(q_{i,31}, q_{i-1,31}, q_{i-2,31}) = \Delta H(q_{i,31}, q_{i-1,31}, q_{i-2,31}) = 2^{31}) = 1 \tag{15}$$

Similarly, since $\Pr(H(b,c,d) = H(\neg b, \neg c, d)) = 1$ and $\Pr(H(b,c,d) = \neg H(\neg b, \neg c, \neg d)) = 1$, we have

$$\begin{aligned} &\Pr(\Delta f_{i+2}(q_{i+1,31}, q_{i,31}, q_{i-1,31}) = 0) = 1 \\ &\Pr(\Delta f_{i+3}(q_{i+2,31}, q_{i+1,31}, q_{i,31}) = 2^{31}) = 1 \end{aligned} \tag{16}$$

Proposition 3. *Four continuous state differences 2^{31} can transfer with probability 1 in case of no input difference.*

Proof. Let q_j to q_{j+3} ($j \geq 29$) be four continuous states, and their differences are 2^{31}. By the state updating in r_3, if $33 \leq i \leq 48$, we have

$$\Delta q_i = \Delta q_{i-1} + (\Delta q_{i-4} + \Delta H(q_{i-1}, q_{i-2}, q_{i-3}) + \Delta w_i) \lll s_i \qquad 33 \leq i \leq 48 \tag{17}$$

Since $\Pr(H(b,c,d) = \neg H(\neg b, \neg c, \neg d)) = 1$, and $\omega(2^{31}) = 1$, then we have

$$\Delta H(q_{i-1}, q_{i-2}, q_{i-3}) = \Delta H(q_{i-1,31}, q_{i-2,31}, q_{i-3,31}) = 2^{31} \qquad 33 \leq i \leq 48 \tag{18}$$

Since $\Delta w_i = 0$ (without input difference), we have

$$\Pr(2^{31} = \Delta f_i(q_{i-1}, q_{i-2}, q_{i-3})) = 1 \qquad 33 \leq i \leq 48 \tag{19}$$

Proposition 4. *Continuous state differences 2^{31} can be generated and transferred with probability 1/2, by fixing some input differences.*

Proof. If all 5 sufficient conditions are satisfied, then four continuous state differences 2^{31} can be generated with probability 1, by Proposition 2. Since four of them are input differences which can be fixed with probability 1, we should only consider condition $\omega(\triangle^{\pm} \sum q_i) = 1$, with probability $\Pr(\omega(\triangle^{\pm} \sum q_i) = 1) = 1/2$. Hence, with these input differences pattern, four continuous state differences can be generated with probability 1/2.

Four continuous state differences 2^{31} can transfer with probability 1 by Proposition 3.

[1] We assume that before state q_{25}, the message modification techniques can be employed unconditionally, with sufficient free bits.

The implementation strategy of practical MD5 collision attack We should just focus on choosing specific input difference patterns and evaluating the complexity of r_4 and r_2 after step q_{25}. By counting the number of strong conditions of these input differences, we can launch successful practical collision attacks if their number of strong conditions are less than 64.

4 MD5 Collision Attack

Conditions Relaxing We consider how to relax the conditions residing on the chaining variable ΔH_1 and construct enough differential paths, to reduce the complexity of MD5 collision searching. If there are 4 MSB differences 2^{31} ($q_{-3} = 2^{31}$, $q_{-2} = 2^{31}$, $q_{-1} = 2^{31}$, $q_0 = 2^{31}$) in IV, then we should construct 7 differential paths to include all branches of the difference in IV. q_{-3} will not be handled anywhere in the round function F, and we can omit its signed differences. q_{-2} to q_0 each has two kinds of signed differences, which total up to be $2 \times 2 \times 2 = 8$. Since the dBB collision includes two situations ($q_{-2,31} = q_{-1,31} = q_{0,31} = 1$ and $q_{-2,31} = q_{-1,31} = q_{0,31} = 0$), 7 differential paths are enough for relaxing two strong conditions residing on IV $\Delta H_1(q_{-2,31} = q_{-1,31} = q_{0,31})$.

Table 1: The number of strong conditions and free words of some input differences

Input difference	Strong conditions	Free words
$\Delta m_5 = 2^{10}$	26	5
$\Delta m_8 = 2^{25}$	29	8
$\Delta m_{11} = 2^{21}$	27	11
$\Delta m_{14} = 2^{16}$	37	14
$\Delta m_5 = 2^{31}$	30	5
$\Delta m_8 = 2^{31}$	25	8
$\Delta m_{11} = 2^{31}$	38	11
$\Delta m_5 = 2^{31}$, $\Delta m_{11} = 2^{31}$	25	5
$\Delta m_{14} = 2^{31}$	43	14
$\Delta m_8 = 2^{31}$, $\Delta m_{14} = 2^{31}$	29	8
$\Delta m_4 = 2^{20}$, $\Delta m_7 = 2^{31}$, $\Delta m_{13} = 2^{31}$	40	4
$\Delta m_{13} = 2^7$, $\Delta m_0 = 2^{31}$, $\Delta m_6 = 2^{31}$	37	0
$\Delta m_6 = 2^8$, $\Delta m_9 = 2^{31}$, $\Delta m_{15} = 2^{31}$	35	6
$\Delta m_9 = 2^{27}$, $\Delta m_{12} = 2^{31}$, $\Delta m_2 = 2^{31}$	29	2
$\Delta m_{11} = 2^{15}$, $\Delta m_{14} = 2^{31}$, $\Delta m_4 = 2^{31}$	28	4

In Crypt 2009, Stevens et al. presented a "fastest" collision attack on MD5, which needs about 2^{16} MD5 compressions [13]. Such an attack is also based on a sufficient condition relaxing, which removes about ten sufficient conditions on the chaining variable. In [5], the authors proved that such an attack is infeasible in practice.

Input Difference Choosing Considering both strong conditions and free words, we choose input difference ($\Delta M_0 = (\Delta m_8 = 2^{31})$, $\Delta M_1 = 0$ or $\Delta M_1 = (\Delta m_8 = 2^{31})$) to generate MD5 collision with two blocks. Based on the initial chaining variables with non difference $\Delta H_0 = 0$, the first block with input difference ΔM_0 is used to generate difference of $\Delta H_1 = (q_{-3} = 2^{31}, q_{-2} = 2^{31}, q_{-1} = 2^{31}, q_0 = 2^{31})$. With the difference of chaining variable ΔH_1, the second block with input difference ΔM_1 is used to generate $\Delta H_2 = 0$, hence an exact collision.

An overall analysis is shown as follows.

1. The first block M_0.
   a. The target of M_0 is to generate difference $\Delta H_1 = (2^{31}, 2^{31}, 2^{31}, 2^{31})$. The MSB differences in r_3 are inherited from r_2, the number of strong conditions in r_3 is 0, there are 16 strong conditions in total[2], the number of strong conditions after q_{25} in r_2 is 7. Therefore, the total number of strong conditions is 23.
   b. The words before m_8 are all free words, hence, there are enough free words.
2. The second block M_1.
   a. If ΔH_1 is the right dBB conditions, then the second block has difference $\Delta M_1 = 0$ for dBB collision generation. There are no condition in r_3, there are 16 conditions in r_4, there are 4 strong conditions after q_{25} in r_2, hence, there are 20 strong conditions in total. Since, each state word $q_i, (i \in r_1)$ has only one condition, there are enough free bits.

[2] Two conditions $q_{-2,31} = q_{-1,31} = q_{0,31}$ in H_1 are relaxed.

b. If ΔH_1 is not dBB conditions, then the second block has difference $\Delta M_1 = (\Delta m_8 = 2^{31})$. The differential path after q_{25} of such paths are the same as that of first block, hence, the number of strong conditions is 23.

4.1 Collision Searching Algorithm

After constructing the differential path and deducing the sufficient conditions, we should construct corresponding collision searching algorithms to get collisions fast. We explore the free bits in each differential path for applying advanced message modification and tunnel technique to search more bits and move the start point of search backwards, which will eventually increase the searching efficiency. Hence, based on tunnel technique, we group the collision searching by applying divide-and-conquer strategy for high efficiency, moreover, we further propose a scheme named group satisfaction to determinately satisfy all conditions of the first three steps in the last tunnel.

Group Satisfaction Scheme *Based on both of the tunnel and advanced message modification, we determinately satisfy all conditions of the first three steps in the last tunnel or the last phase of divide-and-conquer strategy, so as to achieve a full speed of collision searching.*

Advantage. Most of the former collision searching algorithms [4, 11], intended to randomly satisfy sufficient conditions of the last tunnel, but we satisfy all conditions of the first three steps in the last tunnel, which can greatly increase the collision probability and improve the searching efficiency.

Algorithm Implementation We divide the searching of the first block to generate dBB conditions into five stages, which are shown as follows.

- **Stage 1**: Set the conditions of q_3 to q_{16} to be true by basic message modification, randomly set non-conditional bits. Compute the values of m_6 to m_{15}.
- **Stage 2**:

1. Set the conditions of q_{17} to be true, randomly set non-conditional bits. Check whether the carrier conditions of q_{17} are satisfied, if not, goto step 1.
2. Set the conditions of q_{18} to be true, randomly set non-conditional bits. Check whether the carrier conditions of q_{18} are satisfied, if not, goto step 1. Compute m_6 by q_{18}, and recompute q_7 and check its conditions, if not satisfied, goto Stage 1.
3. Compute q_{19} and check its conditions, if not satisfied then apply advance message modification. Check the carrier condition, if not satisfied, goto Stage 1.

- **Stage 3**:

a. Set the conditions of q_{20} to be true, randomly set non-conditional bits. Check whether the carrier conditions of q_{20} are satisfied, if not, goto step a.
b. Compute q_{21} and check its conditions, if not satisfied then apply advance message modification. Check the carrier condition, if not satisfied, goto step a.
c. Compute q_{22} and check its conditions, if not satisfied then apply advance message modification. Check the carrier condition and $q_{22,29}$ and $q_{22,29}$, if any is not satisfied, goto step a.
d. Compute q_{23} and check its conditions, if not satisfied, goto step a. Compute message words m_2 to m_4, m_7 to m_9 and m_{12} to m_{13} from corresponding state words in r_1.

- **Stage 4**:

i. Satisfy the conditions of q_{24} by searching q_4 tunnel in brute force. If all free bits are used, goto step a.

- **Stage 5**:

x. Satisfy the conditions of q_{25} and q_{26} using group satisfaction scheme in q_9 tunnel. If fails, goto step i.
y. Compute q_{27} to q_{64} one by one, and check its conditions, if not satisfied, goto step x.

Complexity Analysis In fact, the differential path of the first block has weak conditions as many as 110 from q_{17} in r_2 to the final step q_{64}, and there are still 39 strong conditions after the application of advanced message modification. The dBB collision of the second block has 28 strong conditions to be satisfied randomly. However, thanks to the divide-and-conquer strategy, the practical complexity is much lower than expected. The details are shown as follows.

The complexity of the total five stages.

- Stage 1 will be satisfied determinately, with complexity of constant C_1;
- Stage 2, including step 1 to 3, has 5 conditions to be randomly satisfied, with complexity less than 2^5 MD5 compressions;

– Stage 3, including step a to d, has 12 conditions to be randomly satisfied, with complexity less than 2^{12} MD5 compressions;
– Stage 4 will be satisfied determinately, with complexity of constant C_2;
– Stage 5, including step x to y, has 20 conditions to be randomly satisfied, with complexity about 2^{18} MD5 compressions.

Since each stage is independent on each other, the complexity of searching the first block C_{b1} is calculated by addition, instead of a multiplication of $2^{5+12+18}$. The computation is shown as follows.

$$C_{b1} = C_1 + 2^5 + 2^{12} + C_2 + 2^{18} \approx 2^{18} \qquad \text{MD5 compressions} \tag{20}$$

The complexity of searching the second block C_{b2} can be analyzed similarly, which is about 2^{18} MD5 compressions. Therefore, the total complexity of the collision attack C_{coll} with 2-block ($\Delta M_0 = (\Delta m_8 = 2^{31})$, $\Delta M_1 = 0$) is computed as follows.

$$C_{coll} = C_{b1} + C_{b2} = 2^{18} + 2^{18} \approx 2^{19} \qquad \text{MD5 compressions} \tag{21}$$

The other six differential paths of the second block have input difference $\Delta M_1 = (\Delta m_8 = 2^{31})$, and are similar to the path of the first block and have the same number of strong conditions. Hence, the other paths' complexity are all 2^{18} MD5 compressions.

Therefore, the average complexity of the collision attack with 2-block input differences ($\Delta M_0 = (\Delta m_8 = 2^{31})$, $\Delta M_1 = 0$ or $\Delta M_1 = (\Delta m_8 = 2^{31})$) is about 2^{19} MD5 compressions.

Further Optimization In fact, if we apply q_{14} tunnel and group satisfaction scheme in q_{26}, the complexity of MD5 collision attack can be reduced to 2^{18} MD5 compressions, for the strong condition of q_{27} can also be determinately satisfied. A collision pair is shown in Table 2

Table 2: A collision pair with $\Delta M_0 = (\Delta m_8 = 2^{31})$ and $\Delta M_1 = 0$

M_0	0x6f5405b5 0xb891efe 0xae153522 0x3dd541ab 0x77cfac08 0xb4ae7077 0xb14ec779 0xa7ccf30 0xf1c56954 0x70dc3345 0x5eda46a1 0xc9fc1730 0x948b9be 0x2ef76cad 0x86149360 0x3bcecd25
M_0'	0x6f5405b5 0xb891efe 0xae153522 0x3dd541ab 0x77cf ac08 0xb4ae7077 0xb14ec779 0xa7ccf30 0xf1c56954 0x70dc3345 0x5eda46a1 0xc9fc1730 0x948b9be 0x2ef76cad 0x86149360 0x3bcecd25
M_1	0x1dea12a 0x50179204 0x6a2ab7f9 0x80e06efa 0x1da137c9 0x22032f7e 0x3af27c94 0xbfd0dda2 0x54dd5054 0xde27de3 0x328eb6dc 0x1da31980 0xf0a9c456 0x720e6177 0xe5ac6c8f 0x15ab7afc
MD5 value	0x281e1404 0x596131cd 0x9cd2262c 0xa5aa822f

4.2 Single-Block MD5 Collision Attack

Input Difference Choosing The essence of the single block collision is to absorb the state differences in the forth round r_4. We note that four continuous MSB differences should be transfered from r_3 to r_4, by taking the weaknesses of MD5. Therefore, we may choose some input differences to absorb all the MSB differences in r_4 ($\Delta H_1 = 0$), so that a single-block MD5 collision attack can be generated.

We present some input differences that can be used for single-block collision attack, with the number of strong conditions in r_4 listed in Table 3.

We can further deduce the strong conditions after q_{25} of each path to choose optimal one for single-block attack, by the pattern of input differences. In fact, we point out that input difference $\Delta M = (\Delta m_5 = 2^{10}, \Delta m_{10} = 2^{31})$ [14] has the least number of strong conditions, with complexity of 2^{47} MD5 compressions for a single block collision, an example pair was presented in [14]. However, the differential path in [14] is not an optimal one, and the complexity can be reduced to 2^{42} MD5 compressions, the partial differential path is listed in Table 4. The details on how such complexities are calculated are omitted, since they are similar to that of 2-block collision attacks, and only the sufficient conditions after q_{26} are involved.

Table 3: Possible input differences for single-block collision attack

Input difference Δw_i	Input difference Δw_{i+5}	$\#r_4$
m_0	m_5	3
m_7	m_{12}	4
m_{14}	m_3	5
m_5	m_{10}	6
m_{12}	m_1	7
m_3	m_8	8
m_{10}	m_{15}	9
m_1	m_6	10
m_8	m_{13}	11
m_{15}	m_4	12
m_6	m_{11}	13
m_{13}	m_2	14
m_4	m_9	15

Table 4: Partial differential path with optimization based on input difference $\Delta M = (\Delta m_5 = 2^{10}, \Delta m_{10} = 2^{31})$

i	modular differenceΔ	signed difference$\triangle^{\pm}$	#
23	$\Delta q_{23} = 2^3 + 2^{24} + 2^{31}$	$q_{23}[3, 24, 31]$	
24	$\Delta q_{24} = 2^7 + 2^{19} + 2^{27} - 2^{29} + 2^{31}$	$q_{24}[7, 19, 27, -29, 31]$	
25	$\Delta q_{25} = -2^2 - 2^5 - 2^{10} + 2^{22} + 2^{31}$	$q_{25}[-2, -5, -10, 22, 31]$	11
26	$\Delta q_{26} = 2^1 - 2^6 + 2^{31}$	$q_{26}[1, -6, 31]$	11
27	$\Delta q_{27} = 2^{17} + 2^{31}$	$q_{27}[17, 31]$	10
28	$\Delta q_{28} = 2^7 + 2^{15} + 2^{27} + 2^{31}$	$q_{28}[7, 15, 27, 31]$	7
29	$\Delta q_{29} = -2^{10} + 2^{31}$	$q_{29}[-10, 31]$	4
30	$\Delta q_{30} = -2^{15} + 2^{31}$	$q_{30}[-15, 31]$	4
31	$\Delta q_{31} = -2^{15} + 2^{31}$	$q_{31}[-15, 31]$	4
32	$\Delta q_{32} = -2^{27} + 2^{31}$	$q_{32}[27, *31]$	1
33	$\Delta q_{33} = -2^{27} + 2^{31}$	$q_{33}[27, *31]$	2
34	$\Delta q_{34} = 2^{31}$	$q_{34}[*31]$	1
35	$\Delta q_{35} = 0$	q_{35}	2
36	$\Delta q_{36} = 0$	q_{36}	0
37	$\Delta q_{37} = 2^{31}$	$q_{37}[*31]$	1
38	$\Delta q_{38} = 2^{31}$	$q_{38}[*31]$	0
39	$\Delta q_{39} = 2^{31}$	$q_{39}[*31]$	0
40	$\Delta q_{40} = 2^{31}$	$q_{40}[*31]$	0
...	...	...	0
48	$\Delta q_{48} = 2^{31}$	$q_{48}[\hat{}31]$	1
49	$\Delta q_{49} = 2^{31}$	$q_{49}[\hat{}31]$	1
50	$\Delta q_{50} = 2^{31}$	$q_{50}[\hat{}31]$	1
51	$\Delta q_{51} = 2^{31}$	$q_{51}[*31]$	1
52	$\Delta q_{52} = 0$	q_{52}	1
53	$\Delta q_{53} = 0$	q_{53}	1
54	$\Delta q_{54} = 0$	q_{54}	0
55	$\Delta q_{55} = 0$	q_{55}	0
...	...	...	0

Table 5: Partial differential path of $\Delta M = (\Delta m_5 = 2^{10}, \Delta m_{10} = 2^{31}, \Delta m_{14} = 2^{31})$

i	modular difference Δ	singed difference $\triangle^{\pm}$	#
23	$\Delta q_{23} = 2^4 + 2^{24} + 2^{31}$	$q_{23}[4, 24, 31]$	
24	$\Delta q_{24} = 2^{19} + 2^{27} + 2^{31}$	$q_{24}[19, 27, 31]$	
25	$\Delta q_{25} = -2^2 - 2^5 - 2^{10} + 2^{22} + 2^{31}$	$q_{25}[-2, -5, -10, -22, 31]$	11
26	$\Delta q_{26} = 2^1 - 2^6 + 2^{18} + 2^{31}$	$q_{26}[1, -6, 18, 31]$	8
27	$\Delta q_{27} = 2^{31}$	$q_{27}[31]$	9
28	$\Delta q_{28} = 2^7 + 2^{15} + 2^{31}$	$q_{28}[7, 15, 31]$	8
29	$\Delta q_{29} = -2^{10} + 2^{27} + 2^{31}$	$q_{29}[-10, 27, 31]$	4
30	$\Delta q_{30} = -2^{15} + 2^{31}$	$q_{30}[-15, 31]$	4
31	$\Delta q_{31} = -2^{15} + 2^{31}$	$q_{31}[-15, 31]$	4
32	$\Delta q_{32} = -2^{27} + 2^{31}$	$q_{32}[27, *31]$	1
33	$\Delta q_{33} = -2^{27} + 2^{31}$	$q_{33}[27, *31]$	2
34	$\Delta q_{34} = 2^{31}$	$q_{34}[*31]$	0
35	$\Delta q_{35} = 0$	q_{35}	2
36	$\Delta q_{36} = 0$	q_{36}	0
37	$\Delta q_{37} = 2^{31}$	$q_{37}[*31]$	1
38	$\Delta q_{38} = 2^{31}$	$q_{38}[*31]$	0
39	$\Delta q_{39} = 2^{31}$	$q_{39}[*31]$	0
40	$\Delta q_{40} = 2^{31}$	$q_{40}[*31]$	0
...	...	...	0
48	$\Delta q_{48} = 2^{31}$	$q_{48}[\hat{}31]$	1
49	$\Delta q_{49} = 2^{31}$	$q_{49}[\hat{}31]$	1
50	$\Delta q_{50} = 2^{31}$	$q_{50}[\hat{}31]$	1
51	$\Delta q_{51} = 2^{31}$	$q_{51}[*31]$	1
52	$\Delta q_{52} = 0$	q_{52}	1
53	$\Delta q_{53} = 0$	q_{53}	1
54	$\Delta q_{54} = 0$	q_{54}	0
55	$\Delta q_{55} = 0$	q_{55}	0
...	...	...	0

Moreover, we can insert another word difference $\Delta m_{14} = 2^{31}$ into the input difference $\Delta M = (\Delta m_5 = 2^{10}, \Delta m_{10} = 2^{31})$, which forms $\Delta M = (\Delta m_5 = 2^{10}, \Delta m_{10} = 2^{31}, \Delta m_{14} = 2^{31})$, to get even less number of strong conditions (with complexity about 2^{41} MD5 compressions for a collision pair), whose partial differential path is shown in Table 5. We did not publish this input difference and preserved it until someone could also find it for the challenge.

In [12], Steven presented a collision attack by input difference $\Delta m_8 = 2^{25}, \Delta m_{13} = 2^{31})$ for our challenge, with complexity of 2^{50} MD5 compressions[3] to generate a collision pair. We point out that the complexity can be further reduced to 2^{46} MD5 compressions by inserting a word difference of $\Delta m_7 = 2^{31}$ to form input difference $\Delta M = (\Delta m_7 = 2^{31}, \Delta m_8 = 2^{25}, \Delta m_{13} = 2^{31})$, the details of partial differential path are shown in Table 6.

In fact, we really appreciate Stevens' single block collision attack on MD5. However, it is not a completely new one compared to our first one [14], furthermore, it has a worse complexity. It is really a pity that he had not found the input difference we preserved with optimal complexity ($\Delta M = (\Delta m_5 = 2^{10}, \Delta m_{10} = 2^{31}, \Delta m_{14} = 2^{31})$), nor $\Delta M = (\Delta m_7 = 2^{31}, \Delta m_8 = 2^{25}, \Delta m_{13} = 2^{31})$. In the sense of above mentioned, the result presented by Stevens is a failure to our challenge, to some extent. Considering the hardship of finding a practical single-block collision, we have decided to give him a half of the award ($5000), which was paid in 2012.

Table 6: Partial differential path based on input difference $\Delta M = (\Delta m_7 = 2^{31}, \Delta m_8 = 2^{25}, \Delta m_{13} = 2^{31})$

i	modular differenceΔ	signed difference$\triangle^{\pm}$	#
23	$\Delta q_{23} = 2^{11} + 2^{13} + 2^{31}$	$q_{23}[11, 13, 31]$	
24	$\Delta q_{24} = 2^{5} + 2^{18} + 2^{25} + 2^{31}$	$q_{24}[5, 18, 25, 31]$	
25	$\Delta q_{25} = -2^{0} + 2^{12} + 2^{16} - 2^{21} + 2^{31}$	$q_{25}[-0, 12, 16, -21, 31]$	11
26	$\Delta q_{26} = -2^{6} - 2^{27} + 2^{31}$	$q_{26}[-6, -27, 31]$	11
27	$\Delta q_{27} = -2^{6} - 2^{20} + 2^{25} + 2^{31}$	$q_{27}[-6, -20, 25, 31]$	9
28	$\Delta q_{28} = 2^{4}$	$q_{28}[4]$	8
29	$\Delta q_{29} = -2^{16} + 2^{20} - 2^{25}$	$q_{29}[-16, 20, -25]$	4
30	$\Delta q_{30} = -2^{4} + 2^{20} - 2^{25}$	$q_{30}[-4, 20, -25]$	4
31	$\Delta q_{31} = -2^{4}$	$q_{31}[-4]$	4
32	$\Delta q_{32} = 0$	q_{32}	1
33	$\Delta q_{33} = 2^{20}$	$q_{33}[20]$	2
34	$\Delta q_{34} = 2^{20}$	$q_{34}[20]$	0
35	$\Delta q_{35} = 0$	q_{35}	2
36	$\Delta q_{36} = 0$	q_{36}	0
37	$\Delta q_{37} = 0$	$q_{37}0$	1
38	$\Delta q_{38} = 2^{31}$	$q_{38}[*31]$	0
39	$\Delta q_{39} = 2^{31}$	$q_{39}[*31]$	0
40	$\Delta q_{40} = 2^{31}$	$q_{40}[*31]$	0
41	$\Delta q_{41} = 2^{31}$	$q_{41}[*31]$	0
...	...	...	0
48	$\Delta q_{48} = 2^{31}$	$q_{48}[\hat{}31]$	1
49	$\Delta q_{49} = 2^{31}$	$q_{49}[\hat{}31]$	1
50	$\Delta q_{50} = 2^{31}$	$q_{50}[\hat{}31]$	1
51	$\Delta q_{51} = 2^{31}$	$q_{51}[\hat{}31]$	1
52	$\Delta q_{52} = 2^{31}$	$q_{52}[\hat{}31]$	1
53	$\Delta q_{53} = 2^{31}$	$q_{53}[\hat{}31]$	1
54	$\Delta q_{54} = 2^{31}$	$q_{54}[\hat{}31]$	1
55	$\Delta q_{55} = 2^{31}$	$q_{55}[\hat{}31]$	1
56	$\Delta q_{56} = 2^{31}$	$q_{56}[*31]$	1
57	$\Delta q_{57} = 0$	q_{57}	1
58	$\Delta q_{58} = 0$	q_{58}	1
59	$\Delta q_{59} = 0$	q_{59}	0
60	$\Delta q_{60} = 0$	q_{60}	0
...	...	...	0

[3] The complexity is computed under the condition that the tunnel of q_{14} is used, where the conditions of q_{26} can be satisfied using group satisfying scheme.

5 Conclusion

In this paper, we show how to choose right input differences for MD5 collision attack, and analyze their complexities. We answered Stevens' challenge response for a completely new single block MD5 collision in three ways. Firstly, Stevens' single block MD5 collision is not a completely new one, since it can be simply derived from our original one. Secondly, Stevens' single block MD5 collision is much more inferior to our original one in computational complexity. Thirdly, Stevens had not found the very optimal solution that we preserved and had been wishing that someone could also find it, whose collision complexity is about 2^{41} MD5 compressions. We feel sorry that Stevens had not found even a better solution than our original one, let alone the optimal one that we preserved. However, we really appreciate Stevens' single block collision attack on MD5 for his hard work.

Acknowledgments

Part of this work is supported by MOST of China through the 973 program under contract 2007CB311202, and by National Science Foundation of China through the 61070228 project.

References

1. Damgård, I.: A Design Principle for Hash Functions. In: Brassard, G. (ed.) Advances in Cryptology CRYPTO' 89 Proceedings, Lecture Notes in Computer Science, vol. 435, pp. 416–427. Springer Berlin / Heidelberg (1990)
2. Daum, M.: Cryptanalysis of Hash Functions of the MD4-Family. Ph.D. thesis, Ruhr-Universityät of Bochum (2005)
3. Eastlake, D.E., Jones, P.: US secure hash algorithm 1 (SHA1). RFC 3174, Internet Engineering Task Force (Sep 2001), http://www.rfc-editor.org/rfc/rfc3174.txt
4. Klima, V.: Tunnels in Hash Functions: MD5 Collisions Within a Minute. Cryptology ePrint Archive, Report 2006/105 (2006), http://eprint.iacr.org/
5. Liu, F.: A note on the fastest collision attack on md5, to appear in International Journal of Security and its Applications in spring 2013
6. Merkle, R.: One Way Hash Functions and DES. In: Brassard, G. (ed.) Advances in Cryptology CRYPTO 89 Proceedings, Lecture Notes in Computer Science, vol. 435, pp. 428–446. Springer Berlin / Heidelberg (1990)
7. National Institute of Standards and Technology: FIPS 180, secure hash standard, federal information processing standard (FIPS), publication 180. Available from http://csrc.nist.gov (May 1993)
8. National Institute of Standards and Technology: FIPS 180-2, secure hash standard, federal information processing standard (FIPS), publication 180-2. Tech. rep., DEPARTMENT OF COMMERCE (Aug 2002), http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf
9. Rivest, R.: The MD4 Message-Digest Algorithm. RFC 1320 (apr 1992), http://www.ietf.org/rfc/rfc320.txt
10. Rivest, R.: The MD5 Message-Digest Algorithm. RFC 1321 (apr 1992), http://www.ietf.org/rfc/rfc321.txt
11. Stevens, M.: On Collisions for MD5. Master's thesis, TU Eindhoven, Faculty of Mathematics and Computer Science (2007)
12. Stevens, M.: Single-block collision attack on md5. Cryptology ePrint Archive, Report 2012/40 (2012), http://eprint.iacr.org/
13. Stevens, M., Sotirov, A., Appelbaum, J., Lenstra, A., Molnar, D., Osvik, D., de Weger, B.: Short chosen-prefix collisions for md5 and the creation of a rogue ca certificate. In: Halevi, S. (ed.) Advances in Cryptology - CRYPTO 2009, Lecture Notes in Computer Science, vol. 5677, pp. 55–69. Springer Berlin / Heidelberg (2009)
14. Xie, T., Feng, D.: Construct MD5 Collisions Using Just A Single Block Of Message. Cryptology ePrint Archive, Report 2010/643 (2010), http://eprint.iacr.org/
15. Xie, T., Liu, F., Feng, D.: Differential cryptanalysis on md5 (2013)