

# Attacking Reduced Round SHA-256

Somitra Kumar Sanadhya\* and Palash Sarkar

Applied Statistics Unit,  
Indian Statistical Institute,  
203, B.T. Road, Kolkata,  
India 700108.  
somitra\_r@isical.ac.in, palash@isical.ac.in

**Abstract.** The SHA-256 hash function has started getting attention recently by the cryptanalysis community due to the various weaknesses found in its predecessors such as MD4, MD5, SHA-0 and SHA-1. We make two contributions in this work. First we describe message modification techniques and use them to obtain an algorithm to generate message pairs which collide for the actual SHA-256 reduced to 18 steps. Our second contribution is to present differential paths for 19, 20, 21, 22 and 23 steps of SHA-256. We construct parity check equations in a novel way to find these characteristics. Further, the 19-step differential path presented here is constructed by using only 15 local collisions, as against the previously known 19-step near collision differential path which consists of interleaving of 23 local collisions. Our 19-step differential path can also be seen as a single local collision at the message word level. We use a linearized local collision in this work. These results do not cause any threat to the security of the SHA-256 hash function.

## 1 Introduction

Cryptanalysis of hash functions has been an area of intense interest to the research community since past decade and a half. Many hash functions were broken in this time, most notable among them are MD4, MD5, SHA-0 and theoretical break of SHA-1. This has directed the attention of the cryptology community to the SHA-2 family of hash functions.

**Known Results for the SHA-2 Family:** Gilbert and Handschuh (GH) [7] were the first to study local collisions in the SHA-2 family. They reported a 9-round local collision and estimated the probability of the differential path to be  $2^{-66}$ . This probability estimate was later improved by [13] and [8]. Sanadhya and Sarkar [19] recently presented 16 new 9-round local collisions for SHA-2 family of hash functions. The message expansion of SHA-256 was studied by Mendel et al. [13], who mentioned an 18-step collision for SHA-256 which was recently corrected in [14]. The work [13] also provided a differential path for 19-step near collision for SHA-256. An earlier work [12] studied a very simplified variant of SHA-256. The encryption mode of SHA-256 is analyzed in [25] and is not relevant to collision search attacks. Recently, at FSE '08, Nikolić and Biryukov [16] reported 21-step collisions for SHA-256 using a nonlinear differential path.

**Our Contributions:** We make two independent contributions in this work :

1. We construct a 18-step collision characteristic using one of the local collisions from [19]. We describe message modification techniques to find messages following this differential characteristic. Using these techniques, we provide an algorithm to generate pairs of messages which collide for 18 step SHA-256 with the standard IV. We show two such pairs of messages.
2. We show multiple differential paths for attacking up to 23-step SHA-256. In obtaining these differential paths, we use coding theoretic methods in a novel way. Using linearized local collisions, there were no colliding differential paths known for SHA-256 beyond 18 rounds. Previously known best differential path was for 19-step SHA-256 which used 23 local collisions and gave rise to a near collision. In contrast, our 19-step characteristic uses only 15 local collisions and is an exact collision path. All the 15 local

---

\* This author is supported by the Ministry of Information Technology, Govt. of India.

collisions start in the same word and therefore this differential path can also be seen as consisting of a single local collision with the starting word difference having a weight of 15 bits. In addition there are no impossible conditions caused by the  $f_{IF}$  and  $f_{MAJ}$  functions for the differential paths reported here. Therefore the search for actual colliding message pairs following these paths is likely to be easier.

We also show that neutral bit technique may not be of much help in finding actual colliding pair of messages while message modification methods seem to hold much more promise.

Note that these results do not cause any threat to the security of the SHA-256 hash function since it has 64 steps per block.

## 2 Notation

In this paper we use the following notation:

- $m_i \in \{0, 1\}^{32}$ ,  $W_i \in \{0, 1\}^{32}$ ,  $W'_i \in \{0, 1\}^{32}$  for any  $i$ .
- The colliding message pair is:  $\{m_0, m_1, m_2, \dots, m_{15}\}$  and  $\{m'_0, m'_1, m'_2, \dots, m'_{15}\}$ .
- The expanded message pair is:  $\{W_0, W_1, W_2, \dots, W_{63}\}$  and  $\{W'_0, W'_1, W'_2, \dots, W'_{63}\}$ .
- $\oplus$ : bitwise XOR.
- $+$ : addition modulo  $2^{32}$ .
- $\Delta W_i = W_i \oplus W'_i$
- $\text{ROTR}^n(x)$ : Right rotation of a 32 bit quantity  $x$  by  $n$  bits.
- $\text{SHR}^n(x)$ : Right shift of a 32 bit quantity  $x$  by  $n$  bits.

## 3 The SHA-256 Hash Function

The newest members of SHA family of hash functions were standardized by US NIST in 2002 [20]. There are 2 differently designed functions in this standard: the SHA-256 and SHA-512. In addition, the standard also specifies a truncated version of SHA-512, namely the SHA-384. The number in the name of the hash function refers to the length of message digest produced by that function. In this work we are interested in reduced round collision attacks against SHA-256. Next we briefly describe SHA-256. For details refer to [20].

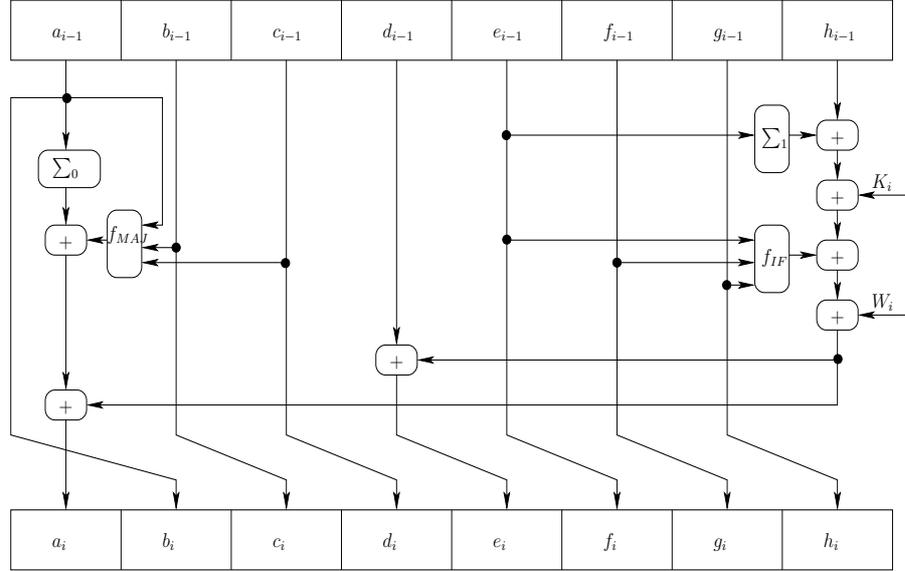
The round function of SHA-256 hash function is shown in Figure 1. Eight registers are used in the evaluation of SHA-256. The initial value in the registers is specified by an 8x32 bit IV. In Step  $i$ , the 8 registers are updated from  $(a_{i-1}, b_{i-1}, c_{i-1}, d_{i-1}, e_{i-1}, f_{i-1}, g_{i-1}, h_{i-1})$  to  $(a_i, b_i, c_i, d_i, e_i, f_i, g_i, h_i)$  according to the following equations:

$$\left. \begin{aligned} a_i &= \Sigma_0(a_{i-1}) + f_{MAJ}(a_{i-1}, b_{i-1}, c_{i-1}) + \Sigma_1(e_{i-1}) \\ &\quad + f_{IF}(e_{i-1}, f_{i-1}, g_{i-1}) + h_{i-1} + K_i + W_i \\ b_i &= a_{i-1} \\ c_i &= b_{i-1} \\ d_i &= c_{i-1} \\ e_i &= d_{i-1} + \Sigma_1(e_{i-1}) + f_{IF}(e_{i-1}, f_{i-1}, g_{i-1}) \\ &\quad + h_{i-1} + K_i + W_i \\ f_i &= e_{i-1} \\ g_i &= f_{i-1} \\ h_i &= g_{i-1} \end{aligned} \right\} \quad (1)$$

The  $f_{IF}$  and the  $f_{MAJ}$  are three variable boolean functions defined as:

$$\begin{aligned} f_{IF}(x, y, z) &= (x \wedge y) \oplus (\neg x \wedge z) \\ f_{MAJ}(x, y, z) &= (x \wedge y) \oplus (y \wedge z) \oplus (z \wedge x) \end{aligned}$$

**Fig. 1.** Round function of SHA-256 hash function



The functions  $\Sigma_0$  and  $\Sigma_1$  are defined as:

$$\begin{aligned}\Sigma_0(x) &= ROTR^2(x) \oplus ROTR^{13}(x) \oplus ROTR^{22}(x) \\ \Sigma_1(x) &= ROTR^6(x) \oplus ROTR^{11}(x) \oplus ROTR^{25}(x)\end{aligned}$$

Round  $i$  uses a 32 bit word  $W_i$  which is derived from the message and a constant word  $K_i$ . There are 64 rounds in all. The hash function operates on a 512 bit message specified as 16 words of 32 bits. Given the message words  $m_0, m_1, \dots, m_{15}$ , the  $W_i$ 's are computed using the equation:

$$W_i = \begin{cases} m_i & \text{for } 0 \leq i \leq 15 \\ \sigma_1(m_{i-2}) + m_{i-7} + \sigma_0(m_{i-15}) + m_{i-16} & \text{for } 16 \leq i \leq 63 \end{cases} \quad (2)$$

The functions  $\sigma_0$  and  $\sigma_1$  are defined as:

$$\begin{aligned}\sigma_0(x) &= ROTR^7(x) \oplus ROTR^{18}(x) \oplus SHR^3(x) \\ \sigma_1(x) &= ROTR^{17}(x) \oplus ROTR^{19}(x) \oplus SHR^{10}(x)\end{aligned}$$

The IV =  $(a_{-1}, b_{-1}, c_{-1}, d_{-1}, e_{-1}, f_{-1}, g_{-1}, h_{-1})$  is defined as (6a09e667, bb67ae85, 3c6ef372, a54ff53a, 510e527f, 9b05688c, 1f83d9ab, 5be0cd19). All additions in Equations 1 and 2 are modulo  $2^{32}$ .

The output hash value of a one block (512 bit) message is obtained by chaining the IV with the register values at the end of the final round as per the Merkle-Damgård construction [6, 15]. A similar strategy is used for multi-block messages, where the IV for next block is taken as the hash output of the previous block.

## 4 Collision Attacks Against Hash Functions

The aim of a hash function attack is to produce two different messages both of which map to the same hash output. This is done by employing differential attack against the hash function in question. First a suitable difference of messages is found such that a pair of messages having that difference is likely to collide to the same hash value with high probability. For example, if a given message differential  $\{\Delta W_0, \Delta W_1, \dots, \Delta W_{15}\}$  is likely to generate colliding pairs with probability  $\frac{1}{2^8}$  then one needs to try roughly  $2^8$  different pairs  $\{W_0, W_1, \dots, W_{15}\}$  and  $\{W'_0, W'_1, \dots, W'_{15}\}$  having the given difference to get a colliding pair of messages.

However, the probability of the specified differential to generate a collision is likely to be very low for most of the practical hash functions. Hence some sophisticated methods are used to search for the right (colliding) pair, rather than generating them at random. Message modification techniques [24, 22] and neutral bit technique [1] are the two widely used methods to find colliding message pairs.

We next describe some terms and techniques used in the differential attack on hash functions.

#### 4.1 Linearized Local Collision

To analyze a hash function, it is advantageous to consider its linearized version. That is, all the non-linear components in the hash function design are first replaced by their linear approximations. Since the resulting design is linear, we have  $h(x) \oplus h(x \oplus \Delta x) = h(\Delta x)$  where  $h$  is one step of the hash function evaluation. Therefore we can look at the behavior of the differential of the messages  $\Delta x$  on the linearized hash function rather than considering a pair of messages  $x$  and  $x \oplus \Delta x$  operated on by a step of linearized hash function, and then taking their differential.

The simplified version of the hash function is then analyzed for a small number of steps. The technique is to introduce a small perturbation in the message at the first step and then correct it by suitable differences in the messages in next few steps. After introducing some message differences for few rounds it is possible to completely cancel the effect of the original perturbation. A sequence of message word differences which cancel their own effect locally are said to construct a “*local collision*”. All the message words are considered to be unrestricted during the search for local collisions. Finally many local collisions are interleaved to obtain (reduced round) colliding pair of messages.

Attacks on the SHA-0 and SHA-1 hash functions use the 5-step local collision obtained by Chabaud and Joux [4]. For SHA-256, a 9-step local collision by Gilbert and Handschuh [7] and sixteen 9-step local collisions by Sanadhya and Sarkar [19] are known. We discuss the case of SHA-256 local collisions in more detail.

#### 4.2 Linearized Local Collisions in SHA-256

Let the first step in SHA-2 be denoted by Step 0. If a 9-step local collision is started at step  $i$ , it defines the 9 word differences  $W_j \oplus W'_j$  for  $i \leq j \leq i + 8$ . We use two types of local collisions in the present work. The first is due to Gilbert and Handschuh [7] and the second is one of the 16 local collisions presented in [19]. From among the 16, we choose the 5<sup>th</sup> local collision because of the following two reasons :

1. It is one of the 4 which are suitable for getting 18-step collision, as explained later (the others being 7<sup>th</sup>, 14<sup>th</sup> and 16<sup>th</sup>).
2. It has the highest probability among these 4.

We call the two local collisions the GH local collision and the SS<sub>5</sub> local collision respectively. The other three local collisions from [19] are denoted by SS<sub>7</sub>, SS<sub>14</sub> and SS<sub>16</sub>.

The following approximations are used in these local collisions :

1. Operator  $+$  is approximated by  $\oplus$ .
2. In GH,  $f_{IF}$  and  $f_{MAJ}$  are approximated by zero function. This causes certain impossible conditions while searching for the message pair following this differential path, as has been observed in [13].
3. In SS<sub>5</sub>,  $f_{IF}$  and  $f_{MAJ}$  are approximated by their middle arguments. These linear approximations avoid two types of impossible conditions encountered when using GH local collision.

See [19] for details on other local collisions.

All the local collisions mentioned above are:

- GH :  $\{x, \Sigma_0(x) \oplus \Sigma_1(x), \Sigma_0(\Sigma_1(x)), 0, x, \Sigma_0(x) \oplus \Sigma_1(x), 0, 0, x\}$
- SS<sub>5</sub> :  $\{x, \Sigma_0(x) \oplus \Sigma_1(x), \Sigma_0(\Sigma_1(x)), \Sigma_0(x) \oplus \Sigma_1(x), 0, \Sigma_0(x) \oplus \Sigma_1(x), 0, 0, x\}$

- $SS_7 : \{x, x \oplus \Sigma_0(x) \oplus \Sigma_1(x), \Sigma_0(x) \oplus \Sigma_0(\Sigma_1(x)), x \oplus \Sigma_0(x) \oplus \Sigma_1(x), 0, x \oplus \Sigma_0(x) \oplus \Sigma_1(x), 0, 0, x\}$
- $SS_{14} : \{x, x \oplus \Sigma_0(x) \oplus \Sigma_1(x), x \oplus \Sigma_1(x) \oplus \Sigma_0(\Sigma_1(x)), \Sigma_1(x), \Sigma_0(x) \oplus \Sigma_1(x), \Sigma_0(x) \oplus \Sigma_1(x), 0, 0, x\}$
- $SS_{16} : \{x, \Sigma_0(x) \oplus \Sigma_1(x), \Sigma_0(x) \oplus \Sigma_1(x) \oplus \Sigma_0(\Sigma_1(x)), x \oplus \Sigma_1(x), x \oplus \Sigma_0(x) \oplus \Sigma_1(x), x \oplus \Sigma_0(x) \oplus \Sigma_1(x), 0, 0, x\}$

Note that in all the above local collisions,  $\Sigma_0$  and  $\Sigma_1$  are used as operators on 32 bit quantities, and  $x$  is any 32 bit message word difference. Once a starting message difference  $x$  is chosen, next 8 words must have the difference in accordance with the local collision.

### 4.3 Differential Path

The introduction of the message differentials causes the internal registers of hash function to differ too. The propagation of the differences in the registers creates a “*differential path*”. For SHA-256, the differential path for  $k$  steps can be completely specified by differences in the 8 registers for these  $k$  steps along with the  $k$  message differences corresponding to the same steps. The term “*linear characteristic*” is also widely used in literature to refer to the differential path. We also use it interchangeably with differential path.

If the differential path includes steps in which the message expansion is involved, then the message words for some steps are computed on the basis of previous message words. In such a case, the message expansion is also linearized. If the register differences are all zero at the last step of the differential path then such a path is said to be a “*colliding differential path*”. The differential of the messages for a colliding differential path are called as “*colliding message differential*”.

The differential path holds for the linearized version of the hash function with probability 1, but with much less probability for the actual hash function.

## 5 Attacking 18 rounds of SHA-256

It is possible to get up to 18 step reduced round collisions for SHA-256 using a single local collision. Such an idea has already been used in [13] and mentioned in [19]. We describe this for clarity of exposition.

First of all, note that any local collision under consideration spans 9 steps and the message expansion of SHA-256 does not play any role in the first 16 steps. Therefore if a local collision spans from Step  $i$  to Step  $(i + 8)$ , and if we take  $\Delta W_0 = \Delta W_1 = \dots = \Delta W_{i-1} = \Delta W_{i+9} = \Delta W_{i+10} = \dots = \Delta W_{15} = 0$ , we get a differential path for 16-step collision for SHA-256.

The issue of message expansion is not considered in obtaining the 16 step colliding differential path described above. Next we tackle two steps of the message expansion.

Message expansion rule for  $W_{16}$  and  $W_{17}$  are given by :

$$W_{16} = \sigma_1(W_{14}) + W_9 + \sigma_0(W_1) + W_0 \quad (3)$$

$$W_{17} = \sigma_1(W_{15}) + W_{10} + \sigma_0(W_2) + W_1 \quad (4)$$

Let a local collision  $\mathcal{L}$  start at Step 3 and hence end at Step 11. This local collision defines the 9 word differences  $\Delta W_3, \Delta W_4, \dots, \Delta W_{11}$ . The first step of the local collision corresponds to  $\Delta W_3$  and the 9<sup>th</sup> step corresponds to  $\Delta W_{11}$ . Taking the differentials of all the message words outside the span of the local collision to be zero, the differential path for  $\mathcal{L}$  will have  $\Delta W_0 = \Delta W_1 = \Delta W_2 = \Delta W_{12} = \Delta W_{13} = \Delta W_{14} = \Delta W_{15} = 0$ .

Note that  $\Delta W_i = 0$  means that  $W_i = W'_i$ . Since  $\Delta W_0 = \Delta W_1 = \Delta W_{14} = 0$  for  $\mathcal{L}$ , from Equation 3,  $W_{16}$  and  $W'_{16}$  may be different only due to the differences in  $W_9$  and  $W'_9$ .

$\Delta W_9$  corresponds to the 7<sup>th</sup> step word difference for  $\mathcal{L}$ . If  $\mathcal{L}$  is chosen such that it's 7<sup>th</sup> step word difference is zero, then  $W_9 = W'_9$ . Therefore even after the message expansion recursion is used, we will have  $W_{16} = W'_{16}$ . This results in a 17-step differential path for SHA-256.

Similarly, if the 8<sup>th</sup> message word difference for  $\mathcal{L}$  is zero, then by Equation 4,  $W_{17} = W'_{17}$ . This results in a 18-step differential path for SHA-256.

Both the 17 and the 18 Step paths discussed above use just one local collision. To increase the probability of this differential path for the case of real SHA-256, we can take starting messages differing in only 1 bit.

All the local collisions listed in the previous section have the 7<sup>th</sup> and the 8<sup>th</sup> message word differences zero. Therefore any one of them can be used to obtain the 18 step colliding differential path for SHA-256. We list one of these differential paths in Table 1. This 18-step colliding path is also a 17-step colliding path for SHA-256.

Further, it can be seen that it is not possible to obtain a differential path for 19 or more steps with a single local collision where the weight of the perturbation in first word is just 1-bit. This impossibility arises due to the message expansion of SHA-256, and because there are no local collisions in which 3 consecutive word differences are zero. We discuss the case of more than 18 steps in later sections.

**Table 1.** 18 step linear characteristic for SHA-256. Only 1  $SS_5$  local collision is used to build this path.

Step $i$	$\Delta W_i$	$\Delta a_i$	$\Delta b_i$	$\Delta c_i$	$\Delta d_i$	$\Delta e_i$	$\Delta f_i$	$\Delta g_i$	$\Delta h_i$
0-2	0	0	0	0	0	0	0	0	0
3	80000000	80000000	0	0	0	80000000	0	0	0
4	22140240	0	80000000	0	0	20040200	80000000	0	0
5	42851098	0	0	80000000	0	80000000	20040200	80000000	0
6	22140240	0	0	0	80000000	0	80000000	20040200	80000000
7	0	0	0	0	0	80000000	0	80000000	20040200
8	22140240	0	0	0	0	0	80000000	0	80000000
9	0	0	0	0	0	0	0	80000000	0
10	0	0	0	0	0	0	0	0	80000000
11	80000000	0	0	0	0	0	0	0	0
12-17	0	0	0	0	0	0	0	0	0

## 6 Message Modification Techniques for SHA-256

We have used XOR differences for registers and message words in the differential path for reduced round SHA-256. The differential path in Table 1 is obtained by using linearized SHA-256. However our aim is to obtain a pair of messages which follows this differential path for real SHA-256. The probability for this to happen for random messages is  $2^{-49}$  for 18-step SHA-256. If the message-pair satisfies certain conditions then the probability of the differential path can be increased significantly. We list conditions on the registers and the message words which help in finding messages following the 18 step differential path shown in Table 1 when actual SHA-256 is used. These conditions try to ensure that the functions  $f_{IF}$  and  $f_{MAJ}$  both behave like their middle arguments, and that  $+$  behaves like  $\oplus$ . These conditions are shown in Table 2. Sufficient conditions for 9 step SHA-256 collision have also been given in [9], Table 3. We next highlight the advantages of our conditions with those in [9].

1. The conditions in [9] are for only 9-step collision in SHA-256. Our conditions are for 18-step collision in SHA-256.
2. The GH local collision is used in [9] whereas we use  $SS_5$  local collision. Further, no explanation is provided in [9] on how these conditions are derived whereas we provide complete details about our conditions. It is now possible to use the method described in this work to derive conditions for 18-step SHA-256 collision using any other local collision.
3. In [9] the conditions are claimed to be “sufficient” but it is not clear if satisfying them will immediately lead to a collision. The conditions that we identify are not claimed to be sufficient. We only note that satisfying them will increase the probability of finding colliding message pairs.

**Table 2.** Conditions for the 18 step differential path in Table 1.  $x^i$  denotes  $i^{th}$  bit of a 32 bit quantity  $x$ .  $\bar{x}$  denotes the bitwise negation of  $x$  which can be a 32 bit or a 1 bit quantity. Operator  $+$  is addition modulo  $2^{32}$  and operator  $*$  is multiplication of 2 single bits. Both these operators are used in steps 6 and 8.

Step $k$	Due to $f_{MAJ}$	Pr.	Due to $f_{IF}$	Pr.	Due to $a_k$	Pr.	Due to $e_k$	Pr.	Step Pr.
0-3	-	-	-	-	-	-	-	-	1
4	$b_3^{31} = c_3^{31}$	$\frac{1}{2}$	$f_3^{31} = g_3^{31}$	$\frac{1}{2}$	$W_4^i = (\Sigma_0(a_3))^i; i = 9, 18, 29$ $W_4^i = (\Sigma_1(e_3))^i; i = 6, 20, 25$	$\frac{1}{2^6}$	bit differences $\Delta W_4^i; i = 9, 18, 29$ propagate into $e_4$	$\frac{1}{2^3}$	$\frac{1}{2^{11}}$
5	$a_4^{31} = c_4^{31}$	$\frac{1}{2}$	$e_4^{31} = 1,$ $e_3^i = f_3^i;$ $i = 9, 18, 29$	$\frac{1}{2^4}$	$W_5^i = (\Sigma_1(e_4))^i;$ $i = 3, 4, 7, 12, 16,$ $18, 23, 25, 30$	$\frac{1}{2^9}$	-	-	$\frac{1}{2^{14}}$
6	$a_5^{31} = b_5^{31}$	$\frac{1}{2}$	$e_5^{31} = 1;$ $i = 9, 18, 29$ $e_4^{31} = \bar{e}_5^{31} * e_3^{31}$	$\frac{1}{2^4}$	$W_5^i = (\Sigma_1(e_5) + f_5)^i;$ $i = 6, 9, 18, 20, 25, 29$	$\frac{1}{2^6}$	-	-	$\frac{1}{2^{11}}$
7	-	-	$e_6^i = 1;$ $i = 9, 18, 29, 31$	$\frac{1}{2^4}$	-	-	-	-	$\frac{1}{2^4}$
8	-	-	$e_6^{31} = \bar{e}_7^{31} * e_5^{31}$	$\frac{1}{2}$	$W_8^i = (\Sigma_1(e_7) + h_7)^i;$ $i = 6, 9, 18, 20, 25, 29$	$\frac{1}{2^6}$	-	-	$\frac{1}{2^7}$
9	-	-	$e_8^{31} = 1$	$\frac{1}{2}$	-	-	-	-	$\frac{1}{2}$
10	-	-	$e_9^{31} = 1$	$\frac{1}{2}$	-	-	-	-	$\frac{1}{2}$
11-17	-	-	-	-	-	-	-	-	1
								Prob.	$\frac{1}{2^{49}}$

## 6.1 Explanation of Conditions in Table 2

$\Delta W_k = 0$  for steps  $k=0, 1$  and  $2$  and hence there are no restrictions due to these steps. In Step 3, although  $\Delta W_3 \neq 0$ , the difference is only in the most significant bit. The  $+$  and  $\oplus$  behave the same with probability 1 for a difference in MSB, so even Step 3 does not impose any restrictions. Hence conditions are needed to tackle the proper differential behavior for the message pair only from Step 4 onwards.

**Conditions Due to  $f_{MAJ}$  and  $f_{IF}$  :** In Step 4,  $f_{MAJ}$  has inputs  $a_3, b_3$  and  $c_3$  with  $\Delta a_3 = 0x80000000$ . In  $SS_5$  local collision  $f_{MAJ}$  is approximated by it's middle argument, which will happen if  $b_3^{31} = c_3^{31}$ . Similarly the  $f_{IF}$  function having arguments  $e_3, f_3$  and  $g_3$  will behave like it's middle argument if  $f_3^{31} = g_3^{31}$ .

**Conditions Due to Register  $a_4$  :** Once the two boolean functions are approximated by their middle arguments, register  $a_4$  is evaluated for both the messages as follows :

$$a_4 = \Sigma_0(a_3) + b_3 + \Sigma_1(e_3) + f_3 + h_3 + K_4 + W_4 \quad \text{and}$$

$$a'_4 = \Sigma_0(a'_3) + b'_3 + \Sigma_1(e'_3) + f'_3 + h'_3 + K_4 + W'_4$$

Registers  $a_3$  and  $a'_3$  (resp.  $e_3$  and  $e'_3$ ) differ in their MSB, and the operator  $\Sigma_0$  (resp.  $\Sigma_1$ ) expands this difference to 3 bit positions 6, 20 and 25 (resp. 9, 18 and 29). The word difference  $\Delta W_4$  at this step has been chosen to differ in these 6 bit positions (namely 6, 20, 25, 9, 18 and 29) with the aim of cancelling these differences.

The cancellation will happen as desired if :

1. The difference of words  $W_4$  and  $W'_4$  is opposite to the difference in words  $\Sigma_0(a_3)$  and  $\Sigma_0(a'_3)$  on bit positions 9, 18 and 29. For example, if  $(\Sigma_0(a_3))^i = 1$  and  $(\Sigma_0(a'_3))^i = 0$ , then we would like  $(W_4)^i = 0$  and  $(W'_4)^i = 1$  so that  $W_4 + \Sigma_0(a_3)$  and  $W'_4 + \Sigma_0(a'_3)$  are equal at the  $i^{th}$  bit position;  $i = 9, 18$  and  $29$ .
2. Similarly,  $(W_4)^i$  and  $(W'_4)^i$  have difference opposite to the difference in  $(\Sigma_1(e_3))^i$  and  $(\Sigma_1(e'_3))^i$  at bit positions  $i = 6, 20$  and  $25$ .

All the 6 bit differences will be cancelled if the conditions shown in Table 2, Step 4, column  $a_k$  are met. Note that this is not a necessary way of cancelling the differences, other possibilities exist when the sum of the terms in  $a_4$  and  $a'_4$  may behave as desired. In particular, we do not use bit carries in addition modulo  $2^{32}$  to cancel these type of differences like Wang et. al do for SHA-1 [23]. We use XOR differences only, unlike [23] where modular differences are used.

**Conditions due to register  $e_4$  :** Having cancelled the 6 bit differences to obtain  $\Delta(a_4) = 0$ , it can be seen that 3 bits from  $\Delta(W_4)$  will certainly propagate into  $\Delta(e_4)$  because there is no  $\Sigma_0$  term in calculating  $e_4$  and  $e'_4$ . If the differential path is to be followed, then these 3 differing bits in  $W_4$  and  $W'_4$  should not carry forward to other positions. Carry propagation to other bits will cause problems in adjusting the register differences in next steps since any single bit difference in  $a$  or  $e$  register is expanded into 3 bit differences by the operators  $\Sigma_0$  and  $\Sigma_1$ . We have chosen the word differences in next steps considering these positions by following the linear (XOR) characteristics. It is possible to allow some bit carries here but it seems that it will only reduce the probability of the differential path.

To complete the analysis of step 4, we finally look at the difference  $\Delta(e_4)$ . The registers  $e_4$  and  $e'_4$  are computed as follows:

$$\begin{aligned} e_4 &= d_3 + \Sigma_1(e_3) + f_{IF}(e_3, f_3, g_3) + h_3 + K_4 + W_4, \\ \text{and } e'_4 &= d'_3 + \Sigma_1(e'_3) + f_{IF}(e'_3, f'_3, g'_3) + h'_3 + K_4 + W'_4. \end{aligned}$$

In these two computations, bits 6, 20 and 25 corresponding to  $\Sigma_1$  rotations of the differing bit 31 in  $e_3$  have already been taken care of while considering  $a_4$ . Bit numbers 9, 18 and 29 are the places where  $W_4$  and  $W'_4$  differ and these differences are required to be propagated to  $\Delta e_4$ . Since  $d_3 = d'_3$ ,  $h_3 = h'_3$  and  $f_{IF}(e_3, f_3, g_3) = f_{IF}(e'_3, f'_3, g'_3)$ ;

$$\begin{aligned} \text{if we write } \quad rest &= \Sigma_1(e_3) + f_{IF}(e_3, f_3, g_3) + h_4 + K_4, \\ \text{then} \quad \quad \quad e_4 &= rest + W_4, \\ \text{and} \quad \quad \quad e'_4 &= rest + W'_4. \end{aligned}$$

If the  $i^{th}$  bit of  $rest$  is 0 and there is no carry into the  $i^{th}$  bit while addition with  $W_4$  takes place, then the XOR difference  $W_4 \oplus W'_4$  will propagate into  $e_4 \oplus e'_4$  as desired. Alternately, if the  $i^{th}$  bit of  $rest$  is 1 and there is a carry into the  $i^{th}$  bit while addition with  $W_4$  takes place, then too the XOR difference  $W_4 \oplus W'_4$  will propagate into  $e_4 \oplus e'_4$ .

Thus either we would like no carry propagation in  $e_4$  and  $e'_4$  at bits 6, 20 and 25 if  $rest$  is 0 at these bit positions or we would like carry propagation in both these registers if  $rest$  is 1 at these bits. We do not have a deterministic way to ensure this since we do not have complete freedom to choose the registers and the message words as desired at this stage. However, the probability of the carries to happen as desired can be increased if we set other free bits of  $W_4$  and  $W'_4$  according to the following conditions :

1. if  $rest^9$  is 0 then  $W_4^7 = W_4^8 = 0$ .
2. if  $rest^9$  is 1 then  $W_4^7 = W_4^8 = 1$ .
3. if  $rest^{18}$  is 0 then  $W_4^{10} = W_4^{11} = \dots = W_4^{17} = 0$ .
4. if  $rest^{18}$  is 1 then  $W_4^{10} = W_4^{11} = \dots = W_4^{17} = 1$ .
5. if  $rest^{29}$  is 0 then  $W_4^{26} = W_4^{27} = W_4^{28} = 0$ .
6. if  $rest^{29}$  is 1 then  $W_4^{26} = W_4^{27} = W_4^{28} = 1$ .

In setting these conditions, we have used the bits between 6, 9, 20 and 9, 18 and 29 which are not restricted.

Similarly we have set conditions for other steps so that the messages follow the differential path as desired.

## 6.2 Method to Satisfy Conditions in Table 2

First 4 words in the differential path are free and hence we choose them randomly. Thereafter, many conditions in Table 2 are easy to fulfill as they depend only on word  $W_k$  in step  $k$ . Some of the conditions on registers can be tackled by suitably choosing the word  $W_k$  at that step which we can choose as desired. However, there may be instances when a previously selected message word causes impossible condition at a later step. As an example, we may not get the bit carry conditions for register  $e_4$  as described previously. Also we wish to have  $e_6^{31}$  following a particular pattern at step 8 whereas this bit has been set at step 6 itself. In such contradicting cases, we choose another message word randomly at the previous step where the condition was breaking down. Then we apply message modification techniques from that step onwards and continue the search process for further steps. We search incrementally proceeding further only when all the conditions at a step are fulfilled and the differential path is as desired. The differential path in Table 1 holds with probability  $2^{-49}$ , but with the procedure described above, we are able to get a much higher probability. In fact, Steps 0 to 7 become very easy to fulfill with the message modification and we are able to satisfy all the conditions till Step 7 in about a minute on an ordinary PC. The only difficult conditions are those imposed due to  $a_8$ . We could find a colliding message pair following exact differential characteristic in time varying from about 40 minutes to a couple of hours on an ordinary PC. Repeatedly running the program we could generate many such pairs. We show two such colliding pairs of messages.

## 6.3 Colliding Message Pairs for 18-Step SHA-256

Tables 3 and 4 show the message pairs found using the techniques described previously. All 18 words of the messages are given in the tables. First 16 words can be used to compute the last two words using the message expansion of SHA-256. Similar method can be used for finding 9-round pseudo collisions for SHA-256 as well. Since we can already find message pairs colliding for 18-step SHA-256 with the standard IV, the only utility for such an exercise would be to see how easy it becomes to find these pseudo collisions due to the benefits of relaxing the IV conditions. However, we found that the time required to find a 9-round pseudo collision is only marginally less than the time required to find an 18-step collision. We give an example of such a pseudo collision in Section A.

**Table 3.** Colliding message pair for 18 step SHA-256 with standard IV. These two messages follow the differential path given in Table 1.

M <sub>1</sub>	0-7	ccea5c17	53ad1a2d	141db23c	b6acfaa8	5ee7fe4d	53c5b764	2bf20d44	87d63bf6
	8-15	63a07869	f305fdea	26ee271f	b973b91c	d0f87828	b724a487	a295fa2a	0a67c97a
M <sub>2</sub>	0-7	ccea5c17	53ad1a2d	141db23c	36acfaa8	7cf3fc0d	1140a7fc	09e60f04	87d63bf6
	8-15	41b47a29	f305fdea	26ee271f	3973b91c	d0f87828	b724a487	a295fa2a	0a67c97a

**Table 4.** Another colliding message pair for 18 step SHA-256 with standard IV. These two messages also follow the differential path given in Table 1.

M <sub>1</sub>	0-7	ed919421	aa75e4fe	8548d0e0	9c1888f7	1da3fc3d	a11f7a02	bb463b64	e9b28365
	8-15	323ecf28	8097e497	4343b78b	dc484e91	bf588b4b	8401140a	42499da1	f88a3e2e
M <sub>2</sub>	0-7	ed919421	aa75e4fe	8548d0e0	1c1888f7	3fb7fe7d	e39a6a9a	99523924	e9b28365
	8-15	102acd68	8097e497	4343b78b	5c484e91	bf588b4b	8401140a	42499da1	f88a3e2e

It seems possible to use neutral bits to increase the efficiency of the search for finding message pairs following the given differential path. We experimented with this idea and found that the gains are not significant. More details about our experiments with neutral bits are available in Section B.

## 7 Using Coding Theoretic Methods to Find Linear Differential Paths for Reduced Round SHA-256

In [18] and [17] coding theoretic techniques were used to search for differential paths in SHA-1. Extension to SHA-2 was mentioned in [13]. We describe a new way of forming parity check equations and then find low weight codewords for the corresponding generator matrix. Each of these codewords can be used to build a differential characteristic for reduced round SHA-256. This method results in tackling up to 23-step reduced SHA-256.

### 7.1 A New Way of Constructing Parity Check Equations

Tackling message expansion in SHA-2 can be a problem. A non-zero value of  $\Delta W_i$  for  $i \geq 16$  necessitates tackling the recursion for message expansion. So one way to avoid this is to ensure that  $\Delta W_i = 0$  for  $i \geq 16$ . Clearly, this cannot work for full SHA-2. But, for reduced round versions, one can find differential paths using this approach, as we describe below.

The technique described below assumes a local collision  $\mathcal{L}$ . The description is not for any particular local collision. It holds for any local collision. Obtaining a particular local collision requires certain linear approximations of the constituents of the SHA-256 round function. This converts the round function into a linear map based on which we define our linear code. We note that the linear code is not straightforwardly obtained from the linear map.

A message consists of 16 32-bit words for a total of 512 bits. We use the Chabaud-Joux [4] type disturbance vector approach. Let  $DV = \{d_0, d_1, d_2, \dots, d_{255}\}$  be a 256-bit disturbance vector. If  $d_i = 1$  then the two initial messages differ in their  $i^{\text{th}}$  bit, and further message bits differ as per the local collision.

We do not consider a 512-bit DV for the following reason. A local collision defines the differences of 9 words of messages and only the first 16 words of SHA-256 are unrestricted. Thereafter the message words are calculated using the message expansion recurrence. This implies that a local collision can not be started after first 8 steps without affecting the message expansion.

Let us now describe the linear code that we require. This is done in two steps. In the first step, we express  $\Delta W_i$  ( $i \geq 16$ ) in terms of  $d_0, \dots, d_{255}$ . In the second step, we define the parity check equations for the code by setting  $\Delta W_i = 0$  for  $i \geq 16$ . Thus, any DV  $(d_0, \dots, d_{255})$  which satisfies these parity check equations is a codeword. Our task then is to look for a low weight codeword as this gives a differential path with a small number of local collisions.

It is clear that such codes can be formed as long as there are less than 256 parity check equations. If we apply this procedure up to  $N$  rounds (corresponding to step  $N - 1$ ), then we will obtain  $32(N - 16)$  parity check equations. Thus, the maximum  $N$  that we can use with this method is  $N = 23$ . The minimum value of  $N$  is clearly 17. Since we already report 18-round collision, we do not consider  $N = 17$  and 18. Instead we report differential paths from 19 to 23 rounds.

The first task is to express  $\Delta W_i$  ( $i \geq 16$ ) in terms of  $d_0, \dots, d_{255}$ . We describe how this is done. For any local collision  $\mathcal{L}$ , the first word determines the next eight words. Consider the 32-bit vector  $(d_0, 0, \dots, 0)$ , where  $d_0$  is treated as a (binary) variable. Then  $\mathcal{L}$  defines the next 8 32-bit words. At this point, the first 9 words have been defined. The rest 7 are taken to be zero. For  $i \geq 16$ ,  $\Delta W_i$  is now obtained using the message recursion. This expresses all the  $\Delta W_i$ s ( $i \geq 16$ ) as linear function of  $d_0$ . Next consider the 32-bit vector  $(0, d_1, 0, \dots, 0)$ ; use  $\mathcal{L}$  to obtain the next eight words and the message expansion recursion to express  $\Delta W_i$ s ( $i \geq 16$ ) as linear function of  $d_1$ . Now, for the 32-bit vector  $(d_0, d_1, 0, \dots, 0)$ , we can express  $\Delta W_i$ s ( $i \geq 16$ ) as linear function of  $d_0$  and  $d_1$  by XORing the separate linear functions corresponding to  $d_0$  and  $d_1$ . Clearly, the procedure can be extended to the entire DV  $(d_0, \dots, d_{255})$ . The exact details are given in Table 5.

Methods presented in [3], [10] and [21] are used to search for low weight codewords from the check-matrices (and the corresponding generator matrices) obtained using the algorithm in Table 5. Codewords of least weight found and the linear differential path for that codeword are shown in Section 8.

**Table 5.** Algorithm for generating parity check equations for linearized  $N$  step SHA-256.

---

**external**  $\text{LC}(x)$  : accepts a 32 bit input  $x$  and returns 9 words of 32 bits conforming to the local collision chosen.

---

Set  $\Delta W_{final} := (U_0, U_1, \dots, U_{N-1})$   $U_i \in \{0, 1\}^{32}$   
Set  $\delta W_{cur} := (V_0, V_1, \dots, V_8)$   $V_i \in \{0, 1\}^{32}$   
Initialize  $\Delta W_{final}$  and  $\delta W_{cur}$  to all zeros.

For( $i = 0$  to 8){  
  For( $j = 0$  to 31){  
    set  $D := (0, 0, \dots, d_{32i+j}, 0, \dots, 0)$ ;      /\* The  $j^{th}$  bit of  $D$  is given by  $d_{32i+j}$ .  
      Each  $d_n \in \{0, 1\}$  is the component of the disturbance vector and  $D \in \{0, 1\}^{32}$  \*/  
    set  $\delta W_{cur} := \text{LC}(D)$ ;  
    For( $k = i$  to  $i + 8$ ){  
       $\Delta W_{final}[k] = \Delta W_{final}[k] \oplus \delta W_{cur}[k - i]$ ;  
    }  
  }  
}

/\* At this point the  $\Delta W_{final}$  list contains  $W_i \oplus W'_i$  for  $0 \leq i < 16$  \*/

Obtain  $\Delta W_i$  for  $16 \leq i < N$  using linearized message expansion of SHA-256.  
Equate all 32 bits of  $\Delta W_i$  for  $i \geq 16$  to zero to get  $32 * (N - 16)$  parity check equations.

---

## 8 Results and Comparison to Previous Work

Low weight disturbance vectors are searched for reduced round SHA-256 by using the probabilistic methods described in [3], [10] and [21]. The minimum weights of codewords found are listed in Table 6. For 19-step SHA-256 the weight of the codeword found is 15 for both GH and  $\text{SS}_5$  local collision. This means that 15 local collisions are interleaved to obtain the 19-step characteristic. Interestingly, all the 15 local collisions start at the same word for both GH and  $\text{SS}_5$ . Thus the case of 19-step characteristic can be considered as consisting of a single local collision starting at Step 3 where the initial message difference is a word with weight 15 bits. There is no colliding differential path known before this work using the linearized local collision. Using this technique, the best known 19-step differential path is for a near collision consisting of 23 GH local collisions [13]. As has already been noted in [13], the GH local collision causes certain impossible conditions in the search for actual colliding pairs. The use of  $\text{SS}_5$  local collision ensures that we do not face two types of impossible conditions.

For 20 to 23 steps, no differential path using a linearized local collision is known so far. We provide the first differential paths for these cases using the linearization technique. For 23-step SHA-256, the size of the corresponding generator matrix is  $32 \times 256$ , i.e. there are only 32 codewords of length 256. It is possible to do exhaustive search on this size, hence we did not use the probabilistic methods for this case. For the 23-step case, the reported codeword weight is actually the best possible. All these differential paths are reported in Section C.

## Acknowledgements

We would like to thank Christian Rechberger for carefully reading an earlier version of this paper and giving helpful suggestions. He also mentioned that analysis similar to ours appears in [11].

## References

1. Eli Biham and Rafi Chen. Near-Collisions of SHA-0. In Matthew K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004, 24th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*, pages 290–305. Springer, 2004.

**Table 6.** Summary of results. Least weight of the codeword found using different local collisions. For 23 step case, the codeword weight is obtained by exhaustive search. For all other cases, methods described in [3], [10] and [21] are used.

Step $i$	Size of Check matrix	using GH	using $SS_5$
18	-	1	1
19	$96 \times 256$	15	15
20	$128 \times 256$	33	31
21	$160 \times 256$	45	45
22	$192 \times 256$	59	60
23	$224 \times 256$	79	75

2. Gilles Brassard, editor. *Advances in Cryptology - CRYPTO 1989, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*. Springer, 1990.
3. Anne Canteaut and Florent Chabaud. A New Algorithm for Finding Minimum-Weight Words in a Linear Code: Application to McEliece's Cryptosystem and to Narrow-Sense BCH Codes of Length 511. *IEEE Transactions on Information Theory*, 44(1):367–378, 1998.
4. Florent Chabaud and Antoine Joux. Differential Collisions in SHA-0. In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO 1998, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, volume 1462 of *Lecture Notes in Computer Science*, pages 56–71. Springer, 1998.
5. Ronald Cramer, editor. *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*. Springer, 2005.
6. Ivan Damgård. A Design Principle for Hash Functions. In Brassard [2], pages 416–427.
7. Henri Gilbert and Helena Handschuh. Security Analysis of SHA-256 and Sisters. In Mitsuru Matsui and Robert J. Zuccherato, editors, *Selected Areas in Cryptography, 10th Annual International Workshop, SAC 2003, Ottawa, Canada, August 14-15, 2003, Revised Papers*, volume 3006 of *Lecture Notes in Computer Science*, pages 175–193. Springer, 2003.
8. Philip Hawkes, Michael Paddon, and Gregory G. Rose. On Corrective Patterns for the SHA-2 Family. *Cryptology eprint Archive*, August 2004. Available at <http://eprint.iacr.org/2004/207>.
9. Marko Hölbl, Christian Rechberger, and Tatjana Welzer. Finding Message Pairs Conforming to Simple SHA-256 Characteristics. In Stefan Lucks, Ahmad-Reza Sadeghi, and Christopher Wolf, editors, *Preliminary Proceeding Records of WEWoRC 2007 - Western European Workshop on Research in Cryptology, July 4-6, 2007, Bochum, Germany*, pages 21–25, 2007. Available at <http://www.hgi.rub.de/weworc07/PreliminaryConferenceRecord.pdf>.
10. Jeffrey S. Leon. A Probabilistic Algorithm for Computing Minimum Weights of Large Error-Correcting Codes. *IEEE Transactions on Information Theory*, 34(5):1354–1359, 1988.
11. Krystian Matusiewicz. *Analysis of Modern Dedicated Cryptographic Hash Functions*. PhD thesis, Macquarie University, August 2007. Available at <http://www.ics.mq.edu.au/~kmatus/Matusiewicz-PhDthesis.pdf>.
12. Krystian Matusiewicz, Josef Pieprzyk, Norbert Pramstaller, Christian Rechberger, and Vincent Rijmen. Analysis of simplified variants of SHA-256. In Christopher Wolf, Stefan Lucks, and Po-Wah Yau, editors, *WEWoRC 2005 - Western European Workshop on Research in Cryptology, July 5-7, 2005, Leuven, Belgium*, volume 74 of *LNI*, pages 123–134. GI, 2005.
13. Florian Mendel, Norbert Pramstaller, Christian Rechberger, and Vincent Rijmen. Analysis of Step-Reduced SHA-256. In Matthew J. B. Robshaw, editor, *Fast Software Encryption, 13th International Workshop, FSE 2006, Graz, Austria, March 15-17, 2006, Revised Selected Papers*, volume 4047 of *Lecture Notes in Computer Science*, pages 126–143. Springer, 2006.
14. Florian Mendel, Norbert Pramstaller, Christian Rechberger, and Vincent Rijmen. Analysis of Step-Reduced SHA-256. *Cryptology eprint Archive*, March 2008. Available at <http://eprint.iacr.org/2008/130>.
15. Ralph C. Merkle. One Way Hash Functions and DES. In Brassard [2], pages 428–446.
16. Ivica Nikolić and Alex Biryukov. Collisions for Step-Reduced SHA-256. In Kaisa Nyberg, editor, *Fast Software Encryption 2008*, volume Pre-proceedings version of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2008.
17. Norbert Pramstaller, Christian Rechberger, and Vincent Rijmen. Exploiting Coding Theory for Collision Attacks on SHA-1. In Nigel P. Smart, editor, *Cryptography and Coding, 10th IMA International Conference, Cirencester, UK, December 19-21, 2005, Proceedings*, volume 3796 of *Lecture Notes in Computer Science*, pages 78–95. Springer, 2005.
18. Vincent Rijmen and Elisabeth Oswald. Update on SHA-1. In Alfred Menezes, editor, *Topics in Cryptology - CT-RSA 2005, The Cryptographers' Track at the RSA Conference 2005, San Francisco, CA, USA, February 14-18, 2005, Proceedings*, volume 3376 of *Lecture Notes in Computer Science*, pages 58–71. Springer, 2005.
19. Somitra Kumar Sanadhya and Palash Sarkar. New Local Collisions for the SHA-2 Hash Family. In Kil-Hyun Nam and Gwangsoo Rhee, editors, *Information Security and Cryptology - ICISC 2007, 10th International Conference, Seoul, Korea, November 29-30, 2007, Proceedings*, volume 4817 of *Lecture Notes in Computer Science*, pages 193–205. Springer, 2007.

20. Secure Hash Standard. *Federal Information Processing Standard Publication 180-2*. U.S. Department of Commerce, National Institute of Standards and Technology(NIST), 2002. Available at <http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf>.
21. Jacques Stern. A Method for Finding Codewords of Small Weight. In Gérard D. Cohen and Jacques Wolfmann, editors, *Coding Theory and Applications, 3rd International Colloquium, Toulon, France, November 2-4, 1988, Proceedings*, volume 388 of *Lecture Notes in Computer Science*, pages 106–113. Springer, 1988.
22. Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, and Xiuyuan Yu. Cryptanalysis of the Hash Functions MD4 and RIPEMD. In Cramer [5], pages 1–18.
23. Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding Collisions in the Full SHA-1. In Victor Shoup, editor, *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, volume 3621 of *Lecture Notes in Computer Science*, pages 17–36. Springer, 2005.
24. Xiaoyun Wang and Hongbo Yu. How to Break MD5 and Other Hash Functions. In Cramer [5], pages 19–35.
25. Hirotaka Yoshida and Alex Biryukov. Analysis of a SHA-256 Variant. In Bart Preneel and Stafford E. Tavares, editors, *Selected Areas in Cryptography, 12th International Workshop, SAC 2005, Kingston, ON, Canada, August 11-12, 2005, Revised Selected Papers*, volume 3897 of *Lecture Notes in Computer Science*, pages 245–260. Springer, 2005.

## A 9-round Pseudo Collision for SHA-256

This is a 9 round pseudo-collision for SHA-256.

**Table 7.** IV and the messages for the 9 round pseudo collision. These two messages follow the differential path given in Table 1 starting from Step 3 till Step 11 (i.e. Step 3 in Table 1 is Step 1 for this message pair).

registers	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>g</i>	<i>h</i>		
IV	1b309331	1d07d226	190c04e9	0b413baf	7c11cf34	2d035d09	76e27935	0fb234e2		
$M_1$	0-8	5ce03f69	a36bfc0b	f99332c1	590db302	7c1cd4df	163c2f4b	2077b003	29ab9330	efb8306e
$M_2$	0-8	dce03f69	817ffe4b	bb162259	7b19b142	7c1cd4df	34282d0b	2077b003	29ab9330	6fb8306e

## B Using Neutral Bits to Search for the Colliding Pairs

Once a linear differential path is obtained, message modification or neutral bit technique are used to find message pairs following that characteristic. It is suggested in [13] that using neutral bits will bring the probability of finding the message pairs to close to 1. We have used message modification technique to find the message pairs, since the number of neutral bits was too low for random messages pairs that were generated during the search for the right pair. To see the usefulness of neutral bit technique for this 18 step case, we next detail the number of 1-neutral bits and their distribution in various words of the message pairs for the two colliding message pairs shown in Tables 3 and 4. This distribution is shown in Tables 8 and 9. The entry in the  $j^{th}$  column of the  $i^{th}$  row in the two tables denotes the number of 1-neutral bits for the  $j^{th}$  message word at step  $i$ . The entry ‘-’ means that the particular message word is not available at that step.

As can be expected, all the bits in the first 4 words in both the tables are 1-neutral. However, as soon as the fifth word is chosen, the number of 1-neutral bits comes down to about 30 and surprisingly almost all the bits of first 4 words do not remain 1-neutral anymore. If we assume that :

- (1) All the 1-neutral bits are also 2-neutral. And
- (2) The set of neutral bits is about  $1/8^{th}$  the size of the set of 2-neutral bits (as in SHA-0),

Then we are likely to get about 10 neutral bits at step 4. The probability of success of this step is  $2^{-11}$ . Similarly the next two steps are also likely to have about 10 neutral bits whereas the probability of success of these steps is  $2^{-14}$  and  $2^{-11}$  respectively. The number of neutral bits at a step and the order of the probability of that step is close enough and we may get a message pair which will follow the characteristic.

**Table 8.** Count of 1-neutral bits and their distribution for the message pair shown in Table 3. Steps 11 to 15 are not shown here since they do not put any restrictions on the messages and hence have all bits neutral.

Step i	0	1	2	3	4	5	6	7	8	9	10	Total
0	32	-	-	-	-	-	-	-	-	-	-	32
1	32	32	-	-	-	-	-	-	-	-	-	64
2	32	32	32	-	-	-	-	-	-	-	-	96
3	32	32	32	32	-	-	-	-	-	-	-	128
4	0	0	0	5	23	-	-	-	-	-	-	28
5	0	0	0	0	4	23	-	-	-	-	-	27
6	0	0	0	0	0	15	26	-	-	-	-	41
7	0	0	0	0	0	2	23	32	-	-	-	57
8	0	0	0	0	0	0	3	21	26	-	-	50
9	0	0	0	0	0	0	2	12	23	32	-	69
10	0	0	0	0	0	0	1	7	17	30	32	87

However, note that both the assumptions about the neutral bits made above are not likely to hold for SHA-256. In particular, the number of 2-neutral bits are going to be much less than the number of 1-neutral bits and the size of the neutral set is likely to be smaller than that in the case of SHA-0. Further, the set of 1-neutral bits shown are for the messages which are actually colliding at 18 step. When we started the search from random message pairs, in most of the cases we could not get even these many 1-neutral bits. This suggests that the application of neutral bits alone in SHA-256 may not help us finding colliding message pairs. A combination of message modification and neutral bit technique may be required to attack longer number of rounds in SHA-2 family.

**Table 9.** Count of 1-neutral bits and their distribution for the message pair shown in Table 4. Steps 11 to 15 are not shown here since they do not put any restrictions on the messages and hence have all bits neutral.

Step i	0	1	2	3	4	5	6	7	8	9	10	Total
0	32	-	-	-	-	-	-	-	-	-	-	32
1	32	32	-	-	-	-	-	-	-	-	-	64
2	32	32	32	-	-	-	-	-	-	-	-	96
3	32	32	32	32	-	-	-	-	-	-	-	128
4	0	0	0	5	26	-	-	-	-	-	-	31
5	0	0	0	0	8	23	-	-	-	-	-	31
6	0	0	0	0	0	13	26	-	-	-	-	39
7	0	0	0	0	0	3	22	32	-	-	-	57
8	0	0	0	0	0	1	6	19	26	-	-	52
9	0	0	0	0	0	1	3	15	25	32	-	76
10	0	0	0	0	0	1	1	9	19	31	32	93

## C Differential Paths for Reduced Round SHA-256

Differential paths for 19, 20, 21, 22 and 23 step SHA-256 are provided in Table 10 to Table 19. All the differential paths have been obtained by using two different local collisions, namely the GH and the  $SS_5$  local collision.









